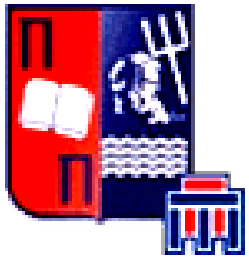


Πανεπιστήμιο Πειραιώς – Τμήμα Πληροφορικής

Πρόγραμμα Μεταπτυχιακών Σπουδών «Πληροφορικής»

Τίτλος Διατριβής	**Σχεδιασμός και Ανάπτυξη Τμήματος Διαδικτυακού Πληροφοριακού Συστήματος Ξενοδοχείων** **Design and Developmend part of a Web Based Hotel Information System**
Ονοματεπώνυμο Φοιτητή	**Παπάζογλου Πάνος**
Πατρώνυμο	**Ευάγγελος**
Αριθμός Μητρώου	**ΜΠΠΛ/11044**
Υπεύθυνος Καθηγητής	**Χρήστος Δουληγέρης, Καθηγητής**
Επιβλέπων Συνεργάτης	**Αγάπιος Αβραμίδης**

Ημερομηνία Παράδοσης	**Οκτώβριος 2015**

Τριμελής Εξεταστική Επιτροπή

(υπογραφή)	(υπογραφή)	(υπογραφή)
Χ. Δουληγέρης **Καθηγητής**	**Δ. Βέργαδος** **Επίκ. Καθηγητής**	**Παν. Κοτζανικολάου** **Επίκ. Καθηγητής**

Ευχαριστίες

Η εργασία αυτή εκπονήθηκε στο πλαίσιο της μεταπτυχιακής μου διατριβής του ΠΜΣ Πληροφορικής. Σχετίζεται με το μάθημα “Αντικειμενοστρεφής Προγραμματισμός – Τεχνολογίες Διαδικτύου” του καθηγητή Δουλιγέρη τον οποίο ευχαριστώ για την ανάθεση της, καθώς επίσης και τον κ. Αγάπιο Αβραμίδη για την βοήθεια σχετικά με τις τεχνολογικές κατευθύνσεις και γνώσεις που μου παρείχε.

Επιπλέον, θα ήθελα να ευχαριστήσω την γυναίκα μου για τις πληροφορίες που μου παρείχε σχετικά με τα ξενοδοχειακά αλλά και τα συστήματα κρατήσεων γενικότερα, καθώς απεδείχθησαν πολύτιμα για την δημιουργία του κεντρικού μοντέλου και όχι μόνο.

Τέλος και περισσότερο απ’ όλους θα ήθελα να ευχαριστήσω την κόρη μου Νεφέλη-Μαρία που ήταν ήσυχη και υπομονετική, αν και μόνο λίγων ημερών, μέχρι να καταφέρω επιτέλους να την ολοκληρώσω επιτυχώς!!

Περίληψη. Η παρούσα διπλωματική διατριβή παρουσιάζει, ένα διαδικτυακό πληροφοριακό σύστημα ξενοδοχείων. Πρόκειται για τον πυρήνα ενός συστήματος μέσω του οποίου μπορούν διάφοροι κάτοχοι καταλυμάτων (ξενοδοχείων, δωματίων κ.τ.λ.) να εγγραφούν στο σύστημα και να διαχειρίζονται την προβολή και την διαθεσιμότητα των δωματίων τους. Επιπλέον το σύστημα διαθέτει την υποδομή για την κατάλληλη επέκταση του, ώστε να είναι δυνατόν να λαμβάνουν χώρα online κρατήσεις, υποβοηθώντας τον κύκλο εργασιών και την πληρότητα των ξενοδοχείων τους.

Στην συνέχεια θα περιγραφούν διεξοδικά οι λειτουργικές του δυνατότητες, ενώ θα γίνει αναφορά στις τεχνολογίες που χρησιμοποιήθηκαν. Ενδεικτικά θα αναφέρουμε ότι χρησιμοποιήθηκε για την υλοποίηση του συστήματος η γλώσσα προγραμματισμού Java και τεχνολογίες J2EE. Ως βασικό framework επιλέχτηκε το Spring , για την διασύνδεση με την βάση σε επίπεδο ORM(Object Relational Mapping) χρησιμοποιήθηκαν JPA Annotations και ως provider (σύστημα υλοποίησης) το hibernate. Χρησιμοποιήθηκαν επίσης Apache Tiles, Ajax τεχνολογίες, Json , Rest, Bootstrap, JQuery κ.α.

Τέλος, ιδιαίτερη σημασία έχει δοθεί στην ασφάλεια του συστήματος με την χρήση του module Spring Security τόσο για την απομόνωση τον πληροφοριών για τα επιχειρηματικά στοιχεία του εκάστοτε ξενοδόχου, όσο επίσης και για τα διαφορετικά επίπεδα ρόλων και τις διαφορετικές δυνατότητες σε υπολειτουργίες που διατίθενται αντιστοίχως.

Abstract. This diploma thesis presents an online information system for hotels. This is the core of a system through which they hold various lodgings (hotels, rooms, etc.) to enter the system and manage the presentation and availability of their rooms. Additionally the system has the infrastructure for proper expansion, so that they can take place online bookings, helping turnover and the completeness of their hotels.

It will thoroughly be described the functional capabilities and will make reference to the technologies used. At the beginning, let's report, that Java (J2EE) was used as the implementation language, and Spring as the basic framework, also in order to interface with the base-level ORM (Object Relational Mapping) it was used JPA Annotations and hibernate as the provider (implementer). It was also used Apache Tiles, Ajax technologies, Json, Rest, Bootstrap, JQuery, etc.

Finally, special attention has been given to the security of the system using the Spring Security module, in order to isolate the information on risk components of the hotelier, as well as different levels of roles and different capabilities allocated respectively.

ΠΕΡΙΕΧΟΜΕΝΑ

1. ΕΙΚΟΝΕΣ

2. ΕΙΣΑΓΩΓΗ

2.1 ΑΝΤΙΚΕΙΜΕΝΟ ΚΑΙ ΔΟΜΗ ΤΗΣ ΜΕΤΑΠΤΥΧΙΑΚΗΣ ΔΙΑΤΡΙΒΗΣ

Το παρόν σύστημα παραδίδεται στο πλαίσιο της μεταπτυχιακής διατριβής με θέμα «Σχεδιασμός και Ανάπτυξη Τμήματος Διαδικτυακού Πληροφοριακού Συστήματος Ξενοδοχείων» και αφορά σε μια πλατφόρμα που εξυπηρετεί την ηλεκτρονική διαχείριση των καταλυμάτων από ξενοδόχους ή άλλους κατόχους επιχειρήσεων διαμονής.

Στο παρόν κείμενο θα αναλυθούν τα αρχιτεκτονικά στοιχεία του συστήματος, καθώς και οι επιλογές υλοποίησης. Γίνεται επίσης ανάλυση των υποστηρικτικών δομών που είναι απαραίτητες για τη λειτουργία του συστήματος και τέλος παρουσιάζεται ένας σύντομος οδηγός χρήσης της πλατφόρμας. Ειδικά για τους σκοπούς επίδειξης και διαμόρφωσης των χαρακτηριστικών οθονών της διαδικτυακής εφαρμογής έχουν χρησιμοποιηθεί δοκιμαστικά δεδομένα (dummy data).

2.2 ΣΥΝΟΠΤΙΚΗ ΠΕΡΙΓΡΑΦΗ ΤΟΥ ΣΥΣΤΗΜΑΤΟΣ

Η πλατφόρμα ηλεκτρονικής διαχείρισης καταλυμάτων απαιτεί τη διασύνδεση με μια υποδομή βάσης δεδομένων, η οποία θα διατηρεί όλα τα δεδομένα που απαιτούνται για την εύρυθμη λειτουργία του συστήματος. Τα στοιχεία αυτά αφορούν σε οτιδήποτε έχει να κάνει με τους χρήστες, τα ξενοδοχεία ή τα καταλύματα τους γενικότερα, τα βασικά τους στοιχεία, καθώς επίσης τους τύπους δωματίων, τις περιόδους, τις τιμές, και όποια άλλη πληροφορία απαιτείται προκειμένου να μπορεί να αποτυπωθεί ολοκληρωμένα μια επιχείρηση διαμονής.

Η μελέτη και ανάπτυξη του συστήματος πραγματοποιήθηκε στα ακόλουθα στάδια:

- Ορισμός απαραίτητων εννοιών και στοιχείων του συστήματος και των συσχετίσεων τους
- Επιλογές υλοποίησης
- Υλοποίηση βασικού συστήματος
- Ενσωμάτωση δοκιμαστικών δεδομένων και έλεγχος λειτουργίας

2.2.1 Ορισμός Απαραίτητων Εννοιών/Στοιχείων του Συστήματος και Συσχετίσεων τους

Η καθολική αποτύπωση ενός συστήματος που διαχειρίζεται επιχειρήσεις διαμονής, απαιτεί τον ορισμό των βασικών εννοιών που διέπουν αυτού του είδους των επιχειρήσεων αλλά και γενικότερα εκείνων που απαιτούνται για την εύρυθμη λειτουργία μιας τέτοιας πλατφόρμας. Απαραίτητη είναι επίσης η καταγραφή τόσο των δομικών στοιχείων που απαρτίζουν αυτές τις έννοιες, όσο και οι συσχετίσεις των εννοιών αυτών μεταξύ τους.

Η αποθήκευση των παραπάνω δεδομένων επιτυγχάνεται μέσω της διατήρησης σε μια δομή δεδομένων (βάση δεδομένων) του συνόλου των στοιχείων αυτών.

Σε αυτά τα απολύτως απαραίτητα στοιχεία, προστίθενται ένα σύνολο με βοηθητικά δεδομένα, που χρησιμοποιούνται ως προεπιλεγμένες τιμές και βοηθούν στην ομογενοποίηση κάποιον δεδομένων βάση κοινών παραδοχών (π.χ. βασικά λεκτικά για τύπους δωματίων – δίκλινο, τρίκλινο κ.τ.λ.).

2.2.2 Επιλογή Προγραμματιστικού Περιβάλλοντος και Συστήματος Υλοποίησης

Για την ανάπτυξη της πλατφόρμας επιλέχθηκε η χρήση της γλώσσας Java, η οποία καταρχάς είναι ανοικτού κώδικα και παρουσιάζει αρκετά μεγάλη λειτουργικότητα και δυναμική. Πρόκειται για μια γλώσσα προγραμματισμού που θεωρείται η πλέον ενδεδειγμένη για την υλοποίηση επαγγελματικών εφαρμογών μέσω διαδικτύου (Enterprise Web Applications). Επιπλέον πρόκειται για μια γλώσσα με έντονη αντικειμενοστρεφή (object oriented) χαρακτήρα, κάτι που βοηθά στην λογική του διαχωρισμού των εννοιών (separation on concerns) που στηρίζονται διάφορα design patterns.

Δεδομένης της μη ύπαρξης κάποιας εφαρμογής ανοιχτού λογισμικού που να καλύπτει τις ανάγκες της πλατφόρμας ή το μεγαλύτερο σύνολο αυτών όπως έχουν προδιαγραφεί, η υλοποίηση της υπηρεσίας δεν θα βασιστεί σε κάποιο έτοιμο σύστημα κρατήσεων. Εν΄ τούτης δεν θα γίνει εκ του μηδενός ή ακόμη και με απλή χρήση βιβλιοθηκών, αλλά θα βασιστούμε σε κάποιο πλαίσιο κώδικα (framework). Ο λόγος γι' αυτή την επιλογή είναι απλός.

Από την μια δεν θέλουμε να υλοποιήσουμε τα πάντα εκ' του μηδενός, δεδομένου ότι θα χανόταν αρκετός χρόνος για την παραγωγή κώδικα και θα έπρεπε από μόνοι μας να

χρησιμοποιήσουμε βέλτιστες πρακτικές (π.χ. design patterns), που σε ένα framework έχουν ήδη υλοποιηθεί και θεσπιστεί πλέον.

Από την άλλη δεν θέλουμε να βασίσουμε ένα σύστημα σε μια γενική εφαρμογή όπως κάποιο σύστημα διαχείρισης περιεχομένου ή ηλεκτρονικού καταστήματος ή ακόμη και σύστημα κρατήσεων, δεδομένου ότι η λογική της εφαρμογής μας θα έπρεπε να προσαρμοστεί στην λογική αυτού του συστήματος, ενώ δεν θα ήταν και λίγες οι περιπτώσεις που θα έπρεπε να γίνουν εκπτώσεις στις απαιτήσεις της πλατφόρμας μας λόγω μη εννοιολογικής συμβατότητας με αυτό ή μη κατάλληλης υποδομής. Επιπλέον ο χρόνος εκμάθησης ενός τέτοιου συστήματος, αλλά και η δημιουργία ειδικά γραμμένου γι' αυτό το σύστημα, και συνεπώς μη επαναχρησιμοποιήσιμου, κώδικα θα ήταν μερικοί από τους λόγους που κάτι τέτοιο δεν θα αποτελούσε στην περίπτωση αυτή την καλύτερη επιλογή.

2.2.3 Υλοποίηση Βασικής Υπηρεσίας

Η υλοποίηση της υπηρεσίας περιλαμβάνει δύο στάδια εφαρμογής. Το λειτουργικό κομμάτι και το κομμάτι παρουσίασης.

Αυτά τα δύο στάδια αναπτύχτηκαν ως διαφορετικές οντότητες και συγχωνεύτηκαν ώστε να δημιουργηθεί το τελικό αποτέλεσμα. Στο τμήμα της παρουσίασης επιλέχθηκε η χρήση του Bootstrap framework, με στόχο την δημιουργία μιας κοινής εικόνας της εφαρμογής (responsiveness), σε οποιαδήποτε πλατφόρμα επιλέγει ο χρήστης(pc/κινητά/tablet). Πέρα από τις πολλές διευκολύνσεις που παρουσιάζει το σύστημα αυτό (grid, icons, buttons, modals), πολύ σημαντικό ρόλο στην επιλογή της διαδραμάτισε και η πολύ καλή υποστήριξη και τεκμηρίωση από την ομάδα ανάπτυξης.

Το λειτουργικό κομμάτι υλοποιήθηκε βασισμένο σε ένα από τα πλέον διαδεδομένα frameworks το "Spring Framework" το οποίο είναι ένα λογισμικό ανοικτού κώδικα (ελεύθερο λογισμικό) που σκοπό έχει να διευκολύνει την ανάπτυξη J2EE λογισμικού σε μεγάλη έκταση.

2.2.4 Ενσωμάτωση Δοκιμαστικών Δεδομένων και Έλεγχος Λειτουργίας.

Για τον έλεγχο του συστήματος καταχωρήθηκαν δοκιμαστικά δεδομένα προκειμένου να μπορεί να γίνει ο έλεγχος λειτουργίας των εκάστοτε υποσυστημάτων. Βάση των αρχικών ρυθμίσεων είχε επιλεγεί, κατά τον χρόνο υλοποίησης, η βάση να διαγράφεται και να αναδομείται οδηγούμενη από μια διαδικασία αρχικοποίησης των δεδομένων που απαιτούνταν κατά την επανεκκίνηση του Spring Container. Αυτό ήταν χρήσιμο ώστε να υπάρχουν πάντα

τα δεδομένα που απαιτούνται με ένα σταθερό τρόπο χωρίς να πρέπει σε κάθε αλλαγή και επανεκκίνηση του εξυπηρετητή να εισάγονται εκ' νέου, αλλά ούτε και να χρειάζεται να τα διαγράφουμε. Το σύστημα μέσω κατάλληλης αλλαγής σε συγκεκριμένο αρχείο ρυθμίσεων δύναται να αλλάξει αυτή του την συμπεριφορά και να διατηρεί την υπάρχουσα βάση χωρίς να την καταστρέφει και να την αναδομεί εκ' νέου, όπως και θα ήταν το επιθυμητό σε συστήματα παραγωγής.

Όπως ήδη αναφερθήκαμε, η επιθυμία για την μελλοντική δυνατότητα χρήσης διαφορετικών τεχνολογιών δομών δεδομένων (βάσεων δεδομένων) μας οδήγησε στην επιλογή της χρήσης ORM μέσω JPA annotation και hibernate implementation, δίνοντας έτσι ένα επίπεδο αφαιρετικότητας, το οποίο είναι επιθυμητό για την σωστή συντήρηση και την επέκταση της εφαρμογής σε βάθος χρόνου. Ότι αφορά στις υποδομές για την βάση δεδομένων χρησιμοποιήθηκε η εγγενής δυνατότητα επικοινωνίας με mysql βάση δεδομένων, ως η πιο διαδεδομένη και χωρίς μεγάλες ανάγκες δέσμευσης πόρων του συστήματος.

Τέλος σε ότι αφορά στις διάφορες κλήσεις που απαιτήθηκαν μεταξύ client και server κυρίως κατά την εμφάνιση στα ημερολόγια του διαχειριστικού που λειτουργούν με ασύγχρονο τρόπο (ajax), έγινε χρήση τεχνολογιών REST με JSON, ενώ για την μετατροπή από αντικείμενα σε Json format και το ανάποδο (marshalling/unmarshalling) έγινε χρήση της βιβλιοθήκης "Jackson".

3. ΑΝΑΛΥΣΗ ΣΥΣΤΗΜΑΤΟΣ

3.1 ΒΑΣΙΚΕΣ ΟΝΤΟΤΗΤΕΣ

Ο ορισμός των απολύτως απαραιτήτων οντοτήτων/εννοιών, η καταγραφή των στοιχείων τους αλλά και η συσχέτιση αυτών των εννοιών μεταξύ τους, έγινε αφού ολοκληρώθηκε η μελέτη με βάση την επιστήμη των τουριστικών επιχειρήσεων που αφορά στα τουριστικά καταλύματα, τα διάφορα συστήματα κρατήσεων που είναι διαθέσιμα στο διαδίκτυο και κατόπιν επαφής με επαγγελματίες του χώρου.

Στην συνέχεια θα αναφερθούν οι βασικές έννοιες και τα απαραίτητα δομικά στοιχεία για κάθε μία από αυτές, καθώς επίσης και οι συσχετίσεις μεταξύ των εννοιών αυτών.

Εδώ να σημειώσουμε ότι ο εντοπισμός των ουσιαστικών και των ρημάτων στην εκάστοτε πρόταση σε ένα κείμενο προδιαγραφών μπορεί να βοηθήσει στην αποσαφήνιση των εννοιών/οντοτήτων (model layer) και των διαδικασιών (service layer) αντιστοίχως. Ενώ η εννοιολογική τους σχέση είναι αυτή που οδηγεί και τις σχέσεις στο σχήμα που θα δημιουργήσουμε.

3.1.1 Οντότητα «Ρόλος»

Αφορά στον ρόλο ή τους ρόλους που διαθέτει ένας χρήστης βάση μέσω του οποίου έχει την δυνατότητα να εκτελεί διάφορες διαδικασίες και να έχει πρόσβαση σε διαβαθμισμένες πληροφορίες.

Τα απαραίτητα δομικά στοιχεία που απαιτούνται είναι :

- ένα μοναδικό αναγνωριστικό (*INT* **role_id**)
- το όνομα του ρόλου (*VARCHAR(45)* **name**)

Το σύστημα διαθέτει εξ' αρχής τρείς βασικούς ρόλους που ορίζονται με τις παρακάτω τιμές :

- *ROLE_ADMIN*
- *ROLE_MANAGER*
- *ROLE_USER*

Η τιμή "ROLE_ADMIN" αφορά στον ρόλο του υπέρ διαχειριστή του συστήματος. Οι χρήστες με αυτό τον ρόλο έχουν την δυνατότητα της πλήρους διαχείρισης του συστήματος και διαθέτουν πρόσβαση στο σύνολο των δεδομένων που διατηρεί το σύστημα. Επιπλέον θα

έχουν μελλοντικές δυνατότητες όπως η ρύθμιση των αρχικοποιημένων τιμών όπως των ονομάτων των τύπων δωματίων, η διαγραφή ενός χρήστη και των ξενοδοχείων του, η προβολή γενικών στατιστικών κ.α.

Η τιμή "ROLE_MANAGE" αφορά στον ρόλο του διαχειριστή του καταλύματος (ξενοδοχείου). Οι χρήστες με αυτό τον ρόλο θα έχουν την δυνατότητα της πλήρους διαχείρισης των ξενοδοχείων που θα δημιουργήσουν στο σύστημα, ορίζοντας για παράδειγμα τύπους δωματίων, διαθεσιμότητες, περιόδους τιμές. Επιπλέον θα έχουν την δυνατότητα να βλέπουν τις κρατήσεις που έχουν γίνει στα ξενοδοχεία τους.

Τέλος η τιμή "ROLE_USER" αφορά στον πιστοποιημένο χρήστη που κάνει χρήση του συστήματος για την ολοκλήρωση μιας κράτησης. Ο χρήστης θα έχει μελλοντικά δυνατότητες όπως την προβολή των κρατήσεων που έχει κάνει.

Κάθε χρήστης που εισέρχεται στο σύστημα ανάλογα με τον ρόλο του (ή τον υψηλότερο ρόλο που διαθέτει) έχει διαφορετικό πίνακα ελέγχου προκειμένου να μπορεί να ολοκληρώσει τις διαδικασίες και τις δυνατότητες που αναφέραμε παραπάνω.

3.1.2 Οντότητα «Χρήστης»

Η έννοια του χρήστη είναι επίσης μια από τις πιο θεμελιώδεις του συστήματος. Αφορά μια ομάδα στοιχείων που θεωρούνται απαραίτητες για λόγους ταυτοποίησης του ατόμου που θα εγγραφεί για την διαχείριση των ξενοδοχείων του ή και των ατόμων που θα προχωρούν σε κρατήσεις.

Τα απαραίτητα δομικά στοιχεία που απαιτούνται είναι :

- ένα μοναδικό αναγνωριστικό (*INT* **user_id**)
- το όνομα χρήστη (*VARCHAR(45)* **username**)
- ο κωδικός (*VARCHAR(100)* **password**)
- η διεύθυνση του ηλεκτρονικού του ταχυδρομείου (*VARCHAR(100)* **email**)
- η κατάσταση του χρήστη στο σύστημα (*VARCHAR(45)* **status**)

Η έννοιες "Χρήστης" και "Ρόλος" συσχετίζονται όπως θα αναλυθεί στην αντίστοιχη ενότητα που αφορά στο διάγραμμα οντοτήτων συσχετίσεων (EER Diagram).

3.1.3 Οντότητα «Ξενοδοχείο»

Η έννοια αυτή αφορά στην πιο κεντρική έννοια όλου του συστήματος, βάση του οποίου συσχετίζονται όλες οι υπόλοιπες έννοιες. Τα απαραίτητα δομικά στοιχεία που απαιτούνται είναι :

- ένα μοναδικό αναγνωριστικό (*INT* **hotel_id**)
- το όνομα του ξενοδοχείου (*VARCHAR(100)* **name**)
- και το αναγνωριστικό του χρήστη κατόχου του ξενοδοχείου (*INT* **user_id**)

Να σημειώσουμε ότι ένας χρήστης μπορεί να διαθέτει κανένα, ένα ή και παραπάνω από ένα ξενοδοχεία.

3.1.4 Οντότητα «Βασικά Στοιχεία Ξενοδοχείου»

Πρόκειται για μια δευτερεύουσα έννοια, που συσχετίζεται με την βασική έννοια «ξενοδοχείο». Πιο συγκεκριμένα, κάθε ξενοδοχείο έχει κάποια βασικά στοιχεία που το περιγράφουν πέρα από τα δομικά. Αυτά μπορεί να αφορούν σε στοιχεία επικοινωνίας, το επίπεδο ποιότητας του όπως τα αστέρια κ.α.

Οι πληροφορίες αυτές αποτυπώνονται σε αυτή την εννοιολογική ενότητα δεδομένου ότι στο μέλλον ίσως να υπάρξουν και άλλα στοιχεία π.χ. εκτεταμένα (extended) στοιχεία τα οποία δεν θα ήταν φρόνιμο τόσο αυτά όσο και εκείνα των βασικών στοιχείων να βρίσκονταν στην έννοια των ξενοδοχείο στην βάση της γενικής αρχής του διαχωρισμού των εννοιών.

Τα απαραίτητα δομικά στοιχεία που απαιτούνται για την έννοια «Βασικά Στοιχεία Ξενοδοχείου» είναι τα εξής :

- ένα μοναδικό αναγνωριστικό συσχετιζόμενο με την έννοια ξενοδοχείο (*INT* **hotel_hotel_id**)
- ένα αναγνωριστικό συσχετιζόμενο με την έννοια αστέρια (*INT* **star_star_id**)
- την διεύθυνση του ξενοδοχείου (*VARCHAR(45)* **name**)

- ένα αναγνωριστικό συσχετιζόμενο με την έννοια πόλη (*INT* **city_ city_id**)
- ένα τηλέφωνο επικοινωνίας του ξενοδοχείου (*VARCHAR(45)* **name**)
- ένα κινητό τηλέφωνο επικοινωνίας του ξενοδοχείου (*VARCHAR(45)* **name**)

3.1.5 Οντότητα «Αστέρι»

Πρόκειται για μια βοηθητική έννοα, η οποία χρησιμοποιείται όπως είδαμε παραπάνω στην δευτερεύουσα έννοια «Βασικά στοιχεία Ξενοδοχείου» και αφορά το επίπεδο του ξενοδοχείου μετρημένο σε αστέρια. Πρόκειται περισσότερο για δεικτικό πεδίο (Look Up) το οποίο καταγράφει το σύνολο των τιμών γι' αυτή την έννοια π.χ. 1-Αστέρι, 2-Αστέρια, .. , 5-άστερο κ.τ.λ.

Τα απαραίτητα δομικά στοιχεία που απαιτούνται για την έννοια «Αστέρι» είναι τα εξής :

- ένα μοναδικό αναγνωριστικό (*INT* **star_id**)
- το όνομα της ποιότητας του αστεριού (*VARCHAR(45)* **name**)

3.1.6 Οντότητα «Πόλη»

Πρόκειται για μια βοηθητική έννοα, η οποία χρησιμοποιείται όπως είδαμε παραπάνω στην δευτερεύουσα έννοια «Βασικά στοιχεία Ξενοδοχείου» και αφορά την πόλη στην οποία εδρεύει το ξενοδοχείου καθώς επίσης και τον ταχυδρομικό κώδικα που τους αντιστοιχεί. Πρόκειται περισσότερο για δεικτικό πεδίο (Look Up) το οποίο καταγράφει το σύνολο των τιμών (πόλεων) γι' αυτή την έννοια π.χ. Αθήνα, Θεσσαλονίκη, Ρέθυμνο, Λαμία κ.τ.λ.

Τα απαραίτητα δομικά στοιχεία που απαιτούνται για την έννοια «Πόλη» είναι τα εξής :

- ένα μοναδικό αναγνωριστικό (*INT* **city_id**)
- το όνομα του ποιότητας του αστεριού (*VARCHAR(45)* **name**)
- ο ταχυδρομικός κώδικας της πόλης (*VARCHAR(45)* **name**)

3.1.7 Οντότητα «Τύπος Δωματίου»

Πρόκειται για μια πρωτεύοντα έννοια, που αφορά στους τύπους δωματίων που διαθέτει ένα ξενοδοχείο. Οι τιμές που συμπεριλαμβάνει αυτή η έννοια μπορεί να είναι είτε γενικευμένες του συστήματος (global=true), είτε προσωποποιημένες ανά κάτοχο ξενοδοχείου (global=false).

Τα απαραίτητα δομικά στοιχεία που απαιτούνται για την έννοια αυτή είναι τα εξής :

- ένα μοναδικό αναγνωριστικό (*INT* **roomtype_id**)
- το όνομα του τύπου δωματίου (*VARCHAR(45)* **name**)
- δυαδικό πεδίο για το αν ο συγκεκριμένος τύπος δωματίου είναι γενικευμένος ή προσωποποιημένος (*BOOLEAN* **global**)

3.1.8 Έννοια «Παροχές»

Πρόκειται για μια πρωτεύοντα έννοια, που αφορά στις παροχές τόσο του ξενοδοχείου όσο και του εκάστοτε τύπου δωματίου. Επιπλέον διατηρεί και την λογική που αναφέραμε και στους τύπους δωματίου, δηλαδή το κατά πόσο οι τιμές που συμπεριλαμβάνει αυτή η έννοια είναι γενικευμένες του συστήματος ή προσωποποιημένες ανά κάτοχο ξενοδοχείου.

Τα απαραίτητα δομικά στοιχεία που απαιτούνται για την έννοια αυτή είναι τα εξής :

- ένα μοναδικό αναγνωριστικό (*INT* **facility_id**)
- το όνομα της παροχής (*VARCHAR(45)* **name**)
- ο τύπος της παροχής, δηλαδή αν πρόκειται για παροχή του ξενοδοχείου γενικά π.χ. πισίνα, internet corner ή αν πρόκειται για παροχή ανά τύπο δωματίου (*VARCHAR(45)* **type**)
- δυαδικό πεδίο για το αν η συγκεκριμένη παροχή είναι γενικευμένη ή προσωποποιημένη (*BOOLEAN* **global**)

3.1.9 Οντότητα «Περίοδος»

Πρόκειται για μια πρωτεύοντα έννοια, που αφορά στις τιμολογιακές περιόδους ώστε να είναι δυνατή η διαφοροποίηση των χρεώσεων ανά περίοδο για το εκάστοτε τύπο δωματίου.

Τα απαραίτητα δομικά στοιχεία που απαιτούνται για την έννοια αυτή είναι τα εξής :

- ένα μοναδικό αναγνωριστικό (*INT* **period_id**)
- αρχική ημερομηνία της περιόδου (*DATE* **startDate**)
- τελική ημερομηνία της περιόδου (*DATE* **endDate**)
- ένα αναγνωριστικό συσχετιζόμενο με την έννοια ξενοδοχείο (*INT* **hotel_hotel_id**)

3.1.10 Οντότητα «Τιμή»

Πρόκειται για μια δευτερεύουσα έννοια, που αφορά στο ημερήσιο κόστος ενός τύπου δωματίου για μια δεδομένη περίοδο.

Τα απαραίτητα δομικά στοιχεία που απαιτούνται για την έννοια αυτή είναι τα εξής :

- ένα μοναδικό αναγνωριστικό (μέρος συνδυασμένου τριπλού κλειδιού) συσχετιζόμενο με την έννοια του ξενοδοχείου στην βάση της αλληλοσύνδεσης ξενοδοχείου-τύπου δωματίου που θα δούμε παρακάτω

  (*INT* **hotel_has_roomtype_hotel_hotel_id**)

- ένα μοναδικό αναγνωριστικό (μέρος συνδυασμένου τριπλού κλειδιού) συσχετιζόμενο με την έννοια του τύπου δωματίου στην βάση της αλληλοσύνδεσης ξενοδοχείου-τύπου δωματίου που θα δούμε παρακάτω

  (*INT* **hotel_has_roomtype_ roomtype _ roomtype _id**)

- ένα μοναδικό αναγνωριστικό (μέρος συνδυασμένου τριπλού κλειδιού) συσχετιζόμενο με την έννοια της περιόδου

  (*INT* **period_period _id**)

- το ημερήσιο κόστος δωματίου για δεδομένο τύπο δωματίου και περίοδο

  (*DECIMAL* **cost**)

3.1.11 Οντότητα «Διαθεσιμότητα»

Πρόκειται για μια δευτερεύουσα έννοια, που αφορά στην διαθεσιμότητα δωματίων για δεδομένο τύπου δωματίου ενός ξενοδοχείου και για δεδομένη ημερομηνία.

Τα απαραίτητα δομικά στοιχεία που απαιτούνται για την έννοια αυτή είναι τα εξής :

- ένα μοναδικό αναγνωριστικό (μέρος συνδυασμένου τριπλού κλειδιού) συσχετιζόμενο με την έννοια του ξενοδοχείου στην βάση της αλληλοσύνδεσης ξενοδοχείου-τύπου δωματίου που θα δούμε παρακάτω

  (*INT* **hotel_has_roomtype_hotel_hotel_id**)

- ένα μοναδικό αναγνωριστικό (μέρος συνδυασμένου τριπλού κλειδιού) συσχετιζόμενο με την έννοια του τύπου δωματίου στην βάση της αλληλοσύνδεσης ξενοδοχείου-τύπου δωματίου που θα δούμε παρακάτω

  (*INT* **hotel_has_roomtype_ roomtype _ roomtype _id**)

- η ημερομηνία βάση της οποίας εξετάζεται η διαθεσιμότητα (*DATE* **date**)

- το πλήθος των διαθέσιμων δωματίων (*INT* **quantity**)

3.1.12 Οντότητα «Κράτηση»

Πρόκειται για μια πρωτεύοντα έννοια, που αφορά στις κρατήσεις που λαμβάνουν χώρα στο σύστημα από τους χρήστες, για το εκάστοτε ξενοδοχείο.

Τα απαραίτητα δομικά στοιχεία που απαιτούνται για την έννοια αυτή είναι τα εξής :

- ένα μοναδικό αναγνωριστικό (*INT* **booking_id**)

- ένα αναγνωριστικό συσχετιζόμενο με την έννοια του χρήστη (*INT* **user_ user_id**)

- ένα μοναδικό αναγνωριστικό (μέρος συνδυασμένου τριπλού κλειδιού) συσχετιζόμενο με την έννοια του ξενοδοχείου στην βάση της αλληλοσύνδεσης ξενοδοχείου-τύπου δωματίου που θα δούμε παρακάτω

  (*INT* **hotel_has_roomtype_hotel_hotel_id**)

- ένα μοναδικό αναγνωριστικό (μέρος συνδυασμένου τριπλού κλειδιού) συσχετιζόμενο με την έννοια του τύπου δωματίου στην βάση της αλληλοσύνδεσης ξενοδοχείου-τύπου δωματίου που θα δούμε παρακάτω

  (*INT* **hotel_has_roomtype_ roomtype _ roomtype _id**)

- η ημερομηνία που έλαβε χώρα η κράτηση (*DATE* **bokingdate**)
- η αρχική ημερομηνία της διαμονής (check-in) (*DATE* **startDate**)
- η τελική ημερομηνία της διαμονής (check-out) (*DATE* **endDate**)
- το συνολικό κόστος της κράτησης (*DECIMAL* **totalcost**)

3.2 ΣΥΣΧΕΤΙΣΕΙΣ ΟΝΤΟΤΗΤΩΝ

3.2.1 Ανάλυση Συσχετίσεων Οντοτήτων

Στην συνέχεια θα προχωρήσουμε στην καταγραφή των συσχετίσεων μεταξύ των οντοτήτων όπως αυτές προέκυψαν από την ανάλυση που έγινε.

3.2.1.1 Χρήστης και Ρόλος

Ένας «χρήστης» διαθέτει ένα ή περισσότερους «ρόλους» (1->N) => (?-N)
Ένας «ρόλος» διατίθεται σε ένα ή περισσότερους «χρήστες» (1->N) => (N-?)
Συνεπώς έχουμε μια σχέση πολλά προς πολλά (N to N)
Έτσι έχουμε μια συσχέτιση οντοτήτων που αποτυπώνεται μέσω του ενδιάμεσου πίνακα (κανονικοποίηση) "user_has_role".

Τα απαραίτητα δομικά στοιχεία που απαιτούνται για την αποτύπωση αυτής της σχέσης στον πίνακα «user_has_role» είναι τα εξής :

- ένα αναγνωριστικό συσχετιζόμενο με την οντότητα «χρήστη»

  (*INT* **user_ user_id**), το οποίο είναι μέρος του κλειδιού (συνεχόμενη γραμμή)

- ένα αναγνωριστικό συσχετιζόμενο με την οντότητα «ρόλος»

  (*INT* **role_ role_id**), το οποίο είναι μέρος του κλειδιού (συνεχόμενη γραμμή)

3.2.1.2 Χρήστης και Ξενοδοχείο

Σε επίπεδο cardinality (max) :
Ένας «χρήστης» **ελέγχει** ένα ή περισσότερα «ξενοδοχεία» (1->N) => (?-N)
Ένα «ξενοδοχείο» **ελέγχεται** από έναν μόνο «χρήστη» (1->1) => (1-?)
Συνεπώς έχουμε μια σχέση ένα προς πολλά (1 το N).

Άρα από την μεριά του «ξενοδοχείου» το εξωτερικό σύμβολο της σχέσης είναι το « -< » (many),
ενώ από την μεριά του «χρήστη» το εξωτερικό σύμβολο της σχέσης είναι το " + " (one)
(crow's foot notation).

Σε επίπεδο cardinality (min):

Ένας «χρήστης» δύναται να μην **ελέγχει** κανένα «ξενοδοχείο»,

Αλλά ένα «ξενοδοχείο» δεν δύναται να μην **ελέγχεται** από κάποιο «χρήστη».

Άρα από την μεριά του «ξενοδοχείου» το εσωτερικό σύμβολο της σχέσης είναι το « o » (optional),
ενώ από την μεριά του «χρήστη» το εσωτερικό σύμβολο της σχέσης είναι το " + " (mandatory)
(crow's foot notation).

Το πρωτεύον σε αυτή την σχέση είναι ο «χρήστης» (**ελέγχει**- ενεργητική φωνή),

ενώ το δευτερεύον είναι το «ξενοδοχείο» (**ελέγχεται** - παθητική φωνή),

συνεπώς το ξένο κλειδί (FK) θα είναι στην μεριά του πίνακα «ξενοδοχείου»
και δεδομένου ότι δεν είναι μέρος του κλειδιού του, η σχέση μεταξύ τους αποτυπώνεται με μια διακεκομμένη γραμμή στο cross foot notation όπως μπορούμε να δούμε στην επόμενη ενότητα.

Το απαραίτητο δομικό στοιχείο που απαιτείται για την αποτύπωση αυτής της σχέσης στον πίνακα «hotel» είναι το εξής :

- ένα αναγνωριστικό συσχετιζόμενο με την οντότητα «χρήστη»

  (*INT* **user_user_id**) , το οποίο δεν είναι μέρος του κλειδιού του πίνακα.

3.2.1.3 Ξενοδοχείο και Βασικά Στοιχεία Ξενοδοχείου

Σε επίπεδο cardinality (max) :
Ένα «ξενοδοχείο» **διαθέτει** ένα και όχι παραπάνω σύνολα «βασικών στοιχείων ξενοδοχείου» (1->1) => (?-1)
Ένα σύνολο «βασικών στοιχείων ξενοδοχείου» **διατίθεται** σε ένα και όχι παραπάνω «ξενοδοχεία» (1->1) => (1-?)
Συνεπώς έχουμε μια σχέση ένα προς ένα (1 το 1).

Άρα από την μεριά των «βασικών στοιχείων ξενοδοχείου» το εξωτερικό σύμβολο της σχέσης είναι το « + » (one),
ενώ και από την μεριά του «ξενοδοχείου» το εξωτερικό σύμβολο της σχέσης είναι επίσης το " + " (one) (crow's foot notation).

Σε επίπεδο cardinality (min):

Ένα «ξενοδοχείο» δεν μπορεί να μην διαθέτει ένα σύνολο από «βασικά στοιχεία ξενοδοχείου» ,

αλλά ούτε ένα σύνολο «βασικών στοιχείων ξενοδοχείου» μπορεί να μην διατίθεται σε κάποιο ξενοδοχεία.

Άρα από την μεριά των «βασικών στοιχείων ξενοδοχείου» το εσωτερικό σύμβολο της σχέσης είναι το « + » (mandatory),
ενώ και από την μεριά του «ξενοδοχείου» το εσωτερικό σύμβολο της σχέσης είναι επίσης το " + " (mandatory) (crow's foot notation).

Το πρωτεύον σε αυτή την σχέση είναι το «ξενοδοχείο» (**διαθέτει** - ενεργητική φωνή),

ενώ το δευτερεύον είναι το σύνολο των «βασικών στοιχείων ξενοδοχείων» (**διατίθεται** - παθητική φωνή),

συνεπώς το ξένο κλειδί (FK) θα είναι στην μεριά του πίνακα «Βασικά Στοιχεία Ξενοδοχείου» και δεδομένου ότι είναι μέρος του κλειδιού του, η σχέση μεταξύ τους αποτυπώνεται με μια συνεχόμενη γραμμή στο cross foot notation όπως μπορούμε να δούμε στην επόμενη ενότητα.

Το απαραίτητο δομικό στοιχείο που απαιτείται για την αποτύπωση αυτής της σχέσης στον πίνακα «hotel_basic_info» είναι το εξής :

- ένα αναγνωριστικό συσχετιζόμενο με την οντότητα «ξενοδοχείο»

  (*INT* **hotel_hotel_id**) , το οποίο είναι μέρος του κλειδιού του πίνακα.

3.2.1.4 Βασικά Στοιχεία Ξενοδοχείου και Αστέρι

Σε επίπεδο cardinality (max) :
Ένα «αστέρι» **χαρακτηρίζει** ένα ή περισσότερα σύνολα «βασικών στοιχείων ξενοδοχείου»

(1->N)=>(?-N)
Ένα σύνολο «βασικών στοιχείων ξενοδοχείου» **χαρακτηρίζεται** από έναν μόνο «αστέρι»

(1->1) => (1-?)
Συνεπώς έχουμε μια σχέση ένα προς πολλά (1 to N).

Άρα από την μεριά του συνόλου «βασικά στοιχεία ξενοδοχείου» το εξωτερικό σύμβολο της σχέσης είναι το « -< » (many),

ενώ από την μεριά του «αστεριού» το εξωτερικό σύμβολο της σχέσης είναι το « + » (one) (crow's foot notation).

Σε επίπεδο cardinality (min):

Ένας «αστέρι» δύναται να μην **χαρακτηρίζει** κανένα σύνολο «βασικών στοιχείων ξενοδοχείου»,

αλλά ένα σύνολο «βασικών στοιχείων ξενοδοχείου» πρέπει να **χαρακτηρίζεται** από ένα «αστέρι».

Άρα από την μεριά του συνόλου «βασικά στοιχεία ξενοδοχείου» το εσωτερικό σύμβολο της σχέσης είναι το « o » (optional),

ενώ από την μεριά του «αστεριού» το εσωτερικό σύμβολο της σχέσης είναι το « + » (mandatory) (crow's foot notation).

Το πρωτεύον σε αυτή την σχέση είναι το «αστέρι» (**χαρακτηρίζει**- ενεργητική φωνή),

ενώ το δευτερεύον είναι το «βασικά στοιχεία ξενοδοχείου» (**χαρακτηρίζεται** - παθητική φωνή), συνεπώς το ξένο κλειδί (FK) θα είναι στην μεριά του πίνακα «βασικά στοιχεία ξενοδοχείου» και δεδομένου ότι δεν είναι μέρος του κλειδιού του, η σχέση μεταξύ τους αποτυπώνεται με μια διακεκομμένη γραμμή στο cross foot notation όπως μπορούμε να δούμε στην επόμενη ενότητα.

Το απαραίτητο δομικό στοιχείο που απαιτείται για την αποτύπωση αυτής της σχέσης στον πίνακα «hotel_basic_info» είναι το εξής :

- ένα αναγνωριστικό συσχετιζόμενο με την οντότητα «αστέρι»

  (*INT* **star_star_id**), το οποίο δεν είναι μέρος του κλειδιού του πίνακα.

3.2.1.5 Βασικά Στοιχεία Ξενοδοχείου και Πόλη

Σε επίπεδο cardinality (max) :
Μια «πόλη» **χαρακτηρίζει** ένα ή περισσότερα σύνολα «βασικών στοιχείων ξενοδοχείου»

(1->N)=>(?-N)
Ένα σύνολο «βασικών στοιχείων ξενοδοχείου» **χαρακτηρίζεται** από μία μόνο «πόλη»

(1->1) => (1-?)
Συνεπώς έχουμε μια σχέση ένα προς πολλά (1 το N).

Άρα από την μεριά του συνόλου «βασικά στοιχεία ξενοδοχείου» το εξωτερικό σύμβολο της σχέσης είναι το « -< » (many),

ενώ από την μεριά της «πόλης» το εξωτερικό σύμβολο της σχέσης είναι το « + » (one) (crow's foot notation).

Σε επίπεδο cardinality (min):

Μια «πόλη» δύναται να μην **χαρακτηρίζει** κανένα σύνολο «βασικών στοιχείων ξενοδοχείου»,

αλλά ένα σύνολο «βασικών στοιχείων ξενοδοχείου» πρέπει να **χαρακτηρίζεται** από μια «πόλη».

Άρα από την μεριά του συνόλου «βασικά στοιχεία ξενοδοχείου» το εσωτερικό σύμβολο της σχέσης είναι το « o » (optional),

ενώ από την μεριά της «πόλης» το εσωτερικό σύμβολο της σχέσης είναι το « + » (mandatory) (crow's foot notation).

Το πρωτεύον σε αυτή την σχέση είναι η «πόλη» (**χαρακτηρίζει**- ενεργητική φωνή),

ενώ το δευτερεύον είναι το «βασικά στοιχεία ξενοδοχείου» (**χαρακτηρίζεται** - παθητική φωνή), συνεπώς το ξένο κλειδί (FK) θα είναι στην μεριά του πίνακα «βασικά στοιχεία ξενοδοχείου» και δεδομένου ότι δεν είναι μέρος του κλειδιού του, η σχέση μεταξύ τους αποτυπώνεται με μια διακεκομμένη γραμμή στο cross foot notation όπως μπορούμε να δούμε στην επόμενη ενότητα.

Το απαραίτητο δομικό στοιχείο που απαιτείται για την αποτύπωση αυτής της σχέσης στον πίνακα «hotel_basic_info» είναι το εξής :

- ένα αναγνωριστικό συσχετιζόμενο με την οντότητα «πόλη»

  (*INT* **city_city_id**), το οποίο δεν είναι μέρος του κλειδιού του πίνακα.

3.2.1.6 Ξενοδοχείο και Παροχή

Ένα «ξενοδοχείο» διαθέτει μία ή περισσότερες «παροχές» (1->N) => (?-N)
Μια «παροχή» διατίθεται σε ένα ή περισσότερα «ξενοδοχεία» (1->N) => (N-?)
Συνεπώς έχουμε μια σχέση πολλά προς πολλά (N to N)
Έτσι έχουμε μια συσχέτιση οντοτήτων που αποτυπώνεται μέσω του ενδιάμεσου πίνακα (κανονικοποίηση) "hotel_has_facility".

Τα απαραίτητα δομικά στοιχεία που απαιτούνται για την αποτύπωση αυτής της σχέσης στον πίνακα «hotel_has_facility» είναι τα εξής :

- ένα αναγνωριστικό συσχετιζόμενο με την οντότητα «ξενοδοχείο»

  (*INT* **hotel_ hotel_id**) , το οποίο είναι μέρος του κλειδιού (συνεχόμενη γραμμή)

- ένα αναγνωριστικό συσχετιζόμενο με την οντότητα «παροχή»

  (*INT* **role_ role_id**) , το οποίο είναι μέρος του κλειδιού (συνεχόμενη γραμμή)

3.2.1.7 Ξενοδοχείο και Τύπος Δωματίου

Ένα «ξενοδοχείο» διαθέτει ένα ή περισσότερους «τύπους δωματίων» (1->N) => (?-N)
Ένας «τύπος δωματίων» διατίθεται σε ένα ή περισσότερα «ξενοδοχεία» (1->N) => (N-?)
Συνεπώς έχουμε μια σχέση πολλά προς πολλά (N to N)
Έτσι έχουμε μια συσχέτιση οντοτήτων που αποτυπώνεται μέσω του ενδιάμεσου πίνακα (κανονικοποίηση) "hotel_has_roomtype".

Τα απαραίτητα δομικά στοιχεία που απαιτούνται για την αποτύπωση αυτής της σχέσης στον πίνακα «hotel_has_roomtype» είναι τα εξής :

- ένα αναγνωριστικό συσχετιζόμενο με την οντότητα «ξενοδοχείο»

  (*INT* **hotel_ hotel_id**) , το οποίο είναι μέρος του κλειδιού (συνεχόμενη γραμμή)

- ένα αναγνωριστικό συσχετιζόμενο με την οντότητα «τύπος δωματίου»

  (*INT* **roomtype_ roomtype_id**) , το οποίο είναι μέρος του κλειδιού (συνεχόμενη γραμμή)

3.2.1.8 Τύπος Δωματίου Ξενοδοχείου και Παροχή

Ένα «τύπος δωματίου ξενοδοχείου» διαθέτει μία ή περισσότερες «παροχές» (1->N) => (?-N)
Μια «παροχή» διατίθεται σε ένα ή περισσότερους «τύπους δωματίου ξενοδοχείου»
(1->N) => (N-?)
Συνεπώς έχουμε μια σχέση πολλά προς πολλά (N to N)
Έτσι έχουμε μια συσχέτιση οντοτήτων που αποτυπώνεται μέσω του ενδιάμεσου πίνακα (κανονικοποίηση) "hotel_has_roomtype_has_facility".

Τα απαραίτητα δομικά στοιχεία που απαιτούνται για την αποτύπωση αυτής της σχέσης στον πίνακα «hotel_has_roomtype_has_facility» είναι τα εξής :

- ένα αναγνωριστικό συσχετιζόμενο με την οντότητα «ξενοδοχείο», που είναι μέρος του κλειδιού του τύπου δωματίου ξενοδοχείου

  (*INT* **hotel_has_roomtype_hotel_hotel_id**), το οποίο είναι μέρος του κλειδιού (συνεχόμενη γραμμή)

- ένα αναγνωριστικό συσχετιζόμενο με την οντότητα «τύπος δωματίου», που είναι μέρος του κλειδιού του τύπου δωματίου ξενοδοχείου

  (*INT* **hotel_has_roomtype_roomtype_roomtype_id**), το οποίο είναι μέρος του κλειδιού (συνεχόμενη γραμμή)

- ένα αναγνωριστικό συσχετιζόμενο με την οντότητα «παροχή»

  (*INT* **facility_ facility_id**) , το οποίο είναι μέρος του κλειδιού (συνεχόμενη γραμμή)

3.2.1.9 Ξενοδοχείο και Περίοδος

Σε επίπεδο cardinality (max) :
Ένα «ξενοδοχείο» **ορίζει** μια ή περισσότερες «περιόδους» (1->N) => (?-N)
Μια «περίοδος» **ορίζεται** από έναν μόνο «ξενοδοχείο» (1->1) => (1-?)
Συνεπώς έχουμε μια σχέση ένα προς πολλά (1 to N).

Άρα από την μεριά της «περιόδου» το εξωτερικό σύμβολο της σχέσης είναι το « -< » (many), ενώ από την μεριά του «ξενοδοχείου» το εξωτερικό σύμβολο της σχέσης είναι το " + " (one) (crow's foot notation).

Σε επίπεδο cardinality (min):

Ένας «ξενοδοχείο» δύναται να μην **ορίζεται** καμία «περίοδο»,

Αλλά μια «περίοδος» δεν δύναται να μην **ορίζει** από κάποιο «ξενοδοχείο».

Άρα από την μεριά της «περιόδου» το εσωτερικό σύμβολο της σχέσης είναι το « o » (optional),
ενώ από την μεριά του «ξενοδοχείου» το εσωτερικό σύμβολο της σχέσης είναι το " + " (mandatory)
(crow's foot notation).

Το πρωτεύον σε αυτή την σχέση είναι ο «ξενοδοχείο» (**κοστολογεί**- ενεργητική φωνή),

ενώ το δευτερεύον είναι η «περίοδος» (**κοστολογείται**- παθητική φωνή),

συνεπώς το ξένο κλειδί (FK) θα είναι στην μεριά του πίνακα «περίοδος»
και δεδομένου ότι δεν είναι μέρος του κλειδιού του, η σχέση μεταξύ τους αποτυπώνεται με μια διακεκομμένη γραμμή στο cross foot notation όπως μπορούμε να δούμε στην επόμενη ενότητα.

Το απαραίτητο δομικό στοιχείο που απαιτείται για την αποτύπωση αυτής της σχέσης στον πίνακα «period» είναι το εξής :

- ένα αναγνωριστικό συσχετιζόμενο με την οντότητα «ξενοδοχείο»

  (*INT* **hotel_hotel_id**) , το οποίο δεν είναι μέρος του κλειδιού του πίνακα.

3.2.1.10 Κόστος - Περίοδος και Τύπος Δωματίου Ξενοδοχείου

Σε επίπεδο cardinality (max) :
Μια «περίοδος» **κοστολογεί** ένα ή περισσότερους «τύπους δωματίου ξενοδοχείου»
(1->N) => (?-N)

Ένας «τύπος δωματίου ξενοδοχείου» **κοστολογείται** από μία ή περισσότερες «περιόδους»
(1->N) => (N-?)
Συνεπώς έχουμε μια σχέση πολλά προς πολλά (N to N)
Έτσι έχουμε μια συσχέτιση οντοτήτων που αποτυπώνεται μέσω του ενδιάμεσου πίνακα (κανονικοποίηση) "price ".

Τα απαραίτητα δομικά στοιχεία που απαιτούνται για την αποτύπωση αυτής της σχέσης στον πίνακα «price» είναι τα εξής :

- ένα αναγνωριστικό συσχετιζόμενο με την οντότητα «ξενοδοχείο», που είναι μέρος του κλειδιού του τύπου δωματίου ξενοδοχείου

  (*INT* **hotel_has_roomtype_hotel_hotel_id**), το οποίο είναι μέρος του κλειδιού (συνεχόμενη γραμμή)

- ένα αναγνωριστικό συσχετιζόμενο με την οντότητα «τύπος δωματίου», που είναι μέρος του κλειδιού του τύπου δωματίου ξενοδοχείου

  (*INT* **hotel_has_roomtype_roomtype_roomtype_id**), το οποίο είναι μέρος του κλειδιού (συνεχόμενη γραμμή)

- ένα αναγνωριστικό συσχετιζόμενο με την οντότητα «περίοδο»

  (*INT* **period_period_id**) , το οποίο είναι μέρος του κλειδιού (συνεχόμενη γραμμή)

3.2.1.11 Τύπος Δωματίου Ξενοδοχείου και Διαθεσιμότητα

Σε επίπεδο cardinality (max) :
Ένας «τύπος δωματίου ξενοδοχείου» **διαθέτει** μια ή περισσότερες μέρες «διαθεσιμότητας»
(1->N) => (?-N)
Μία μέρα «διαθεσιμότητας» **διατίθεται** σε έναν μόνο «τύπο δωματίου ξενοδοχείου»
(1->1) => (1-?)
Συνεπώς έχουμε μια σχέση ένα προς πολλά (1 το N).

Άρα από την μεριά της μέρας της «διαθεσιμότητας» το εξωτερικό σύμβολο της σχέσης είναι το « -< » (many),
ενώ από την μεριά του «τύπου δωματίου ξενοδοχείου» το εξωτερικό σύμβολο της σχέσης είναι το " + " (one) (crow's foot notation).

Σε επίπεδο cardinality (min):

Ένας «Τύπος Δωματίου Ξενοδοχείου» μπορεί να μην **διαθέτει** καμία μέρα «διαθεσιμότητας»,

Αλλά μια μέρα «διαθεσιμότητας» δεν δύναται να μην **διατίθεται σε κάποιο** «τύπο δωματίου ξενοδοχείου».

Άρα από την μεριά της μέρας «διαθεσιμότητας» το εσωτερικό σύμβολο της σχέσης είναι το « o » (optional),
ενώ από την μεριά του «τύπου δωματίου ξενοδοχείου» το εσωτερικό σύμβολο της σχέσης είναι το " + " (mandatory) (crow's foot notation).

Το πρωτεύον σε αυτή την σχέση είναι ο «Τύπος Δωματίου Ξενοδοχείου» (**διαθέτει**- ενεργητική φωνή),

ενώ το δευτερεύον είναι η «διαθεσιμότητα» (**διατίθεται** - παθητική φωνή),

συνεπώς το ξένο κλειδί (FK) θα είναι στην μεριά του πίνακα «διαθεσιμότητα»
και δεδομένου ότι είναι μέρος του κλειδιού του, η σχέση μεταξύ τους αποτυπώνεται με μια συνεχόμενη γραμμή στο cross foot notation όπως μπορούμε να δούμε στην επόμενη ενότητα.

Τα απαραίτητα δομικά στοιχεία που απαιτούνται για την αποτύπωση αυτής της σχέσης στον πίνακα «availability» είναι τα εξής :

- ένα αναγνωριστικό συσχετιζόμενο με την οντότητα «ξενοδοχείο», που είναι μέρος του κλειδιού του τύπου δωματίου ξενοδοχείου

  (*INT* **hotel_has_roomtype_hotel_hotel_id**), το οποίο είναι μέρος του κλειδιού (συνεχόμενη γραμμή)

- ένα αναγνωριστικό συσχετιζόμενο με την οντότητα «τύπος δωματίου», που είναι μέρος του κλειδιού του τύπου δωματίου ξενοδοχείου

  (*INT* **hotel_has_roomtype_roomtype_roomtype_id**), το οποίο είναι μέρος του κλειδιού (συνεχόμενη γραμμή)

3.2.1.12 Τύπος Δωματίου Ξενοδοχείου και Κράτηση

Σε επίπεδο cardinality (max) :
Ένας «τύπος δωματίου ξενοδοχείου» **διαθέτει** μια ή περισσότερες «κρατήσεις»
(1->N) => (?-N)
Μία «κράτηση» **διατίθεται** σε έναν μόνο «τύπο δωματίου ξενοδοχείου»
(1->1) => (1-?)
Συνεπώς έχουμε μια σχέση ένα προς πολλά (1 το N).

Άρα από την μεριά της μέρας της «κράτησης» το εξωτερικό σύμβολο της σχέσης είναι
το « -< » (many),
ενώ από την μεριά του «τύπου δωματίου ξενοδοχείου» το εξωτερικό σύμβολο της σχέσης είναι το " + " (one) (crow's foot notation).

Σε επίπεδο cardinality (min):

Ένας «Τύπος Δωματίου Ξενοδοχείου» μπορεί να μην **διαθέτει** καμία «κράτηση»,

Αλλά μια «κράτηση» δεν δύναται να μην **διατίθεται σε κάποιο** «τύπο δωματίου ξενοδοχείου».

Άρα από την μεριά της «κράτησης» το εσωτερικό σύμβολο της σχέσης είναι το « o » (optional),
ενώ από την μεριά του «τύπου δωματίου ξενοδοχείου» το εσωτερικό σύμβολο της σχέσης είναι το " + " (mandatory) (crow's foot notation).

Το πρωτεύον σε αυτή την σχέση είναι ο «Τύπος Δωματίου Ξενοδοχείου»
(**διαθέτει**- ενεργητική φωνή),

ενώ το δευτερεύον είναι η «κράτηση» (**διατίθεται** - παθητική φωνή),

συνεπώς το ξένο κλειδί (FK) θα είναι στην μεριά του πίνακα «κράτηση»
και δεδομένου ότι δεν είναι μέρος του κλειδιού του, η σχέση μεταξύ τους αποτυπώνεται με μια διακεκομμένη γραμμή στο cross foot notation όπως μπορούμε να δούμε στην επόμενη ενότητα.

Τα απαραίτητα δομικά στοιχεία που απαιτούνται για την αποτύπωση αυτής της σχέσης στον πίνακα «availability» είναι τα εξής :

- ένα αναγνωριστικό συσχετιζόμενο με την οντότητα «ξενοδοχείο», που είναι μέρος του κλειδιού του τύπου δωματίου ξενοδοχείου

  (*INT* **hotel_has_roomtype_hotel_hotel_id**), το οποίο είναι μέρος του κλειδιού (διακεκομμένη γραμμή)

- ένα αναγνωριστικό συσχετιζόμενο με την οντότητα «τύπος δωματίου», που είναι μέρος του κλειδιού του τύπου δωματίου ξενοδοχείου

  (*INT* **hotel_has_roomtype_roomtype_roomtype_id**), το οποίο είναι μέρος του κλειδιού (διακεκομμένη γραμμή)

3.2.1.13 Χρήστης και Κράτηση

Σε επίπεδο cardinality (max) :
Ένας «χρήστης» **διαθέτει** μια ή περισσότερες «κρατήσεις»
(1->N) => (?-N)
Μία «κράτηση» **διατίθεται** από έναν μόνο «χρήστη»
(1->1) => (1-?)
Συνεπώς έχουμε μια σχέση ένα προς πολλά (1 το N).

Άρα από την μεριά της μέρας της «κράτησης» το εξωτερικό σύμβολο της σχέσης είναι το « -< » (many),
ενώ από την μεριά του «χρήστη» το εξωτερικό σύμβολο της σχέσης είναι το " + " (one) (crow's foot notation).

Σε επίπεδο cardinality (min):

Ένας «χρήστης» μπορεί να μην **διαθέτει** καμία «κράτηση»,

Αλλά μια «κράτηση» δεν δύναται να μην **διατίθεται από κάποιο** «χρήστη».
Άρα από την μεριά της «κράτησης» το εσωτερικό σύμβολο της σχέσης είναι το « o » (optional) ,
ενώ από την μεριά του «χρήστη» το εσωτερικό σύμβολο της σχέσης είναι το " + " (mandatory) (crow's foot notation).

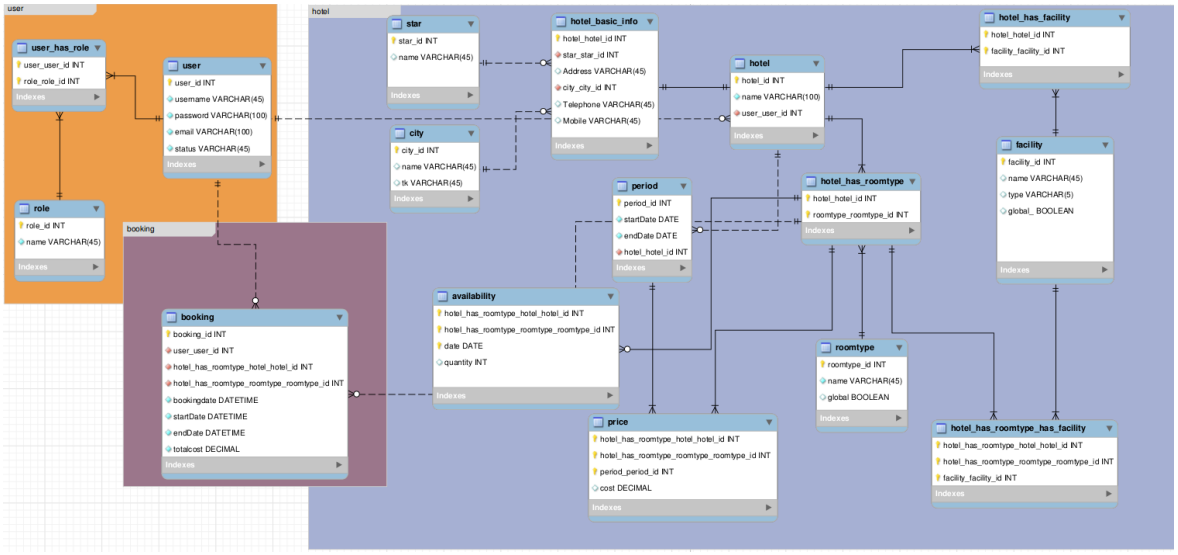
Το πρωτεύον σε αυτή την σχέση είναι ο «χρήστης»
(**διαθέτει**- ενεργητική φωνή),
ενώ το δευτερεύον είναι η «κράτηση» (**διατίθεται** - παθητική φωνή),
συνεπώς το ξένο κλειδί (FK) θα είναι στην μεριά του πίνακα «κράτηση»
και δεδομένου ότι δεν είναι μέρος του κλειδιού του, η σχέση μεταξύ τους αποτυπώνεται με μια διακεκομμένη γραμμή στο cross foot notation όπως μπορούμε να δούμε στην επόμενη ενότητα.
Το απαραίτητο δομικό στοιχείο που απαιτείται για την αποτύπωση αυτής της σχέσης στον πίνακα «κράτηση» είναι το εξής :

- ένα αναγνωριστικό συσχετιζόμενο με την οντότητα «χρήστη»
  (*INT* **user_user_id**) , το οποίο δεν είναι μέρος του κλειδιού του πίνακα.

3.2.2 Διάγραμμα Οντοτήτων-Συσχετίσεων (EER Diagram)

Παρακάτω βλέπουμε το διάγραμμα "Οντοτήτων-Συσχετίσεων" όπως αυτό αποτυπώθηκε βασιζόμενο στην ανάλυση που προηγήθηκε.

Το εργαλείο που χρησιμοποιήθηκε για την παραγωγή του είναι το **MySQL Workbench** και για την αποτύπωση των συσχετισμών (relationship notation) έγινε χρήση του Crow's Foot notation.



Εικόνα 1 – Διάγραμμα Οντοτήτων (EER-Diagram)

4. ΥΛΟΠΟΙΗΣΗ ΥΠΗΡΕΣΙΑΣ – ΑΝΤΙΚΕΙΜΕΝΟ

Ο ορισμός των χρηστών της υπηρεσίας , προηγείται της υλοποίησης, καθώς με βάση αυτόν ορίζονται τα επίπεδα πρόσβασης χρηστών. Μετά τον ορισμό των επιπέδων αυτών (ρόλοι) αναλύονται οι υπηρεσίες που διατίθενται.

Έτσι σε δεύτερο στάδιο ορίστηκε και το σύνολο των επιτρεπτών ενεργειών των χρηστών, κάτι που αποτέλεσε και το βασικό οδηγό προδιαγραφών της ίδιας της εφαρμογής.

4.1 ΧΡΗΣΤΕΣ ΤΗΣ ΥΠΗΡΕΣΙΑΣ

Η πλατφόρμα απευθύνεται στο σύνολο των επιχειρηματιών διαμονής όπως ξενοδόχους ή κατόχους καταλυμάτων ο οποίοι θέλουν να δημοσιεύσουν τα ξενοδοχεία/καταλύματα τους και τις παροχές τους μέσω διαδικτύου, με ένα εύχρηστο και γρήγορο τρόπο.

Στην παρούσα φάση ο σχεδιασμός της υπηρεσίας ορίζει πως αποδέκτες είναι όσοι επιθυμούν να εντοπίσουν το κατάλληλο ξενοδοχείο ή κατάλυμα για την διαμονή τους σε ένα προορισμό. Αυτοί οι αποδέκτες δεν δύνανται να χρησιμοποιήσουν την υπηρεσία αν δεν έχουν κάνει την αρχική τους εγγραφή ή δεν έχουν προβεί σε κάποια κράτηση που θα τους οδηγήσει σε αυτόματη δημιουργία λογαριασμού. Η διαδικασία αυτή είναι μέρος της επόμενης φάσης, όπως και η δυνατότητα αποδέκτες της πλατφόρμας να είναι άλλα B2B συστήματα μέσω web services.

Τέλος, το υπέρ διαχειριστικό τμήμα προφανώς είναι μη προσβάσιμο από οποιονδήποτε πέρα από την κλειστή ομάδα των διαχειριστών με τον αντίστοιχο ρόλο. Σε αυτό θα δίνονται δυνατότητες όπως η συνολική αποτύπωση των κρατήσεων για το σύνολο των επιχειρήσεων διαμονής ή ανά επιχειρηματία, στατιστικά επισκεψημότητας, ορισμός προγραμμάτων φιλοξενίας στην πλατφόρμα, πόροι, διαφημίσεις κ.α.

4.2 ΥΠΗΡΕΣΙΕΣ

Οι παρεχόμενες υπηρεσίες που σχεδιάστηκαν για την εφαρμογή είναι οι εξής:

4.2.1 Δημιουργία Λογαριασμού

Η δημιουργία λογαριασμού, αφορά στην δυνατότητα ενός επιχειρηματία διαμονής (ξενοδόχο, κάτοχο καταλύματος κ.τ.λ.) να μπορεί να δημιουργήσει ελεύθερα τον δικό του λογαριασμό στο σύστημα, προκειμένου να διαχειριστεί τις επιχειρήσεις διαμονής του. Να σημειώσουμε ότι κάθε χρήστης δύναται να έχει παραπάνω από μια επιχειρήσεις π.χ. 2 ξενοδοχεία. Η φόρμα εγγραφής περιγράφεται και μπορούμε να την δούμε στην εικόνα 3.

4.2.2 Πίνακας Ελέγχου Διαχείρισης

Ο πίνακας ελέγχου διαχείρισης, είναι το κέντρο μέσω του οποίου ο χρήστης μπορεί να έχει δυνατότητα για διαχείριση των ξενοδοχείων του, του προγράμματος φιλοξενίας του στην πλατφόρμα κ.τ.λ., όπως μπορούμε να δούμε στην εικόνα 7.

4.2.3 Προβολή Στοιχείων Χρήστη

Όλοι οι χρήστες έχουν την δυνατότητα προβολής των βασικών τους στοιχείων, όπως έχουν καταγραφεί στο σύστημα (όνομα χρήστη, email, ρόλοι κ.α.) όπως μπορούμε να δούμε

στην εικόνα 8. Στην περίπτωση του υπέρ διαχειριστή δίνεται η δυνατότητα απεικόνισης της λίστας χρηστών και της δυνατότητας, στην συνέχεια, προβολής των στοιχείων του εκάστοτε χρήστη πατώντας πάνω στο αντίστοιχο σύνδεσμο.

4.2.4 Απεικόνιση Λίστας Ξενοδοχείων

Ένας επιχειρηματίας διαμονής έχει την δυνατότητα προβολής της λίστας των ξενοδοχείων που έχει δημιουργήσει, καθώς επίσης και την είσοδο σε κάθε ένα από αυτά προκειμένου να τα διαχειριστεί καταλλήλως είτε σε ότι αφορά τα χαρακτηριστικά τους, είτε σε ότι αφορά τις κρατήσεις τους. Την λίστα των ξενοδοχείων όπως την βλέπει μέσω του διαχειριστικού του,

μπορούμε να την δούμε στην εικόνα 9.

4.2.5 Καταχώρηση Νέου Ξενοδοχείου

Είναι εύλογο ότι ο επιχειρηματίας διαμονής μπορεί να δημιουργήσει όσα ξενοδοχεία/καταλλύματα επιθυμεί προκειμένου να τα διαχειριστεί. Έτσι όπως μπορούμε να δούμε στην εικόνα 10, αρκεί να δώσει ένα διακριτό όνομα στο ξενοδοχείο του, ώστε να μπορεί να εμφανιστεί στην λίστα με τα ξενοδοχεία που αναφέραμε παραπάνω και στην συνέχεια να προβεί στις απαραίτητες ρυθμίσεις που το αφορούν.

4.2.6 Πίνακας Ελέγχου Ξενοδοχείου

Ένας χρήστης του συστήματος, όντας διαχειριστής των ξενοδοχείων/καταλλημάτων που έχει δημιουργήσει, έχει την δυνατότητα της εποπτικής απεικόνισης όλων των χαρακτηριστικών και των κρατήσεων που τα αφορούν. Ουσιαστικά αυτό αποτελεί μια εικόνα της κατάστασης του εκάστοτε ξενοδοχείου, στην οποία αναλύονται τα κυριότερα στοιχεία που έχουν σημασία για τους επιχειρηματίες. Αυτά αποτελούνται από μια συνολική εικόνα των χαρακτηριστικών του ξενοδοχείου, και μια εικόνα σχετικά με τις κρατήσεις που έχουν λάβει χώρα. Επιπλέον δίνονται πληροφορίες σχετικά με τις διαθεσιμότητες, τα κόστη κ.α. Σε όλες αυτές τις περιπτώσεις διατίθενται λεπτομέρειες, σχετικά με τις επιπλέον πληροφορίες των χαρακτηριστικών τους που έχουν σημασία για την εύρυθμη λειτουργία του συστήματος και την σωστή εμφάνιση στις αναζητήσεις των απλών χρηστών. Στην εικόνα 11 μπορούμε να δούμε τον πίνακα ελέγχου ενός ξενοδοχείου, όπως θα το έβλεπε ένας ο ξενοδόχος που θα το διαχειριζόταν.

4.2.7 Προβολή/Επεξεργασία Βασικών Πληροφοριών Ξενοδοχείου

Μέσα από τον πίνακα ελέγχου ξενοδοχείου, που αναφέραμε παραπάνω ο διαχειριστής του ξενοδοχείου μπορεί, όπως μπορούμε να δούμε στις εικόνες 12 και 13, να δει αλλά και να διαχειριστεί τις βασικές πληροφορίες του ξενοδοχείου, όπως το διακριτό όνομα, τα στοιχεία επικοινωνίας, τα αστέρια κ.α.

4.2.8 Απεικόνιση Λίστας Παροχών Ξενοδοχείου

Σε ότι αφορά τα χαρακτηριστικά του ξενοδοχείου, ο διαχειριστής θα είναι σε θέση να δει την λίστα με τις παροχές του ξενοδοχείου που έχει θέσει, προσθέτοντας είτε κάποιες από τις προεπιλεγμένες του συστήματος που είναι ευρέως διαδεδομένες προκειμένου να βοηθηθεί, είτε δημιουργώντας τις δικιές του όπως βλέπουμε και στην εικόνα 14.

4.2.9 Διαχείριση Παροχών Ξενοδοχείου

Είναι προφανές, ότι είναι σε θέση να διαχειριστή τις όποιες παροχές που αφορούν στο ξενοδοχείο (π.χ. πισίνα κ.τ.λ.), μετονομάζοντας, διαγράφοντας ή απομακρύνοντας κάποιες από τις γενικές που είχε επιλέξει να συμπεριλάβει. Τα βήματα της διαχείρισης μπορούμε να τα δούμε στις εικόνες 15, 16, 17 και 18.

4.2.10 Απεικόνιση Λίστας Παροχών ανά Τύπο Δωματίων

Στην λογική που αναφερθήκαμε παραπάνω, που αφορούσε την απεικόνιση της λίστας παροχών ξενοδοχείου, ο διαχειριστής του ξενοδοχείου, είναι σε θέση να δει και τις παροχές ανά τύπο δωματίου όπως μπορούμε να δούμε στην εικόνα 19. Η λογική διαφέρει σε σχέση με τις παροχές του ξενοδοχείου, στο γεγονός ότι μπορούν να απεικονιστούν συνολικά οι παροχές για όλα τα δωμάτια, και σε διαφορετικό σημείο ανά τύπους ή τύπο δωματίων/ου.

4.2.11 Διαχείριση Παροχών ανά Τύπο Δωματίου

Ο διαχειριστής μπορεί να ορίσει καταρχάς ποιες είναι συνολικά οι διαθέσιμες παροχές δωματίων ανεξαρτήτως τύπου δωματίου, ενώ στην συνέχεια από αυτές μπορεί να θέσει ποιες είναι διαθέσιμες ανά τύπο δωματίου, προσθέτοντας, ή αφαιρώντας όπως μπορούμε να δούμε στις εικόνες 20, 21, 22, 23, 24 και 25.

4.2.12 Απεικόνιση λίστας / Διαχείριση Τύπων Δωματίου

Άλλη μια δυνατότητα που έχει ο διαχειριστής του ξενοδοχείου, αφορά στην απεικόνιση της λίστας των τύπων δωματίων του εκάστοτε ξενοδοχείου του. Εδώ, όπως έχουμε αναφερθεί και παραπάνω, μπορεί να επιλέξει να συμπεριλάβει κάποιους από τους προεπιλεγμένους τύπους δωματίων αλλά και να δημιουργήσει νέους δικούς του, να τους διαγράψει, να τους μετονομάσει, όπως μπορούμε να δούμε στις εικόνες 27, 28, 29 και 30.

4.2.13 Απεικόνιση Λίστας /Διαχείριση Περιόδων

Αντιστοίχως, ο διαχειριστής μπορεί να δει τις περιόδους που έχει θέσει για το ξενοδοχείο του, αλλά και να τις διαχειριστεί, δημιουργώντας νέες, επεξεργάζοντας τις ημερομηνίες ή ακόμη και διαγράφοντας τες όπως μπορούμε να δούμε στις εικόνες 31, 32, 33 και 34.

4.2.14 Απεικόνιση & Διαχείριση Τιμών ανά Τύπο Δωματίου και Περίοδο

Στις εικόνες 35, 36, 37, 38 και 39 μπορούμε να δούμε την δυνατότητα που δίνεται για την απεικόνιση αλλά και την διαχείριση των τιμών ανά τύπο δωματίου και περίοδο, ώστε ο χρήστης να μπορεί να δημιουργήσει την οικονομική του πολιτική όπως αυτός επιθυμεί. Οι τρόποι που του παρέχονται σε επίπεδο διαχείρισης είναι πολλοί προκειμένου να τον βοηθήσουν. Για παράδειγμα μπορεί να τις τιμές για όλες τις περιόδους όπου στην κάθε περίοδο να αναλύονται ανά τύπο δωματίου ή ανά μια περίοδο όπου μόνο για αυτή αναλύονται οι τιμές ανά τύπο δωματίου. Αλλά μπορεί επίσης να επιλέξει να βλέπει τις τιμές για όλους τους τύπους δωματίων όπου σε κάθε έναν από αυτούς να παρουσιάζονται αυτές για την εκάστοτε περίοδο ή για ένα συγκεκριμένο τύπο δωματίου να παρουσιάζονται οι τιμές για όλες τις διαθέσιμες περιόδους. Έτσι ανάλογα με το τι θεωρεί πιο σημαντικό ο διαχειριστής (τύπο δωματίου ή περίοδο) θα είναι σε θέση να διαχειριστή με τον καλύτερο και ευκολότερο δυνατό τρόπο τις τιμές των δωματίων του.

4.2.15 Απεικόνιση & Διαχείριση Διαθεσιμότητας ανά Τύπο Δωματίου

Επιπλέον, ο διαχειριστής μέσω κατάλληλου διαδραστικού ημερολογίου είναι σε θέση να βλέπει την εναπομείνασα διαθεσιμότητα ανά τύπο δωματίου ανεξαρτήτως περιόδου, αλλά και να είναι σε θέση να την τροποποιεί, είτε αυξάνοντας τα διαθέσιμα δωμάτια, είτε μειώνοντας τα, είτε ακόμη και μηδενίζοντας τα. Το διαχειριστικό που αφορά στα παραπάνω μπορούμε να το δούμε στην εικόνα 40.

4.2.16 Απεικόνιση Κρατήσεων σε Λίστα ή Ημερολόγιο

Στην εικόνα 41 και 42 αντίστοιχα, μπορούμε να δούμε τις κρατήσεις που έχουν γίνει στο σύστημα για το εκάστοτε ξενοδοχείο ενός χρήστη από τους απλούς χρήστες, τόσο σε λίστα με τα βασικά στοιχεία τους όπως συνολικό κόστος, περίοδος διαμονής κ.τ.λ. αλλά και σε ημερολόγιο με κατάλληλα χρώματα, όπως αυτά ορίζονται κατά την δημιουργία των τύπων δωματίου, που μέσα αναγράφουν το όνομα του χρήστη που έκανε την κράτηση. Τα χρώματα βοηθούν τον διαχειριστή να βλέπει με μια ματιά τον τύπο δωματίου που αφορά η εκάστοτε κράτηση του χρήστη που αναγράφεται πάνω στην γραμμή.

4.2.17 Προβολή Πίνακα Ελέγχου Χρήστη / Διαχειριστή

Τέλος όπως μπορούμε να δούμε και στις εικόνες 43 και 44, ο αρχικός πίνακας ελέγχου χρηστών με διαφορετικούς ρόλους από αυτούς του επιχειρηματία διαμονής, δηλαδή για τους ρόλους του απλού χρήστη που κάνει μια κράτηση, ή του υπέρ-διαχειριστή του συστήματος διαφέρουν, αφού έχουν διαφορετικούς σκοπούς να εξυπηρετήσουν. Έτσι για παράδειγμα στην περίπτωση του απλού χρήστη ο πίνακας ελέγχου θα του είχε την επιλογή να δει τις κρατήσεις του και τις αποδείξεις/τιμολόγια του, ενώ στον υπέρ διαχειριστή όπως αναφέραμε και σε προηγούμενο σημείο να διαχειριστεί τις συνδρομές των επιχειρηματιών διαμονής στην πλατφόρμα, τα στατιστικά επισκέψεων, τις κρατήσεις και τους τζίρους που γίνονται κ.α.

4.3 ΡΥΘΜΙΣΕΙΣ ΕΦΑΡΜΟΓΗΣ

Η εγκατάσταση της εφαρμογής προϋποθέτει τον ορισμό ενός συνόλου ρυθμίσεων, απαραίτητων για την ομαλή λειτουργία. Αυτές οι ρυθμίσεις αφορούν τις ρυθμίσεις για την διασύνδεση με την βάση δεδομένων αλλά και τα υποσυστήματα framework που χρησιμοποιούνται όπως για παράδειγμα το υποσύστημα της ασφάλειας (security.xml)

Στις παραπάνω ρυθμίσεις θα πρέπει να προστεθούν και οι ρυθμίσεις του αρχείου καταγραφής.

Σε αυτές ορίζονται ένα η περισσότερα κανάλια καταγραφής και ένας η περισσότεροι handlers που αναλαμβάνουν να καταγράψουν τα διαφορετικά μηνύματα της εφαρμογής.

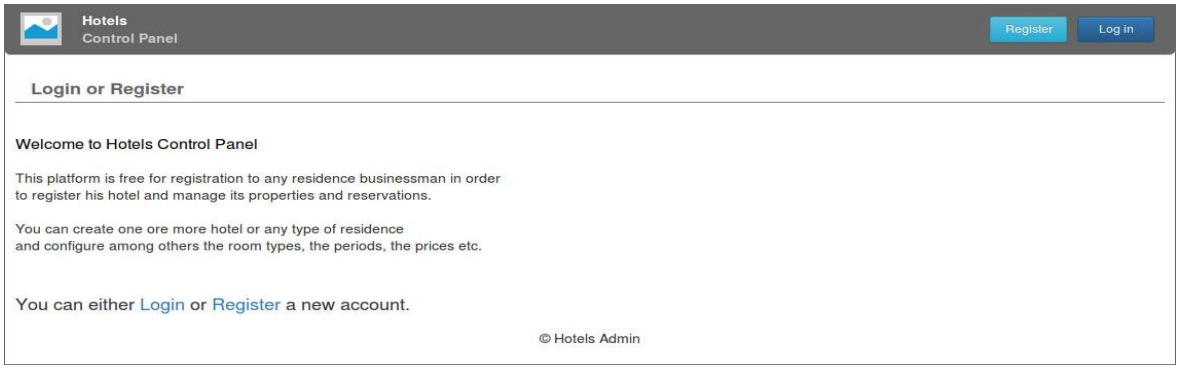
Σημειώνεται πως ακολουθείται το πρότυπο των επιπέδων καταγραφής FATAL, ERROR, WARN, INFO, DEBUG, TRACE.

Το σύνολο των ρυθμίσεων αυτών, αποθηκεύεται σε διάφορα αρχεία ρυθμίσεων (.xml) τα οποία αποτυπώνονται λεπτομερώς στο παράρτημα.

5. ΣΕΝΑΡΙΑ ΧΡΗΣΗΣ

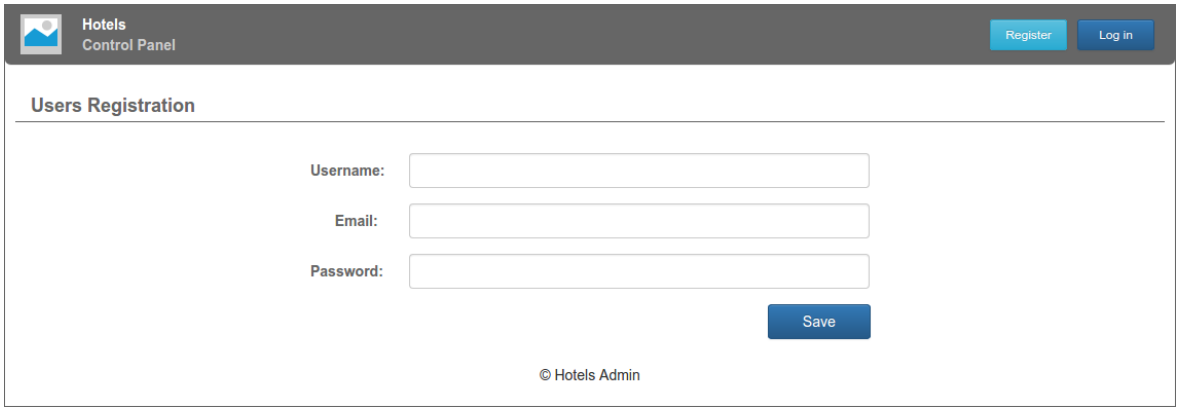
Ακολούθως αναλύεται όλη η διαδικασία χρήσης της πλατφόρμας και πιο συγκεκριμένα των υποσυστημάτων που έχουν υλοποιηθεί όπως αυτή παρουσιάζεται στο χρήστη.

Η είσοδος στην εφαρμογή παρουσιάζει στον χρήστη μια σελίδα που τον καλωσορίζει και του και τον ενημερώνει για τους στόχους της υπηρεσίας (πλατφόρμας) και τα οφέλη που αποκομίζονται από αυτή. Επιπλέον, δίνει τις επιλογές εισόδου ή εγγραφής είτε μέσω link στο σώμα της σελίδας είτε μέσω των κουμπιών στο πάνω δεξιά μέρος., που ενημερώνει για τους στόχους της υπηρεσίας και τα οφέλη που αποκομίζονται από αυτή. Οι δύο αυτές περιπτώσεις φαίνονται ακολούθως:



Εικόνα 2 – Αρχική Σελίδα

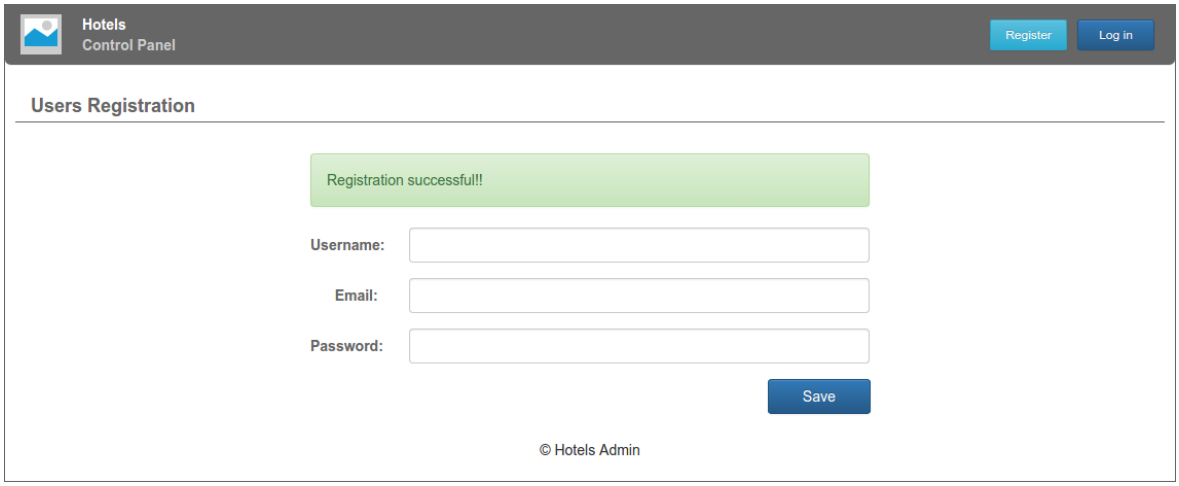
Στην περίπτωση που ο χρήστης δεν έχει λογαριασμό μπορεί να κάνει εγγραφή στο σύστημα συμπληρώνοντας το όνομα χρήστη το email και τον κωδικό που επιθυμεί όπως φαίνεται στην παρακάτω εικόνα.



Εικόνα 3 – Φόρμα Εγγραφής Χρήστη

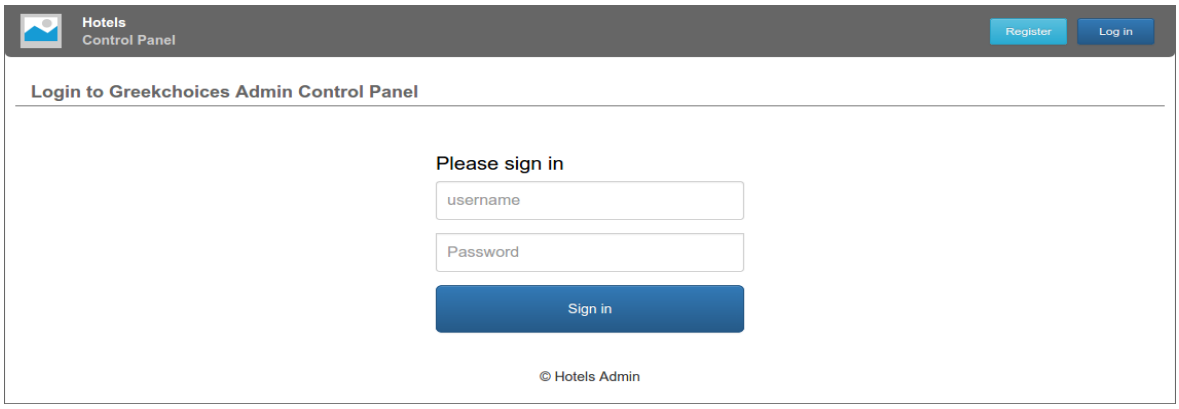
Να σημειώσουμε εδώ ότι στην παρούσα υλοποίηση δεν γίνεται έλεγχος αν το όνομα χρήστη είναι υπαρκτό, αν το email είναι έγκυρο ή έχει ξαναχρησιμοποιηθεί αλλά ούτε αν γίνεται σεβαστή η όποια πολιτική ασφαλείας που σχετίζεται με τους κωδικούς καθώς αυτό θα αφορά μελλοντική υλοποίηση (next iteration).

Εφόσον η εγγραφή είναι επιτυχής, εμφανίζεται κατάλληλο μήνυμα σε πράσινο πλαίσιο για την επιβεβαίωση του χρήστη ότι όλα ολοκληρώθηκαν ομαλά με το μήνυμα "Registration Successful".



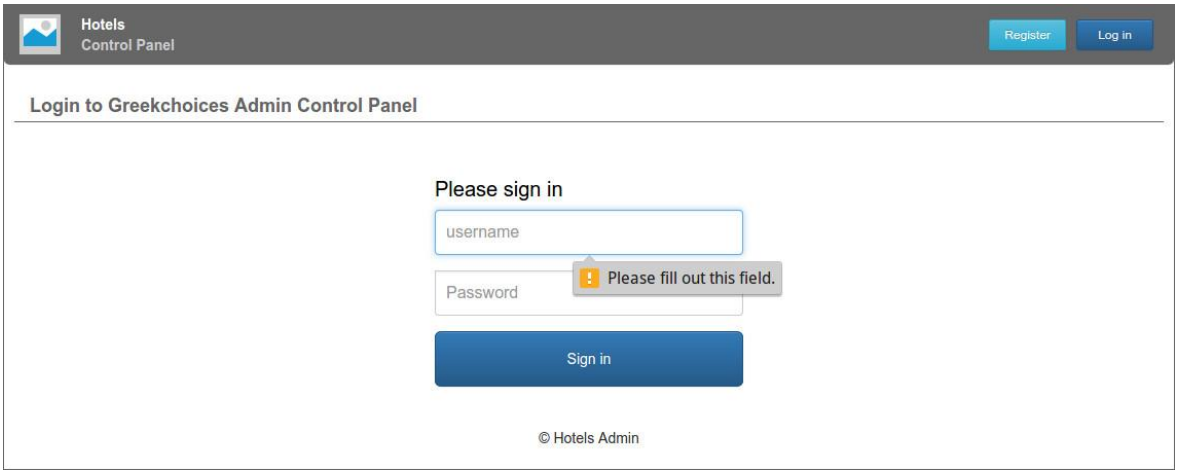
Εικόνα 4 – Μήνυμα Επιτυχούς Εγγραφής Χρήστη

Στην περίπτωση που ο χρήστης επιλέξει να εισέλθει στο σύστημα δίνοντας τα στοιχεία σύνδεσης του, παρουσιάζεται η φόρμα εισόδου μέσω της οποίας του ζητείται να εισάγει το όνομα χρήστη του και τον κωδικό του, όπως φαίνεται στην εικόνα παρακάτω :



Εικόνα 5 – Φόρμα Εισόδου στο Σύστημα

Σε περίπτωση που ο χρήστης δεν συμπληρώσει κάποιο από τα απαραίτητα στοιχεία για την είσοδο του και πατήσει το κουμπί εισόδου ”Sign in” θα εμφανιστεί κατάλληλο μήνυμα προτρέποντάς τον να το συμπληρώσει.



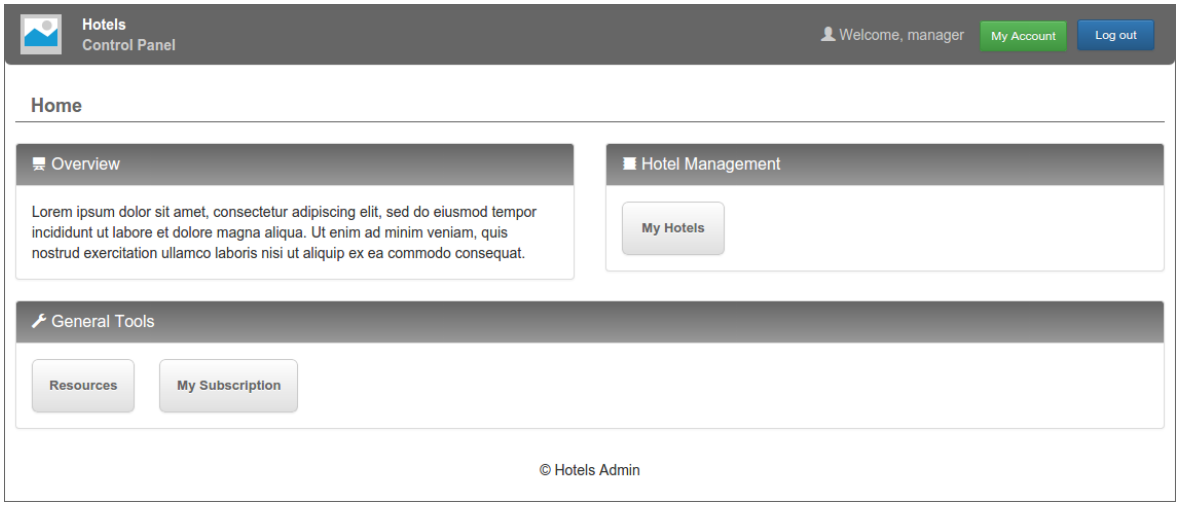
Εικόνα 6 – Σφάλμα στην Συμπλήρωση της Φόρμας Εισόδου

Σε περίπτωση που ξανά προσπαθήσει να εισέλθει χωρίς να συμπληρώσει το πεδίο που είναι κενό και πάλι θα δει ένα αντίστοιχο μήνυμα λάθους.

Παρακάτω θα περιγράψουμε τις οθόνες του χρήστη που έχει τον ρόλο του κατόχου μιας επιχείρησης διαμονής.

Έτσι όπως θα δούμε και στην εικόνα παρακάτω, όταν ο χρήστης εισέλθει επιτυχώς στο σύστημα βλέπει ένα πίνακα ελέγχου ο οποίος είναι χωρισμένος σε τρεις βασικές ζώνες.

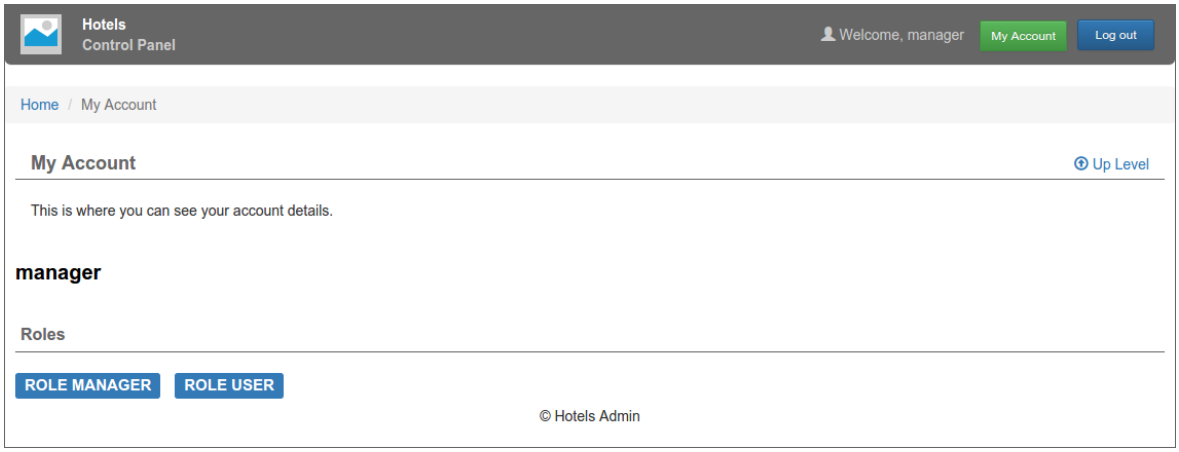
Την περίληψη (Overview), που μπορεί να τον ενημερώνει για διάφορα στοιχεία που έχει επιλέξει να του εμφανίζονται εκεί (π.χ. εκκρεμείς ή νέες κρατήσεις) , την διαχείριση των επιχειρήσεων διαμονής του (Hotel Management) και τέλος τα γενικά εργαλεία. Από εκεί που ορίζει τον τύπο συνδρομής του στην πλατφόρμα κ.τ.λ. Επιπλέον στο πάνω μέρος στα δεξιά, βλέπουμε ότι του εμφανίζει μήνυμα καλωσορίσματος με το όνομα χρήστη του καθώς και δύο κουμπιά που αφορούν στον λογαριασμό του (My Account) και στην αποσύνδεση από το σύστημα (Log out).



Εικόνα 7 – Πίνακας Ελέγχου Διαχειριστή (Ξενοδόχου)

Στην περίπτωση που πατήσει το κουμπί “My Account” θα εμφανιστεί μια σελίδα που θα του εμφανίσει τα βασικά στοιχεία του λογαριασμού του, όπως το όνομα χρήστη του και τον ρόλο που διαθέτει στην πλατφόρμα.

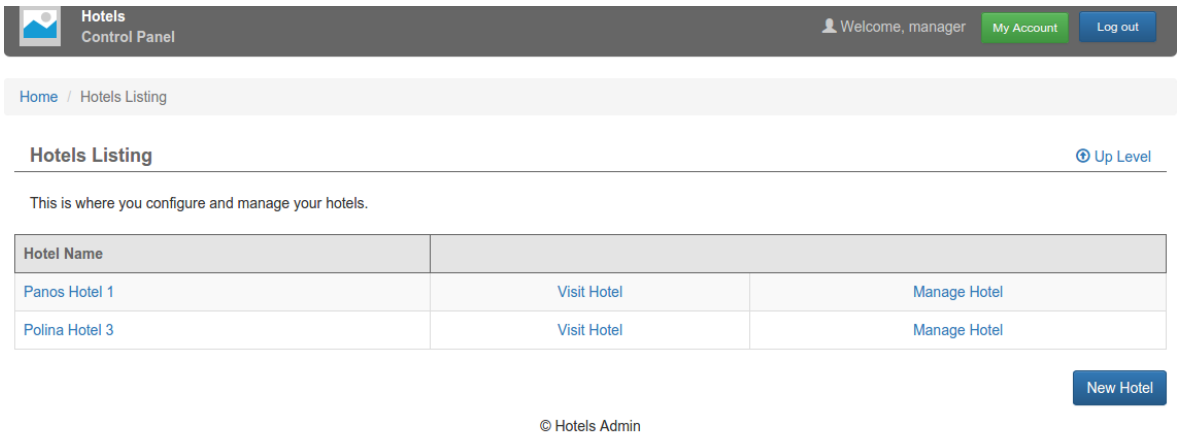
Στο μέλλον μπορεί να εμφανίζονται περισσότερες σχετικές πληροφορίες.



Εικόνα 8 – Προβολή Στοιχείων Χρήστη

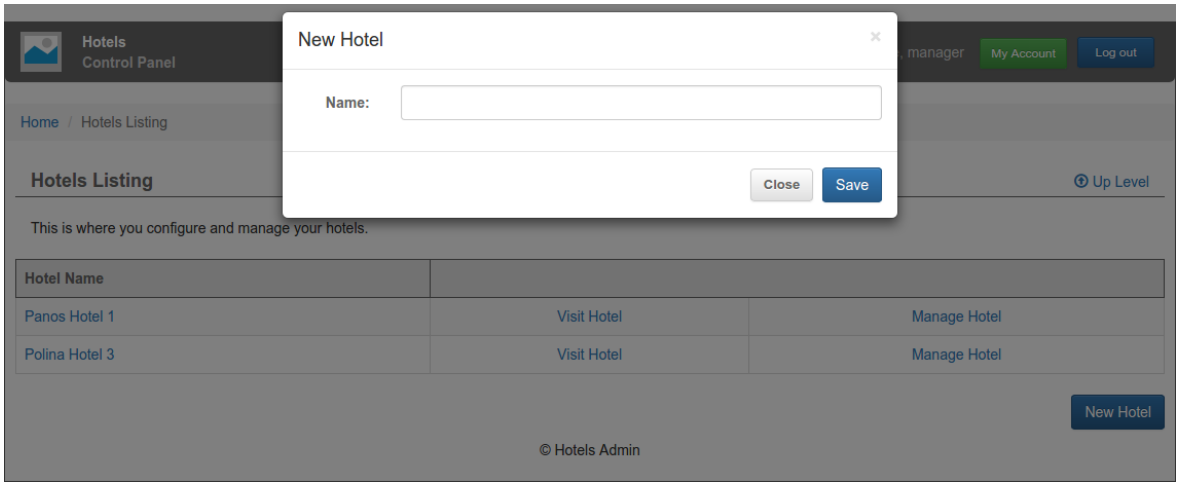
Κάνοντας κλικ στο κουμπί “My Hotels” στο Hotel Management όπως είδαμε στην εικόνα 7, βρισκόμαστε στην οθόνη που εμφανίζεται μια λίστα από τα ξενοδοχεία που έχουμε ήδη δημιουργήσει και μπορούμε να διαχειριστούμε. Αν είναι η πρώτη φορά που βλέπουμε αυτή την οθόνη ή δεν έχουμε δημιουργήσει ξενοδοχείο τότε η λίστα αυτή θα είναι άδεια.

Μέσω του μπλε κουμπιού στο κάτω μέρος δεξιά “New Hotel” μπορουμε να καταχωρήσουμε στο σύστημα ένα νέο ξενοδοχείο.



Εικόνα 9 – Λίστα Ξενοδοχείων Ξενοδόχου

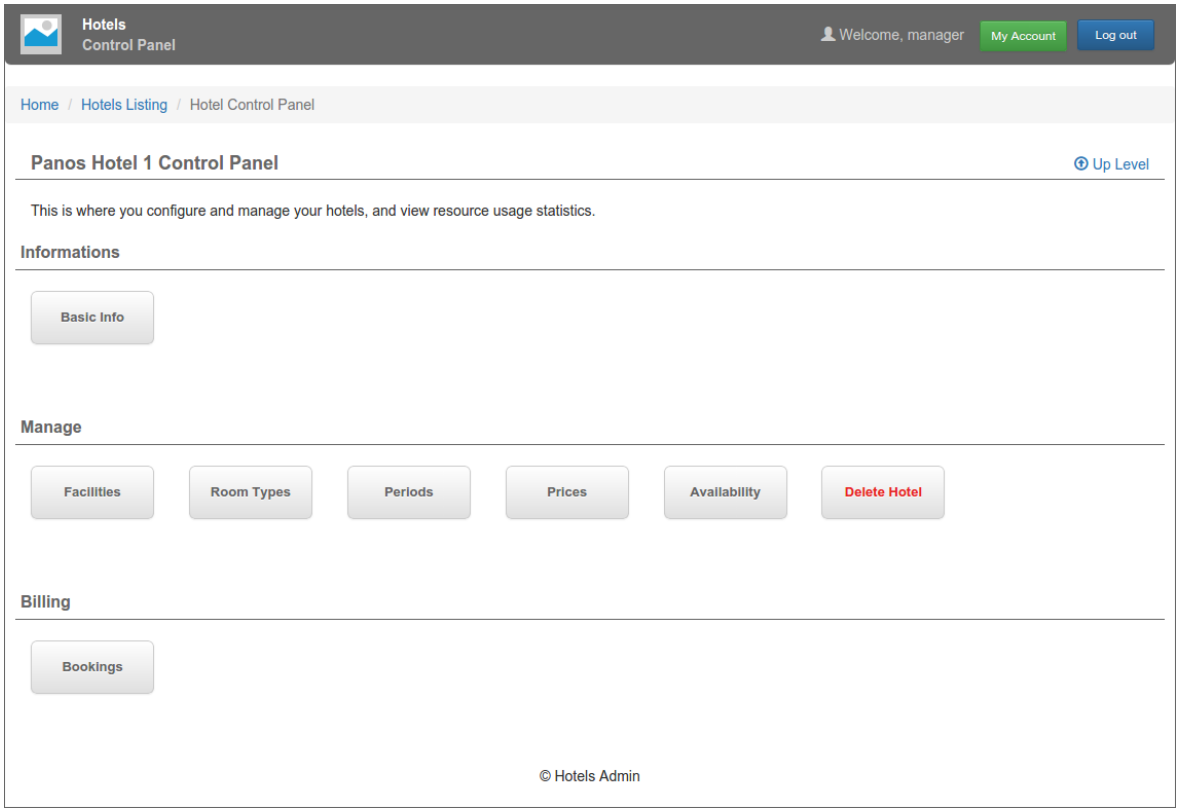
Στον χρήστη αφού πατήσει το κουμπί “New Hotel” θα του εμφανιστεί μια αναδυόμενη φόρμα (pop-up/modal) μέσω της οποίας μπορεί να δηλώσει το όνομα το οποίο θα χαρακτηρίζει αυτό το ξενοδοχείο και με το οποίο θα εμφανίζεται στην λίστα που προαναφέραμε.



Εικόνα 10 – Καταχώρηση Νέου Ξενοδοχείου

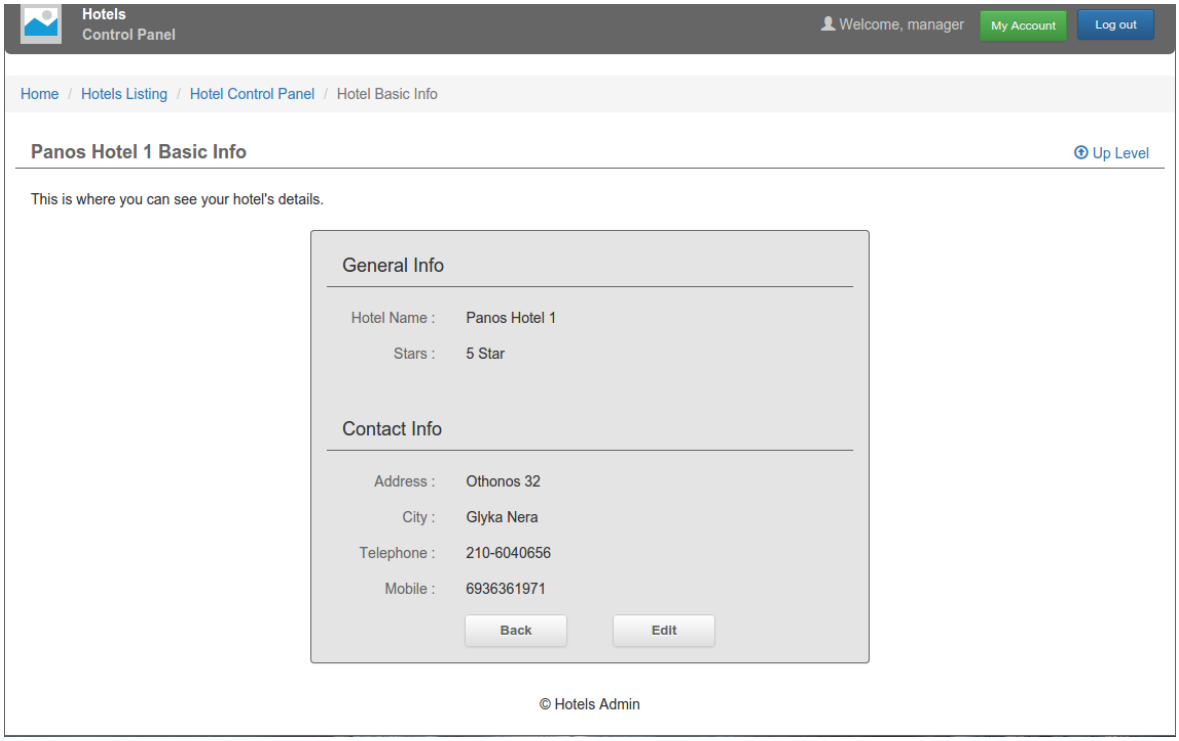
Στην περίπτωση που ο χρήστης πατήσει το κουμπί “Manage Hotel” όπως φαίνεται στην εικόνα 9, τότε θα εισέλθει στον πίνακα ελέγχου του ξενοδοχείου που αυτό σχετιζόταν κατά την οριζόντια διάταξη.

Στον παρακάτω πίνακα ελέγχου μπορούμε να δούμε ότι η διαχείριση χωρίζεται σε τρείς βασικές ζώνες. Τις πληροφορίες (Informations), την διαχείριση (Manage) και τις χρεώσεις (Billing). Σε κάθε μια από αυτές τις τρείς ζώνες, υπάρχουν ένα η περισσότερα κουμπιά που παραπέμπουν στις αντίστοιχες ενέργειες που ο χρήστης θέλει να εκτελέσει.



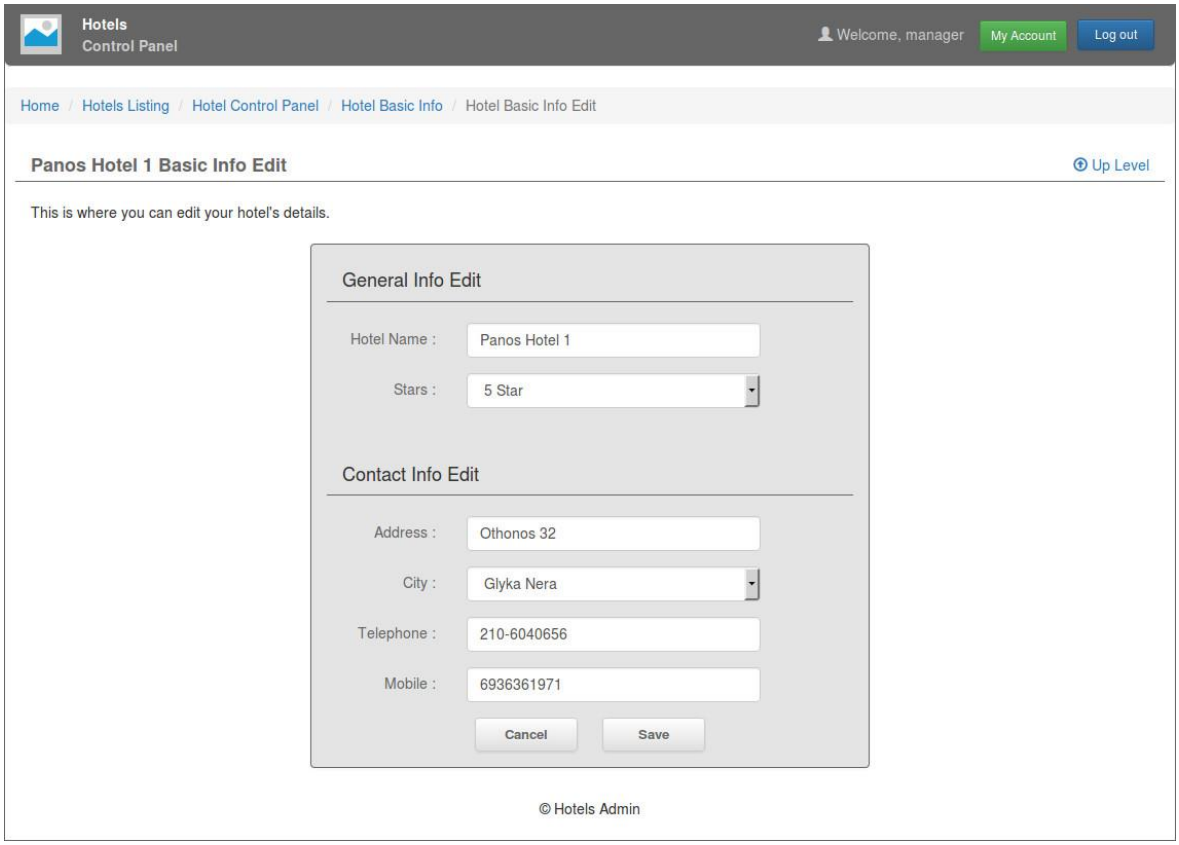
Εικόνα 11 – Πίνακας Ελέγχου Ξενοδοχείου

Στην πρώτη ζώνη, πατώντας στο πλήκτρο “Basic Info”, ο χρήστης θα οδηγηθεί σε μια οθόνη μέσω της οποίας θα δει τις βασικές πληροφορίες που σχετίζονται με το συγκεκριμένο ξενοδοχείο. Αυτές απαρτίζονται από δύο μέρη, τις γενικές πληροφορίες και τις πληροφορίες που σχετίζονται με την επικοινωνία όπως φαίνεται στην παρακάτω εικόνα.



Εικόνα 12 – Βασικές Πληροφορίες Ξενοδοχείου

Ο χρήστης έχει δύο επιλογές από αυτή την οθόνη. Είτε να γυρίσει πίσω στον πίνακα ελέγχου του ξενοδοχείου πατώντας το κουμπί “Back”, είτε να προχωρήσει στην οθόνη επεξεργασίας πατώντας το κουμπί “Edit”. Σε αυτή την περίπτωση θα βρεθεί σε μια προ-συμπληρωμένη φόρμα με τα στοιχεία που είναι ήδη καταχωρημένα και την δυνατότητα να τα τροποποιήσει αλλάζοντας τα πεδία που επιθυμεί και πατώντας “Save” ή να επιστρέψει στην προηγούμενη οθόνη πατώντας “Cancel” όπως μπορούμε να δούμε στην παρακάτω εικόνα.



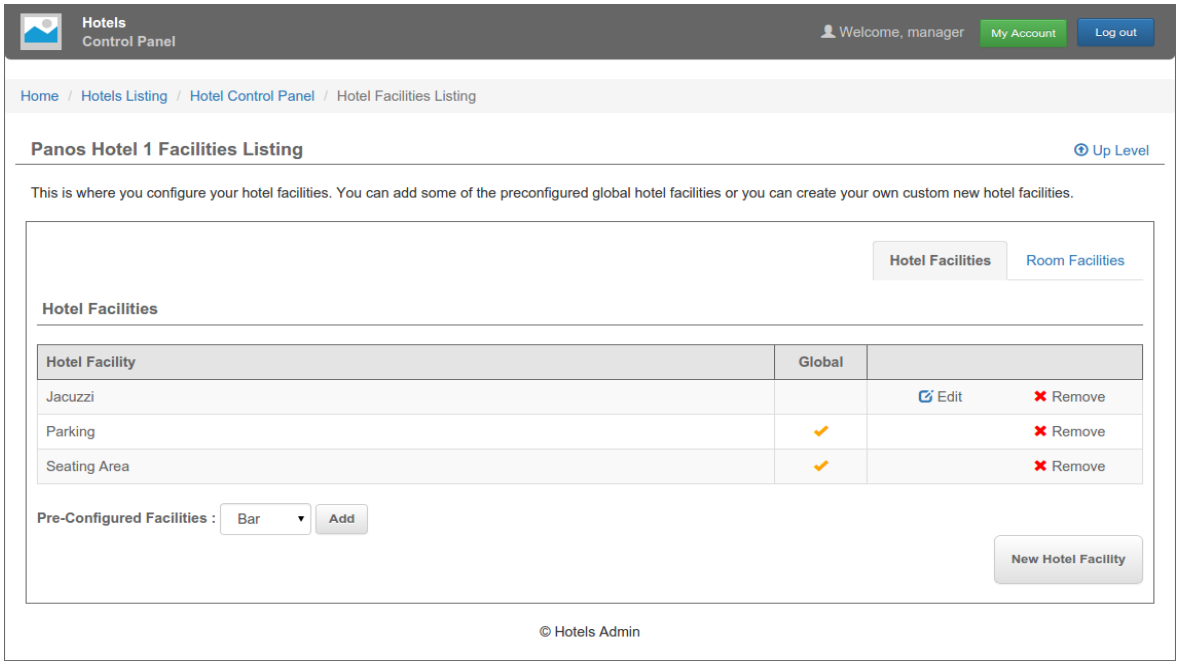
Εικόνα 13 - Επεξεργασία Βασικών Πληροφοριών

Όπως είδαμε στην εικόνα 11, στην δεύτερη ζώνη (Manage) ο χρήστης μπορεί να επιλέξει μέσα από ένα πλήθος κουμπιών όπως αναφέρθηκαμε και παραπάνω προκειμένου να εκτέλεσει διάφορες εργασίες που αφορούν το συγκεκριμένο ξενοδοχείο που διαχειρίζεται.

Το πρώτο κουμπί αυτής της ομάδας είναι το “Facilities” και αφορά στην διαχείριση των παροχών που έχει το ξενοδοχείο γενικότερα. Έτσι όπως βλέπουμε στην παρακάτω εικόνα οι παροχές χωρίζονται σε δύο κατηγορίες, σε αυτές του ξενοδοχείο και εκείνες ανά τύπο δωματίου, ενώ μπορούμε να τις διαχειριστούμε ξεχωριστά επιλέγοντας την αντίστοιχη επιλογή “tab” στο πάνω μέρος.

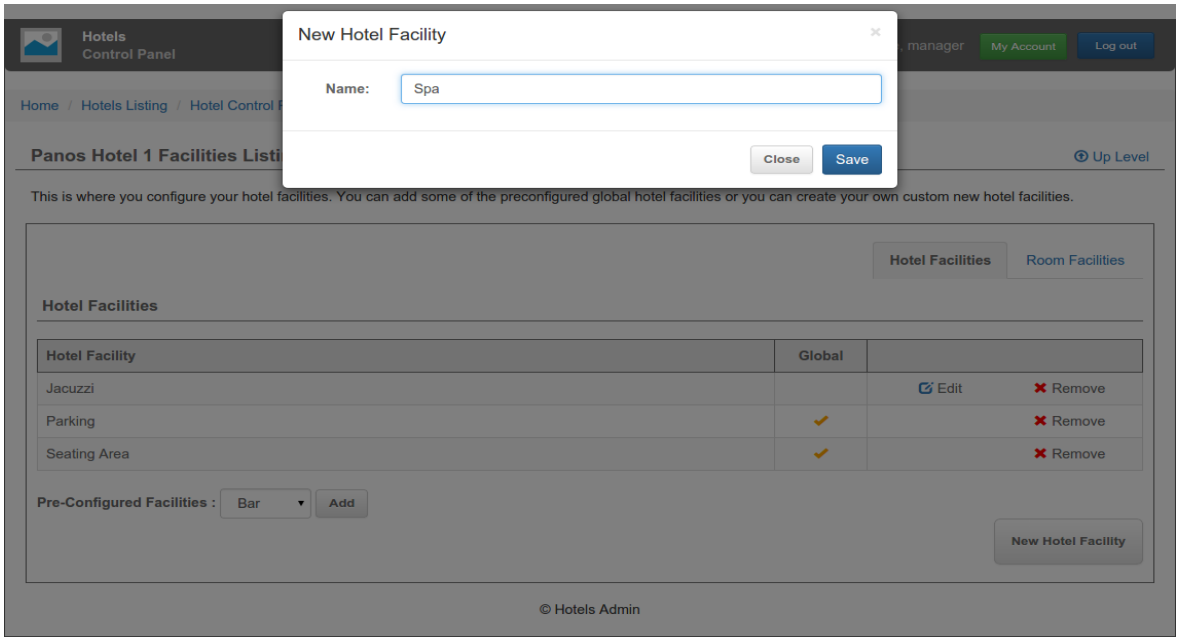
Κατά την είσοδο μας σε αυτή την οθόνη από τον πίνακα ελέγχου του ξενοδοχείου μέσω του κουμπιού “Facilities” είναι προεπιλεγμένες όπως είναι εμφανές οι παροχές που αφορούν στο ξενοδοχείο και οι οποίες εμφανίζονται από κάτω μέσω μιας λίστας.

Τέλος μέσω του κουμπιού “New Hotel Facility” μπορεί ο χρήστης να δημιουργήσει μια νέα προσωποποιημένη για το ξενοδοχείο του παροχή, αφού αυτή δεν είναι διαθέσιμη στις προεπιλεγμένες για να την συσχετίσει.



Εικόνα 14 – Λίστα Παροχών Ξενοδοχείου

Πατώντας το κουμπί “New Hotel Facility”, θα εμφανιστεί ένα αναδυόμενο παράθυρο (pop-up/modal) προκειμένου να εισάγει το όνομα της παροχής που επιθυμεί να καταχωρήσει και συσχετίσει με το ξενοδοχείο που διαχειρίζεται την δεδομένη στιγμή.



Εικόνα 15 –Προσθήκη Νέας Ξενοδοχειακής Παροχής

Στην παρακάτω εικόνα μπορούμε να δούμε τις παροχές που είναι διαθέσιμες κεντρικά μέσω της πλατφόρμας προς χρήση, για ευκολία του χρήστη μέσω του κίτρινου εικονιδίου (check), ενώ οι υπόλοιπες (στο γκρί) είναι προσωποιημένες του χρήστη (άνευ εικονιδίου).

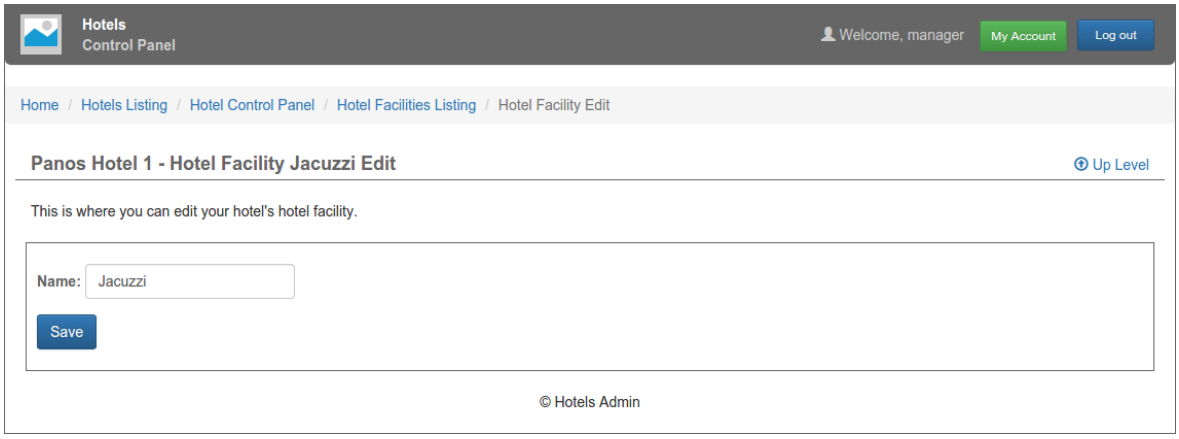
Επιπλέον, στο κάτω μέρος αριστερά βλέπουμε ότι υπάρχουν οι κεντροποιημένες/προδιαγεγραμένες παροχές (pre-configured Facilities) που μπορεί ένας χρήστης να επιλέξει. Αν κάποια από αυτές έχει ήδη χρησιμοποιηθεί, τότε αυτή δεν θα εμφανίζεται πλέον στις διαθέσιμες επιλογές (drop down).

Εικόνα 16 – Προσθήκη Προεπιλεγμένων Ξενοδοχειακών Παροχών

Στην τελευταία στήλη υπάρχει η δυνατότητα επεξεργασίας ή διαγραφής. Σε ότι αφορά τις προσωποποιημένες υπάρχει η επιλογή της επεξεργασίας προκειμένου να αλλάξει το όνομα της παροχής καθώς επίσης και η διαγραφή προκειμένου να την διαγράψει.

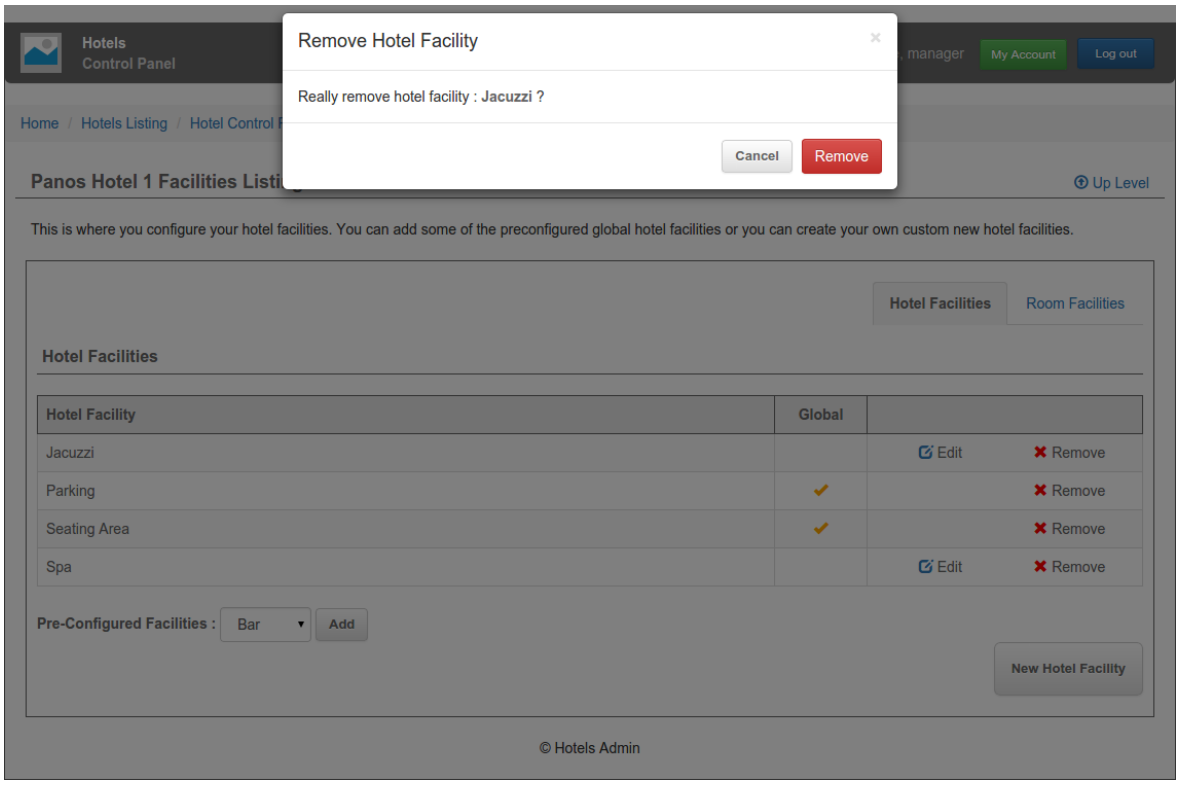
Στη περίπτωση δε των γενικευμένων εκ' του συστήματος παροχών, ο χρήστης δεν δικαιούται την επεξεργασία και τροποποίηση της καθώς αυτό θα σήμαινε ότι θα την άλλαζε μαζικά για όλους όσους επέλεξαν να την χρησιμοποιήσουν ως ήταν, ενώ η διαγραφή απλά σημαίνει ότι θα αποσυσχετιζόταν από τις παροχές του συγκεκριμένου ξενοδοχείου.

Στην παρακάτω εικόνα βλέπουμε την οθόνη επεξεργασίας μιας παροχής.



Εικόνα 17 – Επεξεργασία Ξενοδοχειακής Παροχής

Ενώ στην επόμενη εικόνα, βλέπουμε το αναδυόμενο παράθυρο προειδοποίησης, στην περίπτωση που ο χρήστης επίλεξε την διαγραφή μιας παροχής.

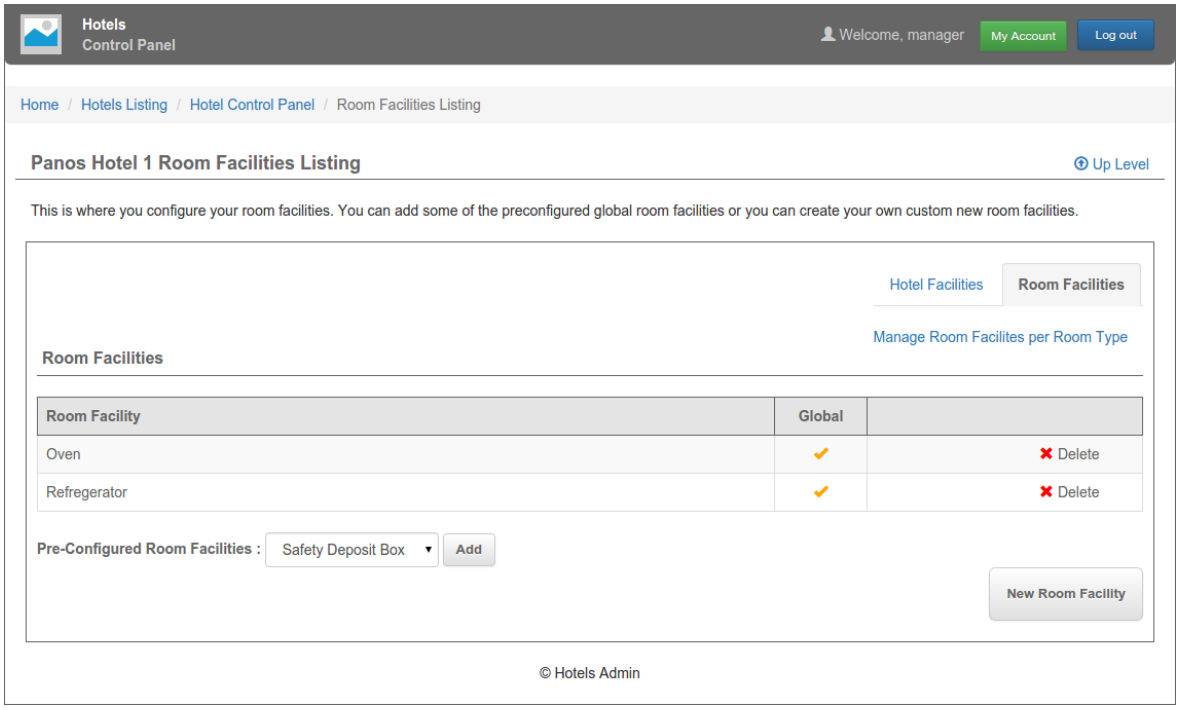


Εικόνα 18 – Διαγραφή Ξενοδοχειακής Παροχής

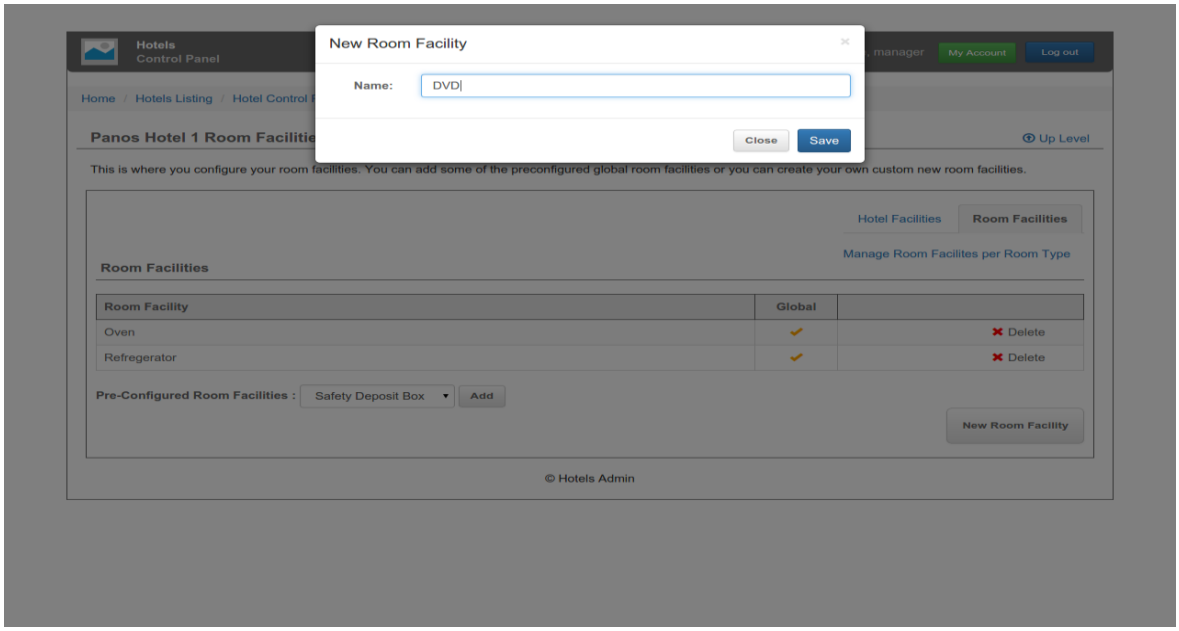
Στην περίπτωση που η παροχή είναι κεντροποιημένη/προκαθορισμένη τότε απλά θα αποσυσχετιστεί με το ξενοδοχείο και θα επιστρέψει ως επιλογή στις διαθέσιμες προεπιλεγμένες (pre-configured Facilities), ενώ στην περίπτωση που είναι προσωποποιημένη θα διαγραφεί εντελώς από το σύστημα.

Όπως και στις παροχές που σχετίζονται με το ξενοδοχείο, έτσι και εδώ μπορούμε να κάνουμε χρήση των προκαθορισμένων ή των προσωποποιημένων, να δημιουργήσουμε νέες προσωποποιημένες παροχές και τέλος να τις επεξεργαστεί ή να τις διαγράψει βάση της λογικής στην οποία αναφερθήκαμε.

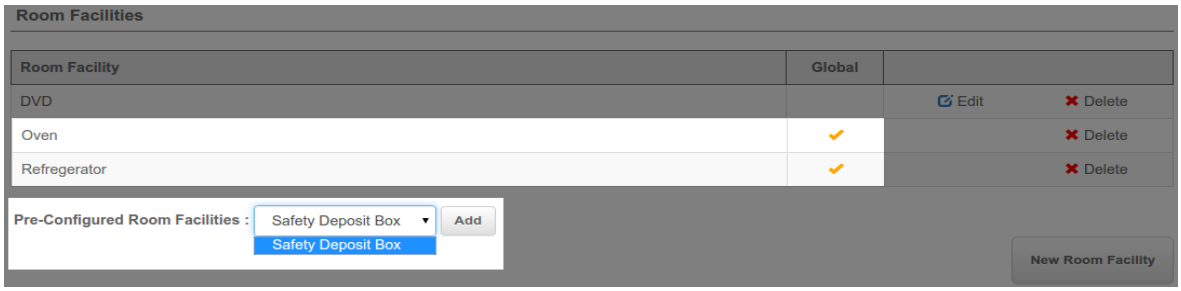
Παρακάτω βλέπουμε τις οθόνες που αφορούν στις παροχές που σχετίζονται με τα δωμάτια.



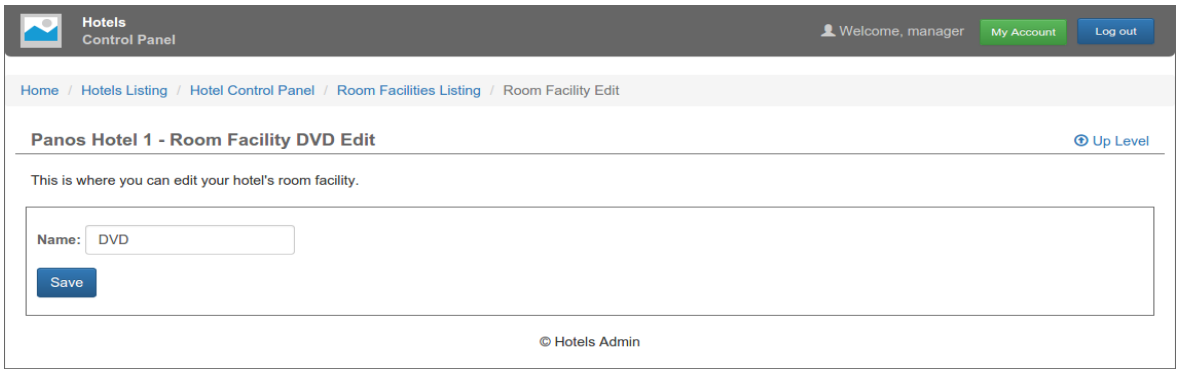
Εικόνα 19 – Λίστα Παροχών Δωματίων



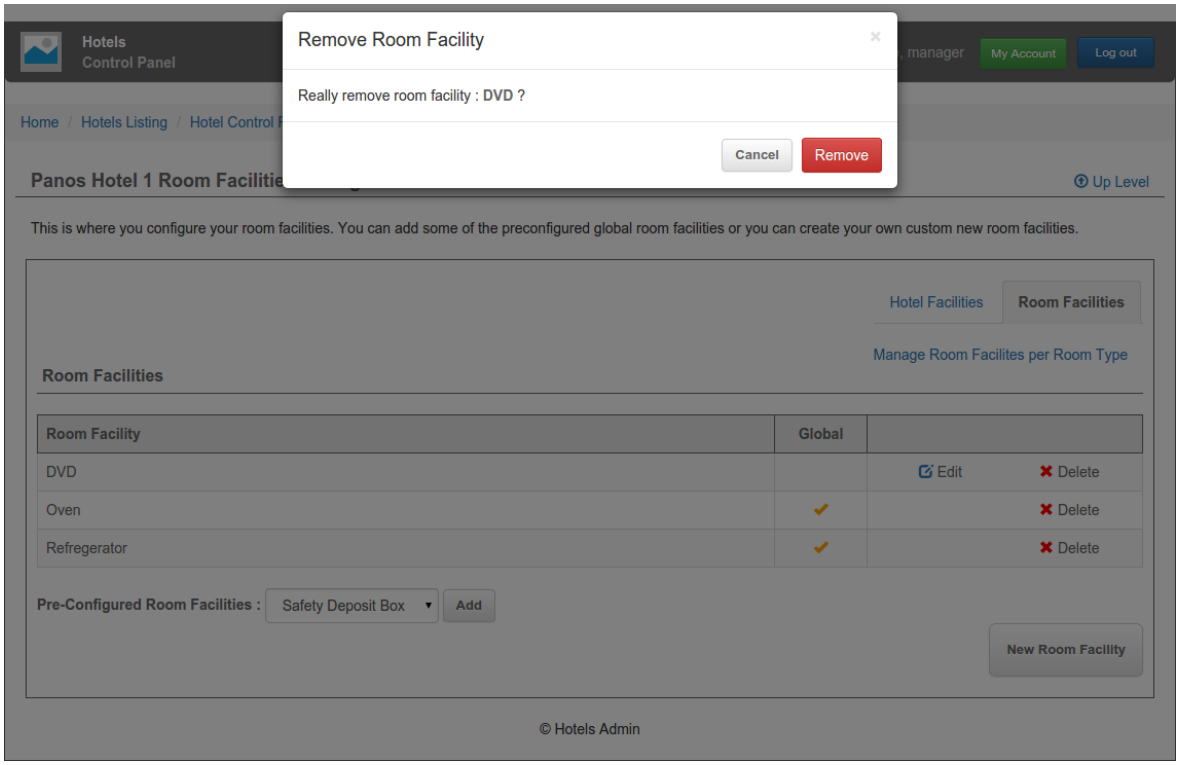
Εικόνα 20 – Προσθήκη Νέας Παροχής Δωματίου



Εικόνα 21 – Προσθήκη Προεπιλεγμένης Παροχής Δωματίου

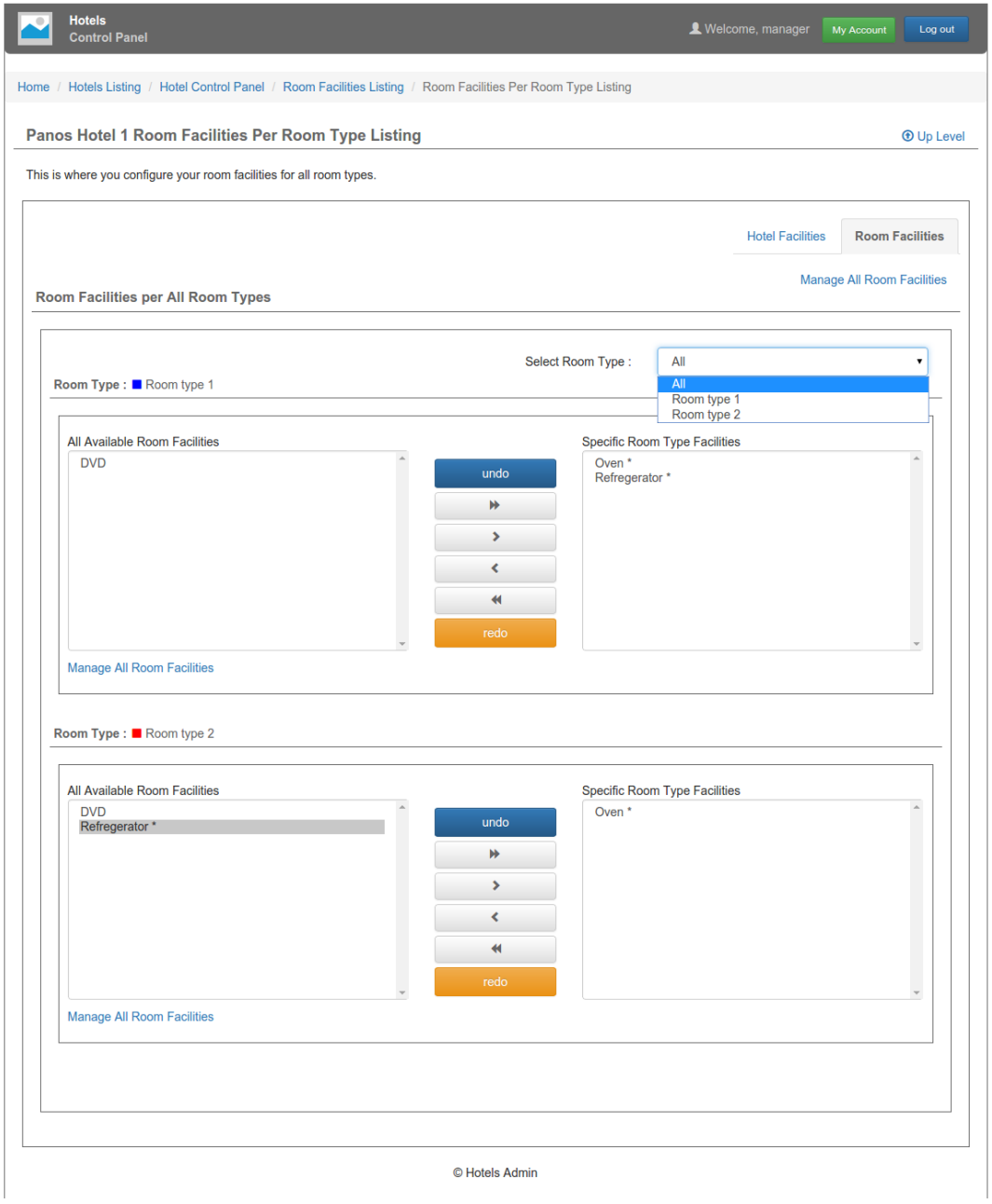


Εικόνα 22 – Επεξεργασία Παροχής Δωματίου



Εικόνα 23 – Διαγραφή Παροχής Δωματίου

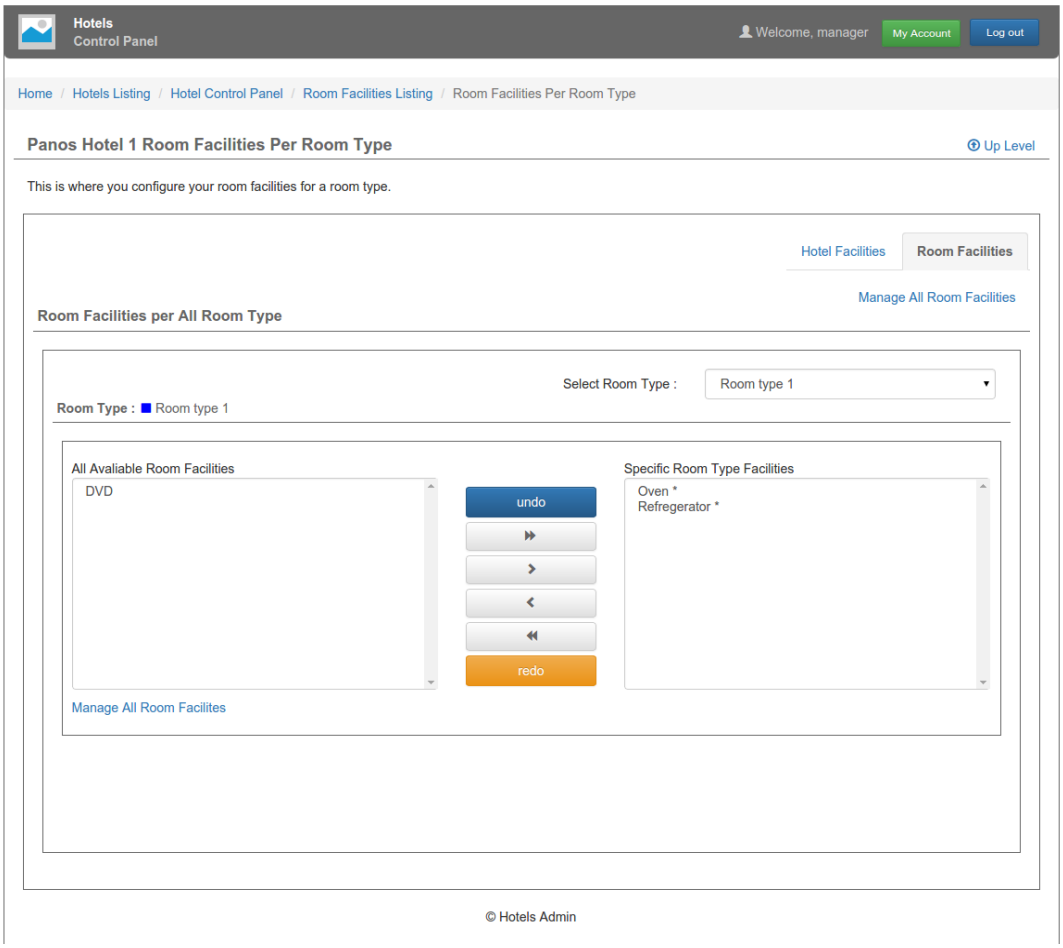
Εκτός από τα κοινά στοιχεία που μπορούμε να δούμε μεταξύ των διαχειριστικών οθονών που αφορούν τις παροχές ξενοδοχείου και δωματίων, υπάρχει άλλη μια επιλογή αυτή της διαχείρισης των παροχών δωματίου ανά τύπο δωματίου μέσω του συνδέσμου “Manage Room Facilities per Room Type”. Αυτή η επιλογή οδηγεί στην οθόνη μέσω της οποίας μπορούμε να ορίσουμε τις παροχές δωματίων ανά τύπο δωματίου, επιλέγοντας από τις διαθέσιμες που διαχειριστήκαμε στην προηγούμενη οθόνη.



Εικόνα 24 – Παροχές Δωματίου ανά Τύπο Δωματίου

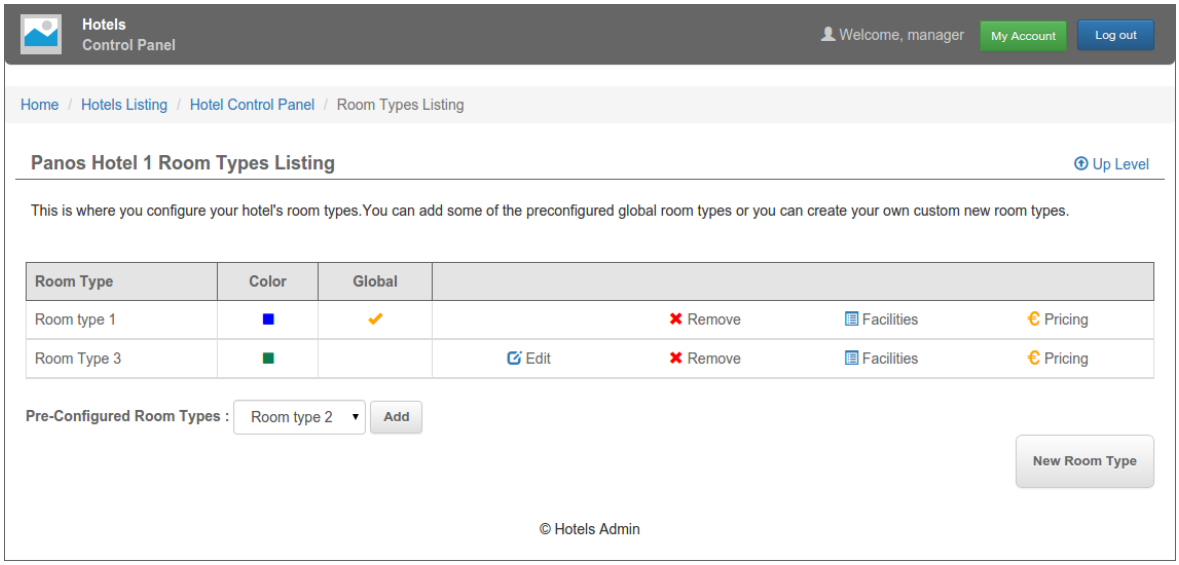
Στα αριστερά εμφανίζονται όλες οι διαθέσιμες παροχές δωματίων όπως έχουν οριστεί εκτός από αυτές που βρίσκονται ήδη στα δεξιά και αυτό εμφανίζεται ανά τύπο δωματίου. Η μεταφορά μιας παροχής γίνεται απλά επιλέγοντας μια ή παραπάνω και πατώντας το μονό βελάκι που κοιτά στα δεξιά. Αντιστοίχως η απ' επιλογή μπορεί να γίνει είτε με το κουμπί "Undo" είτε με την επιλογή από την δεξιά λίστα του εκάστοτε τύπου δωματίου και πατώντας το κουμπί με το μονό βελάκι που δείχνει στ' αριστερά. Τα διπλά βελάκια μεταφέρουν το μεν δεξί όλες τις διαθέσιμες παροχές στον τύπο δωματίου, το δεν αριστερό αφαιρεί όλες τις παροχές από ένα τύπου δωματίου. Τέλος το κουμπί "Redo" σε αντίθεση με το κουμπί "Undo" εκτελεί εκ' νέου την τελευταία εντολή. Να σημειώσουμε επίσης ότι με αστεράκι εμφανίζονται εκείνες οι παροχές που είναι προκαθορισμένες/κεντροποιημένες εκ' του συστήματος.

Αν το επιθυμεί ο χρήστης μπορεί να επιλέξει συγκεκριμένο τύπο δωματίου για να διαχειριστή τις παροχές δωματίου μέσω της επιλογής στο "Select Room Type" (Drop Down Menu).



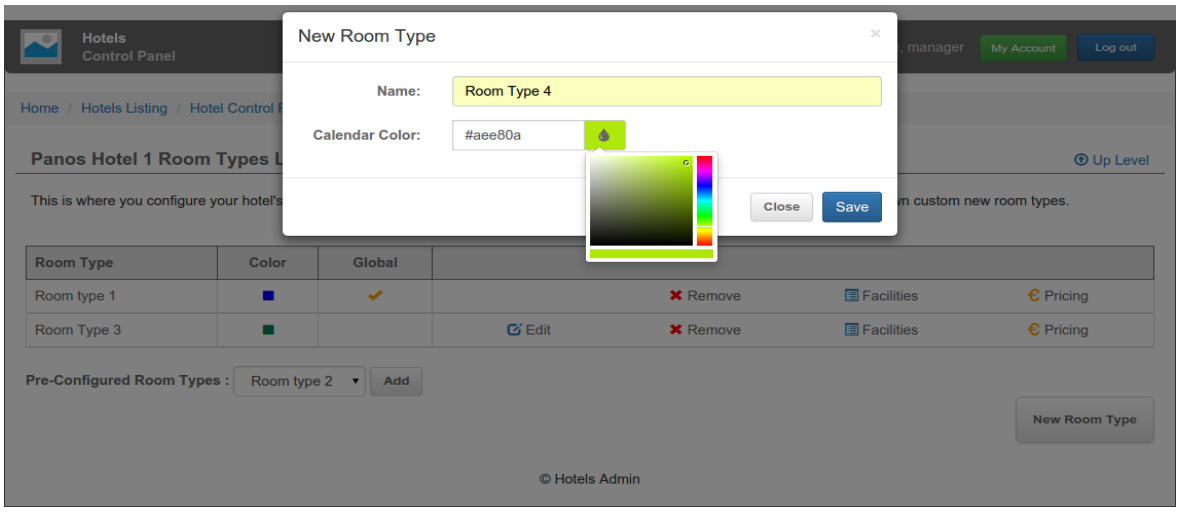
Εικόνα 25 – Παροχές Δωματίου ανά Συγκεκριμένο Τύπο Δωματίου

Το δεύτερο κουμπί της δεύτερης ζώνης (βλέπε εικόνα 11) είναι το “Room Types” και αφορά στην διαχείριση των τύπων δωματίου. Όπως βλέπουμε και στην παρακάτω εικόνα η οθόνη απαρτίζεται από μια λίστα με τους όποιους διαθέσιμους τύπους δωματίου ενώ και εδώ διατηρείται η λογική των προκαθορισμένων και τις δημιουργίας των προσωποποιημένων.



Εικόνα 26 – Λίστα Τύπων Δωματίου

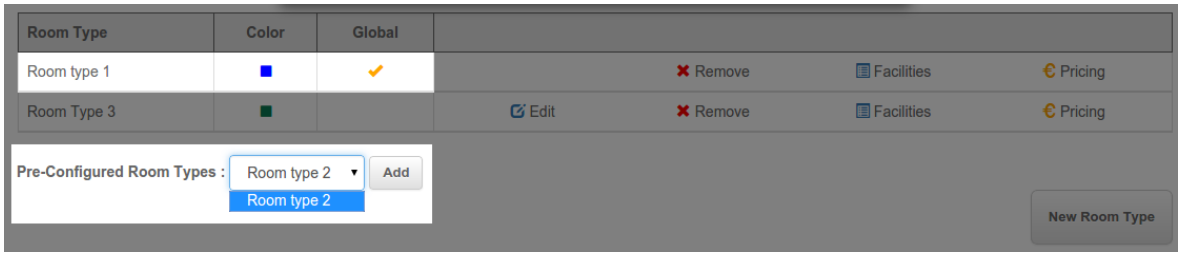
Κατά την δημιουργία ενός νέου τύπου δωματίου, ο χρήστης θα πρέπει να συμπληρώσει στην αναδυόμενη φόρμα το όνομα που επιθυμεί καθώς επίσης και το χρώμα που θέλει να το συνοδεύει στο σύστημα μέσω του επιλογέα χρώματος (color picker).



Εικόνα 27 – Επεξεργασία Τύπου Δωματίου

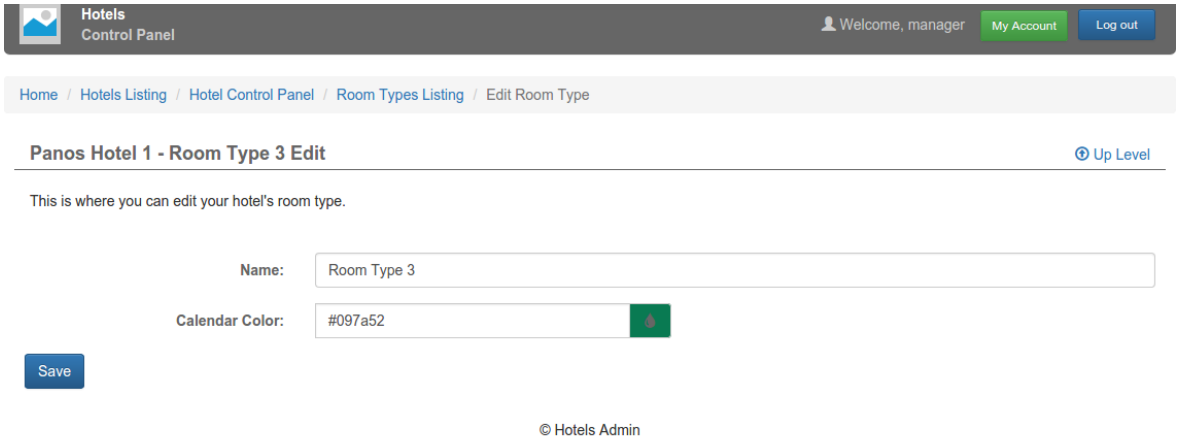
Τον λόγο για τον οποίο συσχετίζεται ένας τύπος δωματίου με ένα χρώμα θα το δούμε πιο καθαρά στις κρατήσεις και μάλιστα στο πως αυτές εμφανίζονται στο ημερολόγιο και πόσο πιο ευανάγνωστο κάνουν την αποτύπωση τους.

Έτσι κάθε μια γραμμή αφορά και ένα τύπο δωματίου, με την στήλη όνομα να αφορά στο όνομα που έχει δοθεί για τον εκάστοτε τύπο, την στήλη χρώμα με το χρώμα που έχει επιλεγεί από τον χρήστη και το κατά πόσο αφορά προεπιλεγμένο τύπο δωματίου ή όχι. Ενώ όπως και στις παροχές έτσι λοιπόν και εδώ υπάρχει η γνωστή επιλογή των προεπιλεγμένων τύπων δωματίου μέσου ενός καταδυόμενου μενού προκειμένου να επιλέξουμε προς συσχέτιση με το ξενοδοχείο μας και βέβαια πατώντας στο κουμπί “Add”.



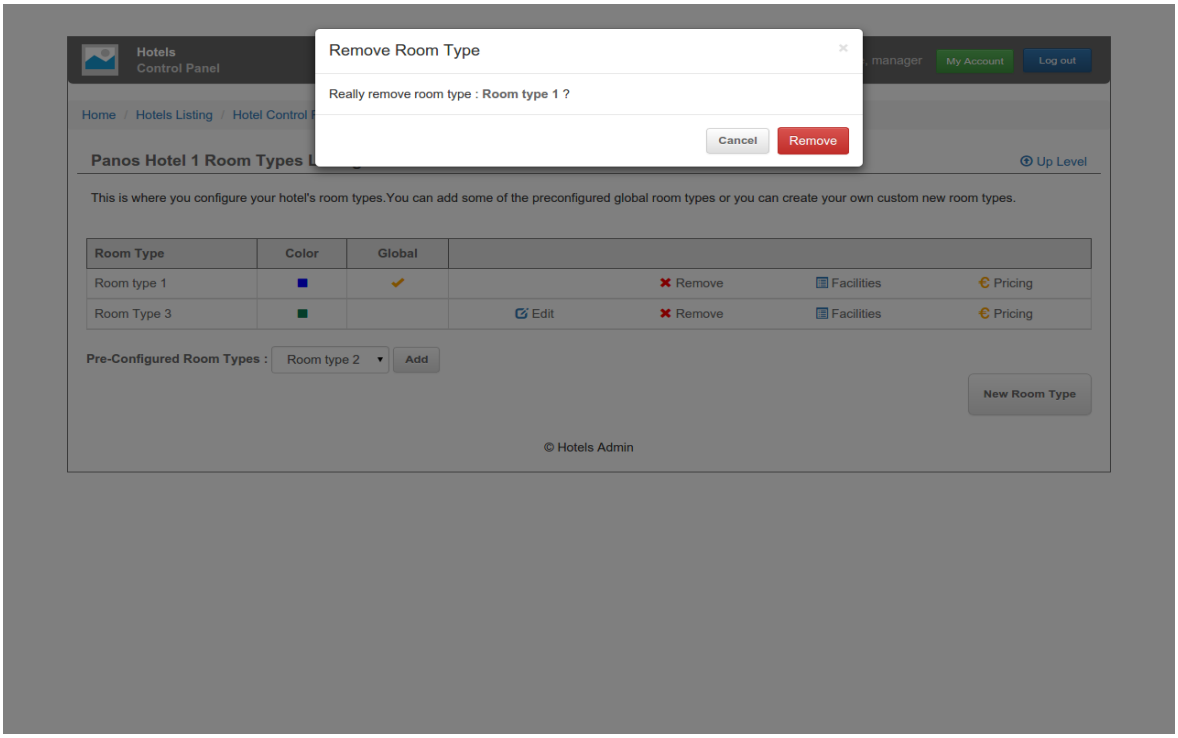
Εικόνα 28 – Προσθήκη Προεπιλεγμένου Τύπου Δωματίου

Η οθόνη που αφορά στην επεξεργασία ενός τύπου δωματίου, είναι παρόμοια με εκείνη της δημιουργίας ενώ το χρώμα που έχει επιλεγεί εμφανίζεται στο κουμπί του επιλογέα χρώματος δίπλα από τον χρωματικό κώδικα που ο ίδιος είχε παράγει.



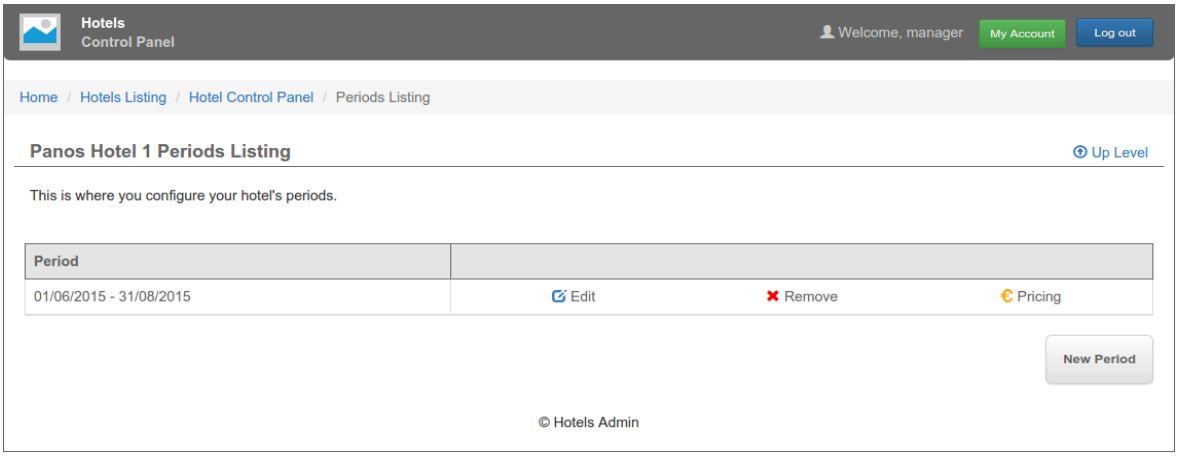
Εικόνα 29 - Επεξεργασία Τύπου Δωματίου

Στην επόμενη εικόνα, βλέπουμε το αναδυόμενο παράθυρο προειδοποίησης, στην περίπτωση που ο χρήστης επίλεξε την διαγραφή ενός τύπου δωματίου.



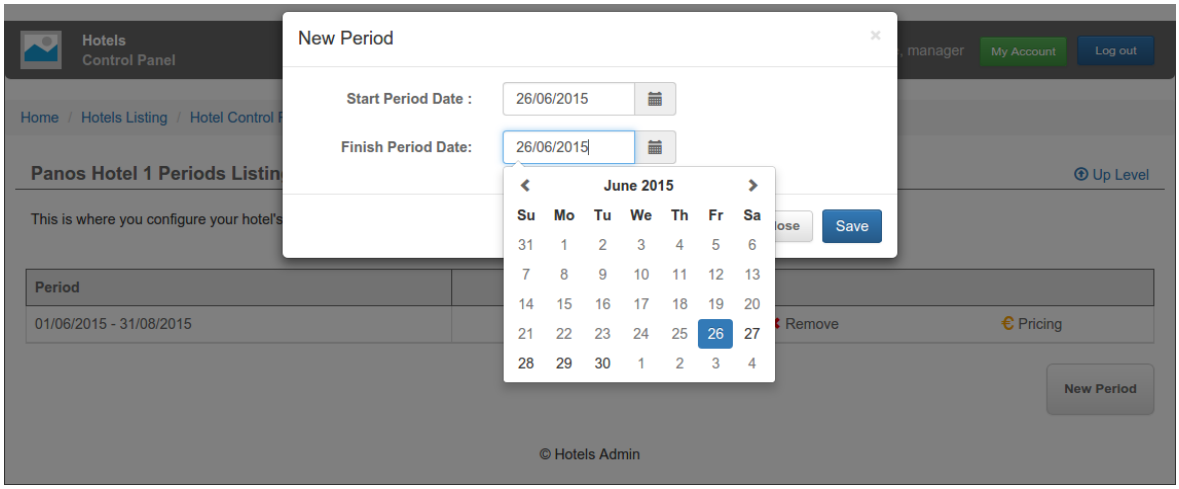
Εικόνα 30 – Διαγραφή Τύπου Δωματίου

Το τρίτο κουμπί της δεύτερης ζώνης (βλέπε εικόνα 11) είναι το “Periods ” και αφορά στην διαχείριση των περιόδων. Όπως βλέπουμε και στην παρακάτω εικόνα η οθόνη απαρτίζεται από μια λίστα με τις όποιες διαθέσιμες περιόδους.



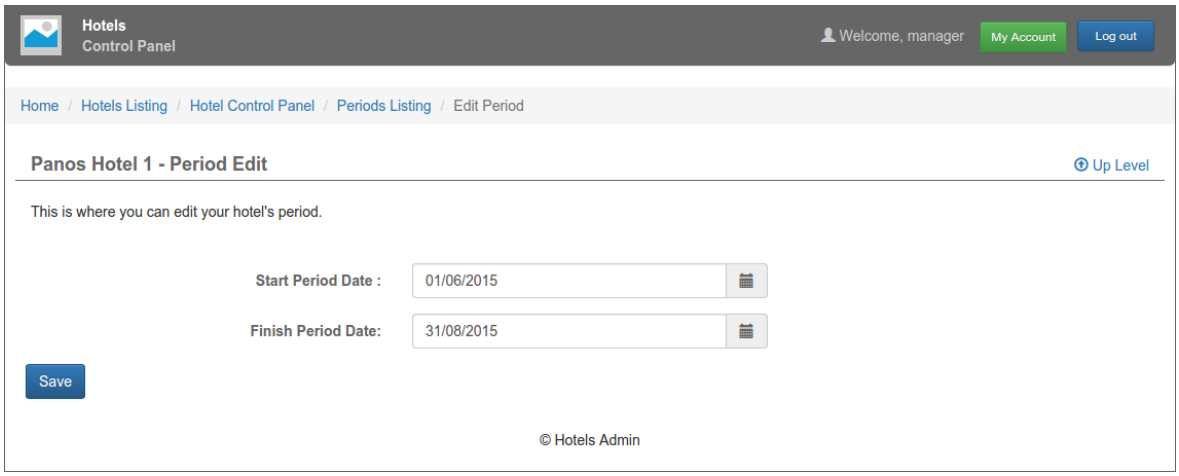
Εικόνα 31 – Λίστα Περιόδων

Πατώντας το κουμπί "New Period" ο χρήστης θα δει ένα αναδυόμενο παράθυρο με μια φόρμα για την υποβολή νέας περιόδου. Η φόρμα αποτελείτε από δύο πεδία την ημερομηνία εκκίνησης της περιόδου και την ημερομηνία τερματισμού της, ενώ προς ευκολία του χρήστη πατώντας στο εικονιδιάκη του ημερολογίου δεξιά των εκάστοτε πεδίων εμφανίζεται ένας επιλογέας ημερομηνίας με την μορφή ημερολογίου. Να σημειώσουμε ότι το σύστημα δεν επιτρέπει ο ημερομηνία τερματισμού να είναι πριν από την ημερομηνία εκίνησης.



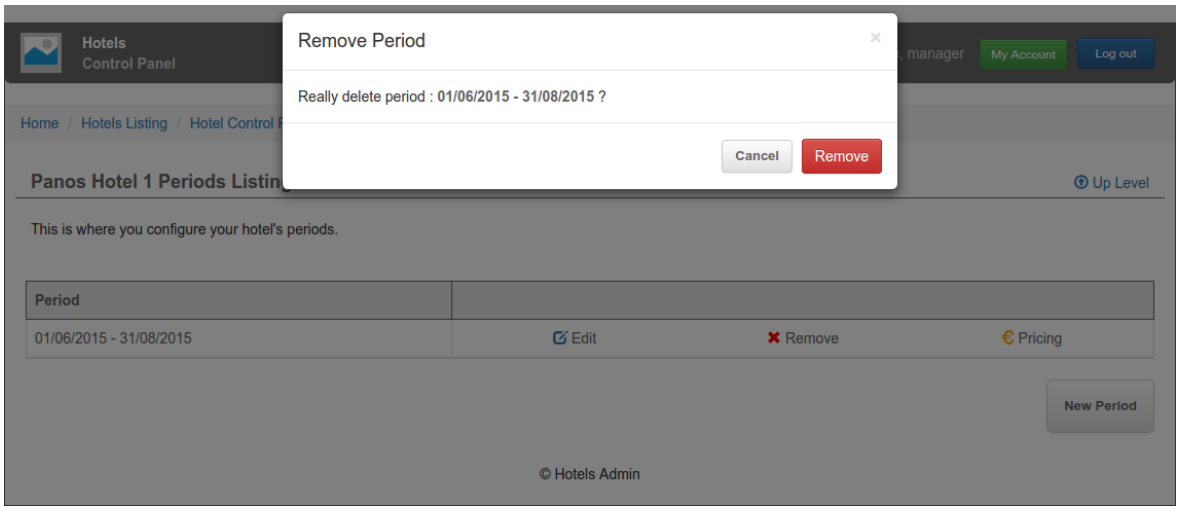
Εικόνα 32 – Δημιουργία Νέας Περιόδου

Ο χρήστης πατώντας τον σύνδεσμο "Edit" στην εικόνα 31 για την αντίστοιχη περίοδο, οδηγείται στην οθόνη επεξεργασίας της.



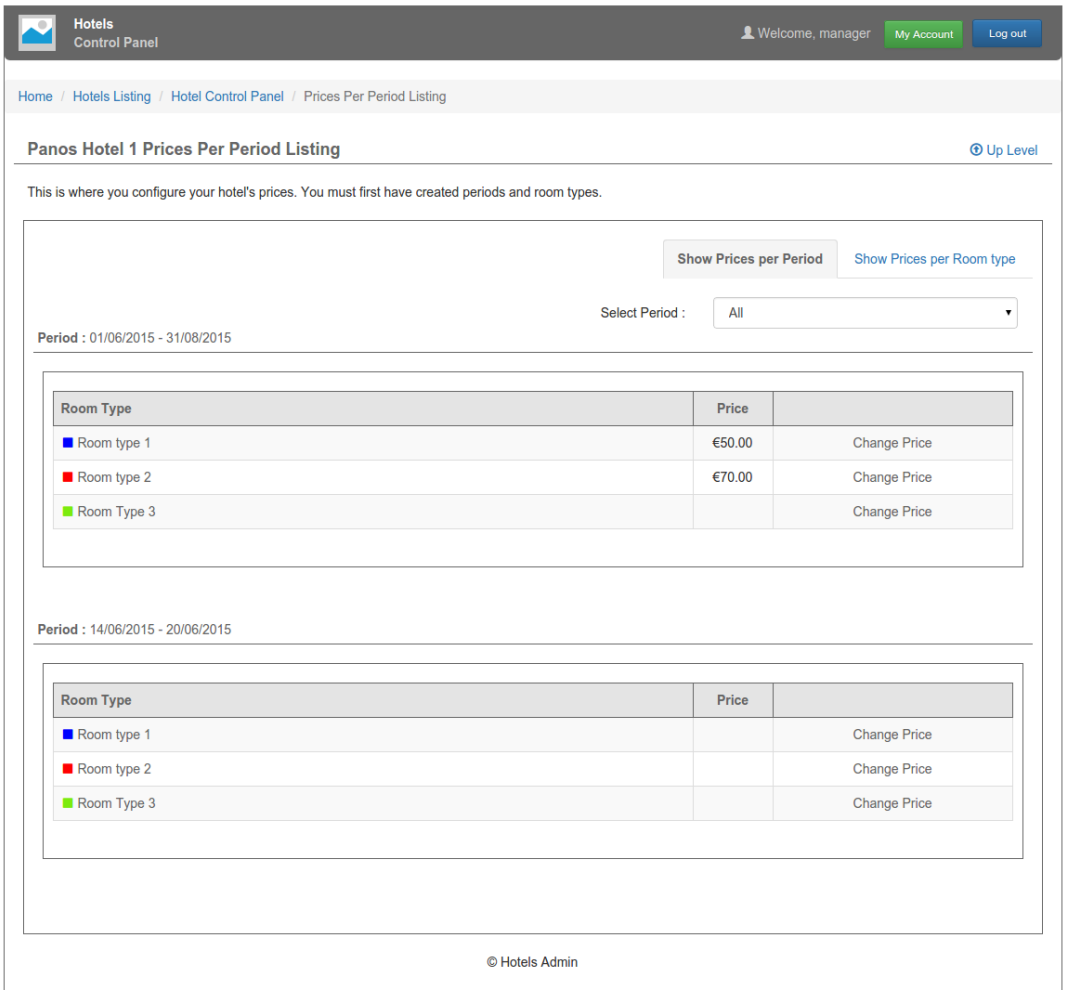
Εικόνα 33 – Επεξεργασία Περιόδου

Στην επόμενη εικόνα, βλέπουμε το αναδυόμενο παράθυρο προειδοποίησης, στην περίπτωση που ο χρήστης επίλεξε την διαγραφή μιας περιόδου.



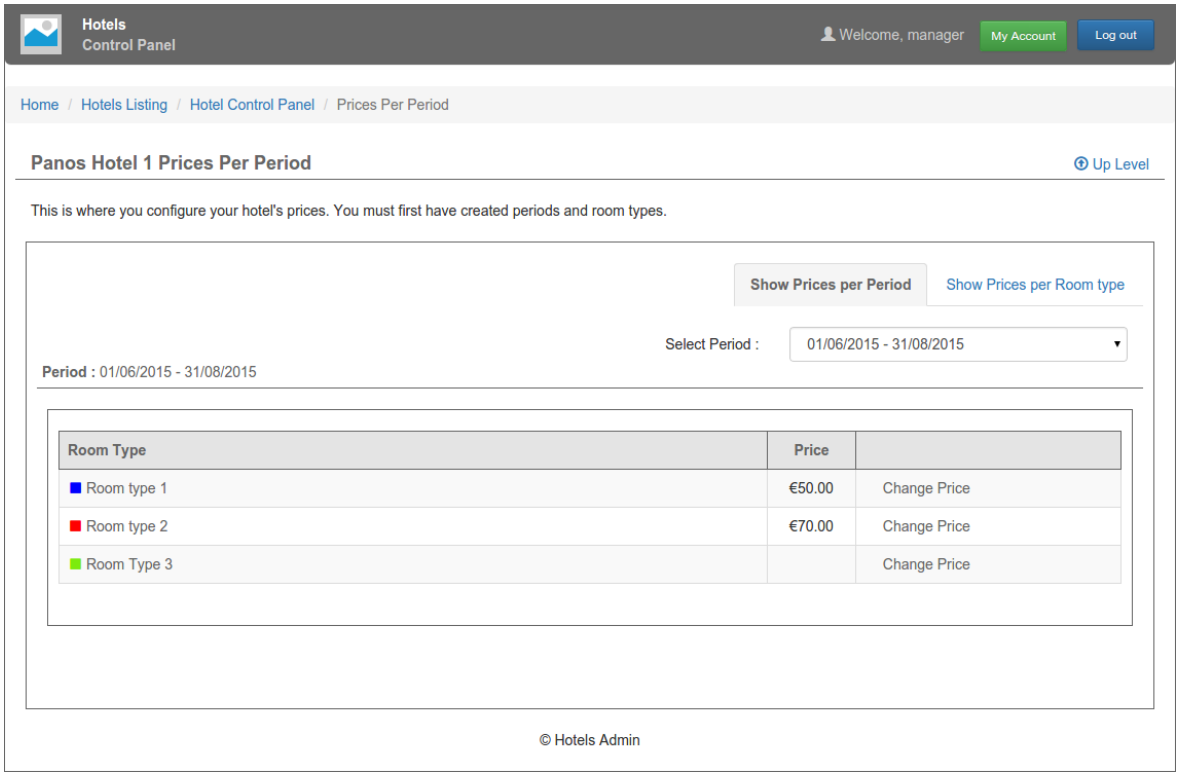
Εικόνα 34 – Διαγραφή Περιόδου

Το τέταρτο κουμπί της δεύτερης ζώνης (βλέπε εικόνα 11) είναι το “Prices” και αφορά στην διαχείριση των τιμών. Όπως βλέπουμε και στην παρακάτω εικόνα η οθόνη απαρτίζεται από μια λίστα με τις όποιες διαθέσιμες τιμές ανά περίοδο και τύπο δωματίου για όλες τις περιόδους. Αυτή είναι και η επιλεγμένη διάταξη όταν ερχόμαστε σε αυτή την οθόνη, κάτι που άλλωστε φαίνεαι και από το επιλεγμένο κουμπί (tab) στο πάνω μέρος “Show Prices per Period”. Να σημειώσουμε ότι στην παρακάτω οθόνη, ο χρήστης μπορεί να οδηγηθεί ακόμη και από τον σύνδεσμο “Pricing” όπως φαίνεται στην εικόνα 31.



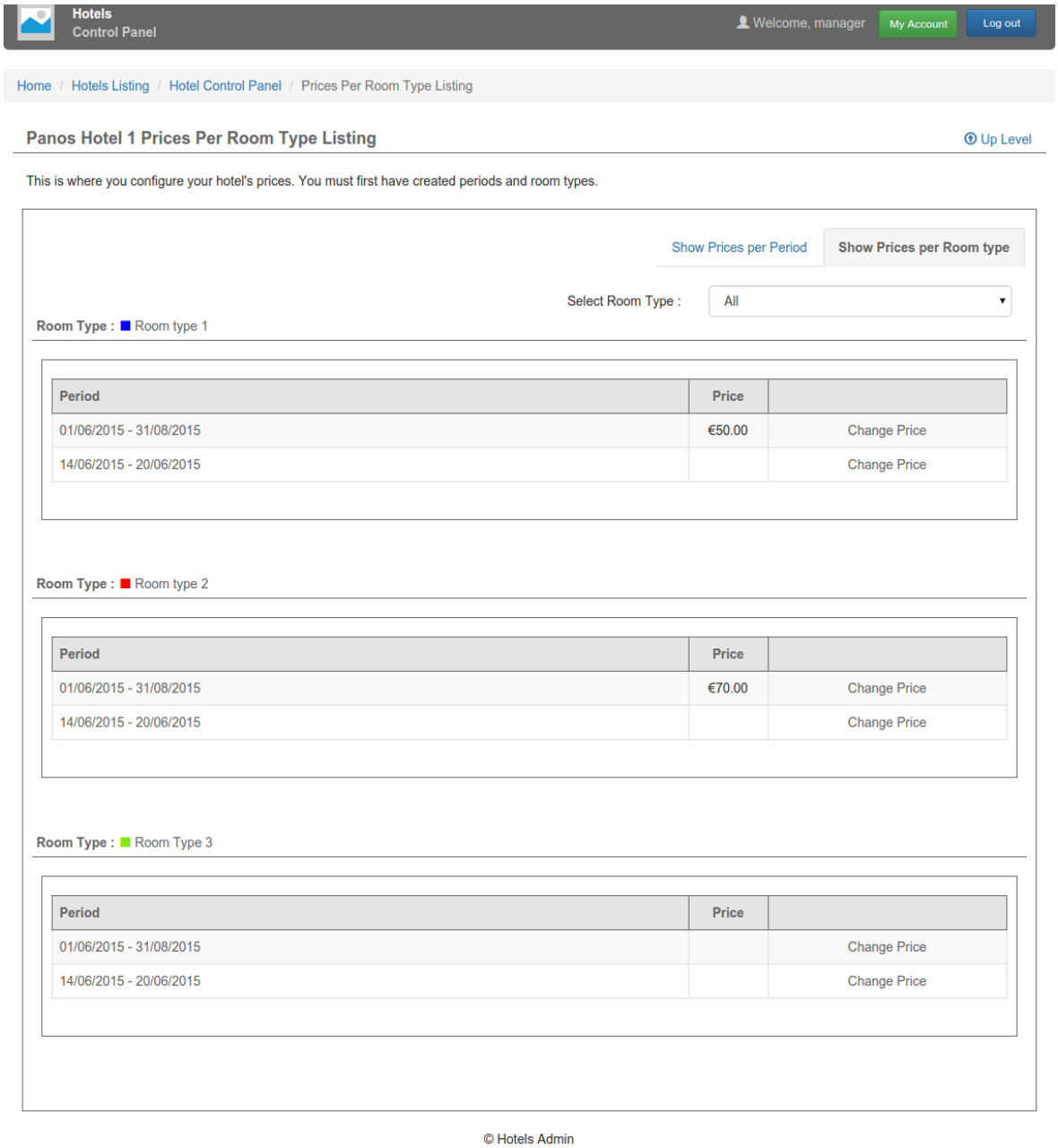
Εικόνα 35 – Εμφάνιση Λίστα Τιμών ανά Περιόδους

Αν ο χρήστης το επιθυμεί μπορεί να επιλέξει μια συγκεκριμένη περίοδο από το καταδυόμενο μενού (Drop Down Menu) “Select Period”. Παρακάτω εμφανίζονται σε λίστα οι όποιες καταχωρημένες τιμές ανά τύπου δωματίο για την επιλεγμένη περίοδο.



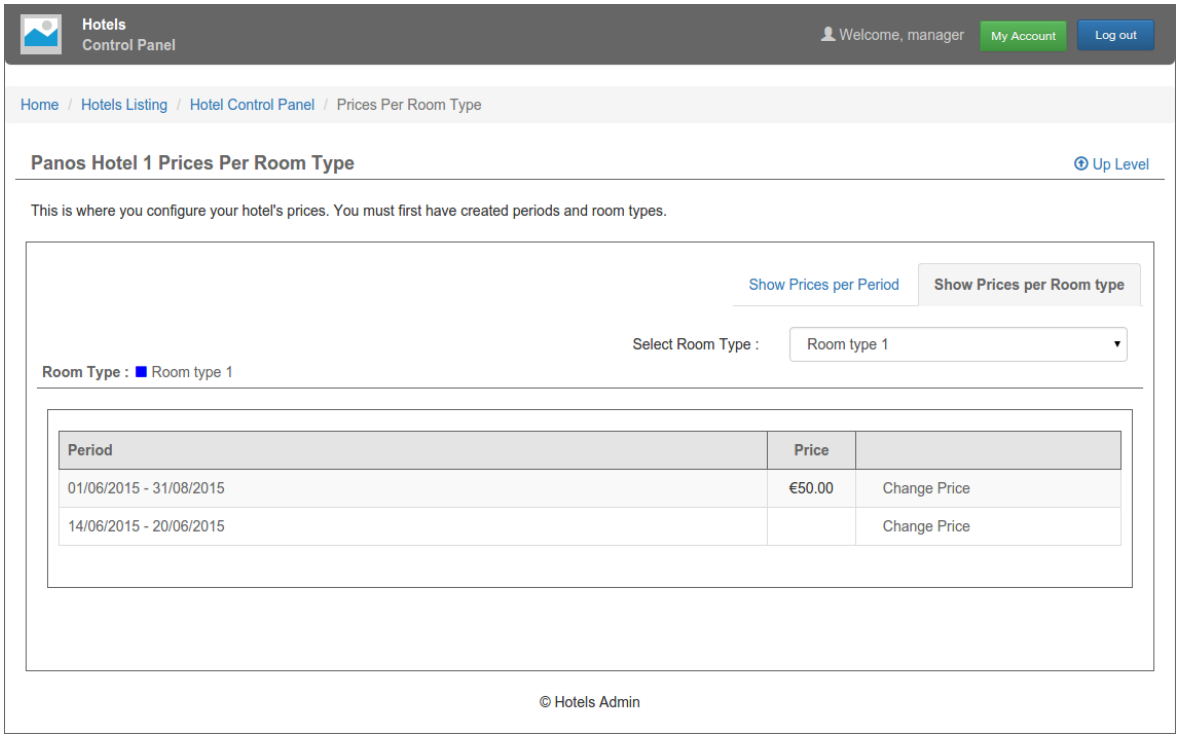
Εικόνα 36 – Εμφάνιση Λίστας Τιμών ανά Συγκεκριμένη Περίοδο

Ο χρήστης έχει την δυνατότητα επιλέγοντας το κουμπί “Show Prices per Room Type” να δει την ίδια πληροφορία προβιβάζοντας ως πρωτεύον έννοια τον τύπο δωματίου. Έτσι στην παρακάτω εικόνα η οθόνη απαρτίζεται από μια λίστα με τις όποιες διαθέσιμες τιμές ανά τύπο δωματίου και περίοδο για όλους τους τύπους δωματίων. Αυτή είναι η δευτερεύουσα διάταξη και μπορεί να διαπιστωθεί από το επιλεγμένο κουμπί (tab) στο πάνω μέρος “Show Prices per Room Type”.



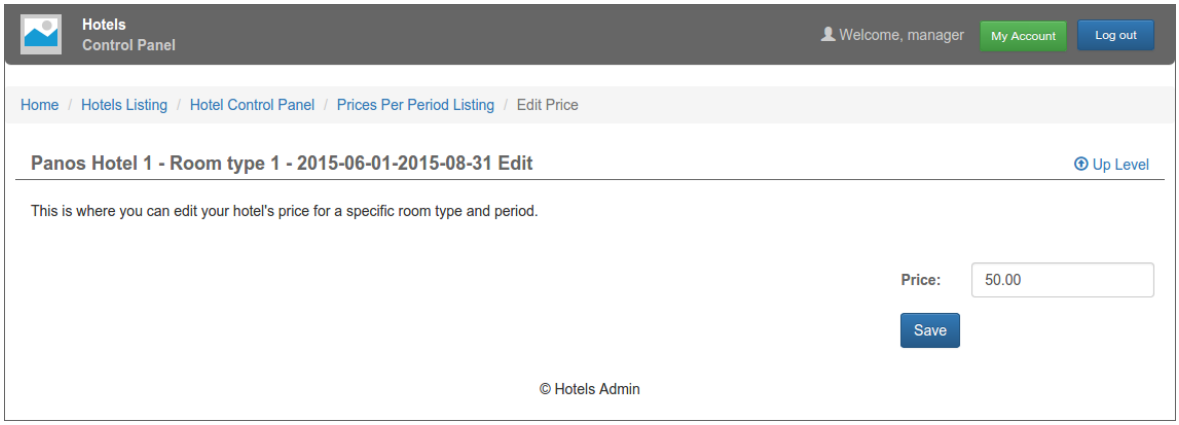
Εικόνα 37 – Εμφάνιση Λίστας Τιμών ανά Τύπο Δωματίου

Στην ίδια λογική με πριν, ο χρήστης μπορεί να επιλέξει συγκεκριμένο τύπο δωματίου από το καταδυόμενο μενού (Drop Down Menu) “Select Room Type”. Παρακάτω εμφανίζονται σε λίστα οι όποιες καταχωρημένες τιμές ανά περίοδο για τον επιλεγμένο τύπο δωματίου.



Εικόνα 38 – Εμφάνιση Λίστας Τιμών ανά Συγκεκριμένο Τύπο Δωματίου

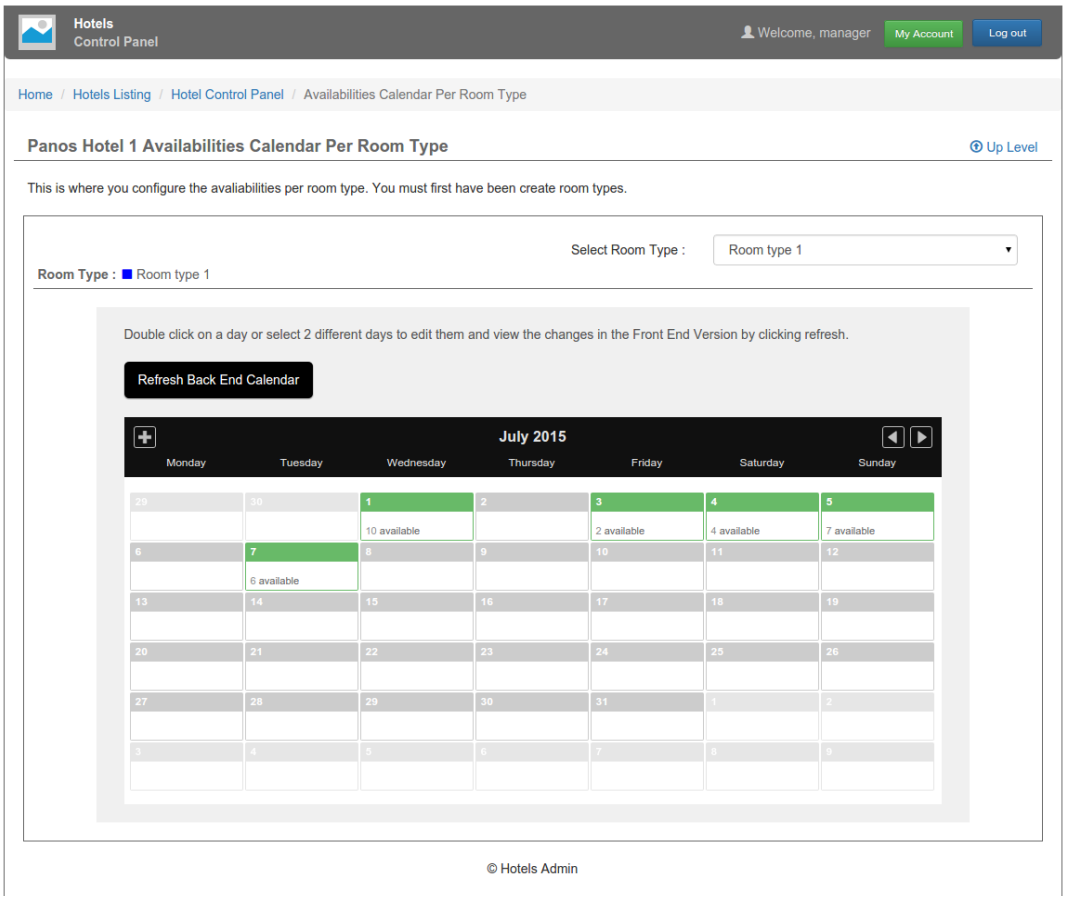
Ο χρήστης πατώντας στο σύνδεσμο “Change Price” όπως εμφανίζεται στις παραπάνω εικόνες, θα οδηγηθεί στην οθόνη επεξεργασίας της τιμής για συγκεκριμένη περίοδο και τύπο δωματίου.



Εικόνα 39 – Επεξεργασία Τιμής ανά Περίοδο και Τύπο Δωματίου

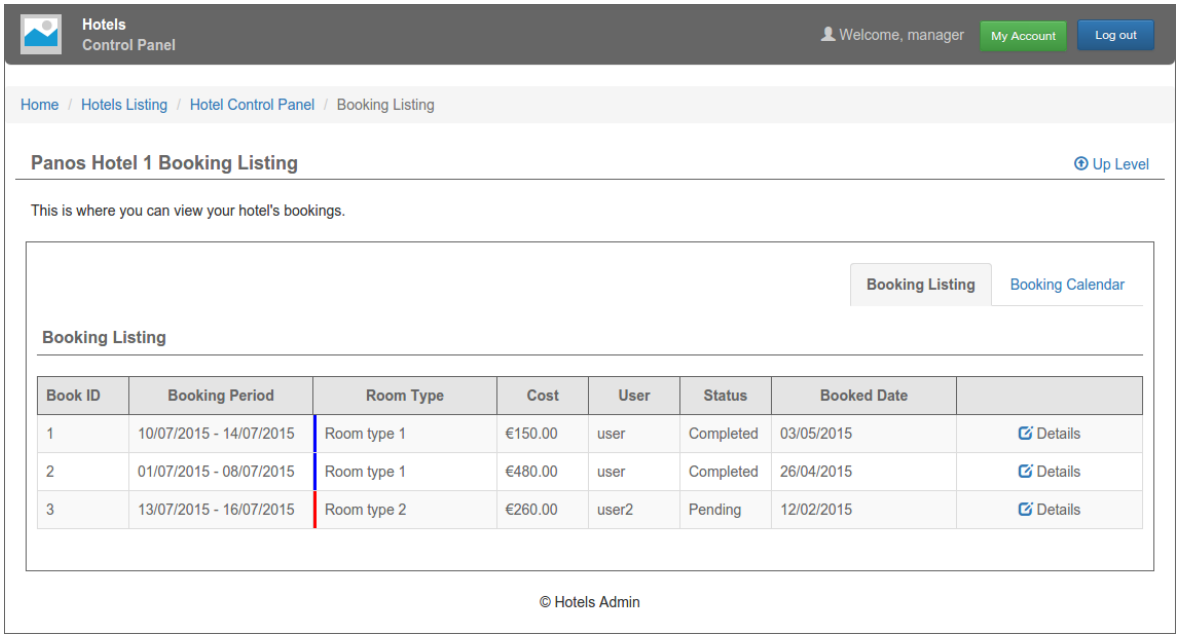
Το πέμπτο κουμπί της δεύτερης ζώνης (βλέπε εικόνα 11) είναι το “Availability” και αφορά στην διαχείριση των διαθεσιμοτήτων των δωματίων ανά τύπο δωματίου. Όπως βλέπουμε και στην παρακάτω εικόνα στην οθόνη, για δεδομένο τύπου δωματίου π.χ. “Room Type 1”, εμφανίζεται ένα διαδραστικό ημερολόγιο που εμφανίζει τις διαθεσιμότητες που έχουν τεθεί για την εκάστοτε ημερομηνία.

Ο χρήστης είναι σε θέση κάνοντας κλικ σε μια ημερομήνια να θέτει την ημερομηνία εκκίνησης και με δεύτερο κλικ την ημερομηνία τερματισμού προκειμένου να ορίσει ή να τροποποιήση/διαγράψει την διαθεσιμότητα δωματίων για τον συγκεκριμένο τύπο δωματίου. Αξίζει να σημειωθεί ότι αν θέλει να ορίσει μια συγκεκριμένη ημερομηνία αρκεί να κάνει κλικ δύο φορές στην ίδια. Επιπλέον με το κουμπί + μπορεί να δει και τους επόμενους μήνες αφού εμφανίζεται κάτω από το υπάρχον νέο ημερολόγιο του επόμενου μήνα. Εναλακτικά μπορεί να πλοηγηθεί με τα βελάκια (αριστερό δεξί) στο πάνω δεξιά μέρος του ημερολογίου. Τόσο τα αποτελέσματα όσο και οι καταχωρήσεις γίνονται με ασύγχρονη κλήση μέσω Rest και Json.



Εικόνα 40 – Διαθεσιμότητα ανά Τύπο Δωματίου

Το πρώτο κουμπί της τρίτης ζώνης Billing (βλέπε εικόνα 11) είναι το “Booking” και αφορά στην διαχείριση των κρατήσεων που έχουν λάβει χώρα για το συγκεκριμένο ξενοδοχείο του χρήστη. Όπως βλέπουμε και στην παρακάτω εικόνα στην οθόνη, η προκαθορισμένη εμφάνιση,όπως φαίνεται και από το επιλεγμένο μενού (tab) “Booking Listing”, είναι μια λίστα με τα γενικά στοιχεία της εκάστοτε κράτησης.



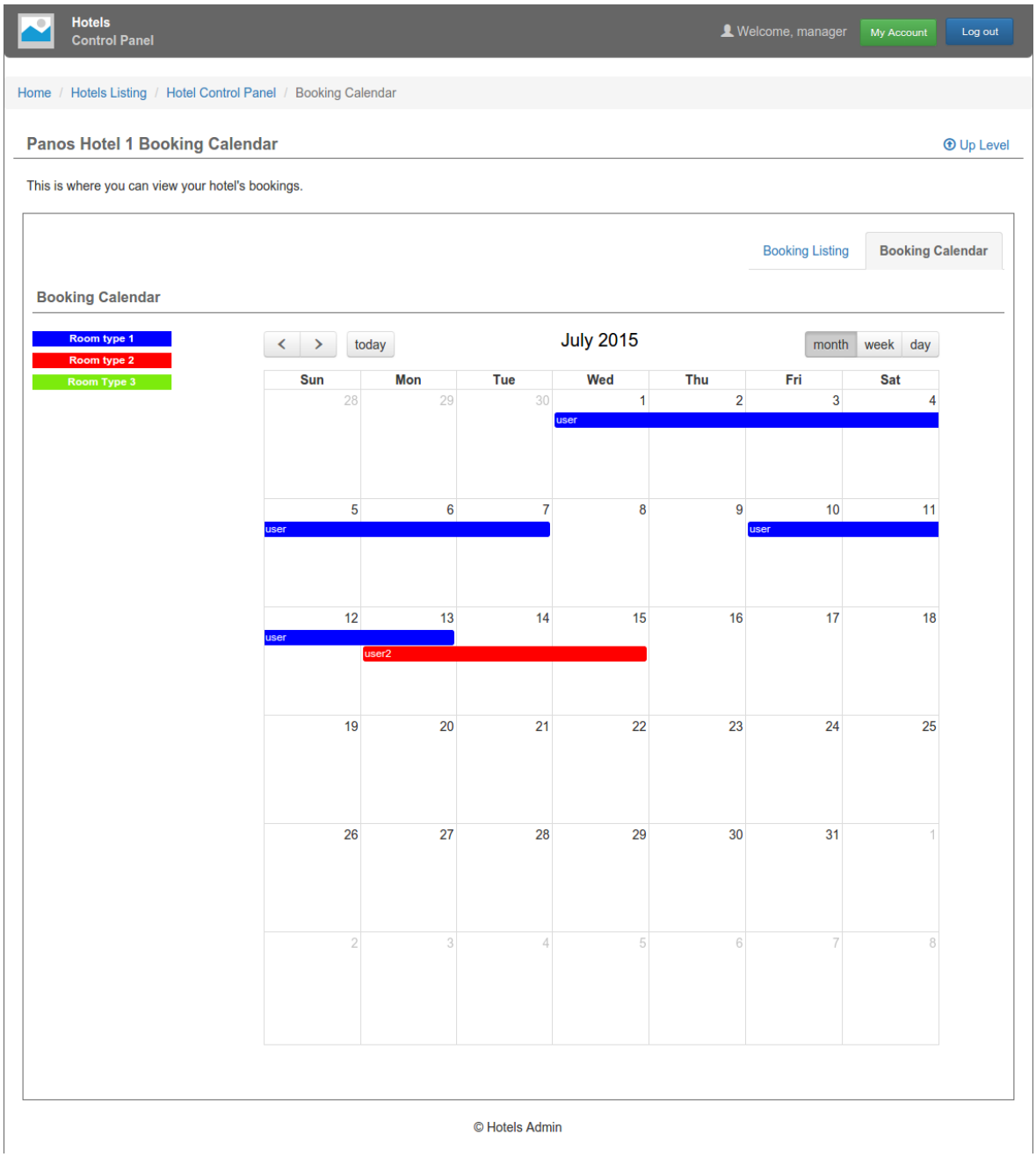
Εικόνα 41 - Εμφάνιση Λίστας Κρατήσεων

Εναλλακτικά ο χρήστης μπορεί να επιλέξει το μενού “Booking Calendar” και να έχει μια εποπτική εικόνα των κρατήσεων πάνω σε ένα ημερολόγιο.

Είναι προφανές ότι τα χρώματα που είχαν καταχωρηθεί κατά την δημιουργία του εκάστοτε τύπου δωματίου βοηθούν στην εύκολη κατανόηση του χαρακτηριστικού των κρατήσεων που σχετίζεται με τον τύπο δωματίου, δεδομένης της λίστας στα αριστερά που αντιστοιχεί το όνομα με το χρώμα.

Ομοίως με το προηγούμενο ημερολόγιο, ο χρήστης έχει την δυνατότητα με τα βελάκια στο πάνω μέρος αριστερά του ημερολογίου, να εμφανίσει προηγούμενες ή επόμενους μήνες.

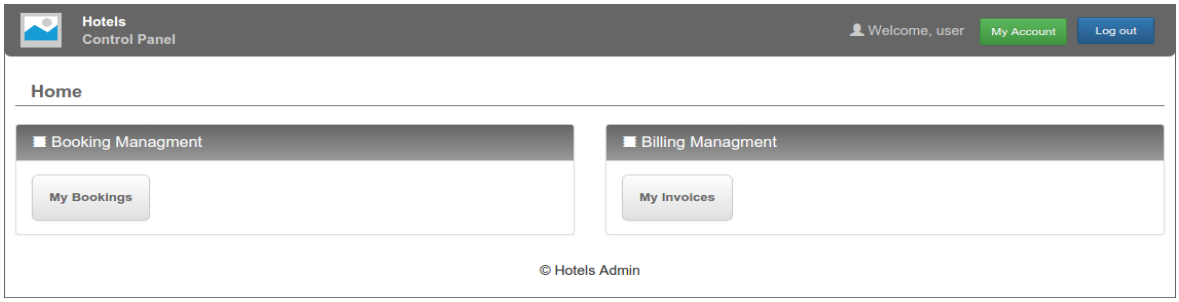
Τέλος πάνω στο ημερολόγιο και τις χρωματικές μπάρες που αντιπροσωπεύουν την περιοδο της εκάστοτε κράτησης καταγράφεται και το όνομα του χρήστη που την υπέβαλε, ενώ το όνομα του λειτουργει ως σύνδεσμος στην κράτηση αυτή καθ' αυτή για περισσότερες πληροφορίες.



Εικόνα 42 – Ημερολόγιο Κρατήσεων

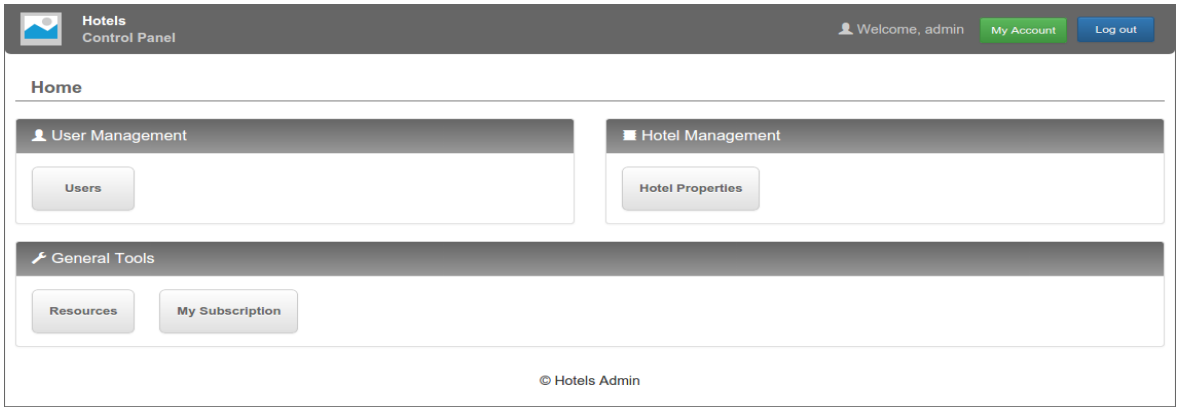
Στις δύο τελευταίες εικόνες μπορούμε να δούμε πως το σύστημα μπορεί να υποστηρίξει διαφορέτικούς πίνακες ελέγχου ανά ρόλο. Έτσι στην παρακάτω εικόνα μπορούμε να δούμε τον πίνακα ελέγχου του χρήστη ρόλο απλού χρήστη/ταξιδιώτη/πελάτη (εικόνα 43), ενώ στην δεύτερη τον πίνακα ελέγχου του χρήστη με ρόλο διαχειριστή (εικόνα 44).

Ο μεν πρώτος έχει δύο ζώνες με ένα κουμπί έκαστη, σε αυτή την φάση, για την διαχείριση των κρατήσεων και των τιμολογίων του.



Εικόνα 43 – Πίνακας Ελέγχου Χρήστη

Ο δε, δεύτερος έχει τρεις ζώνες μέσω τον οποίων μπορεί να δει και να διαχειριστή τους χρήστες, τις προκαθορισμένες επιλογές για τα ξενοδοχεία, όπως αναφερθήκαμε παραπάνω, τους πόρους του συστήματος καθώς επίσης και τις συνδρομές των ξενοδόχων στην πλατφόρμα. Να σημειώσουμε ότι αυτά δεν είναι σε αυτή την φάση λειτουργικά και ο σκόπος είναι απλά η εμφάνιση της δυνατότητας, όπως ήδη αναφέραμε, της τροποποίησης της συμπεριφοράς του συστήματος ανά ρόλο για παράδειγμα στην εμφάνιση των διαθέσιμων επιλογών των πινάκων ελέγχου.



Εικόνα 44 – Πίνακας Ελέγχου Διαχειριστή

6. ΤΕΧΝΙΚΑ – ΑΡΧΙΤΕΚΤΟΝΙΚΑ ΣΤΟΙΧΕΙΑ

6.1 ΕΠΙΛΟΓΕΣ ΥΛΟΠΟΙΗΣΗΣ

Το βασικό υποσύστημα του Spring συμπεριλαμβάνει την έκχυση εξαρτήσεων(Dependency Injection) την αντιστροφή ροής (Invertion of Control) αλλά και την χρήση των Design Patterns που αποτελούν το Model View Controller (MVC).

Έτσι στο λογικό επίπεδο η εφαρμογή διατηρεί την λογική MVC. Με την λογική αυτή, διαχωρίζεται πλήρως το κομμάτι της εμφάνισης, από το λειτουργικό κομμάτι, και από το επίπεδο επικοινωνιών με τη βάση δεδομένων και τις υποδομές καταλόγου. Για την υλοποίηση αυτού του διαχωρισμού χρησιμοποιήθηκαν τα κατάλληλα annotations.

Εκτός απ' το "Spring Core" έγινε χρήση του "Spring Data" για την διασύνδεση με την βάση αλλά και του "Spring Security" για την θωράκιση της εφαρμογής από μη εξουσιοδοτημένη χρήση.

Επιπλέον έγινε χρήση μιας σειράς βιβλιοθηκών που διευκολύνουν την διεκπεραίωση των αναγκών της εφαρμογής. Αυτές οι βιβλιοθήκες παρότι αποτελούν σημαντικό κομμάτι της εφαρμογής, υλοποιούν μόνο ένα περιφερειακό τμήμα της. Μερικές από αυτές, σε ότι αφορά το backend, είναι η "Joda" library για την διαχείριση των ημερομηνιών και των ημερολογίων, ενώ σε επίπεδο client χρησιμοποιήθηκε η βασική βιβλιοθήκη της javascript "JQuery" αλλά και διάφορες επεκτάσεις της όπως ο επιλογέας χρώματος (color picker), ή ο επιλογέας ημερομηνίας (Calendar).

Σε επίπεδο διασύνδεσης με την βάση χρησιμοποιήθηκε τεχνολογία αντιστοίχησης αντικειμένων δεδομένων (ORM – Object Relational Mapping), μέσω JPA Annotation και με την χρήση του hibernate ως υλοποιητή (implementer).

Να σημειώσουμε ότι έγινε χρήση του Maven ως εργαλείο εξαρτήσεων και αυτοματοποίησης για την ευκολότερη διαχείριση των απαραίτητων επεκτάσεων/βιβλιοθηκών της java (jars) αλλά και για την διαδικασία του deployment, ενώ για το υποσύστημα της καταγραφής (logging) έγινε χρήση του log4j.

Εκτός από τις παραπάνω βιβλιοθήκες έχει χρησιμοποιηθεί μια τοπική βάση δεδομένων.

Κατά την μελέτη της εφαρμογής επιλέχθηκε να χρησιμοποιηθεί μια τοπική βάση, όσο το δυνατόν πιο ελαφρού αποτυπώματος. Στόχος της βάσης είναι η αποθήκευση των δεδομένων που πρέπει να διατηρούνται (persist) σε σχέση με τα χαρακτηριστικά των ξενοδοχείων όπως για παράδειγμα τις τιμές, τους τύπους δωματίων, τα χαρακτηριστικά τους, τις κρατήσεις. Η επιλογή της τοπικής βάσης έχει σαν αποτέλεσμα και τη δυνατότητα για εγκατάσταση της εφαρμογής σε έναν εξυπηρετητή, χωρίς την χρήση διεπαφής με εξωτερικούς εξυπηρετητές όπως τρίτα συστήματα ίδιου τύπου. Επιπλέον κρίθηκε πως δεν χρειάζεται η ύπαρξη ενός πολύπλοκου σχήματος, αλλά περισσότερο μια απλή δομή για την αποτύπωση των ανωτέρων.

Για τους παραπάνω λόγους επιλέχθηκε η χρήση της MySql. Η εγκατάσταση της γίνεται παράλληλα με αυτήν της εφαρμογής και αποτελεί απαραίτητη προϋπόθεση για την λειτουργία της.

Τέλος πρέπει να αναφερθεί και η χρήση του Bootstrap Framework. Η χρήση της έγινε με στόχο την καλή παρουσίαση αλλά και ευκολία στο επίπεδο της προβολής (view layer). Επιπλέον έγινε χρήση τόσο του υποσυστήματος της javascript για τα modals όσο και των εικόνων που διαθέτει.

6.2 ΑΡΧΙΤΕΚΤΟΝΙΚΗ-ΔΙΑΛΕΙΤΟΥΡΓΙΚΟΤΗΤΑ

Σε φυσικό επίπεδο, ολόκληρη η εφαρμογή τοποθετείται σε μια εικονική μηχανή καθώς δεν απαιτεί μεγαλύτερη ποσότητα πόρων. Στην παρούσα φάση έχει υλοποιηθεί το διαχειριστικό μέρος του συστήματος, ενώ υπάρχει ήδη η υποδομή για πολλαπλά επίπεδα βάση τριών βασικών ρόλων (πελάτης, κάτοχος ξενοδοχείου/ων, διαχειριστής). Η λειτουργία απαιτεί την συνδεσιμότητα με ένα εσωτερικό σύστημα. Πρόκειται για αυτό της βάση δεδομένων , το οποίο αποτελεί το πρωτογενές σύστημα παροχής δεδομένων.

Το υποσύστημα της εμφάνισης των αποτελεσμάτων ανάλογα με την διαθεσιμότητα ανά τύπο δωματίου και περιοχή μπορεί είτε να υλοποιηθεί ως συνέχεια του υπάρχον συστήματος, είτε ως εξωτερικό σύστημα μέσω σύνδεσης με το υπάρχον.

Στο μέλλον θα υποστηρίζεται η υπηρεσία διαλειτουργικότητας με ανάλογα συστήματα μέσω web services και η αποδέσμευση της από το εσωτερικό της εφαρμογής. Τα συστήματα αυτά θα είναι είτε παροχείς πρωτόλειας πληροφορίας(producer) για εμφάνιση μέσα στην

υπάρχουσα πλατφόρμα, είτε καταναλωτές της υπάρχουσας πληροφορίας (πρωτογενής και εξωτερική) για άλλα ετερογενή συστήματα αυτού του τύπου. Και τα δυο αυτά συστήματα τοποθετούνται σε εξωτερικά συστήματα.

Τέλος η συνδεσιμότητα προβλέπεται να γίνει με την χρήση κλήσεων Rest και JSON, κάτι που άλλωστε ήδη υποστηρίζεται από το σύστημα.

6.3 ΑΥΘΕΝΤΙΚΟΠΟΙΗΣΗ ΚΑΙ ΕΓΚΡΙΣΗ

Οι χρήστες του συστήματος έχουν την δυνατότητα να αποκτήσουν τον λογαριασμό τους στην πλατφόρμα μέσω του υποσυστήματος της εγγραφής (registration). Ο ρόλος τους αφορά εκείνον του κατόχων επιχειρήσεων διαμονής. Η διαδικασία είναι απλή αφού αρκεί να προσκομίσουν ένα όνομα χρήστη, ένα κωδικό και ένα email, ώστε να είναι σε θέση να κάνουν άμεσα χρήση της πλατφόρμας, δημιουργώντας το ξενοδοχείο τους και εισάγοντας τα στοιχεία που το περιγράφουν. Τόσο η ταυτοποίηση(authentication) όσο και η επικύρωση (verification) ως υπομέρη της διαδικασίας της αυθεντικοποίησης υλοποιείται από το υποσύστημα του Spring Security. Ο αλγόριθμος κρυπτογράφησης ορίζεται σε κατάλληλο configuration file και είναι sha1.

Σε ότι αφορά την έγκριση (authorization) έχουν υλοποιηθεί τρία επίπεδα.

Πελάτη : Πρόκειται για τον χρήστη που μπορεί να εισέλθει στο σύστημα να δει τις υπάρχουσες κρατήσεις του

Ξενοδόχο : Πρόκειται για τον χρήστη που μπορεί να δημιουργήσει ένα λογαριασμό, όπως αναφερθήκαμε παραπάνω, και να καταχωρήσει το δικό του ξενοδοχείο, εισάγοντας τα στοιχεία που το περιγράφουν.

Διαχειριστή : Πρόκειται για τον χρήστη που έχει την δυνατότητα γενικής εποπτείας του συστήματος (κρατήσεις ανά ξενοδόχο, τζίρους, διαχείριση χρηστών κ.τ.λ.)

Ενώ η υλοποίηση των τριών επιπέδων τόσο σε επίπεδο αυθεντικοποίησης και έγκρισης όσο και σε επίπεδο διαφορετικής εμφάνισης διαχειριστικού και δυνατοτήτων, εκκρεμεί για μελλοντική ανάπτυξη η υλοποίηση μέρους του υποσυστήματος κρατήσεων του ξενοδόχου καθώς και σχεδόν το σύνολο των άλλο δύο επιπέδων εκτός από αυτό της διαχείρισης των χρηστών στο επίπεδο του διαχειριστή.

7. ΣΥΜΠΕΡΑΣΜΑΤΑ

Στα πλαίσια αυτής της διπλωματικής εργασίας, αναπτύχθηκε ένα τμήμα ενός πληροφοριακού συστήματος ξενοδοχείων. Πρόκειται για μια διαδικτυακή εφαρμογή πολλαπλών επιπέδων, με σκοπό την δημιουργία μιας κεντροποιημένης πλατφόρμας μέσω της οποίας οι εκάστοτε επιχειρηματίες τουριστικών καταλυμάτων θα μπορούσαν να εγγραφούν και να διαχειριστούν τα ξενοδοχεία τους.

Για την υλοποίηση επιλέχτηκε η γλώσσα Java και τεχνολογίες J2EE, ως η πλέον διαδεδομένη λύση για διαδικτυακές επιχειρηματικές εφαρμογές (web app/enterprise app). Για την καλύτερη και πιο στιβαρή ανάπτυξη επιλέχτηκε το πλέον διαδεδομένο framework “Spring”, το οποίο αποτελεί σήμερα το στάνταρ στην αγορά της ανάπτυξης λογισμικού και ήταν και η κινητήριος δύναμη για τις επιλογές που προαναφέρθηκαν σε επίπεδο απόκτησης γνώσεων και εμπειρίας.

Στην πρώτη φάση, που είναι και αυτή που παρουσιάστηκε, υλοποιήθηκε το βασικό υποσύστημα της καταχώρησης των γενικών χαρακτηριστικών των ξενοδοχείων που αφορά στα απλά στοιχεία του ξενοδοχείου (στοιχεία επικοινωνίας, τύπος ξενοδοχείου/αστέρια), τύπους δωματίων, περίοδοι, διαθεσιμότητες, τιμές ανά τύπο δωματίου, καθώς επίσης και μέρος του υποσυστήματος των κρατήσεων με την εμφάνιση των κρατήσεων που έχουν καταγραφεί τόσο σε πλέγμα όσο και σε κατάλληλο ημερολόγιο μέσω χρωματικών ράβδων ανά τύπου δωματίου,

Μελλοντικές υλοποιήσεις, θα μπορούσε να ήταν η ολοκλήρωση του συστήματος κρατήσεων, δεδομένου ότι σε επίπεδο βάσεως υφίσταται ως υποδομή, αλλά και η ολοκλήρωση των υπολοίπων επιπέδων διαχείρισης, όπως εκείνο του απλού πελάτη και του υπερδιαχειριστή ο οποίος θα πρέπει να έχει μια εποπτική εικόνα όλου του συστήματος. Επιπλέον η ανάπτυξη βελτιστοποιημένου αλγορίθμου για την καταλληλότερη δέσμευση ημερομηνιών ως προς τα διαθέσιμα δωμάτια ανά τύπου δωματίου. Αυτό θα βοηθούσε ώστε να διατηρούνται διαθέσιμα μεγαλύτεροι περίοδοι που μπορεί να απαιτηθούν σε μια επόμενη κράτηση.

8. ΑΝΑΦΟΡΕΣ

Java. https://www.oracle.com/java.

Spring. https://spring.io/.

Spring Data. http://projects.spring.io/spring-data/.

Spring Security. http://projects.spring.io/spring-security/.

JPA. https://en.wikipedia.org/wiki/1410Java_Persistence_API.

JPA Annocations. http://www.techferry.com/articles/hibernate-jpa-annotations.html

Object Oriented Mapping. https://en.wikipedia.org/wiki/Object-relational_mapping.

Hibernate. http://hibernate.org/.

MySql. https://www.mysql.com/.

Apache Tiles. https://tiles.apache.org/.

Log4j. http://logging.apache.org/log4j/2.x/

Bootstrap. http://getbootstrap.com/.

JQuery. https://jquery.com/.

Design Patterns. https://en.wikipedia.org/wiki/Software_design_pattern.

ModelViewController. https://en.wikipedia.org/wiki/Model-view-controller.

Inversion of Control. https://en.wikipedia.org/wiki/Inversion_of_control.

Dependency Injection. https://en.wikipedia.org/wiki/Dependency_injection.

Factory Pattern. https://en.wikipedia.org/wiki/Factory_method_pattern.

Interceptor pattern. https://en.wikipedia.org/wiki/Interceptor_pattern.

Data Modeling. https://en.wikipedia.org/wiki/Data_modeling.

Enchanced Entity Relationship model - Crow's Foot Notation.

https://en.wikipedia.org/wiki/Enhanced_entity-relationship_model.

DAOs / Repositories. https://en.wikipedia.org/wiki/Data_access_object

AJAX. https://en.wikipedia.org/wiki/Ajax_(programming).

Web Services. https://en.wikipedia.org/wiki/Web_service.

JSON. https://en.wikipedia.org/wiki/JSON.

REST. https://en.wikipedia.org/wiki/Representational_state_transfer.

Password Hashing. http://crypto.stanford.edu/PwdHash/

BCrypt. https://en.wikipedia.org/wiki/Bcrypt

Md5. https://en.wikipedia.org/wiki/MD5

SHA-1. https://en.wikipedia.org/wiki/SHA-1

Datasource. http://docs.oracle.com/javase/7/docs/api/javax/sql/DataSource.html

JDBC. https://en.wikipedia.org/wiki/Java_Database_Connectivity

Application Server(Tomcat). https://en.wikipedia.org/wiki/Application_server.

Servlet. https://en.wikipedia.org/wiki/Java_servlet.

JSP. https://en.wikipedia.org/wiki/JavaServer_Pages.

Expression Language. https://docs.oracle.com/javaee/6/tutorial/doc/gjddd.html.

JSTL. http://docs.oracle.com/javaee/5/tutorial/doc/bnake.html.

Maven. https://maven.apache.org/.

RDBMS. https://en.wikipedia.org/wiki/Relational_database_management_system.

Axure (Wireframe Software). http://www.axure.com/.

HTML. http://www.w3schools.com/html/html5_intro.asp.

CSS. http://www.w3.org/Style/CSS/.

Javascript. http://www.w3schools.com/js/.

Spring Tool Suite. https://spring.io/tools/sts/all

Mysql Workbench. https://www.mysql.com/products/workbench/.

Git (Versioning Protocol). http://git-scm.com/book/be/v2/Git-Internals-Transfer-Protocols.

Bitbucket. https://bitbucket.org/.

Joda. http://www.joda.org/joda-time/

Jackson. https://github.com/FasterXML/jackson.

Linux. https://www.linux.com/.

9. ΠΑΡΑΡΤΗΜΑ

9.1 ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΕΦΗΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ

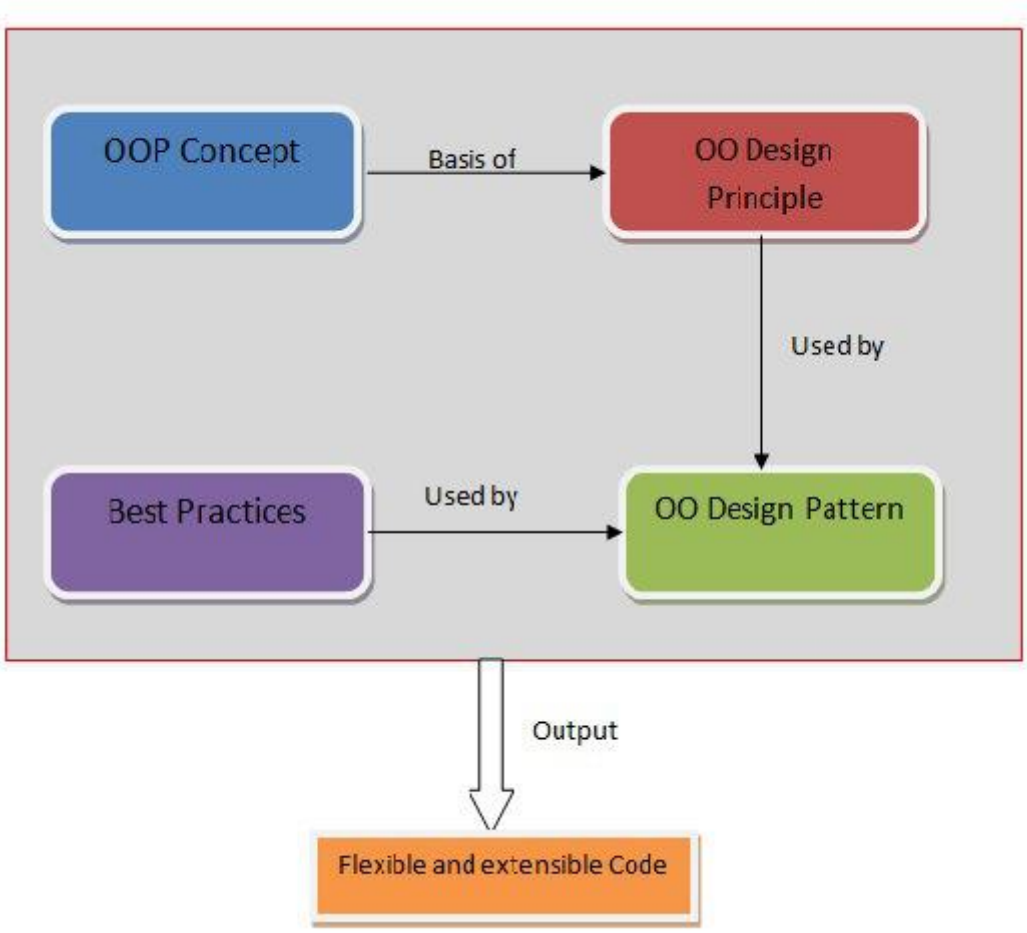
9.1.1 Γενικά

Στην πληροφορική αντικειμενοστρεφή προγραμματισμό (object-oriented programming), ή ΑΠ, ονομάζουμε ένα προγραμματιστικό υπόδειγμα το οποίο εμφανίστηκε στα τέλη της δεκαετίας του 1960 και καθιερώθηκε κατά τη δεκαετία του 1990, αντικαθιστώντας σε μεγάλο βαθμό το παραδοσιακό υπόδειγμα του δομημένου προγραμματισμού. Πρόκειται για μία μεθοδολογία ανάπτυξης προγραμμάτων, υποστηριζόμενη από κατάλληλες γλώσσες προγραμματισμού, όπου ο χειρισμός σχετιζόμενων δεδομένων και των διαδικασιών που επενεργούν σε αυτά γίνεται από κοινού, μέσω μίας δομής δεδομένων που τα περιβάλλει ως αυτόνομη οντότητα με ταυτότητα και δικά της χαρακτηριστικά. Αυτή η δομή δεδομένων καλείται αντικείμενο και αποτελεί πραγματικό στιγμιότυπο στη μνήμη ενός σύνθετου, και πιθανώς οριζόμενου από τον χρήστη, τύπου δεδομένων ονόματι κλάση. Η κλάση προδιαγράφει τόσο δεδομένα όσο και τις διαδικασίες οι οποίες επιδρούν επάνω τους· αυτή υπήρξε η πρωταρχική καινοτομία του ΑΠ.

Έτσι μπορεί να οριστεί μία προδιαγραφή δομής αποθήκευσης (π.χ. μία κλάση «τηλεόραση») η οποία να περιέχει τόσο ιδιότητες (π.χ. μία μεταβλητή «τρέχον κανάλι») όσο και πράξεις ή χειρισμούς επί αυτών των ιδιοτήτων (π.χ. μία διαδικασία «άνοιγμα της τηλεόρασης»). Στο εν λόγω παράδειγμα κάθε υλική τηλεόραση (κάθε αντικείμενο αποθηκευμένο πραγματικά στη μνήμη) αναπαρίσταται ως ξεχωριστό, «φυσικό» στιγμιότυπο αυτής της πρότυπης, ιδεατής κλάσης. Επομένως μόνο τα αντικείμενα καταλαμβάνουν χώρο στη μνήμη του υπολογιστή ενώ οι κλάσεις αποτελούν απλώς «καλούπια». Οι αιτίες που ώθησαν στην ανάπτυξη του ΑΠ ήταν οι ίδιες με αυτές που οδήγησαν στην ανάπτυξη του δομημένου προγραμματισμού (ευκολία συντήρησης, οργάνωσης, χειρισμού και επαναχρησιμοποίησης κώδικα μεγάλων και πολύπλοκων εφαρμογών), όμως τελικώς η αντικειμενοστρέφεια επικράτησε καθώς μπορούσε να αντεπεξέλθει σε προγράμματα πολύ μεγαλύτερου όγκου και πολυπλοκότητας.

9.1.2 Επίτευξη Βέλτιστης Προγραμματιστικής Σχεδίασης

Η java ως γλώσσα, δεδομένου ότι είναι αντικειμενοστρεφήςμ, εμπεριέχει τις βασικές έννοιες του αντικειμενοστρεφούς προγραμματισμού. Αυτό είναι η αρχή για την καλή σχεδίαση αλλά δεν είναι το μόνο που απαιτείται. Έτσι θα μπορούσαμε γενικά να πούμε ότι η πορεία προς τον καλό σχεδιασμό ακολουθεί την εξής πορεία όπως φαίνεται και στην παρακάτω εικόνα.



Εικόνα 45 – Αρχιτεκτονική Ροή Βελτιστοποίησης Σχεδιασμού

Όπως είναι φανερό, η βάση είναι οι τέσσερις βασικές έννοιες του αντικειμενοστραφούς προγραμματισμού.

Στην συνέχεια σε αυτές βασίζονται οι αντικειμενοστρεφής αρχές σχεδιασμού, που είναι υποσύνολο των αρχών σχεδιασμού ανεξαρτήτου αντικειμενοστρέφιας όπως για παράδειγμα (μην επαναλαμβάνεις τον εαυτό σου..).

Σε αυτές τις αντικειμενοστρεφείς αρχές σχεδιασμού, αλλά και σε διάφορες βέλτιστες πρακτικές που έχουν με το πέρας του χρόνου καταγραφεί, βασίζονται τα αντικειμενοστρεφή σχεδιαστικά πρότυπα.

9.1.3 Έννοιες Αντικειμενοστραφούς Προγραμματισμού (OOP Concepts)

Οι βασικές έννοιες του αντικειμενοστραφούς προγραμματισμού είναι τέσσερις και είναι οι ακόλουθες επιγραμματικά.

- Αφαιρετικότητα (Abstraction)
- Ενθυλάκωση (Encapsulation)
- Πολυμορφισμός (Polymorphism)
- Κληρονομικότητα (Inheritance)

9.2 ΣΧΕΔΙΑΣΤΙΚΕΣ ΑΡΧΕΣ - DESIGN PRINCIPLES

9.2.1 Γενικά

Όπως αναφέραμε και παραπάνω εκτός από τις τέσσερις έννοιες του αντικειμενοστραφούς προγραμματισμού υπάρχουν γενικότερα οι σχεδιαστικές αρχές. Έτσι υπάρχει ένα ακρωνύμιο στην αγγλική γλώσσα, το S.O.L.I.D για τις πρώτες πέντε σύμφωνα με τον Robert C. Martin. Αυτές οι αρχές, συνδυασμένες μεταξύ τους, παρέχουν ευκολία σε ένα προγραμματιστή να δημιουργήσει ένα νέο πρόγραμμα, να το συντηρήσει αλλά και να το επεκτείνει. Επίσης βοηθούν στην αποφυγή προβληματικού κώδικα, στην ευκολία του επαναπρογραμματισμού, ενώ τέλος είναι και μέρος του agile και adaptive software development. Επιγραμματικά οι πέντε αρχές στην αγγλική είναι :

- S – Single-responsiblity principle
- O – Open-closed principle
- L – Liskov substitution principle
- I – Interface segregation principle
- D – Dependency Inversion Principle

9.2.2 Διαχωρισμός Ανησυχιών (Seperation of Concerns)

Στην επιστήμη των υπολογιστών, ο διαχωρισμός των ανησυχιών (SoC) είναι μια αρχή σχεδιασμού (design principle) για το διαχωρισμό ενός προγράμματος ηλεκτρονικού υπολογιστή σε διακριτά τμήματα, έτσι ώστε κάθε τμήμα καλύπτει ένα ξεχωριστό ενδιαφέρον.

Η αξία του διαχωρισμού των "ανησυχιών" απλοποιεί ανάπτυξη και συντήρηση των προγραμμάτων ηλεκτρονικών υπολογιστών. Όταν αυτά είναι καλά διαχωρισμένα, τότε πρόκειται για μεμονωμένα τμήματα που μπορούν να επαναχρησιμοποιηθούν, καθώς αναπτύσσονται και ενημερώνονται ανεξάρτητα. Ιδιαίτερης αξίας είναι η δυνατότητα αργότερα για βελτίωση ή τροποποίηση ενός τμήμα του κώδικα χωρίς να χρειάζεται να δοθούν στοιχεία άλλων τμήματα, και χωρίς να χρειάζεται να γίνουν οι αντίστοιχες αλλαγές σε αυτά τα τμήματα.

9.2.3 Βασικές Αντικειμενοστρεφής Σχεδιαστικές Αρχές (Basic OO Principles)

Η εξειδίκευση των σχεδιαστικών αρχών στον αντικειμενοστρεφή προγραμματισμό μας οδηγεί στις αντικεμενοστραφείς σχεδιαστικές αρχές.

Αυτές χωρίζονται στις **βασικές** που είναι τρεις :

- Ενθυλακώστε ότι διαφέρει (Encapsulate what varies)
- Προτιμήστε την σύνθεση έναντι της κληρονομικότητας (Favor Composition over Inheritance)
- Προγραμματίστε επί των Διεπαφών και όχι επί των υλοποιήσεων (Program to Interface Not to Implementations)

9.2.4 Αντικειμενοστρεφής Σχεδιαστικές Αρχές (OO Design Principles)

Παρακάτω θα αναφέρουμε μερικές από τις σχεδιαστικές αρχές σύμφωνα με το βιβλίο "Head First Design Patterns" καθώς επίσης και την συσχέτιση τους με τα αντίστοιχα design patterns τα οποία θα δούμε παρακάτω. Πρόκειται επί της ουσίας για σχεδιαστικές "συμβουλές"

- Προσπάθησε για χαλαρά συνδεδεμένους σχεδιασμούς μεταξύ αντικειμένων που αλληλεπιδρούν.
  (Strive for loosely coupled designs betweens objects tha interact)
  Αυτή η αρχή συνδέεται με το "**Observer**" design pattern .

- Οι κλάσεις θα πρέπει να είναι ανοικτές για επέκταση αλλά κλειστές για τροποποίηση.
  (Classes should be open for extension but close for modification)
  Αυτή η αρχή συνδέεται με το "**Decorator**" design pattern .

- Βασίσου στις αφαιρέσεις και όχι στις συγκεκριμένες υλοποιήσεις/κλάσεις
  (Depend on abstractions. Do not depend on concrete classes.)
  Αυτή η αρχή συνδέεται με το "**Factory Method**" design pattern .

- Μίλα μόνο στους ομοειδείς σου
  (Only talk to your friends)
  Αυτή η αρχή συνδέεται με το "**Adapter and Fascade**" design pattern .

- Μην μας καλέσεις θα σε καλέσουμε εμείς!
  (Don't call us – we 'll call you!!)
  Αυτή η αρχή ονομάζεται και αλλιώς **Inversion Of Control** και συνδέεται με το "**Template Method**" design pattern.

- Μια κλάση θα έπρεπε να έχει μόνο ένα λόγο για να αλλάζει.
  (A class should have only one reason to change)
  Αυτή η αρχή συνδέεται με το "Composite" design pattern.

9.3 ΣΧΕΔΙΑΣΤΙΚΑ ΠΡΟΤΥΠΑ - DESIGN PATTERNS

9.3.1 Γενικά

Στην πληροφορική, ένα σχεδιαστικό πρότυπο ή σχεδιαστικό μοτίβο (design pattern) ορίζεται ως μία αποδεδειγμένα καλή λύση που έχει εφαρμοστεί με επιτυχία στην επίλυση ενός επαναλαμβανόμενου προβλήματος σχεδίασης συστημάτων λογισμικού. Οι λύσεις αυτές ελήφθησαν από την δοκιμή και το λάθος από πολλούς προγραμματιστές λογισμικού πάνω από ένα αρκετά σημαντικό χρονικό διάστημα.

Τα πρότυπα σχεδίασης ορίζονται τόσο σε επίπεδο μακροσκοπικής σχεδίασης όσο και σε επίπεδο υλοποίησης.

Μερικά από τα πιο γνωστά Design Patterns είναι τα ακόλουθα :

- Observer Pattern
- Decorator Pattern
- Factory Pattern
- Singleton Pattern
- Command Pattern
- Adapter and Façade Patterns
- Template Method Pattern
- Iterator and Composite Patterns
- State Pattern
- Proxy Pattern
- Compound Patterns (Pattern of Patterns)

9.3.2 Σχεδιαστικές Αρχές σε σχέση με Σχεδιαστικά Μοτίβα (Design Principles vs Design Patterns)

Σύμφωνα με τα παραπάνω, σχετικά με τα Design Principles και τα Design Patterns θα μπορούσαμε να δώσουμε δύο ορισμούς.

Σχεδιαστικές Αρχές (Design Principles)

Οι προγραμματιστικές σχεδιαστικές αρχές αντιπροσωπεύουν ένα πλήθος κατευθυντήριων γραμμών, που μας βοηθούν να αποφύγουμε κακούς σχεδιασμούς (π.χ. Open Close Principle)

Σχεδιαστικά Πρότυπα (Design Patterns)

Ένα σχεδιαστικό πρότυπο είναι γενικά μια επαναχρησιμοποιούμενη λύση σε ένα πρόβλημα που συναντάται συχνά σε ένα συγκεκριμένο γενικό πλαίσιο στον προγραμματιστικό σχεδιασμό. (π.χ. Singleton Design Pattern)

Σύμφωνα με τα παραπάνω θα μπορούσαμε να καταλήξουμε στο συμπέρασμα ότι τα Design Patterns είναι για τα Design Principles ότι οι υλοποιήσεις για τα Design Patterns.

9.4 MODEL VIEW CONTROLLER

9.4.1 Γενικά

Το Model–view–controller (σε συντομογραφία αναφέρεται ως MVC) είναι ένα μοντέλο αρχιτεκτονικής λογισμικού το οποίο χρησιμοποιείται για την δημιουργία περιβαλλόντων αλληλεπίδρασης χρήστη. Στο μοντέλο αυτό η εφαρμογή διαιρείται σε τρία διασυνδεδεμένα μέρη ώστε να διαχωριστεί η παρουσίαση της πληροφορίας στον χρήση από την μορφή που έχει αποθηκευτεί στο σύστημα. Το κύριο μέρος του μοντέλου είναι το αντικείμενο Model το οποίο διαχειρίζεται την ανάκτηση/αποθήκευση των δεδομένων στο σύστημα. Το αντικείμενο View χρησιμοποιείται μόνο για να παρουσιάζεται η πληροφορία στον χρήστη (π.χ. με γραφικό τρόπο). Το τρίτο μέρος είναι ο Controller ο οποίος δέχεται την είσοδο και στέλνει εντολές στο αντικείμενο Model και στο View.

9.4.2 Αλληλεπίδραση Μοντέλων

Εκτός από το να διαιρείται η εφαρμογή σε τρία μοντέλα, η σχεδίαση model–view–controller ορίζει και τις αλληλεπιδράσεις των μοντέλων.

Ο "**Controller**" μπορεί να στέλνει εντολές στο μοντέλο και να ενημερώνει την κατάσταση του μοντέλου. Μπορεί επίσης να στέλνει εντολές ώστε να γίνει η αντίστοιχη αναπαράσταση των δεδομένων του μοντέλου μέσω του View.

Το "Model" ενημερώνει τις αντίστοιχες αναπαραστάσεις views και τους controllers όταν υπάρχει αλλαγή στα δεδομένα. Αυτή η ενημέρωση επιτρέπει στα views να ενημερώνουν την γραφική απεικόνιση.

Το "View" αναπαριστά με γραφικό τρόπο την πληροφορία που περιέχει το model δημιουργώντας γραφική παρουσίαση στο χρήστη.

9.4.3 MVC , Design Patterns και Seperation Of Concerns

Αναλύοντας το MVC υπό το πρίσμα των Design Patterns που αναφερθήκαμε πρωτύτερα, θα δούμε ότι πρόκειται για συνδυασμό τους.

Έτσι, το μοντέλο χρησιμοποιεί το Observer Design Pattern, προκειμένου να κρατά τα views και τους controller στην τελευταία κατάσταση αλλαγών.

Από την άλλη μεριά, το view και ο controller υλοποιούν το Strategy Pattern. Ο ελεγκτής είναι η συμπεριφορά (behavior) του view, και μπορεί εύκολα να αντικατασταθεί από άλλο ελεγκτή αν επιθυμείται η αλλαγή της συμπεριφοράς. Το view επίσης χρησιμοποιεί ένα pattern εσωτερικά ώστε να διαχειρίζεται τα παράθυρα, τα κουμπιά και τα άλλα στοιχεία εμφάνισης (components) και δεν είναι άλλο από το Composite Pattern.

Επομένως καταλήγουμε στο συμπέρασμα ότι πρόκειται για ένα συνδυασμό από Design Patterns (Observer, Stategy, Composite) το οποίο έχει γνωρίσει μεγάλη διάδοση λόγω της λογικής του διαχωρισμού των ευθυνών (Seperation of Concerns).

.

9.5 ΑΝΤΙΣΤΡΟΦΗ ΕΛΕΓΧΟΥ ΚΑΙ ΕΚΧΥΣΗ ΕΞΑΡΤΗΣΗΣ (IOC AND DI)

9.5.1 Αντιστροφή Ελέγχου (Inversion of Control)

Η αντιστροφή ελέγχου (Invertion of Control) είναι μια σχεδιαστική αρχή (design principal) κατά την οποία ο έλεγχος ροής του προγράμματος αντιστρέφεται. Έτσι αντί ο προγραμματιστής να ελέγχει την ροή του προγράμματος, εξωτερικός κώδικας (framework, services ή άλλα στοιχεία) αναλαμβάνουν τον έλεγχο. Είναι σαν να συνδέουμε κάτι (τον κώδικα μας) κάπου αλλού. Αυτό οδηγεί και στην θεμελιώδη διαφορά μεταξύ προγραμματιστικού πλαισίου (framework) και βιβλιοθηκών (libraries). Έτσι αντί ο δικός μας κώδικας να καλεί μια μέθοδο μιας βιβλιοθήκη και ο έλεγχος να επιστρέφει σε εμάς, γράφουμε κώδικα υλοποιώντας κλήσεις με συγκεκριμένες προδιαγραφές, που θα γίνουν από το framework σε δεδομένες συνθήκες, ενώ ο έλεγχος επιστρέφεται πίσω στον γενικό επαναχρησιμοποιημένο κώδικα (framework).

Η αντιστροφή ελέγχου, χρησιμοποιείται για να αυξήσουμε την σπονδυλώτητα του κώδικά μας (modularity), προκειμένου να κάνουμε την εφαρμογή μας επεκτάσιμη, εξυπηρετώντας τα ακόλουθους σχεδιαστικούς σκοπούς :

- Να αποσυσχετίσει (decouple) την εκτέλεση μιας λειτουργίας από μια υλοποίηση
- Να μείνει “προσηλωμένο” το συγκεκριμένο κομμάτι του κώδικα (module), σε αυτό για το οποίο σχεδιάστηκε.

- Να απελευθερώσει το συγκεκριμένο κομμάτι κώδικα (module) από τις όποιες εικασίες σχετικά με το πως άλλα υποσυστήματα επιτυγχάνουν αυτό που πρέπει να κάνουν. Αντ' αυτού, του επιτρέπει να βασίζεται στις συμβάσεις (contracts).
- Να αποφεύγονται ανεπιθύμητες επιπλοκές κατά την αντικατάσταση ενός module.

Όπως έχουμε ήδη πει παραπάνω, το Invertion Of Control μερικές φορές αποκαλείται και ως "Hollywood Principle" ή εναλλακτικά.. "Μην μας καλέσετε. Θα σας καλέσουμε εμείς!!" (Don't call us – We' ll call you1!)

9.5.2 Έκχυση Εξάρτησης (Dependency Injection)

Στον τεχνολογία λογισμικού (software engineering), η "έκχυση εξάρτησης" (Dependency Injection) είναι ένα προγραμματιστικό σχεδιαστικό πρότυπο που υλοποιεί την σχεδιαστική αρχή της αναστροφής του ελέγχου προκειμένου να επιλύσει τις εξαρτήσεις.

Μια εξάρτηση είναι ένα αντικείμενο το οποίο μπορεί να χρησιμοποιηθεί (π.χ. μια υπηρεσία/service). Μια έκχυση (injection), είναι η διαδικασία παροχής μιας εξάρτησης σε ένα εξαρτώμενο αντικείμενο (ένα πελάτη/client) προκειμένου να το χρησιμοποιήσει. Αυτή η υπηρεσία υφίσταται εν' μέρη από την κατάσταση του πελάτη. Το να δίδεται η υπηρεσία στον πελάτη (pass service to client) αντί να επιτρέπεται ο πελάτης/client να δημιουργεί (hardcoded...πιθανότατα) ή να βρίσκει την υπηρεσία, είναι η θεμελιώδης απαίτηση αυτού του σχεδιαστικού προτύπου.

Η έκχυση εξάρτησης επιτρέπει σε ένα πρόγραμμα να ακολουθήσει την προγραμματιστική σχεδιαστική αρχή της Αντιστροφής της Εξάρτησης (Dependency Inversion Principle). Έτσι ο πελάτης προωθεί σε εξωτερικό κώδικα (τον εγχυνών), την ευθύνη της παροχής των εξαρτήσεων του. Ο πελάτης δεν είναι δυνατόν να καλεί τον κώδικα του εγχύνων (injector). Ο κώδικας του εγχύνων είναι εκείνος που θα δημιουργήσει την υπηρεσιακά θα καλέσει τον πελάτη προκειμένου να του εκχύσει τις εξαρτήσεις του.

Αυτό σημαίνει ότι ο πελάτης δεν χρειάζεται να ξέρει τίποτα σχετικά με το κώδικα που του εκχύεται. Επιπλέον ο πελάτης/client δεν χρειάζεται να ξέρει πως θα χρησιμοποιήσει την υπηρεσία/service. Ενώ ο πελάτης δεν χρειάζεται να ξέρει πια συγκεκριμένη (concrete) υπηρεσία/service χρησιμοποιεί. Ο πελάτης χρειάζεται μόνο να ξέρει σχετικά με τις εγγενείς

διεπαφές της υπηρεσίας διότι αυτές περιγράφουν πως θα μπορέσει (ο πελάτης) να τις χρησιμοποιήσει. Με αυτή την διαδικασία διαχωρίζονται οι ευθύνες της χρήσης από αυτές της κατασκευής (της υπηρεσίας).

Υπάρχουν τρείς βασικοί τρόποι για να δεκτοί ένας πελάτης/client μια έκχυση εξάρτησης (dependency injection) και αυτοί είναι διαμέσου των ακόλουθων:

- Setters
- Interface
- Constructor

9.6 ΥΠΟΚΛΟΠΕΙΣ ΚΑΙ ΦΙΛΤΡΑ (INTERCEPTORS AND FILTERS)

9.6.1 Υποκλοπείς (Interceptors)

Στον τομέα της ανάπτυξης λογισμικού, το πρότυπο υποκλοπέα (interceptor pattern) είναι ένα σχεδιαστικό πρότυπο (design pattern) το οποίο χρησιμοποιείται όταν προγράμματα (software systems) ή πλαίσια προγραμμάτων (frameworks) θέλουν να προσφέρουν ένα τρόπο να αλλάξουν , ή να αυξηθεί, ο συνήθης κύκλος ζωής τους. Για παράδειγμα, μια (απλοποιημένα) τυπική ακολουθία επεξεργασίας (processing sequence) ενός web-server είναι να λαμβάνει μια διεύθυνση (URI) από τον browser, να τον αντιστοιχεί σε ένα αρχείο, να ανοίγει αυτό το αρχείο και να αποστέλλει πίσω τα περιεχόμενα του στον περιηγητή. Οποιαδήποτε από αυτά τα στάδια θα μπορούσε να αντικατασταθεί ή να τροποποιηθεί, π.χ. με την αντικατάσταση του τρόπου με τον οποίο αντιστοιχίζονται τα URIs με τα αρχεία, ή εισάγοντας ένα νέο βήμα που επεξεργάζεται τα περιεχόμενα αρχεία.

9.6.2 Φίλτρα (Filters)

Ένα παράδειγμα υλοποίησης του προτύπου υποκλοπέα είναι και τα φίλτρα, που είναι και μέρος των προδιαγραφών(specification) των Servlets (javax.servlet.Filter interface).

Τα φίλτρα υποκλέπτουν δυναμικά τις αιτήσεις (requests) και τις αποκρίσεις (responses) που λαμβάνονται και αποστέλλονται μεταξύ ενός περιηγητή (client) και μιας εφαρμογής (server side), προκειμένου είτε να τις τροποποιήσουν, είτε να τις χρησιμοποιήσουν αποκτώντας πρόσβαση στις πληροφορίες που περιέχονται σε αυτές. Τυπικά τα φίλτρα δεν δημιουργούν

αποκρίσεις (responses), αλλά παρέχουν γενικευμένες λειτουργίες που μπορούν να προσαρτηθούν σε κάθε τύπου servlet ή JSP σελίδας.

Γενικά τα φίλτρα μπορούν εκτελέσουν μια σειρά από διαφορετικές λειτουργίες μερικές τον οποίον περιγράφονται επιγραμματικά παρακάτω.

- Αυθεντικοποίηση – Απαγόρευση αιτήσεων βάση των στοιχείων χρήστη (**Authentication Filters**)
- Καταγραφή (logging) ή παρακολούθηση (auditing-tracking) χρηστών σε μια διαδικτυακή εφαρμογή (**Logging and Auditing Filters)**
- Μετατροπή εικόνων – Αλλαγή διαστάσεων χαρτών κ.τ.λ. (**Image conversion Filters)**
- Συμπίεση δεδομένων προκειμένου να είναι πιο μικρά τα αρχεία για λήψη (download) (**Data Compression Filters)**
- Εντοπιότητα – Στόχευση (localization-Targeting) τα αιτήματα και τις αποκρίσεις σε μια συγκεκριμένη εντοπιότητα

Άλλα φίλτρα θα μπορούσαν να ήταν : Encryption Filters, Tokenizing Filters, Filters that trigger resource access events. XSL/T filters, Mime-type chain Filter.

9.6.3 Διαφορά μεταξύ Υποκλοπέων και Φίλτρων (Interceptors vs Filters)

Οι υποκλοπείς χωρίζονται σε δύο κατηγορίες στους handlers Interceptors και τους Method Interceptors. Έτσι όπως είδαμε και παραπάνω, τα φίλτρα είναι μια υλοποίηση της έννοιας των υποκλοπέων (handler interceptor) και μάλιστα ως υλοποίηση μέσω του servlet specification γίνονται μόνο στο λεγόμενο web layer κατά την λήψη ενός αιτήματος και την αποστολή μιας απάντησης.

Από την άλλη διάφορα frameworks όπως το Struts το Spring έχουν υλοποιήσει υποκλοπείς οι οποίοι υλοποιούν την κατηγορία του Method Interceptor. Έτσι είναι δυνατόν η επίκληση τους πριν ή μετά την επίκληση μιας μεθόδου μιας κλάσης την οποία θέλουμε να κάνουμε intercept. Αυτό τον ακριβή έλεγχο για το πότε θα εκτελεστεί μια διεργασία, χωρίς να

μολύνουμε τον κώδικα της υφιστάμενης κλάσης, μπορεί να μας τον παράσχει μόνο ένας method interceptor.

Ο πτυχοκεντρικός προγραμματισμός (AOP), όπως θα δούμε στην συνέχεια, μπορεί επίσης να χρησιμοποιηθεί σε μερικές περιπτώσεις προκειμένου να παρέχει δυνατότητες ενός υποκλοπέα, παρ' ότι δεν χρησιμοποιεί τα στοιχεία που τυπικά περιγράφονται μέσω του σχεδίου του πρότυπου υποκλοπέα (interceptor pattern).

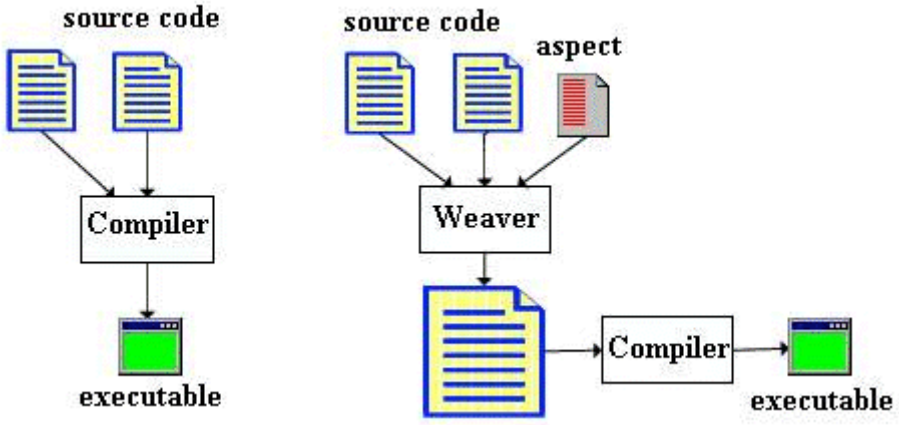
9.7 ASPECT ORIENTED PROGRAMMING (AOP)

9.7.1 Τι είναι το AOP

Ο AOP είναι μια νέα σχετικά μεθοδολογία για το διαχωρισμό ενός προβλήματος σε μεμονωμένες μονάδες που ονομάζονται πτυχές (aspects). Μια πτυχή είναι μια μονάδα που «κόβει» εγκάρσια τη λογική ροή μίας εφαρμογής (εξού και η καλύτερη απόδοση στα ελληνικά αντί της λέξης «θέμα»). Έτσι συμπυκνώνει συμπεριφορές (μεθόδους) που επηρεάζουν πολλαπλές κλάσεις και ενότητες του κώδικα μας και είναι επαναχρησιμοποιήσιμες αλλά η αλλαγή τους δεν έχει απολύτως καμία επίδραση στον υπόλοιπο κώδικα.

Με τον AOP, ξεκινάμε την υλοποίηση του έργου μας, χρησιμοποιώντας την αντικειμενοστρεφή γλώσσα μας (για παράδειγμα, Java), και στη συνέχεια απασχολούμαστε ξεχωριστά με τα θέματα που σχετίζονται στη δημιουργία επαναχρησιμοποιούμενου κώδικα που τέμνει εγκάρσια τη ροή του προγράμματος, με τη δημιουργία πτυχών. Τέλος, τόσο τα αντικείμενα όσο και οι πτυχές συνδυάζονται σε μια τελική εκτελέσιμη μορφή μέσω ενός «υφαντή πτυχών» (Aspect Weaver). Ως εκ τούτου, μια ενιαία πτυχή μπορεί να συμβάλει αποφασιστικά στην υλοποίηση μιας σειράς μεθόδων, ενοτήτων, ή αντικειμένων του λογισμικού, αυξάνοντας τόσο την δυνατότητα επαναχρησιμοποίησης όσο και της συντήρησης του κώδικα. Θα πρέπει να σημειωθεί ότι ο αρχικός κώδικας δεν χρειάζεται να έχει επίγνωση για κάθε λειτουργία που έχει προσθέσει μία πτυχή.

Το παρακάτω σχήμα εξηγεί τη διαδικασία της ύφανσης.



Εικόνα 46 – Διάγραμμα Πτυχοκεντρικού Προγραμματισμού

9.7.2 Βασικές Έννοιες

Για να κατανοήσουμε καλύτερα τον πτυχό-στρεφή προγραμματισμό ας δούμε αναλυτικότερα κάποιες από τις βασικές έννοιες πάνω στις οποίες στηρίζεται.

- **Aspect (πτυχή):** Όπως είπαμε και πριν, μια πτυχή είναι μια μονάδα που μπαίνει εγκάρσια στη λογική ροή μίας εφαρμογής διακόπτοντας την για να εκτελέσει μία λειτουργία.
- **Join point:** ένα σημείο κατά τη διάρκεια εκτέλεσης του προγράμματος όπως πχ μία μέθοδος ή το handing κάποιου exception.
- **Advice:** Η ενέργεια που κάνει μία πτυχή για ένα συγκεκριμένο join point. Υπάρχουν διάφοροι τύποι advice και περιλαμβάνουν τα «around», «before» και «after». (Θα τα δούμε παρακάτω). Πολλά frameworks όπως το Spring, μοντελοποιούν το advice σαν interceptor.
- **Pointcut:** Εκφράζει το σημείο που γίνεται η τομή στη ροή των join points δηλαδή στη ροή του προγράμματος. Ένα Advice συσχετίζεται με μία pointcut expression και καλείται σε οποιοδήποτε join point ταιριάζει σε αυτή την expression (π.χ. η εκτέλεση μίας μεθόδου η ονομασία της οποίας ταιριάζει στην expression του pointcut). Η έννοια του join points που συσχετίζεται με pointcut expressions είναι κεντρική στον AOP.

- **Weaving:** Το τελικό κομμάτι όπου γίνεται η σύνδεση πτυχών (aspects) με τους άλλους τύπους εφαρμογών ή αντικειμένων για τη δημιουργία του advice αντικειμένου που θα εκτελείται στις τομές. Ο συνυφασμός αυτός μπορεί να συμβεί κατά το compile (χρησιμοποιώντας τον AspectJ compiler), κατά τη φόρτωση ή και κατά την εκτέλεση.

9.7.3 Κλασικά Παραδείγματα AOP

Ένα κλασσικό παράδειγμα όπου κώδικας τέμνει εγκάρσια τη ροή του προγράμματος είναι το logging. Σε αυτή τη περίπτωση θέλουμε για συγκεκριμένες πράξεις που λαμβάνουν χώρα, να σταματάει η ροή για να αφήσει το σύστημα ένα αποτύπωμα με κάποιες συγκεκριμένες πληροφορίες. Άλλο παράδειγμα θα ήταν η ασφάλεια (security).

9.8 ΑΡΧΕΙΑ ΡΥΘΜΙΣΕΩΝ

9.8.1 security.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns="http://www.springframework.org/schema/security"
xmlns:beans="http://www.springframework.org/schema/beans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.springframework.org/schema/beans
                    http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
                    http://www.springframework.org/schema/security
                    http://www.springframework.org/schema/security/spring-security.xsd">
        <global-method-security pre-post-annotations="enabled" />
        <http use-expressions="true">
                <intercept-url pattern="/admin/users**" access="hasRole('ROLE_ADMIN')"/>
                <intercept-url pattern="/admin/users/**" access="hasRole('ROLE_ADMIN')"/>
                <intercept-url pattern="/admin/**" access="isAuthenticated()" />
                <form-login login-page="/login.html" />
                <logout logout-success-url="/admin/home.html" />
        </http>
        <authentication-manager>
                <authentication-provider>
                        <password-encoder hash="bcrypt" />
      <!-- <password-encoder hash="md5" /> -->
      <!-- <password-encoder hash="sha-256" /> -->
                        <jdbc-user-service data-source-ref="dataSource"
                                authorities-by-username-query="SELECT User.username as user, Role.name
                                                                FROM User
                                                                JOIN user_has_role ON User.id = user_has_role.user_id
                                                                JOIN Role ON user_has_role.role_id = Role.id
                                                        WHERE username = ?"
                                users-by-username-query="SELECT username as name ,password , status as enabled
FROM User WHERE username = ?" />
                </authentication-provider>
        </authentication-manager>
</beans:beans>
```

9.8.2 applicationContext.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
	xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
	xmlns:context="http://www.springframework.org/schema/context"
	xmlns:jdbc="http://www.springframework.org/schema/jdbc"
	xmlns:tx="http://www.springframework.org/schema/tx"
	xmlns:jpa="http://www.springframework.org/schema/data/jpa"
	xsi:schemaLocation="http://www.springframework.org/schema/jdbc
http://www.springframework.org/schema/jdbc/spring-jdbc-4.0.xsd
		http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd
		http://www.springframework.org/schema/data/jpa http://www.springframework.org/schema/data/jpa/spring-jpa-1.3.xsd
		http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-4.0.xsd
		http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-4.0.xsd">




	<context:component-scan base-package="gr.greekchoices.admin">
		<context:exclude-filter type="annotation" expression="org.springframework.stereotype.Controller" />
	</context:component-scan>
	<tx:annotation-driven transaction-manager="transactionManager" />
	<bean class="org.springframework.orm.jpa.JpaTransactionManager" id="transactionManager">
		<property name="dataSource" ref="dataSource" />
	</bean>
	<jpa:repositories base-package="gr.greekchoices.admin.repository" entity-manager-factory-ref="emf" transaction-manager-ref="transactionManager"/>
	<import resource="security.xml"/>
	<beans profile="dev">
		<import resource="database-dev.xml"/>
	</beans>
	<beans profile="prod">
		<import resource="database-prod.xml"/>
	</beans>
</beans>
```

9.8.3 dispatcher-servlet.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"

	xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:context="http://www.springframework.org/schema/context"

	xmlns:mvc="http://www.springframework.org/schema/mvc"

	xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd

		http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-
context-4.0.xsd

  http://www.springframework.org/schema/mvc

  http://www.springframework.org/schema/mvc/spring-mvc.xsd">

	<context:component-scan base-package="gr.greekchoices.admin.controller" />

	<mvc:annotation-driven />

	<bean id="tilesConfigurer"

		class="org.springframework.web.servlet.view.tiles3.TilesConfigurer" >

		<property name="definitions">

			<list>

				<value>/WEB-INF/defs/general.xml</value>

			</list>

		</property>

	</bean>

	<bean id="viewResolver"

		class="org.springframework.web.servlet.view.UrlBasedViewResolver">

		<property name="viewClass"

			value="org.springframework.web.servlet.view.tiles3.TilesView" />

	</bean>

</beans>
```

9.8.4 Αρχείο Ρυθμίσεων Apache Tiles (general.xml)

Στο αρχείο αυτό μπορούμε να ρυθμίσουμε τα Apache Tiles. Πρόκειται για ένα Templating Framework της apache.org το οποίο επιτρέπει τον ορισμό υποσυνόλων μιας σελίδας (όπως header, footer, breadcrumb κ.τ.λ.) και την σύνθεση της σελίδας σε πραγματικό χρόνο. Επιπλέον διάφορες μεταβλητές μπορούν να θέτωνται σε αυτό το κεντροποιημένο αρχείο ρυθμίσεων και μπορεί να επικαλεσθεί μέσα στις σελίδες jsp. Να σημειώσουμε επιπλέον, ότι το αρχείο αυτό βρίσκεται στον φάκελο "/src/main/webapp/WEB-INF/defs/general.xml"

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE tiles-definitions PUBLIC   "-//Apache Software Foundation//DTD Tiles Configuration 3.0//EN"
     "http://tiles.apache.org/dtds/tiles-config_3_0.dtd">
<tiles-definitions>
  <definition name="common" template="/WEB-INF/layout/classic.jsp">
   <put-attribute name="header" value="/WEB-INF/layout/header.jsp" />
   <put-attribute name="breadcrumb" value="/WEB-INF/layout/breadcrumb.jsp" cascade="true"/>
   <put-attribute name="pageHeader" value="/WEB-INF/layout/pageHeader.jsp" cascade="true"/>
   <put-attribute name="footer" value="/WEB-INF/layout/footer.jsp" />
  </definition>
  <!--  Index (Starts) -->
  <definition name="index" extends="common">
   <put-attribute name="title" value="Login or Register" cascade="true"/>
   <put-attribute name="current" value="index" cascade="true"/>
   <put-attribute name="body" value="/WEB-INF/jsp/index.jsp" />
  </definition>
              <!--  Index (Ends) -->
              <!--  Login (Starts) -->
  <definition name="login" extends="common">
   <put-attribute name="title" value="Login to Greekchoices Admin Control Panel" cascade="true"/>
   <put-attribute name="current" value="login" cascade="true"/>
   <put-attribute name="body" value="/WEB-INF/jsp/login.jsp" />
  </definition>
              <!--  Login (Ends) -->
<!--  Admin Pages (Starts) -->
  <!--  Home (Starts) -->
  <definition name="home" extends="common">
   <put-attribute name="title" value="Home" cascade="true"/>
   <put-attribute name="current" value="home" cascade="true"/>
   <put-attribute name="body" value="/WEB-INF/jsp/admin/home.jsp" />
  </definition>
  <!--  Home (Ends) -->
  <!--  User (Starts) -->
  <definition name="userRegister" extends="common">
   <put-attribute name="title" value="Users Registration" cascade="true"/>
   <put-attribute name="current" value="register" cascade="true"/>
   <put-attribute name="upLevel" value="/index.html" cascade="true"/>
   <put-attribute name="descr" value="This is where you can register yourself." cascade="true"/>
   <put-attribute name="body" value="/WEB-INF/jsp/admin/user/userRegister.jsp" />
   <put-attribute name="breadcrumbPairs" expression="[Users Registration,]" cascade="true" />
  </definition>
```

```
 <definition name="users" extends="common">
  <put-attribute name="title" value="Users Listing" cascade="true"/>
  <put-attribute name="current" value="users" cascade="true"/>
  <put-attribute name="upLevel" value="/admin/home.html" cascade="true"/>
  <put-attribute name="descr" value="This is where you configure and manage your users" cascade="true"/>
  <put-attribute name="body" value="/WEB-INF/jsp/admin/user/users.jsp" />
  <put-attribute name="breadcrumbPairs" expression="[Users Listing,]" cascade="true" />
 </definition>
 <definition name="userDetail" extends="common">
  <put-attribute name="title" value="User Detail" cascade="true"/>
  <put-attribute name="current" value="userDetail" cascade="true"/>
  <put-attribute name="upLevel" value="/admin/users.html" cascade="true"/>
  <put-attribute name="descr" value="This is where you can see your user's details." cascade="true"/>
  <put-attribute name="body" value="/WEB-INF/jsp/admin/user/userDetail.jsp" />
  <put-attribute name="breadcrumbPairs" expression="[Users Listing,/admin/users.html]|[User Detail,]"  cascade="true"/>
 </definition>
 <definition name="account" extends="common">
  <put-attribute name="title" value="My Account" cascade="true"/>
  <put-attribute name="current" value="account" cascade="true"/>
  <put-attribute name="upLevel" value="/admin/home.html" cascade="true"/>
  <put-attribute name="descr" value="This is where you can see your account details." cascade="true"/>
  <put-attribute name="body" value="/WEB-INF/jsp/admin/user/userDetail.jsp" />
  <put-attribute name="breadcrumbPairs" expression="[My Account,]"  cascade="true"/>
 </definition>
            <!--  User (Ends) -->
            <!-- Hotel Control Panel (Starts) -->
 <definition name="hotelControlPanel" extends="common">
  <put-attribute name="title" expression="${hotel.name} Control Panel" cascade="true"/>
  <put-attribute name="current" value="hotelControlPanel" cascade="true" />
  <put-attribute name="upLevel" value="/admin/hotels.html" cascade="true"/>
  <put-attribute name="descr" value="This is where you configure and manage your hotels, and view resource usage statistics."
cascade="true"/>
  <put-attribute name="body" value="/WEB-INF/jsp/admin/hotels/hotelControlPanel.jsp" />
  <put-attribute name="breadcrumbPairs" expression="[Hotels Listing,/admin/hotels.html]|[Hotel Control Panel,]"  cascade="true"/>
 </definition>
            <!-- Hotel Control Panel (Ends) -->
            <!-- Hotel Listing (Starts) -->
 <definition name="hotelsListing" extends="common">
  <put-attribute name="title" value="Hotels Listing" cascade="true"/>
  <put-attribute name="current" value="hotelsListing" cascade="true" />
  <put-attribute name="upLevel" value="/admin/home.html" cascade="true"/>
  <put-attribute name="descr" value="This is where you configure and manage your hotels." cascade="true"/>
```

```
  <put-attribute name="body" value="/WEB-INF/jsp/admin/hotels/hotelsListing.jsp" />
  <put-attribute name="breadcrumbPairs" expression="[Hotels Listing,]" cascade="true" />
 </definition>
 <!-- Hotel Listing (Ends) -->
 <!-- Hotel Basic Info (Starts) -->
 <definition name="hotelBasicInfoRead" extends="common">
  <put-attribute name="title" expression="${hotel.name} Basic Info" cascade="true"/>
  <put-attribute name="current" value="hotelBasicInfoRead" cascade="true" />
  <put-attribute name="upLevel" expression="/admin/hotels/${hotel.id}/control_panel.html" cascade="true"/>
  <put-attribute name="descr" value="This is where you can see your hotel's details." cascade="true"/>
  <put-attribute name="body" value="/WEB-INF/jsp/admin/hotels/basicInfo/hotelBasicInfoRead.jsp" />
  <put-attribute name="breadcrumbPairs" expression="[Hotels Listing,/admin/hotels.html]|[Hotel Control
Panel,/admin/hotels/${hotel.id}/control_panel.html]|[Hotel Basic Info,]" cascade="true" />
 </definition>
 <!-- Hotel Basic Info (Ends) -->
 <!-- Hotel Facilities (Starts) -->
 <definition name="hotelFacilitiesListing" extends="common">
  <put-attribute name="title" expression="${hotel.name} Facilities Listing" cascade="true"/>
  <put-attribute name="current" value="hotelFacilitiesListing" cascade="true" />
  <put-attribute name="upLevel" expression="/admin/hotels/${hotel.id}/control_panel.html" cascade="true"/>
  <put-attribute name="descr" value="This is where you configure your hotel facilities. You can add some of the preconfigured global
hotel facilities or you can create your own custom new hotel facilities." cascade="true"/>
  <put-attribute name="body" value="/WEB-INF/jsp/admin/hotels/facilities/hotelFacilities/hotelFacilitiesListing.jsp" />
  <put-attribute name="breadcrumbPairs" expression="[Hotels Listing,/admin/hotels.html]|[Hotel Control
Panel,/admin/hotels/${hotel.id}/control_panel.html]|[Hotel Facilities Listing,]" cascade="true" />
 </definition>
 <definition name="hotelFacilityEdit" extends="common">
  <put-attribute name="title" expression="${hotel.name} - Hotel Facility ${facility.name} Edit" cascade="true"/>
  <put-attribute name="current" value="hotelFacilityEdit" cascade="true" />
  <put-attribute name="upLevel" expression="/admin/hotels/${hotel.id}/facilities/hotel-facilities.html" cascade="true"/>
  <put-attribute name="descr" value="This is where you can edit your hotel's hotel facility." cascade="true"/>
  <put-attribute name="body" value="/WEB-INF/jsp/admin/hotels/facilities/hotelFacilities/hotelFacilityEdit.jsp" />
  <put-attribute name="breadcrumbPairs" expression="[Hotels Listing,/admin/hotels.html]|[Hotel Control
Panel,/admin/hotels/${hotel.id}/control_panel.html]|[Hotel Facilities Listing,/admin/hotels/${hotel.id}/facilities/hotel-
facilities.html]|[Hotel Facility Edit,]" cascade="true" />
 </definition>
 <!-- Hotel Facilities (Ends) -->
 <!-- Room Facilities (Starts) -->
 <definition name="roomFacilitiesListing" extends="common">
  <put-attribute name="title" expression="${hotel.name} Room Facilities Listing" cascade="true"/>
  <put-attribute name="current" value="roomFacilitiesListing" cascade="true" />
  <put-attribute name="upLevel" expression="/admin/hotels/${hotel.id}/control_panel.html" cascade="true"/>
  <put-attribute name="descr" value="This is where you configure your room facilities. You can add some of the preconfigured global
room facilities or you can create your own custom new room facilities." cascade="true"/>
```

```
   <put-attribute name="body" value="/WEB-INF/jsp/admin/hotels/facilities/roomFacilities/roomFacilitiesListing.jsp" />

   <put-attribute name="breadcrumbPairs" expression="[Hotels Listing,/admin/hotels.html]|[Hotel Control
Panel,/admin/hotels/${hotel.id}/control_panel.html]|[Room Facilities Listing,]" cascade="true" />

 </definition>

 <definition name="roomFacilityEdit" extends="common">

   <put-attribute name="title" expression="${hotel.name} - Room Facility ${facility.name} Edit" cascade="true"/>

   <put-attribute name="current" value="roomFacilityEdit" cascade="true" />

   <put-attribute name="upLevel" expression="/admin/hotels/${hotel.id}/facilities/room-facilities.html" cascade="true"/>

   <put-attribute name="descr" value="This is where you can edit your hotel's room facility." cascade="true"/>

   <put-attribute name="body" value="/WEB-INF/jsp/admin/hotels/facilities/roomFacilities/roomFacilityEdit.jsp" />

   <put-attribute name="breadcrumbPairs" expression="[Hotels Listing,/admin/hotels.html]|[Hotel Control
Panel,/admin/hotels/${hotel.id}/control_panel.html]|[Room Facilities Listing,/admin/hotels/${hotel.id}/facilities/room-
facilities.html]|[Room Facility Edit,]" cascade="true" />

 </definition>

 <!-- Room Facilities (Ends) -->

 <!-- Room Facilities Per Room Type All (Starts) -->

 <definition name="roomFacilitiesPerRoomTypeListing" extends="common">

   <put-attribute name="title" expression="${hotel.name} Room Facilities Per Room Type Listing" cascade="true"/>

   <put-attribute name="current" value="roomFacilitiesPerRoomTypeListing" cascade="true" />

   <put-attribute name="upLevel" expression="/admin/hotels/${hotel.id}/facilities/room-facilities.html" cascade="true"/>

   <put-attribute name="descr" value="This is where you configure your room facilities for all room types." cascade="true"/>

   <put-attribute name="body" value="/WEB-
INF/jsp/admin/hotels/facilities/roomFacilities/perRoomType/roomFacilitiesPerRoomTypeListing.jsp" />

   <put-attribute name="breadcrumbPairs" expression="[Hotels Listing,/admin/hotels.html]|[Hotel Control
Panel,/admin/hotels/${hotel.id}/control_panel.html]|[Room Facilities Listing,/admin/hotels/${hotel.id}/facilities/room-
facilities.html]|[Room Facilities Per Room Type Listing,]" cascade="true" />

 </definition>

 <!-- Room Facilities Per Room Type All (Ends) -->

 <!-- Room Facilities Per Room Type (Starts) -->

 <definition name="roomFacilitiesPerRoomType" extends="common">

   <put-attribute name="title" expression="${hotel.name} Room Facilities Per Room Type" cascade="true"/>

   <put-attribute name="current" value="roomFacilitiesPerRoomType" cascade="true" />

   <put-attribute name="upLevel" expression="/admin/hotels/${hotel.id}/facilities/room-facilities.html" cascade="true"/>

   <put-attribute name="descr" value="This is where you configure your room facilities for a room type." cascade="true"/>

   <put-attribute name="body" value="/WEB-
INF/jsp/admin/hotels/facilities/roomFacilities/perRoomType/roomFacilitiesPerRoomType.jsp" />

   <put-attribute name="breadcrumbPairs" expression="[Hotels Listing,/admin/hotels.html]|[Hotel Control
Panel,/admin/hotels/${hotel.id}/control_panel.html]|[Room Facilities Listing,/admin/hotels/${hotel.id}/facilities/room-
facilities.html]|[Room Facilities Per Room Type,]" cascade="true" />

 </definition>

 <!-- Room Facilities Per Room Type (Ends) -->

        <!-- Prices Per Period All (Starts) -->

 <definition name="pricePerAllRoomTypePerAllPeriodListing" extends="common">

   <put-attribute name="title" expression="${hotel.name} Prices Per Period Listing" cascade="true"/>

   <put-attribute name="current" value="pricePerAllRoomTypePerAllPeriodListing" cascade="true" />
```

```
  <put-attribute name="upLevel" expression="/admin/hotels/${hotel.id}/control_panel.html" cascade="true"/>

  <put-attribute name="descr" value="This is where you configure your hotel's prices. You must first have created periods and room types." cascade="true"/>

  <put-attribute name="body" value="/WEB-INF/jsp/admin/hotels/prices/perPeriod/pricePerAllRoomTypePerAllPeriodListing.jsp" />

  <put-attribute name="breadcrumbPairs" expression="[Hotels Listing,/admin/hotels.html]|[Hotel Control Panel,/admin/hotels/${hotel.id}/control_panel.html]|[Prices Per Period Listing,]" cascade="true" />

 </definition>

 <!-- Prices Per Period All (Ends) -->

 <!-- Prices Per Period (Starts) -->

 <definition name="pricePerAllRoomTypePerPeriod" extends="common">

  <put-attribute name="title" expression="${hotel.name} Prices Per Period" cascade="true"/>

  <put-attribute name="current" value="pricePerAllRoomTypePerPeriod" cascade="true" />

  <put-attribute name="upLevel" expression="/admin/hotels/${hotel.id}/control_panel.html" cascade="true"/>

  <put-attribute name="descr" value="This is where you configure your hotel's prices. You must first have created periods and room types." cascade="true"/>

  <put-attribute name="body" value="/WEB-INF/jsp/admin/hotels/prices/perPeriod/pricePerAllRoomTypePerPeriod.jsp" />

  <put-attribute name="breadcrumbPairs" expression="[Hotels Listing,/admin/hotels.html]|[Hotel Control Panel,/admin/hotels/${hotel.id}/control_panel.html]|[Prices Per Period,]" cascade="true" />

 </definition>

 <!-- Prices Per Period (Ends) -->

 <!-- Prices Per Room Type All (Starts) -->

 <definition name="pricePerAllPeriodPerAllRoomTypeListing" extends="common">

  <put-attribute name="title" expression="${hotel.name} Prices Per Room Type Listing" cascade="true"/>

  <put-attribute name="current" value="pricePerAllPeriodPerAllRoomTypeListing" cascade="true" />

  <put-attribute name="upLevel" expression="/admin/hotels/${hotel.id}/control_panel.html" cascade="true"/>

  <put-attribute name="descr" value="This is where you configure your hotel's prices. You must first have created periods and room types." cascade="true"/>

  <put-attribute name="body" value="/WEB-INF/jsp/admin/hotels/prices/perRoomType/pricePerAllPeriodPerAllRoomTypeListing.jsp" />

  <put-attribute name="breadcrumbPairs" expression="[Hotels Listing,/admin/hotels.html]|[Hotel Control Panel,/admin/hotels/${hotel.id}/control_panel.html]|[Prices Per Room Type Listing,]" cascade="true" />

 </definition>

 <!-- Prices Per Room Type All (Ends) -->

 <!-- Prices Per Room Type (Starts) -->

 <definition name="pricePerAllPeriodPerRoomType" extends="common">

  <put-attribute name="title" expression="${hotel.name} Prices Per Room Type" cascade="true"/>

  <put-attribute name="current" value="pricePerAllPeriodPerRoomType" cascade="true" />

  <put-attribute name="upLevel" expression="/admin/hotels/${hotel.id}/control_panel.html" cascade="true"/>

  <put-attribute name="descr" value="This is where you configure your hotel's prices. You must first have created periods and room types." cascade="true"/>

  <put-attribute name="body" value="/WEB-INF/jsp/admin/hotels/prices/perRoomType/pricePerAllPeriodPerRoomType.jsp" />

  <put-attribute name="breadcrumbPairs" expression="[Hotels Listing,/admin/hotels.html]|[Hotel Control Panel,/admin/hotels/${hotel.id}/control_panel.html]|[Prices Per Room Type,]" cascade="true" />

 </definition>

 <!-- Prices Per Room Type (Ends) -->
```

```
 <!-- Room Types (Starts) -->
 <definition name="roomTypesListing" extends="common">
  <put-attribute name="title" expression="${hotel.name} Room Types Listing" cascade="true"/>
  <put-attribute name="current" value="roomTypesListing" cascade="true" />
  <put-attribute name="upLevel" expression="/admin/hotels/${hotel.id}/control_panel.html" cascade="true"/>
  <put-attribute name="descr" value="This is where you configure your hotel's room types.You can add some of the preconfigured global room types or you can create your own custom new room types." cascade="true"/>
  <put-attribute name="body" value="/WEB-INF/jsp/admin/hotels/roomTypes/roomTypesListing.jsp" />
  <put-attribute name="breadcrumbPairs" expression="[Hotels Listing,/admin/hotels.html]|[Hotel Control Panel,/admin/hotels/${hotel.id}/control_panel.html]|[Room Types Listing,]" cascade="true" />
 </definition>
  <definition name="roomTypeEdit" extends="common">
  <put-attribute name="title" expression="${hotel.name} - ${roomType.name} Edit" cascade="true"/>
  <put-attribute name="current" value="roomTypesEdit" cascade="true" />
  <put-attribute name="upLevel" expression="/admin/hotels/${hotel.id}/room-types.html" cascade="true"/>
  <put-attribute name="descr" value="This is where you can edit your hotel's room type." cascade="true"/>
  <put-attribute name="body" value="/WEB-INF/jsp/admin/hotels/roomTypes/roomTypeEdit.jsp" />
  <put-attribute name="breadcrumbPairs" expression="[Hotels Listing,/admin/hotels.html]|[Hotel Control Panel,/admin/hotels/${hotel.id}/control_panel.html]|[Room Types Listing,/admin/hotels/${hotel.id}/room-types.html]|[Edit Room Type,]" cascade="true" />
 </definition>
 <!-- Room Types (Ends) -->
 <!-- Periods (Starts) -->
 <definition name="periodsListing" extends="common">
  <put-attribute name="title" expression="${hotel.name} Periods Listing" cascade="true"/>
  <put-attribute name="current" value="periodsListing" cascade="true" />
  <put-attribute name="upLevel" expression="/admin/hotels/${hotel.id}/control_panel.html" cascade="true"/>
  <put-attribute name="descr" value="This is where you configure your hotel's periods." cascade="true"/>
  <put-attribute name="body" value="/WEB-INF/jsp/admin/hotels/periods/periodsListing.jsp" />
  <put-attribute name="breadcrumbPairs" expression="[Hotels Listing,/admin/hotels.html]|[Hotel Control Panel,/admin/hotels/${hotel.id}/control_panel.html]|[Periods Listing,]" cascade="true" />
 </definition>
 <definition name="periodEdit" extends="common">
  <put-attribute name="title" expression="${hotel.name} - Period Edit" cascade="true"/>
  <put-attribute name="current" value="periodEdit" cascade="true" />
  <put-attribute name="upLevel" expression="/admin/hotels/${hotel.id}/periods.html" cascade="true"/>
  <put-attribute name="descr" value="This is where you can edit your hotel's period." cascade="true"/>
  <put-attribute name="body" value="/WEB-INF/jsp/admin/hotels/periods/periodEdit.jsp" />
  <put-attribute name="breadcrumbPairs" expression="[Hotels Listing,/admin/hotels.html]|[Hotel Control Panel,/admin/hotels/${hotel.id}/control_panel.html]|[Periods Listing,/admin/hotels/${hotel.id}/periods.html]|[Edit Period,]" cascade="true" />
 </definition>
 <!-- Periods (Ends) -->
 <!-- Availabilities Calendar Per Room Type All (Starts) -->
```

```
 <definition name="availabilitiesCalendarPerAllRoomType" extends="common">
  <put-attribute name="title" expression="${hotel.name} Availabilities Calendar Per Room Type Listing" cascade="true"/>
  <put-attribute name="current" value="availabilitiesCalendarPerAllRoomType" cascade="true" />
  <put-attribute name="upLevel" expression="/admin/hotels/${hotel.id}/control_panel.html" cascade="true"/>
  <put-attribute name="descr" value="This is where you configure the avaliabilities per room type. You must first have been create room types." cascade="true"/>
  <put-attribute name="body" value="/WEB-INF/jsp/admin/hotels/availability/availabilitiesCalendarPerAllRoomType.jsp" />
  <put-attribute name="breadcrumbPairs" expression="[Hotels Listing,/admin/hotels.html]|[Hotel Control Panel,/admin/hotels/${hotel.id}/control_panel.html]|[Availabilities Calendar Per All RoomType,]" cascade="true" />
 </definition>
 <!-- Availabilities Calendar Per Room Type All (Ends) -->
 <!-- Availabilities Calendar Per Room Type (Starts) -->
 <definition name="availabilitiesCalendarPerRoomType" extends="common">
  <put-attribute name="title" expression="${hotel.name} Availabilities Calendar Per Room Type" cascade="true"/>
  <put-attribute name="current" value="availabilitiesCalendarPerRoomType" cascade="true" />
  <put-attribute name="upLevel" expression="/admin/hotels/${hotel.id}/control_panel.html" cascade="true"/>
  <put-attribute name="descr" value="This is where you configure the avaliabilities per room type. You must first have been create room types." cascade="true"/>
  <put-attribute name="body" value="/WEB-INF/jsp/admin/hotels/availability/availabilitiesCalendarPerRoomType.jsp" />
  <put-attribute name="breadcrumbPairs" expression="[Hotels Listing,/admin/hotels.html]|[Hotel Control Panel,/admin/hotels/${hotel.id}/control_panel.html]|[Availabilities Calendar Per Room Type,]" cascade="true" />
 </definition>
 <!-- Availabilities Calendar Per Room Type (Ends) -->
 <!-- Booking Listing (Starts) -->
 <definition name="bookingListing" extends="common">
  <put-attribute name="title" expression="${hotel.name} Booking Listing" cascade="true"/>
  <put-attribute name="current" value="bookingListing" cascade="true" />
  <put-attribute name="upLevel" expression="/admin/hotels/${hotel.id}/control_panel.html" cascade="true"/>
  <put-attribute name="descr" value="This is where you can view your hotel's bookings." cascade="true"/>
  <put-attribute name="body" value="/WEB-INF/jsp/admin/hotels/booking/bookingListing.jsp" />
  <put-attribute name="breadcrumbPairs" expression="[Hotels Listing,/admin/hotels.html]|[Hotel Control Panel,/admin/hotels/${hotel.id}/control_panel.html]|[Booking Listing,]" cascade="true" />
 </definition>
 <!-- Booking Listing (Ends) -->
 <!-- Booking Calendar (Starts) -->
 <definition name="bookingCalendar" extends="common">
  <put-attribute name="title" expression="${hotel.name} Booking Calendar" cascade="true"/>
  <put-attribute name="current" value="bookingCalendar" cascade="true" />
  <put-attribute name="upLevel" expression="/admin/hotels/${hotel.id}/control_panel.html" cascade="true"/>
  <put-attribute name="descr" value="This is where you can view your hotel's bookings." cascade="true"/>
  <put-attribute name="body" value="/WEB-INF/jsp/admin/hotels/booking/bookingCalendar.jsp" />
  <put-attribute name="breadcrumbPairs" expression="[Hotels Listing,/admin/hotels.html]|[Hotel Control Panel,/admin/hotels/${hotel.id}/control_panel.html]|[Booking Calendar,]" cascade="true" />
 </definition>
```

```
 <!-- Booking Calendar (Ends) -->
<!--  Admin Pages (Ends) -->
</tiles-definitions>
```

9.8.5 database-dev.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
	xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:jdbc="http://www.springframework.org/schema/jdbc"
	xmlns:p="http://www.springframework.org/schema/p"
	xsi:schemaLocation="http://www.springframework.org/schema/jdbc
http://www.springframework.org/schema/jdbc/spring-jdbc-4.0.xsd
		http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-
beans.xsd">

	<!-- <jdbc:embedded-database type="HSQL" id="dataSource" /> -->

	<bean id="propertyConfigurer"
		class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer"
		p:location="/WEB-INF/jdbc-dev.properties" />

	<bean id="dataSource" class="org.apache.commons.dbcp.BasicDataSource"
		destroy-method="close" p:driverClassName="${jdbc.driverClassName}"
		p:url="${jdbc.databaseurl}" p:username="${jdbc.username}" p:password="${jdbc.password}" />

	<bean
		class="org.springframework.orm.jpa.LocalContainerEntityManagerFactoryBean"
		id="emf">
		<property name="packagesToScan" value="gr.greekchoices.admin.entity" />
		<property name="dataSource" ref="dataSource" />
		<property name="jpaProperties">
			<props>
				<prop key="hibernate.show_sql">true</prop>
				<prop key="hibernate.hbm2ddl.auto">create</prop>
			</props>
		</property>
		<property name="persistenceProvider">
			<bean class="org.hibernate.jpa.HibernatePersistenceProvider" />
		</property>
	</bean>

</beans>
```

9.8.6 web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee"
	xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
	version="3.0">
	<display-name>greekchoices-booking-backoffice</display-name>
	<welcome-file-list>
		<welcome-file>index.html</welcome-file>
		<welcome-file>index.htm</welcome-file>
		<welcome-file>index.jsp</welcome-file>
		<welcome-file>default.html</welcome-file>
		<welcome-file>default.htm</welcome-file>
		<welcome-file>default.jsp</welcome-file>
	</welcome-file-list>
	<servlet>
		<servlet-name>dispatcher</servlet-name>
		<servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
		<load-on-startup>1</load-on-startup>
	</servlet>
	<servlet-mapping>
		<servlet-name>dispatcher</servlet-name>
		<url-pattern>*.html</url-pattern>
		<url-pattern>*.htm</url-pattern>
		<url-pattern>*.json</url-pattern>
		<url-pattern>*.xml</url-pattern>
	</servlet-mapping>
	<listener>
		<listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
	</listener>
	<context-param>
		<param-name>spring.profiles.default</param-name>
		<param-value>prod</param-value>
	</context-param>
	<filter>
		<filter-name>characterEncodingFilter</filter-name>
		<filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>
		<init-param>
			<param-name>encoding</param-name>
			<param-value>UTF-8</param-value>
		</init-param>
		<init-param>
```

```
                <param-name>forceEncoding</param-name>
                <param-value>true</param-value>
            </init-param>
        </filter>
        <filter-mapping>
            <filter-name>characterEncodingFilter</filter-name>
            <url-pattern>/*</url-pattern>
        </filter-mapping>
        <filter>
            <filter-name>springSecurityFilterChain</filter-name>
            <filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>
        </filter>
        <filter-mapping>
            <filter-name>springSecurityFilterChain</filter-name>
            <url-pattern>/*</url-pattern>
        </filter-mapping>
        <context-param>
            <param-name>defaultHtmlEscape</param-name>
            <param-value>true</param-value>
        </context-param>

</web-app>
```

9.8.7 jdbc-dev.properties

```
jdbc.driverClassName=com.mysql.jdbc.Driver
jdbc.dialect=org.hibernate.dialect.MySQLDialect
jdbc.databaseurl=jdbc:mysql://localhost:3306/greekchoices_backend_jpa
jdbc.username=mysql_username
jdbc.password=mysql_password

#These configurations depends on your setup
```

9.8.8 Αρχείο Ρυθμίσεων Καταγραφής (log4j.xml)

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE log4j:configuration SYSTEM "http://logging.apache.org/log4j/1.2/apidocs/org/apache/log4j/xml/doc-files/log4j.dtd">
<log4j:configuration debug="false">
  <appender name="CONSOLE" class="org.apache.log4j.ConsoleAppender">
    <param name="Threshold" value="DEBUG"/>
    <param name="Target" value="System.out"/>
    <layout class="org.apache.log4j.PatternLayout">
      <param name="ConversionPattern" value="%d{ABSOLUTE} %-5p [%c{1}] %m%n"/>
    </layout>
  </appender>
  <appender name="FILE_APPENDER" class="org.apache.log4j.DailyRollingFileAppender">
    <param name="File" value="/home/panos/logs/myapp2.log"/>
    <param name="DatePattern" value="'-'yyyy-MM-dd'.txt'"/>
    <param name="Threshold" value="DEBUG"/>
    <layout class="org.apache.log4j.PatternLayout">
      <param name="ConversionPattern" value="%d{yyyy-MM-dd HH:mm:ss} %-4p [%t] %C{2} - %m%n"/>
    </layout>
  </appender>
  <appender name="syslog" class="LoggerAppenderSyslog">
    <param name="ident" value="greekchoices" />
    <param name="facility" value="LOCAL5" />
    <param name="option" value="NDELAY|PID" />
     <layout class="LoggerLayoutPattern">
    <param name="conversionPattern" value="%logger %-5level %sessionid %location %line %msg%n" />
     </layout>
  </appender>
  <!-- Spring -->
  <logger name="org.springframework" additivity="false">
    <!--
    <level value="DEBUG"/>
    <appender-ref ref="FILE_APPENDER"/>
    -->
    <level value="INFO"/>
    <appender-ref ref="CONSOLE"/>
  </logger>
    <logger name="gr.greekchoices.admin" additivity="false">
    <!-- <level value="INFO"/> -->
    <appender-ref ref="CONSOLE"/>
  </logger>
  <root>
```

```
    <level value="DEBUG"/>
    <appender-ref ref="FILE_APPENDER"/>
  </root>
</log4j:configuration>
```

9.8.9 Αρχείο Ρυθμίσεων για Maven (pom.xml)

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
        <modelVersion>4.0.0</modelVersion>
        <groupId>gr.greekchoices.admin</groupId>
        <artifactId>greekchoices-booking-backoffice</artifactId>
        <version>0.0.1-SNAPSHOT</version>
        <packaging>war</packaging>
        <!-- Shared version number properties -->
        <properties>
                <org.springframework.version>4.0.2.RELEASE</org.springframework.version>
                <apache.tiles>3.0.5</apache.tiles>
                <spring.security.version>4.0.0.RELEASE</spring.security.version>
                 <jackson.databind-version>2.4.1</jackson.databind-version>
        </properties>
        <dependencies>
          <!--  Gson: Java to Json conversion -->
          <dependency>
           <groupId>com.google.code.gson</groupId>
           <artifactId>gson</artifactId>
           <version>2.2.2</version>
           <scope>compile</scope>
          </dependency>
                <dependency>
                        <groupId>org.codehaus.jackson</groupId>
                        <artifactId>jackson-mapper-asl</artifactId>
                        <version>1.9.13</version>
                </dependency>
                <!-- Jackson JSON Mapper -->
    <dependency>
      <groupId>com.fasterxml.jackson.core</groupId>
      <artifactId>jackson-databind</artifactId>
      <version>${jackson.databind-version}</version>
```

```
    </dependency>
                <dependency>
                        <groupId>joda-time</groupId>
                        <artifactId>joda-time</artifactId>
                        <version>2.7</version>
                </dependency>
                <dependency>
                        <groupId>joda-time</groupId>
                        <artifactId>joda-time-hibernate</artifactId>
                        <version>1.4</version>
                </dependency>
                <dependency>
                        <groupId>org.jadira.usertype</groupId>
                        <artifactId>usertype.core</artifactId>
                        <version>3.2.0.GA</version>
                </dependency>
                <dependency>
                        <groupId>joda-time</groupId>
                        <artifactId>joda-time-jsptags</artifactId>
                        <version>1.1.1</version>
                </dependency>
                <dependency>
                        <groupId>org.springframework.security</groupId>
                        <artifactId>spring-security-taglibs</artifactId>
                        <version>4.0.0.RELEASE</version>
                </dependency>
                <dependency>
                        <groupId>org.springframework.security</groupId>
                        <artifactId>spring-security-web</artifactId>
                        <version>${spring.security.version}</version>
                </dependency>
                <dependency>
                        <groupId>org.springframework.security</groupId>
                        <artifactId>spring-security-config</artifactId>
                        <version>${spring.security.version}</version>
                </dependency>
                <dependency>
                        <groupId>jstl</groupId>
                        <artifactId>jstl</artifactId>
                        <version>1.2</version>
                </dependency>
```

```
<dependency>
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
        <version>5.1.6</version>
</dependency>
<dependency>
        <groupId>commons-dbcp</groupId>
        <artifactId>commons-dbcp</artifactId>
        <version>1.4</version>
</dependency>
<dependency>
        <groupId>commons-pool</groupId>
        <artifactId>commons-pool</artifactId>
        <version>1.4</version>
</dependency>
<dependency>
        <groupId>org.hsqldb</groupId>
        <artifactId>hsqldb</artifactId>
        <version>2.3.2</version>
</dependency>
<dependency>
        <groupId>org.springframework.data</groupId>
        <artifactId>spring-data-jpa</artifactId>
        <version>1.7.2.RELEASE</version>
</dependency>
<dependency>
        <groupId>org.hibernate</groupId>
        <artifactId>hibernate-entitymanager</artifactId>
        <version>4.3.4.Final</version>
</dependency>
<dependency>
        <groupId>javax.servlet</groupId>
        <artifactId>javax.servlet-api</artifactId>
        <version>3.0.1</version>
        <scope>provided</scope>
</dependency>
<dependency>
        <groupId>javax.servlet.jsp</groupId>
        <artifactId>javax.servlet.jsp-api</artifactId>
        <version>2.2.1</version>
        <scope>provided</scope>
```

```
</dependency>
<dependency>
        <groupId>org.apache.tiles</groupId>
        <artifactId>tiles-core</artifactId>
        <version>${apache.tiles}</version>
</dependency>
<dependency>
        <groupId>org.apache.tiles</groupId>
        <artifactId>tiles-jsp</artifactId>
        <version>${apache.tiles}</version>
</dependency>
<dependency>
        <groupId>org.apache.tiles</groupId>
        <artifactId>tiles-el</artifactId>
        <version>${apache.tiles}</version>
</dependency>
<dependency>
        <groupId>org.slf4j</groupId>
        <artifactId>slf4j-log4j12</artifactId>
        <version>1.7.10</version>
</dependency>
<!-- Core utilities used by other modules. Define this if you use Spring
        Utility APIs (org.springframework.core.*/org.springframework.util.*) -->
<dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-core</artifactId>
        <version>${org.springframework.version}</version>
</dependency>
<!-- Expression Language (depends on spring-core) Define this if you use
        Spring Expression APIs (org.springframework.expression.*) -->
<dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-expression</artifactId>
        <version>${org.springframework.version}</version>
</dependency>
<!-- Bean Factory and JavaBeans utilities (depends on spring-core) Define
        this if you use Spring Bean APIs (org.springframework.beans.*) -->
<dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-beans</artifactId>
        <version>${org.springframework.version}</version>
```

```
</dependency>
<!-- Aspect Oriented Programming (AOP) Framework (depends on spring-core,
        spring-beans) Define this if you use Spring AOP APIs (org.springframework.aop.*) -->
<dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-aop</artifactId>
        <version>${org.springframework.version}</version>
</dependency>
<!-- Application Context (depends on spring-core, spring-expression, spring-aop,
        spring-beans) This is the central artifact for Spring's Dependency Injection
        Container and is generally always defined -->
<dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-context</artifactId>
        <version>${org.springframework.version}</version>
</dependency>
<!-- Various Application Context utilities, including EhCache, JavaMail,
        Quartz, and Freemarker integration Define this if you need any of these integrations -->
<dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-context-support</artifactId>
        <version>${org.springframework.version}</version>
</dependency>
<!-- Transaction Management Abstraction (depends on spring-core, spring-beans,
        spring-aop, spring-context) Define this if you use Spring Transactions or
        DAO Exception Hierarchy (org.springframework.transaction.*/org.springframework.dao.*) -->
<dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-tx</artifactId>
        <version>${org.springframework.version}</version>
</dependency>
<!-- JDBC Data Access Library (depends on spring-core, spring-beans, spring-context,
        spring-tx) Define this if you use Spring's JdbcTemplate API (org.springframework.jdbc.*) -->
<dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-jdbc</artifactId>
        <version>${org.springframework.version}</version>
</dependency>
<!-- Object-to-Relation-Mapping (ORM) integration with Hibernate, JPA,
        and iBatis. (depends on spring-core, spring-beans, spring-context, spring-tx)
        Define this if you need ORM (org.springframework.orm.*) -->
```

```
<dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-orm</artifactId>
        <version>${org.springframework.version}</version>
</dependency>
<!-- Object-to-XML Mapping (OXM) abstraction and integration with JAXB,
        JiBX, Castor, XStream, and XML Beans. (depends on spring-core, spring-beans,
        spring-context) Define this if you need OXM (org.springframework.oxm.*) -->
<dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-oxm</artifactId>
        <version>${org.springframework.version}</version>
</dependency>
<!-- Web application development utilities applicable to both Servlet and
        Portlet Environments (depends on spring-core, spring-beans, spring-context)
        Define this if you use Spring MVC, or wish to use Struts, JSF, or another
        web framework with Spring (org.springframework.web.*) -->
<dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-web</artifactId>
        <version>${org.springframework.version}</version>
</dependency>
<!-- Spring MVC for Servlet Environments (depends on spring-core, spring-beans,
        spring-context, spring-web) Define this if you use Spring MVC with a Servlet
        Container such as Apache Tomcat (org.springframework.web.servlet.*) -->
<dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-webmvc</artifactId>
        <version>${org.springframework.version}</version>
</dependency>
<!-- Spring MVC for Portlet Environments (depends on spring-core, spring-beans,
        spring-context, spring-web) Define this if you use Spring MVC with a Portlet
        Container (org.springframework.web.portlet.*) -->
<dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-webmvc-portlet</artifactId>
        <version>${org.springframework.version}</version>
</dependency>

<!-- Support for testing Spring applications with tools such as JUnit and
        TestNG This artifact is generally always defined with a 'test' scope for
```

```
                the integration testing framework and unit testing stubs -->
        <dependency>
                <groupId>org.springframework</groupId>
                <artifactId>spring-test</artifactId>
                <version>${org.springframework.version}</version>
                <scope>test</scope>
        </dependency>
    </dependencies>
    <build>
        <plugins>
            <plugin>
                <groupId>org.eclipse.jetty</groupId>
                <artifactId>jetty-maven-plugin</artifactId>
                <version>9.1.3.v20140225</version>
            </plugin>
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-compiler-plugin</artifactId>
                <version>3.1</version>
                <configuration>
                    <source>1.6</source>
                    <target>1.6</target>
                </configuration>
            </plugin>
        </plugins>
    </build>
</project>
```

9.9 ΑΠΑΡΑΙΤΗΤΑ ΣΧΗΜΑΤΑ

Δεδομένου ότι χρησιμοποιείται η τεχνολογία ORM μέσω JPA και JPA annotations και το hibernate ως υλοποιητής (provider), το σχήμα δημιουργείται αυτόματα όταν στο database-dev.xml, όπως βλέπουμε παρακάτω, το property με key “hibernate.hbm2ddl.auto” είναι “create”. Στην παραγωγή αυτό θα άλλαζε σε “update” ώστε να μην διαγράφεται η βάση (drop) και ξανά δημιουργείται.

```
<property name="jpaProperties">
    <props>
        <prop key="hibernate.show_sql">true</prop>
        <prop key="hibernate.hbm2ddl.auto">create</prop>
    </props>
</property>
```

9.9.1 Πίνακας “availability”

```
DROP TABLE IF EXISTS `Availability`;
CREATE TABLE `Availability` (
 `date` date NOT NULL,
 `hotelId` int(11) NOT NULL,
 `roomTypeId` int(11) NOT NULL,
 `quantity` int(11) NOT NULL,
 PRIMARY KEY (`date`,`hotelId`,`roomTypeId`),
 KEY `FK_b18b7qaj22hfhywt6ro892mha` (`hotelId`,`roomTypeId`),
 CONSTRAINT `FK_b18b7qaj22hfhywt6ro892mha` FOREIGN KEY (`hotelId`, `roomTypeId`) REFERENCES `HotelHasRoomType` (`hotelId`, `roomTypeId`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

9.9.2 Πίνακας “booking”

```
DROP TABLE IF EXISTS `Booking`;
CREATE TABLE `Booking` (
 `id` int(11) NOT NULL AUTO_INCREMENT,
 `bookingDate` date DEFAULT NULL,
 `endDate` date DEFAULT NULL,
 `startDate` date DEFAULT NULL,
 `status` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
 `totalCost` decimal(19,2) DEFAULT NULL,
```

```
  `hotelId` int(11) DEFAULT NULL,
  `roomTypeId` int(11) DEFAULT NULL,
  `user_id` int(11) DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `FK_eyo2muei8gyfiemega9hcabf` (`hotelId`,`roomTypeId`),
  KEY `FK_8u559gulj0bgoclx8064ngetf` (`user_id`),
  CONSTRAINT `FK_8u559gulj0bgoclx8064ngetf` FOREIGN KEY (`user_id`) REFERENCES `User` (`id`),
  CONSTRAINT `FK_eyo2muei8gyfiemega9hcabf` FOREIGN KEY (`hotelId`, `roomTypeId`) REFERENCES `HotelHasRoomType` (`hotelId`, `roomTypeId`)
) ENGINE=InnoDB AUTO_INCREMENT=6 DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

9.9.3 Πίνακας "city"

```
DROP TABLE IF EXISTS `City`;
CREATE TABLE `City` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  `tk` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

9.9.4 Πίνακας "facility"

```
DROP TABLE IF EXISTS `Facility`;
CREATE TABLE `Facility` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `global` bit(1) DEFAULT NULL,
  `name` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  `type` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=13 DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

9.9.5 Πίνακας "hotel"

```
DROP TABLE IF EXISTS `Hotel`;
CREATE TABLE `Hotel` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  `user_id` int(11) DEFAULT NULL,
  PRIMARY KEY (`id`),
```

```
  KEY `FK_i67323f38nhmvtxp9phcptriv` (`user_id`),
  CONSTRAINT `FK_i67323f38nhmvtxp9phcptriv` FOREIGN KEY (`user_id`) REFERENCES `User` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

9.9.6 Πίνακας "hotelBasicInfo"

```
DROP TABLE IF EXISTS `HotelBasicInfo`;
CREATE TABLE `HotelBasicInfo` (
  `hotelId` int(11) NOT NULL,
  `address` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  `mobile` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  `telephone` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  `city_id` int(11) DEFAULT NULL,
  `star_id` int(11) DEFAULT NULL,
  PRIMARY KEY (`hotelId`),
  KEY `FK_6rxyby6ut5g0a88u281xt9v1j` (`city_id`),
  KEY `FK_q6yv8k8i9yyyu80ui1ib8kicj` (`star_id`),
  CONSTRAINT `FK_6rxyby6ut5g0a88u281xt9v1j` FOREIGN KEY (`city_id`) REFERENCES `City` (`id`),
  CONSTRAINT `FK_g0775xsfr70b9m8b36ymm5oxh` FOREIGN KEY (`hotelId`) REFERENCES `Hotel` (`id`),
  CONSTRAINT `FK_q6yv8k8i9yyyu80ui1ib8kicj` FOREIGN KEY (`star_id`) REFERENCES `Star` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

9.9.7 Πίνακας "hotelHasRoomType"

```
DROP TABLE IF EXISTS `HotelHasRoomType`;
CREATE TABLE `HotelHasRoomType` (
  `hotelId` int(11) NOT NULL,
  `roomTypeId` int(11) NOT NULL,
  PRIMARY KEY (`hotelId`,`roomTypeId`),
  KEY `FK_8powsrhtq07ls23jqjyhut1s5` (`roomTypeId`),
  CONSTRAINT `FK_8powsrhtq07ls23jqjyhut1s5` FOREIGN KEY (`roomTypeId`) REFERENCES `RoomType` (`id`),
  CONSTRAINT `FK_m6wc9etppik0w9he8xdb786pp` FOREIGN KEY (`hotelId`) REFERENCES `Hotel` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

9.9.8 Πίνακας "period"

```
DROP TABLE IF EXISTS `Period`;
CREATE TABLE `Period` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `endDate` date DEFAULT NULL,
```

```
  `startDate` date DEFAULT NULL,
  `hoteLid` int(11) DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `FK_snos24p6mpnjpvib75y703avv` (`hoteLid`),
  CONSTRAINT `FK_snos24p6mpnjpvib75y703avv` FOREIGN KEY (`hoteLid`) REFERENCES `Hotel` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

9.9.9 Πίνακας "price"

```
DROP TABLE IF EXISTS `Price`;
CREATE TABLE `Price` (
  `hotelId` int(11) NOT NULL,
  `periodId` int(11) NOT NULL,
  `roomTypeId` int(11) NOT NULL,
  `cost` decimal(19,2) DEFAULT NULL,
  PRIMARY KEY (`hotelId`,`periodId`,`roomTypeId`),
  KEY `FK_9n15qs9v15mo7orwjagmf6p5e` (`hotelId`,`roomTypeId`),
  KEY `FK_22p1yfm6vcice71qxdy0r5fnk` (`periodId`),
  CONSTRAINT `FK_22p1yfm6vcice71qxdy0r5fnk` FOREIGN KEY (`periodId`) REFERENCES `Period` (`id`),
  CONSTRAINT `FK_9n15qs9v15mo7orwjagmf6p5e` FOREIGN KEY (`hotelId`, `roomTypeId`) REFERENCES `HotelHasRoomType`
(`hotelId`, `roomTypeId`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

9.9.10 Πίνακας "role"

```
DROP TABLE IF EXISTS `Role`;
CREATE TABLE `Role` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

9.9.11 Πίνακας "roomType"

```
DROP TABLE IF EXISTS `RoomType`;
CREATE TABLE `RoomType` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `color` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  `global` bit(1) DEFAULT NULL,
  `name` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
```

```
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=5 DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

9.9.12 Πίνακας "star"

```
DROP TABLE IF EXISTS `Star`;
CREATE TABLE `Star` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=6 DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

9.9.13 Πίνακας "user"

```
DROP TABLE IF EXISTS `User`;
CREATE TABLE `User` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `email` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  `password` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  `status` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  `username` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=6 DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

9.9.14 Πίνακας "hotel_has_facility"

```
DROP TABLE IF EXISTS `User`;
CREATE TABLE `User` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `email` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  `password` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  `status` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  `username` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=6 DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

9.9.15 Πίνακας "hotel_has_roomtype_has_facility"

```
DROP TABLE IF EXISTS `hotel_has_roomtype_has_facility`;

CREATE TABLE `hotel_has_roomtype_has_facility` (

 `hotelId` int(11) NOT NULL,

 `roomTypeId` int(11) NOT NULL,

 `facilityId` int(11) NOT NULL,

 KEY `FK_1td7lg7fj97vfn9o1v40j8nyb` (`facilityId`),

 KEY `FK_b52omimgbyhf0u629x31kp8ul` (`hotelId`,`roomTypeId`),

 CONSTRAINT `FK_1td7lg7fj97vfn9o1v40j8nyb` FOREIGN KEY (`facilityId`) REFERENCES `Facility` (`id`),

 CONSTRAINT `FK_b52omimgbyhf0u629x31kp8ul` FOREIGN KEY (`hotelId`, `roomTypeId`) REFERENCES `HotelHasRoomType` (`hotelId`, `roomTypeId`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

9.9.16 Πίνακας "user_has_role"

```
DROP TABLE IF EXISTS `user_has_role`;

CREATE TABLE `user_has_role` (

 `user_id` int(11) NOT NULL,

 `role_id` int(11) NOT NULL,

 KEY `FK_konapgay0pjmouer0gkgm3ar0` (`role_id`),

 KEY `FK_6nr7r0p2086fjmmiwkg25s3b7` (`user_id`),

 CONSTRAINT `FK_6nr7r0p2086fjmmiwkg25s3b7` FOREIGN KEY (`user_id`) REFERENCES `User` (`id`),

 CONSTRAINT `FK_konapgay0pjmouer0gkgm3ar0` FOREIGN KEY (`role_id`) REFERENCES `Role` (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

9.10 ΚΩΔΙΚΑΣ

Παρακάτω θα παρουσιαστεί μέρος του κώδικα της εφαρμογής χωρισμένο σε διάφορες κατηγορίες. Ανάλογα με την κατηγορία θα δωθεί μια σύντομη εξήγηση σε σχέση είτε με το Spring Framework, είτε με τα Design Patterns που αυτό χρησιμοποιεί (όπως MVC, IoC κτλ), αλλά και γενικότερα σε σχέση με τα web apps.

9.10.1 Entity

Στο πακέτο "gr.greekchoices.admin.entity" βρίσκονται τα αρχεία πηγαίου κώδικα, που παρουσιάζονται παρακάτω, και έχουν να κάνουν με το μοντέλο (Model), όπως αυτό περιγράφεται από το Design Pattern "MVC" (Model View Controller).

Βάση του JPA και του JPA Annotation (@Entities) αυτά τα αντικείμενα θα γίνουν και οι πίνακες στην βάση δεδομένων μας. Τα JPA Annotation αλλά και όλη η λογική του framework που δεν επιβάλει την ενσωμάτωση άσχετου κώδικα π.χ. extend ή implement κτλ , επιτρέπει σε αυτά τα αρχεία να διατηρούν την απλότητα τους (POJO's - Plain Old Java Object) και να μπορούν να ξαναχρησιμοποιηθούν ακόμη και αν αλλάζαμε framework, ικανοποιόντας την αρχή της χαλαρής διασύνδεσης (Loosly Coupled).

9.10.1.1 Availability.java

```
package gr.greekchoices.admin.entity;
import javax.persistence.Column;
import javax.persistence.EmbeddedId;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.JoinColumns;
import javax.persistence.ManyToOne;
import org.hibernate.annotations.Type;
import org.joda.time.LocalDate;
import org.springframework.format.annotation.DateTimeFormat;
@Entity
public class Availability {
	@EmbeddedId
	protected AvailabilityPK AvailabilityPK;
	private int quantity;
```

```
@ManyToOne
@JoinColumns  ({
   @JoinColumn(name = "hotelId", referencedColumnName = "hotelId", insertable = false, updatable = false),
   @JoinColumn(name = "roomTypeId", referencedColumnName = "roomTypeId", insertable = false, updatable = false)})
private HotelHasRoomType hotelHasRoomType;
        @Type(type = "org.jadira.usertype.dateandtime.joda.PersistentLocalDate")
        @DateTimeFormat(pattern = "dd/MM/yyyy")
        @Column(insertable = false, updatable = false)
        private LocalDate date;
        public AvailabilityPK getAvailabilityPK() {
                return AvailabilityPK;
        }
        public void setAvailabilityPK(AvailabilityPK availabilityPK) {
                AvailabilityPK = availabilityPK;
        }
        public int getQuantity() {
                return quantity;
        }
        public void setQuantity(int quantity) {
                this.quantity = quantity;
        }
        public HotelHasRoomType getHotelHasRoomType() {
                return hotelHasRoomType;
        }
        public void setHotelHasRoomType(HotelHasRoomType hotelHasRoomType) {
                this.hotelHasRoomType = hotelHasRoomType;
        }
        public LocalDate getDate() {
                return date;
        }
        public void setDate(LocalDate date) {
                this.date = date;
        }
        @Override
        public int hashCode() {
                final int prime = 31;
                int result = 1;
                result = prime * result
                                + ((AvailabilityPK == null) ? 0 : AvailabilityPK.hashCode());
                result = prime * result + ((date == null) ? 0 : date.hashCode());
                result = prime
```

```
                                        * result
                                        + ((hotelHasRoomType == null) ? 0 : hotelHasRoomType.hashCode());
                result = prime * result + quantity;
                return result;
        }
        @Override
        public boolean equals(Object obj) {
                if (this == obj)
                        return true;
                if (obj == null)
                        return false;
                if (getClass() != obj.getClass())
                        return false;
                Availability other = (Availability) obj;
                if (AvailabilityPK == null) {
                        if (other.AvailabilityPK != null)
                                return false;
                } else if (!AvailabilityPK.equals(other.AvailabilityPK))
                        return false;
                if (date == null) {
                        if (other.date != null)
                                return false;
                } else if (!date.equals(other.date))
                        return false;
                if (hotelHasRoomType == null) {
                        if (other.hotelHasRoomType != null)
                                return false;
                } else if (!hotelHasRoomType.equals(other.hotelHasRoomType))
                        return false;
                if (quantity != other.quantity)
                        return false;
                return true;
        }
        @Override
        public String toString() {
                return "Availability [AvailabilityPK=" + AvailabilityPK + ", quantity="
                                        + quantity + ", hotelHasRoomType=" + hotelHasRoomType
                                        + ", date=" + date + "]";
        }
}
```

9.10.1.2 AvailabilityPK.java

```
package gr.greekchoices.admin.entity;
import java.io.Serializable;
import java.util.Date;
import javax.persistence.Embeddable;
import org.hibernate.annotations.Type;
import org.joda.time.LocalDate;
import org.springframework.format.annotation.DateTimeFormat;
@Embeddable
public class AvailabilityPK implements Serializable{
	private static final long serialVersionUID = 1L;
	private int hotelId;
	private int roomTypeId;
	@Type(type = "org.jadira.usertype.dateandtime.joda.PersistentLocalDate")
	@DateTimeFormat(pattern = "dd/MM/yyyy")
	private LocalDate date;
	public int getHotelId() {
		return hotelId;
	}
	public void setHotelId(int hotelId) {
		this.hotelId = hotelId;
	}
	public int getRoomTypeId() {
		return roomTypeId;
	}
	public void setRoomTypeId(int roomTypeId) {
		this.roomTypeId = roomTypeId;
	}
	public LocalDate getDate() {
		return date;
	}
	public void setDate(LocalDate date) {
		this.date = date;
	}
	public static long getSerialversionuid() {
		return serialVersionUID;
	}
	@Override
	public int hashCode() {
		final int prime = 31;
		int result = 1;
```

```
                result = prime * result + ((date == null) ? 0 : date.hashCode());
                result = prime * result + hotelId;
                result = prime * result + roomTypeId;
                return result;
        }
        @Override
        public boolean equals(Object obj) {
                if (this == obj)
                        return true;
                if (obj == null)
                        return false;
                if (getClass() != obj.getClass())
                        return false;
                AvailabilityPK other = (AvailabilityPK) obj;
                if (date == null) {
                        if (other.date != null)
                                return false;
                } else if (!date.equals(other.date))
                        return false;
                if (hotelId != other.hotelId)
                        return false;
                if (roomTypeId != other.roomTypeId)
                        return false;
                return true;
        }
        @Override
        public String toString() {
                return "AvailabilityPK [hotelId=" + hotelId + ", roomTypeId="
                                + roomTypeId + ", date=" + date + "]";
        }
}
```

9.10.1.3 Booking.java

```
package gr.greekchoices.admin.entity;
import java.math.BigDecimal;
import java.util.Date;
import javax.persistence.Entity;
import javax.persistence.EnumType;
import javax.persistence.Enumerated;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.JoinColumns;
import javax.persistence.ManyToOne;
import org.hibernate.annotations.Type;
import org.joda.time.LocalDate;
import org.springframework.format.annotation.DateTimeFormat;
@Entity
public class Booking {
	@Id
	@GeneratedValue
	private Integer id;
	@Type(type = "org.jadira.usertype.dateandtime.joda.PersistentLocalDate")
	@DateTimeFormat(pattern = "dd/MM/yyyy")
	private LocalDate bookingDate;
	@Type(type = "org.jadira.usertype.dateandtime.joda.PersistentLocalDate")
	@DateTimeFormat(pattern = "dd/MM/yyyy")
	private LocalDate startDate;
	@Type(type = "org.jadira.usertype.dateandtime.joda.PersistentLocalDate")
	@DateTimeFormat(pattern = "dd/MM/yyyy")
	private LocalDate endDate;
	private BigDecimal totalCost;
	@Enumerated(EnumType.STRING)
	private BookingStatus status;
	@ManyToOne
	@JoinColumn(name="user_id")
	private User user;
	@ManyToOne
	@JoinColumns({
		@JoinColumn(name="hotelId" , referencedColumnName="hotelId"),
		@JoinColumn(name="roomTypeId", referencedColumnName="roomTypeId")
	})
	private HotelHasRoomType hotelHasRoomType;
```

```
public Integer getId() {
		return id;
}
public void setId(Integer id) {
		this.id = id;
}
public LocalDate getBookingDate() {
		return bookingDate;
}
public void setBookingDate(LocalDate bookingDate) {
		this.bookingDate = bookingDate;
}
public LocalDate getStartDate() {
		return startDate;
}
public void setStartDate(LocalDate startDate) {
		this.startDate = startDate;
}
public LocalDate getEndDate() {
		return endDate;
}
public void setEndDate(LocalDate endDate) {
		this.endDate = endDate;
}
public BigDecimal getTotalCost() {
		return totalCost;
}
public void setTotalCost(BigDecimal totalCost) {
		this.totalCost = totalCost;
}
public BookingStatus getStatus() {
		return status;
}
public void setStatus(BookingStatus status) {
		this.status = status;
}
public User getUser() {
		return user;
}
public void setUser(User user) {
		this.user = user;
```

```
	}
	public HotelHasRoomType getHotelHasRoomType() {
		return hotelHasRoomType;
	}
	public void setHotelHasRoomType(HotelHasRoomType hotelHasRoomType) {
		this.hotelHasRoomType = hotelHasRoomType;
	}
	@Override
	public int hashCode() {
		final int prime = 31;
		int result = 1;
		result = prime * result
				+ ((bookingDate == null) ? 0 : bookingDate.hashCode());
		result = prime * result + ((endDate == null) ? 0 : endDate.hashCode());
		result = prime
				* result
				+ ((hotelHasRoomType == null) ? 0 : hotelHasRoomType.hashCode());
		result = prime * result + ((id == null) ? 0 : id.hashCode());
		result = prime * result
				+ ((startDate == null) ? 0 : startDate.hashCode());
		result = prime * result + ((status == null) ? 0 : status.hashCode());
		result = prime * result
				+ ((totalCost == null) ? 0 : totalCost.hashCode());
		result = prime * result + ((user == null) ? 0 : user.hashCode());
		return result;
	}
	@Override
	public boolean equals(Object obj) {
		if (this == obj)
			return true;
		if (obj == null)
			return false;
		if (getClass() != obj.getClass())
			return false;
		Booking other = (Booking) obj;
		if (bookingDate == null) {
			if (other.bookingDate != null)
				return false;
		} else if (!bookingDate.equals(other.bookingDate))
			return false;
		if (endDate == null) {
```

```
                        if (other.endDate != null)
                                return false;
                } else if (!endDate.equals(other.endDate))
                        return false;
                if (hotelHasRoomType == null) {
                        if (other.hotelHasRoomType != null)
                                return false;
                } else if (!hotelHasRoomType.equals(other.hotelHasRoomType))
                        return false;
                if (id == null) {
                        if (other.id != null)
                                return false;
                } else if (!id.equals(other.id))
                        return false;
                if (startDate == null) {
                        if (other.startDate != null)
                                return false;
                } else if (!startDate.equals(other.startDate))
                        return false;
                if (status != other.status)
                        return false;
                if (totalCost == null) {
                        if (other.totalCost != null)
                                return false;
                } else if (!totalCost.equals(other.totalCost))
                        return false;
                if (user == null) {
                        if (other.user != null)
                                return false;
                } else if (!user.equals(other.user))
                        return false;
                return true;
        }
        @Override
        public String toString() {
                return "Booking [id=" + id + ", bookingDate=" + bookingDate
                                + ", startDate=" + startDate + ", endDate=" + endDate
                                + ", totalCost=" + totalCost + ", status=" + status + ", user="
                                + user + ", hotelHasRoomType=" + hotelHasRoomType + "]";
        }
}
```

9.10.1.4 **Booking.java**

```
package gr.greekchoices.admin.entity;
import java.math.BigDecimal;
import java.util.Date;
import javax.persistence.Entity;
import javax.persistence.EnumType;
import javax.persistence.Enumerated;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.JoinColumns;
import javax.persistence.ManyToOne;
import org.hibernate.annotations.Type;
import org.joda.time.LocalDate;
import org.springframework.format.annotation.DateTimeFormat;
@Entity
public class Booking {
	@Id
	@GeneratedValue
	private Integer id;
	@Type(type = "org.jadira.usertype.dateandtime.joda.PersistentLocalDate")
	@DateTimeFormat(pattern = "dd/MM/yyyy")
	private LocalDate bookingDate;
	@Type(type = "org.jadira.usertype.dateandtime.joda.PersistentLocalDate")
	@DateTimeFormat(pattern = "dd/MM/yyyy")
	private LocalDate startDate;
	@Type(type = "org.jadira.usertype.dateandtime.joda.PersistentLocalDate")
	@DateTimeFormat(pattern = "dd/MM/yyyy")
	private LocalDate endDate;
	private BigDecimal totalCost;
	@Enumerated(EnumType.STRING)
	private BookingStatus status;
	@ManyToOne
	@JoinColumn(name="user_id")
	private User user;
	@ManyToOne
	@JoinColumns({
		@JoinColumn(name="hotelId" , referencedColumnName="hotelId"),
		@JoinColumn(name="roomTypeId", referencedColumnName="roomTypeId")
	})
	private HotelHasRoomType hotelHasRoomType;
```

```
public Integer getId() {
		return id;
}
public void setId(Integer id) {
		this.id = id;
}
public LocalDate getBookingDate() {
		return bookingDate;
}
public void setBookingDate(LocalDate bookingDate) {
		this.bookingDate = bookingDate;
}
public LocalDate getStartDate() {
		return startDate;
}
public void setStartDate(LocalDate startDate) {
		this.startDate = startDate;
}
public LocalDate getEndDate() {
		return endDate;
}
public void setEndDate(LocalDate endDate) {
		this.endDate = endDate;
}
public BigDecimal getTotalCost() {
		return totalCost;
}
public void setTotalCost(BigDecimal totalCost) {
		this.totalCost = totalCost;
}
public BookingStatus getStatus() {
		return status;
}
public void setStatus(BookingStatus status) {
		this.status = status;
}
public User getUser() {
		return user;
}
public void setUser(User user) {
		this.user = user;
```

```
	}
	public HotelHasRoomType getHotelHasRoomType() {
		return hotelHasRoomType;
	}
	public void setHotelHasRoomType(HotelHasRoomType hotelHasRoomType) {
		this.hotelHasRoomType = hotelHasRoomType;
	}
	@Override
	public int hashCode() {
		final int prime = 31;
		int result = 1;
		result = prime * result
				+ ((bookingDate == null) ? 0 : bookingDate.hashCode());
		result = prime * result + ((endDate == null) ? 0 : endDate.hashCode());
		result = prime
				* result
				+ ((hotelHasRoomType == null) ? 0 : hotelHasRoomType.hashCode());
		result = prime * result + ((id == null) ? 0 : id.hashCode());
		result = prime * result
				+ ((startDate == null) ? 0 : startDate.hashCode());
		result = prime * result + ((status == null) ? 0 : status.hashCode());
		result = prime * result
				+ ((totalCost == null) ? 0 : totalCost.hashCode());
		result = prime * result + ((user == null) ? 0 : user.hashCode());
		return result;
	}
	@Override
	public boolean equals(Object obj) {
		if (this == obj)
			return true;
		if (obj == null)
			return false;
		if (getClass() != obj.getClass())
			return false;
		Booking other = (Booking) obj;
		if (bookingDate == null) {
			if (other.bookingDate != null)
				return false;
		} else if (!bookingDate.equals(other.bookingDate))
			return false;
		if (endDate == null) {
```

```
                        if (other.endDate != null)
                                return false;
                } else if (!endDate.equals(other.endDate))
                        return false;
                if (hotelHasRoomType == null) {
                        if (other.hotelHasRoomType != null)
                                return false;
                } else if (!hotelHasRoomType.equals(other.hotelHasRoomType))
                        return false;
                if (id == null) {
                        if (other.id != null)
                                return false;
                } else if (!id.equals(other.id))
                        return false;
                if (startDate == null) {
                        if (other.startDate != null)
                                return false;
                } else if (!startDate.equals(other.startDate))
                        return false;
                if (status != other.status)
                        return false;
                if (totalCost == null) {
                        if (other.totalCost != null)
                                return false;
                } else if (!totalCost.equals(other.totalCost))
                        return false;
                if (user == null) {
                        if (other.user != null)
                                return false;
                } else if (!user.equals(other.user))
                        return false;
                return true;
        }
        @Override
        public String toString() {
                return "Booking [id=" + id + ", bookingDate=" + bookingDate
                                + ", startDate=" + startDate + ", endDate=" + endDate
                                + ", totalCost=" + totalCost + ", status=" + status + ", user="
                                + user + ", hotelHasRoomType=" + hotelHasRoomType + "]";
        }
}
```

9.10.1.5 City.java

```
package gr.greekchoices.admin.entity;
import java.util.List;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.OneToMany;
@Entity
public class City {
	@Id
	@GeneratedValue
	private Integer id;
	private String name;
	private String tk;
	@OneToMany(mappedBy="city")
	private List<HotelBasicInfo> hotelBasicInfoList;
	public Integer getId() {
		return id;
	}
	public void setId(Integer id) {
		this.id = id;
	}
	public String getName() {
		return name;
	}
	public void setName(String name) {
		this.name = name;
	}
	public String getTk() {
		return tk;
	}
	public void setTk(String tk) {
		this.tk = tk;
	}
	public List<HotelBasicInfo> getHotelBasicInfoList() {
		return hotelBasicInfoList;
	}
	public void setHotelBasicInfoList(List<HotelBasicInfo> hotelBasicInfoList) {
		this.hotelBasicInfoList = hotelBasicInfoList;
	}
	@Override
```

```
	public int hashCode() {
		final int prime = 31;
		int result = 1;
		result = prime
				* result
				+ ((hotelBasicInfoList == null) ? 0 : hotelBasicInfoList
						.hashCode());
		result = prime * result + ((id == null) ? 0 : id.hashCode());
		result = prime * result + ((name == null) ? 0 : name.hashCode());
		result = prime * result + ((tk == null) ? 0 : tk.hashCode());
		return result;
	}
	@Override
	public boolean equals(Object obj) {
		if (this == obj)
			return true;
		if (obj == null)
			return false;
		if (getClass() != obj.getClass())
			return false;
		City other = (City) obj;
		if (hotelBasicInfoList == null) {
			if (other.hotelBasicInfoList != null)
				return false;
		} else if (!hotelBasicInfoList.equals(other.hotelBasicInfoList))
			return false;
		if (id == null) {
			if (other.id != null)
				return false;
		} else if (!id.equals(other.id))
			return false;
		if (name == null) {
			if (other.name != null)
				return false;
		} else if (!name.equals(other.name))
			return false;
		if (tk == null) {
			if (other.tk != null)
				return false;
		} else if (!tk.equals(other.tk))
			return false;
```

```
                return true;
        }
        @Override
        public String toString() {
                return "City [id=" + id + ", name=" + name + ", tk=" + tk
                                + ", hotelBasicInfoList=" + hotelBasicInfoList + "]";
        }
}
```

9.10.1.6 **Facility.java**

```
package gr.greekchoices.admin.entity;
import java.util.List;
import javax.persistence.CascadeType;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.ManyToMany;
import javax.persistence.NamedQuery;
import javax.persistence.Table;
@Entity
@NamedQuery(name = "Facility.findByType", query = "SELECT f FROM Facility f WHERE f.type=?1 order by f.name ASC")
@Table(name = "Facility")
public class Facility {
        @Id
        @GeneratedValue
        private Integer id;
        private String name;
        private String type;
        private Boolean global;
        @ManyToMany(mappedBy="facilityList")
        private List<Hotel> hotelList;
        @ManyToMany(mappedBy="facilityList")
        private List<HotelHasRoomType> hotelHasRoomTypeList;
        public Integer getId() {
                return id;
        }
```

```
public void setId(Integer id) {
		this.id = id;
}
public String getName() {
		return name;
}
public void setName(String name) {
		this.name = name;
}
public String getType() {
		return type;
}
public void setType(String type) {
		this.type = type;
}
public Boolean getGlobal() {
		return global;
}
public void setGlobal(Boolean global) {
		this.global = global;
}
public List<Hotel> getHotelList() {
		return hotelList;
}
public void setHotelList(List<Hotel> hotelList) {
		this.hotelList = hotelList;
}
public List<HotelHasRoomType> getHotelHasRoomTypeList() {
		return hotelHasRoomTypeList;
}
public void setHotelHasRoomTypeList(List<HotelHasRoomType> hotelHasRoomTypeList) {
		this.hotelHasRoomTypeList = hotelHasRoomTypeList;
}
@Override
public int hashCode() {
		final int prime = 31;
		int result = 1;
		result = prime * result + ((global == null) ? 0 : global.hashCode());
		result = prime
				* result
				+ ((hotelHasRoomTypeList == null) ? 0 : hotelHasRoomTypeList
```

```
                                                        .hashCode());
                result = prime * result
                                + ((hotelList == null) ? 0 : hotelList.hashCode());
                result = prime * result + ((id == null) ? 0 : id.hashCode());
                result = prime * result + ((name == null) ? 0 : name.hashCode());
                result = prime * result + ((type == null) ? 0 : type.hashCode());
                return result;
        }
        @Override
        public boolean equals(Object obj) {
                if (this == obj)
                        return true;
                if (obj == null)
                        return false;
                if (getClass() != obj.getClass())
                        return false;
                Facility other = (Facility) obj;
                if (global == null) {
                        if (other.global != null)
                                return false;
                } else if (!global.equals(other.global))
                        return false;
/*              if (hotelHasRoomTypeList == null) {
                        if (other.hotelHasRoomTypeList != null)
                                return false;
                } else if (!hotelHasRoomTypeList.equals(other.hotelHasRoomTypeList))
                        return false;*/
/*              if (hotelList == null) {
                        if (other.hotelList != null)
                                return false;
                } else if (!hotelList.equals(other.hotelList))
                        return false;*/
                if (id == null) {
                        if (other.id != null)
                                return false;
                } else if (!id.equals(other.id))
                        return false;
                if (name == null) {
                        if (other.name != null)
                                return false;
                } else if (!name.equals(other.name))
```

```
                return false;
            if (type == null) {
                if (other.type != null)
                    return false;
            } else if (!type.equals(other.type))
                return false;
            return true;
        }
        @Override
        public String toString() {
            return "Facility [id=" + id + ", name=" + name + ", type=" + type
                    + ", global=" + global + ", hotelList=" + hotelList
                    + ", hotelHasRoomTypeList=" + hotelHasRoomTypeList + "]";
        }
}
```

9.10.1.7 **Hotel.java**

```
package gr.greekchoices.admin.entity;
import java.util.List;
import javax.persistence.Basic;
import javax.persistence.CascadeType;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.JoinTable;
import javax.persistence.ManyToMany;
import javax.persistence.ManyToOne;
import javax.persistence.OneToMany;
import javax.persistence.OneToOne;
```

```
@Entity
public class Hotel {
	@Id
	@GeneratedValue(strategy = GenerationType.IDENTITY)
	@Basic(optional = false)
	private Integer id;
	private String name;
	@ManyToOne
	@JoinColumn(name="user_id")
	private User user;
	/*@ManyToMany(cascade=CascadeType.ALL)*/
	@ManyToMany
	@JoinTable(name = "hotel_has_facility",
	  joinColumns = { @JoinColumn(name = "hotel_id", referencedColumnName = "id") },
	  inverseJoinColumns = { @JoinColumn(name = "facility_id" , referencedColumnName="id") })
	private List<Facility> facilityList;
	@OneToMany(mappedBy="hotel",cascade=CascadeType.ALL)
	private List<HotelHasRoomType> hotelHasRoomTypeList;
	@OneToOne(cascade = CascadeType.ALL, mappedBy = "hotel")
	private HotelBasicInfo hotelBasicInfo;
	@OneToMany(mappedBy="hotel",cascade=CascadeType.ALL)
	private List<Period> periodList;
	public Integer getId() {
		return id;
	}
	public void setId(Integer id) {
		this.id = id;
	}
	public String getName() {
		return name;
	}
	public void setName(String name) {
		this.name = name;
	}
	public User getUser() {
		return user;
	}
	public void setUser(User user) {
		this.user = user;
	}
	public List<Facility> getFacilityList() {
```

```
		return facilityList;
	}
	public void setFacilityList(List<Facility> facilityList) {
		this.facilityList = facilityList;
	}
	public List<HotelHasRoomType> getHotelHasRoomTypeList() {
		return hotelHasRoomTypeList;
	}
	public void setHotelHasRoomTypeList(List<HotelHasRoomType> hotelHasRoomTypeList) {
		this.hotelHasRoomTypeList = hotelHasRoomTypeList;
	}
	public HotelBasicInfo getHotelBasicInfo() {
		return hotelBasicInfo;
	}
	public void setHotelBasicInfo(HotelBasicInfo hotelBasicInfo) {
		this.hotelBasicInfo = hotelBasicInfo;
	}
	public List<Period> getPeriodList() {
		return periodList;
	}
	public void setPeriodList(List<Period> periodList) {
		this.periodList = periodList;
	}
	@Override
	public int hashCode() {
		final int prime = 31;
		int result = 1;
		result = prime * result
				+ ((facilityList == null) ? 0 : facilityList.hashCode());
		result = prime * result
				+ ((hotelBasicInfo == null) ? 0 : hotelBasicInfo.hashCode());
		result = prime
				* result
				+ ((hotelHasRoomTypeList == null) ? 0 : hotelHasRoomTypeList
						.hashCode());
		result = prime * result + ((id == null) ? 0 : id.hashCode());
		result = prime * result + ((name == null) ? 0 : name.hashCode());
		result = prime * result
				+ ((periodList == null) ? 0 : periodList.hashCode());
		result = prime * result + ((user == null) ? 0 : user.hashCode());
		return result;
```

```
	}
	@Override
	public boolean equals(Object obj) {
		if (this == obj)
			return true;
		if (obj == null)
			return false;
		if (getClass() != obj.getClass())
			return false;
		Hotel other = (Hotel) obj;
		if (facilityList == null) {
			if (other.facilityList != null)
				return false;
		} else if (!facilityList.equals(other.facilityList))
			return false;
		if (hotelBasicInfo == null) {
			if (other.hotelBasicInfo != null)
				return false;
		} else if (!hotelBasicInfo.equals(other.hotelBasicInfo))
			return false;
		if (hotelHasRoomTypeList == null) {
			if (other.hotelHasRoomTypeList != null)
				return false;
		} else if (!hotelHasRoomTypeList.equals(other.hotelHasRoomTypeList))
			return false;
		if (id == null) {
			if (other.id != null)
				return false;
		} else if (!id.equals(other.id))
			return false;
		if (name == null) {
			if (other.name != null)
				return false;
		} else if (!name.equals(other.name))
			return false;
		if (periodList == null) {
			if (other.periodList != null)
				return false;
		} else if (!periodList.equals(other.periodList))
			return false;
		if (user == null) {
```

```
                                if (other.user != null)
                                        return false;
                        } else if (!user.equals(other.user))
                                return false;
                        return true;
                }
                @Override
                public String toString() {
                        return "Hotel [id=" + id + ", name=" + name + ", user=" + user
                                        + ", facilityList=" + facilityList + ", hotelHasRoomTypeList="
                                        + hotelHasRoomTypeList + ", hotelBasicInfo=" + hotelBasicInfo
                                        + ", periodList=" + periodList + "]";
                }
}
```

9.10.1.8 HotelBasicInfo.java

```
package gr.greekchoices.admin.entity;
import javax.persistence.Basic;
import javax.persistence.CascadeType;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.MapsId;
import javax.persistence.OneToOne;
@Entity
public class HotelBasicInfo {
        @Id
        @Basic(optional = false)
        private Integer hotelId;
        private String address;
        private String telephone;
        private String mobile;
   @MapsId
   @JoinColumn(name = "hotelId", referencedColumnName = "id", insertable = false, updatable = false)
   @OneToOne(optional = false)
```

```
private Hotel hotel;
	@ManyToOne
	@JoinColumn(name="star_id")
	private Star star;
	@ManyToOne
	@JoinColumn(name="city_id")
	private City city;
	public String getAddress() {
		return address;
	}
	public void setAddress(String address) {
		this.address = address;
	}
	public String getTelephone() {
		return telephone;
	}
	public void setTelephone(String telephone) {
		this.telephone = telephone;
	}
	public String getMobile() {
		return mobile;
	}
	public void setMobile(String mobile) {
		this.mobile = mobile;
	}
	public Integer getHotelId() {
		return hotelId;
	}
	public void setHotelId(Integer hotelId) {
		this.hotelId = hotelId;
	}
	public Star getStar() {
		return star;
	}
	public void setStar(Star star) {
		this.star = star;
	}
	public City getCity() {
		return city;
	}
	public void setCity(City city) {
```

```
		this.city = city;
	}
	public Hotel getHotel() {
		return hotel;
	}
	public void setHotel(Hotel hotel) {
		this.hotel = hotel;
	}
	@Override
	public int hashCode() {
		final int prime = 31;
		int result = 1;
		result = prime * result + ((address == null) ? 0 : address.hashCode());
		result = prime * result + ((city == null) ? 0 : city.hashCode());
		result = prime * result + ((hotel == null) ? 0 : hotel.hashCode());
		result = prime * result + ((hotelId == null) ? 0 : hotelId.hashCode());
		result = prime * result + ((mobile == null) ? 0 : mobile.hashCode());
		result = prime * result + ((star == null) ? 0 : star.hashCode());
		result = prime * result
				+ ((telephone == null) ? 0 : telephone.hashCode());
		return result;
	}
	@Override
	public boolean equals(Object obj) {
		if (this == obj)
			return true;
		if (obj == null)
			return false;
		if (getClass() != obj.getClass())
			return false;
		HotelBasicInfo other = (HotelBasicInfo) obj;
		if (address == null) {
			if (other.address != null)
				return false;
		} else if (!address.equals(other.address))
			return false;
		if (city == null) {
			if (other.city != null)
				return false;
		} else if (!city.equals(other.city))
			return false;
```

```
                if (hotel == null) {
                        if (other.hotel != null)
                                return false;
                } else if (!hotel.equals(other.hotel))
                        return false;
                if (hotelId == null) {
                        if (other.hotelId != null)
                                return false;
                } else if (!hotelId.equals(other.hotelId))
                        return false;
                if (mobile == null) {
                        if (other.mobile != null)
                                return false;
                } else if (!mobile.equals(other.mobile))
                        return false;
                if (star == null) {
                        if (other.star != null)
                                return false;
                } else if (!star.equals(other.star))
                        return false;
                if (telephone == null) {
                        if (other.telephone != null)
                                return false;
                } else if (!telephone.equals(other.telephone))
                        return false;
                return true;
        }
        @Override
        public String toString() {
                return "HotelBasicInfo [hotelId=" + hotelId + ", address=" + address
                                + ", telephone=" + telephone + ", mobile=" + mobile
                                + ", hotel=" + hotel + ", star=" + star + ", city=" + city
                                + "]";
        }
}
```

9.10.1.9 HotelHasRoomType.java

```
package gr.greekchoices.admin.entity;


import java.util.List;


import javax.persistence.CascadeType;
import javax.persistence.EmbeddedId;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.JoinColumns;
import javax.persistence.JoinTable;
import javax.persistence.ManyToMany;
import javax.persistence.ManyToOne;
import javax.persistence.OneToMany;


@Entity
public class HotelHasRoomType {
		@EmbeddedId
		protected HotelHasRoomTypePK hotelHasRoomTypePK;
		@ManyToOne
		@JoinColumn(name="hotelId", insertable = false, updatable = false)
		private Hotel hotel;
		@ManyToOne
		@JoinColumn(name="roomTypeId", insertable = false, updatable = false)
		private RoomType roomType;
		@ManyToMany
  @JoinTable(name="hotel_has_roomtype_has_facility",
		joinColumns = {
					@JoinColumn(name="hotelId" , referencedColumnName="hotelId"),
					@JoinColumn(name="roomTypeId", referencedColumnName="roomTypeId")
			},
			inverseJoinColumns = {
					@JoinColumn(name="facilityId" , referencedColumnName ="id")
			}
  )
  private List<Facility> facilityList;
  @OneToMany(mappedBy="hotelHasRoomType",cascade=CascadeType.REMOVE)
		private List<Price> priceList;
		@OneToMany(mappedBy="hotelHasRoomType",cascade=CascadeType.REMOVE)
		private List<Availability> AvailabilityList;
```

```
@OneToMany(mappedBy="hotelHasRoomType",cascade=CascadeType.REMOVE)
private List<Booking> bookingList;
public HotelHasRoomTypePK getHotelHasRoomTypePK() {
        return hotelHasRoomTypePK;
}
public void setHotelHasRoomTypePK(HotelHasRoomTypePK hotelHasRoomTypePK) {
        this.hotelHasRoomTypePK = hotelHasRoomTypePK;
}
public Hotel getHotel() {
        return hotel;
}
public void setHotel(Hotel hotel) {
        this.hotel = hotel;
}
public RoomType getRoomType() {
        return roomType;
}
public void setRoomType(RoomType roomType) {
        this.roomType = roomType;
}
public List<Facility> getFacilityList() {
        return facilityList;
}
public void setFacilityList(List<Facility> facilityList) {
        this.facilityList = facilityList;
}
public List<Price> getPriceList() {
        return priceList;
}
public void setPriceList(List<Price> priceList) {
        this.priceList = priceList;
}
public List<Availability> getAvailabilityList() {
        return AvailabilityList;
}
public void setAvailabilityList(List<Availability> AvailabilityList) {
        this.AvailabilityList = AvailabilityList;
}
public List<Booking> getBookingList() {
        return bookingList;
}
```

```
	public void setBookingList(List<Booking> bookingList) {
		this.bookingList = bookingList;
	}
	@Override
	public int hashCode() {
		final int prime = 31;
		int result = 1;
		result = prime
				* result
				+ ((AvailabilityList == null) ? 0 : AvailabilityList.hashCode());
		result = prime * result
				+ ((bookingList == null) ? 0 : bookingList.hashCode());
		result = prime * result
				+ ((facilityList == null) ? 0 : facilityList.hashCode());
		result = prime * result + ((hotel == null) ? 0 : hotel.hashCode());
		result = prime
				* result
				+ ((hotelHasRoomTypePK == null) ? 0 : hotelHasRoomTypePK
						.hashCode());
		result = prime * result
				+ ((priceList == null) ? 0 : priceList.hashCode());
		result = prime * result
				+ ((roomType == null) ? 0 : roomType.hashCode());
		return result;
	}
	@Override
	public boolean equals(Object obj) {
		if (this == obj)
			return true;
		if (obj == null)
			return false;
		if (getClass() != obj.getClass())
			return false;
		HotelHasRoomType other = (HotelHasRoomType) obj;
		if (AvailabilityList == null) {
			if (other.AvailabilityList != null)
				return false;
		} else if (!AvailabilityList.equals(other.AvailabilityList))
			return false;
		if (bookingList == null) {
			if (other.bookingList != null)
```

```
                                        return false;
                        } else if (!bookingList.equals(other.bookingList))
                                return false;
                        if (facilityList == null) {
                                if (other.facilityList != null)
                                        return false;
                        } else if (!facilityList.equals(other.facilityList))
                                return false;
                        if (hotel == null) {
                                if (other.hotel != null)
                                        return false;
                        } else if (!hotel.equals(other.hotel))
                                return false;
                        if (hotelHasRoomTypePK == null) {
                                if (other.hotelHasRoomTypePK != null)
                                        return false;
                        } else if (!hotelHasRoomTypePK.equals(other.hotelHasRoomTypePK))
                                return false;
                        if (priceList == null) {
                                if (other.priceList != null)
                                        return false;
                        } else if (!priceList.equals(other.priceList))
                                return false;
                        if (roomType == null) {
                                if (other.roomType != null)
                                        return false;
                        } else if (!roomType.equals(other.roomType))
                                return false;
                        return true;
                }
                @Override
                public String toString() {
                        return "HotelHasRoomType [hotelHasRoomTypePK=" + hotelHasRoomTypePK
                                        + ", hotel=" + hotel + ", roomType=" + roomType
                                        + ", facilityList=" + facilityList + ", priceList=" + priceList
                                        + ", AvailabilityList=" + AvailabilityList + ", bookingList="
                                        + bookingList + "]";
                }
}
```

9.10.1.10 HotelHasRoomTypePK.java

```
package gr.greekchoices.admin.entity;
import java.io.Serializable;
import javax.persistence.Embeddable;
@Embeddable
public class HotelHasRoomTypePK implements Serializable {
	private static final long serialVersionUID = 1L;
	private int hotelId;
	private int roomTypeId;
	public int getHotelId() {
		return hotelId;
	}
	public void setHotelId(int hotelId) {
		this.hotelId = hotelId;
	}
	public int getRoomTypeId() {
		return roomTypeId;
	}
	public void setRoomTypeId(int roomTypeId) {
		this.roomTypeId = roomTypeId;
	}
	@Override
	public int hashCode() {
		final int prime = 31;
		int result = 1;
		result = prime * result + hotelId;
		result = prime * result + roomTypeId;
		return result;
	}
	@Override
	public boolean equals(Object obj) {
		if (this == obj)
			return true;
		if (obj == null)
			return false;
		if (getClass() != obj.getClass())
			return false;
		HotelHasRoomTypePK other = (HotelHasRoomTypePK) obj;
		if (hotelId != other.hotelId)
			return false;
		if (roomTypeId != other.roomTypeId)
```

```
                                        return false;
                        return true;
                }
                @Override
                public String toString() {
                        return "HotelHasRoomTypePK [hotelId=" + hotelId + ", roomTypeId="
                                        + roomTypeId + "]";
                }
}
```

9.10.1.11 Period.java

```
package gr.greekchoices.admin.entity;
import java.util.Date;
import java.util.List;
import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.OneToMany;
import org.hibernate.annotations.Type;
import org.joda.time.LocalDate;
import org.springframework.format.annotation.DateTimeFormat;
@Entity
public class Period {
                @Id
                @GeneratedValue
                private Integer id;
                @Type(type = "org.jadira.usertype.dateandtime.joda.PersistentLocalDate")
                @DateTimeFormat(pattern = "dd/MM/yyyy")
                private LocalDate startDate;
                @Type(type = "org.jadira.usertype.dateandtime.joda.PersistentLocalDate")
                @DateTimeFormat(pattern = "dd/MM/yyyy")
                private LocalDate endDate;
                @ManyToOne
                @JoinColumn(name="hoteLid")
                private Hotel hotel;
                @OneToMany(mappedBy="period",cascade=CascadeType.REMOVE)
```

```
	private List<Price> priceList;
	public Integer getId() {
		return id;
	}
	public void setId(Integer id) {
		this.id = id;
	}
	public LocalDate getStartDate() {
		return startDate;
	}
	public void setStartDate(LocalDate startDate) {
		this.startDate = startDate;
	}
	public LocalDate getEndDate() {
		return endDate;
	}
	public void setEndDate(LocalDate endDate) {
		this.endDate = endDate;
	}
	public Hotel getHotel() {
		return hotel;
	}
	public void setHotel(Hotel hotel) {
		this.hotel = hotel;
	}
	public List<Price> getPriceList() {
		return priceList;
	}
	public void setPriceList(List<Price> priceList) {
		this.priceList = priceList;
	}
	@Override
	public int hashCode() {
		final int prime = 31;
		int result = 1;
		result = prime * result + ((endDate == null) ? 0 : endDate.hashCode());
		result = prime * result + ((hotel == null) ? 0 : hotel.hashCode());
		result = prime * result + ((id == null) ? 0 : id.hashCode());
		result = prime * result
				+ ((priceList == null) ? 0 : priceList.hashCode());
		result = prime * result
```

```
                                        + ((startDate == null) ? 0 : startDate.hashCode());
                return result;
        }
        @Override
        public boolean equals(Object obj) {
                if (this == obj)
                        return true;
                if (obj == null)
                        return false;
                if (getClass() != obj.getClass())
                        return false;
                Period other = (Period) obj;
                if (endDate == null) {
                        if (other.endDate != null)
                                return false;
                } else if (!endDate.equals(other.endDate))
                        return false;
                if (hotel == null) {
                        if (other.hotel != null)
                                return false;
                } else if (!hotel.equals(other.hotel))
                        return false;
                if (id == null) {
                        if (other.id != null)
                                return false;
                } else if (!id.equals(other.id))
                        return false;
                if (priceList == null) {
                        if (other.priceList != null)
                                return false;
                } else if (!priceList.equals(other.priceList))
                        return false;
                if (startDate == null) {
                        if (other.startDate != null)
                                return false;
                } else if (!startDate.equals(other.startDate))
                        return false;
                return true;
        }
        @Override
        public String toString() {
```

```
                return "Period [id=" + id + ", startDate=" + startDate + ", endDate="
                                + endDate + ", hotel=" + hotel + ", priceList=" + priceList
                                + "]";
        }
}
```

9.10.1.12 Price.java

```
package gr.greekchoices.admin.entity;
import java.math.BigDecimal;
import javax.persistence.EmbeddedId;
import javax.persistence.Entity;
import javax.persistence.JoinColumn;
import javax.persistence.JoinColumns;
import javax.persistence.JoinTable;
import javax.persistence.ManyToMany;
import javax.persistence.ManyToOne;
@Entity
public class Price {
        @EmbeddedId
        protected PricePK pricePK;
        private BigDecimal cost;
        @ManyToOne
        @JoinColumns({
                @JoinColumn(name="hotelId" , referencedColumnName="hotelId", insertable = false, updatable = false),
                @JoinColumn(name="roomTypeId", referencedColumnName="roomTypeId", insertable = false, updatable =
false)
        })
        private HotelHasRoomType hotelHasRoomType;
        @ManyToOne
        @JoinColumn(name="periodId", insertable = false, updatable = false)
        private Period period;
        public BigDecimal getCost() {
                return cost;
        }
        public void setCost(BigDecimal cost) {
                this.cost = cost;
        }
        public PricePK getPricePK() {
```

```
		return pricePK;
	}
	public void setPricePK(PricePK pricePK) {
		this.pricePK = pricePK;
	}
	public HotelHasRoomType getHotelHasRoomType() {
		return hotelHasRoomType;
	}
	public void setHotelHasRoomType(HotelHasRoomType hotelHasRoomType) {
		this.hotelHasRoomType = hotelHasRoomType;
	}
	public Period getPeriod() {
		return period;
	}
	public void setPeriod(Period period) {
		this.period = period;
	}
	@Override
	public int hashCode() {
		final int prime = 31;
		int result = 1;
		result = prime * result + ((cost == null) ? 0 : cost.hashCode());
		result = prime
				* result
				+ ((hotelHasRoomType == null) ? 0 : hotelHasRoomType.hashCode());
		result = prime * result + ((period == null) ? 0 : period.hashCode());
		result = prime * result + ((pricePK == null) ? 0 : pricePK.hashCode());
		return result;
	}
	@Override
	public boolean equals(Object obj) {
		if (this == obj)
			return true;
		if (obj == null)
			return false;
		if (getClass() != obj.getClass())
			return false;
		Price other = (Price) obj;
		if (cost == null) {
			if (other.cost != null)
				return false;
```

```
                } else if (!cost.equals(other.cost))
                        return false;
                if (hotelHasRoomType == null) {
                        if (other.hotelHasRoomType != null)
                                return false;
                } else if (!hotelHasRoomType.equals(other.hotelHasRoomType))
                        return false;
                if (period == null) {
                        if (other.period != null)
                                return false;
                } else if (!period.equals(other.period))
                        return false;
                if (pricePK == null) {
                        if (other.pricePK != null)
                                return false;
                } else if (!pricePK.equals(other.pricePK))
                        return false;
                return true;
        }
        @Override
        public String toString() {
                return "Price [pricePK=" + pricePK + ", cost=" + cost
                                + ", hotelHasRoomType=" + hotelHasRoomType + ", period="
                                + period + "]";
        }
}
```

9.10.1.13 PricePK.java

```
package gr.greekchoices.admin.entity;
import java.io.Serializable;
import javax.persistence.Embeddable;
@Embeddable
public class PricePK implements Serializable {
	private int hotelId;
	private int roomTypeId;
	private int periodId;
	public int getHotelId() {
		return hotelId;
	}
	public void setHotelId(int hotelId) {
		this.hotelId = hotelId;
	}

	public int getRoomTypeId() {
		return roomTypeId;
	}
	public void setRoomTypeId(int roomTypeId) {
		this.roomTypeId = roomTypeId;
	}
	public int getPeriodId() {
		return periodId;
	}
	public void setPeriodId(int periodId) {
		this.periodId = periodId;
	}
	@Override
	public int hashCode() {
		final int prime = 31;
		int result = 1;
		result = prime * result + hotelId;
		result = prime * result + periodId;
		result = prime * result + roomTypeId;
		return result;
	}
	@Override
	public boolean equals(Object obj) {
		if (this == obj)
			return true;
```

```
                if (obj == null)
                        return false;
                if (getClass() != obj.getClass())
                        return false;
                PricePK other = (PricePK) obj;
                if (hotelId != other.hotelId)
                        return false;
                if (periodId != other.periodId)
                        return false;
                if (roomTypeId != other.roomTypeId)
                        return false;
                return true;
        }
        @Override
        public String toString() {
                return "PricePK [hotelId=" + hotelId + ", roomTypeId=" + roomTypeId
                                + ", periodId=" + periodId + "]";
        }
}
```

9.10.1.14 Role.java

```
package gr.greekchoices.admin.entity;
import java.util.List;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.ManyToMany;
@Entity
public class Role {
        @Id
        @GeneratedValue
        private Integer id;
        private String name;
        @ManyToMany(mappedBy = "roleList")
        private List<User> userList;
        public Integer getId() {
                return id;
        }
```

```
public void setId(Integer id) {
		this.id = id;
}
public String getName() {
		return name;
}

public void setName(String name) {
		this.name = name;
}
public List<User> getUserList() {
		return userList;
}
public void setUserList(List<User> userList) {
		this.userList = userList;
}
@Override
public int hashCode() {
		final int prime = 31;
		int result = 1;
		result = prime * result + ((id == null) ? 0 : id.hashCode());
		result = prime * result + ((name == null) ? 0 : name.hashCode());
		result = prime * result
				+ ((userList == null) ? 0 : userList.hashCode());
		return result;
}
@Override
public boolean equals(Object obj) {
		if (this == obj)
			return true;
		if (obj == null)
			return false;
		if (getClass() != obj.getClass())
			return false;
		Role other = (Role) obj;
		if (id == null) {
			if (other.id != null)
				return false;
		} else if (!id.equals(other.id))
			return false;
		if (name == null) {
```

```
                                        if (other.name != null)
                                                return false;
                                } else if (!name.equals(other.name))
                                        return false;
                                if (userList == null) {
                                        if (other.userList != null)
                                                return false;
                                } else if (!userList.equals(other.userList))
                                        return false;
                                return true;
                        }
                        @Override
                        public String toString() {
                                return "Role [id=" + id + ", name=" + name + ", userList=" + userList
                                                + "]";
                        }
}
```

9.10.1.15 RoomType.java

```
package gr.greekchoices.admin.entity;
import java.util.List;
import javax.persistence.CascadeType;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.OneToMany;
@Entity
public class RoomType {
        @Id
        @GeneratedValue
        private Integer id;
        private String name;
        private Boolean global;
        private String color;
        @OneToMany(mappedBy="roomType",cascade=CascadeType.REMOVE)
        private List<HotelHasRoomType> hotelHasRoomTypeList;
```

```
	public Integer getId() {
		return id;
	}
	public void setId(Integer id) {
		this.id = id;
	}
	public String getName() {
		return name;
	}
	public void setName(String name) {
		this.name = name;
	}
	public Boolean getGlobal() {
		return global;
	}
	public void setGlobal(Boolean global) {
		this.global = global;
	}
	public String getColor() {
		return color;
	}
	public void setColor(String color) {
		this.color = color;
	}
	public List<HotelHasRoomType> getHotelHasRoomTypeList() {
		return hotelHasRoomTypeList;
	}
	public void setHotelHasRoomTypeList(List<HotelHasRoomType> hotelHasRoomTypeList) {
		this.hotelHasRoomTypeList = hotelHasRoomTypeList;
	}
	@Override
	public int hashCode() {
		final int prime = 31;
		int result = 1;
		result = prime * result + ((color == null) ? 0 : color.hashCode());
		result = prime * result + ((global == null) ? 0 : global.hashCode());
		result = prime
				* result
				+ ((hotelHasRoomTypeList == null) ? 0 : hotelHasRoomTypeList
						.hashCode());
		result = prime * result + ((id == null) ? 0 : id.hashCode());
```

```
		result = prime * result + ((name == null) ? 0 : name.hashCode());
		return result;
	}
	@Override
	public boolean equals(Object obj) {
		if (this == obj)
			return true;
		if (obj == null)
			return false;
		if (getClass() != obj.getClass())
			return false;
		RoomType other = (RoomType) obj;
		if (color == null) {
			if (other.color != null)
				return false;
		} else if (!color.equals(other.color))
			return false;
		if (global == null) {
			if (other.global != null)
				return false;
		} else if (!global.equals(other.global))
			return false;
		/*if (hotelHasRoomTypeList == null) {
			if (other.hotelHasRoomTypeList != null)
				return false;
		} else if (!hotelHasRoomTypeList.equals(other.hotelHasRoomTypeList))
			return false;*/
		if (id == null) {
			if (other.id != null)
				return false;
		} else if (!id.equals(other.id))
			return false;
		if (name == null) {
			if (other.name != null)
				return false;
		} else if (!name.equals(other.name))
			return false;
		return true;
	}
	@Override
	public String toString() {
```

```
                return "RoomType [id=" + id + ", name=" + name + ", global=" + global
                                + ", color=" + color + ", hotelHasRoomTypeList="
                                + hotelHasRoomTypeList + "]";
        }
}
```

9.10.1.16 Star.java

```
package gr.greekchoices.admin.entity;
import java.util.List;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.OneToMany;
@Entity
public class Star {
        @Id
        @GeneratedValue
        private Integer id;
        private String name;
        @OneToMany(mappedBy="star")
        private List<HotelBasicInfo> hotelBasicInfoList;
        public Integer getId() {
                return id;
        }
        public void setId(Integer id) {
                this.id = id;
        }
        public String getName() {
                return name;
        }
        public void setName(String name) {
                this.name = name;
        }
        public List<HotelBasicInfo> getHotelBasicInfoList() {
                return hotelBasicInfoList;
        }
```

```
public void setHotelBasicInfoList(List<HotelBasicInfo> hotelBasicInfoList) {
		this.hotelBasicInfoList = hotelBasicInfoList;
}
@Override
public int hashCode() {
		final int prime = 31;
		int result = 1;
		result = prime
				* result
				+ ((hotelBasicInfoList == null) ? 0 : hotelBasicInfoList
						.hashCode());
		result = prime * result + ((id == null) ? 0 : id.hashCode());
		result = prime * result + ((name == null) ? 0 : name.hashCode());
		return result;
}
@Override
public boolean equals(Object obj) {
		if (this == obj)
			return true;
		if (obj == null)
			return false;
		if (getClass() != obj.getClass())
			return false;
		Star other = (Star) obj;
		if (hotelBasicInfoList == null) {
			if (other.hotelBasicInfoList != null)
				return false;
		} else if (!hotelBasicInfoList.equals(other.hotelBasicInfoList))
			return false;
		if (id == null) {
			if (other.id != null)
				return false;
		} else if (!id.equals(other.id))
			return false;
		if (name == null) {
			if (other.name != null)
				return false;
		} else if (!name.equals(other.name))
			return false;
		return true;
}
```

```
	@Override
	public String toString() {
		return "Star [id=" + id + ", name=" + name + ", hotelBasicInfoList="
				+ hotelBasicInfoList + "]";
	}
}
```

9.10.1.17 User.java

```
package gr.greekchoices.admin.entity;
import java.util.List;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.JoinTable;
import javax.persistence.ManyToMany;
import javax.persistence.OneToMany;
@Entity
public class User {
	@Id
	@GeneratedValue
	private Integer id;
	private String username;
	private String password;
	private String email;
	private String status;
	@ManyToMany
	@JoinTable(name = "user_has_role",
				joinColumns = { @JoinColumn(name = "user_id", referencedColumnName = "id") },
				inverseJoinColumns = { @JoinColumn(name = "role_id" , referencedColumnName="id") })
	private List<Role> roleList;
	@OneToMany(mappedBy="user")
	private List<Hotel> hotelList;
	@OneToMany(mappedBy="user")
	private List<Booking> bookingList;
	@OneToMany(mappedBy="user")
```

```
private List<Blog> blogs;
@OneToMany(mappedBy="user")
private List<Log> logList;
public Integer getId() {
        return id;
}
public void setId(Integer id) {
        this.id = id;
}
public String getUsername() {
        return username;
}
public void setUsername(String username) {
        this.username = username;
}
public String getPassword() {
        return password;
}
public void setPassword(String password) {
        this.password = password;
}
public String getEmail() {
        return email;
}
public void setEmail(String email) {
        this.email = email;
}
public String getStatus() {
        return status;
}
public void setStatus(String status) {
        this.status = status;
}
public List<Role> getRoleList() {
        return roleList;
}
public void setRoleList(List<Role> roleList) {
        this.roleList = roleList;
}
public List<Hotel> getHotelList() {
        return hotelList;
```

```
	}
	public void setHotelList(List<Hotel> hotelList) {
		this.hotelList = hotelList;
	}
	public List<Booking> getBookingList() {
		return bookingList;
	}
	public void setBookingList(List<Booking> bookingList) {
		this.bookingList = bookingList;
	}
		public List<Blog> getBlogs() {
			return blogs;
		}
		public void setBlogs(List<Blog> blogs) {
			this.blogs = blogs;
		}
		@Override
		public int hashCode() {
			final int prime = 31;
			int result = 1;
			result = prime * result + ((blogs == null) ? 0 : blogs.hashCode());
			result = prime * result
					+ ((bookingList == null) ? 0 : bookingList.hashCode());
			result = prime * result + ((email == null) ? 0 : email.hashCode());
			result = prime * result
					+ ((hotelList == null) ? 0 : hotelList.hashCode());
			result = prime * result + ((id == null) ? 0 : id.hashCode());
			result = prime * result
					+ ((password == null) ? 0 : password.hashCode());
			result = prime * result
					+ ((roleList == null) ? 0 : roleList.hashCode());
			result = prime * result
					+ ((status == null) ? 0 : status.hashCode());
			result = prime * result
					+ ((username == null) ? 0 : username.hashCode());
			return result;
		}
		@Override
		public boolean equals(Object obj) {
			if (this == obj)
				return true;
```

```
if (obj == null)
		return false;
if (getClass() != obj.getClass())
		return false;
User other = (User) obj;
if (blogs == null) {
		if (other.blogs != null)
			return false;
} else if (!blogs.equals(other.blogs))
		return false;
if (bookingList == null) {
		if (other.bookingList != null)
			return false;
} else if (!bookingList.equals(other.bookingList))
		return false;
if (email == null) {
		if (other.email != null)
			return false;
} else if (!email.equals(other.email))
		return false;
if (hotelList == null) {
		if (other.hotelList != null)
			return false;
} else if (!hotelList.equals(other.hotelList))
		return false;
if (id == null) {
		if (other.id != null)
			return false;
} else if (!id.equals(other.id))
		return false;
if (password == null) {
		if (other.password != null)
			return false;
} else if (!password.equals(other.password))
		return false;
if (roleList == null) {
		if (other.roleList != null)
			return false;
} else if (!roleList.equals(other.roleList))
		return false;
if (status == null) {
```

```
                                        if (other.status != null)
                                                return false;
                                } else if (!status.equals(other.status))
                                        return false;
                                if (username == null) {
                                        if (other.username != null)
                                                return false;
                                } else if (!username.equals(other.username))
                                        return false;
                                return true;
                        }
                        @Override
                        public String toString() {
                                return "User [id=" + id + ", username=" + username + ", password="
                                                + password + ", email=" + email + ", status=" + status
                                                + ", roleList=" + roleList + ", hotelList=" + hotelList
                                                + ", bookingList=" + bookingList + ", blogs=" + blogs + "]";
                        }
}
```

9.10.2 Controller

Στο πακέτο "gr.greekchoices.admin.controller" βρίσκονται τα αρχεία πηγαίου κώδικα, που παρουσιάζονται παρακάτω, και έχουν να κάνουν με τον ελεγκτή (Controller), όπως αυτό περιγράφεται από το Design Pattern "MVC" (Model View Controller).

Βάση του JPA και του JPA Annotation (@Controller) αυτά τα αντικείμενα είναι οι ελεγκτές για συγκεκριμένες διευθύνσεις(URLs) ή τύπους διευθύνσεων (URL Patterns) μέσω του annotation @RequestMapping("/[url-path]"). Όταν εντοπίζεται από το Spring Framework ένα τέτοιο path στο Request Object, ο έλεγχος του προγράμματος οδηγείται στην μέθοδο που έχει το συγκεκριμένο annotation και από εκεί..αφού γίνουν οι όποιες απαραίτητες διαδικασίες θα προωθηθεί ο έλεγχος στο Response μέσω κάποιου View αφού έχει εμπλουτιστεί ο dispatcher με το μοντέλο, βάση των ενδεχόμενων παραμέτρων που έχουν σταλεί.

9.10.2.1 AvailabilityController.java

```
package gr.greekchoices.admin.controller;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import gr.greekchoices.admin.entity.HotelHasRoomType;
import gr.greekchoices.admin.entity.RoomType;
import gr.greekchoices.admin.model.AvailabilityCalendarData;
import gr.greekchoices.admin.repository.HotelRepository;
import gr.greekchoices.admin.repository.RoomTypeRepository;
import gr.greekchoices.admin.service.AvailabilityService;
import gr.greekchoices.admin.service.RoomTypeService;

import org.apache.log4j.Logger;
import org.joda.time.LocalDate;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
```

```
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.bind.annotation.ResponseStatus;
import com.fasterxml.jackson.core.JsonFactory;
import com.fasterxml.jackson.databind.AnnotationIntrospector.ReferenceProperty.Type;
import com.fasterxml.jackson.databind.ObjectMapper;
import org.codehaus.jackson.type.TypeReference;
import com.google.gson.Gson;
import com.google.gson.reflect.TypeToken;
import java.io.File;
import java.io.IOException;
import org.codehaus.jackson.JsonGenerationException;
import org.codehaus.jackson.map.JsonMappingException;
@Controller
public class AvailabilityController {
	private static final Logger logger = Logger
				.getLogger(AvailabilityController.class);
	@Autowired
	private HotelRepository hotelRepository;
	@Autowired
	private RoomTypeService roomTypeService;
	@Autowired
	private RoomTypeRepository roomTypeRepository;
	@Autowired
	private AvailabilityService availabilityService;
	// Availabilities Calendar
	@RequestMapping("/admin/hotels/{hotelId}/availabilities-calendar/all")
	public String availabilitiesCalendarPerAllRoomType(Model model,@PathVariable int hotelId){
		model.addAttribute("hotel", hotelRepository.findOne(hotelId));
		model.addAttribute("allRoomTypePerHotel",roomTypeService.allRoomTypePerHotel(hotelId));
		return "availabilitiesCalendarPerAllRoomType";
	}
	// Availabilities Calendar
	@RequestMapping("/admin/hotels/{hotelId}/availabilities-calendar/{roomTypeId}")
	public String availabilitiesCalendarPerRoomType(Model model,@PathVariable int hotelId,@PathVariable int roomTypeId){
		model.addAttribute("hotel", hotelRepository.findOne(hotelId));
		model.addAttribute("roomType", roomTypeRepository.findOne(roomTypeId));
		model.addAttribute("allRoomTypePerHotel",roomTypeService.allRoomTypePerHotel(hotelId));
		return "availabilitiesCalendarPerRoomType";
	}
	// Availability Calendar Ajax Read (Starts)
```

```
	@RequestMapping(value = "/admin/hotels/{hotelId}/availabilities-calendar/{roomTypeId}/dataurl", method = RequestMethod.POST)
	public @ResponseBody Map<LocalDate, AvailabilityCalendarData> availabilitiesCalendarPerRoomTypeAjaxDataUrl(Model model,@PathVariable int hotelId,@PathVariable int roomTypeId) {
			logger.debug("DataUrl");
			logger.debug("hotelId:"+hotelId+" | roomTypeId:"+roomTypeId);
			Map<LocalDate, AvailabilityCalendarData> availabilitiesCalendarMapToJson = availabilityService.availabilitiesCalendarMapToJson(roomTypeId,hotelId);
			return availabilitiesCalendarMapToJson;
	}
	@RequestMapping(value = "/admin/hotels/{hotelId}/availabilities-calendar/{roomTypeId}/saveurl", method = RequestMethod.POST)
	@ResponseStatus(value = HttpStatus.OK)
	public void availabilitiesCalendarPerRoomTypeAjaxSaveUrl(Model model,@PathVariable int hotelId,@PathVariable int roomTypeId,@RequestParam("dopbcp_schedule") String dopbcp_schedule) {
			logger.debug("SaveUrl");
			logger.debug("hotelId:"+hotelId+" | roomTypeId:"+roomTypeId);
			Gson gson = new Gson();
			java.lang.reflect.Type stringStringMap = new TypeToken<Map<String, AvailabilityCalendarData>>(){}.getType();
			Map<String,AvailabilityCalendarData> availabilityCalendarDataHashMap = gson.fromJson(dopbcp_schedule, stringStringMap);
			availabilityService.saveAll(availabilityCalendarDataHashMap, roomTypeId ,hotelId);
	}
	// Availability Calendar Ajax Read (Ends)
}
```

9.10.2.2 BookingController.java

```
package gr.greekchoices.admin.controller;
import gr.greekchoices.admin.entity.Facility;
import gr.greekchoices.admin.repository.HotelRepository;
import gr.greekchoices.admin.service.BookingService;
import gr.greekchoices.admin.service.RoomTypeService;
import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
```

```
@Controller
public class BookingController {
	private static final Logger logger = Logger
					.getLogger(BookingController.class);
	@Autowired
	private HotelRepository hotelRepository;
	@Autowired
	private BookingService bookingService;
	@Autowired
	private RoomTypeService roomTypeService;
	// Booking Listing
	@RequestMapping("/admin/hotels/{hotelId}/booking/booking-listing")
	public String hotelFacilities(Model model,@PathVariable int hotelId){
		model.addAttribute("hotel", hotelRepository.findOne(hotelId));
		model.addAttribute("allBookingPerAllRoomTypePerHotel",
bookingService.findAllBookingPerAllRoomTypeByHotelId(hotelId));
		return "bookingListing";
	}
	// Booking Calendar
	@RequestMapping("/admin/hotels/{hotelId}/booking/booking-calendar")
	public String roomFacilities(Model model,@PathVariable int hotelId){
		model.addAttribute("hotel", hotelRepository.findOne(hotelId));
		model.addAttribute("allRoomTypeByHotelId",roomTypeService.allRoomTypePerHotel(hotelId));
		model.addAttribute("allBookingPerAllRoomTypePerHotel",
bookingService.findAllBookingPerAllRoomTypeByHotelId(hotelId));
		return "bookingCalendar";
	}
}
```

9.10.2.3 EventController.java

```
package gr.greekchoices.admin.controller;
import java.util.ArrayList;
import java.util.List;
import java.util.Locale;
import gr.greekchoices.Constants;
import gr.greekchoices.admin.entity.Booking;
import gr.greekchoices.admin.entity.Shop;
import gr.greekchoices.admin.model.Event;
import gr.greekchoices.admin.repository.HotelRepository;
import gr.greekchoices.admin.service.BookingService;
import org.apache.log4j.Logger;
import org.joda.time.DateTimeZone;
import org.joda.time.LocalDate;
import org.joda.time.format.DateTimeFormat;
import org.joda.time.format.DateTimeFormatter;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.ResponseBody;
@Controller
public class EventController implements Constants{
	private static final Logger logger = Logger
				.getLogger(EventController.class);
	private static DateTimeFormatter eventFormater = DateTimeFormat.forPattern(PATTERN_EVENT_DAY);
	@Autowired
	private  HotelRepository hotelRepository;
	@Autowired
	private BookingService bookingService;
	// Booking Listing
	@RequestMapping(value="/admin/hotels/{hotelId}/events", method = RequestMethod.GET, produces = "application/json")
	@ResponseBody
	public List<Event> hotelEvent(@PathVariable int hotelId){
		ArrayList<Event> eventList = new ArrayList<Event>();
		List<Booking> bookingList = bookingService.findAllBookingPerAllRoomTypeByHotelId(hotelId);
		for (Booking booking: bookingList){
			eventList.add(new Event(booking.getUser().getUsername(),  booking.getStartDate(),
booking.getEndDate(), booking.getHotelHasRoomType().getRoomType().getColor(), "http://google.com/"));
```

```
                        }
                        return eventList;
                }
}
```

9.10.2.4 FacilityController.java

```
package gr.greekchoices.admin.controller;
import java.security.Principal;
import java.util.HashMap;
import gr.greekchoices.admin.entity.Facility;
import gr.greekchoices.admin.entity.Hotel;
import gr.greekchoices.admin.entity.RoomType;
import gr.greekchoices.admin.repository.FacilityRepository;
import gr.greekchoices.admin.repository.HotelRepository;
import gr.greekchoices.admin.repository.RoomTypeRepository;
import gr.greekchoices.admin.service.FacilityService;
import gr.greekchoices.admin.service.HotelService;
import gr.greekchoices.admin.service.RoomTypeService;
import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RequestParam;
@Controller
public class FacilityController {
        private static final Logger logger = Logger
                                .getLogger(FacilityController.class);
        @Autowired
        private HotelService hotelService;
```

```
@Autowired
private HotelRepository hotelRepository;
@Autowired
private FacilityService facilityService;
@Autowired
private FacilityRepository facilityRepository;
@Autowired
private RoomTypeService roomTypeService;
@Autowired
private RoomTypeRepository roomTypeRepository;
@ModelAttribute
public Facility constructFacility(){
        return new Facility();
}
@RequestMapping("/admin/hotels/{hotelId}/facilities/hotel-facilities")
public String hotelFacilities(Model model,@PathVariable int hotelId){
        model.addAttribute("hotel", hotelRepository.findOne(hotelId));
        model.addAttribute("allHotelFacilityByHotelId",facilityService.findAllHotelFacilityByHotelId(hotelId));
        model.addAttribute("allGlobalHotelFacilityToAdd",facilityService.findAllGlobalHotelFacilityToAdd(hotelId));
        return "hotelFacilitiesListing";
}
@RequestMapping(value="/admin/hotels/{hotelId}/facilities/hotel-facilities",method=RequestMethod.POST)
public String doAddHotelFacility(@ModelAttribute("facility") Facility facility , @PathVariable int hotelId, Principal principal){
        String username = principal.getName();
        facilityService.save(facility , username , hotelId);
        return "redirect:/admin/hotels/{hotelId}/facilities/hotel-facilities.html";
}
//Room Facilities Listing//
@RequestMapping("/admin/hotels/{hotelId}/facilities/room-facilities")
public String roomFacilities(Model model,@PathVariable int hotelId){
        model.addAttribute("hotel", hotelRepository.findOne(hotelId));
        model.addAttribute("allRoomFacilityByHotelId",facilityService.findAllRoomFacilityByHotelId(hotelId));
        model.addAttribute("allGlobalRoomFacilityToAdd",facilityService.findAllGlobalRoomFacilityToAdd(hotelId));
        return "roomFacilitiesListing";
}
@RequestMapping(value="/admin/hotels/{hotelId}/facilities/room-facilities",method=RequestMethod.POST)
public String doAddRoomFacility(@ModelAttribute("facility") Facility facility , @PathVariable int hotelId, Principal principal){
        String username = principal.getName();
        facilityService.save(facility , username , hotelId);
        return "redirect:/admin/hotels/{hotelId}/facilities/room-facilities.html";
}
```

```
	//Room Facilities Per Room Type All
	@RequestMapping("/admin/hotels/{hotelId}/facilities/room-facilities/per-roomtype/all")
	public String roomFacilitiesPerRoomTypeAll(Model model,@PathVariable int hotelId){
		model.addAttribute("hotel", hotelRepository.findOne(hotelId));
		model.addAttribute("allRoomTypePerHotel",roomTypeService.allRoomTypePerHotel(hotelId));
	model.addAttribute("allAvaliableRoomFacilityPerAllRoomTypePerHotel",facilityService.allAvaliableRoomFacilityPerAllRoom
TypePerHotel(hotelId));
	model.addAttribute("allRoomFacilityPerAllRoomTypePerHotel",facilityService.allRoomFacilityPerAllRoomTypePerHotel(hote
lId));
		return "roomFacilitiesPerRoomTypeListing";
	}
	@RequestMapping(value="/admin/hotels/{hotelId}/facilities/room-facilities/per-
roomtype/all",method=RequestMethod.POST)
	public String doAddRoomFacilityPerRoomTypeAll(@RequestParam(value="roomTypeId", required=false) int
roomTypeId,@RequestParam(value="to", required=false) int[] roomTypeFacilitiesArray,  @PathVariable int hotelId, Principal principal){
		String username = principal.getName();
		logger.debug("Updating Room Facilities  (roomTypeId:"+roomTypeId+")");
		facilityService.updateRoomFacilities( roomTypeFacilitiesArray, roomTypeId , hotelId);
		return "redirect:/admin/hotels/{hotelId}/facilities/room-facilities/per-roomtype/all.html";
	}
	//Room Facilities per Specific Room Types (ok)
	@RequestMapping("/admin/hotels/{hotelId}/facilities/room-facilities/per-roomtype/{roomTypeId}")
	public String roomFacilitiesPerRoomType(Model model,@PathVariable int hotelId,@PathVariable int roomTypeId){
		model.addAttribute("hotel", hotelRepository.findOne(hotelId));
		model.addAttribute("roomType", roomTypeRepository.findOne(roomTypeId));
		model.addAttribute("allRoomTypePerHotel",roomTypeService.findAllRoomTypeByHotelId(hotelId));
	model.addAttribute("allAvailableRoomFacilityPerRoomTypePerHotel",facilityService.allAvailableRoomFacilityPerRoomType
PerHotel(hotelId,roomTypeId));
	model.addAttribute("allRoomFacilityPerRoomTypePerHotel",facilityService.allRoomFacilityPerRoomTypePerHotel(hotelId,r
oomTypeId));
		return "roomFacilitiesPerRoomType";
	}
	@RequestMapping(value="/admin/hotels/{hotelId}/facilities/room-facilities/per-
roomtype/{roomTypeId}",method=RequestMethod.POST)
	public String doAddRoomFacilityPerRoomType(@RequestParam(value="to", required=false) int[] roomTypeFacilitiesArray,
@PathVariable int roomTypeId,  @PathVariable int hotelId, Principal principal){
		String username = principal.getName();
		logger.debug("Updating Room Facilities  (roomTypeId:"+roomTypeId+")");
		facilityService.updateRoomFacilities( roomTypeFacilitiesArray, roomTypeId , hotelId);
		return "redirect:/admin/hotels/{hotelId}/facilities/room-facilities/per-roomtype/{roomTypeId}.html";
	}
	@RequestMapping(value="/admin/hotels/{hotelId}/facilities/hotel-facility/{facilityId}/edit")
	public String hotelFacilityEdit(Model model, @PathVariable int facilityId, @PathVariable int hotelId){
		model.addAttribute("hotel", hotelRepository.findOne(hotelId));
		model.addAttribute("facility", facilityRepository.findOne(facilityId));
```

```
                return "hotelFacilityEdit";
        }
        @RequestMapping(value="/admin/hotels/{hotelId}/facilities/hotel-facility/{facilityId}/edit",method=RequestMethod.POST)
        public String doUpdateHotelFacility(@ModelAttribute("facility") Facility facility , @PathVariable int hotelId, Principal
principal){
                logger.debug("Update Hotel Facility ->"+facility.getId());
                logger.debug("Update Hotel Facility ->"+facility.getName());
                String username = principal.getName();
                facilityService.save(facility , username , hotelId);
                return "redirect:/admin/hotels/{hotelId}/facilities/hotel-facilities.html";
        }
        @RequestMapping(value="/admin/hotels/{hotelId}/facilities/hotel-facility/{facilityId}/remove")
        public String removeHotelFacility(@PathVariable int facilityId,@PathVariable int hotelId){
                Hotel hotel = hotelRepository.findOne(hotelId);
                Facility facility = facilityRepository.findOne(facilityId);
                facilityService.remove(facility,hotel);
                return "redirect:/admin/hotels/{hotelId}/facilities/hotel-facilities.html";
        }
        @RequestMapping(value="/admin/hotels/{hotelId}/facilities/room-facility/{facilityId}/edit")
        public String roomFacilityEdit(Model model, @PathVariable int facilityId, @PathVariable int hotelId){
                model.addAttribute("hotel", hotelRepository.findOne(hotelId));
                model.addAttribute("facility", facilityRepository.findOne(facilityId));
                return "roomFacilityEdit";
        }
        @RequestMapping(value="/admin/hotels/{hotelId}/facilities/room-facility/{facilityId}/edit",method=RequestMethod.POST)
        public String doUpdateRoomFacility(@ModelAttribute("facility") Facility facility , @PathVariable int hotelId, Principal
principal){
                logger.debug("Update Room Facility ->"+facility.getId());
                logger.debug("Update Room Facility ->"+facility.getName());
                String username = principal.getName();
                facilityService.save(facility , username , hotelId);
                return "redirect:/admin/hotels/{hotelId}/facilities/room-facilities.html";
        }
        @RequestMapping(value="/admin/hotels/{hotelId}/facilities/room-facility/{facilityId}/remove")
        public String removeRoomFacility(@PathVariable int facilityId,@PathVariable int hotelId){
                Hotel hotel = hotelRepository.findOne(hotelId);
                Facility facility = facilityRepository.findOne(facilityId);
                facilityService.remove(facility,hotel);
                return "redirect:/admin/hotels/{hotelId}/facilities/room-facilities.html";
        }
}
```

9.10.2.5 HotelController.java

```
package gr.greekchoices.admin.controller;
import java.security.Principal;
import java.util.List;
import gr.greekchoices.admin.entity.City;
import gr.greekchoices.admin.entity.Hotel;
import gr.greekchoices.admin.entity.HotelBasicInfo;
import gr.greekchoices.admin.entity.Period;
import gr.greekchoices.admin.model.CombinedHotelHotelBasicInfoCommand;
import gr.greekchoices.admin.repository.CityRepository;
import gr.greekchoices.admin.repository.HotelBasicInfoRepository;
import gr.greekchoices.admin.repository.HotelRepository;
import gr.greekchoices.admin.service.CityService;
import gr.greekchoices.admin.service.HotelBasicInfoService;
import gr.greekchoices.admin.service.HotelService;
import gr.greekchoices.admin.service.StarService;
import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.access.method.P;
import org.springframework.security.access.prepost.PreAuthorize;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
@Controller
public class HotelController {
        private static final Logger logger = Logger
                        .getLogger(HotelController.class);
        @Autowired
        private HotelService hotelService;
        @Autowired
        private CityService cityService;
        @Autowired
        private StarService starService;
        @Autowired
        private HotelRepository hotelRepository;
        @Autowired
        private HotelBasicInfoRepository hotelBasicInfoRepository;
        @Autowired
```

```
	private HotelBasicInfoService hotelBasicInfoService;
	@ModelAttribute
	public Hotel constructHotel(){
		return new Hotel();
	}
	@ModelAttribute
	public HotelBasicInfo constructHotelBasicInfo(){
		return new HotelBasicInfo();
	}
	@ModelAttribute
	public CombinedHotelHotelBasicInfoCommand combinedHotelHotelBasicInfoCommand(){
		return new CombinedHotelHotelBasicInfoCommand();
	}
	@RequestMapping("/admin/hotels/{id}/control_panel")
	public String controlPanel(Model model, @PathVariable int id){
		Hotel hotel = hotelRepository.findOne(id);
		hotelService.checkAccess(hotel);
		model.addAttribute("hotel", hotel);
		return "hotelControlPanel";
	}
	@RequestMapping("/admin/hotels")
	public String hotels(Model model , Principal principal){
		String username = principal.getName();
		model.addAttribute("hotels", hotelService.findByUsername(username));
		return "hotelsListing";
	}
	@RequestMapping(value="/admin/hotels" , method=RequestMethod.POST)
	public String doAddHotels(@ModelAttribute("hotel") Hotel hotel, Principal principal){
		String username = principal.getName();
		hotelService.create(hotel , username);
		logger.debug("Create hotel-name ->"+hotel.getName());
		return "redirect:/admin/hotels.html";
	}
	@RequestMapping("/admin/hotels/{id}/basic_info")
	public String basicInfo(Model model, @PathVariable int id){
		model.addAttribute("hotel", hotelRepository.findOne(id));
		return "hotelBasicInfoRead";
	}
	@RequestMapping(value="/admin/hotels/{id}/basic_info" , method=RequestMethod.POST)
	public String doUpdateHotels(@ModelAttribute("hotel") Hotel hotel, Principal principal){
		logger.debug("Update hotel-name ->"+hotel.getId());
```

```
                        logger.debug("Update hotel-name ->"+hotel.getName());
                        String username = principal.getName();
                        hotelService.save(hotel , username);
                        return "redirect:/admin/hotels/{id}/basic_info.html";
        }
        @RequestMapping("/admin/hotels/{id}/basic_info/edit")
        public String editBasicInfo(Model model, @PathVariable int id){
                        int hotelId = id;
                        Hotel hotel = hotelRepository.findOne(hotelId);
                        HotelBasicInfo hotelBasicInfo = hotelBasicInfoRepository.findOne(hotelId);
                        model.addAttribute("hotel", hotel);
                        model.addAttribute("allCities",cityService.getAllCities());
                        model.addAttribute("allStars",starService.getAllStars());
                        CombinedHotelHotelBasicInfoCommand combinedHotelHotelBasicInfoCommand = new
CombinedHotelHotelBasicInfoCommand();
                        combinedHotelHotelBasicInfoCommand.setHotel(hotel);
                        combinedHotelHotelBasicInfoCommand.setHotelBasicInfo(hotelBasicInfo);
                        model.addAttribute("combinedHotelHotelBasicInfoCommand", combinedHotelHotelBasicInfoCommand);
                        return "hotelBasicInfoEdit";
        }
        @RequestMapping(value="/admin/hotels/{id}/basic_info/edit" , method=RequestMethod.POST)
        public String doUpdateBasicInfo(@ModelAttribute("combinedHotelHotelBasicInfoCommand")
CombinedHotelHotelBasicInfoCommand combinedHotelHotelBasicInfoCommand, Principal principal){
                        String username = principal.getName();
                        hotelService.save(combinedHotelHotelBasicInfoCommand.getHotel() , username);
                        hotelBasicInfoRepository.save(combinedHotelHotelBasicInfoCommand.getHotelBasicInfo());
                        logger.debug("Hotel updated successfully!!
(hotelId:"+combinedHotelHotelBasicInfoCommand.getHotel().getId()+")");
                        return "redirect:/admin/hotels/{id}/basic_info.html";
        }
        @RequestMapping(value="/admin/hotels/{hotelId}/remove")
        public String removeHotel(@PathVariable int hotelId){
                        hotelService.remove(hotelRepository.findOne(hotelId));
                        return "redirect:/admin/hotels.html";
        }
}
```

9.10.2.6 IndexController.java

```
package gr.greekchoices.admin.controller;
import gr.greekchoices.admin.service.UserService;
import java.security.Principal;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.RequestMapping;
@Controller
public class IndexController {
	@Autowired
	private UserService userService;
	@RequestMapping("/index")
	public String index(){
		return "index";
	}
	@RequestMapping(value="/admin/home")
	public String account(Model model , Principal principal){
		String username = principal.getName();
		model.addAttribute("user", userService.findOneWithRoles(username));
		return "home";
	}
}
```

9.10.2.7 LoginController.java

```
package gr.greekchoices.admin.controller;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
@Controller
public class LoginController {
	@RequestMapping("/login")
	public String login(){
		return "login";
	}
}
```

9.10.2.8 PeriodController.java

```
package gr.greekchoices.admin.controller;
import java.security.Principal;
import gr.greekchoices.admin.entity.Period;
import gr.greekchoices.admin.entity.RoomType;
import gr.greekchoices.admin.repository.HotelRepository;
import gr.greekchoices.admin.repository.PeriodRepository;
import gr.greekchoices.admin.service.PeriodService;
import org.apache.log4j.Logger;
import org.joda.time.LocalDate;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
@Controller
public class PeriodController {
	private static final Logger logger = Logger
				.getLogger(PeriodController.class);
	@Autowired
	HotelRepository hotelRepository;
	@Autowired
	PeriodService periodService;
	@Autowired
	PeriodRepository periodRepository;
	@ModelAttribute
	public Period constructPeriod(){
		Period period = new Period();
		return period;
	}
	@RequestMapping("/admin/hotels/{hotelId}/periods")
	public String periods(Model model, @PathVariable int hotelId){
		model.addAttribute("hotel", hotelRepository.findOne(hotelId));
		model.addAttribute("allPeriodPerHotel",periodService.allPeriodPerHotel(hotelId));
		return "periodsListing";
	}
	@RequestMapping(value="/admin/hotels/{hotelId}/periods",method=RequestMethod.POST)
	public String doAddPeriod(@ModelAttribute("period") Period period , @PathVariable int hotelId, Principal principal){
		//period.setStartdate(new LocalDate(2014, 8, 31));
```

```
                logger.debug("startDate ->"+period.getStartDate());
                String username = principal.getName();
                periodService.save(period , hotelId);
                return "redirect:/admin/hotels/{hotelId}/periods.html";
        }
        @RequestMapping(value="/admin/hotels/{hotelId}/period/{periodId}/edit")
        public String editPeriod(Model model, @PathVariable int periodId, @PathVariable int hotelId){
                model.addAttribute("hotel", hotelRepository.findOne(hotelId));
                model.addAttribute("period", periodRepository.findOne(periodId));
                return "periodEdit";
        }
        @RequestMapping(value="/admin/hotels/{hotelId}/period/{periodId}/edit",method=RequestMethod.POST)
        public String doUpdatePeriod(@ModelAttribute("period") Period period , @PathVariable int hotelId, Principal principal){
                logger.debug("Update period ->"+period.getId());
                logger.debug("Update period ->"+period.getStartDate()+"-"+period.getEndDate());
                String username = principal.getName();
                periodService.save(period , hotelId);
                return "redirect:/admin/hotels/{hotelId}/periods.html";
        }
        @RequestMapping(value="/admin/hotels/{hotelId}/period/{periodId}/remove")
        public String removePeriod(@PathVariable int periodId){
                Period period = periodService.findOne(periodId);
                periodService.remove(period);
                return "redirect:/admin/hotels/{hotelId}/periods.html";
        }
}
```

9.10.2.9 PriceController.java

```
package gr.greekchoices.admin.controller;
import java.security.Principal;
import javax.servlet.http.HttpServletRequest;
import gr.greekchoices.admin.entity.Facility;
import gr.greekchoices.admin.entity.Hotel;
import gr.greekchoices.admin.entity.Period;
import gr.greekchoices.admin.entity.Price;
import gr.greekchoices.admin.entity.PricePK;
import gr.greekchoices.admin.repository.HotelRepository;
import gr.greekchoices.admin.repository.PeriodRepository;
```

```
import gr.greekchoices.admin.repository.PriceRepository;
import gr.greekchoices.admin.repository.RoomTypeRepository;
import gr.greekchoices.admin.service.HotelService;
import gr.greekchoices.admin.service.PeriodService;
import gr.greekchoices.admin.service.PriceService;
import gr.greekchoices.admin.service.RoomTypeService;
import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.ui.ModelMap;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RequestParam;
@Controller
public class PriceController {
	private static final Logger logger = Logger
					.getLogger(PriceController.class);
	@Autowired
	private HotelService hotelService;
	@Autowired
	private HotelRepository hotelRepository;
	@Autowired
	private PriceRepository priceRepository;
	@Autowired
	private RoomTypeService roomTypeService;
	@Autowired
	private RoomTypeRepository roomTypeRepository;
	@Autowired
	private PeriodService periodService;
	@Autowired
	private PeriodRepository periodRepository;
	@Autowired
	private PriceService priceService;
	@ModelAttribute
	public Price constructPrice(){
		return new Price();
	}
	@Autowired
```

```
	private HttpServletRequest request;

	//Prices per Period All

	@RequestMapping("/admin/hotels/{hotelId}/prices/per-period/all")

	public String pricesPerPeriodAll(Model model,@PathVariable int hotelId){

		model.addAttribute("hotel", hotelRepository.findOne(hotelId));

		model.addAttribute("allPeriodPerHotel",periodService.allPeriodPerHotel(hotelId));

		model.addAttribute("allRoomTypePerHotel",roomTypeService.allRoomTypePerHotel(hotelId));

		model.addAttribute("allPricePerAllRoomTypePerAllPeriodPerHotel",
priceService.allPricePerAllRoomTypePerAllPeriodPerHotel(hotelId));

		return "pricePerAllRoomTypePerAllPeriodListing";

	}

	//Prices per Specific Period (ok)

	@RequestMapping("/admin/hotels/{hotelId}/prices/per-period/{periodId}")

	public String pricesPerPeriod(Model model,@PathVariable int hotelId,@PathVariable int periodId){

		model.addAttribute("hotel", hotelRepository.findOne(hotelId));

		model.addAttribute("allPeriodPerHotel",periodService.allPeriodPerHotel(hotelId));

		model.addAttribute("allRoomTypePerHotel",roomTypeService.allRoomTypePerHotel(hotelId));

		model.addAttribute("period", periodRepository.findOne(periodId));

		model.addAttribute("allPricePerAllRoomTypePerPeriodPerHotel",
priceService.allPricePerAllRoomTypePerPeriodPerHotel(periodId, hotelId));

		return "pricePerAllRoomTypePerPeriod";

	}

	//Prices per Room Type All

	@RequestMapping("/admin/hotels/{hotelId}/prices/per-roomtype/all")

	public String roomFacilitiesPerRoomTypeAll(Model model,@PathVariable int hotelId){

		model.addAttribute("hotel", hotelRepository.findOne(hotelId));

		model.addAttribute("allPeriodPerHotel",periodService.allPeriodPerHotel(hotelId));

		model.addAttribute("allRoomTypePerHotel",roomTypeService.allRoomTypePerHotel(hotelId));

		model.addAttribute("allPricePerAllPeriodPerAllRoomTypePerHotel",
priceService.allPricePerAllPeriodPerAllRoomTypePerHotel(hotelId));

		return "pricePerAllPeriodPerAllRoomTypeListing";

	}

	//Prices per Specific Room Type (ok)

	@RequestMapping("/admin/hotels/{hotelId}/prices/per-roomtype/{roomTypeId}")

	public String roomFacilitiesPerRoomType(Model model,@PathVariable int hotelId,@PathVariable int roomTypeId){

		model.addAttribute("hotel", hotelRepository.findOne(hotelId));

		model.addAttribute("allPeriodPerHotel",periodService.allPeriodPerHotel(hotelId));

		model.addAttribute("allRoomTypePerHotel",roomTypeService.allRoomTypePerHotel(hotelId));

		model.addAttribute("roomType", roomTypeRepository.findOne(roomTypeId));

		model.addAttribute("allPricePerAllPeriodPerRoomTypePerHotel",
priceService.allPricePerAllPeriodPerRoomTypePerHotel(roomTypeId, hotelId));

		return "pricePerAllPeriodPerRoomType";

	}
```

```
	// Edit Price
	@RequestMapping(value="/admin/hotels/{hotelId}/price/{periodId}/{roomTypeId}/edit")
	public String editPrice (Model model, @PathVariable int periodId, @PathVariable int roomTypeId, @PathVariable int hotelId){
		model.addAttribute("hotel", hotelRepository.findOne(hotelId));
		model.addAttribute("period", periodRepository.findOne(periodId));
		model.addAttribute("roomType", roomTypeRepository.findOne(roomTypeId));
		PricePK pricePK = new PricePK();
		pricePK.setHotelId(hotelId);
		pricePK.setPeriodId(periodId);
		pricePK.setRoomTypeId(roomTypeId);
		String referrer = request.getHeader("referer");
		model.addAttribute("referrer", referrer);
		logger.debug("Referrer:"+referrer);
		if(priceRepository.findByPricePK(pricePK)!=null){
			model.addAttribute("price", priceRepository.findByPricePK(pricePK));
		}
		return "priceEdit";
	}
	// Update Price
	@RequestMapping(value="/admin/hotels/{hotelId}/price/{periodId}/{roomTypeId}/edit",method=RequestMethod.POST)
	public String doUpdatePrice(@ModelAttribute("price") Price price,@PathVariable int hotelId, @PathVariable int periodId, @PathVariable int roomTypeId, Principal principal,@RequestParam("referrer") String referrer){
		logger.debug("Referrer34:"+referrer);
		// Briskoume to pricePK apo to url..ta hotelId,periodId,roomTypeId
		PricePK pricePK = new PricePK();
		pricePK.setHotelId(hotelId);
		pricePK.setPeriodId(periodId);
		pricePK.setRoomTypeId(roomTypeId);
		Price priceToUpdate;
		// An einai i proti fora pou dinoume timi..tote ftiaxe ena neo Price Object
		if(priceRepository.findByPricePK(pricePK)==null){
			priceToUpdate = new Price();
			priceToUpdate.setPricePK(pricePK);
		// Allios bres to sigkekrimeno price..
		}else{
			priceToUpdate = priceRepository.findByPricePK(pricePK);
		}
		// Tora pou exoume sta xeria mas to neo i to palio Price Object..these tou tin timi..kai kanto save/update..
		priceToUpdate.setCost(price.getCost());
		priceRepository.save(priceToUpdate);
		// Kai kane redirect..
```

```
                return "redirect:"+referrer;
        }
}
```

9.10.2.10 RoomTypeController.java

```
package gr.greekchoices.admin.controller;
import gr.greekchoices.admin.entity.Hotel;
import gr.greekchoices.admin.entity.Period;
import gr.greekchoices.admin.entity.RoomType;
import gr.greekchoices.admin.exception.MyCustomException;
import gr.greekchoices.admin.repository.HotelRepository;
import gr.greekchoices.admin.repository.RoomTypeRepository;
import gr.greekchoices.admin.service.RoomTypeService;
import java.security.Principal;
import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
@Controller
public class RoomTypeController {
        private static final Logger logger = Logger
                        .getLogger(RoomTypeController.class);
        @Autowired
        HotelRepository hotelRepository;
        @Autowired
        RoomTypeRepository roomTypeRepository;
        @Autowired
        RoomTypeService roomTypeService;
        @ModelAttribute
        public RoomType constructRoomType(){
                return new RoomType();
        }
        @RequestMapping("/admin/hotels/{hotelId}/room-types")
        public String roomTypes(Model model, @PathVariable int hotelId){
                /*model.addAttribute("roomtype", new RoomType());*/
                model.addAttribute("hotel", hotelRepository.findOne(hotelId));
```

```
                model.addAttribute("allRoomTypeByHotelId",roomTypeService.allRoomTypePerHotel(hotelId));
                model.addAttribute("allGlobalRoomTypeToAdd",roomTypeService.findAllGlobalRoomTypeToAdd(hotelId));
                return "roomTypesListing";
        }
        @RequestMapping(value="/admin/hotels/{hotelId}/room-types",method=RequestMethod.POST)
        public String doAddRoomType(@ModelAttribute("roomType") RoomType roomType , @PathVariable int hotelId, Principal
principal){
                String username = principal.getName();
                roomTypeService.save(roomType , username , hotelId);
                return "redirect:/admin/hotels/{hotelId}/room-types.html";
        }
        @RequestMapping(value="/admin/hotels/{hotelId}/room-type/{roomTypeId}/edit")
        public String editRoomType(Model model, @PathVariable int roomTypeId, @PathVariable int hotelId){
                model.addAttribute("hotel", hotelRepository.findOne(hotelId));
                model.addAttribute("roomType", roomTypeRepository.findOne(roomTypeId));
                return "roomTypeEdit";
        }
        @RequestMapping(value="/admin/hotels/{hotelId}/room-type/{roomTypeId}/edit",method=RequestMethod.POST)
        public String doUpdateRoomType(@ModelAttribute("roomType") RoomType roomType , @PathVariable int hotelId,
Principal principal){
                logger.debug("Update room type ->"+roomType.getId());
                logger.debug("Update room type ->"+roomType.getName());
                String username = principal.getName();
                roomTypeService.save(roomType , username , hotelId);
                return "redirect:/admin/hotels/{hotelId}/room-types.html";
        }
        @RequestMapping(value="/admin/hotels/{hotelId}/room-type/{roomTypeId}/remove")
        public String removeRoomType(@PathVariable int roomTypeId,@PathVariable int hotelId){
                Hotel hotel = hotelRepository.findOne(hotelId);
                RoomType roomType = roomTypeService.findOne(roomTypeId);
                roomTypeService.remove(roomType,hotel);
                return "redirect:/admin/hotels/{hotelId}/room-types.html";
        }
}
```

9.10.2.11 UserController.java

```
package gr.greekchoices.admin.controller;
import java.security.Principal;
import gr.greekchoices.admin.entity.Blog;
import gr.greekchoices.admin.entity.User;
```

```
import gr.greekchoices.admin.service.BlogService;
import gr.greekchoices.admin.service.UserService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
@Controller
public class UserController {
	@Autowired
	private UserService userService;
	@Autowired
	private BlogService blogService;
	@ModelAttribute("user")
	public User constructUser(){
		return new User();
	}
	@ModelAttribute("blog")
	public Blog constructBlog(){
		return new Blog();
	}
	@RequestMapping("/register")
	public String showRegister(){
		return "userRegister";
	}
	@RequestMapping(value="/register" , method=RequestMethod.POST)
	public String doRegister(@ModelAttribute("user") User user){
		userService.save(user);
		return "redirect:/register.html?success=true";
	}
	@RequestMapping("/admin/users")
	public String users(Model model){
		model.addAttribute("users", userService.findAll());
		return "users";
	}
	@RequestMapping("/admin/users/{id}")
	public String detail(Model model, @PathVariable int id){
		model.addAttribute("user", userService.findOneWithBlogs(id));
		return "userDetail";
```

```
	}
	@RequestMapping("/admin/account")
	public String account(Model model , Principal principal){
		String username = principal.getName();
		model.addAttribute("user", userService.findOneWithBlogs(username));
		return "account";
	}
	@RequestMapping(value="/admin/account" , method=RequestMethod.POST)
	public String doAddBlog(@ModelAttribute("blog") Blog blog, Principal principal){
		String username = principal.getName();
		blogService.save(blog, username);
		return "redirect:/admin/account.html";
	}
}
```

9.10.3 Repository

Στο πακέτο "gr.greekchoices.admin.repository" βρίσκονται τα αρχεία πηγαίου κώδικα, που παρουσιάζονται παρακάτω, και έχουν να κάνουν με το repository. Οι κλάσεις αυτές υλοποιούν την "διασύνδεση" (interface) JpaRepository του Spring Framework. Το interface αυτό είναι έτσι δομημένο ώστε να μπορεί βάση συγκεκριμένων παραδοχών στο όνομα της κλάσης που το υλοποιεί να μπορεί να αναδομεί τα κατάλληλα DAO's και κατ' επέκταση τα κατάλληλα ερωτήματα βάση του hibernate provider. Έτσι επιτυγχάνεται μέσω του ORM να έχουμε έτοιμες τις βασικές διαδικασίες που άπτωνται σε μια βάση δεδομένων με το ακρονύμιο CRUD (Creat, Read, Update, Delete), ενώ το Data Access Object (υλοποιείται μέσα στο framework) για τις πολύ απλές λειτουργίες όπως αυτή που μόλις αναφέραμε. Τέλος να σημειώσουμε ότι στο παρακάτω url μπορούεμε να δούμε όλες τις παραδοχές ονοματοδοσίας μιας repository class ώστε να μειωθεί στο ελλάχιστο η ανάγκη για εξειδικευμένα plsql ερωτήματα.

http://docs.spring.io/spring-data/jpa/docs/current/reference/html/

9.10.3.1 AvailabilityRepository.java

```
package gr.greekchoices.admin.repository;
import java.util.List;
import gr.greekchoices.admin.entity.Availability;
import gr.greekchoices.admin.entity.AvailabilityPK;
import gr.greekchoices.admin.entity.HotelHasRoomType;
import org.springframework.data.jpa.repository.JpaRepository;
public interface AvailabilityRepository extends JpaRepository<Availability, Integer> {
	List<Availability> findByHotelHasRoomTypeOrderByDateAsc(
					HotelHasRoomType hotelHasRoomType);
}
```

9.10.3.2 BookingRepository.java

```
package gr.greekchoices.admin.repository;
import java.util.List;
import gr.greekchoices.admin.entity.Booking;
import gr.greekchoices.admin.entity.HotelHasRoomType;
import org.springframework.data.jpa.repository.JpaRepository;
public interface BookingRepository extends JpaRepository<Booking, Integer> {
	List<Booking> findByHotelHasRoomType(HotelHasRoomType hotelHasRoomType);
}
```

9.10.3.3 FacilityRepository.java

```
package gr.greekchoices.admin.repository;
import java.util.List;
import gr.greekchoices.admin.entity.Blog;
import gr.greekchoices.admin.entity.Facility;
import gr.greekchoices.admin.entity.Hotel;
import gr.greekchoices.admin.entity.HotelHasRoomType;
import gr.greekchoices.admin.entity.Item;
import org.springframework.data.jpa.repository.JpaRepository;
public interface FacilityRepository extends JpaRepository<Facility, Integer> {
	//List<Facility> findByType(String type);
	List<Facility> findByType(String type);
	List<Facility> findByTypeAndHotelListOrderByNameAsc(String string,List<Hotel> hotelList);
	List<Facility> findByTypeAndGlobal(String string, boolean b);
	List<Facility> findByhotelHasRoomTypeListOrderByNameAsc(
				List<HotelHasRoomType> hotelHasRoomtypeList);
	List<Facility> findByHotelListOrderByNameAsc(List<Hotel> hotelList);
	List<Facility> findByGlobalAndHotelList(Boolean boolean1, List<Hotel> hotelList);
}
```

9.10.3.4 HotelBasicInfoRepository.java

```
package gr.greekchoices.admin.repository;
import gr.greekchoices.admin.entity.Hotel;
import gr.greekchoices.admin.entity.HotelBasicInfo;
import org.springframework.data.jpa.repository.JpaRepository;
public interface HotelBasicInfoRepository extends JpaRepository<HotelBasicInfo, Integer> {
	HotelBasicInfo findByHotel(Hotel hotel);
}
```

9.10.3.5 HotelHasRoomTypeRepository.java

```
package gr.greekchoices.admin.repository;
import java.util.List;
import gr.greekchoices.admin.entity.Hotel;
import gr.greekchoices.admin.entity.HotelHasRoomType;
import gr.greekchoices.admin.entity.HotelHasRoomTypePK;
import gr.greekchoices.admin.entity.RoomType;
import org.springframework.data.jpa.repository.JpaRepository;
public interface HotelHasRoomTypeRepository extends JpaRepository<HotelHasRoomType, Integer> {
	List<HotelHasRoomType> findByHotel(Hotel hotel);
	HotelHasRoomType findByHotelHasRoomTypePK(HotelHasRoomTypePK hotelHasRoomTypePK);
}
```

9.10.3.6 HotelRepository.java

```
package gr.greekchoices.admin.repository;
import java.util.List;
import gr.greekchoices.admin.entity.Hotel;
import gr.greekchoices.admin.entity.User;
import org.springframework.data.jpa.repository.JpaRepository;
public interface HotelRepository extends JpaRepository<Hotel, Integer> {
	List<Hotel> findByUser(User user);
}
```

9.10.3.7 PeriodRepository.java

```
package gr.greekchoices.admin.repository;
import java.util.List;
import gr.greekchoices.admin.entity.Hotel;
import gr.greekchoices.admin.entity.Period;
import gr.greekchoices.admin.entity.RoomType;
import org.springframework.data.jpa.repository.JpaRepository;
public interface PeriodRepository extends JpaRepository<Period, Integer> {
		List<Period> findByHotel(Hotel hotel);
		List<Period> findByHotelOrderByStartDateAsc(Hotel hotel);
}
```

9.10.3.8 PriceRepository.java

```
package gr.greekchoices.admin.repository;
import gr.greekchoices.admin.entity.Price;
import gr.greekchoices.admin.entity.PricePK;
import org.springframework.data.jpa.repository.JpaRepository;
public interface PriceRepository extends JpaRepository<Price, Integer> {
		Price findByPricePK(PricePK pricePK);
}
```

9.10.3.9 RoleRepository.java

```
package gr.greekchoices.admin.repository;
import gr.greekchoices.admin.entity.Role;
import org.springframework.data.jpa.repository.JpaRepository;
public interface RoleRepository extends JpaRepository<Role, Integer> {
		Role findByName(String bane);
}
```

9.10.3.10 RoomTypeRepository.java

```
package gr.greekchoices.admin.repository;
import java.util.List;
import gr.greekchoices.admin.entity.Hotel;
import gr.greekchoices.admin.entity.HotelHasRoomType;
import gr.greekchoices.admin.entity.RoomType;
import org.springframework.data.jpa.repository.JpaRepository;
public interface RoomTypeRepository extends JpaRepository<RoomType, Integer> {
	List<RoomType> findByGlobal(boolean b);
	List<RoomType> findByHotelHasRoomTypeListOrderByNameAsc(
				List<HotelHasRoomType> hotelHasRoomTypeList);
	List<RoomType> findByGlobalAndHotelHasRoomTypeList(Boolean boolean1,
				List<HotelHasRoomType> hotelHasRoomTypeList);
}
```

9.10.3.11 UserRepository.java

```
package gr.greekchoices.admin.repository;
import gr.greekchoices.admin.entity.User;
import org.springframework.data.jpa.repository.JpaRepository;
public interface UserRepository extends JpaRepository<User, Integer> {
	User findByUsername(String username);
}
```

9.10.4 Service

Στο πακέτο "gr.greekchoices.admin.service" βρίσκονται τα αρχεία πηγαίου κώδικα, που παρουσιάζονται παρακάτω, και έχουν να κάνουν με το service. Οι κλάσεις αυτές υλοποιούν την "διασύνδεση" (interface) JpaRepository του Spring Framework.

Βάση του JPA και του JPA Annotation (@Service) αυτά τα αντικείμενα είναι οι "υπηρεσίες" οι μέθοδοι που τρέχουν και κάνουν τις διάφορες εννιολογικές διαδικασίες. Σε αυτές τις κλάσεις βρίσκεται όλο το business Logic της εφαρμογής. Πρόκειται προφανώς για πιο πολύπλοκες διαδικασίες από αυτές που είδαμε στην προηγούμενη ενότητα που έχουν να κάνουν με τα repositories και κατ' επέκταση τα DAO's αλλά είναι πολύ σύνηθες να καλούνται μέθοδοι των Repository κλάσεων μέσα σε κλάσεις Services. Τέλος να σημειώσουμε ότι στο σώμα μιας μεθόδου ενός ελεγκτή γίνεται επίσης μεγάλη χρήση μεθόδων των Service κλάσεων πριν προωθηθεί ο έλεγχος στο View.

9.10.4.1 **AvailabilityService.java**

```
package gr.greekchoices.admin.service;
import java.math.BigDecimal;
import java.util.HashMap;
import java.util.List;
import java.util.Locale;
import java.util.Map;
import gr.greekchoices.Constants;
import gr.greekchoices.admin.entity.Availability;
import gr.greekchoices.admin.entity.AvailabilityPK;
import gr.greekchoices.admin.entity.Hotel;
import gr.greekchoices.admin.entity.HotelHasRoomType;
import gr.greekchoices.admin.entity.HotelHasRoomTypePK;
import gr.greekchoices.admin.entity.Period;
import gr.greekchoices.admin.entity.Price;
import gr.greekchoices.admin.entity.PricePK;
import gr.greekchoices.admin.model.AvailabilityCalendarData;
import gr.greekchoices.admin.repository.AvailabilityRepository;
import org.apache.log4j.Logger;
import org.joda.time.DateTimeZone;
import org.joda.time.LocalDate;
```

```
import org.joda.time.format.DateTimeFormat;
import org.joda.time.format.DateTimeFormatter;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
@Service
public class AvailabilityService  implements Constants{
	private static final Logger logger = Logger
				.getLogger(AvailabilityService.class);
	@Autowired
	private AvailabilityRepository availabilityRepository;
	public void save(Availability availability, int roomTypeId, int hotelId, LocalDate date) {
		AvailabilityPK availabilityPK = new AvailabilityPK();
		availabilityPK.setHotelId(hotelId);
		availabilityPK.setRoomTypeId(roomTypeId);
		availabilityPK.setDate(date);
		availability.setAvailabilityPK(availabilityPK);
		availabilityRepository.save(availability);
		logger.debug("Availability Saved [quantity:\""+availability.getQuantity()+"\" , roomtypeId:\""+roomTypeId+"\" ,
hotelId:\""+hotelId+"\"]");
		logger.debug("");
	}
	public Map<LocalDate, AvailabilityCalendarData> availabilitiesCalendarMapToJson(int roomTypeId, int hotelId){
		HashMap<LocalDate, AvailabilityCalendarData> availabilitiesCalendarMapToJson = new HashMap<LocalDate,
AvailabilityCalendarData>();
		HotelHasRoomTypePK hotelHasRoomTypePK = new HotelHasRoomTypePK();
		hotelHasRoomTypePK.setHotelId(hotelId);
		hotelHasRoomTypePK.setRoomTypeId(roomTypeId);
		HotelHasRoomType hotelHasRoomType = new HotelHasRoomType();
		hotelHasRoomType.setHotelHasRoomTypePK(hotelHasRoomTypePK);
		List<Availability> availabilityList =
availabilityRepository.findByHotelHasRoomTypeOrderByDateAsc(hotelHasRoomType);
		for(Availability availability: availabilityList){
			logger.debug(availability.getDate()+" -> "+Integer.toString(availability.getQuantity()));
			availabilitiesCalendarMapToJson.put(availability.getDate(), new
AvailabilityCalendarData(Integer.toString(availability.getQuantity()), 0, "", "", "", "", "available"));
		}
		return availabilitiesCalendarMapToJson;
	}
	public void saveAll(Map<String, AvailabilityCalendarData> availabilityCalendarDataHashMap, int roomTypeId ,int hotelId) {
		DateTimeFormatter availabilityFormater = DateTimeFormat.forPattern(PATTERN_DAY);
		DateTimeFormatter jsonAjaxAvailabilityFormater = DateTimeFormat.forPattern("yyyy-MM-dd");
		Map<String, AvailabilityCalendarData> availabilitiesCalendarMapToJsonToDelete = new HashMap<String,
AvailabilityCalendarData>();
```

```
                Map<LocalDate, AvailabilityCalendarData> availabilitiesCalendarMapToJson =
availabilitiesCalendarMapToJson(roomTypeId,hotelId);

                for (Map.Entry<LocalDate, AvailabilityCalendarData> entry : availabilitiesCalendarMapToJson.entrySet()) {

                        LocalDate jsonAjaxAvailabilityDateTemp =
jsonAjaxAvailabilityFormater.parseLocalDate(entry.getKey().toString());

                LocalDate availabilityDateTemp =
availabilityFormater.parseLocalDate(jsonAjaxAvailabilityDateTemp.getDayOfMonth()+"/"+jsonAjaxAvailabilityDateTemp.getMonthOfY
ear()+"/"+jsonAjaxAvailabilityDateTemp.getYear());

                        availabilitiesCalendarMapToJsonToDelete.put(availabilityDateTemp.toString(), new
AvailabilityCalendarData(entry.getValue().getAvailable(), 0, "", "", "", "", "available"));

        }

                availabilitiesCalendarMapToJsonToDelete.keySet().removeAll(availabilityCalendarDataHashMap.keySet());

                for (Map.Entry<String, AvailabilityCalendarData> entry : availabilitiesCalendarMapToJsonToDelete.entrySet()) {

                        logger.debug("--------Delete----------");

                        logger.debug("date:"+entry.getKey()+" | availability:"+entry.getValue().getAvailable());

                        AvailabilityPK availabilityPK = new AvailabilityPK();

                        availabilityPK.setHotelId(hotelId);

                        availabilityPK.setRoomTypeId(roomTypeId);

                        LocalDate jsonAjaxAvailabilityDateTemp =
jsonAjaxAvailabilityFormater.parseLocalDate(entry.getKey().toString());

                LocalDate availabilityDateTemp =
availabilityFormater.parseLocalDate(jsonAjaxAvailabilityDateTemp.getDayOfMonth()+"/"+jsonAjaxAvailabilityDateTemp.getMonthOfY
ear()+"/"+jsonAjaxAvailabilityDateTemp.getYear());

                        availabilityPK.setDate(availabilityDateTemp);

                        Availability availability = new Availability();

                        availability.setAvailabilityPK(availabilityPK);

                        availabilityRepository.delete(availability);

                        logger.debug("----------------------------");

        }

        for (Map.Entry<String, AvailabilityCalendarData> entry : availabilityCalendarDataHashMap.entrySet()) {

                logger.debug("date:"+entry.getKey()+" | availability:"+entry.getValue().getAvailable());

                LocalDate jsonAjaxAvailabilityDate = jsonAjaxAvailabilityFormater.parseLocalDate(entry.getKey());

                        logger.debug(">1> "+jsonAjaxAvailabilityDate+" <1<");

                LocalDate availabilityDate =
availabilityFormater.parseLocalDate(jsonAjaxAvailabilityDate.getDayOfMonth()+"/"+jsonAjaxAvailabilityDate.getMonthOfYear()+"/"+js
onAjaxAvailabilityDate.getYear());

                        logger.debug(">2> "+availabilityDate+" <2<");

                        Availability availability = new Availability();

                        availability.setQuantity(Integer.parseInt(entry.getValue().getAvailable()));

                        save(availability, roomTypeId, hotelId, availabilityDate);

        }

       }
}
```

9.10.4.2 BookingService.java

```
package gr.greekchoices.admin.service;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import javax.transaction.Transactional;
import gr.greekchoices.admin.entity.Booking;
import gr.greekchoices.admin.entity.Hotel;
import gr.greekchoices.admin.entity.HotelHasRoomType;
import gr.greekchoices.admin.entity.Period;
import gr.greekchoices.admin.entity.RoomType;
import gr.greekchoices.admin.entity.User;
import gr.greekchoices.admin.repository.BookingRepository;
import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
@Service
public class BookingService {
	private static final Logger logger = Logger
				.getLogger(BookingService.class);
	@Autowired
	private BookingRepository bookingRepository;
	@Autowired
	private RoomTypeService roomTypeService;
	@Autowired
	private HotelHasRoomTypeService hotelHasRoomTypeService;
	public List<Booking> findAllBookingPerAllRoomTypeByHotelId(int hotelId) {
		List<Booking> findAllBookingPerAllRoomTypeByHotelId = new ArrayList<Booking>();
		List<HotelHasRoomType> allHotelHasRoomTypePerAllRoomTypePerHotel =
hotelHasRoomTypeService.allHotelHasRoomTypePerAllRoomTypePerHotel(hotelId);
		for(HotelHasRoomType hotelHasRoomType : allHotelHasRoomTypePerAllRoomTypePerHotel){
			for(Booking booking : bookingRepository.findByHotelHasRoomType(hotelHasRoomType)){
				findAllBookingPerAllRoomTypeByHotelId.add(booking);
			}
		}
		logger.debug("");
		logger.debug("Bookings");
		logger.debug("--------");
		for(Booking booking : findAllBookingPerAllRoomTypeByHotelId){
			logger.debug("BookingId: "+booking.getId()+" , BookingCost: "+booking.getTotalCost()+" ,
BookingUser:"+booking.getUser().getUsername());
```

```
                }
                logger.debug("");
                return findAllBookingPerAllRoomTypeByHotelId;
        }
}
```

9.10.4.3 CityService.java

```
package gr.greekchoices.admin.service;
import java.util.HashMap;
import java.util.List;
import javax.transaction.Transactional;
import gr.greekchoices.admin.entity.City;
import gr.greekchoices.admin.entity.Hotel;
import gr.greekchoices.admin.entity.HotelHasRoomType;
import gr.greekchoices.admin.entity.RoomType;
import gr.greekchoices.admin.repository.CityRepository;
import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
@Service
public class CityService {
        private static final Logger logger = Logger
                        .getLogger(CityService.class);
        @Autowired
        private CityRepository cityRepository;
        @Transactional
        public HashMap<Long, City> getAllCities() {
                HashMap<Long,City> getAllCities = new HashMap<Long, City>();

                List<City> cityList = cityRepository.findAll();
                logger.debug(".::findAllCities::.");
                for (City city : cityList) {
                        getAllCities.put(new Long(city.getId()) , city);
                        logger.debug("cityId: "+city.getId()+" | cityName: "+city.getName());
                }
                return getAllCities;
```

```
		}
}
```

9.10.4.4 FacilityService.java

```
package gr.greekchoices.admin.service;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Iterator;
import java.util.List;
import javax.transaction.Transactional;
import gr.greekchoices.admin.controller.IndexController;
import gr.greekchoices.admin.entity.Blog;
import gr.greekchoices.admin.entity.Facility;
import gr.greekchoices.admin.entity.Hotel;
import gr.greekchoices.admin.entity.HotelBasicInfo;
import gr.greekchoices.admin.entity.HotelHasRoomType;
import gr.greekchoices.admin.entity.HotelHasRoomTypePK;
import gr.greekchoices.admin.entity.Item;
import gr.greekchoices.admin.entity.Role;
import gr.greekchoices.admin.entity.RoomType;
import gr.greekchoices.admin.entity.User;
import gr.greekchoices.admin.repository.FacilityRepository;
import gr.greekchoices.admin.repository.HotelHasRoomTypeRepository;
import gr.greekchoices.admin.repository.HotelRepository;
import gr.greekchoices.admin.repository.RoomTypeRepository;
import gr.greekchoices.admin.repository.UserRepository;
import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.PageRequest;
import org.springframework.data.domain.Sort.Direction;
import org.springframework.stereotype.Service;
@Service
public class FacilityService {
		private static final Logger logger = Logger
					.getLogger(FacilityService.class);
		@Autowired
```

```
	private FacilityRepository facilityRepository;
	@Autowired
	private UserRepository userRepository;
	@Autowired
	private HotelRepository hotelRepository;
	@Autowired
	private RoomTypeService roomTypeService;
	@Autowired
	private HotelHasRoomTypeService hotelHasRoomTypeService;
	@Autowired
	private HotelHasRoomTypeRepository hotelHasRoomTypeRepository;
	public List<Facility> findAll(){
		return facilityRepository.findAll();
	}
	public List<Facility> findAllHotelFacility(){
		return facilityRepository.findByType("hotel");
	}
	public List<Facility> findAllRoomTypeFacility(){
		return facilityRepository.findByType("roomtype");
	}
	@Transactional
	public List<Facility> findAllHotelFacilityByTypeAndGlobal(){
		return facilityRepository.findByTypeAndGlobal("hotel",true);
	}
	@Transactional
	public List<Facility> findAllRoomFacilityByTypeAndGlobal(){
		return facilityRepository.findByTypeAndGlobal("roomtype",true);
	}
	@Transactional
	public List<Facility> findAllHotelFacilityByHotelId(int hotelId){
		Hotel hotel = hotelRepository.findOne(hotelId);
		List<Hotel> hotelList = new ArrayList<Hotel>();
		hotelList.add(hotel);
		return facilityRepository.findByTypeAndHotelListOrderByNameAsc("hotel", hotelList);
	}
	@Transactional
	public List<Facility> findAllRoomFacilityByHotelId(int hotelId) {
		Hotel hotel = hotelRepository.findOne(hotelId);
		List<Hotel> hotelList = new ArrayList<Hotel>();
		hotelList.add(hotel);
		return facilityRepository.findByTypeAndHotelListOrderByNameAsc("roomtype", hotelList);
```

```
	}
	@Transactional
	public List<Facility> findAllGlobalHotelFacilityToAdd(int hotelId) {
		List<Facility> hotelFacilityList = findAllHotelFacilityByTypeAndGlobal();
		List<Facility> hotelFacilityByHotelIdList = findAllHotelFacilityByHotelId(hotelId);
		List<Facility> globalHotelFacilityToAddList = hotelFacilityList;
		globalHotelFacilityToAddList.removeAll(hotelFacilityByHotelIdList);
		return globalHotelFacilityToAddList;
	}
	@Transactional
	public List<Facility> findAllGlobalRoomFacilityToAdd(int hotelId) {
		List<Facility> roomFacilityList = findAllRoomFacilityByTypeAndGlobal();
		List<Facility> roomFacilityByHotelIdList = findAllRoomFacilityByHotelId(hotelId);
		List<Facility> globalRoomFacilityToAddList = roomFacilityList;
		globalRoomFacilityToAddList.removeAll(roomFacilityByHotelIdList);
		return globalRoomFacilityToAddList;
	}
	@Transactional
	public void save(Facility facility, String username, int hotelId) {
		// An to global einai true...mas irthe meso ADD apo ta pre-configured facilities.
		if(facility.getGlobal()){
			Facility facilityToUpdateInDb = facilityRepository.findOne(facility.getId());
			Hotel hotel = hotelRepository.findOne(hotelId);
			List<Hotel> hotelList = new ArrayList<Hotel>();
			hotelList.add(hotel);
			facilityToUpdateInDb.setHotelList(hotelList);
			facilityRepository.save(facilityToUpdateInDb);
			List<Facility> facilityList = hotel.getFacilityList();
			facilityList.add(facilityToUpdateInDb);
			hotel.setFacilityList(facilityList);
			hotelRepository.save(hotel);
			logger.debug("Added \""+facilityToUpdateInDb.getName()+"\"");
		// An to global einai false...mas irthe meso NEW
		}else{
			facilityRepository.save(facility);
			Hotel hotel = hotelRepository.findOne(hotelId);
			List<Facility> facilityList = hotel.getFacilityList();
			// An den periexei to facility..einai add allios einai update..kai den to xanabazoume..
			if(!facilityList.contains(facility)){
				facilityList.add(facility);
			}
```

```
                                hotel.setFacilityList(facilityList);
                                hotelRepository.save(hotel);
                                logger.debug("Created  \""+facility.getName()+"\"");
                        }
                }
                public List<HotelHasRoomType> allHotelHasRoomTypePerRoomTypePerHotel(int hotelId , int roomTypeId) {
                        HotelHasRoomTypePK hotelHasRoomTypePK = new HotelHasRoomTypePK();
                        hotelHasRoomTypePK.setHotelId(hotelId);
                        hotelHasRoomTypePK.setRoomTypeId(roomTypeId);
                        HotelHasRoomType hotelHasRoomType = new HotelHasRoomType();
                        hotelHasRoomType.setHotelHasRoomTypePK(hotelHasRoomTypePK);
                        List<HotelHasRoomType> allHotelHasRoomTypePerRoomTypePerHotel = new ArrayList<HotelHasRoomType>();
                        allHotelHasRoomTypePerRoomTypePerHotel.add(hotelHasRoomType);
                        return allHotelHasRoomTypePerRoomTypePerHotel;
                }
                public List<Facility> allRoomFacilityPerRoomTypePerHotel(int hotelId , int roomTypeId) {
                        HotelHasRoomType hotelHasRoomTypePerRoomTypePerHotel =
hotelHasRoomTypeService.hotelHasRoomTypePerRoomTypePerHotel(roomTypeId, hotelId);
                        List<HotelHasRoomType> hotelHasRoomtypeList = new ArrayList<HotelHasRoomType>();
                        hotelHasRoomtypeList.add(hotelHasRoomTypePerRoomTypePerHotel);
                        return facilityRepository.findByhotelHasRoomTypeListOrderByNameAsc(hotelHasRoomtypeList);
                }
                public List<Facility> allAvailableRoomFacilityPerRoomTypePerHotel(int hotelId  , int roomTypeId) {
                        List<Facility> allRoomFacilityPerHotel = findAllRoomFacilityByHotelId(hotelId);
                        List<Facility> allRoomFacilityPerRoomTypePerHotel =
allRoomFacilityPerRoomTypePerHotel(hotelId,roomTypeId);
                        List<Facility> allAvailableRoomFacilityPerRoomTypePerHotel = allRoomFacilityPerHotel;
                        allAvailableRoomFacilityPerRoomTypePerHotel.removeAll(allRoomFacilityPerRoomTypePerHotel);
                        return allAvailableRoomFacilityPerRoomTypePerHotel;
                }
          // Find All Available Room Facilities per Room Type (HashMap)
                public HashMap<Long, List<Facility>> allAvaliableRoomFacilityPerAllRoomTypePerHotel(int hotelId) {
                        HashMap<Long, List<Facility>> allAvaliableRoomFacilityPerAllRoomTypePerHotel = new HashMap<Long,
List<Facility>>();
                        for (RoomType roomType : roomTypeService.findAllRoomTypeByHotelId(hotelId)) {
                                logger.debug("#0#"+roomType.getId()+"@0@");
                                List<Facility> facilityList = allAvailableRoomFacilityPerRoomTypePerHotel(hotelId,roomType.getId());
                                for ( Facility item : facilityList) {
                                        //@roomType : 1 | Room type 1@
                                        logger.debug("#1#"+item.getName()+"@1@");
                                }
                                allAvaliableRoomFacilityPerAllRoomTypePerHotel.put(new Long(roomType.getId()), facilityList);
                        }
```

```
		Iterator<Long> keySetIterator = allAvaliableRoomFacilityPerAllRoomTypePerHotel.keySet().iterator();
		while(keySetIterator.hasNext()){
		 Long key = keySetIterator.next();
		   logger.debug("#99# key: " + key + " value: " +"#99#");
		 for ( Facility item : allAvaliableRoomFacilityPerAllRoomTypePerHotel.get(key)) {
				//@roomType : 1 | Room type 1@
				logger.debug("##"+item.getName()+"@@");
		 }
		}
		return allAvaliableRoomFacilityPerAllRoomTypePerHotel;
	}
	public HashMap<Long, List<Facility>> allRoomFacilityPerAllRoomTypePerHotel(int hotelId) {
		HashMap<Long, List<Facility>> allRoomFacilityPerAllRoomTypePerHotel = new HashMap<Long, List<Facility>>();
		for (RoomType roomType : roomTypeService.findAllRoomTypeByHotelId(hotelId)) {
				List<Facility> facilityList = allRoomFacilityPerRoomTypePerHotel(hotelId,roomType.getId());
				allRoomFacilityPerAllRoomTypePerHotel.put(new Long(roomType.getId()), facilityList);
		}
		return allRoomFacilityPerAllRoomTypePerHotel;
	}
	public void remove(Facility facility, Hotel hotel) {
		if(facility.getGlobal()){
				List<Hotel> hotelList = new ArrayList<Hotel>();
				hotelList.add(hotel);
				List<Facility> facilityList = facilityRepository.findByHotelListOrderByNameAsc(hotelList);
				logger.debug("");
				logger.debug("");
				logger.debug("-------------------------");
				logger.debug("Facility list Before (Hotel Name :"+hotel.getName());
				for(Facility tempfacility : facilityList){
					logger.debug(tempfacility.getName());
				}
				logger.debug("");
				logger.debug("-------------------------");
				logger.debug("Facility list After (Hotel Name :"+hotel.getName());
				facilityList.remove(facility);
				for(Facility tempfacility : facilityList){
					logger.debug(tempfacility.getName());
				}
				hotel.setFacilityList(facilityList);
				hotelRepository.save(hotel);
				logger.debug("Facility ["+facility.getName()+"] removed!!");
```

```
                }else{
                        /*Hotel hotel = hotelRepository.findOne(hotelId);*/
                        List<Hotel> hotelList = new ArrayList<Hotel>();
                        hotelList.add(hotel);
                        List<Facility> facilityList = facilityRepository.findByHotelListOrderByNameAsc(hotelList);
                        logger.debug("");
                        logger.debug("");
                        logger.debug("-------------------------");
                        logger.debug("Facility list Before (Hotel Name :"+hotel.getName());
                        for(Facility tempfacility : facilityList){
                                logger.debug(tempfacility.getName());
                        }
                        logger.debug("");
                        logger.debug("-------------------------");
                        logger.debug("Facility list After (Hotel Name :"+hotel.getName());
                        facilityList.remove(facility);
                        for(Facility tempfacility : facilityList){
                                logger.debug(tempfacility.getName());
                        }
                        hotel.setFacilityList(facilityList);
                        hotelRepository.save(hotel);
                        facilityRepository.delete(facility);
                        logger.debug("Facility ["+facility.getName()+"] deleted!!");
                }
        }
        public void updateRoomFacilities(int[] roomTypeFacilitiesArray,int roomTypeId, int hotelId) {
                List<Facility> facilityList = new ArrayList<Facility>();
                if(roomTypeFacilitiesArray != null){
                        for (int i = 0; i < roomTypeFacilitiesArray.length; i++){
                                Facility facility = facilityRepository.findOne(roomTypeFacilitiesArray[i]);
                                logger.debug("Added Facility: "+facility.getName()+"(id:"+roomTypeFacilitiesArray[i]+")");
                                facilityList.add(facility);
                   }
                }
                HotelHasRoomTypePK hotelHasRoomTypePK = new HotelHasRoomTypePK();
                hotelHasRoomTypePK.setHotelId(hotelId);
                hotelHasRoomTypePK.setRoomTypeId(roomTypeId);
                HotelHasRoomType hotelHasRoomType =
hotelHasRoomTypeRepository.findByHotelHasRoomTypePK(hotelHasRoomTypePK);
                hotelHasRoomType.setFacilityList(facilityList);
                hotelHasRoomTypeRepository.save(hotelHasRoomType);
        }
```

```
	public List<Facility> getCustomFacilities(Hotel hotel) {
		List<Hotel> hotelList = new ArrayList<Hotel>();
		hotelList.add(hotel);
		List<Facility> facilityList = facilityRepository.findByGlobalAndHotelList(new Boolean(false), hotelList);
		return facilityList;
	}
	public void removeCustomFacilities(List<Facility> facilityList) {
		for(Facility tempfacility : facilityList){
			logger.debug("Deleting facility "+tempfacility.getName());
			facilityRepository.delete(tempfacility);
			logger.debug("Facility with name '"+tempfacility.getName()+"' deleted succesfully!!");
		}
	}
}
```

9.10.4.5 HotelBasicInfoService.java

```
package gr.greekchoices.admin.service;
import gr.greekchoices.admin.entity.City;
import gr.greekchoices.admin.entity.Hotel;
import gr.greekchoices.admin.entity.HotelBasicInfo;
import gr.greekchoices.admin.entity.Star;
import gr.greekchoices.admin.repository.HotelBasicInfoRepository;
import gr.greekchoices.admin.repository.HotelRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
@Service
public class HotelBasicInfoService {
	@Autowired
	private HotelBasicInfoRepository hotelBasicInfoRepository;
	@Autowired
	private HotelRepository hotelRepository;
}
```

9.10.4.6HotelHasRoomTypeService.java

```
package gr.greekchoices.admin.service;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import gr.greekchoices.admin.entity.HotelHasRoomType;
import gr.greekchoices.admin.entity.HotelHasRoomTypePK;
import gr.greekchoices.admin.entity.RoomType;
import gr.greekchoices.admin.repository.HotelHasRoomTypeRepository;
import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
@Service
public class HotelHasRoomTypeService {
	private static final Logger logger = Logger
				.getLogger(HotelHasRoomTypeService.class);
	@Autowired
	private HotelHasRoomTypeRepository hotelHasRoomTypeRepository;
	@Autowired
	private RoomTypeService roomTypeService;
	public HotelHasRoomType hotelHasRoomTypePerRoomTypePerHotel(int roomTypeId , int hotelId) {
		HotelHasRoomTypePK hotelHasRoomTypePK = new HotelHasRoomTypePK();
		hotelHasRoomTypePK.setHotelId(hotelId);
		hotelHasRoomTypePK.setRoomTypeId(roomTypeId);
		HotelHasRoomType hotelHasRoomType = new HotelHasRoomType();
		hotelHasRoomType.setHotelHasRoomTypePK(hotelHasRoomTypePK);
		return hotelHasRoomType;
	}
	public List<HotelHasRoomType> allHotelHasRoomTypePerAllRoomTypePerHotel(int hotelId) {
		List<HotelHasRoomType> allHotelHasRoomTypePerAllRoomTypePerHotel = new
ArrayList<HotelHasRoomType>();
		HashMap<Long , RoomType> allRoomTypePerHotel = roomTypeService.allRoomTypePerHotel(hotelId);
		for(Map.Entry<Long, RoomType> entryOfRoomType : allRoomTypePerHotel.entrySet() ) {
			HotelHasRoomType hotelHasRoomTypePerRoomTypePerHotel =
hotelHasRoomTypePerRoomTypePerHotel(entryOfRoomType.getValue().getId() , hotelId);
			allHotelHasRoomTypePerAllRoomTypePerHotel.add(hotelHasRoomTypePerRoomTypePerHotel);
			logger.debug(hotelHasRoomTypePerRoomTypePerHotel);
		}
		return allHotelHasRoomTypePerAllRoomTypePerHotel;
	}
}
```

9.10.4.7 HotelService.java

```
package gr.greekchoices.admin.service;

import java.util.List;

import javax.transaction.Transactional;

import gr.greekchoices.admin.entity.City;
import gr.greekchoices.admin.entity.Facility;
import gr.greekchoices.admin.entity.Hotel;
import gr.greekchoices.admin.entity.HotelBasicInfo;
import gr.greekchoices.admin.entity.HotelHasRoomType;
import gr.greekchoices.admin.entity.RoomType;
import gr.greekchoices.admin.entity.User;
import gr.greekchoices.admin.repository.FacilityRepository;
import gr.greekchoices.admin.repository.HotelBasicInfoRepository;
import gr.greekchoices.admin.repository.HotelHasRoomTypeRepository;
import gr.greekchoices.admin.repository.HotelRepository;
import gr.greekchoices.admin.repository.UserRepository;
import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.PageRequest;
import org.springframework.data.domain.Sort.Direction;
import org.springframework.security.access.method.P;
import org.springframework.security.access.prepost.PreAuthorize;
import org.springframework.stereotype.Service;
@Service
public class HotelService {
	private static final Logger logger = Logger
					.getLogger(HotelService.class);
	@Autowired
	private HotelRepository hotelRepository;
	@Autowired
	private UserRepository userRepository;
	@Autowired
	private HotelBasicInfoRepository hotelBasicInfoRepository;
	@Autowired
	private FacilityService facilityService;
	@Autowired
	private RoomTypeService roomTypeService;
	@Autowired
```

```
	private HotelHasRoomTypeRepository hotelHasRoomTypeRepository;
	public List<Hotel> findAll(){
	 return hotelRepository.findAll();
	}
	public List<Hotel> findByUsername(String username) {
			User user = userRepository.findByUsername(username);
			return hotelRepository.findByUser(user);
	}
	public void create(Hotel hotel, String username) {
			User user = userRepository.findByUsername(username);
			hotel.setUser(user);
			HotelBasicInfo hotelBasicInfoFirst = new HotelBasicInfo();
			hotelBasicInfoFirst.setHotel(hotel);
			hotelBasicInfoRepository.save(hotelBasicInfoFirst);
	}
	public void save(Hotel hotel, String username) {
			User user = userRepository.findByUsername(username);
			hotel.setUser(user);
			hotelRepository.save(hotel);
	}
	@PreAuthorize("#hotel.user.username == authentication.name or hasRole('ROLE_ADMIN')")
	public void checkAccess(@P("hotel") Hotel hotel) {
			// TODO Auto-generated method stub
	}
	@PreAuthorize("#hotel.user.username == authentication.name or hasRole('ROLE_ADMIN')")
	public void remove(@P("hotel") Hotel hotel) {
			logger.debug("Holding custom facilities for hotel with id "+hotel.getId()+"...");
			List<Facility> customFacilityList = facilityService.getCustomFacilities(hotel);
			logger.debug("Holding custom room types for hotel with id "+hotel.getId()+"...");
			List<RoomType> customRoomTypeList = roomTypeService.getCustomRoomTypesPerHotel(hotel);
			logger.debug("Deleting hotel with id "+hotel.getId()+"...");
			hotelRepository.delete(hotel);
			logger.debug("Hotel with id "+hotel.getId()+" deleted successfully!!");
			logger.debug("Deleting hotel custom facilities from hotel with id "+hotel.getId()+"...");
			facilityService.removeCustomFacilities(customFacilityList);
			logger.debug("Custom Facilities deleted successfully");
			logger.debug("Deleting hotel custom room types from hotel with id "+hotel.getId()+"...");
			roomTypeService.removeCustomRoomTypes(customRoomTypeList);
			logger.debug("Custom Room Types deleted successfully");
	}
}
```

9.10.4.8 InitDbService.java

Η κλάση InitDbService κάνει χρήση της μεθόδου init() και του annotation @PostConstruct που σύμφωνα με τον κύκλο ζωής (Life Cycle) ενός Servlet εκτελείται πριν από όλα. Με αυτό τον τρόπο μπορούμε να γεμίζουμε με δεδομένα (αρχικοποίηση) την βάση δεδομένων μας (μετά πρέπει να την βάλουμε σε σχόλιο..) ή χρησιμοποιείται κατά την υλοποίηση (development) ώστε να έχουμε συνεχώς τα ίδια αρχικοποιημένα δεδομένα σε κάθε επανεκίνηση του Application Server (π.χ. του tomcat, jetty κτλ)

```
package gr.greekchoices.admin.service;
import gr.greekchoices.Constants;
import gr.greekchoices.admin.entity.Availability;
import gr.greekchoices.admin.entity.Blog;
import gr.greekchoices.admin.entity.Booking;
import gr.greekchoices.admin.entity.BookingStatus;
import gr.greekchoices.admin.entity.City;
import gr.greekchoices.admin.entity.Facility;
import gr.greekchoices.admin.entity.Hotel;
import gr.greekchoices.admin.entity.HotelBasicInfo;
import gr.greekchoices.admin.entity.HotelHasRoomType;
import gr.greekchoices.admin.entity.HotelHasRoomTypePK;
import gr.greekchoices.admin.entity.Item;
import gr.greekchoices.admin.entity.Log;
import gr.greekchoices.admin.entity.Period;
import gr.greekchoices.admin.entity.Price;
import gr.greekchoices.admin.entity.PricePK;
import gr.greekchoices.admin.entity.Role;
import gr.greekchoices.admin.entity.RoomType;
import gr.greekchoices.admin.entity.Star;
import gr.greekchoices.admin.entity.User;
import gr.greekchoices.admin.model.Event;
import gr.greekchoices.admin.repository.BlogRepository;
import gr.greekchoices.admin.repository.BookingRepository;
import gr.greekchoices.admin.repository.CityRepository;
import gr.greekchoices.admin.repository.FacilityRepository;
import gr.greekchoices.admin.repository.HotelBasicInfoRepository;
import gr.greekchoices.admin.repository.HotelHasRoomTypeRepository;
import gr.greekchoices.admin.repository.HotelRepository;
import gr.greekchoices.admin.repository.ItemRepository;
import gr.greekchoices.admin.repository.LogRepository;
import gr.greekchoices.admin.repository.PeriodRepository;
```

```
import gr.greekchoices.admin.repository.PriceRepository;
import gr.greekchoices.admin.repository.RoleRepository;
import gr.greekchoices.admin.repository.RoomTypeRepository;
import gr.greekchoices.admin.repository.StarRepository;
import gr.greekchoices.admin.repository.UserRepository;
import java.awt.Color;
import java.math.BigDecimal;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import java.util.Locale;
import javax.annotation.PostConstruct;
import javax.transaction.Transactional;
import org.joda.time.DateTime;
import org.joda.time.DateTimeZone;
import org.joda.time.LocalDate;
import org.joda.time.format.DateTimeFormat;
import org.joda.time.format.DateTimeFormatter;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.stereotype.Service;
@Transactional
@Service
public class InitDbService implements Constants{
        @Autowired
        private RoleRepository roleRepository;
        @Autowired
        private UserRepository userRepository;
        @Autowired
        private BlogRepository blogRepository;
        @Autowired
        private ItemRepository itemRepository;
        @Autowired
        private HotelRepository hotelRepository;
        @Autowired
        private HotelBasicInfoRepository hotelBasicInfoRepository;
        @Autowired
        private StarRepository starRepository;
        @Autowired
        private CityRepository cityRepository;
        @Autowired
```

```
	private FacilityRepository facilityRepository;
	@Autowired
	private LogRepository logRepository;
	@Autowired
	private RoomTypeRepository roomTypeRepository;
	@Autowired
	private HotelHasRoomTypeRepository hotelHasRoomTypeRepository;
	@Autowired
	private PeriodRepository periodRepository;
	@Autowired
	private PriceRepository priceRepository;
	@Autowired
	private PriceService priceService;
	@Autowired
	private BookingRepository bookingRepository;
	@Autowired
	private AvailabilityService availabilityService;
	@PostConstruct
	public void init(){
//			if(roleRepository.findByName("ROLE_ADMIN") == null){.
//
//			}
			BCryptPasswordEncoder encoder = new BCryptPasswordEncoder();
			Role roleAdmin = new Role();
			roleAdmin.setName("ROLE_ADMIN");
			roleRepository.save(roleAdmin);
			Role roleManager = new Role();
			roleManager.setName("ROLE_MANAGER");
			roleRepository.save(roleManager);
			Role roleUser = new Role();
			roleUser.setName("ROLE_USER");
			roleRepository.save(roleUser);
			User admin = new User();
			admin.setUsername("admin");
			admin.setEmail("admin@email.com");
			admin.setStatus("1");
			admin.setPassword(encoder.encode("admin"));
			List<Role> adminRoles = new ArrayList<Role>();
			adminRoles.add(roleAdmin);
			adminRoles.add(roleUser);
			admin.setRoleList(adminRoles);
```

```
userRepository.save(admin);
/*
Log log = new Log();
log.setUser(admin);
log.setMessage("message1");
log.setLevel("info");
logRepository.save(log);
*/
User manager = new User();
manager.setUsername("manager");
manager.setEmail("manager@email.com");
manager.setStatus("1");
manager.setPassword(encoder.encode("manager"));
List<Role> managerRoles = new ArrayList<Role>();
managerRoles.add(roleManager);
managerRoles.add(roleUser);
manager.setRoleList(managerRoles);
userRepository.save(manager);
User manager2 = new User();
manager2.setUsername("manager2");
manager2.setEmail("manager2@email.com");
manager2.setStatus("1");
manager2.setPassword(encoder.encode("manager2"));
List<Role> manager2Roles = new ArrayList<Role>();
manager2Roles.add(roleManager);
manager2Roles.add(roleUser);
manager2.setRoleList(manager2Roles);
userRepository.save(manager2);
Star starOne = new Star();
starOne.setName("1 Star");
starRepository.save(starOne);
Star starTwo = new Star();
starTwo.setName("2 Star");
starRepository.save(starTwo);
Star starThree = new Star();
starThree.setName("3 Star");
starRepository.save(starThree);
Star starFour = new Star();
starFour.setName("4 Star");
starRepository.save(starFour);
Star starFive = new Star();
```

```
starFive.setName("5 Star");
starRepository.save(starFive);
City cityOne = new City();
cityOne.setName("Glyka Nera");
cityOne.setTk("15354");
cityRepository.save(cityOne);
City cityTwo = new City();
cityTwo.setName("Pallini");
cityTwo.setTk("15351");
cityRepository.save(cityTwo);
City cityThree = new City();
cityThree.setName("Ampelokipoi");
cityThree.setTk("11526");
cityRepository.save(cityThree);
Facility facilityOne = new Facility();
facilityOne.setName("Restaurant");
facilityOne.setGlobal(false);
facilityOne.setType("hotel");
facilityRepository.save(facilityOne);
Facility facilityTwo = new Facility();
facilityTwo.setName("Free! WiFi is available in all areas and is free of charge.");
facilityTwo.setGlobal(false);
facilityTwo.setType("hotel");
facilityRepository.save(facilityTwo);
Facility facilityThree = new Facility();
facilityThree.setName("Seating Area");
facilityThree.setGlobal(true);
facilityThree.setType("hotel");
facilityRepository.save(facilityThree);
Facility facilityFour = new Facility();
facilityFour.setName("Bar");
facilityFour.setGlobal(true);
facilityFour.setType("hotel");
facilityRepository.save(facilityFour);
Facility facilityFive = new Facility();
facilityFive.setName("Parking");
facilityFive.setGlobal(true);
facilityFive.setType("hotel");
facilityRepository.save(facilityFive);
Facility facilitySix = new Facility();
facilitySix.setName("Safety");
```

```
facilitySix.setGlobal(true);
facilitySix.setType("hotel");
facilityRepository.save(facilitySix);
Facility facilitySeven = new Facility();
facilitySeven.setName("Flat-screen TV");
facilitySeven.setGlobal(false);
facilitySeven.setType("roomtype");
facilityRepository.save(facilitySeven);
Facility facilityEight = new Facility();
facilityEight.setName("Air Conditioning");
facilityEight.setGlobal(false);
facilityEight.setType("roomtype");
facilityRepository.save(facilityEight);
Facility facilityNine = new Facility();
facilityNine.setName("Ironing Facilities");
facilityNine.setGlobal(false);
facilityNine.setType("roomtype");
facilityRepository.save(facilityNine);
Facility facilityTen = new Facility();
facilityTen.setName("Safety Deposit Box");
facilityTen.setGlobal(true);
facilityTen.setType("roomtype");
facilityRepository.save(facilityTen);
Facility facilityEleven = new Facility();
facilityEleven.setName("Oven");
facilityEleven.setGlobal(true);
facilityEleven.setType("roomtype");
facilityRepository.save(facilityEleven);
Facility facilityTwelve = new Facility();
facilityTwelve.setName("Refregerator");
facilityTwelve.setGlobal(true);
facilityTwelve.setType("roomtype");
facilityRepository.save(facilityTwelve);
List<Facility> facilityList = new ArrayList<Facility>();
facilityList.add(facilityFive);
facilityList.add(facilityEleven);
facilityList.add(facilityTwelve);
Hotel hotelFirst = new Hotel();
hotelFirst.setName("Panos Hotel 1");
hotelFirst.setFacilityList(facilityList);
hotelFirst.setUser(manager);
```

```
HotelBasicInfo hotelBasicInfoFirst = new HotelBasicInfo();
hotelBasicInfoFirst.setHotel(hotelFirst);
hotelBasicInfoFirst.setAddress("Othonos 32");
hotelBasicInfoFirst.setTelephone("210-6040656");
hotelBasicInfoFirst.setMobile("6936361971");
hotelBasicInfoFirst.setStar(starFive);
hotelBasicInfoFirst.setCity(cityOne);
hotelBasicInfoRepository.save(hotelBasicInfoFirst);

List<Facility> roomfacilityList = new ArrayList<Facility>();
roomfacilityList.add(facilityEleven);
roomfacilityList.add(facilityTwelve);
RoomType roomtype1 = new RoomType();
roomtype1.setName("Room type 1");
roomtype1.setGlobal(true);
roomtype1.setColor("#0000FF");
roomTypeRepository.save(roomtype1);
RoomType roomtype2 = new RoomType();
roomtype2.setName("Room type 2");
roomtype2.setGlobal(true);
roomtype2.setColor("#ff0000");
roomTypeRepository.save(roomtype2);
HotelHasRoomTypePK hotelHasRoomTypePK1 = new HotelHasRoomTypePK();
hotelHasRoomTypePK1.setHotelId(hotelFirst.getId());
hotelHasRoomTypePK1.setRoomTypeId(roomtype1.getId());
HotelHasRoomType hotelHasRoomType = new HotelHasRoomType();
hotelHasRoomType.setHotelHasRoomTypePK(hotelHasRoomTypePK1);
hotelHasRoomType.setFacilityList(roomfacilityList);
hotelHasRoomTypeRepository.save(hotelHasRoomType);
HotelHasRoomTypePK hotelHasRoomTypePK2 = new HotelHasRoomTypePK();
hotelHasRoomTypePK2.setHotelId(hotelFirst.getId());
hotelHasRoomTypePK2.setRoomTypeId(roomtype2.getId());
HotelHasRoomType hotelHasRoomType2 = new HotelHasRoomType();
hotelHasRoomType2.setHotelHasRoomTypePK(hotelHasRoomTypePK2);
hotelHasRoomType2.setFacilityList(roomfacilityList);
hotelHasRoomTypeRepository.save(hotelHasRoomType2);
List<Facility> facilityList2 = new ArrayList<Facility>();
facilityList2.add(facilityFour);
Hotel hotelSecond = new Hotel();
hotelSecond.setName("Panos Hotel 2");
hotelSecond.setUser(manager2);
```

```
HotelBasicInfo hotelBasicInfoSecond = new HotelBasicInfo();
hotelBasicInfoSecond.setHotel(hotelSecond);
hotelBasicInfoSecond.setAddress("Karaiskaki 40");
hotelBasicInfoSecond.setTelephone("213-0123454");
hotelBasicInfoSecond.setMobile("6942617300");
hotelBasicInfoSecond.setHotel(hotelSecond);
hotelBasicInfoSecond.setStar(starFour);
hotelBasicInfoSecond.setCity(cityTwo);
hotelBasicInfoRepository.save(hotelBasicInfoSecond);
hotelSecond.setFacilityList(facilityList2);
hotelRepository.save(hotelSecond);
HotelHasRoomTypePK hotelHasRoomTypePK3 = new HotelHasRoomTypePK();
hotelHasRoomTypePK3.setHotelId(hotelSecond.getId());
hotelHasRoomTypePK3.setRoomTypeId(roomtype2.getId());
HotelHasRoomType hotelHasRoomType3 = new HotelHasRoomType();
hotelHasRoomType3.setHotelHasRoomTypePK(hotelHasRoomTypePK3);
hotelHasRoomTypeRepository.save(hotelHasRoomType3);
Hotel hotelThird = new Hotel();
hotelThird.setName("Polina Hotel 3");
hotelThird.setUser(manager);
HotelBasicInfo hotelBasicInfoThird = new HotelBasicInfo();
hotelBasicInfoThird.setHotel(hotelThird);
hotelBasicInfoThird.setAddress("Mosxou 3");
hotelBasicInfoThird.setTelephone("210-6998968");
hotelBasicInfoThird.setMobile("6946705026");
hotelBasicInfoThird.setHotel(hotelThird);
hotelBasicInfoThird.setStar(starTwo);
hotelBasicInfoThird.setCity(cityThree);
hotelBasicInfoRepository.save(hotelBasicInfoThird);
HotelHasRoomTypePK hotelHasRoomTypePK4 = new HotelHasRoomTypePK();
hotelHasRoomTypePK4.setHotelId(hotelThird.getId());
hotelHasRoomTypePK4.setRoomTypeId(roomtype1.getId());
HotelHasRoomType hotelHasRoomType4 = new HotelHasRoomType();
hotelHasRoomType4.setHotelHasRoomTypePK(hotelHasRoomTypePK4);
hotelHasRoomTypeRepository.save(hotelHasRoomType4);
User user = new User();
user.setUsername("user");
user.setEmail("user@email.com");
user.setStatus("1");
user.setPassword(encoder.encode("user"));
List<Role> userRoles = new ArrayList<Role>();
```

```
                        userRoles.add(roleUser);
                        user.setRoleList(userRoles);
                        userRepository.save(user);
                        User user2 = new User();
                        user2.setUsername("user2");
                        user2.setEmail("user2@email.com");
                        user2.setStatus("1");
                        user2.setPassword(encoder.encode("user2"));
                        List<Role> userRoles2 = new ArrayList<Role>();
                        userRoles2.add(roleUser);
                        user2.setRoleList(userRoles2);
                        userRepository.save(user2);
                        String startDateAsString ="01/06/2015";
                        String endDateAsString ="31/08/2015";
                        DateTimeFormatter formater = DateTimeFormat.forPattern(PATTERN_DAY);
                        LocalDate startDate = new LocalDate(formater.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(startDateAsString));
                        LocalDate endDate = new LocalDate(formater.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(endDateAsString));
/*                      LocalDate startDate = new LocalDate(2015, 6, 1);
                        LocalDate endDate = new LocalDate(2015, 8, 31); */

/*                      LocalDate startDate = LocalDate.now();
                        LocalDate endDate = LocalDate.now();*/
                        Period period1 = new Period();
                        period1.setStartDate(startDate);
                        period1.setEndDate(endDate);
                        period1.setHotel(hotelFirst);
                        periodRepository.save(period1);
                        Price price1 = new Price();
                        price1.setCost(new BigDecimal(50));
                        priceService.save(price1, roomtype1.getId() , period1.getId(), hotelFirst.getId());
                        Price price2 = new Price();
                        price2.setCost(new BigDecimal(70));
                        priceService.save(price2, roomtype2.getId() , period1.getId(), hotelFirst.getId());
                        DateTimeFormatter bookingFormater = DateTimeFormat.forPattern(PATTERN_DAY);
                        DateTime now = new DateTime();
                        LocalDate today = now.toLocalDate();
                        LocalDate tomorrow = today.plusDays(1);
                        String booking1DateAsString     = "03/05/2015";
/*                      String startBooking1DateAsString = "10/06/2015";
                        String endBooking1DateAsString   = "12/06/2015";*/
```

```
                        LocalDate booking1Date = new LocalDate(bookingFormater.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(booking1DateAsString));
/*                      LocalDate startBooking1Date = new LocalDate(formaterBooking1.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(startBooking1DateAsString));
                        LocalDate endBooking1Date = new LocalDate(formaterBooking1.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(endBooking1DateAsString));*/
                        LocalDate startBooking1Date = now.toLocalDate().plusWeeks(2);
                        LocalDate endBooking1Date = startBooking1Date.plusDays(4);
                        Booking booking1 = new Booking();
                        booking1.setBookingDate(booking1Date);
                        booking1.setStartDate(startBooking1Date);
                        booking1.setEndDate(endBooking1Date);
                        booking1.setTotalCost(new BigDecimal(150));
                        booking1.setUser(user);
                        booking1.setHotelHasRoomType(hotelHasRoomType);
                        booking1.setStatus(BookingStatus.COMPLETED);
                        bookingRepository.save(booking1);
                        String booking2DateAsString     = "26/04/2015";
/*                      String startBooking2DateAsString = "17/07/2015";
                        String endBooking2DateAsString   = "24/07/2015";*/
                        LocalDate booking2Date = new LocalDate(bookingFormater.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(booking2DateAsString));
/*                      LocalDate startBooking2Date = new LocalDate(formaterBooking2.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(startBooking2DateAsString));
                        LocalDate endBooking2Date = new LocalDate(formaterBooking2.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(endBooking2DateAsString));*/
                        LocalDate startBooking2Date = now.toLocalDate().plusDays(5);
                        LocalDate endBooking2Date = startBooking2Date.plusWeeks(1);
                        Booking booking2 = new Booking();
                        booking2.setBookingDate(booking2Date);
                        booking2.setStartDate(startBooking2Date);
                        booking2.setEndDate(endBooking2Date);
                        booking2.setTotalCost(new BigDecimal(480));
                        booking2.setUser(user);
                        booking2.setHotelHasRoomType(hotelHasRoomType);
                        booking2.setStatus(BookingStatus.COMPLETED);
                        bookingRepository.save(booking2);
                        String booking3DateAsString     = "12/02/2015";
/*                      String startBooking3DateAsString = "03/07/2015";
                        String endBooking3DateAsString   = "07/07/2015";*/
                        LocalDate booking3Date = new LocalDate(bookingFormater.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(booking3DateAsString));
/*                      LocalDate startBooking3Date = new LocalDate(formaterBooking3.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(startBooking3DateAsString));
```

```
                        LocalDate endBooking3Date = new LocalDate(formaterBooking3.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(endBooking3DateAsString));*/
                        LocalDate startBooking3Date = now.toLocalDate().plusDays(17);
                        LocalDate endBooking3Date = startBooking3Date.plusDays(3);
                        Booking booking3 = new Booking();
                        booking3.setBookingDate(booking3Date);
                        booking3.setStartDate(startBooking3Date);
                        booking3.setEndDate(endBooking3Date);
                        booking3.setTotalCost(new BigDecimal(260));
                        booking3.setUser(user2);
                        booking3.setHotelHasRoomType(hotelHasRoomType2);
                        booking3.setStatus(BookingStatus.PENDING);
                        bookingRepository.save(booking3);
                        String booking4DateAsString     = "12/03/2015";
/*                      String startBooking4DateAsString = "01/08/2015";
                        String endBooking4DateAsString   = "05/08/2015";*/
                          LocalDate booking4Date = new LocalDate(bookingFormater.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(booking4DateAsString));
/*                      LocalDate startBooking4Date = new LocalDate(formaterBooking4.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(startBooking4DateAsString));
                        LocalDate endBooking4Date = new LocalDate(formaterBooking4.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(endBooking4DateAsString));*/
                        LocalDate startBooking4Date = now.toLocalDate().plusDays(6);
                        LocalDate endBooking4Date = startBooking4Date.plusDays(5);
                        Booking booking4 = new Booking();
                        booking4.setBookingDate(booking4Date);
                        booking4.setStartDate(startBooking4Date);
                        booking4.setEndDate(endBooking4Date);
                        booking4.setTotalCost(new BigDecimal(120));
                        booking4.setUser(user2);
                        booking4.setHotelHasRoomType(hotelHasRoomType3);
                        booking4.setStatus(BookingStatus.PENDING);
                        bookingRepository.save(booking4);
                        String booking5DateAsString     = "12/03/2015";
/*                      String startBooking5DateAsString = "11/09/2015";
                        String endBooking5DateAsString   = "23/09/2015";*/
                          LocalDate booking5Date = new LocalDate(bookingFormater.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(booking5DateAsString));
/*                      LocalDate startBooking5Date = new LocalDate(formaterBooking5.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(startBooking5DateAsString));
                        LocalDate endBooking5Date = new LocalDate(formaterBooking5.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(endBooking5DateAsString));*/


                        LocalDate startBooking5Date = now.toLocalDate().plusDays(20);
```

```
LocalDate endBooking5Date = startBooking5Date.plusDays(6);
Booking booking5 = new Booking();
booking5.setBookingDate(booking5Date);
booking5.setStartDate(startBooking5Date);
booking5.setEndDate(endBooking5Date);
booking5.setTotalCost(new BigDecimal(1150));
booking5.setUser(user);
booking5.setHotelHasRoomType(hotelHasRoomType4);
booking5.setStatus(BookingStatus.COMPLETED);
bookingRepository.save(booking5);
// Availability (Starts)
LocalDate availability1aDate = now.toLocalDate().plusDays(5);
Availability availability1a = new Availability();
availability1a.setQuantity(10);
availabilityService.save(availability1a, roomtype1.getId(), hotelFirst.getId(), availability1aDate);
LocalDate availability1bDate = now.toLocalDate().plusDays(7);
Availability availability1b = new Availability();
availability1b.setQuantity(2);
availabilityService.save(availability1b, roomtype1.getId(), hotelFirst.getId(), availability1bDate);
LocalDate availability1cDate = now.toLocalDate().plusDays(8);
Availability availability1c = new Availability();
availability1c.setQuantity(4);
availabilityService.save(availability1c, roomtype1.getId(), hotelFirst.getId(), availability1cDate);
LocalDate availability1dDate = now.toLocalDate().plusDays(9);
Availability availability1d = new Availability();
availability1d.setQuantity(7);
availabilityService.save(availability1d, roomtype1.getId(), hotelFirst.getId(), availability1dDate);
LocalDate availability1eDate = now.toLocalDate().plusDays(11);
Availability availability1e = new Availability();
availability1e.setQuantity(6);
availabilityService.save(availability1e, roomtype1.getId(), hotelFirst.getId(), availability1eDate);
LocalDate availability2Date = now.toLocalDate().plusDays(7);
Availability availability2 = new Availability();
availability2.setQuantity(6);
availabilityService.save(availability2, roomtype2.getId(), hotelFirst.getId(), availability2Date);
LocalDate availability3Date = now.toLocalDate().plusDays(2);
Availability availability3 = new Availability();
availability3.setQuantity(2);
availabilityService.save(availability3, roomtype2.getId(), hotelSecond.getId(), availability3Date);
LocalDate availability4Date = now.toLocalDate().plusDays(15);
Availability availability4 = new Availability();
```

```
availability4.setQuantity(2);

availabilityService.save(availability4, roomtype1.getId(), hotelThird.getId(), availability4Date);

// Availability (Ends)

/*  DateTimeFormatter formaterEvent = DateTimeFormat.forPattern(PATTERN_EVENT_DAY);
/*  DateTimeFormatter formaterEvent = DateTimeFormat.forPattern(PATTERN_EVENT_DAY);

String startEvent1DateAsString = "2014-11-10";

String endEvent1DateAsString   = "2014-11-15";

LocalDate startEvent1Date = new LocalDate(formaterEvent.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(startEvent1DateAsString));

 LocalDate endEvent1Date = new LocalDate(formaterEvent.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(endEvent1DateAsString));

Event event1 = new  Event("Αλεξάκης Νικόλαος", startEvent1Date, endEvent1Date, "blue",
"http://google.com/");

*/

/*

String startEvent2DateAsString = "2014-11-10";

String endEvent2DateAsString   = "2014-11-10";

LocalDate startEvent2Date = new LocalDate(formaterEvent.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(startEvent2DateAsString));

 LocalDate endEvent2Date = new LocalDate(formaterEvent.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(endEvent2DateAsString));

Event event2 = new  Event("Παπάζογλου Πάνος", startEvent2Date, endEvent2Date, "green",
"http://google.com/");

String startEvent3DateAsString = "2014-11-11";

String endEvent3DateAsString   = "2014-11-13";

LocalDate startEvent3Date = new LocalDate(formaterEvent.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(startEvent3DateAsString));

 LocalDate endEvent3Date = new LocalDate(formaterEvent.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(endEvent3DateAsString));

Event event3 = new  Event("Βασιλάκου Πολύμνια", startEvent3Date, endEvent3Date, "blue",
"http://google.com/");

String startEvent4DateAsString = "2014-11-12";

String endEvent4DateAsString   = "2014-11-16";

LocalDate startEvent4Date = new LocalDate(formaterEvent.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(startEvent4DateAsString));

 LocalDate endEvent4Date = new LocalDate(formaterEvent.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(endEvent4DateAsString));

Event event4 = new  Event("Βασιλάκου Πολύμνια", startEvent4Date, endEvent4Date, "orange",
"http://google.com/");

String startEvent5DateAsString = "2014-11-12";

String endEvent5DateAsString   = "2014-11-16";

LocalDate startEvent5Date = new LocalDate(formaterEvent.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(startEvent5DateAsString));

 LocalDate endEvent5Date = new LocalDate(formaterEvent.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(endEvent5DateAsString));

Event event5 = new  Event("Παππούς Πέτρος", startEvent5Date, endEvent5Date, "orange",
"http://google.com/");
```

```
                String startEvent6DateAsString = "2014-11-7";

                String endEvent6DateAsString   = "2014-11-13";

                LocalDate startEvent6Date = new LocalDate(formaterEvent.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(startEvent6DateAsString));

                 LocalDate endEvent6Date = new LocalDate(formaterEvent.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(endEvent6DateAsString));

                Event event6 = new  Event("Τρουγκάκος Σπύρος", startEvent6Date, endEvent6Date, "purple",
"http://google.com/");

                String startEvent7DateAsString = "2014-11-9";

                String endEvent7DateAsString   = "2014-11-13";

                LocalDate startEvent7Date = new LocalDate(formaterEvent.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(startEvent7DateAsString));

                 LocalDate endEvent7Date = new LocalDate(formaterEvent.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(endEvent7DateAsString));

                Event event7 = new  Event("Δαμιανός Πιστώλας", startEvent7Date, endEvent7Date, "red",
"http://google.com/");

                String startEvent8DateAsString = "2014-11-26";

                String endEvent8DateAsString   = "2014-11-30";

                LocalDate startEvent8Date = new LocalDate(formaterEvent.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(startEvent8DateAsString));

                 LocalDate endEvent8Date = new LocalDate(formaterEvent.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(endEvent8DateAsString));

                Event event8 = new  Event("Πικής Μιχάλης", startEvent8Date, endEvent8Date, "blue", "http://google.com/");
                */

        }
}
```

9.10.4.9 PeriodService.java

```
package gr.greekchoices.admin.service;

import gr.greekchoices.Constants;

import gr.greekchoices.admin.entity.Hotel;

import gr.greekchoices.admin.entity.Period;

import gr.greekchoices.admin.repository.HotelRepository;

import gr.greekchoices.admin.repository.PeriodRepository;

import java.util.HashMap;

import java.util.List;

import java.util.Locale;

import org.apache.log4j.Logger;

import org.joda.time.DateTimeZone;

import org.joda.time.LocalDate;

import org.joda.time.format.DateTimeFormat;

import org.joda.time.format.DateTimeFormatter;

import org.springframework.beans.factory.annotation.Autowired;
```

```
import org.springframework.security.access.method.P;
import org.springframework.security.access.prepost.PreAuthorize;
import org.springframework.stereotype.Service;
@Service
public class PeriodService implements Constants{
	private static final Logger logger = Logger
					.getLogger(PeriodService.class);
	@Autowired
	private HotelRepository hotelRepository;
	@Autowired
	private PeriodRepository periodRepository;
	public HashMap<Long , Period> allPeriodPerHotel(int hotelId) {
		HashMap<Long, Period> allPeriodPerHotel = new HashMap<Long, Period>();
		Hotel hotel = hotelRepository.findOne(hotelId);
		List<Period> periodList = periodRepository.findByHotelOrderByStartDateAsc(hotel);
		for(Period period : periodList){
			allPeriodPerHotel.put(new Long(period.getId()), period);
		}
		return allPeriodPerHotel;
	}
	public void save(Period period, int hotelId) {
		Hotel hotel = hotelRepository.findOne(hotelId);
		period.setHotel(hotel);
/*		String startDateAsString ="01/08/2015";
		DateTimeFormatter formater = DateTimeFormat.forPattern(PATTERN_DAY);
		LocalDate startDate = new LocalDate(formater.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(startDateAsString));
		period.setStartDate(startDate);*/
		periodRepository.save(period);
		logger.debug("Period \""+period.getStartDate()+" - "+period.getEndDate()+"\" has been stored!!");
	}
	@PreAuthorize("#period.hotel.user.username == authentication.name or hasRole('ROLE_ADMIN')")
	public void remove(@P("period") Period period) {
		periodRepository.delete(period);
		logger.debug("Period ["+period.getStartDate()+"-"+period.getEndDate()+"] deleted!!");
	}
	public Period findOne(int periodId) {
		return periodRepository.findOne(periodId);
	}
}
```

9.10.4.10 PriceService.java

```
package gr.greekchoices.admin.service;
import java.math.BigDecimal;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import javax.transaction.Transactional;
import gr.greekchoices.admin.entity.Hotel;
import gr.greekchoices.admin.entity.HotelHasRoomType;
import gr.greekchoices.admin.entity.HotelHasRoomTypePK;
import gr.greekchoices.admin.entity.Period;
import gr.greekchoices.admin.entity.Price;
import gr.greekchoices.admin.entity.PricePK;
import gr.greekchoices.admin.entity.RoomType;
import gr.greekchoices.admin.repository.HotelRepository;
import gr.greekchoices.admin.repository.PeriodRepository;
import gr.greekchoices.admin.repository.PriceRepository;
import gr.greekchoices.admin.repository.RoomTypeRepository;
import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
@Service
public class PriceService {
        private static final Logger logger = Logger
                        .getLogger(PriceService.class);
        @Autowired
        private PriceRepository priceRepository;
        @Autowired
        private RoomTypeService roomTypeService;
        @Autowired
        private RoomTypeRepository roomTypeRepository;
        @Autowired
        private HotelRepository hotelRepository;
        @Autowired
        private PeriodService periodService;
        @Autowired
```

```
	private PeriodRepository periodRepository;
	@Transactional
	public HashMap<Long , Price> allPricePerAllRoomTypePerPeriodPerHotel(int periodId, int hotelId) {
			HashMap<Long , Price> allPricePerAllRoomTypePerPeriodPerHotel = new HashMap<Long, Price>();
		HashMap<Long , RoomType> allRoomTypePerHotel = roomTypeService.allRoomTypePerHotel(hotelId);
			for(Map.Entry<Long, RoomType> entryOfRoomType : allRoomTypePerHotel.entrySet() ) {
				PricePK pricePK = new PricePK();
				pricePK.setHotelId(hotelId);
				pricePK.setPeriodId(periodId);
				pricePK.setRoomTypeId(entryOfRoomType.getKey().intValue());
				Price price = priceRepository.findByPricePK(pricePK);
				logger.debug("RoomType Id:"+entryOfRoomType.getKey()+" |
Name:"+entryOfRoomType.getValue().getName());
				if(price != null){
					logger.debug("Price Cost:"+price.getCost());
				}
				logger.debug("--------------------------------");
				allPricePerAllRoomTypePerPeriodPerHotel.put(entryOfRoomType.getKey(), price);
			}
		return allPricePerAllRoomTypePerPeriodPerHotel;
	}
	@Transactional
	public HashMap<Long, HashMap<Long , Price>> allPricePerAllRoomTypePerAllPeriodPerHotel(int hotelId) {
		HashMap<Long, HashMap<Long , Price>> allPricePerAllRoomTypePerAllPeriodPerHotel = new HashMap<Long,
HashMap<Long , Price>>();
		HashMap<Long , Period> allPeriodPerHotel = periodService.allPeriodPerHotel(hotelId);
		HashMap<Long , Price> allPricePerAllRoomTypePerPeriodPerHotel;
		for(Map.Entry<Long, Period> entryOfPeriod : allPeriodPerHotel.entrySet() ) {
			logger.debug("Period Id:"+entryOfPeriod.getKey()+" | Start
Date:"+entryOfPeriod.getValue().getStartDate()+" | End Date:"+entryOfPeriod.getValue().getEndDate());
			logger.debug("============================================================");
			allPricePerAllRoomTypePerPeriodPerHotel =
allPricePerAllRoomTypePerPeriodPerHotel(entryOfPeriod.getKey().intValue(), hotelId);

			allPricePerAllRoomTypePerAllPeriodPerHotel.put(entryOfPeriod.getKey(),
allPricePerAllRoomTypePerPeriodPerHotel);
		}
		// logger (Starts)
		logger.debug("");
		logger.debug("#################################################");
		logger.debug("");
		for(Map.Entry<Long, HashMap<Long , Price>> entryOfPeriod1 :
allPricePerAllRoomTypePerAllPeriodPerHotel.entrySet() ) {
```

```
                                logger.debug("Period Id:"+entryOfPeriod1.getKey()+" | Start
Date:"+periodRepository.findOne(entryOfPeriod1.getKey().intValue()).getStartDate() +" | End
Date:"+periodRepository.findOne(entryOfPeriod1.getKey().intValue()).getEndDate());
                                logger.debug("============================================================");
                                for(Map.Entry<Long, Price> entryOfRoomType1 : entryOfPeriod1.getValue().entrySet() ) {
                                        logger.debug("RoomType Id:"+entryOfRoomType1.getKey()+" |
Name:"+roomTypeRepository.findOne(entryOfRoomType1.getKey().intValue()).getName());
                                        if(entryOfRoomType1.getValue() != null){
                                                logger.debug("Price Cost:"+entryOfRoomType1.getValue().getCost());
                                        }
                                        logger.debug("---------------------------------");
                                }
                        }
                        // logger (Ends)
                        return allPricePerAllRoomTypePerAllPeriodPerHotel;
                }
                @Transactional
                public HashMap<Long , Price> allPricePerAllPeriodPerRoomTypePerHotel(int roomTypeId, int hotelId) {
                                HashMap<Long , Price> allPricePerAllPeriodPerRoomTypePerHotel = new HashMap<Long, Price>();
                        HashMap<Long , Period> allPeriodPerHotel = periodService.allPeriodPerHotel(hotelId);
                        for(Map.Entry<Long, Period> entryOfPeriod : allPeriodPerHotel.entrySet() ) {
                                PricePK pricePK = new PricePK();
                                pricePK.setHotelId(hotelId);
                                pricePK.setPeriodId(entryOfPeriod.getKey().intValue());
                                pricePK.setRoomTypeId(roomTypeId);
                                Price price = priceRepository.findByPricePK(pricePK);
                                logger.debug("Period Id:"+entryOfPeriod.getKey()+" | Start
Date:"+entryOfPeriod.getValue().getStartDate()+" | End Date:"+entryOfPeriod.getValue().getEndDate());
                                if(price != null){
                                        logger.debug("Price Cost:"+price.getCost());
                                }
                                logger.debug("---------------------------------");

                                allPricePerAllPeriodPerRoomTypePerHotel.put(entryOfPeriod.getKey(), price);
                        }
                        return allPricePerAllPeriodPerRoomTypePerHotel;
                }
                @Transactional
                public HashMap<Long, HashMap<Long , Price>> allPricePerAllPeriodPerAllRoomTypePerHotel(int hotelId) {
                        HashMap<Long, HashMap<Long , Price>> allPricePerAllPeriodPerAllRoomTypePerHotel = new HashMap<Long,
HashMap<Long , Price>>();
                        HashMap<Long , RoomType> allRoomTypePerHotel = roomTypeService.allRoomTypePerHotel(hotelId);
                        HashMap<Long , Price> allPricePerAllPeriodPerRoomTypePerHotel;
```

```
                for(Map.Entry<Long, RoomType> entryOfRoomType : allRoomTypePerHotel.entrySet() ) {
                        logger.debug("RoomType Id:"+entryOfRoomType.getKey()+" |
Name:"+entryOfRoomType.getValue().getName());
                        logger.debug("============================================================");
                        allPricePerAllPeriodPerRoomTypePerHotel =
allPricePerAllPeriodPerRoomTypePerHotel(entryOfRoomType.getKey().intValue(), hotelId);
                        allPricePerAllPeriodPerAllRoomTypePerHotel.put(entryOfRoomType.getKey(),
allPricePerAllPeriodPerRoomTypePerHotel);
                }
                // logger (Starts)
                logger.debug("");
                logger.debug("#####################################################");
                logger.debug("");
                for(Map.Entry<Long, HashMap<Long , Price>> entryOfRoomType1 :
allPricePerAllPeriodPerAllRoomTypePerHotel.entrySet() ) {
                        logger.debug("RoomType Id:"+entryOfRoomType1.getKey()+" |
Name:"+roomTypeRepository.findOne(entryOfRoomType1.getKey().intValue()).getName());
                        logger.debug("============================================================");
                        for(Map.Entry<Long, Price> entryOfPeriod1 : entryOfRoomType1.getValue().entrySet() ) {
                                logger.debug("Period Id:"+entryOfPeriod1.getKey()+" | Start
Date:"+periodRepository.findOne(entryOfPeriod1.getKey().intValue()).getStartDate() +" | End
Date:"+periodRepository.findOne(entryOfPeriod1.getKey().intValue()).getEndDate());
                                if(entryOfPeriod1.getValue() != null){
                                        logger.debug("Price Cost:"+entryOfPeriod1.getValue().getCost());
                                }
                                logger.debug("--------------------------------");
                        }
                }
                // logger (Ends)
                return allPricePerAllPeriodPerAllRoomTypePerHotel;
        }
        public void save(Price price, int roomTypeId, int periodId, int hotelId) {

                PricePK pricePK = new PricePK();
                pricePK.setHotelId(hotelId);
                pricePK.setPeriodId(periodId);
                pricePK.setRoomTypeId(roomTypeId);
                price.setPricePK(pricePK);
                priceRepository.save(price);
                logger.debug("Price Saved [priceCost:\""+price.getCost()+"\" , roomtypeId:\""+roomTypeId+"\" ,
periodId:\""+periodId+"\" , hotelId:\""+hotelId+"\"]");
        }
        public List<Price> allPricePerHotel(int hotelId){
                List<Price> priceList = priceRepository.findAll();
                for(Price price:priceList){
```

```
                                    logger.debug("Price Cost:"+price.getCost()+"[hotelId:"+price.getPricePK().getHotelId()+" ,
periodId:"+price.getPricePK().getPeriodId()+" , roomTypeId:"+price.getPricePK().getRoomTypeId()+"]");
                        }
                        return priceList;
            }
}
```

9.10.4.11 RoleService.java

```
package gr.greekchoices.admin.service;
import gr.greekchoices.admin.repository.RoleRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
@Service
public class RoleService {
            @Autowired
            private RoleRepository roleRepository;
}
```

9.10.4.12 RoomTypeService.java

```
package gr.greekchoices.admin.service;
import gr.greekchoices.admin.entity.Facility;
import gr.greekchoices.admin.entity.Hotel;
import gr.greekchoices.admin.entity.HotelHasRoomType;
import gr.greekchoices.admin.entity.HotelHasRoomTypePK;
import gr.greekchoices.admin.entity.Period;
import gr.greekchoices.admin.entity.Price;
import gr.greekchoices.admin.entity.RoomType;
import gr.greekchoices.admin.repository.HotelHasRoomTypeRepository;
import gr.greekchoices.admin.repository.HotelRepository;
import gr.greekchoices.admin.repository.PeriodRepository;
import gr.greekchoices.admin.repository.RoomTypeRepository;
import java.util.ArrayList;
import java.util.HashMap;
```

```
import java.util.Iterator;
import java.util.List;
import java.util.Map;
import javax.transaction.Transactional;
import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
@Service
public class RoomTypeService {
	private static final Logger logger = Logger
				.getLogger(RoomTypeService.class);
	@Autowired
	private RoomTypeRepository roomTypeRepository;
	@Autowired
	private HotelRepository hotelRepository;
	@Autowired
	private PeriodService periodService;
	@Autowired
	private PriceService priceService;
	@Autowired
	private HotelHasRoomTypeRepository hotelHasRoomTypeRepository;
	@Transactional
	public List<RoomType> findAllRoomTypeByGlobal(Boolean booleanParam){
		List<RoomType> roomTypeList = new ArrayList<RoomType>();
		roomTypeList = roomTypeRepository.findByGlobal(booleanParam);
		for (RoomType roomType : roomTypeList) {
			logger.debug("#roomType : "+roomType.getId()+" | "+roomType.getName()+"#");
		}
		return roomTypeList;
	}
	@Transactional
	public HashMap<Long, RoomType> allRoomTypePerHotel(int hotelId) {
		Hotel hotel = hotelRepository.findOne(hotelId);
		List<HotelHasRoomType> hotelHasRoomTypeList = hotelHasRoomTypeRepository.findByHotel(hotel);
		HashMap<Long,RoomType> allRoomTypePerHotel = new HashMap<Long, RoomType>();
		logger.debug(".::findAllRoomTypeByHotelId::.");
		for (HotelHasRoomType hotelHasRoomType : hotelHasRoomTypeList) {
			allRoomTypePerHotel.put(new Long(hotelHasRoomType.getRoomType().getId()) ,
hotelHasRoomType.getRoomType());
			logger.debug("roomTypeId: "+hotelHasRoomType.getRoomType().getId()+" | roomTypeName:
"+hotelHasRoomType.getRoomType().getName());
		}
```

```
                return allRoomTypePerHotel;
        }
        @Transactional
        public List<RoomType> findAllRoomTypeByHotelId(int hotelId) {
                Hotel hotel = hotelRepository.findOne(hotelId);
                List<HotelHasRoomType> hotelHasRoomTypeList = hotelHasRoomTypeRepository.findByHotel(hotel);
                List<RoomType> roomTypeList = new ArrayList<RoomType>();
                logger.debug(".::findAllRoomTypeByHotelId::.");
                for (HotelHasRoomType hotelHasRoomType : hotelHasRoomTypeList) {
                        roomTypeList.add(hotelHasRoomType.getRoomType());
                        logger.debug("roomTypeId: "+hotelHasRoomType.getRoomType().getId()+" | roomTypeName:
"+hotelHasRoomType.getRoomType().getName());
                }
                logger.debug("");
                return roomTypeList;
        }
        @Transactional
        public List<RoomType> findAllGlobalRoomTypeToAdd(int hotelId) {
                List<RoomType> RoomTypeList = findAllRoomTypeByGlobal(new Boolean(true));
                List<RoomType> RoomTypeByHotelIdList = findAllRoomTypeByHotelId(hotelId);
                List<RoomType> globalRoomTypeToAddList = RoomTypeList;
                globalRoomTypeToAddList.removeAll(RoomTypeByHotelIdList);
                return globalRoomTypeToAddList;
        }
        @Transactional
        public List<RoomType> findAllCustomRoomType(int hotelId) {
                // Briskoume ola ta custom Room Types poy iparxoun stin basi..
                List<RoomType> customRoomTypeListAll = findAllRoomTypeByGlobal(new Boolean(false));
                // Briskoume ola ta Room Types (custom kai global) poy uparxoyn stin basi gia to sigkekrimeno hotel
                List<RoomType> roomTypeListByHotelId = findAllRoomTypeByHotelId(hotelId);
                // Thetoume katarxas ta Room Type poy einai custom gia ola ta alla xenodoxeia me ola ta custom pouy iparxoyn
stin basi..
                List<RoomType> customRoomTypeListPerAllOtherHotels = customRoomTypeListAll;
                // kai apo ayta..afairoyme ola ta room types pou yparxoyn se ena xenodoxeio (custom kai global) opou profanos
tha aferethoun mono ta custom toy xenodoxeiou..
                customRoomTypeListPerAllOtherHotels.removeAll(roomTypeListByHotelId);
                // Thetoume ta room types poy einai custom se ena xenodoxeio me ola ta custom poy iparxoun stin basi
                List<RoomType> customRoomTypeListPerHotel = findAllRoomTypeByGlobal(new Boolean(false));
                // kai telos apo ola ta custom..afairome ayta pouy brikame parapano..pou aforoun ola ta custom apo ola ta alla
xenodoxeia mono..
                // kai sinepos menoume me ta custom..tou sigkekrimenou xenodoxeiou
                customRoomTypeListPerHotel.removeAll(customRoomTypeListPerAllOtherHotels);
                return customRoomTypeListPerHotel;
```

```
	}
	@Transactional
	public void save(RoomType roomType, String username, int hotelId) {
		logger.debug(roomType.getGlobal());
		// An to global einai true...mas irthe meso ADD apo ta pre-configured room types.
		if(roomType.getGlobal()){
			RoomType roomTypeToUpdateInDb = roomTypeRepository.findOne(roomType.getId());
			Hotel hotel = hotelRepository.findOne(hotelId);
			HotelHasRoomTypePK hotelHasRoomTypePK = new HotelHasRoomTypePK();
			hotelHasRoomTypePK.setHotelId(hotelId);
			hotelHasRoomTypePK.setRoomTypeId(roomTypeToUpdateInDb.getId());
		  HotelHasRoomType hotelHasRoomType = new HotelHasRoomType();
		  hotelHasRoomType.setHotelHasRoomTypePK(hotelHasRoomTypePK);
		  hotelHasRoomType.setRoomType(roomTypeToUpdateInDb);
		  hotelHasRoomType.setHotel(hotel);
		  hotelHasRoomTypeRepository.save(hotelHasRoomType);
			logger.debug("Added \""+roomTypeToUpdateInDb.getName()+"\"");
		// An to global einai false...mas irthe meso NEW
		}else{
			roomTypeRepository.save(roomType);
			Hotel hotel = hotelRepository.findOne(hotelId);
			HotelHasRoomTypePK hotelHasRoomTypePK = new HotelHasRoomTypePK();
			hotelHasRoomTypePK.setHotelId(hotelId);
			hotelHasRoomTypePK.setRoomTypeId(roomType.getId());
		  HotelHasRoomType hotelHasRoomType = new HotelHasRoomType();
		  hotelHasRoomType.setHotelHasRoomTypePK(hotelHasRoomTypePK);
		  hotelHasRoomType.setRoomType(roomType);
		  hotelHasRoomType.setHotel(hotel);
		  hotelHasRoomTypeRepository.save(hotelHasRoomType);
			logger.debug("Room Type \""+roomType.getName()+"\" has been stored!!");
		}
	}
	public RoomType findOne(int roomTypeId) {
		return roomTypeRepository.findOne(roomTypeId);
	}
	public void remove(RoomType roomType, Hotel hotel) {
		if(roomType.getGlobal()){
			HotelHasRoomTypePK hotelHasRoomTypePK = new HotelHasRoomTypePK();
			hotelHasRoomTypePK.setHotelId(hotel.getId());
			hotelHasRoomTypePK.setRoomTypeId(roomType.getId());
		  HotelHasRoomType hotelHasRoomType =
hotelHasRoomTypeRepository.findByHotelHasRoomTypePK(hotelHasRoomTypePK);
```

```
                        hotelHasRoomTypeRepository.delete(hotelHasRoomType);
                                logger.debug("Room Type ["+roomType.getName()+"] removed!!");
                        }else{
                                roomTypeRepository.delete(roomType);
                                logger.debug("Room Type ["+roomType.getName()+"] deleted!!");
                        }
        }
        public List<RoomType> getCustomRoomTypesPerHotel(Hotel hotel) {
                List<RoomType> customRoomTypeList = findAllCustomRoomType(hotel.getId());
                return customRoomTypeList;
        }
        public void removeCustomRoomTypes(List<RoomType> customRoomTypeList) {
                for(RoomType customRoomType : customRoomTypeList){
                        logger.debug("Deleting custom roomt type with name  "+customRoomType.getName());
                        roomTypeRepository.delete(customRoomType);
                        logger.debug("Room Type with name '"+customRoomType.getName()+"' deleted succesfully!!");
                }
        }
}
```

9.10.4.13 StarService.java

```
package gr.greekchoices.admin.service;
import java.util.HashMap;
import java.util.List;
import javax.transaction.Transactional;
import gr.greekchoices.admin.entity.City;
import gr.greekchoices.admin.entity.Star;
import gr.greekchoices.admin.repository.StarRepository;
import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
@Service
public class StarService {
        private static final Logger logger = Logger
                        .getLogger(StarService.class);
        @Autowired
        private StarRepository starRepository;
        @Transactional
```

```
	public HashMap<Long, Star> getAllStars() {
		HashMap<Long,Star> getAllStars = new HashMap<Long, Star>();
		List<Star> starList = starRepository.findAll();
		logger.debug(".::findAllStars::.");
		for (Star star : starList) {
			getAllStars.put(new Long(star.getId()) , star);
			logger.debug("starId: "+star.getId()+" | starName: "+star.getName());
		}
		return getAllStars;
	}
}
```

9.10.4.14 UserService.java

```
package gr.greekchoices.admin.service;
import gr.greekchoices.admin.entity.Blog;
import gr.greekchoices.admin.entity.Item;
import gr.greekchoices.admin.entity.Role;
import gr.greekchoices.admin.entity.User;
import gr.greekchoices.admin.repository.BlogRepository;
import gr.greekchoices.admin.repository.ItemRepository;
import gr.greekchoices.admin.repository.RoleRepository;
import gr.greekchoices.admin.repository.UserRepository;
import java.util.ArrayList;
import java.util.List;
import javax.transaction.Transactional;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.PageRequest;
import org.springframework.data.domain.Sort.Direction;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.stereotype.Service;
@Service
@Transactional
public class UserService {
	@Autowired
	private UserRepository userRepository;
	@Autowired
	private RoleRepository roleRepository;
	@Autowired
	private BlogRepository blogRepository;
	@Autowired
```

```
	private ItemRepository itemRepository;
	public List<User> findAll(){
		return userRepository.findAll();
	}
	public User findOne(int id) {
		return userRepository.findOne(id);
	}
	@Transactional
	public User findOne(String username){
		return userRepository.findByUsername(username);
	}
	@Transactional
	public User findOneWithBlogs(int id) {
		User user = findOne(id);
		List<Role> roles = new ArrayList<Role>();
		for (Role role : user.getRoleList()){
		  roles.add(roleRepository.findOne(role.getId()));
		}
		user.setRoleList(roles);
		List<Blog> blogs = blogRepository.findByUser(user);
		for (Blog blog : blogs) {
			List<Item> items = itemRepository.findByBlog(blog,new PageRequest(0, 10, Direction.DESC,
"publishedDate"));
			blog.setItems(items);
		}
		user.setBlogs(blogs);
		return user;
	}
	public void save(User user) {
		user.setStatus("1");
		BCryptPasswordEncoder encoder = new BCryptPasswordEncoder();
		user.setPassword(encoder.encode(user.getPassword()));
		List<Role> roles = new ArrayList<Role>();
		roles.add(roleRepository.findByName("ROLE_MANAGER") );
		roles.add(roleRepository.findByName("ROLE_USER") );
		user.setRoleList(roles);
		userRepository.save(user);
	}
	@Transactional
	public User findOneWithBlogs(String username) {
		User user = userRepository.findByUsername(username);
		return findOneWithBlogs(user.getId());
```

```
	}
	@Transactional
	public Object findOneWithRoles(String username) {
		User user = userRepository.findByUsername(username);
		List<Role> roles = new ArrayList<Role>();
		for (Role role : user.getRoleList()){
		  roles.add(roleRepository.findOne(role.getId()));
		}
		user.setRoleList(roles);
		return user;
	}
}
```

9.10.5 Model

Στο πακέτο "gr.greekchoices.admin.model" βρίσκονται τα αρχεία πηγαίου κώδικα, που παρουσιάζονται παρακάτω, και έχουν να κάνουν με το model αλλά δεν είναι entities στην βάση της λογικής του JPA. Πρόκειται για βοηθητικές κλάσεις κυρίως για λόγους μετασχηματισμού από αντικείμενα σε JSON (marslaing/unmarshaling) μέσω του Jackson Library στα πλαίσια των webservices / REST μέσω ασύχρονων κλήσεων (Ajax Calls) για την ορθή λειτουργία υποσυστημάτων όπως το ημερολόγιο κρατήσεων(Booking Calendar), το ημερολόγιο διαθεσιμότητας (Availability Calendar) κ.α.

9.10.5.1 AvailabilityCalendarData.java

```
package gr.greekchoices.admin.model;
import java.math.BigDecimal;
public class AvailabilityCalendarData {
	/*		"available":"10","bind":0,"info":"","notes":"","price":"","promo":"","status":"available" */
	private String available;
	private Integer bind;
	private String info;
	private String notes;
	private String price;
	private String promo;
	private String status;
	public AvailabilityCalendarData(String available, Integer bind,
				String info, String notes, String price, String promo, String status) {
		super();
		this.available = available;
		this.bind = bind;
		this.info = info;
		this.notes = notes;
		this.price = price;
		this.promo = promo;
		this.status = status;
	}
	public String getAvailable() {
		return available;
	}
	public void setAvailable(String available) {
		this.available = available;
```

```
	}
	public Integer getBind() {
		return bind;
	}
	public void setBind(Integer bind) {
		this.bind = bind;
	}
	public String getInfo() {
		return info;
	}
	public void setInfo(String info) {
		this.info = info;
	}
	public String getNotes() {
		return notes;
	}
	public void setNotes(String notes) {
		this.notes = notes;
	}
	public String getPrice() {
		return price;
	}
	public void setPrice(String price) {
		this.price = price;
	}
	public String getPromo() {
		return promo;
	}
	public void setPromo(String promo) {
		this.promo = promo;
	}
	public String getStatus() {
		return status;
	}
	public void setStatus(String status) {
		this.status = status;
	}
	@Override
	public int hashCode() {
		final int prime = 31;
		int result = 1;
```

```
                result = prime * result
                                + ((available == null) ? 0 : available.hashCode());
                result = prime * result + ((bind == null) ? 0 : bind.hashCode());
                result = prime * result + ((info == null) ? 0 : info.hashCode());
                result = prime * result + ((notes == null) ? 0 : notes.hashCode());
                result = prime * result + ((price == null) ? 0 : price.hashCode());
                result = prime * result + ((promo == null) ? 0 : promo.hashCode());
                result = prime * result + ((status == null) ? 0 : status.hashCode());
                return result;
        }
        @Override
        public boolean equals(Object obj) {
                if (this == obj)
                        return true;
                if (obj == null)
                        return false;
                if (getClass() != obj.getClass())
                        return false;
                AvailabilityCalendarData other = (AvailabilityCalendarData) obj;
                if (available == null) {
                        if (other.available != null)
                                return false;
                } else if (!available.equals(other.available))
                        return false;
                if (bind == null) {
                        if (other.bind != null)
                                return false;
                } else if (!bind.equals(other.bind))
                        return false;
                if (info == null) {
                        if (other.info != null)
                                return false;
                } else if (!info.equals(other.info))
                        return false;
                if (notes == null) {
                        if (other.notes != null)
                                return false;
                } else if (!notes.equals(other.notes))
                        return false;
                if (price == null) {
                        if (other.price != null)
```

```
                                        return false;
                        } else if (!price.equals(other.price))
                                return false;
                        if (promo == null) {
                                if (other.promo != null)
                                        return false;
                        } else if (!promo.equals(other.promo))
                                return false;
                        if (status == null) {
                                if (other.status != null)
                                        return false;
                        } else if (!status.equals(other.status))
                                return false;
                        return true;
                }
                @Override
                public String toString() {
                        return "AvailabilityCalendarData [available=" + available + ", bind="
                                        + bind + ", info=" + info + ", notes=" + notes + ", price="
                                        + price + ", promo=" + promo + ", status=" + status + "]";
                }
}
```

9.10.5.2 Event.java

```
package gr.greekchoices.admin.model;
import gr.greekchoices.CustomDateSerializer;
import org.hibernate.annotations.Type;
import org.joda.time.LocalDate;
import org.springframework.format.annotation.DateTimeFormat;
import com.fasterxml.jackson.databind.annotation.JsonSerialize;
public class Event {
        String title;
        @Type(type = "org.jadira.usertype.dateandtime.joda.PersistentLocalDate")
        @DateTimeFormat(pattern = "yyyy-MM-dd")
        private LocalDate start;
```

```
@Type(type = "org.jadira.usertype.dateandtime.joda.PersistentLocalDate")
@DateTimeFormat(pattern = "yyyy-MM-dd")
private LocalDate end;
String color;
String url;
public Event(String title, LocalDate start, LocalDate end, String color,
                        String url) {
        super();
        this.title = title;
        this.start = start;
        this.end = end;
        this.color = color;
        this.url = url;
}
public String getTitle() {
        return title;
}
public void setTitle(String title) {
        this.title = title;
}
@JsonSerialize(using = CustomDateSerializer.class)
public LocalDate getStart() {
        return start;
}
public void setStart(LocalDate start) {
        this.start = start;
}
@JsonSerialize(using = CustomDateSerializer.class)
public LocalDate getEnd() {
        return end;
}
public void setEnd(LocalDate end) {
        this.end = end;
}
public String getColor() {
        return color;
}
public void setColor(String color) {
        this.color = color;
}
public String getUrl() {
```

```
		return url;
	}
	public void setUrl(String url) {
		this.url = url;
	}
	@Override
	public int hashCode() {
		final int prime = 31;
		int result = 1;
		result = prime * result + ((color == null) ? 0 : color.hashCode());
		result = prime * result + ((end == null) ? 0 : end.hashCode());
		result = prime * result + ((start == null) ? 0 : start.hashCode());
		result = prime * result + ((title == null) ? 0 : title.hashCode());
		result = prime * result + ((url == null) ? 0 : url.hashCode());
		return result;
	}
	@Override
	public boolean equals(Object obj) {
		if (this == obj)
			return true;
		if (obj == null)
			return false;
		if (getClass() != obj.getClass())
			return false;
		Event other = (Event) obj;
		if (color == null) {
			if (other.color != null)
				return false;
		} else if (!color.equals(other.color))
			return false;
		if (end == null) {
			if (other.end != null)
				return false;
		} else if (!end.equals(other.end))
			return false;
		if (start == null) {
			if (other.start != null)
				return false;
		} else if (!start.equals(other.start))
			return false;
		if (title == null) {
```

```
                                        if (other.title != null)
                                                return false;
                        } else if (!title.equals(other.title))
                                        return false;
                        if (url == null) {
                                        if (other.url != null)
                                                return false;
                        } else if (!url.equals(other.url))
                                        return false;
                        return true;
        }
        @Override
        public String toString() {
                        return "Event [title=" + title + ", start=" + start + ", end=" + end
                                        + ", color=" + color + ", url=" + url + "]";
        }
}
```

9.10.5.3 CombinedHotelHotelBasicInfoCommand.java

```
package gr.greekchoices.admin.model;
import gr.greekchoices.admin.entity.Hotel;
import gr.greekchoices.admin.entity.HotelBasicInfo;
public class CombinedHotelHotelBasicInfoCommand {
        Hotel hotel;
        HotelBasicInfo hotelBasicInfo;
        public Hotel getHotel() {
                return hotel;
        }
        public void setHotel(Hotel hotel) {
                this.hotel = hotel;
        }
        public HotelBasicInfo getHotelBasicInfo() {
                return hotelBasicInfo;
        }
        public void setHotelBasicInfo(HotelBasicInfo hotelBasicInfo) {
                this.hotelBasicInfo = hotelBasicInfo;
        }
}
```

9.10.6 Other

Τέλος άλλες βοηθητικές κλάσεις ή διεπαφές όπως η Constants.java η οποία έχει διάφορες σταθερές όπως τα Date Formats παρουσιάζονται παρακάτω.

9.10.6.1 Constants.java

```
package gr.greekchoices;
mport java.text.SimpleDateFormat;
import java.util.Random;
public interface Constants {
		public static final Random RANDOM_GENERATOR = new Random(100);
		// PATTERNS
		public final static String PATTERN_DAY = "dd/MM/yyyy";
		public final static String PATTERN_EVENT_DAY = "yyyy-MM-dd";
		public final static String PATTERN_DAY_FOR_DB = "dd-MM-yyyy";
		public final static String PATTERN_DAYTIME = "dd/MM/yyyy HH:mm:ss";
		public final static String PATTERN_TIME = "HH:mm";
		// FORMAT
		public final static SimpleDateFormat FORMAT_TO_TIME = new SimpleDateFormat(PATTERN_TIME);
		public final static SimpleDateFormat FORMAT_TO_DAY_FOR_DB = new SimpleDateFormat(PATTERN_DAY_FOR_DB);
}
```

9.10.6.2 CustomDateSerializer.java

```
package gr.greekchoices;
import java.io.IOException;
import org.joda.time.DateTime;
import org.joda.time.LocalDate;
import org.joda.time.format.DateTimeFormat;
import org.joda.time.format.DateTimeFormatter;
import com.fasterxml.jackson.core.JsonGenerator;
import com.fasterxml.jackson.core.JsonProcessingException;
import com.fasterxml.jackson.databind.JsonSerializer;
import com.fasterxml.jackson.databind.SerializerProvider;
public class CustomDateSerializer extends JsonSerializer<LocalDate> implements Constants{
		private static DateTimeFormatter formatter = DateTimeFormat.forPattern(PATTERN_EVENT_DAY);
```

```
    @Override
    public void serialize(LocalDate value, JsonGenerator gen,
                    SerializerProvider arg2)
        throws IOException, JsonProcessingException {
        gen.writeString(formatter.print(value));
    }
}
```

9.10.6.3 Tools.java

```
package gr.greekchoices;
import gr.greekchoices.admin.entity.Hotel;
import gr.greekchoices.admin.entity.HotelHasRoomType;
import java.util.HashMap;
import java.util.List;
import java.util.Locale;
import java.util.StringTokenizer;
import javax.transaction.Transactional;
import org.apache.log4j.Logger;
import org.joda.time.DateTimeZone;
import org.joda.time.LocalDate;
import org.joda.time.LocalTime;
import org.joda.time.format.DateTimeFormat;
import org.joda.time.format.DateTimeFormatter;
public class Tools {
        private final static Logger logger = Logger.getLogger(Tools.class);
        private final static int HOURS = 1;
        private final static int MINUTES = 2;
        private final static int SECONDS = 3;
        private final static DateTimeFormatter formater = DateTimeFormat.forPattern("d/MM/YYYY");
        public static boolean containsLatinCharacters(String text) {
                boolean result = false;
                char[] chrArray = text.toCharArray();
                for (int i = 0; i < chrArray.length; i++) {
```

```
        if ((chrArray[i] >= 'A' && chrArray[i] <= 'Z') || (chrArray[i] >= 'a' && chrArray[i] <= 'z')) {
            result = true;
          break;
        }
      }
                        return result;
            }
            /**
             * The format must be HH:mm
             * @param timeAsString
             * @return
             */
            public static LocalTime getLocalTimeFromFormatedString(String timeAsString) {
                        return new LocalTime(transformTimeFormatToSecondsAmount(timeAsString) * 1000, DateTimeZone.UTC);
            }
            public static LocalDate getLocalDateFromString(String dayAsString) {
                        if (dayAsString == null) {
                                    throw new IllegalArgumentException("Start date is null!");
                        }
                        LocalDate date = new LocalDate(formater.withLocale(new
Locale("el")).withZone(DateTimeZone.UTC).parseMillis(dayAsString));
                        return date;
            }
            /**
             * Transforms a timeformat from type HH:MM:SS to the number of seconds
             * @param callTime
             * @return
             */
            public static int transformTimeFormatToSecondsAmount(String callTime) {
                        int sumOfSeconds = 0;
                        StringTokenizer st = new StringTokenizer(callTime, ":");
                        int position = 0;
                        while (st.hasMoreTokens()) {
                                    position++;
                                    String part = st.nextToken();
                                    switch(position) {
                                    case HOURS:
                                                sumOfSeconds += Integer.parseInt(part) * 3600;
                                                break;
                                    case MINUTES:
                                                sumOfSeconds += Integer.parseInt(part) * 60;
                                                break;
```

```
                        case SECONDS:
                                sumOfSeconds += Integer.parseInt(part);
                                break;
                        default:
                        }
                }
                return sumOfSeconds;
        }
        /**
         * Transforms a number of seconds to a string from type HH:MM:SS
         * @param sumOfSeconds
         * @return
         */
        public static String transformToTimeFormat(int sumOfSeconds) {
                int hours = sumOfSeconds/3600;
                int minutes = (sumOfSeconds%3600)/60;
                int seconds = (sumOfSeconds%60);
                return String.valueOf(zeroPad(hours, 2) + ":" + zeroPad(minutes, 2) + ":" + zeroPad(seconds, 2));
        }
        /**
         * Returns as String the given number and adds so many zerow before
         * the the number as width - number.length
         * @param number
         * @param width is the total length of the output number, not just the number of zero digits
         * @return
         */
        public static String zeroPad( int number, int width )
        {
                // width is the total length of the output number, not just the number of zero digits
                StringBuffer result = new StringBuffer("");
                for( int i = 0; i < width-Integer.toString(number).length(); i++ ) {
                        result.append( "0" );
                }
                result.append( Integer.toString(number) );
                return result.toString();
        }
        /**
         * Returns as String the given number and adds so many zerow before
         * the the number as width - number.length
         * @param String
         * @param width is the total length of the output number, not just the number of zero digits
```

```
	 * @return
	 */
	public static String zeroPad( String value, int width )
	{
		// width is the total length of the output number, not just the number of zero digits
		StringBuffer result = new StringBuffer("");
		for( int i = 0; i < width-value.length(); i++ ) {
			result.append( "0" );
		}
		result.append( value );
		return result.toString();
	}
	/**
	 * Rounds the double in 4 digis
	 * @param in
	 * @return
	 */
	public static double roundTo4(double in) {
		return Math.round(in*10000.0) / 10000.0;
	}
	public static long getMillisFromLocalTime(LocalTime localTime) {
		return ((localTime.getHourOfDay() * 3600)
				+ (localTime.getMinuteOfHour() * 60)
				+ localTime.getSecondOfMinute()) * 1000;
	}
	public static String getDriversFullName(String firstName, String lastName) {
		return firstName + ", " + lastName;
	}
}
```

9.10.7 View

Στους φακέλους "/src/main/webapp/WEB-INF/jsp" και "/src/main/webapp/WEB-INF/layout" βρίσκονται τα αρχεία ".jsp" που έχουν να κάνουν με την δημιουργεία του Response από τον Server πίσω στον client (browser) συνήθως σε html format και σε σχέση με το Design Pattern MVC, η συσχέτιση είναι με το View.

Τα αρχεία που βρίσκονται στο "/src/main/webapp/WEB-INF/layout" είναι εκείνα τα αρχεία που χρησιμοποιούνται από το Apache Tiles.

Τα αρχεία που βρίσκονται στο "/src/main/webapp/WEB-INF/jsp" είναι και τα βασικά αρχεία που έχουν να κάνουν με το View όπως προ-αναφέραμε τα οποία είναι συνδιασμός html, jsp , EL και JSTL, ενώ η μεταβλητές που χρησιμοποιουύνται έχουν ήδη οριστεί στον Controller. Παρακάτω παραθέτονται αυτά τα αρχεια ανά φάκελο.

9.10.7.1 index.jsp

```
<%@ include file="../layout/taglib.jsp" %>
<c:url var="logoutUrl" value="/logout" />
<div id="control_panel">
	<div class="row">
		<div class="col-xs-12">
			<h3>Welcome to Hootels Control Panel</h3>
			<h4>
				You can either <a href='<spring:url value="/admin/home.html"/>'>Login</a> or
				<a href='<spring:url value="/register.html"/>'>Register</a> a new account.
			</h4>
	 </div>
	</div>
</div>
```

9.10.7.2 login.jsp

```
<%@ include file="../layout/taglib.jsp" %>
		<link rel="stylesheet" href="../../resources/css/signin.css" type="text/css" />
		<c:url value="/login" var="loginAction"/>
		<form:form cssClass="form-signin" method="POST" action="${loginAction}">
    <h2 class="form-signin-heading">Please sign in</h2>
    <div class="form-group">
			<label for="username" class="sr-only">Username</label>
			<input type="text" name="username" id="username" class="form-control" placeholder="username" required
autofocus>
    </div>
    <div class="form-group">
			<label for="inputPassword" class="sr-only">Password</label>
			<input type="password" name="password" id="inputPassword" class="form-control" placeholder="Password" required>
				</div>
				<div class="form-group">
			<button class="btn btn-lg btn-primary btn-block" type="submit">Sign in</button>
    </div>
  </form:form>
```

9.10.7.3 /admin/home.jsp

```
<%@ include file="../../layout/taglib.jsp" %>
<c:set var="rolesStr" value="ROLE_USER" />
<c:forEach var="role" items="${user.roleList}">
 <c:if test="${role.name eq 'ROLE_ADMIN'}">
  <c:set var="rolesStr" value="${rolesStr} | ROLE_ADMIN" />
 </c:if>
 <c:if test="${role.name eq 'ROLE_MANAGER'}">
  <c:set var="rolesStr" value="${rolesStr} | ROLE_MANAGER" />
 </c:if>
</c:forEach>
<c:choose>
 <c:when test="${fn:contains(rolesStr,'ROLE_ADMIN')}">
			<%@ include file="home/adminRoleHome.jsp" %>
 </c:when>
 <c:when test="${fn:contains(rolesStr,'ROLE_MANAGER')}">
			<%@ include file="home/managerRoleHome.jsp" %>
 </c:when>
```

```
 <c:otherwise>
            <%@ include file="home/userRoleHome.jsp" %>
            </c:otherwise>
</c:choose>
```

9.10.7.4 /admin/home/adminRoleHome.jsp

```
<%@ include file="../../../layout/taglib.jsp" %>
<div id="control_panel" class="admin_control_panel">
            <div class="row">
                        <div class="col-xs-6">
                                    <div id="hotel_managment" class="panel panel-default">
                        <div class="panel-heading">
                                                <div class="pull-left panel_title_icons">
                                                            <span class="glyphicon glyphicon-user gc_greylight"></span>
                                                </div>
                                                <h3 class="panel-title">User Management</h3>
                                    </div>
                                    <div class="panel-body">
                                                <div class="row">
                                                            <div class="col-xs-6">
                                                                        <a href="<spring:url value="/admin/users.html"/>"
class="btn btn-default btn-lg centered" role="button">Users</a>
                                                            </div>
                                                </div>
                                    </div>
                                    </div>
                        </div>
                        <div class="col-xs-6">
                                    <div id="hotel_managment" class="panel panel-default">
                        <div class="panel-heading">
                                                <div class="pull-left panel_title_icons">
                                                            <span class="glyphicon glyphicon-oil"></span>
                                                </div>
                                                <h3 class="panel-title">Hotel Management</h3>
                                    </div>
```

```
                                        <div class="panel-body">
                                                <div class="row">
                                                        <div class="col-xs-6">
                                                                <a href="<spring:url value="/admin/hotelcreate.html"/>"
class="btn btn-default btn-lg centered" role="button">Hotel Properties</a>
                                                        </div>
                                                </div>
                                        </div>
                                </div>
                        </div>
                <div class="row">
                 <div class="col-xs-12">
                        <div id="general_tools" class="panel panel-default">
                                        <div class="panel-heading">
                                                <div class="pull-left panel_title_icons">
                                                        <span class="glyphicon glyphicon-wrench"></span>
                                                </div>
                                                <h3 class="panel-title">General Tools</h3>
                                        </div>
                                <div class="panel-body">
                                                <div class="row">
                                                        <div class="col-xs-12 pull-left">
                                                                <a href="#" class="btn btn-default btn-lg centered"
role="button">Resources</a>
                                                                <a href="#" class="btn btn-default btn-lg centered"
role="button">My Subscription</a>
                                                        </div>
                                                </div>
                                </div>
                        </div>
                 </div>
                </div>

</div>
```

9.10.7.5 /admin/home/managerRoleHome.jsp

```
<%@ include file="../../../layout/taglib.jsp" %>
<c:url var="hotelCreateurl" value="/admin/hotels.html"></c:url>
<div id="control_panel" class="manager_control_panel">
	<div class="row">
		<div class="col-xs-6">
			<div id="overview" class="panel panel-default">
				<div class="panel-heading">
					<div class="pull-left panel_title_icons">
						<span class="glyphicon glyphicon-blackboard"></span>
					</div>
					<h3 class="panel-title">Overview</h3>
				</div>
				<div class="panel-body">
					Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
					Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.
				</div>
			</div>
		</div>
		<div class="col-xs-6">
			<div id="hotel_managment" class="panel panel-default">
		<div class="panel-heading">
					<div class="pull-left panel_title_icons">
						<span class="glyphicon glyphicon-oil"></span>
					</div>
					<h3 class="panel-title">Hotel Management</h3>
				</div>
				<div class="panel-body">
					<div class="row">
						<div class="col-xs-6 text-left">
							<a href="<spring:url value="/admin/hotels.html"/>" class="btn btn-default btn-lg centered" role="button">My Hotels</a>
						</div>
					</div>
				</div>
			</div>
		</div>
	</div>
	<div class="row">
	 <div class="col-xs-12">
```

```
                    <div id="general_tools" class="panel panel-default">
                                <div class="panel-heading">
                                        <div class="pull-left panel_title_icons">
                                                <span class="glyphicon glyphicon-wrench"></span>
                                        </div>
                                        <h3 class="panel-title">General Tools</h3>
                                </div>
                            <div class="panel-body">
                                        <div class="row">
                                                <div class="col-xs-12 pull-left">
                                                        <a href="#" class="btn btn-default btn-lg centered"
role="button">Resources</a>
                                                        <a href="#" class="btn btn-default btn-lg centered"
role="button">My Subscription</a>
                                                </div>
                                        </div>
                            </div>
                    </div>
             </div>
            </div>
</div>
```

9.10.7.6 /admin/home/userRoleHome.jsp

```
<%@ include file="../../../layout/taglib.jsp" %>
<div id="control_panel" class="user_control_panel">
            <div class="row">
                    <div class="col-xs-6">
                            <div id="hotel_managment" class="panel panel-default">
                    <div class="panel-heading">
                                        <div class="pull-left panel_title_icons">
                                                <span class="glyphicon glyphicon-oil"></span>
                                        </div>
                                        <h3 class="panel-title">Booking Managment</h3>
                                </div>
                                <div class="panel-body">
                                        <div class="row">
                                                <div class="col-xs-12">
```

```
                                                    <a href="<spring:url value="/admin/hotels.html"/>"
class="btn btn-default btn-lg centered" role="button">My Bookings</a>
                                                </div>
                                            </div>
                                        </div>
                                    </div>
                                </div>
                                <div class="col-xs-6">
                                    <div id="hotel_managment" class="panel panel-default">
                                <div class="panel-heading">
                                            <div class="pull-left panel_title_icons">
                                                <span class="glyphicon glyphicon-oil"></span>
                                            </div>
                                            <h3 class="panel-title">Billing Managment</h3>
                                        </div>
                                        <div class="panel-body">
                                            <div class="row">
                                                <div class="col-xs-12">
                                                    <a href="<spring:url value="/admin/hotels.html"/>"
class="btn btn-default btn-lg centered" role="button">My Invoices</a>
                                                </div>
                                            </div>
                                        </div>
                                    </div>
                                </div>
                            </div>
</div>
```

9.10.7.7 /admin/hotels/hotelsListing.jsp

```
<%@ include file="../../../layout/taglib.jsp" %>
<div class="row">
	<div class="col-xs-12">
		<table class="table table-bordered table-hover table-striped">
			<thead>
				<tr>
					<th>Hotel Name</th>
					<th colspan="2" />
				</tr>
			</thead>
			<tbody>
				<c:forEach items="${hotels}" var="hotel">
					<tr>
						<td>
 <a href="<spring:url value="/admin/hotels/${hotel.id}/control_panel.html"/>">
 <c:out value="${hotel.name}" />
 </a>
 </td>
 <td align="center">
 <a target="_blank" href="<spring:url value="#"/>">Visit Hotel</a>
 </td>
 <td align="center">
 <a href="<spring:url value="/admin/hotels/${hotel.id}/control_panel.html"/>">
 Manage Hotel
 </a>
 </td>
 </tr>
 </c:forEach>
 </tbody>
 </table>
 </div>
</div>
<div class="row">
 <div class="col-xs-12">
 <div class="pull-right">
 <!-- Button trigger modal -->
 <button type="button" class="btn btn-primary btn-md" data-toggle="modal" data-target="#myModal">
```

```
                                    New Hotel
                                </button>
                                <form:form commandName="hotel" cssClass="form-horizontal">
                                        <!-- Modal -->
                                        <div class="modal fade" id="myModal" tabindex="-1" role="dialog" aria-
labelledby="myModalLabel" aria-hidden="true">
                                         <div class="modal-dialog">
                                          <div class="modal-content">
                                           <div class="modal-header">
                                            <button type="button" class="close" data-dismiss="modal" aria-label="Close"><span
aria-hidden="true">&times;</span></button>
                                            <h4 class="modal-title" id="myModalLabel">New Hotel</h4>
                                           </div>
                                           <div class="modal-body">
                                                                                        <div class="form-group">
                                                                                                <label for="name" class="col-sm-2
control-label">Name:</label>
                                                                                                <div class="col-sm-10">
                                                                                                        <form:input
path="name" cssClass="form-control"/>
                                                                                        </div>
                                                                                </div>
                                           </div>
                                           <div class="modal-footer">
                                            <button type="button" class="btn btn-default" data-dismiss="modal">Close</button>
                                          <input type="submit" class="btn btn-primary" value="Save">
                                           </div>
                                          </div>
                                         </div>
                                        </div>
                                </form:form>
                        </div>
                </div>
</div>
```

9.10.7.8 /admin/hotels/hotelControlPanel.jsp

```
<%@ include file="../../../layout/taglib.jsp" %>

<div id="information_page_header" class="page_header_wrapper">

	<div class="section_page_header">Informations</div>

	<div id="page_header_descr" class="row">

		<div class="col-xs-12 pull-left">

			<a href='<spring:url value="/admin/hotels/${hotel.id}/basic_info.html"/>' class="btn btn-default btn-lg centered" role="button">Basic Info</a>

		</div>

	</div>

</div>

<div id="information_page_header" class="page_header_wrapper">

	<div class="section_page_header">Manage</div>

	<div id="page_header_descr" class="row">

		<div class="col-xs-12 pull-left">

			<a href='<spring:url value="/admin/hotels/${hotel.id}/facilities/hotel-facilities.html"/>' class="btn btn-default btn-lg centered" role="button">Facilities</a>

			<a href='<spring:url value="/admin/hotels/${hotel.id}/room-types.html"/>' class="btn btn-default btn-lg centered" role="button">Room Types</a>

			<a href='<spring:url value="/admin/hotels/${hotel.id}/periods.html"/>' class="btn btn-default btn-lg centered" role="button">Periods</a>

			<a href='<spring:url value="/admin/hotels/${hotel.id}/prices/per-period/all.html"/>' class="btn btn-default btn-lg centered" role="button">Prices</a>

			<a href='<spring:url value="/admin/hotels/${hotel.id}/availabilities-calendar/all.html"/>' class="btn btn-default btn-lg centered" role="button">Availability</a>

			<a href='<spring:url value="/admin/hotels/${hotel.id}/remove.html"/>' class="btn btn-default btn-lg centered" role="button" id="btn_delete_hotel">Delete Hotel</a>

		</div>

	</div>

</div>

<div id="information_page_header" class="page_header_wrapper">

	<div class="section_page_header">Billing</div>

	<div id="page_header_descr" class="row">

		<div class="col-xs-12 pull-left">

			<a href='<spring:url value="/admin/hotels/${hotel.id}/booking/booking-listing.html"/>' class="btn btn-default btn-lg centered" role="button">Bookings</a>

		</div>

	</div>

</div>

<!-- Modal -->

<div class="modal fade" id="hotelRemove" tabindex="-1" role="dialog" aria-labelledby="myModalLabel" aria-hidden="true">

 <div class="modal-dialog">

  <div class="modal-content">

   <div class="modal-header">
```

```
      <button type="button" class="close" data-dismiss="modal" aria-label="Close"><span aria-
hidden="true">&times;</span></button>
      <h4 class="modal-title" id="myModalLabel">Delete Hotel</h4>
    </div>
    <div class="modal-body">
      Really delete hotel : <b><span class="hotel_placeholder"></span></b> ?
    </div>
    <div class="modal-footer">
      <button type="button" class="btn btn-default" data-dismiss="modal">Cancel</button>
      <a href="" class="btn btn-danger removeBtn">Delete</a>
    </div>
  </div>
 </div>
</div>
```

9.10.7.9/admin/hotels/availability/availabilitiesCalendarPerAllRoomType.jsp

```
<%@ include file="../../../../layout/taglib.jsp" %>
<div class="bordered_padded_wrapper">
            <div class="padded_wrapper">
            <div id="select_room_type" class="row" >
                        <div class="pull-right">
                                    <div class="col-xs-12">
                                                <div class="col-xs-3 col-xs-offset-5 text-right">
                                                            <div class="select_room_type_text">Select Room Type :</div>
                                                </div>
                                                <div class="col-xs-4">
                                                            <select class="form-control select_room_type">
                                                                        <option class="active">-Select-</option>
                                                                                    <c:forEach items="${allRoomTypePerHotel}"
var="iteratedRoomType">
                                                                                    <option
value="${iteratedRoomType.value.id}">${iteratedRoomType.value.name}</option>
                                                                                    </c:forEach>
                                                            </select>
                                                </div>
                                    </div>
                        </div>
```

```
                    </div>
                    <div class="section_page_header room_type_name">Availabilities Calendar</div>
                    <div class="row">
                        <div class="col-xs-12">
                            Please, select a room type to set it's availabilities!!
                        </div>
                    </div>
            </div>
</div>
```

9.10.7.10 /admin/hotels/availability/availabilitiesCalendarPerRoomType.jsp

```
<%@ include file="../../../../layout/taglib.jsp" %>
<div class="bordered_padded_wrapper">
        <div class="padded_wrapper">
                <div id="select_room_type" class="row" >
                    <div class="pull-right">
                        <div class="col-xs-12">
                                <div class="col-xs-3 col-xs-offset-5 text-right">
                                    <div class="select_room_type_text">Select Room Type
:</div>
                                </div>
                                <div class="col-xs-4">
                                    <select class="form-control select_room_type">
                                        <option value="all">-Select-</option>

                                        <c:forEach
items="${allRoomTypePerHotel}" var="iteratedRoomType">
                                        <option
${iteratedRoomType.value.id == roomType.id ? 'class="active" selected' : ''}
value="${iteratedRoomType.value.id}">${iteratedRoomType.value.name}</option>
                                        </c:forEach>
                                    </select>
                                </div>
                            </div>
                        </div>
                </div>
                <div class="section_page_header room_type_name">
                    Room Type : <span style="color:${roomType.color}" class="glyphicon glyphicon-stop"></span>
<span>${roomType.name}</span>
```

```
                    </div>
                    <div class="row">
                            <div class="col-xs-12">
                        <div id="wrapper">
                          <div id="backend-container">
                            <p>
                              Double click on a day or select 2 different days to edit them and view the changes in the Front End
Version by clicking refresh.
                            </p>
                            <input type="button" name="backend-refresh" id="backend-refresh" class="reload-btn" value="Refresh
Back End Calendar" />
                            <div id="backend"></div>
                          </div>
                        </div>
                            </div>
                    </div>
            </div>
</div>
```

9.10.7.11 /admin/hotels/basic-info/hotelBasicInfoEdit.jsp

```
<%@ include file="../../../../layout/taglib.jsp" %>
            <div class="row">
             <div class="col-xs-6 col-xs-offset-3">
                    <div id="hotel_basic_info" class="panel panel-default">
                            <div class="panel-body">
                                    <form:form commandName="combinedHotelHotelBasicInfoCommand" cssClass="form-
horizontal">
                                            <div id="general_info" class="row">
                                                    <div class="col-xs-12 ">
                                                            <div class="page-header section_header_cp">
                                                                    <h4>General Info Edit</h4>

                                                            </div>
                                                            <div class="info_per_line">
                                                             <div class="row">
                                                                    <div class="col-xs-3 text-right"><span>Hotel
Name :</span></div>
                                                                    <div class="col-xs-7 text-right"><form:input
path="hotel.name" cssClass="form-control"/></div>
```

```
                                                                    </div>
                                                                    <div class="row">
                                                                        <div class="col-xs-3 text-right"><span>Stars
:</span></div>
                                                                        <div class="col-xs-7 text-left">
                                                                            <form:select
path="hotelBasicInfo.star.id" cssClass="form-control"  >
                                                                                <c:forEach
items="${allStars}" var="iteratedStar">
                                                                                    <option
${iteratedStar.value.id == combinedHotelHotelBasicInfoCommand.hotelBasicInfo.star.id ? 'class="active" selected' : ''}
value="${iteratedStar.value.id}">${iteratedStar.value.name}</option>
                                                                                </c:forEach>
                                                                            </form:select>
                                                                        </div>
                                                                    </div>
                                                                </div>
                                                            </div>
                                                            <div id="contact_info" class="row">
                                                                <div class="col-xs-12 ">
                                                                    <div class="page-header section_header_cp">
                                                                        <h4>Contact Info Edit</h4>

                                                                    </div>
                                                                    <div class="info_per_line">


                                                                     <div class="row">
                                                                        <div class="col-xs-3 text-
right"><span>Address :</span></div>
                                                                        <div class="col-xs-7 text-left"><form:input
path="hotelBasicInfo.address" cssClass="form-control"/></div>
                                                                    </div>
                                                                    <div class="row">
                                                                        <div class="col-xs-3 text-right"><span>City
:</span></div>
                                                                        <div class="col-xs-7 text-left">
                                                                            <form:select
path="hotelBasicInfo.city.id" cssClass="form-control"  >
                                                                                <c:forEach
items="${allCities}" var="iteratedCity">
                                                                                    <option
${iteratedCity.value.id == combinedHotelHotelBasicInfoCommand.hotelBasicInfo.city.id ? 'class="active" selected' : ''}
value="${iteratedCity.value.id}">${iteratedCity.value.name}</option>
                                                                                </c:forEach>
                                                                            </form:select>
```

```
                                                                    </div>
                                                                </div>
                                                                <div class="row">
                                                                    <div class="col-xs-3 text-
right"><span>Telephone :</span></div>
                                                                    <div class="col-xs-7 text-left"><form:input
path="hotelBasicInfo.telephone" cssClass="form-control"/></div>
                                                                </div>
                                                                <div class="row">
                                                                    <div class="col-xs-3 text-right"><span>Mobile
:</span></div>
                                                                    <div class="col-xs-7 text-left"><form:input
path="hotelBasicInfo.mobile" cssClass="form-control"/></div>
                                                                </div>
                                                            </div>
                                                        </div>
                                                        <div class="row">
                                                            <div class="col-xs-12 text-center">
                                                                <a id="back_btn" href='<spring:url
value="/admin/hotels/${hotel.id}/basic_info.html"></spring:url>' class="btn btn-default btn-md centered" role="button">Cancel</a>
                                                                <form:hidden path="hotel.id" />
                                                                <form:hidden path="hotelBasicInfo.hotelId" />
                                                                <!-- Button trigger modal -->
                                                                <input type="submit" class="btn btn-default btn-md
centered" value="Save">
                                                            </div>
                                                        </div>
                                                    </form:form>
                                                </div>
                                            </div>
                                        </div>
                                    </div>
```

9.10.7.12 /admin/hotels/basic-info/hotelBasicInfoRead.jsp

```
<%@ include file="../../../../layout/taglib.jsp" %>
		<div class="row">
		 <div class="col-xs-6 col-xs-offset-3">
				<div id="hotel_basic_info" class="panel panel-default">
					<div class="panel-body">
							<div id="general_info" class="row">
								<div class="col-xs-12 ">
									<div class="page-header section_header_cp">
											<h4>General Info</h4>

									</div>
									<div class="info_per_line">


									 <div class="row">
											<div class="col-xs-3 text-right"><span>Hotel
Name :</span></div>
											<div class="col-xs-9 text-
left">${hotel.name}</div>
									 </div>
									 <div class="row">
											<div class="col-xs-3 text-right"><span>Stars
:</span></div>
											<div class="col-xs-9 text-
left">${hotel.hotelBasicInfo.star.name}</div>
									 </div>
									</div>
								</div>
							</div>
							<div id="contact_info" class="row">
								<div class="col-xs-12 ">
									<div class="page-header section_header_cp">
											<h4>Contact Info</h4>
									</div>
									<div class="info_per_line">
									 <div class="row">
											<div class="col-xs-3 text-
right"><span>Address :</span></div>
											<div class="col-xs-9 text-
left">${hotel.hotelBasicInfo.address}</div>
									 </div>
									 <div class="row">
											<div class="col-xs-3 text-right"><span>City
:</span></div>
```

```
                                                                    <div class="col-xs-9 text-
left">${hotel.hotelBasicInfo.city.name}</div>
                                                                </div>
                                                                <div class="row">
                                                                    <div class="col-xs-3 text-
right"><span>Telephone :</span></div>
                                                                    <div class="col-xs-9 text-
left">${hotel.hotelBasicInfo.telephone}</div>
                                                                </div>
                                                                <div class="row">
                                                                    <div class="col-xs-3 text-right"><span>Mobile
:</span></div>
                                                                    <div class="col-xs-9 text-
left">${hotel.hotelBasicInfo.mobile}</div>
                                                                </div>
                                                            </div>
                                                        </div>
                                                        <div class="row">
                                                            <div class="col-xs-12 text-center">
                                                                <a id="back_btn" href='<spring:url
value="/admin/hotels/${hotel.id}/control_panel.html"></spring:url>' class="btn btn-default btn-md centered"
role="button">Back</a>
                                                                <a id="back_btn" href='<spring:url
value="/admin/hotels/${hotel.id}/basic_info/edit.html"></spring:url>' class="btn btn-default btn-md centered"
role="button">Edit</a>
                                                            </div>
                                                        </div>
                                                    </div>
                                                </div>
                                            </div>
                                        </div>
</div>
```

9.10.7.13 /admin/hotels/booking /bookingCalendar.jsp

```
<%@ include file="../../../../layout/taglib.jsp" %>
<div class="bordered_padded_wrapper">
	<div class="padded_wrapper">
		<div class="row">
			<div class="col-xs-12 tab_nav">
				<ul class="nav nav-tabs pull-right">
				 <li role="presentation"><a href="booking-listing.html">Booking Listing</a></li>
				 <li role="presentation" class="active"><a href="#">Booking Calendar</a></li>
				</ul>
			</div>
		</div>
		<div class="section_page_header">Booking Calendar</div>
		<div class="row">
			<div class="col-xs-2">
				<c:forEach items="${allRoomTypeByHotelId}" var="roomType">
					<div class="col-xs-12 roomtype_colorfull_labels" style="background-color:${roomType.value.color}">${roomType.value.name}</div>
				</c:forEach>
			</div>
			<div class="col-xs-10">
				<div id='script-warning'>
					<code>include/calendar_engine/get-events.php</code> must be running.
				</div>
				<div id='loading'>loading...</div>
				<div id='calendar'></div>
			</div>
		</div>
	</div>
</div>
```

9.10.7.14 /admin/hotels/booking /bookingListing.jsp

```
<%@ include file="../../../../layout/taglib.jsp" %>
<div class="bordered_padded_wrapper">
	<div class="padded_wrapper">
		<div class="row">
			<div class="col-xs-12 tab_nav">
				<ul class="nav nav-tabs pull-right">
				 <li role="presentation" class="active"><a href="#">Booking Listing</a></li>
				 <li role="presentation"><a href="booking-calendar.html">Booking
Calendar</a></li>
				</ul>
			</div>
		</div>
		<div class="section_page_header">Booking Listing</div>
		<div class="row">
			<div class="col-xs-12">
				<table class="table table-bordered table-hover table-striped">
					<thead>
						<tr>
							<th class="col-xs-1">Book ID</th>
							<th class="col-xs-2 text-center">Booking Period</th>
							<th class="col-xs-2 text-center">Room Type</th>
							<th class="col-xs-1 text-center">Cost</th>
							<th class="col-xs-1 text-center">User</th>
							<th class="col-xs-1 text-center">Status</th>
							<th class="col-xs-2 text-center">Booked Date</th>
							<th class="col-xs-2 text-center"/>
						</tr>
					</thead>
					<tbody>
						<c:forEach items="${allBookingPerAllRoomTypePerHotel}"
var="booking">
							<tr>
								<td>
								 <span>${booking.id}</span>
								</td>
								<td>
								 <span><joda:format
value="${booking.startDate}" pattern="dd/MM/yyyy"/> - <joda:format value="${booking.endDate}" pattern="dd/MM/yyyy"/></span>
								</td>
								<td style="padding:0px;">
									<table>
```

```
                                                                                    <tr>
                                                                                        <td style="
height:100%; border-left: 3px solid ${booking.hotelHasRoomType.roomType.color}; float:left;">
            <div style="padding:8px; float:left;">
            <span>${booking.hotelHasRoomType.roomType.name}</span>
            </div>
                                                                                        </td>
                                                                                    </tr>
                                                                                </table>
                                                                            </td>
                                                                            <td>
                                                                             <span>&euro;${booking.totalCost}</span>
                                                                            </td>
                                                                            <td>
                                                                             <span>${booking.user.username}</span>
                                                                            </td>
                                                                            <td>

<span>${fn:toUpperCase(fn:substring(booking.status, 0, 1))}${fn:toLowerCase(fn:substring(booking.status, 1, -1))}</span>
                                                                            </td>
                                                                            <td>
                                                                             <span><joda:format
value="${booking.bookingDate}" pattern="dd/MM/yyyy"/></span>
                                                                            </td>
                                                                            <td align="center">
                                                                             <div class="col-xs-12">
                                                                              <a class="gc_grey" href="<spring:url
value="/hotels/${hotel.id}/control_panel.html"/>">
                                                                                    <span class="glyphicon glyphicon-
edit gc_blue"></span> Details
                                                                              </a>
                                                                             </div>
                                                                            </td>
                                                                        </tr>
                                                                    </c:forEach>
                                                                </tbody>
                                                            </table>
                                                        </div>
                                                    </div>
            </div>
</div>
```

9.10.7.15 /admin/hotels/facilities/hotelFacilities/hotelFacilitiesListing.jsp

```
<%@ include file="../../../../../layout/taglib.jsp" %>
<jsp:setProperty name="facility" property="type" value="hotel"/>
<div class="bordered_padded_wrapper">
	<div class="padded_wrapper">
		<div class="row">
			<div class="col-xs-12 tab_nav">
				<ul class="nav nav-tabs pull-right">
				 <li role="presentation" class="active"><a href="#">Hotel Facilities</a></li>
				 <li role="presentation"><a href="room-facilities.html">Room
Facilities</a></li>
				</ul>
			</div>
		</div>
		<div class="section_page_header">Hotel Facilities</div>
		<div class="row">
			<div class="col-xs-12">
				<table class="table table-bordered table-hover table-striped">
				<thead>
					<tr>
						<th class="col-xs-8">Hotel Facility</th>
						<th class="col-xs-1 text-center">Global</th>
						<th class="col-xs-3 text-center"/>
					</tr>
				</thead>
				<tbody>
					<c:forEach items="${allHotelFacilityByHotelId}" var="hotelFacility">
						<tr>
							<td>
							 <span class="facility"
id="${hotelFacility.id}"><c:out value="${hotelFacility.name}" /></span>
							</td>
							<td align="center">
								<c:if
test="${hotelFacility.global}"><span class="glyphicon glyphicon-ok gc_orange"></span></c:if>
							</td>
							<td align="center">
							 <div class="col-xs-6">
							 <c:if test="${!hotelFacility.global}">
								<a class="gc_grey"
href="<spring:url value="/admin/hotels/${hotel.id}/facilities/hotel-facility/${hotelFacility.id}/edit.html"/>">
<span class="glyphicon
glyphicon-edit gc_blue"></span> Edit
```

```
                                                                                    </a>
                                                                                </c:if>
                                                                                </div>
                                                                                <div class="col-xs-6">
                                                                                <a class="gc_grey triggerRemove"
href="<spring:url value="/admin/hotels/${hotel.id}/facilities/hotel-facility/${hotelFacility.id}/remove.html"/>">
                                                                                    <span class="glyphicon glyphicon-
remove gc_red"></span> Remove
                                                                                </a>
                                                                                </div>
                                                                            </td>
                                                                        </tr>
                                                                    </c:forEach>
                                                                </tbody>
                                                            </table>
                                                        </div>
                                                    </div>
                                                    <div class="row">
                                                        <div class="col-xs-12">
                                                            <c:if test="${!empty allGlobalHotelFacilityToAdd}">
                                                                <form:form commandName="facility" cssClass="form-inline">
                                                                    <div class="form-group">
                                                                        <form:label path="id" for="id">Pre-Configured Facilities :
</form:label>
                                                                        <form:select path="id" cssClass="form-control">
                                                                            <c:forEach
items="${allGlobalHotelFacilityToAdd}" var="globalHotelFacilityToAdd">
                                                                                <form:option
value="${globalHotelFacilityToAdd.id}">${globalHotelFacilityToAdd.name}</form:option>
                                                                            </c:forEach>
                                                                        </form:select>
                                                                        <form:hidden path="type" />
                                                                        <input type="hidden" name="global" value="true">
                                                                        <input type="submit" class="btn btn-default"
value="Add">
                                                                    </div>
                                                                </form:form>
                                                            </c:if>
                                                        </div>
                                                    </div>
                                                    <div class="row">
                                                        <div class="col-xs-12">
                                                            <div class="pull-right">
                                                                <!-- Button trigger modal -->
```

```
                                        <button type="button" class="btn btn-default btn-lg" data-toggle="modal"
data-target="#myModal">
                                         New Hotel Facility
                                        </button>
                                        <form:form commandName="facility" cssClass="form-horizontal">
                                                <!-- Modal -->
                                                <div class="modal fade" id="myModal" tabindex="-1" role="dialog"
aria-labelledby="myModalLabel" aria-hidden="true">
                                                 <div class="modal-dialog">
                                                  <div class="modal-content">
                                                   <div class="modal-header">
                                                    <button type="button" class="close" data-dismiss="modal" aria-
label="Close"><span aria-hidden="true">&times;</span></button>
                                                    <h4 class="modal-title" id="myModalLabel">New Hotel
Facility</h4>
                                                   </div>
                                                   <div class="modal-body">
                                                                                                <div class="form-
group">
                                                                                                    <label
for="name" class="col-sm-2 control-label">Name:</label>
                                                                                                    <div
class="col-sm-10">
                <form:input path="name" cssClass="form-control"/>
                                                                                            </div>
                                                                                        </div>
                                                   </div>
                                                   <div class="modal-footer">
                                                            <form:hidden path="type" />
                                                            <input type="hidden" name="global" value="false">
                                                    <button type="button" class="btn btn-default" data-
dismiss="modal">Close</button>
                                                  <input type="submit" class="btn btn-primary" value="Save">
                                                   </div>
                                                  </div>
                                                 </div>
                                                </div>
                                        </form:form>
                                </div>
                        </div>
                </div>
        </div>
</div>
        <!-- Modal -->
```

```
            <div class="modal fade" id="facilityRemove" tabindex="-1" role="dialog" aria-labelledby="myModalLabel" aria-
hidden="true">
              <div class="modal-dialog">
                <div class="modal-content">
                  <div class="modal-header">
                    <button type="button" class="close" data-dismiss="modal" aria-label="Close"><span aria-
hidden="true">&times;</span></button>
                    <h4 class="modal-title" id="myModalLabel">Remove Hotel Facility</h4>
                  </div>
                  <div class="modal-body">
                    Really remove hotel facility : <b><span class="facility_placeholder"></span></b> ?
                  </div>
                  <div class="modal-footer">
                    <button type="button" class="btn btn-default" data-dismiss="modal">Cancel</button>
                    <a href="" class="btn btn-danger removeBtn">Remove</a>
                  </div>
                </div>
              </div>
            </div>
```

9.10.7.16 /admin/hotels/facilities/hotelFacilities/hotelFacilityEdit.jsp

```
<%@ include file="../../../../../layout/taglib.jsp" %>
<jsp:setProperty name="facility" property="type" value="hotel"/>
<div class="bordered_padded_wrapper">
        <div class="padded_wrapper">
                <div class="row">
                        <div class="col-xs-12">
                                <div class="pull-left">
                                        <form:form commandName="facility" cssClass="form-horizontal">
                                                <div class="form-group">
                                                        <label for="name" class="col-sm-2 control-
label">Name:</label>
                                                        <div class="col-sm-10">
                                                                <form:input path="name" cssClass="form-
control"/>
                                                        </div>
```

```
                                                    </div>
                                                    <form:hidden path="id" />
                                                    <form:hidden path="type" />
                                                    <input type="hidden" name="global" value="false">
                                                    <input type="submit" class="btn btn-primary" value="Save">
                                            </form:form>
                                    </div>
                            </div>
                    </div>
            </div>
</div>
```

9.10.7.17 /admin/hotels/facilities/roomFacilities/roomFacilitiesListing.jsp

```
<%@ include file="../../../../../layout/taglib.jsp" %>
<jsp:setProperty name="facility" property="type" value="roomtype"/>
<div class="bordered_padded_wrapper">
            <div class="padded_wrapper">
                    <div class="row">
                            <div class="col-xs-12 tab_nav">
                                    <ul class="nav nav-tabs pull-right">
                                     <li role="presentation"><a href="hotel-facilities.html">Hotel
Facilities</a></li>
                                     <li role="presentation" class="active"><a href="#">Room Facilities</a></li>
                                    </ul>
                            </div>
                    </div>
                    <div class="col-xs-12">
                            <div class="pull-right">
                                    <a class="all_room_facilities" href='<spring:url
value="/admin/hotels/${hotel.id}/facilities/room-facilities/per-roomtype/all.html"/>'>Manage Room Facilites per Room Type</a>
                            </div>
                    </div>
                    <div class="section_page_header">Room Facilities</div>
                    <div class="row">
                            <div class="col-xs-12">
```

```
                                    <table class="table table-bordered table-hover table-striped">
                                        <thead>
                                            <tr>
                                                <th class="col-xs-8">Room Facility</th>
                                                <th class="col-xs-1 text-center">Global</th>
                                                <th class="col-xs-3 text-center"/>
                                            </tr>
                                        </thead>
                                        <tbody>
                                            <c:forEach items="${allRoomFacilityByHotelId}" var="roomFacility">
                                                <tr>
                                                    <td>
                                                     <span class="facility"
id="${roomFacility.id}"><c:out value="${roomFacility.name}" /></span>
                                                    </td>
                                                    <td align="center">
                                                        <c:if
test="${roomFacility.global}"><span class="glyphicon glyphicon-ok gc_orange"></span></c:if>
                                                    </td>
                                                    <td align="center">
                                                     <div class="col-xs-6">
                                                      <c:if test="${!roomFacility.global}">
                                                          <a class="gc_grey"
href="<spring:url value="/admin/hotels/${hotel.id}/facilities/room-facility/${roomFacility.id}/edit.html"/>">
                                                              <span class="glyphicon
glyphicon-edit gc_blue"></span> Edit
                                                          </a>
                                                      </c:if>

                                                     </div>
                                                     <div class="col-xs-6">
                                                     <a class="gc_grey triggerRemove"
href="<spring:url value="/admin/hotels/${hotel.id}/facilities/room-facility/${roomFacility.id}/remove.html"/>">
                                                         <span class="glyphicon glyphicon-
remove gc_red"></span> Delete
                                                     </a>
                                                     </div>
                                                    </td>
                                                </tr>
                                            </c:forEach>
                                        </tbody>
                                    </table>
                                </div>
                            </div>
```

```
                    <div class="row">
                        <div class="col-xs-12">
                            <c:if test="${!empty allGlobalRoomFacilityToAdd}">
                                <form:form commandName="facility" cssClass="form-inline">
                                    <div class="form-group">
                                        <form:label path="id" for="id">Pre-Configured Room
Facilities : </form:label>
                                        <form:select path="id" cssClass="form-control">
                                            <c:forEach
items="${allGlobalRoomFacilityToAdd}" var="globalRoomFacilityToAdd">
                                                <form:option
value="${globalRoomFacilityToAdd.id}">${globalRoomFacilityToAdd.name}</form:option>
                                            </c:forEach>
                                        </form:select>
                                        <form:hidden path="type" />
                                        <input type="hidden" name="global" value="true">
                                        <input type="submit" class="btn btn-default"
value="Add">
                                    </div>
                                </form:form>
                            </c:if>
                        </div>
                    </div>
                    <div class="row">
                        <div class="col-xs-12">
                            <div class="pull-right">
                                <!-- Button trigger modal -->
                                <button type="button" class="btn btn-default btn-lg" data-toggle="modal"
data-target="#myModal">
                                 New Room Facility
                                </button>
                                <form:form commandName="facility" cssClass="form-horizontal">
                                    <!-- Modal -->
                                    <div class="modal fade" id="myModal" tabindex="-1" role="dialog"
aria-labelledby="myModalLabel" aria-hidden="true">
                                      <div class="modal-dialog">
                                        <div class="modal-content">
                                          <div class="modal-header">
                                            <button type="button" class="close" data-dismiss="modal" aria-
label="Close"><span aria-hidden="true">&times;</span></button>
                                            <h4 class="modal-title" id="myModalLabel">New Room
Facility</h4>
                                          </div>
                                          <div class="modal-body">
```

```
<div class="form-
group">
<label
for="name" class="col-sm-2 control-label">Name:</label>
<div
class="col-sm-10">
<form:input path="name" cssClass="form-control"/>
</div>
</div>
</div>
<div class="modal-footer">
<form:hidden path="type" />
<input type="hidden" name="global" value="false">
<button type="button" class="btn btn-default" data-
dismiss="modal">Close</button>
<input type="submit" class="btn btn-primary" value="Save">
</div>
</div>
</div>
</div>
</form:form>
</div>
</div>
</div>
</div>
</div>
<!-- Modal -->
<div class="modal fade" id="facilityRemove" tabindex="-1" role="dialog" aria-labelledby="myModalLabel" aria-
hidden="true">
<div class="modal-dialog">
<div class="modal-content">
<div class="modal-header">
<button type="button" class="close" data-dismiss="modal" aria-label="Close"><span aria-
hidden="true">&times;</span></button>
<h4 class="modal-title" id="myModalLabel">Remove Room Facility</h4>
</div>
<div class="modal-body">
Really remove room facility : <b><span class="facility_placeholder"></span></b> ?
</div>
<div class="modal-footer">
<button type="button" class="btn btn-default" data-dismiss="modal">Cancel</button>
<a href="" class="btn btn-danger removeBtn">Remove</a>
</div>
```

```
                </div>
              </div>
            </div>
```

9.10.7.18 /admin/hotels/facilities/roomFacilities/roomFacilityEdit.jsp

```
<%@ include file="../../../../../layout/taglib.jsp" %>
<jsp:setProperty name="facility" property="type" value="roomtype"/>
<div class="bordered_padded_wrapper">
        <div class="padded_wrapper">
                <div class="row">
                        <div class="col-xs-12">
                                <div class="pull-left">
                                        <form:form commandName="facility" cssClass="form-horizontal">
                                                <div class="form-group">
                                                        <label for="name" class="col-sm-2 control-
label">Name:</label>
                                                        <div class="col-sm-10">
                                                                <form:input path="name" cssClass="form-
control"/>
                                                </div>
                                                </div>
                                                <form:hidden path="id" />
                                                <form:hidden path="type" />
                                                <input type="hidden" name="global" value="false">
                                                <input type="submit" class="btn btn-primary" value="Save">
                                        </form:form>
                                </div>
                        </div>
                </div>
        </div>
</div>
```

9.10.7.19 /admin/hotels/facilities/roomFacilities/perRoomType/roomFacilitiesPerRoomType.jsp

```
<%@ include file="../../../../../layout/taglib.jsp" %>
<jsp:setProperty name="facility" property="type" value="roomtype"/>
<div class="bordered_padded_wrapper">
	<div class="padded_wrapper">
		<div class="row">
			<div class="col-xs-12 tab_nav">
				<ul class="nav nav-tabs pull-right">
				 <li role="presentation"><a href="../../hotel-facilities.html">Hotel Facilities</a></li>
				 <li role="presentation" class="active"><a href="#">Room Facilities</a></li>
				</ul>
			</div>
		</div>
		<div class="col-xs-12">
			<div class="pull-right">
				<a class="all_room_facilities" href='<spring:url value="/admin/hotels/${hotel.id}/facilities/room-facilities.html"/>'>Manage All Room Facilities</a>
			</div>
		</div>
		<div class="section_page_header">Room Facilities per All Room Type</div>
		<div class="bordered_padded_wrapper">
			<div class="padded_wrapper">
				<div id="select_room_type" class="row" >
					<div class="pull-right">
						<div class="col-xs-12">
<div class="col-xs-3 col-xs-offset-5 text-right">
<div class="select_room_type_text">Select Room Type :</div>
</div>
<div class="col-xs-4">
<select class="form-control select_room_type">
<option value="all">All</option>
<c:forEach items="${allRoomTypePerHotel}" var="iteratedRoomType">
<option ${iteratedRoomType.id == roomType.id ? 'class="active" selected' : ''} value="${iteratedRoomType.id}">${iteratedRoomType.name}</option>
</c:forEach>
</select>
</div>
```

```
                                                    </div>
                                                </div>
                                            </div>
                                            <div class="section_page_header room_type_name">Room Type :  <span
style="color:${roomType.color}" class="glyphicon glyphicon-stop"></span> <span>${roomType.name}</span></div>
                                            <div class="bordered_padded_wrapper bottom_margin">
                                                <div class="padded_wrapper">
                                                    <div class="row">
                                                        <div class="col-xs-5">
                                                            <div style="height:20px;">All
Avaliable Room Facilities</div>
                                                            <select name="from"
id="undo_redo" class="form-control" size="13" multiple="multiple">
                                                                <c:forEach
items="${allAvailableRoomFacilityPerRoomTypePerHotel}" var="availableRoomFacility">
                                                                    <option
value="${availableRoomFacility.id}">${availableRoomFacility.name} ${availableRoomFacility.global=='true' ? "*": ""}</option>
                                                                </c:forEach>
                                                            </select>
                                                            <div
class="manage_all_room_facilities_twoside_selection_above">
                                                                <a
class="all_room_facilities" href='<spring:url value="/admin/hotels/${hotel.id}/facilities/room-facilities.html"/>'>Manage All Room
Facilites</a>
                                                            </div>
                                                        </div>
                                                        <div class="col-xs-2" style="padding-
top:30px;">
                                                            <button type="button"
id="undo_redo_undo" class="btn btn-primary btn-block">undo</button>
                                                            <button type="button"
id="undo_redo_rightAll" class="btn btn-default btn-block"><i class="glyphicon glyphicon-forward"></i></button>
                                                            <button type="button"
id="undo_redo_rightSelected" class="btn btn-default btn-block"><i class="glyphicon glyphicon-chevron-right"></i></button>
                                                            <button type="button"
id="undo_redo_leftSelected" class="btn btn-default btn-block"><i class="glyphicon glyphicon-chevron-left"></i></button>
                                                            <button type="button"
id="undo_redo_leftAll" class="btn btn-default btn-block"><i class="glyphicon glyphicon-backward"></i></button>
                                                            <button type="button"
id="undo_redo_redo" class="btn btn-warning btn-block">redo</button>
                                                        </div>
                                                        <div class="col-xs-5">
                                                            <form:form
modelAttribute="roomFacilities" cssClass="form-horizontal">
                                                                <div>
                                                                    <div class="col-xs-10"
style="height:25px;padding-left:0px;">Specific Room Type Facilities</div>
```

```
                                                                                <div class="col-xs-1"
style="height:25px;padding:0px;margin:0px;float:right;">
                                                                                    <button
type="submit" onclick="checkAllOptions();" class="glyphicon glyphicon-floppy-disk"></button>
                                                                                </div>
                                                                            </div>


                                                                            <select name="to"
id="undo_redo_to" class="form-control" size="13" multiple="multiple">
                                                                                <c:forEach
items="${allRoomFacilityPerRoomTypePerHotel}" var="roomFacility">
                                                                                    <option
value="${roomFacility.id}">${roomFacility.name} ${roomFacility.global=='true' ? "*": ""}</option>
                                                                                </c:forEach>

                                                                            </select>
                                                                            </form:form>
                                                                        </div>
                                                                    </div>
                                                                </div>
                                                            </div>
                                                        </div>
                                                    </div>
                                                </div>
                                            </div>
                                        </div>
                                    </div>
                                </div>
                            </div>
                        </div>
                    </div>
                </div>
                <script type="text/javascript">
                    $(document).ready(function() {
                        $('#undo_redo').multiselect();
                        $('#undo_redo2').multiselect();
                    });
                </script>
```

9.10.7.20 /admin/hotels/facilities/roomFacilities/perRoomType/roomFacilitiesPerRoomTypeListing.jsp

```
<%@ include file="../../../../../layout/taglib.jsp" %>
<jsp:setProperty name="facility" property="type" value="roomtype"/>
<div class="bordered_padded_wrapper">
	<div class="padded_wrapper">
		<div class="row">
			<div class="col-xs-12 tab_nav">
				<ul class="nav nav-tabs pull-right">
				 <li role="presentation"><a href="../../hotel-facilities.html">Hotel Facilities</a></li>
				 <li role="presentation" class="active"><a href="#">Room Facilities</a></li>
				</ul>
			</div>
		</div>
		<div class="col-xs-12">
			<div class="pull-right">
				<a class="all_room_facilities" href='<spring:url value="/admin/hotels/${hotel.id}/facilities/room-facilities.html"/>'>Manage All Room Facilities</a>
			</div>
		</div>
		<div class="section_page_header">Room Facilities per All Room Types</div>
		<div class="bordered_padded_wrapper">
			<div class="padded_wrapper">
				<div id="select_room_type" class="row" >
					<div class="pull-right">
						<div class="col-xs-12">
<div class="col-xs-3 col-xs-offset-5 text-right">
<div class="select_room_type_text">Select Room Type :</div>
</div>
<div class="col-xs-4">
<select class="form-control select_room_type">
<option class="active">All</option>
<c:forEach items="${allRoomTypePerHotel}" var="iteratedRoomType">
<option value="${iteratedRoomType.value.id}">${iteratedRoomType.value.name}</option>
</c:forEach>
</select>
</div>
```

```
                                                    </div>
                                                </div>
                                            </div>
                                            <script type="text/javascript">
                                                $(document).ready(function() {
                                            </script>
                        <c:forEach items="${allAvaliableRoomFacilityPerAllRoomTypePerHotel}" var="items">
                                                    <div class="section_page_header room_type_name">Room Type :
<span style="color:${allRoomTypePerHotel[items.key].color}" class="glyphicon glyphicon-stop"></span>
<span>${allRoomTypePerHotel[items.key].name}</span></div>
                                                    <div class="bordered_padded_wrapper bottom_margin">
                                                        <div class="padded_wrapper">
                                                            <div class="row">
                                                                <div class="col-xs-5">
                                                                    <div
style="height:25px;">All Available Room Facilities</div>
                                                                    <select
name="from" id="undo_redo${items.key}" class="form-control" size="13" multiple="multiple">
                                                                    <c:forEach
items="${items.value}" var="roomFacility">

            <option value="${roomFacility.id}">${roomFacility.name} ${roomFacility.global=='true' ? "*": ""}</option>

                                                                    </c:forEach>
                                                                    </select>
                                                                    <div
class="manage_all_room_facilities_twoside_selection_above">

            <a class="all_room_facilities" href='<spring:url value="/admin/hotels/${hotel.id}/facilities/room-facilities.html"/>'>Manage
All Room Facilities</a>
                                                                    </div>

                                                                </div>


                                                                <div class="col-xs-2"
style="padding-top:35px;">
                                                                    <button
type="button" id="undo_redo${items.key}_undo" class="btn btn-primary btn-block">undo</button>
                                                                    <button
type="button" id="undo_redo${items.key}_rightAll" class="btn btn-default btn-block"><i class="glyphicon glyphicon-
forward"></i></button>
                                                                    <button
type="button" id="undo_redo${items.key}_rightSelected" class="btn btn-default btn-block"><i class="glyphicon glyphicon-chevron-
right"></i></button>
                                                                    <button
type="button" id="undo_redo${items.key}_leftSelected" class="btn btn-default btn-block"><i class="glyphicon glyphicon-chevron-
left"></i></button>
                                                                    <button
type="button" id="undo_redo${items.key}_leftAll" class="btn btn-default btn-block"><i class="glyphicon glyphicon-
backward"></i></button>
```

```
<button type="button" id="undo_redo${items.key}_redo" class="btn btn-warning btn-block">redo</button>
</div>
<div class="col-xs-5">
<form:form modelAttribute="roomFacilities" cssClass="form-horizontal">
<div>
	<div class="col-xs-10" style="height:25px;padding-left:0px;">Specific Room Type Facilities</div>
	<div class="col-xs-1" style="height:25px;padding:0px;margin:0px;float:right;">
	<button type="submit" onclick="checkAllOptions(${items.key});" class="glyphicon glyphicon-floppy-disk"></button>
	</div>
</div>
<input type="hidden" name="roomTypeId" value="${items.key}">
<select name="to" id="undo_redo${items.key}_to" class="form-control" size="13" multiple="multiple">
<c:forEach items="${allRoomFacilityPerAllRoomTypePerHotel[items.key]}" var="roomFacility">
	<option value="${roomFacility.id}">${roomFacility.name} ${roomFacility.global=='true' ? "*": ""}</option>
</c:forEach>
</select>
	</form:form>
</div>
</div>
</div>
</div>
<script type="text/javascript">
  $('#undo_redo${items.key}').multiselect();
</script>
</c:forEach>
<script type="text/javascript">
});
</script>
</div>
</div>
</div>
</div>
```

9.10.7.21 /admin/hotels/periods/periodEdit.jsp

```
<%@ include file="../../../../layout/taglib.jsp" %>
	<div class="padded_wrapper">
		<div class="row">
			<div class="col-xs-12">
				<div class="pull-right">
					<form:form commandName="period" cssClass="form-horizontal">
						<div class="form-group">
							<label for="startDate" class="col-sm-4 control-
label">Start Period Date :</label>
		<div class="col-sm-4">
			<div class='input-group date' id='startPeriodDate'>
				<form:input path="startDate" cssClass="form-control"/>
			  <span class="input-group-addon">
			    <span class="glyphicon glyphicon-calendar"></span>
			  </span>
			</div>
		</div>
						</div>
						<div class="form-group">
							<label for="endDate" class="col-sm-4 control-
label">Finish Period Date:</label>
		<div class="col-sm-4">
			<div class='input-group date' id='finishPeriodDate'>
				<form:input path="endDate" cssClass="form-control"/>
			  <span class="input-group-addon">
			    <span class="glyphicon glyphicon-calendar"></span>
			  </span>
			</div>
		</div>
						</div>
<script type="text/javascript">
  $(function () {
		$('#startPeriodDate').datetimepicker({
			format: 'DD/MM/YYYY'
		  });
		$('#finishPeriodDate').datetimepicker({
			format: 'DD/MM/YYYY'
			 });
	$("#startPeriodDate").on("dp.change", function (e) {
$('#finishPeriodDate').data("DateTimePicker").minDate(e.date);
	});
```

```
                                                $("#finishPeriodDate").on("dp.change", function (e) {

$('#startPeriodDate').data("DateTimePicker").maxDate(e.date);
                                                });
                                            });
                                        </script>
                                        <form:hidden path="id" />

                                        <input type="submit" class="btn btn-primary" value="Save">
                                    </form:form>
                                </div>
                            </div>
                        </div>
                    </div>
```

9.10.7.22 /admin/hotels/periods/periodsListing.jsp

```
<%@ include file="../../../../layout/taglib.jsp" %>
    <div class="padded_wrapper">
        <div class="row">
            <div class="col-xs-12">
                <table class="table table-bordered table-hover table-striped">
                    <thead>
                        <tr>
                            <th>Period</th>
                            <th></th>
                        </tr>
                        <c:forEach items="${allPeriodPerHotel}" var="period">
                            <tr>
                                <td><span class="period"
id="${period.value.id}"><joda:format value="${period.value.startDate}" pattern="dd/MM/yyyy"/> - <joda:format
value="${period.value.endDate}" pattern="dd/MM/yyyy"/></span></td>
                                <td align="center">
                                 <div class="col-xs-4">
                                  <a class="gc_grey" href="<spring:url
value="/admin/hotels/${hotel.id}/period/${period.value.id}/edit.html"/>">
```

```
                                                                                    <span class="glyphicon glyphicon-
edit gc_blue"></span> Edit
                                                                                </a>
                                                                                </div>
                                                                                <div class="col-xs-4">
                                                                                <a class="gc_grey triggerRemove"
href="<spring:url value="/admin/hotels/${hotel.id}/period/${period.value.id}/remove.html"/>">
                                                                                    <span class="glyphicon glyphicon-
remove gc_red"></span> Remove
                                                                                </a>
                                                                                </div>
                                                                                <div class="col-xs-4">
                                                                                <a class="gc_grey" href="<spring:url
value="/admin/hotels/${hotel.id}/prices/per-period/${period.value.id}.html"/>">
                                                                                    <span class="glyphicon glyphicon-
euro gc_orange"></span> Pricing
                                                                                </a>
                                                                                </div>
                                                                                </td>
                                                                            </tr>
                                                                        </c:forEach>
                                                                    </thead>
                                                                </table>
                                                            </div>
                                                        </div>
                                                        <div class="row">
                                                            <div class="col-xs-12">
                                                                <div class="pull-right">
                                                                    <!-- Button trigger modal -->
                                                                    <button type="button" class="btn btn-default btn-lg" data-toggle="modal"
data-target="#myModal">
                                                                     New Period
                                                                    </button>
                                                                    <form:form commandName="period" cssClass="form-horizontal">
                                                                        <!-- Modal -->
                                                                        <div class="modal fade" id="myModal" tabindex="-1" role="dialog"
aria-labelledby="myModalLabel" aria-hidden="true">
                                                                          <div class="modal-dialog">
                                                                            <div class="modal-content">
                                                                              <div class="modal-header">
                                                                                <button type="button" class="close" data-dismiss="modal" aria-
label="Close"><span aria-hidden="true">&times;</span></button>
                                                                                <h4 class="modal-title" id="myModalLabel">New Period</h4>
                                                                              </div>
                                                                              <div class="modal-body">
                                                                                                                                    <div class="form-
group">
```

```
                                                                                                    <label
for="startDate" class="col-sm-4 control-label">Start Period Date :</label>
                                                        <div class="col-sm-4">
                                                                <div class='input-group date' id='startPeriodDate'>
                                                                        <form:input path="startDate"
cssClass="form-control"/>
                                                                  <span class="input-group-addon">
                                                                    <span class="glyphicon glyphicon-calendar"></span>
                                                                  </span>
                                                                </div>
                                                        </div>
                                                                                                        </div>
                                                                                                        <div class="form-
group">
                                                                                                    <label
for="endDate" class="col-sm-4 control-label">Finish Period Date:</label>
                                                        <div class="col-sm-4">
                                                                <div class='input-group date' id='finishPeriodDate'>
                                                                        <form:input path="endDate" cssClass="form-
control"/>
                                                                  <span class="input-group-addon">
                                                                    <span class="glyphicon glyphicon-calendar"></span>
                                                                  </span>
                                                                </div>
                                                        </div>
                                                                                                        </div>
                                                                </div>
                                                                                                <script type="text/javascript">
                                                                                                  $(function () {

        $('#startPeriodDate').datetimepicker({
        format: 'DD/MM/YYYY'
        $('#finishPeriodDate').datetimepicker({
        format: 'DD/MM/YYYY'
});
$("#startPeriodDate").on("dp.change", function (e) {
$('#finishPeriodDate').data("DateTimePicker").minDate(e.date);
$("#finishPeriodDate").on("dp.change", function (e) {
$('#startPeriodDate').data("DateTimePicker").maxDate(e.date);
                                                                                                      });
                                                                                                    });
                                                                                                </script>
                                                                <div class="modal-footer">
                                                                  <button type="button" class="btn btn-default" data-
dismiss="modal">Close</button>
```

```
                                                    <input type="submit" class="btn btn-primary" value="Save">
                                                    </div>
                                                   </div>
                                                  </div>
                                                 </div>
                                            </form:form>
                                        </div>
                                    </div>
                                </div>
                            </div>
                            <!-- Modal -->
                            <div class="modal fade" id="periodRemove" tabindex="-1" role="dialog" aria-labelledby="myModalLabel" aria-hidden="true">
                             <div class="modal-dialog">
                              <div class="modal-content">
                               <div class="modal-header">
                                <button type="button" class="close" data-dismiss="modal" aria-label="Close"><span aria-hidden="true">&times;</span></button>
                                <h4 class="modal-title" id="myModalLabel">Remove Period</h4>
                               </div>
                               <div class="modal-body">
                                Really delete period : <b><span class="period_placeholder"></span></b> ?
                               </div>
                               <div class="modal-footer">
                                <button type="button" class="btn btn-default" data-dismiss="modal">Cancel</button>
                                <a href="" class="btn btn-danger removeBtn">Remove</a>
                               </div>
                              </div>
                             </div>
                            </div>
```

9.10.7.23 /admin/hotels/prices/priceEdit.jsp

```
<%@ include file="../../../../layout/taglib.jsp" %>
            <div class="padded_wrapper">
```

```
                <div class="row">
                        <div class="col-xs-12">
                                <div class="pull-right">
                                        <form:form commandName="price" cssClass="form-horizontal">
                                                <div class="form-group">
                                                        <label for="cost" class="col-sm-3 control-
label">Price:</label>
                                                        <div class="col-sm-9">
                                                                <form:input path="cost" cssClass="form-
control"/>
                                                </div>
                                                </div>
                                                <input type="hidden" name="referrer" value="${referrer}">
                                                <input type="submit" class="btn btn-primary" value="Save">
                                        </form:form>
                                </div>
                        </div>
                </div>
        </div>
```

9.10.7.24 /admin/hotels/prices/perPeriod/pricePerAllRoomTypePerAllPeriodListing.jsp

```
<%@ include file="../../../../layout/taglib.jsp" %>
        <%@ include file="../../../../../layout/taglib.jsp" %>
<div class="bordered_padded_wrapper">
        <div class="padded_wrapper">
                <div class="row">
                        <div class="col-xs-12 tab_nav">
                                        <ul class="nav nav-tabs pull-right">
                                         <li role="presentation" class="active"><a href="#">Show Prices per
Period</a></li>
                                         <li role="presentation"><a href="../per-roomtype/all.html">Show Prices per
Room type</a></li>
                                        </ul>
                        </div>
                </div>
                <div id="select_room_type" class="row" >
                        <div class="pull-right">
```

```
                                        <div class="col-xs-12">
                                        <div class="col-xs-3 col-xs-offset-5 text-right">
                                                        <div class="select_room_type_text">Select Period
:</div>
                                                </div>
                                                <div class="col-xs-4">
                                                        <select class="form-control select_room_type">
                                                                <option class="active">All</option>
                                                                        <c:forEach
items="${allPeriodPerHotel}" var="iteratedPeriod">
                                                                        <option
value="${iteratedPeriod.value.id}">
                                                                                <joda:format
value="${iteratedPeriod.value.startDate}" pattern="dd/MM/yyyy"/> - <joda:format value="${iteratedPeriod.value.endDate}"
pattern="dd/MM/yyyy"/>
                                                                        </option>
                                                                </c:forEach>
                                                        </select>
                                                </div>
                                        </div>
                                        </div>
                        </div>
                        <c:forEach items="${allPricePerAllRoomTypePerAllPeriodPerHotel}" var="entryOfPeriod">
                                                <div class="section_page_header room_type_name">
                                                        Period : <span><joda:format
value="${allPeriodPerHotel[entryOfPeriod.key].startDate}" pattern="dd/MM/yyyy"/> - <joda:format
value="${allPeriodPerHotel[entryOfPeriod.key].endDate}" pattern="dd/MM/yyyy"/></span>
                                                </div>
                                                <div class="bordered_padded_wrapper bottom_margin">
                                                        <div class="padded_wrapper">
                                                                <div class="row">
                                                                        <div class="col-xs-12">
                                                                                <table
class="table table-bordered table-hover table-striped">
            <thead>
            <tr>
                        <th class="col-xs-8">Room Type</th>
                        <th class="col-xs-1 text-center">Price</th>
                        <th class="col-xs-3 text-center"/>
            </tr>
            </thead>
            <tbody>
            <c:forEach items="${entryOfPeriod.value}" var="entryOfRoomType">
                        <tr>
                                    <td>
```

```
                                    <span style="color:${allRoomTypePerHotel[entryOfRoomType.key].color}" class="glyphicon
glyphicon-stop"></span> <span>${allRoomTypePerHotel[entryOfRoomType.key].name}</span>
                                </td>
                                <td align="center">
                                        ${entryOfRoomType.value.cost > 0 ? "&euro;" : ''}${entryOfRoomType.value.cost}
                                </td>
                                <td align="center">
                                 <div class="col-xs-12">
                                 <a class="gc_grey" href="<spring:url
value="/admin/hotels/${hotel.id}/price/${entryOfPeriod.key}/${entryOfRoomType.key}/edit.html"/>">
                                        Change Price
                                 </a>
                                 </div>
                                </td>
                        </tr>
                </c:forEach>
                </tbody>
                                                                                                        </table>
                                                                                                </div>
                                                                                                </div>
                                                                                        </div>
                                                                                </div>
                                </c:forEach>
                </div>
</div>
```

9.10.7.25 /admin/hotels/prices/perPeriod/pricePerAllRoomTypePerPeriod.jsp

```
<%@ include file="../../../../../layout/taglib.jsp" %>
<div class="bordered_padded_wrapper">
	<div class="padded_wrapper">
		<div class="row">
			<div class="col-xs-12 tab_nav">
				<ul class="nav nav-tabs pull-right">
				 <li role="presentation" class="active"><a href="#">Show Prices per Period</a></li>
				 <li role="presentation"><a href="../per-roomtype/all.html">Show Prices per Room type</a></li>
				</ul>
			</div>
		</div>
			<div id="select_room_type" class="row" >
				<div class="pull-right">
					<div class="col-xs-12">
						<div class="col-xs-3 col-xs-offset-5 text-right">
							<div class="select_room_type_text">Select Period :</div>
						</div>
						<div class="col-xs-4">
							<select class="form-control select_room_type">
								<option value="all">All</option>
								<c:forEach items="${allPeriodPerHotel}" var="iteratedPeriod">
									<option ${iteratedPeriod.value.id == period.id ? 'class="active" selected' : ''} value="${iteratedPeriod.value.id}">
	<joda:format value="${iteratedPeriod.value.startDate}" pattern="dd/MM/yyyy"/> - <joda:format value="${iteratedPeriod.value.endDate}" pattern="dd/MM/yyyy"/>
									</option>
								</c:forEach>
							</select>
						</div>

					</div>

				</div>
			</div>
			<div class="section_page_header room_type_name">Period : <span><joda:format value="${period.startDate}" pattern="dd/MM/yyyy"/> - <joda:format value="${period.endDate}" pattern="dd/MM/yyyy"/></div></span>
```

```
                                        <div class="bordered_padded_wrapper bottom_margin">
                                            <div class="padded_wrapper">
                                                <div class="row">
                                                    <div class="col-xs-12">
                                                        <table class="table table-bordered
table-hover table-striped">
                                                            <thead>
                                                                <tr>

            <th class="col-xs-8">Room Type</th>
            <th class="col-xs-1 text-center">Price</th>
            <th class="col-xs-3 text-center"/>
                                                                </tr>
                                                            </thead>
                                                            <tbody>
                                                                <c:forEach
items="${allPricePerAllRoomTypePerPeriodPerHotel}" var="entryOfRoomType">
            <tr>
            <td>
             <span style="color:${allRoomTypePerHotel[entryOfRoomType.key].color}" class="glyphicon glyphicon-stop"></span>
<span>${allRoomTypePerHotel[entryOfRoomType.key].name}</span>
            </td>
            <td align="center">
                    ${entryOfRoomType.value.cost > 0 ? "&amp;euro;" : ''}${entryOfRoomType.value.cost}
            </td>
            <td align="center">
             <div class="col-xs-6">
             <a class="gc_grey" href="<spring:url
value="/admin/hotels/${hotel.id}/price/${period.id}/${entryOfRoomType.key}/edit.html"/>">
              Change Price
             </a>
             </div>
            </td>
            </tr>
                                                                </c:forEach>
                                                            </tbody>
                                                        </table>
                                                    </div>
                                                </div>
                                            </div>
                                        </div>
            </div>
</div>
```

9.10.7.26 /admin/hotels/prices/perRoomType/pricePerAllPeriodPerAllRoomTypeListing.jsp

```
<%@ include file="../../../../../layout/taglib.jsp" %>
<div class="bordered_padded_wrapper">
	<div class="padded_wrapper">
		<div class="row">
			<div class="col-xs-12 tab_nav">
				<ul class="nav nav-tabs pull-right">
				 <li role="presentation"><a href="../per-period/all.html">Show Prices per Period</a></li>
				 <li role="presentation" class="active"><a href="#">Show Prices per Room type</a></li>
				</ul>
			</div>
		</div>
		<div id="select_room_type" class="row" >
			<div class="pull-right">
				<div class="col-xs-12">
					<div class="col-xs-3 col-xs-offset-5 text-right">
						<div class="select_room_type_text">Select Room Type :</div>
					</div>
					<div class="col-xs-4">
						<select class="form-control select_room_type">
							<option class="active">All</option>
							<c:forEach items="${allRoomTypePerHotel}" var="iteratedRoomType">
							<option value="${iteratedRoomType.value.id}">${iteratedRoomType.value.name}</option>
							</c:forEach>
						</select>
					</div>
				</div>
			</div>
		</div>
		<c:forEach items="${allPricePerAllPeriodPerAllRoomTypePerHotel}" var="entryOfRoomType">
			<div class="section_page_header room_type_name">
				Room Type : <span style="color:${allRoomTypePerHotel[entryOfRoomType.key].color}" class="glyphicon glyphicon-stop"></span>
<span>${allRoomTypePerHotel[entryOfRoomType.key].name}</span>
			</div>
			<div class="bordered_padded_wrapper bottom_margin">
				<div class="padded_wrapper">
```

```
                                                                                        <div class="row">
                                                                                                <div class="col-xs-12">
                                                                                                        <table
class="table table-bordered table-hover table-striped">
        <thead>
        <tr>
                <th class="col-xs-8">Period</th>
                <th class="col-xs-1 text-center">Price</th>
                <th class="col-xs-3 text-center"/>
        </tr>
        </thead>
        <tbody>

        <c:forEach items="${entryOfRoomType.value}" var="entryOfPeriod">
                <tr>
                        <td>
                         <span><joda:format value="${allPeriodPerHotel[entryOfPeriod.key].startDate}"
pattern="dd/MM/yyyy"/> - <joda:format value="${allPeriodPerHotel[entryOfPeriod.key].endDate}" pattern="dd/MM/yyyy"/></span>
                        </td>
                        <td align="center">
                                ${entryOfPeriod.value.cost > 0 ? "&euro;" : ''}${entryOfPeriod.value.cost}
                        </td>
                        <td align="center">
                         <div class="col-xs-12">
                         <a class="gc_grey" href="<spring:url
value="/admin/hotels/${hotel.id}/price/${entryOfPeriod.key}/${entryOfRoomType.key}/edit.html"/>">
                                Change Price
                         </a>
                         </div>
                        </td>
                </tr>
        </c:forEach>
        </tbody>
                                                                                                        </table>
                                                                                                </div>
                                                                                                </div>
                                                                        </div>
                                                                </div>
                        </c:forEach>
        </div>
</div>
```

9.10.7.27 /admin/hotels/prices/perRoomType/pricePerAllPeriodPerRoomType.jsp

```
<%@ include file="../../../../../layout/taglib.jsp" %>
<div class="bordered_padded_wrapper">
	<div class="padded_wrapper">
		<div class="row">
			<div class="col-xs-12 tab_nav">
				<ul class="nav nav-tabs pull-right">
				 <li role="presentation"><a href="../per-period/all.html">Show Prices per Period</a></li>
				 <li role="presentation" class="active"><a href="#">Show Prices per Room type</a></li>
				</ul>
			</div>
		</div>
			<div id="select_room_type" class="row" >
				<div class="pull-right">
					<div class="col-xs-12">
						<div class="col-xs-3 col-xs-offset-5 text-right">
							<div class="select_room_type_text">Select Room Type :</div>
						</div>
						<div class="col-xs-4">
							<select class="form-control select_room_type">
								<option value="all">All</option>
								<c:forEach items="${allRoomTypePerHotel}" var="iteratedRoomType">
									<option ${iteratedRoomType.value.id == roomType.id ? 'class="active" selected' : ''} value="${iteratedRoomType.value.id}">${iteratedRoomType.value.name}</option>
								</c:forEach>
							</select>
						</div>
					</div>
				</div>
			</div>
			<div class="section_page_header room_type_name">Room Type :  <span style="color:${roomType.color}" class="glyphicon glyphicon-stop"></span> <span>${roomType.name}</span></div>
			<div class="bordered_padded_wrapper bottom_margin">
				<div class="padded_wrapper">
					<div class="row">
						<div class="col-xs-12">
							<table class="table table-bordered table-hover table-striped">
```

```
                                                                                    <thead>
                                                                                        <tr>
            <th class="col-xs-8">Period</th>
            <th class="col-xs-1 text-center">Price</th>
            <th class="col-xs-3 text-center"/>
                                                                                        </tr>
                                                                                    </thead>
                                                                                    <tbody>
                                                                                        <c:forEach
items="${allPricePerAllPeriodPerRoomTypePerHotel}" var="entryOfPeriod">
            <tr>
            <td>
              <span><joda:format value="${allPeriodPerHotel[entryOfPeriod.key].startDate}" pattern="dd/MM/yyyy"/> - <joda:format
value="${allPeriodPerHotel[entryOfPeriod.key].endDate}" pattern="dd/MM/yyyy"/></span>
            </td>
            <td align="center">
                    ${entryOfPeriod.value.cost > 0 ? "&euro;" : ''}${entryOfPeriod.value.cost}
            </td>
            <td align="center">
             <div class="col-xs-6">
                    <a class="gc_grey" href="<spring:url
value="/admin/hotels/${hotel.id}/price/${entryOfPeriod.key}/${roomType.id}/edit.html"/>">
                        Change Price
                    </a>
             </div>
            </td>
            </tr>
                                                                                        </c:forEach>
                                                                                    </tbody>
                                                                                </table>
                                                                            </div>
<%--                                                                        <div class="col-xs-5">
                                                                                <div style="height:20px;">All
Avaliable Room Facilities</div>
                                                                                <select name="from"
id="undo_redo" class="form-control" size="13" multiple="multiple">
                                                                                    <c:forEach
items="${allAvailableRoomFacilityPerRoomTypePerHotel}" var="availableRoomFacility">
                                                                                        <option
value="${availableRoomFacility.id}">${availableRoomFacility.name} ${availableRoomFacility.global=='true' ? "*": ""}</option>
                                                                                    </c:forEach>
                                                                                </select>
                                                                                <div
class="manage_all_room_facilities_twoside_selection_above">
```

```
                                                                                <a
class="all_room_facilities" href='<spring:url value="/admin/hotels/${hotel.id}/facilities/room-facilities.html"/>'>Manage All Room
Facilites</a>
                                                                        </div>
                                                                </div>


                                                                <div class="col-xs-2" style="padding-
top:30px;">
                                                                        <button type="button"
id="undo_redo_undo" class="btn btn-primary btn-block">undo</button>
                                                                        <button type="button"
id="undo_redo_rightAll" class="btn btn-default btn-block"><i class="glyphicon glyphicon-forward"></i></button>
                                                                        <button type="button"
id="undo_redo_rightSelected" class="btn btn-default btn-block"><i class="glyphicon glyphicon-chevron-right"></i></button>
                                                                        <button type="button"
id="undo_redo_leftSelected" class="btn btn-default btn-block"><i class="glyphicon glyphicon-chevron-left"></i></button>
                                                                        <button type="button"
id="undo_redo_leftAll" class="btn btn-default btn-block"><i class="glyphicon glyphicon-backward"></i></button>
                                                                        <button type="button"
id="undo_redo_redo" class="btn btn-warning btn-block">redo</button>
                                                                </div>
                                                                <div class="col-xs-5">
                                                                        <div style="height:20px;">Specific
Room Type Facilities</div>
                                                                        <select name="to"
id="undo_redo_to" class="form-control" size="13" multiple="multiple">
                                                                                <c:forEach
items="${allRoomFacilityPerRoomTypePerHotel}" var="roomFacility">
                                                                                        <option
value="${roomFacility.id}">${roomFacility.name} ${roomFacility.global=='true' ? "*": ""}</option>
                                                                                </c:forEach>

                                                                        </select>
                                                                </div> --%>
                                                        </div>
                                                </div>
                                        </div>
                </div>
</div>
```

9.10.7.28 /admin/hotels/roomTypes/perRoomType/roomTypeEdit.jsp

```
<%@ include file="../../../../layout/taglib.jsp" %>
<jsp:setProperty name="roomType" property="global" value="false"/>
	<div class="padded_wrapper">
		<div class="row">
			<div class="col-xs-12">
				<div class="pull-right">
					<form:form commandName="roomType" cssClass="form-horizontal">
						<div class="form-group">
							<label for="name" class="col-sm-3 control-label">Name:</label>
							<div class="col-sm-9">
								<form:input path="name" cssClass="form-control"/>
							</div>
						</div>
						<div class="form-group">
							<label for="color" class="col-sm-3 control-label">Calendar Color:</label>
							<div class="col-sm-4">
								<div class="input-group" id="color_picker">
									<i></i><form:input  path="color" cssClass="form-control"/>
			<span class="input-group-addon">
		<span class="glyphicon glyphicon-tint"></span>
	</span>
</div>
<script>
	$(function(){
	  $('#color_picker').colorpicker();
	});
</script>
</div>
</div>
<form:hidden path="id" />
<form:hidden path="global" />
<input type="submit" class="btn btn-primary" value="Save">
</form:form>
</div>
</div>
</div>
</div>
```

9.10.7.29 /admin/hotels/roomTypes/perRoomType/roomTypesListing.jsp

```
<%@ include file="../../../../layout/taglib.jsp" %>
<jsp:setProperty name="roomType" property="global" value="false"/>
	<div class="padded_wrapper">
		<div class="row">
			<div class="col-xs-12">
				<table class="table table-bordered table-hover table-striped">
					<thead>
						<tr>
							<th>Room Type</th>
							<th class="text-center">Color</th>
							<th class="text-center">Global</th>
							<th></th>
						</tr>
						<c:forEach items="${allRoomTypeByHotelId}" var="roomType">
							<tr>
<td><span class="roomType" id="${roomType.value.id}"><c:out value="${roomType.value.name}" /></span></td>
<td class="text-center"><span style="color:<c:out value="${roomType.value.color}" />" class="glyphicon glyphicon-stop"></span></td>
<td class="text-center"><c:if test="${roomType.value.global}"><span class="glyphicon glyphicon-ok gc_orange"></span></c:if></td>
<td align="center">
<div class="col-xs-3">
<c:if test="${!roomType.value.global}">
<a class="gc_grey" href="<spring:url value="/admin/hotels/${hotel.id}/room-type/${roomType.value.id}/edit.html"/>">
<span class="glyphicon glyphicon-edit gc_blue"></span> Edit
</a>
</c:if>
</div>
<div class="col-xs-3">
<a class="gc_grey triggerRemove" href="<spring:url value="/admin/hotels/${hotel.id}/room-type/${roomType.value.id}/remove.html"/>">
<span class="glyphicon glyphicon-remove gc_red"></span> Remove
</a>
</div>
<div class="col-xs-3">
```

```
                                                                    <a class="gc_grey" href="<spring:url
value="/admin/hotels/${hotel.id}/facilities/room-facilities/per-roomtype/${roomType.value.id}.html"/>">
                                                                        <span class="glyphicon glyphicon-
list-alt gc_blue"></span> Facilities
                                                                    </a>

                                                                    </div>
                                                                    <div class="col-xs-3">
                                                                    <a class="gc_grey" href="<spring:url
value="/admin/hotels/${hotel.id}/prices/per-roomtype/${roomType.value.id}.html"/>">
                                                                        <span class="glyphicon glyphicon-
euro gc_orange"></span> Pricing
                                                                    </a>

                                                                    </div>
                                                                   </td>

                                                        </tr>
                                                    </c:forEach>
                                                </thead>
                                            </table>
                                        </div>
                                    </div>
                                    <div class="row">
                                        <div class="col-xs-12">
                                            <c:if test="${!empty allGlobalRoomTypeToAdd}">
                                                <form:form commandName="roomType" cssClass="form-inline">
                                                    <div class="form-group">
                                                        <form:label path="id" for="id">Pre-Configured Room
Types : </form:label>
                                                        <form:select path="id" cssClass="form-control">
                                                            <c:forEach
items="${allGlobalRoomTypeToAdd}" var="globalRoomTypeToAdd">
                                                                <form:option
value="${globalRoomTypeToAdd.id}">${globalRoomTypeToAdd.name}</form:option>
                                                            </c:forEach>
                                                        </form:select>
                                                        <input type="hidden" name="global" value="true">
                                                        <input type="submit" class="btn btn-default"
value="Add">
                                                    </div>
                                                </form:form>
                                            </c:if>
                                        </div>
                                    </div>
                                    <div class="row">
```

```
                                <div class="col-xs-12">
                                        <div class="pull-right">
                                                <!-- Button trigger modal -->
                                                <button type="button" class="btn btn-default btn-lg" data-toggle="modal"
data-target="#myModal">
                                                 New Room Type
                                                </button>
                                                <form:form commandName="roomType" cssClass="form-horizontal">
                                                        <!-- Modal -->
                                                        <div class="modal fade" id="myModal" tabindex="-1" role="dialog"
aria-labelledby="myModalLabel" aria-hidden="true">
                                                         <div class="modal-dialog">
                                                          <div class="modal-content">
                                                           <div class="modal-header">
                                                            <button type="button" class="close" data-dismiss="modal" aria-
label="Close"><span aria-hidden="true">&times;</span></button>
                                                            <h4 class="modal-title" id="myModalLabel">New Room
Type</h4>
                                                           </div>
                                                           <div class="modal-body">
                                                                                                <div class="form-
group">
                                                                                                        <label
for="name" class="col-sm-3 control-label">Name:</label>
                                                                                                        <div
class="col-sm-9">
        <form:input path="name" cssClass="form-control"/>
                                                                                                </div>
                                                                                        </div>
                                                                                        <div class="form-
group">
                                                                                                        <label
for="color" class="col-sm-3 control-label">Calendar Color:</label>
                                                                                                        <div
class="col-sm-4">
        <div class="input-group" id="color_picker">
        <i></i><form:input  path="color" cssClass="form-control"/>
                                                                                                <span class="input-group-addon">
                                                                                        <span class="glyphicon glyphicon-tint"></span>
                                                                                </span>
        </div>
        <script>
        $(function(){
           $('#color_picker').colorpicker();
        });
```

```
</script>
                                                            </div>
                                                        </div>
                                            </div>
                                            <div class="modal-footer">
                                                <form:hidden path="global" />
                                              <button type="button" class="btn btn-default" data-
dismiss="modal">Close</button>
                                            <input type="submit" class="btn btn-primary" value="Save">
                                            </div>
                                          </div>
                                        </div>
                                      </div>
                                </form:form>
                            </div>
                        </div>
                    </div>
                </div>
                <!-- Modal -->
                <div class="modal fade" id="roomTypeRemove" tabindex="-1" role="dialog" aria-labelledby="myModalLabel" aria-
hidden="true">
                  <div class="modal-dialog">
                    <div class="modal-content">
                      <div class="modal-header">
                        <button type="button" class="close" data-dismiss="modal" aria-label="Close"><span aria-
hidden="true">&times;</span></button>
                        <h4 class="modal-title" id="myModalLabel">Remove Room Type</h4>
                      </div>
                      <div class="modal-body">
                        Really remove room type : <b><span class="roomType_placeholder"></span></b> ?
                      </div>
                      <div class="modal-footer">
                        <button type="button" class="btn btn-default" data-dismiss="modal">Cancel</button>
                        <a href="" class="btn btn-danger removeBtn">Remove</a>
                      </div>
                    </div>
                  </div>
                </div>
```

9.10.7.30 /admin/user/userDetail.jsp

```
<%@ include file="../../../layout/taglib.jsp" %>
<h2><b><c:out value="${user.username}" /></b></h2> <br>
<div class="section_page_header">Roles</div>
<div  id="user_roles" class="row">
	<div class="col-xs-12 left-block">
		<c:forEach items="${user.roleList}" var="role">
			<h3><span class="label label-primary">${fn:replace(role.name,'_', ' ')}</span></h3>
		</c:forEach>
	</div>
</div>
```

9.10.7.31 /admin/user/users.jsp

```
<%@ include file="../../../layout/taglib.jsp" %>
<table class="table table-bordered table-hover table-striped">
	<thead>
		<tr>
			<th>user name</th>
		</tr>
	</thead>
	<tbody>
		<c:forEach items="${users}" var="user">
			<tr>
				<td>
				 <a href="<spring:url value="/admin/users/${user.id}.html"/>">
					<c:out value="${user.username}" />
				 </a>
				</td>
			</tr>
		</c:forEach>
	</tbody>
</table>
```

9.10.8 CSS

Στουν φάκελο "/src/main/webapp/resources/css" βρίσκονται τα αρχεία μορφοποίησης (css styles). Πρόκειται για τα αρχεία που μορφοποιούν την παραγώμενη html σελίδα ώστε να είναι πιο ευανάγνωστη και πιο ευπαρουσίαστη. Δεδομένου ότι χρησιμοποιούμε το CSS Framework "Bootstrap" τόσο το βασικό αρχείο, όσο και οι βασικές βιβλιοθήκες όπως αυτή για την επέκταση που αφορά στον επιλογέα χρώματος ή ημερομηνίας αλλά και css τρίτων βιβλιωθηκών παραθέτνται παρακάτω ενδεικτικά. Επιπλέον παρουσιάζεται το αρχείο "styles.css" που αφορά στα χειροποίητα (custom) styles που γράφτηκαν προκειμένου το σύστημα να έρθει στο προσωπικό επιθυμητό αποτέλεσμα.

9.10.8.1 Αρχεία CSS τρίτων βιβλιοθηκών

- bootstrap-colorpicker.min.css
- bootstrap-datetimepicker.min.css
- bootstrap-theme.min.css
- bootstrap.min.css
- fullcalendar.css
- fullcalendar.print.css
- jquery.dop.BackendBookingCalendarPRO.css
- signin.css

9.10.8.2 styles.css

```
body{
  color: #333333;
}
#top-page_header{
		background-color:#666;
		padding-top:10px;
		color:#cccccc;
}
#top-page_header select{
		color:#666666;
}
#footer{
		margin-top:10px;
}
a.moto_greekchoices{
		font-weight:bold;
		color:#fff;
		text-decoration: none;
}
a.moto_control_panel{
		font-weight:bold;
		color:#cccccc;
		text-decoration: none;
}
#register{
		 width:90px;
		 margin-top:5px;
		padding-right:0px;
		margin-right:-5px;
}
#login{
		 margin-top:5px;
		padding-right:0px;
		width:90px;
		margin-right:20px;
}
#account a{
		color:#fff;
		margin-top:-1px;
}
```

```
#account span{
		margin-top:2px;
}
#user{
		padding-right:0px;
		width: auto;
		margin-top:10px;
}
#account{
		padding-right:0px;
		width:100px;
		margin-top: 7px;
		margin-right:-5px;
}
#logout{
		padding-right:20px;
		width:110px;
}
#page_header_title_up_level{
 padding-bottom: 0px;
 padding-left:15px;
 padding-right:15px;
 margin: 0px 0 20px 0px;
 border-bottom: 1px solid #666;
}
#page_header_title_up_level h4{
				margin-bottom:5px;
				font-weight: bold;
				color: #666666;
}
#up_level{
		margin-top:10px;
}
#up_level h4{
		color: #666666;
}
.panel_title_icons{
		padding-right:5px;
}
.btn-lg{
		line-height: 2.3333;
```

```
		font-size: 14px;
}
a.btn-lg{
		color:#333333;
}
#general_tools .panel-body a{
		margin-right:20px;
}
#control_panel .panel-default>.panel-heading{
		background-image:linear-gradient(to bottom,#666666 0,#999999 100%)
}
.panel-heading h3{
		color:#fff;
}
.panel-heading span{
		color:#fff;
}
#page_header_descr{
		padding-left:15px;
		padding-bottom:20px;
}
.table th{
		background-color:#E4E4E4;
}
.table-bordered>thead>tr>th{
		border:1px solid #666
}
.table-bordered>thead>tr{
		border:1px solid #666
}
.section_header_cp{
		font-weight: normal;
		 padding-bottom: 0px;
		 padding-left:15px;
		 padding-right:15px;
		 margin: 0px 0 20px 0px;
		 border-bottom: 1px solid #666;
}
#hotel_basic_info{
		background-color: #E4E4E4;
		border:1px solid #666;
```

```
}
a#btn_delete_hotel{
		color:#e82424;
}
#main_borders{
		border-left:1px solid #666;
		border-right:1px solid #666;
		border-bottom:1px solid #666;
}
#main_borders .img-rounded{
		border-top-left-radius:0px;
		border-top-right-radius:0px;
}
#main_wrapper{
		padding:0px 10px 20px 10px;
}
.bordered_padded_wrapper{
				border:1px solid #666;
				margin:0px 10px 20px 10px;
}
.padded_wrapper{
		padding:20px 10px 20px 10px;
}
#content-page-header{
		padding:0px 10px 0px 10px;
}
#hotel_basic_info #contact_info{
		margin-top:30px;
}
.info_per_line .row{
		padding-bottom:15px;
}
#hotel_basic_info #back_btn{
		margin-right:20px;
		width:100px;
}
#hotel_basic_info #save_btn{
		margin-left:20px;
		width:100px;
}
.tab_nav{
```

```
		margin-bottom:20px;
}
.nav-tabs>li.active>a{
		background-color:#f5f5f5;
		font-weight:bold;
		color:#666666;
}
.section_page_header{
 padding-bottom: 5px;
 padding-left:5px;
 padding-right:15px;
 margin: 0px 0 20px 0px;
 border-bottom: 1px solid #666;
 font-weight:bold;
 color:#666666;
 font-size:16px;
}
.page_header_wrapper{
		margin-bottom:30px;
}
.page_header_wrapper .row a{
		margin-right:30px;
		margin-bottom:20px;
		min-width: 120px;
}
.table-bordered>thead>tr>th{
		 color: #666666;
}
label {
		color: #666666;
}
.btn-default, .btn-default:hover , a.btn-lg {
		color: #666666;
		font-size:13px;
		font-weight:bold;
}
.gc_grey , span {
		color: #666666;
}
.gc_blue{
		color:#337ab7;
```

```
}
.gc_red{
		color:red;
}
.gc_green{
		color:#318550;
}
.gc_orange{
		color:orange;
}
.gc_gold{
		color:#FFD700;
}
.gc_greylight{
		color:#ccc;
}
#logout button{
		 margin-top: 5px;
}
#registrationForm input[type="submit"] {
   width:100px;
   margin-right: 25px;
}
#user_roles h3{
		display:inline;
		margin-right:10px;
		font-size:20px;
}
#control_panel a.btn-lg {
		min-width:100px;
}
#select_room_type .pull-right{
		width:100%;
}
.select_room_type_text{
		padding-top:7px;
}
.room_type_name{
		font-size:14px;
}
.room_type_name span{
```

```
		font-weight: normal;
}
.bottom_margin{
		margin-bottom:60px;
}
.manage_all_room_facilities_twoside_selection_above{
		margin-top:10px;
}
#roomFacilities select{
  display:inline;
}
.roomtype_colorfull_labels{	margin-bottom:7px;
		color:#ffffff;
		text-align:center;
		font-size:12px;
		font-weight:bold;
}
/* Booking Calendar (Starts) */
#script-warning {
		display: none;
		background: #eee;
		border-bottom: 1px solid #ddd;
		padding: 0 10px;
		line-height: 40px;
		text-align: center;
		font-weight: bold;
		font-size: 12px;
		color: red;
}
#loading {
		display: none;
		position: absolute;
		top: 10px;
		right: 10px;
}
#calendar {
		max-width: 768px;
		margin: 40px auto;
		margin-top:0px;
		padding: 0 10px;
}
```

```
.fc-title, .fc-time{
		color:#ffffff;
}
#hotel_basic_info form .col-xs-3{
  padding-top:6px
}
#hotel_basic_info form input[type="submit"]{
		width:100px;
}
/* Booking Calendar (Ends) */
/* Availability Calendar (Starts) */
/* body{
    background-color: #f0f0f0;
    font: normal 14px/20px Helvetica, Arial, sans-serif;
    margin: 0;
    padding: 0;
}
 */
.clear{
    clear: both;
}
strong{
    font-weight: bold;
}
#wrapper{
    max-width: 940px;
    margin: auto;
}
h1{
    border-bottom: 1px dashed #555555;
    color: #555555;
    font: normal 32px/39px Helvetica, Arial, sans-serif;
    margin: auto;
    max-width: 900px;
    padding: 40px 30px;
}
h2{
    color: #000000;
    font: normal 20px/20px Helvetica, Arial, sans-serif;
    margin: 20px 0;
}
```

```
h3{
   color: #000000;
   font: normal 16px/20px Helvetica, Arial, sans-serif;
   margin: 20px 0;
}
p{
   color: #555555;
   font: 14px/20px Helvetica, Arial, sans-serif;
   margin: 0 0 20px 0;
}
.reload-btn{
   background: #000000;
   border: none;
   -webkit-border-radius: 5px;
   -moz-border-radius: 5px;
   border-radius: 5px;
   -moz-box-shadow: 0px 1px 1px #666666;
   -webkit-box-shadow: 0px 1px 1px #666666;
   box-shadow: 0px 1px 1px #666666;
   color: #ffffff;
   cursor: pointer;
   font: normal 14px/40px Helvetica, Arial, sans-serif;
   margin: 0 0 20px 0;
   padding: 0 15px;
}
.reload-btn:hover{
   background: #212121;
}
#backend-container,
#frontend-container{
   background-color: #f0f0f0;
   padding: 20px 30px;
}
#frontend-container{
   margin: 0 0 20px 0;
}
#backend,
#frontend{
   max-width: 900px;
}
ul{
```

```
    list-style-type: square;
}
#DOPBCP_status{
                visibility:hidden;
                display:none;
}
/* Availability Calendar (Ends) */
```

9.10.9 JavaScript

Στουν φάκελο "/src/main/webapp/resources/js" βρίσκονται τα αρχεία javascript. Πρόκειται για τα αρχεία που επεκτήνουν την λειτουργικότητα της παραγώμενη html σελίδας ώστε να είναι πιο διαδραστική και σε ορισμένες περιπτώσεις ασύγχρονη (Ajax Calls). Δεδομένου ότι χρησιμοποιούμε το CSS Framework "Bootstrap" τόσο το βασικό αρχείο, όσο και οι βασικές βιβλιοθήκες όπως αυτή για την επέκταση που αφορά στον επιλογέα χρώματος ή ημερομηνίας αλλά και javascript τρίτων βιβλιωθηκών παρατίθενται παρακάτω ενδεικτικά. Επιπλέον παρουσιάζεται το αρχείο "script.js" που αφορά στα χειροποίητα (custom) javascript που γράφτηκαν προκειμένου το σύστημα να λειτουργει με τον προσωπικό προκαθορισμένο τρόπο.

9.10.9.1 Αρχεία JS τρίτων βιβλιοθηκών

- bootstrap.min.js
- bootstrap-colorpicker.js
- bootstrap-datetimepicker.min.js
- fullcalendar.min.js
- jquery.dop.BackendBookingCalendarPRO.js
- jquery.min.js
- moment.min.js
- multiselect.min.js

9.10.9.2 script.js

```
var csrf = $('meta[name=_csrf]').attr("content");
function getElementFromPath(path,arg_no){
		urlParts = path.split("/")
		return urlParts[arg_no];
}
$(function(){
/*  function getParameterByName(name) {
			name = name.replace(/[\[]/, "\\[").replace(/[\]]/, "\\]");
			var regex = new RegExp("[\\?&]" + name + "=([^&#]*)"),
			  results = regex.exec(location.search);
```

```
                return results === null ? "" : decodeURIComponent(results[1].replace(/\+/g, " "));
}*/
// Room Facilities Per Room Type Listing - Select Room Type (Starts)
$('.select_room_type').bind('change', function () {
  var url = $(this).val()+".html"; // get selected option value
  if (url) { // require url to have value
    window.location = url; // open url
  }
  return false;
});
// Room Facilities Per Room Type Listing - Select Room Type (Starts)
/* Booking Calendar (Starts) */
            var today = new Date();
            var dd = today.getDate();
            var mm = today.getMonth()+1; //January is 0!
            var yyyy = today.getFullYear();
            // Get Hotel Id from Url (Starts)
            var pathname = window.location.pathname;
            urlParts = pathname.split("/")
            var hotelId = urlParts[3];
  // Get Hotel Id from Url (Ends)
            $('#calendar').fullCalendar({
                        header: {
                                    left: 'prev,next today',
                                    center: 'title',
                                    right: 'month,agendaWeek,agendaDay'
                        },
                        defaultDate: yyyy+'-'+mm+'-'+dd,
                        editable: false,
                        eventLimit: true, // allow "more" link when too many events
                        events: {
                                    url: '/admin/hotels/'+hotelId+'/events.json',
                                    error: function() {
                                                $('#script-warning').show();
                                    }
                        },
                        loading: function(bool) {
                                    $('#loading').toggle(bool);
                        }
            });
            $('#calendar').fullCalendar('option', 'aspectRatio', 1.0);
```

```
 /* Booking Calendar (Ends) */
 /* Booking Calendar (Starts) */
        if ( $( "#backend" ).length ) {
                // Get Hotel Id from Url (Starts)
                var csrf = $('meta[name=_csrf]').attr("content");
                var pathname = window.location.pathname;
                urlParts = pathname.split("/")
                var hotelId = urlParts[3];
                var roomTypeIdTemp = urlParts[5];
                var roomTypeId = roomTypeIdTemp.replace(".html", "");
        /*
                $.ajax({
                        method: "POST",
                        url: "/admin/hotels/"+hotelId+"/availabilities-calendar/"+roomTypeId+"/ajax.json",
                        data: { name: "John", location: "Boston", _csrf:csrf }
                       })
                        .done(function( msg ) {
                                alert(JSON.stringify(msg));
                        });*/
          // Get Hotel Id from Url (Ends)
                //  $('#backend').DOPBackendBookingCalendarPRO();
                //  $('#frontend').DOPFrontendBookingCalendarPRO();
                //Get data from the database example.
                 $('#backend').DOPBackendBookingCalendarPRO({'DataURL': '/admin/hotels/'+hotelId+'/availabilities-
calendar/'+roomTypeId+'/dataurl.json',
                                                'SaveURL': '/admin/hotels/'+hotelId+'/availabilities-
calendar/'+roomTypeId+'/saveurl.json'});
                 $('#backend-refresh').click(function(){
                //     $('#backend').DOPBackendBookingCalendarPRO({'Reinitialize': true});
                //Get data from the database example.
                   $('#backend').DOPBackendBookingCalendarPRO({'Reinitialize': true,
                                                'DataURL': '/admin/hotels/'+hotelId+'/availabilities-
calendar/'+roomTypeId+'/dataurl.json',
                                                'SaveURL': '/admin/hotels/'+hotelId+'/availabilities-
calendar/'+roomTypeId+'/saveurl.json'});
                 });
        }
 /* Booking Calendar (Ends) */
 /* Period Javascript (Starts) */
        if ($( "#periodRemove" ).length ) {
                $(".triggerRemove").click(function(e) {
                        e.preventDefault();
                        $("#periodRemove .removeBtn").attr("href",$(this).attr("href"));
```

```
                                period_id = (getElementFromPath($(this).attr("href"),5));
                                $(".modal .period_placeholder").html($("#"+period_id+".period").html());
                                $("#periodRemove").modal();
                        });
                }
 /* Period Javascript (Ends) */
/* Room Type Javascript (Starts) */
                if ($( "#roomTypeRemove" ).length ) {
                        $(".triggerRemove").click(function(e) {
                                e.preventDefault();
                                $("#roomTypeRemove .removeBtn").attr("href",$(this).attr("href"));
                                roomType_id = (getElementFromPath($(this).attr("href"),5));
                                $(".modal .roomType_placeholder").html($("#"+roomType_id+".roomType").html());
                                $("#roomTypeRemove").modal();
                        });
                }
 /* Room Type Javascript (Ends) */
 /* Facility Javascript (Starts) */
                if ($( "#facilityRemove" ).length ) {
                        $(".triggerRemove").click(function(e) {
                                e.preventDefault();
                                $("#facilityRemove .removeBtn").attr("href",$(this).attr("href"));
                                facility_id = (getElementFromPath($(this).attr("href"),6));
                                $(".modal .facility_placeholder").html($("#"+facility_id+".facility").html());
                                $("#facilityRemove").modal();
                        });
                }
 /* Facility Javascript (Ends) */
 /* Hotel Javascript (Starts) */
                if ($( "#hotelRemove" ).length ) {
                        $("#btn_delete_hotel").click(function(e) {
                                e.preventDefault();
                                $("#hotelRemove .removeBtn").attr("href",$(this).attr("href"));
                                /*hotel_id = (getElementFromPath($(this).attr("href"),3));*/
                                $(".modal .hotel_placeholder").html($("#page_header_title_up_level h4").html().replace(" Control
Panel", ""));


                                $("#hotelRemove").modal();
                        });
                }
 /* Hotel Javascript (Ends) */
});
```

```
// Room Facilities per All Room Types
function checkAllOptions(roomTypeId){
	if(roomTypeId === undefined || roomTypeId === null){
		$('#undo_redo_to option').prop('selected', true);
	}else{
		$('#undo_redo'+roomTypeId+'_to option').prop('selected', true);
	}
}
```