ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

ΤΜΗΜΑ ΨΗΦΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΤΙΤΛΟΣ: Σχεδίαση και υλοποίηση εφαρμογής on-line κρατήσεων δωματίου ξενοδοχείου Relax Hotel σε πλατφόρμα υπολογιστικού νέφους

ΟΝΟΜΑΤΕΠΩΝΥΜΟ: ΔΕΛΙΤΖΙΑΣ ΙΩΑΝΝΗΣ

ΑΜ: Ε/08031

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ: ΜΗΛΙΩΝΗΣ ΑΠΟΣΤΟΛΟΣ

ΕΥΧΑΡΙΣΤΙΕΣ

Σε αυτό το σημείο θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα καθηγητή κ. Μηλιώνη Απόστολο για την ευκαιρία που μου έδωσε να ασχοληθώ με ένα τόσο σοβαρό και ενδιαφέρον θέμα , όπως είναι το Cloud Computing. Ακόμη τον ευχαριστώ για την στήριξη και τον πολύτιμο χρόνο που διέθεσε για να έχουμε όσο το δυνατόν καλύτερη συνεργασία.

Θα ήθελα επίσης να ευχαριστήσω ολόψυχα την οικογένειά μου για την συμπαράσταση και την πολύτιμη στήριξη που μου παρείχαν σε όλα τα χρόνια της φοιτητικής μου ζωής.

Περιεχόμενα

1. Εισαγωγή στο Cloud Computing

Είναι αδιαμφισβήτητο το γεγονός πως η πρόοδος της τεχνολογίας είναι αλματώδης στην εποχή μας. Η Πληροφορική αν και νεότατη επιστήμη κατάφερε σε ελάχιστο χρόνο να αλλάξει σε παγκόσμιο επίπεδο τόσο την κοινωνία στην οποία ζούμε, καθώς και την οικονομία. Ο 21ος αιώνας χαρακτηρίζεται ως **'ο αιώνας της πληροφορίας'**, αν σκεφτούμε την ανάπτυξη που γνώρισε ο τομέας της Πληροφορικής και των υπολογιστών. Με τον καιρό όμως, δημιουργήθηκαν μεγαλύτερες ανάγκες όσον αφορά στις πληροφορίες που διαχειρίζονται και επεξεργάζονται οι υπολογιστές. Προέκυψε λοιπόν ανάγκη για μεταφορά δεδομένων.

Έτσι δημιουργήθηκαν τα δίκτυα για να εξυπηρετήσουν τις ανάγκες, που προέκυψαν από την εξάπλωση της χρήσης των υπολογιστών. Σκοπός των δικτύων είναι ο διαμερισμός των πόρων του συστήματος και η ανταλλαγή πληροφοριών κάθε μορφής. Υπάρχουν σαφείς διακρίσεις ανάμεσα στο λογισμικό και το υλικό των δικτύων, καθώς επίσης και στον τύπο των δικτύων όσον αφορά την έκτασή τους αλλά και στις τεχνολογίες που χρησιμοποιούν. Επιπλέον η διασύνδεση τοπικών δικτύων και δικτύων ευρείας περιοχής, ανεξαρτήτως λειτουργικού συστήματος και αρχιτεκτονικής, δημιουργεί το Internet.

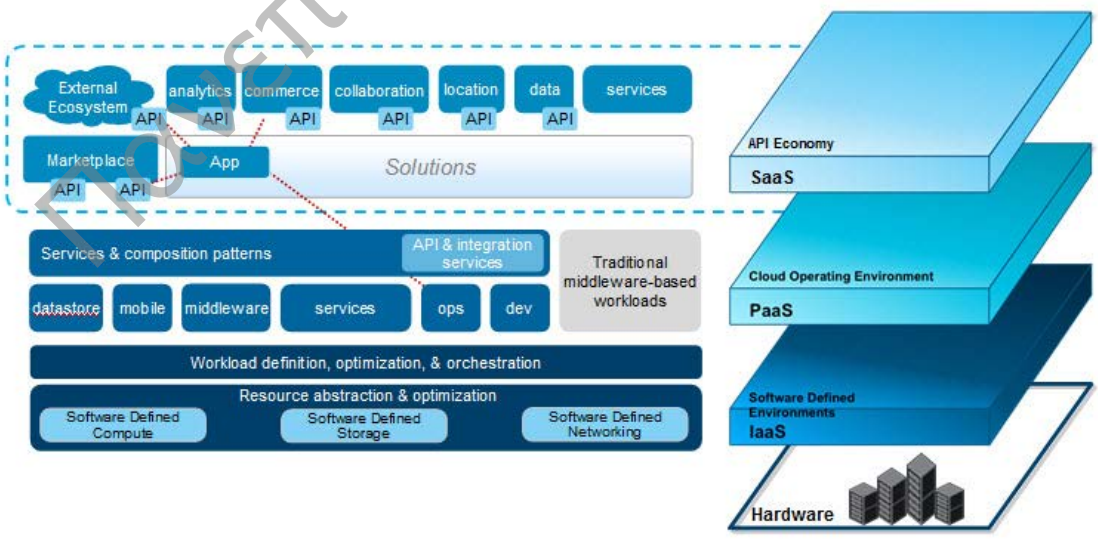
Σήμερα η χρήση του Internet σε πολλές κατηγορίες επαγγελμάτων αλλά και σε πολλές δραστηριότητες της καθημερινής μας ζωής είναι πια διαδεδομένη. Με την γρήγορη εξάπλωση του Internet και σε συνδυασμό με την ύπαρξη δικτύων υψηλών ταχυτήτων και την τεχνολογική ανάπτυξη των υπολογιστών αλλά και του λογισμικού, έχουν δημιουργηθεί νέα δεδομένα τα οποία ανοίγουν νέους δρόμους στην ανάπτυξη της τεχνολογίας. Ειδικότερα στον τομέα της Πληροφορικής, νέες τεχνολογίες υιοθετούνται για την επίτευξη των στόχων των εργαζομένων και των επιχειρήσεων. Μία τέτοια τεχνολογία είναι και το Cloud Computing το οποίο θα μελετηθεί σε αυτή την εργασία. Πλέον έχει μπει για τα καλά στο χώρο των επιχειρήσεων και αποτελεί αναπόσπαστο κομμάτι τους για την πρόοδο και την οικονομική ευημερία τους. Το Cloud Computing βρίσκεται παντού, σε περιοδικά τεχνολογίας καθώς και στο Διαδίκτυο , σε σχετικές ιστοσελίδες. Αυτό που σηκώνει πολλή κουβέντα είναι το τι είναι το Cloud Computing. Παρακάτω θα δώσουμε διάφορους ορισμούς του Cloud Computing.

Το Cloud Computing είναι μια τεχνολογία που χρησιμοποιεί το Διαδίκτυο μεταξύ των απομακρυσμένων κεντρικών παροχέων (servers) για τη συντήρηση δεδομένων και εφαρμογών. Επιτρέπει σε καταναλωτές

και επιχειρήσεις να χρησιμοποιούν εφαρμογές χωρίς απαραίτητα να τις εγκαταστήσουν και να έχουν πρόσβαση στους προσωπικούς τους φακέλους μέσω οποιουδήποτε υπολογιστή συνδεδεμένου στο Διαδίκτυο. Ένα απλό παράδειγμα υπολογιστικού νέφους είναι τα ηλεκτρονικά ταχυδρομεία Yahoomail, Gmail, Hotmail. Δεν χρειάζεται λογισμικό ή παροχέας για να τα χρησιμοποιήσουμε. Το λογισμικό του παροχέα και του διαχειριστή των ηλεκτρονικών μηνυμάτων βρίσκεται σε αυτό το διαδικτυακό σύννεφο και ρυθμίζεται εξ' ολοκλήρου από τον αντίστοιχο παροχέα υπηρεσιών π.χ Yahoo, Google. Οι καταναλωτές απλά χρησιμοποιούν το λογισμικό μόνοι τους και απολαμβάνουν τις παροχές και τα οφέλη των υπηρεσιών αυτών , δηλαδή τη λήψη και την αποστολή μηνυμάτων.

Το Cloud Computing είναι η παροχή της πληροφορικής ως υπηρεσία και όχι ως ένα προϊόν, σύμφωνα με την οποία, μοιράζονται πόρους, λογισμικό και πληροφορίες παρέχονται στους υπολογιστές και σε άλλες συσκευές, μέσω δικτύου (συνήθως του Ίντερνετ).

Το Cloud Computing είναι ένα μοντέλο που επιτρέπει ευέλικτη, on-demand δικτυακή πρόσβαση σε ένα κοινόχρηστο σύνολο παραμετροποιήσιμων υπολογιστικών πόρων (π.χ. δίκτυα, servers, αποθηκευτικοί χώροι, εφαρμογές και υπηρεσίες), το οποίο μπορεί να τροφοδοτηθεί γρήγορα και να διατεθεί με ελάχιστη προσπάθεια διαχείρισης ή αλληλεπίδραση με τον πάροχο της υπηρεσίας. Αυτό το cloud μοντέλο προωθεί την διαθεσιμότητα και αποτελείται από πέντε βασικά χαρακτηριστικά, τρία μοντέλα παροχής υπηρεσιών, και τέσσερα μοντέλα ανάπτυξης.



Εικόνα 1.1- Απεικόνιση ορισμού NIST για την αρχιτεκτονική του Cloud Computing.

Το Cloud Computing αναπτύχθηκε από τεχνολογίες και επιχειρηματικές προσεγγίσεις που προέκυψαν κατά τη διάρκεια πολλών ετών.

1.1 Στοιχεία για την προέλευση του cloud computing

Τα σημαντικότερα στοιχεία για την προέλευση του cloud computing συνοψίζονται παρακάτω:

- **Grid Computing** (**Υπολογιστικό Πλέγμα**) **:**

  Η έννοια του Υπολογιστικού Πλέγματος αρχικά εμφανίστηκε στα μέσα της δεκαετίας του 90' προβάλλοντας μια πρόταση που έγινε τότε για την χρήση κατανεμημένης υπολογιστικής υποδομής για προηγμένες επιστήμες και μηχανική. Το Υπολογιστικό Πλέγμα προσπαθεί να φέρει κοντά, σε συνεργασία και με διάφανο τρόπο υπολογιστικούς πόρους, οι οποίοι είναι διάσπαρτοι ανά τον κόσμο. Το Grid Computing είναι μια μορφή κατανεμημένου υπολογιστικού συστήματος. Επίσης αποτελεί μια συλλογή από υπολογιστικά συστήματα τα οποία αναφέρονται συχνά σαν <<κόμβοι>>, <<πόροι>>, <<πελάτες>>, << hosts>> και άλλους παρόμοιους όρους. Όλα αυτά τα συστήματα συνεισφέρουν διάφορους συνδυασμούς πόρων στο πλέγμα σαν σύνολο. Κάποιοι πόροι μπορούν να χρησιμοποιηθούν από όλους τους χρήστες του πλέγματος, ενώ άλλοι μπορεί να έχουν συγκεκριμένους περιορισμούς.

- **Utility Computing:**

  Είναι η "συσκεύαση" υπολογιστικών και αποθηκευτικών πόρων και η παροχή τους ως μια υπηρεσία παρόμοια με αυτές της κοινής ωφέλειας.

- **Autonomic Computing:**

  Πολλές Cloud υλοποιήσεις σήμερα βασίζονται πάνω στα Grid και έχουν αυτονομιστικά χαρακτηριστικά και χρεώνουν τις υπηρεσίες τους ως παροχές. Αλλά το Cloud computing αποτελεί κάτι περισσότερο από αυτά και δεν είναι απαραίτητο κάθε διαφορετικό cloud να περιέχει τα παραπάνω. Για παράδειγμα δίκτυα τύπου peer to peer, όπως το BitTorrent και το Skype έχουν ελάχιστο ή και καθόλου κεντρικό έλεγχο.

- **Platform virtualization:**

Είναι η λογική κατάτμηση των φυσικών υπολογιστών πόρων σε πολλαπλά περιβάλλοντα εκτέλεσης, συμπεριλαμβανομένων των servers, εφαρμογές, και λειτουργικά συστήματα. Το Virtualization βασίζεται στην έννοια μιας εικονικής μηχανής που εκτελείται σε μια φυσική υπολογιστική πλατφόρμα. Το virtualization ελέγχεται από ένα Virtual Machine Monitor (VMM), γνωστό ως hypervisor. Το Xen, ένα open-source hypervisor, είναι ένα hypervisor που ευρέως χρησιμοποιείται για το cloud computing.

- **Software as a Service(SaaS):**

Είναι η διανομή του λογισμικού και το μοντέλο ανάπτυξης στο οποίο οι αιτήσεις παρέχονται στους πελάτες ως υπηρεσία. Οι εφαρμογές μπορούν να τρέχουν στα υπολογιστικά συστήματα των χρηστών ή στους διακομιστές Web της υπηρεσίας παροχής. Το SaaS παρέχει την αποτελεσματική διαχείριση του patch και προωθεί την συνεργασία.

- **Service Oriented Architectures(SOA):**

Μια σειρά από υπηρεσίες που επικοινωνούν μεταξύ τους, των οποίων τα interfaces είναι γνωστά και περιγράφονται, των οποίων οι λειτουργίες είναι χαλαρά συνδεδεμένα(ο τύπος του interface δεν συνδέεται με την εφαρμογή), και των οποίων η χρήση μπορεί να ενσωματωθεί από πολλαπλούς οργανισμούς. Οι SOA υπηρεσία interfaces που ορίζονται στην XML και οι υπηρεσίες εκφράζονται σε WSDL. Οι εφαρμογές μπορούν να έχουν πρόσβαση σε υπηρεσίες UDDI(Universal Description,Definition and Integration)καταχώρηση καταλόγου.

- **Παραδείγματα cloud υπηρεσιών:**

Η επιχείρηση Salesforce.com παρέχει υπηρεσίες cloud computing από το 1999.Η Amazon Web Services διαθέτει υπηρεσίες Cloud Computing από το 2002.Το Elastic Compute Cloud (EC2) προσφέρεται από την Amazon σε μικρές επιχειρήσεις και ιδιώτες σύμφωνα με την οποία υπολογιστικοί πόροι μπορούν να ενοικιάζονται. Το Google προσφέρει το Google Apps, που περιλαμβάνει εφαρμογές Web,όπως το Gmail,Docs και το calendar. Η Microsoft Azure Services Platform υποστηρίζει Cloud εφαρμογές οι οποίες φιλοξενούνται και εκτελούνται στα Microsoft data centers. Η VMware είναι μια εταιρεία που παρέχει virtualization λογισμικό για διάφορες πλατφόρμες. Η

IBM και Juniper Networks σχηματίζουν μία συνεργασία για την παροχή υπηρεσιών cloud computing.

1.2 Βασικά χαρακτηριστικά του cloud computing

Τα βασικά χαρακτηριστικά του cloud computing είναι τα εξής:

- **On-demand self-service**:

Ένας καταναλωτής μπορεί να δεσμεύσει από μόνος του τους υπολογιστικούς πόρους που χρειάζεται, όπως χρόνο στον server και αποθηκευτικό χώρο στο δίκτυο, ανάλογα με τις ανάγκες του αυτόματα, χωρίς να απαιτείται ανθρώπινη αλληλεπίδραση με το φορέα παροχής κάθε υπηρεσίας.

- **Broad network access** (Ευρεία πρόσβαση στο δίκτυο):

Οι δυνατότητες είναι διαθέσιμες μέσω του δικτύου και προσβάσιμες μέσω τυποποιημένων μηχανισμών που προωθούν την χρήση από ετερογενείς thin ή thick client πλατφόρμες (π.χ. κινητά τηλέφωνα, φορητούς υπολογιστές και PDAs).

- **Resource pooling (Κοινή διάθεση των πόρων):**

Οι υπολογιστικοί πόροι του παρόχου χρησιμοποιούνται για να εξυπηρετήσουν πολλαπλούς καταναλωτές με τη χρήση του μοντέλου πολλαπλών μισθωτών (multi-tenant), με τους διάφορους φυσικούς και εικονικούς πόρους να ανατίθενται δυναμικά και εκ νέου ανάλογα με τη ζήτηση των καταναλωτών. Υπάρχει μια αίσθηση ανεξαρτησίας από τον τόπο στο γεγονός ότι ο πελάτης δεν έχει γενικά κανέναν έλεγχο ή γνώση σχετικά με την ακριβή τοποθεσία των παρεχόμενων πόρων, αλλά μπορεί να είναι σε θέση να προσδιορίζει την τοποθεσία σε ένα υψηλότερο επίπεδο αφαίρεσης (π.χ. χώρα, κράτος, ή datacenter). Παραδείγματα πόρων αποτελούν οι αποθηκευτικοί χώροι, η επεξεργασία, η μνήμη, το bandwidth του δικτύου, καθώς και οι εικονικές μηχανές.

- **Rapid elasticity** (Ταχεία ελαστικότητα):

Οι πόροι μπορούν να δεσμευτούν προς χρήση γρήγορα και ελαστικά, σε ορισμένες περιπτώσεις αυτόματα, έτσι ώστε να εμφανιστούν άμεσα ως μη διαθέσιμοι (scale out) και επίσης να αποδεσμευτούν γρήγορα για να εμφανιστούν ξανά ως διαθέσιμοι (scale in). Για τον καταναλωτή, οι διαθέσιμες δυνατότητες για δέσμευση και χρήση συχνά

φαίνεται να είναι απεριόριστες και μπορούν να αγοραστούν ανά πάσα στιγμή και σε οποιαδήποτε ποσότητα.

- **Measured Service** (Μετρήσιμα επίπεδα παροχής υπηρεσιών):

Τα συστήματα cloud ελέγχουν και βελτιστοποιούν αυτόματα τη χρήση των πόρων, αξιοποιώντας μια δυνατότητα μέτρησης σε κάποιο επίπεδο αφαίρεσης που είναι κατάλληλο για το είδος της υπηρεσίας (π.χ. αποθήκευση, επεξεργασία, bandwidth, ενεργοί λογαριασμοί χρηστών). Η χρήση των πόρων μπορεί να παρακολουθείται, να ελέγχεται, και να παρουσιάζεται με τη μορφή reports, παρέχοντας διαφάνεια τόσο για τον πάροχο όσο και για τον καταναλωτή της χρησιμοποιούμενης υπηρεσίας.

Βιβλιογραφία :

1. Grid Computing, Wikipedia
http://en.wikipedia.org/wiki/Grid_computing

2. Cloud Computing, Wikipedia
http://en.wikipedia.org/wiki/Cloud_computing

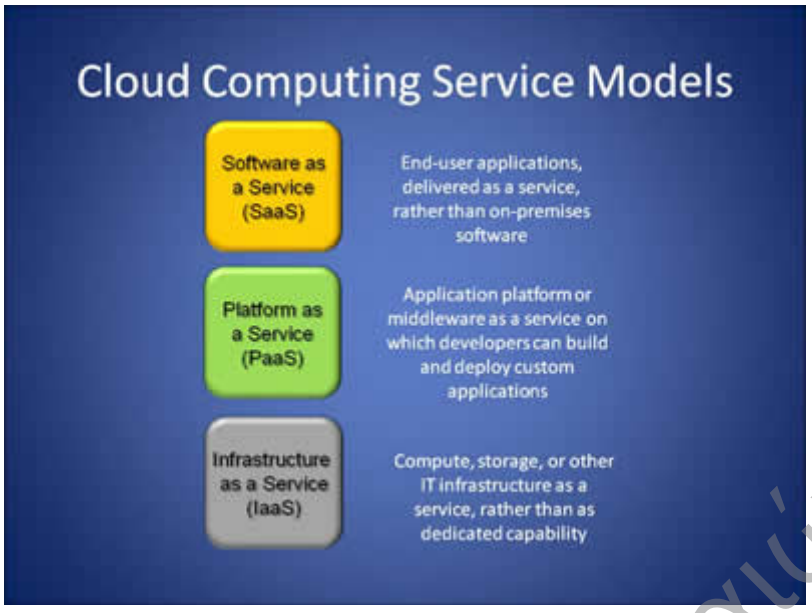
3. I. Foster, and C.Kesselman, "The GRID 2, Blueprint for a New Computing Infrastracture", Morgan Kauffman, 2004

4. Introduction to cloud computer architecture, 1st edition, Sun Microsystems

2. Τα μοντέλα υπηρεσίας και ανάπτυξης στο Cloud Computing

2.1 Τα μοντέλα υπηρεσίας στο cloud computing:

Το Cloud Computing μπορεί να διαχωριστεί σε δυο κατηγορίες :ως προς το είδος της υπηρεσίας που προσφέρεται και ως προς τα μοντέλα ανάπτυξης που υπάρχουν.
Ξεκινώντας από τα είδη των υπηρεσιών, τα διαθέσιμα μοντέλα του cloud computing είναι τα Software-as-as-a-Service, Platform-as-a-Service και το Infrastructure-as-a-Service. Το κάθε ένα από αυτά εξυπηρετεί διαφορετικές ανάγκες και προσφέρει διαφορετικές υπηρεσίες.



Εικόνα 2.1-Τα μοντέλα υπηρεσίας στο Cloud

- **Cloud Software as a Service (SaaS)**:

Βασίζεται στην λογική της ενοικίασης λογισμικού από έναν πάροχο υπηρεσιών αντί της αγοράς της άδειας χρήσης. Ο πάροχος αναλαμβάνει όλες τις αναβαθμίσεις καθώς και τη συντήρηση της εφαρμογής. Όλες οι υπηρεσίες παρέχονται στον πελάτη μέσω του Διαδικτύου και τις χρησιμοποεί όταν τις ζητήσει. Οι Cloud εφαρμογές δεν χρειάζονται εγκατάσταση και δεν καταναλώνουν τους φυσικούς πόρους του συστήματος. Παρέχουν δίκτυο βασισμένο στην πρόσβαση, διαχείριση, κεντρική ενημέρωση και επιδιόρθωση, χωρίς να χρειάζεται λήψη ενημερώσεων από την πλευρά του πελάτη.

Η δυνατότητα που παρέχεται στον καταναλωτή είναι να χρησιμοποιεί τις εφαρμογές του παρόχου που τρέχουν σε μια cloud υποδομή. Οι εφαρμογές είναι προσβάσιμες από διάφορες client συσκευές μέσω ενός thin client interface, όπως ένα πρόγραμμα περιήγησης στο Web (π.χ. web-based email). Ο καταναλωτής δεν έχει τη διαχείριση ή τον έλεγχο της χρησιμοποιούμενης cloud υποδομής συμπεριλαμβανομένων των δικτύων, των servers, των λειτουργικών συστημάτων, των αποθηκευτικών μονάδων, ή ακόμα και μεμονωμένων δυνατοτήτων της εφαρμογής, με την πιθανή εξαίρεση κάποιων περιορισμένων user-specific ρυθμίσεων παραμετροποίησης των εφαρμογών.

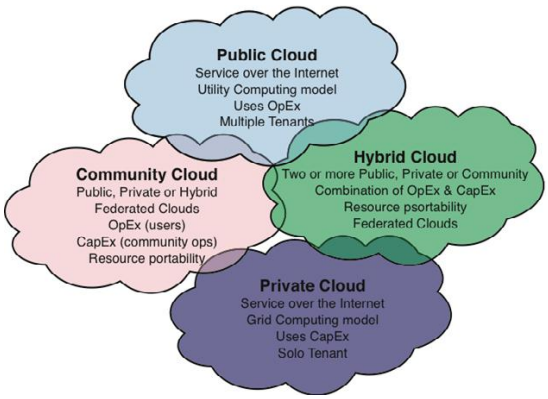
- **Cloud Platform as a Service (PaaS):**

Η δυνατότητα που παρέχεται στον καταναλωτή είναι να αναπτύσσει πάνω στην cloud υποδομή εφαρμογές που έχει δημιουργήσει ή εφαρμογές που έχει αποκτήσει, οι οποίες έχουν δημιουργηθεί με χρήση γλωσσών προγραμματισμού και εργαλείων που υποστηρίζονται από τον πάροχο. Ο καταναλωτής δεν διαχειρίζεται ούτε ελέγχει τη σχετική cloud υποδομή που συμπεριλαμβάνει τα δίκτυα, τους servers, τα λειτουργικά συστήματα ή τα αποθηκευτικά μέσα, αλλά έχει τον έλεγχο των εφαρμογών που έχουν αναπτυχθεί, και ενδεχομένως, των παραμετροποιήσεων του περιβάλλοντος φιλοξενίας των εφαρμογών.

- **Cloud Infrastructure as a Service (IaaS):**

Η δυνατότητα που παρέχεται στον καταναλωτή είναι να μπορεί να δεσμεύσει προς χρήση επεξεργαστική ισχύ, αποθηκευτικά μέσα, δίκτυα, και άλλους θεμελιώδεις υπολογιστικούς πόρους, όπου ο καταναλωτής είναι σε θέση να αναπτύξει και να εκτελέσει αυθαίρετο λογισμικό, το οποίο μπορεί να περιλαμβάνει λειτουργικά συστήματα και εφαρμογές. Ο καταναλωτής δεν έχει τη διαχείριση ή τον έλεγχο της χρησιμοποιούμενης cloud υποδομής, αλλά έχει τον έλεγχο των λειτουργικών συστημάτων, των αποθηκευτικών μέσων, των εφαρμογών που έχουν αναπτυχθεί και πιθανόν κάποιον περιορισμένο έλεγχο επιλεγμένου εξοπλισμού δικτύωσης (π.χ. firewalls). Επίσης, είναι δυνατή η μεταφορά εικονικών μηχανών από το ιδιόκτητο περιβάλλον της εταιρίας ή του ιδιώτη στο Cloud, με συνοπτικές διαδικασίες.

2.2 Τα μοντέλα ανάπτυξης του cloud computing:



Εικόνα 2.2 - Τα μοντέλα ανάπτυξης των cloud computing υπηρεσιών

Μέσα σε κάθε ένα από τα τρία μοντέλα παράδοσης υπάρχουν και πολλαπλά μοντέλα ανάπτυξης. Για παράδειγμα, ένα μοντέλο παράδοσης

SaaS μπορεί να παρουσιαστεί σε χρήστες διαφόρων μοντέλων ανάπτυξης, όπως ενός private ή public cloud. Αυτά τα μοντέλα ανάπτυξης είναι τεχνικά και λειτουργικά άσχετα μεταξύ των μοντέλων παράδοσης, δηλαδή οποιοδήποτε από τα μοντέλα παράδοσης μπορεί να υπάρξει σε οποιαδήποτε από τα μοντέλα ανάπτυξης, αν και ένα συγκεκριμένο μοντέλο παράδοσης /ανάπτυξης μπορεί να είναι πιο κοινό από άλλα (π.χ. το μοντέλο παράδοσης SaaS συναντάτε συνήθως σε public cloud). Επιπλέον, με βάση τη χρήση του cloud από έναν οργανισμό και την σχέση του με την επιχείρηση ως σύνολο, αυτά τα μοντέλα ανάπτυξης cloud συχνά αναφέρεται ως εξωτερικά (external) ή εσωτερικά (internal) clouds.Κάθε ένα από αυτά τα μοντέλα, ωστόσο, πρέπει να συμμερίζονται τις θεμελιώδεις αρχές του cloud computing. Κάθε μοντέλο ανάπτυξης χρησιμοποιεί συσκευές που συνδέονται με το Internet.Κάθε μοντέλο προβλέπει τη δυναμική κλιμάκωση των εικονικών πόρων. Οι χρήστες του κάθε μοντέλου συνήθως δεν έχουν τον έλεγχο της τεχνολογίας που χρησιμοποιείται.

- **Private cloud:**

Η υποδομή cloud λειτουργεί αποκλειστικά και μόνο για έναν οργανισμό. Μπορεί να διαχειρίζεται από την οργάνωση ή από κάποιον τρίτο και μπορεί να υπάρχει με on premise η’off premise(με άδεια η’ χωρίς).

- **Community cloud:**

Η υποδομή cloud μοιράζεται από πολλούς οργανισμούς και υποστηρίζει μια συγκεκριμένη προδιαγραφή που έχει κοινές ανησυχίες(για παράδειγμα, η αποστολή, οι απαιτήσεις ασφάλειας, η πολιτικής της και τις εκτιμήσεις της συμμόρφωσης).Μπορεί να γίνεται από τις οργανώσεις ή από τρίτους και μπορεί να υπάρχουν με on premise η off premise.

- **Public cloud:**

Η υποδομή cloud είναι διαθέσιμη στο ευρύ κοινό ή σε μια μεγάλη ομάδα της βιομηχανίας και ανήκει σε μια οργάνωση που παρέχει cloud υπηρεσίες.

- **Hybrid cloud :**

Η υποδομή cloud είναι μια σύνθεση δύο ή περισσότερων cloud (private,community ή public) όπου παραμένουν ως μοναδικές οντότητες, αλλά συνδέονται μεταξύ τους με τυποποιημένη ή αποκλειστική τεχνολογία που επιτρέπει φορατότητα δεδομένων και

εφαρμογών (π.χ.,cloud γεμάτο για εξισορρόπηση φορτίου μεταξύ των cloud).Ένας οργανισμός μπορεί να εφαρμόσει ένα μοντέλο ή πολλά διαφορετικά μοντέλα, ανάλογα με το μοντέλο cloud που του παρέχει την καλύτερη λύση. Για παράδειγμα, μια κρίσιμη εφαρμογή που έχει την τήρηση ή άλλη προδιαγραφή ασφαλείας ενδέχεται να απαιτεί ένα hybrid ή private μοντέλο cloud. Αντίθετα, μια γενική εφαρμογή που μπορεί να απαιτείται για ένα προσωρινό έργο μπορεί να είναι ιδανική για ένα public cloud. Είναι σημαντικό επίσης να γνωρίζουμε ότι τα τέσσερα αυτά μοντέλα, δεν καθορίζουν την φυσική θέσης των υποδομών ή των εφαρμογών αλλά την συν-τοποθεσία εγκατάστασης που θα μπορούσε να φιλοξενήσει τόσο ένα public όσο και ένα private cloud.

2.3 Σημαντικότερες εφαρμογές στο cloud computing:

- **Εφαρμογές Γραφείου** (Office Applications)

  Περιλαμβάνονται εργαλεία σουίτας γραφείου (επεξεργασία κειμένου, spreadsheet, project management κτλ), όπως επίσης εργαλεία που επιτρέπουν την online συνεργασία μεταξύ ατόμων ή ομάδων ατόμων (collaboration applications).

- **Εμπορικές Εφαρμογές** (Business Applications)

  Οι εμπορικές εφαρμογές Cloud Computing περιλαμβάνουν κυρίως συστήματα CRM και εφαρμογές ηλεκτρονικού εμπορίου καθώς και ένα πλήθος άλλων εφαρμογών όπως ανάλυση δεδομένων και διαχείριση παγίων. Η συμβατότητα των εφαρμογών μεταξύ τους επιτρέπει την εύκολη ανταλλαγή δεδομένων ενώ εξοικονομείται χρόνος εκπαίδευσης του προσωπικού.

- **Εφαρμογές Κοινωνικής Δικτύωσης** (Social Media Applications)

  Περιλαμβάνονται εφαρμογές κοινωνικής δικτύωσης, ιστοχώροι αποθήκευσης φωτογραφιών καινvideo, ιστολόγια (blogs) κτλ. Σημαντικό πλεονέκτημα αποτελεί η δυνατότητα διεξαγωγής onlineδημοψηφισμάτων και online διαβουλεύσεων σε ζητήματα που αφορούν όλη την κοινωνία όπως νέα νομοσχέδια, διαγωνισμοί κτλ.

Βιβλιογραφία :

1. http://en.wikipedia.org/wiki/Cloud_computing

2. http://www.nist.gov/index.html

3. http://www.ibiblio.org/pioneers/licklider.html

4. Cloud computing principles, Systems and Applications, Nick Antonopoulos, Lee Gillam, Springer

5. Cloud Computing and Software Services Theory and Techniques, Syed A. Ahson, Mohammad Illyas

3. Πλατφόρμα υπολογιστικού νέφους ως υπηρεσία **(PaaS)**

Επειδή το αντικείμενο της εργασίας μας είναι το μοντέλο «πλατφόρμα πολογιστικού νέφους ως υπηρεσία» , σ αυτό το κεφάλαιο γίνεται εκτενής αναφορά στα χαρακτηριστικά του, τις διαφορές του από τα υπόλοιπα μοντέλα καθώς και στα οφέλη από τη χρήση του.

Το μοντέλο «πλατφόρμα υπολογιστικού νέφους ως υπηρεσία» είναι παρόμοιο με το μοντέλο «λογισμικό σύννεφου ως υπηρεσία» , αλλά η υπηρεσία είναι ένα ολόκληρο περιβάλλον ανάπτυξης εφαρμογών, όχι μόνο η χρήση μιας εφαρμογής. Οι λύσεις μοντέλου «πλατφόρμα υπολογιστικού νέφους ως υπηρεσία» διαφέρουν από τις λύσεις μοντέλου «λογισμικό σύννεφου ως υπηρεσία» δεδομένου ότι παρέχουν μια cloud-host εικονική πλατφόρμα ανάπτυξης, προσβάσιμη διαμέσου ενός web browser.

Οι προμηθευτές του μοντέλου «πλατφόρμα υπολογιστικού νέφους ως υπηρεσία» παραδίδουν ταυτόχρονα και την πλατφόρμα computing αλλά και το solution stack ως υπηρεσία. Αυτό επιταχύνει την ανάπτυξη και την επέκταση των εφαρμογών λογισμικού.

Μια computing πλατφόρμα περιλαμβάνει κάποιου είδους αρχιτεκτονικής υλικού και ένα πλαίσιο λογισμικού (συμπεριλαμβανομένων των πλαισίων εφαρμογής).Ο συνδυασμός επιτρέπει την εκτέλεση του λογισμικού. Οι τυπικές πλατφόρμες περιλαμβάνουν την αρχιτεκτονική ενός υπολογιστή, το λειτουργικό σύστημα, γλώσσες προγραμματισμού και περιβάλλον εργασίας χρήστη(user interface) (run-time βιβλιοθήκες συστήματος ή γραφικό περιβάλλον εργασίας(graphical user interface).

Στην επιστήμη των υπολογιστών, ένα solution stack είναι ένα σύνολο υποσυστημάτων λογισμικού ή των στοιχείων που απαιτούνται για να παραδοθεί μια πλήρως λειτουργική λύση, π.χ. ένα προϊόν ή μια υπηρεσία.
Για παράδειγμα, για να αναπτυχθεί μια web εφαρμογή, ο σχεδιαστής πρέπει να χρησιμοποιήσει ένα λειτουργικό σύστημα, ένα web server, μια βάση δεδομένων, και μια γλώσσα προγραμματισμού. Μια άλλη εκδοχή του solution stack είναι ένα λειτουργικό σύστημα, ένα ενδιάμεσο λογισμικό, μια βάση δεδομένων και οι εφαρμογές

Χρησιμοποιώντας την έννοια του μοντέλου «πλατφόρμα υπολογιστικού νέφους ως υπηρεσία» , οι προγραμματιστές λογισμικού

μπορούν να “χτίσουν” Web εφαρμογές χωρίς να χρειάζεται να εγκαθιστούν τα building εργαλεία του λογισμικού στον υπολογιστή τους, και κατόπιν διανέμουν ή επεκτείνουν τις εφαρμογές τους εύκολα στο cloud.

Το μοντέλο «πλατφόρμα υπολογιστικού νέφους ως υπηρεσία» ενσωματώνει ένα επίπεδο του λογισμικού και το παρέχει ως υπηρεσία που μπορεί να χρησιμοποιηθεί για να “χτίσει” υπηρεσίες υψηλότερου επιπέδου.

Επιπλέον το μοντέλο «πλατφόρμα υπολογιστικού νέφους ως υπηρεσία» περιγράφεται ως εξής: Η δυνατότητα που παρέχεται στον καταναλωτή είναι να αναπτύξει άνω στην υποδομή cloud που ο καταναλωτή δημιούργησε ή οι εφαρμογές που δημιουργήθηκαν χρησιμοποιώντας γλώσσες προγραμματισμού και εργαλείων που υποστηρίζονται από τον πάροχο. Ο καταναλωτής δεν έχει τη διαχείριση ή τον έλεγχο της σχετικής υποδομής cloud συμπεριλαμβανομένου του δικτύου, των servers, λειτουργικά συστήματα ή της αποθήκευσης, αλλά έχει τον έλεγχο πάνω από τις χρησιμοποιούμενες εφαρμογές και ενδεχομένως το hosting εφαρμογών του περιβάλλοντος των ρυθμίσεων.

3.1 Υπηρεσίες που παρέχονται από την πλατφόρμα για τους προγραμματιστές

Ο προμηθευτής μοντέλου «πλατφόρμα υπολογιστικού νέφους ως υπηρεσία» παρέχει διάφορες υπηρεσίες για τους προγραμματιστές εφαρμογών :

- Ένα εικονικό περιβάλλον ανάπτυξης ή προδιαγραφές των εφαρμογών, συνήθως βασιζόμενο σε απαιτήσεις των προγραμματιστών
- Εργαλειοθήκες που διαμορφώνεται για τον εικονικό περιβάλλον ανάπτυξης.
- Έναν έτοιμο δίαυλο διανομής για τους δημόσιους προγραμματιστές εφαρμογών.

Το μοντέλο «πλατφόρμα υπολογιστικού νέφους ως υπηρεσία» παρέχει ένα χαμηλό κόστος εισόδου για τους σχεδιαστές εφαρμογών και τους διανομείς τους, με την υποστήριξη του πλήρους κύκλου ζωής ανάπτυξης λογισμικού (SDLC) των Web εφαρμογών. Έτσι με αυτόν τον τρόπο εξαλείφεται η ανάγκη για απόκτηση υλικών και λογισμικών πόρων. Το μοντέλο «πλατφόρμα υπολογιστικού νέφους ως υπηρεσία» μπορεί να περιλαμβάνει μια πλήρη end-to-end εφαρμογή για την ανάπτυξη, τη δοκιμή, και την επέκταση μιας εφαρμογής ή μπορεί να είναι μια μικρότερη, πιό εξειδικευμένη πρόταση, εστιάζοντας σε μια συγκεκριμένη περιοχή όπως την διαχείριση του περιεχομένου .

3.2 Απαραίτητα στοιχεία για να θεωρείται μια πλατφόρμα πραγματική λύση

Για να θεωρείται μια πλατφόρμα ανάπτυξης λογισμικού μοντέλου «πλατφόρμα υπολογιστικού νέφους ως υπηρεσία» πραγματική λύση, πρέπει να υφίστανται βασικά στοιχεία όπως:

1. Ο βασικός έλεγχος τις χρήσεις των εφαρμογών θα πρέπει να χρησιμοποιείται για την βελτίωση της διαδικασίας των πλατφορμών .

2. η λύση θα πρέπει να παρέχει απόλυτη ενσωμάτωση με άλλους πόρους του cloud,όπως οι βασισμένες στο WEB βάσεις δεδομένων και άλλες βασισμένες στο WEB υπηρεσίες και στοιχειά υποδομών.

3. Η δυναμική πολλαπλή-μίσθωση θα πρέπει να είναι επιτεύξιμη καθώς επίσης και η συνεργασία μέσω του cloud μεταξύ των προγραμματιστών, των πελατών, και των χρηστών σε όλο το SDLC.

4. Η ασφάλεια, η μυστικότητα, και η αξιοπιστία πρέπει να είναι βασική υπηρεσία.

5. η πλατφόρμα ανάπτυξης θα πρέπει να είναι browser-based.

3.3 Οφέλη από τη χρήση του μοντέλου «πλατφόρμα υπολογιστικού νέφους ως υπηρεσία»

Η δημιουργία ενός έτοιμου καναλιού για πωλήσεις και διανομή είναι επίσης ένα όφελος του μοντέλου «πλατφόρμα υπολογιστικού νέφους ως υπηρεσία» . Οι μικροί ή οι νέοι προγραμματιστές λογισμικού

μπορούν να χρησιμοποιήσουν έναν προμηθευτή μοντέλου «πλατφόρμα υπολογιστικού νέφους ως υπηρεσία»

για να έχουν πρόσβαση σε πηγές ανάπτυξης που σε άλλη περίπτωση θα τους ήταν μη διαθέσιμες.

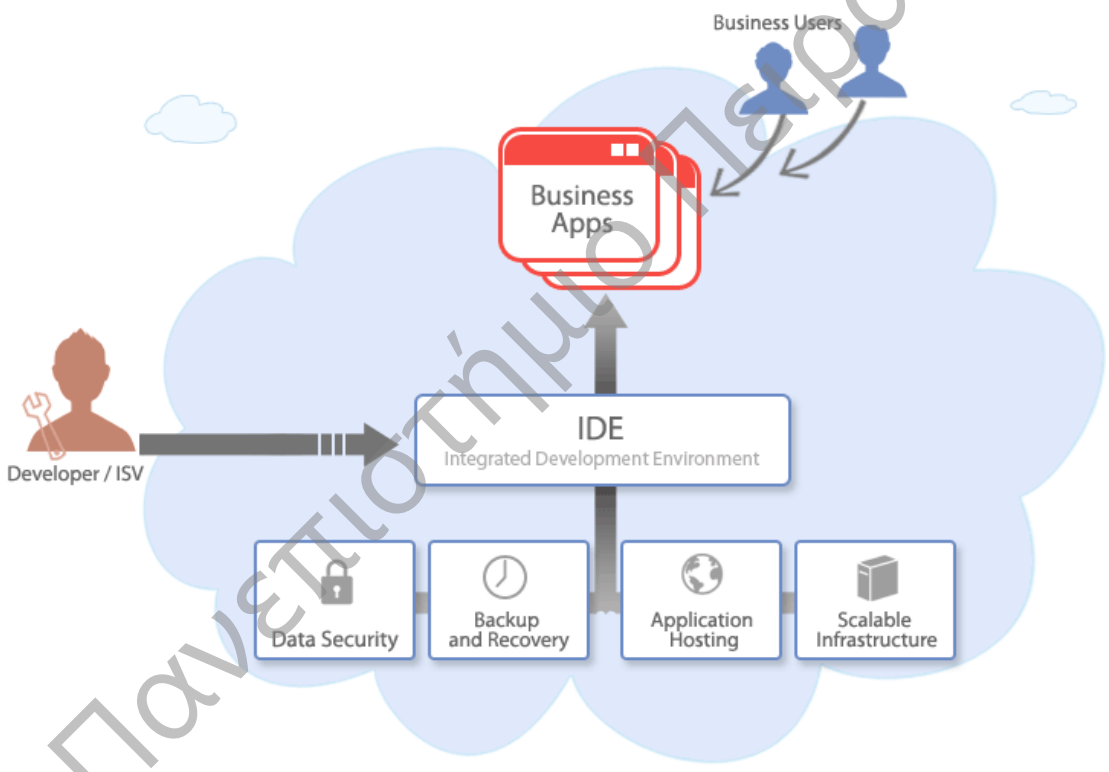
Οι διάφοροι τύποι προμηθευτών μοντέλου «πλατφόρμα υπολογιστικού νέφους ως υπηρεσία» μπορούν να είναι εκτενής έτσι ώστε να μπορούν να εμπεριέχουν μια πλήρη "φιλοξενία" εφαρμογής, την ανάπτυξη, την δοκιμή και το περιβάλλον επέκτασης καθώς επίσης και τις εκτενείς ενσωματωμένες υπηρεσίες που περιλαμβάνουν την εξέλιξη και τη συντήρηση.

Ο κατάλογος προμηθευτών μοντέλου «πλατφόρμα υπολογιστικού νέφους ως υπηρεσία» δεν είναι τόσο εκτενής όσο των μοντέλων «λογισμικό σύννεφου ως υπηρεσία», κυρίως επειδή το μοντέλο «πλατφόρμα υπολογιστικού νέφους ως υπηρεσία» στοχεύει σε μια μικρότερη αγορά, στους προγραμματιστές παρά στους τελικούς χρήστες. Αλλά μερικοί προμηθευτές μοντέλου «λογισμικό σύννεφου ως υπηρεσία» έχουν αρχίσει να παρουσιάζουν τις υπηρεσίες μοντέλου «πλατφόρμα υπολογιστικού νέφους ως υπηρεσία» ως λογική επέκταση των υπηρεσιών μοντέλου «λογισμικό σύννεφου ως υπηρεσία» τους. Όπως για παράδειγμα το salesforce.com έχει αρχίσει μια υπηρεσία μοντέλου «πλατφόρμα υπολογιστικού νέφους ως υπηρεσία» στο force.com.

Οι Amazon Web Services έχει καθιερώσει τις υπηρεσίες μοντέλου «πλατφόρμα υπολογιστικού νέφους ως υπηρεσία» για τους προγραμματιστές, κατά ένα μεγάλο μέρος μέσω της ολοκλήρωσης και του συνεταιρισμού με την Amazon Web Services, για να παρέχει τις πλατφόρμες ανάπτυξης πάνω από το Amazon Web Services. Παραδείγματος χάριν, η Pegasystems Inc, είναι ένας προμηθευτής διαχείρισης επιχειρηματικών διαδικασιών (business process management) λύσεων λογισμικού, προσφέρει την πλατφόρμα SmartPass ως υπηρεσία που τρέχει πανω σε Amazon Web Services. Ένα άλλο παράδειγμα μιας υπηρεσίας μοντέλου «πλατφόρμα υπολογιστικού νέφους ως υπηρεσία» περιλαμβάνει την Google App Engine,η οποία εξυπηρετεί τις εφαρμογές στην Google υποδομή.

Επιπλέον, κάποιος που παράγει το μοντέλο «πλατφόρμα υπολογιστικού νέφους ως υπηρεσία» μπορεί να παράγει μια πλατφόρμα με την ενσωμάτωση ενός λειτουργικού συστήματος, του υλικού λογισμικού, του λογισμικού των εφαρμογών, ακόμη και ενός περιβάλλοντος ανάπτυξης που έπειτα παρέχεται στον πελάτη ως υπηρεσία.

Επίσης, κάποιος που χρησιμοποιεί το μοντέλο «πλατφόρμα υπολογιστικού νέφους ως υπηρεσία» θα έβλεπε μια ενσωματωμένη υπηρεσία που παρουσιάζεται σε αυτόν μέσω μιας διασύνδεσης προγραμματισμού εφαρμογών (application programming interface) .Ο πελάτης αλληλεπιδρά με την πλατφόρμα μέσω του application programming interface και η πλατφόρμα κάνει αυτό που είναι απαραίτητο, για να διαχειρίζεται, να διαβαθμίζει από μόνος του και για να παρέχει ένα δεδομένο επίπεδο υπηρεσίας.



Εικόνα 3.1- Γενική περιγραφή του προτύπου PaaS

Βιβλιογραφία :

1. http://en.wikipedia.org/wiki/Cloud_computing

2. http://www.gartner.com/it-glossary/platform-as-a-service-paas

3. The NIST Definition of Cloud Computing Recommendations of the National Institute of Standards and Technology. Peter Mell Timothy Grance

4. Introduction to Cloud Computing Architecture White Paper 1st Edition, June 2009 Sun Microsystems, Inc.

4. Θέματα ασφάλειας στο Cloud

Η ασφάλεια έχει αναδειχθεί ως αναμφισβήτητα το πιο σημαντικό εμπόδιο για την υιοθέτηση του Cloud Computing. Η κοινή χρήση των πόρων σημαίνει ότι η δραστηριότητα ενός χρήστη cloud μπορεί να είναι ορατή και σε άλλους χρήστες cloud, που χρησιμοποιούν τους ίδιους πόρους και ενδέχεται μάλιστα να περιλαμβάνουν εμπιστευτικές πληροφορίες των επιχειρήσεων. Ορισμένοι από τους συμμετέχοντες μπορεί να είναι υπονομευτές, οι οποίοι διατηρούν την εμφάνιση ενός τακτικού χρήστη cloud, αλλά στην πραγματικότητα κάνουν επιθέσεις στο Διαδίκτυο. Ανταγωνιστικές επιχειρήσεις μέσα στο ίδιο Cloud Computing, είτε χρησιμοποιώντας το ίδιο cloud, είτε καταλήγοντας σε μια σχέση παρόχου –χρήστη. Αυτό μπορεί να οδηγήσει σε ισχυρές συγκρούσεις συμφερόντων και να δημιουργήσει πρόσθετα κίνητρα, για πρόσβαση σε εμπιστευτικές πληροφορίες ενός ανταγωνιστή.

Η κρυπτογράφηση θα μπορούσε να είναι μια λύση για την ασφάλεια των δεδομένων. Μπορούν να γίνουν υπολογισμοί σε κρυπτογραφημένα δεδομένα, αλλά το κόστος υπολογισμού είναι μέχρι σήμερα απαγορευτικό. Δεν υπάρχουν δυνατότητες αναζήτησης σε κρυπτογραφημένα δεδομένα, εκτός και αν ο πάροχος έχει το κλειδί.

4.1 Ζητήματα ασφάλειας που πρέπει να ληφθούν υπ όψη

Παρακάτω παρουσιάζονται επτά ζητήματα ασφαλείας που πρέπει να ληφθούν σοβαρά υπόψη όσον αφορά τα συστήματα Cloud Computing :

- Προνομιακή πρόσβαση των χρηστών (Privileged user access), έχει να κάνει με το ποιος έχει πρόσβαση στα δεδομένα και για τη μίσθωση και διαχείριση ανάλογων διαχειριστών.

- Κανονιστική Συμμόρφωση (Regulatory compliance). Αν ο προμηθευτής είναι πρόθυμος να υποβληθεί σε εξωτερικούς ελέγχους ή και πιστοποιήσεις ασφαλείας

- Θέση δεδομένων (Data location) . Αν ο προμηθευτής επιτρέπει οποιονδήποτε έλεγχο στην τοποθεσία των δεδομένων

- Διαχωρισμός δεδομένων (Data segregation) . Αν υπάρχει διαθέσιμη κρυπτογράφηση σε όλα τα στάδια και ότι τα σχέδια κρυπτογράφησης σχεδιάστηκαν και εξετάστηκαν από πεπειραμένους επαγγελματίες.

- Ανάκτηση (Recovery) . Τι θα συμβεί στα δεδομένα σε περίπτωση καταστροφής, εάν προσφέρουν πλήρη αποκατάσταση και τη χρονική διάρκεια αποκατάστασης

- Ερευνητική υποστήριξη (Investigative support) . Αν οι πωλητές έχουν τη δυνατότητα να διερευνήσουν οποιαδήποτε ανάρμοστη ή παράνομη δραστηριότητα

- Μακροπρόθεσμη βιωσιμότητα (Long- term viability) . Αν ο πάροχος κλείσει , πώς θα επιστραφούν τα δεδομένα και σε τι μορφή.

Με το μοντέλο Cloud οι επιχειρήσεις χάνουν τον έλεγχο της φυσικής ασφάλειας. Οι περισσότερες επιχειρήσεις δεν γνωρίζουν αν τα δεδομένα τους είναι κρυπτογραφημένα ή όχι, ούτε ποια μέθοδος κρυπτογράφησης έχει εφαρμοστεί. Ακόμη, αν τα δεδομένα έχουν κρυπτογραφηθεί, δεν γνωρίζουν ποιος ελέγχει τα κλειδιά κρυπτογράφησης-αποκρυπτογράφησης. Η χρήση του **SSL** (Secure Sockets Layer) θα μπορούσε να αποτελέσει μια αξιόπιστη λύση. Το SSL είναι μια τεχνολογία κρυπτογράφησης που προστατεύει τις ιδιωτικές πληροφορίες των επισκεπτών , ενώ είναι υπό μεταφορά μέσω του Διαδικτύου. Πρέπει επίσης να εξασφαλιστεί η ακεραιότητα των δεδομένων, δηλαδή να γίνονται αλλαγές στα δεδομένα μόνο σε απάντηση στις εξουσιοδοτημένες συναλλαγές.

Η υπηρεσία αποθήκευσης δεδομένων παρέχει αυτοματοποιημένη αναπαραγωγή και διατήρηση δεδομένων, κρυπτογράφηση όταν τα δεδομένα βρίσκονται σε κατάσταση αναμονής (at rest) και κατά τη μεταφορά τους, κατακερματισμό και διασπορά δεδομένων. Στόχος είναι να δοθεί η δυνατότητα στους χρήστες να μπορούν να αξιοποιήσουν τα αποθηκευμένα δεδομένα, όπως να τα εντοπίσουν , να τα μεταφέρουν, να δουλέψουν με αυτά, να έχουν πρόσβαση σε αποθήκες δεδομένων και τέλος να μπορούν να τα αποθηκεύουν με ασφάλεια.

Τις περισσότερες φορές , ο πιο αδύναμος κρίκος στην αλυσίδα της ασφάλειας των συστημάτων Cloud δεν είναι το λειτουργικό σύστημα ή οι όποιες εφαρμογές προστασίας, αλλά ο ίδιος ο χρήστης- άνθρωπος. Η γνώση είναι το αποτελεσματικότερο εργαλείο για τη διαχείριση των κινδύνων. Οι πάροχοι των Clouds πρέπει να παρακολουθούν τα συστήματα συνεχώς, να αντιμετωπίζουν το οποιοδήποτε περιστατικό και

να προσδιορίζουν συγκεκριμένες απειλές μετά από σχεδιασμό και υλοποίηση συγκεκριμένων ελέγχων.

4.2 Πλεονεκτήματα όσον αφορά την ασφάλεια δεδομένων

Τα πλεονεκτήματα όσον αφορά την ασφάλεια των δεδομένων στα συστήματα Cloud Computing είναι τα εξής :

- Η μετατόπιση των δημόσιων δεδομένων σε ένα εξωτερικό Cloud μειώνει την έκθεση των εσωτερικών ευαίσθητων δεδομένων
- Η ομοιογένεια του Cloud κάνει την ασφάλεια ελέγχου- δοκιμών απλούστερη
- Τα Clouds επιτρέπουν την αυτοματοποιημένη διαχείριση της ασφάλειας
- Πλεονάζον προσωπικό για την αποκατάσταση από πιθανές καταστροφές

4.3 Προκλήσεις που αφορούν την ασφάλεια δεδομένων

Προκλήσεις που αφορούν την ασφάλεια των δεδομένων στα συστήματα Cloud Computing:

- Να εμπιστευθεί το πρότυπο ασφάλειας του προμηθευτή
- Αδυναμία πελατών να ανταποκριθούν στις διαπιστώσεις του ελέγχου
- Λήψη υποστήριξης για τις έρευνες
- Έμμεση ευθύνη διαχειριστή
- Απώλεια φυσικού ελέγχου
- Αποκλειστικές εφαρμογές δεν είναι δυνατόν να εξεταστούν
- Έλξη στους χάκερ (κάτι τόσο εμπορικό και επαναστατικό είναι λογικό να τραβάει τους χάκερ).

Η μεταφορά στο Cloud μας προσφέρει εξοικονόμηση κόστους και ενέργειας και αυξημένη ευελιξία όσον αφορά την εγκατάσταση λογισμικού. Τα θέματα ασφάλειας των clouds μπορούν να καθορίσουν το πώς θα υιοθετηθούν και θα αναπτυχθούν λύσεις για το Cloud Computing.

Τα ιδιωτικά (private) clouds μπορεί να έχουν μικρότερη έκθεση απειλής από τα clouds κοινότητας (community), τα οποία με τη σειρά τους έχουν μικρότερη έκθεση απειλής από τα δημόσια (public) clouds. Τα ογκώδη public clouds μπορεί να είναι πιο αποδοτικά από τα μεγάλα community clouds, τα οποία μπορεί να είναι πιο αποδοτικά από τα μικρά private clouds.

Τα Clouds έχουν συνήθως μια ενιαία αρχιτεκτονική ασφάλειας, αλλά έχουν πολλούς πελάτες με διαφορετικές ανάγκες. Θα πρέπει λοιπόν να προσπαθήσουν να παρέχουν όσο το δυνατόν περισσότερο διαμορφώσιμους μηχανισμούς ασφάλειας. Οι οργανισμοί έχουν περισσότερο έλεγχο της αρχιτεκτονικής ασφάλειας των ιδιωτικών clouds που ακολουθούνται από τα clouds κοινότητας και στη συνέχεια τα δημόσια. Αυτό όμως δεν σημαίνει τίποτα για την πραγματική ασφάλεια. Αυτό συμβαίνει γιατί, τα ευαίσθητα δεδομένα είναι πιθανό να υποβληθούν σε επεξεργασία στα clouds, όπου οι οργανισμοί έχουν τον έλεγχο του μοντέλου ασφάλειας. Τα περισσότερα Clouds απαιτούν πολύ ισχυρούς ελέγχους ασφάλειας και όλα τα μοντέλα μπορούν να χρησιμοποιηθούν για διάφορες ανταλλαγές μεταξύ της έκθεσης απειλών και της αποτελεσματικότητας. Όπως αναφέρθηκε και προηγουμένως δεν υπάρχει ένα Cloud, υπάρχουν πολλά μοντέλα και αρχιτεκτονικές.

Βιβλιογραφία:

1. J. Rittinghouse, J. Ransome, " Cloud Computing Implementation, Management and Security

2. T. Mather, S. Kumaraswamy, S. Latif, " Cloud Security and Privacy.

3. http://www.itsecuritypro.gr/contents_article.php?id=93&catid=4

4. http://www.excelixi.org/el/Knowledge-Base/e-Business/Poso_Asfali_Einai_ta_Dedomena_Mias_Epixeirisis_Meso_Cloud_Computing

5. Υλοποίηση συστήματος κρατήσεων δωματίων του Relax Hotel

Σκοπός της παρούσας εργασίας είναι η δημιουργία ενός συστήματος κρατήσεων δωματίων του Relax Hotel που θα διευκολύνει τόσο τον ιδιοκτήτη της ξενοδοχειακής μονάδας, όσο και τους πελάτες του. Έτσι η διαδικασία κράτησης ενός δωματίου θα γίνει πιο γρήγορη και ευέλικτη και θα αφήσει ικανοποιημένους και τους πιο απαιτητικούς χρήστες.

5.1 Έγγραφο προσδιορισμού απαιτήσεων συστήματος

5.1.1 Ορισμός Προβλήματος

Αφορμή για τη δημιουργία του συστήματος που μελετάμε στην εργασία μας στάθηκε το γεγονός ότι, το ξενοδοχείο Relax hotel, του οποίου οι εγκαταστάσεις στεγάζονται στην Κρήτη δεν διέθετε πληροφοριακό σύστημα για online κρατήσεις. Στις μέρες μας, η πρόοδος της τεχνολογίας και με δεδομένο ότι ο κλάδος της Πληροφορικής, έχει μπει για τα καλά στη ζωή μας, θεωρείται αδιανόητο για μια επιχείρηση ή οργανισμό να επιβιώσει αν δεν μπορεί να εξυπηρετήσει τους πελάτες της μέσω του Διαδικτύου. Επομένως χάνει μεγάλο μέρος των κερδών της, δεν μπορεί να εξυπηρετήσει τους σκοπούς της, δεν μπορεί να ανταγωνιστεί επιχειρήσεις που ανήκουν στον ίδιο κλάδο με αυτή και τελικά χάνεται.

5.1.2 Αιτιολόγηση Συστήματος

Το σύστημα αυτό κρίνεται αναγκαίο τόσο για την ίδια την επιχείρηση όσο και για τους πελάτες της. Αφενός η πρώτη δεν μπορεί να επιβιώσει, πόσο μάλλον σε μια εποχή οικονομικής κρίσης που έχει εξαπλωθεί γενικά στην αγορά της χώρας μας. Αφετέρου οι πελάτες

δυσκολεύονται να επιλέξουν το ξενοδοχείο, μιας και δεν υπάρχει τρόπος διαφήμισης που να έχει τόσο μεγάλη απήχηση στο καταναλωτικό κοινό, όπως είναι η διαφήμιση μέσω του Διαδικτύου. Καλή η τηλεόραση και το ραδιόφωνο αλλά, πλέον το Διαδίκτυο έχει κατακτήσει τα πάντα. Όλοι είμαστε χρήστες, για να ενημερωθούμε, να ψυχαγωγηθούμε και να συλλέξουμε πληροφορίες για πράγματα που μας ενδιαφέρουν, ή για να εκτελούμε με μεγαλύτερη άνεση τις υποχρεώσεις μας. Επομένως με τη δημιουργία σελίδας στο Διαδίκτυο, που θα περιέχει όλες τις απαραίτητες πληροφορίες σχετικά με το ξενοδοχείο, ο πελάτης θα μπορεί εύκολα και γρήγορα από την άνεση του σπιτιού του, να δει αν τον ενδιαφέρει το ξενοδοχείο και καλύπτει τις προσδοκίες του. Επομένως το ξενοδοχείο γίνεται ευρέως γνωστό και μπορεί και αυτό να προσελκύσει περισσότερους πελάτες και ενδεχομένως να βελτιώσει και τη λειτουργία του.

5.1.3 Σκοπός του Συστήματος και του έργου

Σκοπός του Συστήματος και του έργου είναι, οι πελάτες του ξενοδοχείου, να μπορούν, αφού συλλέξουν όλες τις πληροφορίες σχετικά με αυτό, να κάνουν κράτηση δωματίων. Αυτό έλειπε από το ξενοδοχείο και το έκανε να μειονεκτεί σε σχέση με τους ανταγωνιστές του, οι οποίοι διέθεταν ολοκληρωμένο Πληροφοριακό Σύστημα για online κρατήσεις δωματίων. Στο ξενοδοχείο που μελετάμε όλες οι κρατήσεις δωματίων γινόταν τηλεφωνικά. Αυτό δημιουργούσε πολλά προβλήματα. Πολλές φορές χανόταν το σήμα, δεν γινόταν ακριβής επεξήγηση των υπηρεσιών που παρείχε το ξενοδοχείο, με συνέπεια οι πελάτες να μην το προτιμούν. Επομένως δημιουργούνταν προβλήματα στην ομαλή λειτουργία του ξενοδοχείου, έχανε μεγάλο μέρος πελατών και τα κέρδη του ήταν μειωμένα. Τώρα με την κατασκευή τις ιστοσελίδας ο πελάτης θα βλέπει από τον υπολογιστή του, ή από το κινητό του τις υπηρεσίες που του προσφέρει το ξενοδοχείο , τη διαθεσιμότητα των δωματίων, την περίοδο που θα επιλέξει να το επισκεφτεί και μπορεί να κρίνει καλύτερα τι τον συμφέρει περισσότερο. Επομένως το σύστημά μας αυτοματοποιεί τη διαδικασία κράτησης δωματίων, η οποία μέχρι τώρα γινόταν τηλεφωνικά και ήταν πονοκέφαλος τόσο για τους πελάτες όσο και για τους υπαλλήλους. Οι μεν πελάτες δεν καταλάβαιναν ακριβώς τις υπηρεσίες που τους εξηγούσαν οι υπάλληλοι, οι δε υπάλληλοι έχαναν πολλή ώρα προσπαθώντας να εξηγήσουν τις υπηρεσίες στον πελάτη. Επιπλέον

πολλοί πελάτες έπαιρναν τηλέφωνο να ενημερωθούν και δεν έκαναν ποτέ κρατήσεις, άρα τα έσοδα του ξενοδοχείου ήταν προφανώς μειωμένα.

5.1.4 Περιορισμοί του Συστήματος και του έργου

Το σύστημά μας θα πρέπει να έχει υψηλό βαθμό ασφάλειας για να προστατέψει τα προσωπικά δεδομένα των πελατών του, οι οποίοι θα κάνουν online κρατήσεις δωματίων. Επειδή ο χώρος του Διαδικτύου, όσες ευκολίες και ανέσεις μας παρέχει, άλλο τόσο επικίνδυνος είναι, πρέπει η ασφάλεια του συστήματός μας να είναι τέτοια ώστε να προστατεύονται στοιχεία όπως το όνομα και το τηλέφωνο του πελάτη, ή η διεύθυνση κατοικίας του. Ακόμα αν κάποιος πελάτης επιθυμεί να πληρώσει με κάρτα, πρέπει να υπάρχει ειδική μέριμνα από τους κατασκευαστές του συστήματος , να προστατευτεί και αυτός ο τύπος πληρωμής.

5.1.5 Λειτουργίες ανά συνιστώσα του Συστήματος

Υλικό

Λόγω του μεγάλου όγκου των δεδομένων που έχει να διαχειριστεί το σύστημά μας και των πολλών παράλληλων συνδέσεων στο σύστημα, αφού πολλοί πελάτες θα είναι ταυτόχρονα συνδεδεμένοι για να μπορούν να κάνουν κράτηση, το σύστημα θα υλοποιηθεί σε τεχνολογία Cloud Computing και συγκεκριμένα σε IaaS (Infrastructure as a Service).

Λογισμικό

Το λογισμικό θα υλοποιηθεί σε SaaS (Software as a Service)

Άνθρωποι

Οι άνθρωποι(χρήστες) του συστήματός μας είναι εργαζόμενοι σε επιχειρήσεις, δημόσιες η ιδιωτικές, καθώς και υπάλληλοι του ξενοδοχείου οι οποίοι θα βλέπουν καθημερινά, αν υπάρχουν ανανεώσεις σχετικά με τις κρατήσεις, αν υπάρχουν διαθέσιμα δωμάτια και γενικά θα είναι υπεύθυνοι για την ομαλή λειτουργία του συστήματος και κατά συνέπεια ολόκληρου του ξενοδοχείου.

5.1.6 Χαρακτηριστικά χρηστών

Οι υπάλληλοι του ξενοδοχείου δεν είναι ιδιαίτερα εξοικειωμένοι με τις νέες τεχνολογίες, επομένως το σύστημα θα πρέπει να είναι εύκολο, γρήγορο και απλό στη χρήση του για να διευκολύνεται τόσο ο υπάλληλος όσο και ολόκληρη η λειτουργία του ξενοδοχείου. Επειδή οι κρατήσεις μέχρι τώρα γινόταν τηλεφωνικά πρέπει στην αρχή να υπάρχει κάποια καθοδήγηση από τους κατασκευαστές του συστήματος προς τους υπαλλήλους. Μάλιστα σε αυτό το σημείο πρέπει να αντιμετωπιστεί και η δυσπιστία των υπαλλήλων που δείχνουν πάντα αρνητική προδιάθεση, τουλάχιστον στο ξεκίνημα, προς κάθε τι καινούριο και καινοτόμο

5.1.7 Περιβάλλοντα

Ανάπτυξης

Οι γλώσσες προγραμματισμού που μπορούν να χρησιμοποιηθούν είναι JAVA, PHP, C#. Λόγω περιορισμένου χρόνου και έλλειψης εξειδίκευσης θα προτιμηθεί η PHP.

Λειτουργίας

Είσοδος:

- κατηγορία δωματίου(μονό, δίκλινο, τρίκλινο)
- αριθμός δωματίων(ένας πελάτης μπορεί να κλείσει πολλά δωμάτια)
- ημερομηνία κράτησης

Επεξεργασία:

- Έλεγχος διαθεσιμότητας δωματίων από τον υπάλληλο του ξενοδοχείου

Έξοδος:

- Επιβεβαίωση κράτησης δωματίων

Πόροι:

- Λογισμικό(SaaS)
- Υλικό(IaaS)
- Δίκτυο(IaaS)

Εξαρτήσεις- επικοινωνία με άλλα ΠΣ:

- Επικοινωνία με τραπεζικά συστήματα για τις συναλλαγές(ανάλογα με τον τρόπο πληρωμής που επιθυμεί ο πελάτης)
- Επικοινωνία με ΠΣ άλλων επιχειρήσεων (ανταγωνιστών ή μη)

Εμπλεκόμενοι:

- Πελάτες
- Επιχειρήσεις
- Υπάλληλοι του ξενοδοχείου

Συντήρησης

Η συντήρηση θα πρέπει να γίνεται σε τοπικό επίπεδο για να μην επηρεάζεται η συνολική λειτουργία του συστήματος, διότι ένα πρόβλημα στο σύστημα κρατήσεων μπορεί να επηρεάσει αρνητικά τη συνολική λειτουργία του ξενοδοχείου. Βεβαίως η συντήρηση του συστήματος θα πρέπει να γίνεται από ειδικά εκπαιδευμένους τεχνικούς για να ελαχιστοποιηθεί η πιθανότητα λάθους.

5.1.8 Στρατηγική Λύσης του Συστήματος

Το σύστημα θα υλοποιηθεί σε αρχιτεκτονική cloud , την οποία χρησιμοποιούν πολλές παρόμοιες επιχειρήσεις, για να μην υπάρξει πρόβλημα διαθεσιμότητας, επειδή θα υπάρχουν πολλοί χρήστες, που θα θέλουν να κάνουν online κράτηση ταυτόχρονα. Επομένως δεν θα πρέπει να υπάρχουν καθυστερήσεις για να μην χάνει και ο πελάτης την υπομονή του και εγκαταλείπει τη διαδικασία κράτησης δωματίων στο ξενοδοχείο, κάτι που έχει αρνητικό αντίκτυπο στην εικόνα του προς τους πελάτες αλλά και στα κέρδη της επιχείρησης.

5.1.9 Προτεραιότητες στα Χαρακτηριστικά του Συστήματος

Το σύστημα θα πρέπει να είναι διαθέσιμο σε πολλούς πελάτες ταυτόχρονα που θα επισκέπτονται την ιστοσελίδα του, προκειμένου να κάνουν κράτηση. Επίσης θα πρέπει να είναι αξιόπιστο για να αποφεύγονται οι παρεξηγήσεις και να μην χάνονται πελάτες. Επιπλέον

θα πρέπει να είναι ασφαλές, γιατί οι πελάτες θα δίνουν τα προσωπικά τους δεδομένα, για να κάνουν κράτηση. Άρα τα προσωπικά δεδομένα των πελατών πρέπει να προστατεύονται αυστηρά. Τέλος θα πρέπει να είναι γρήγορο και αποτελεσματικό για να μένουν ικανοποιημένοι και οι πελάτες και το ξενοδοχείο να καλύπτει τις φιλοδοξίες του και το σκοπό για τον οποίο δημιουργήθηκε.

5.1.10 Κριτήρια Αποδοχής του Συστήματος

Για να γίνει αποδεκτό το σύστημά μας τόσο στους πελάτες όσο και στους χρήστες- υπαλλήλους του ξενοδοχείου θα πρέπει να είναι έτοιμο σύμφωνα με το αρχικό χρονοδιάγραμμα, να μην ξεπεράσει τον προϋπολογισμό που διατέθηκε γι' αυτό, να είναι αξιόπιστο, να είναι αποτελεσματικό και γρήγορο. Έτσι θα αυτοματοποιεί τη διαδικασία κράτησης δωματίων, η οποία πλέον θα γίνεται ηλεκτρονικά και θα καλύπτει τους στόχους των πελατών και του ξενοδοχείου γενικότερα. Επιπλέον θα βελτιώσει συνολικά τη διαδικασία κράτησης δωματίων καθότι ο πελάτης θα γλυτώνει την αναμονή που υπήρχε στις τηλεφωνικές γραμμές, θα αποφεύγονται μπερδέματα και ο χρόνος που θα χρειάζεται ο πελάτης για να κλείσει ένα δωμάτιο θα μειωθεί αισθητά.

5.1.11 Πηγές Πληροφοριών

Για την συλλογή των πληροφοριών μοιράστηκαν σε τυχαίους πελάτες ερωτηματολόγια με διάφορες ερωτήσεις σχετικά με τη λειτουργία του ξενοδοχείου, στο οποίο θα γίνει η δημιουργία του συστήματος που μελετάμε . Ζητήθηκε η γνώμη τους, για το αν είναι ευχαριστημένοι από τη μέχρι τώρα λειτουργία του ξενοδοχείου, αν είναι ευχαριστημένοι με τη διαδικασία κράτησης δωματίων, έτσι όπως γινόταν μέχρι τώρα και τους ζητήθηκε να μας πουν τι θα ήθελαν από ένα ολοκληρωμένο ΠΣ, που θα προχωρούσε στην εκτέλεση της διαδικασίας κράτησης δωματίων, ηλεκτρονικά.

5.1.12 Λεξιλόγιο

Στο λεξιλόγιο περιλαμβάνεται ορολογία για συντομογραφίες και ειδικευμένους όρους που χρησιμοποιούνται στο έγγραφο προσδιορισμού των απαιτήσεων του συστήματος.

- Cloud= Cloud Computing(Υπολογιστικό νέφος)
- DB= Data Base(Βάση Δεδομένων)
- IaaS= Infrastructure as a Service
- SaaS= Software as a Service
- PHP= Hypertext Preprocessor

5.2 Διαδικασία που ακολουθεί ο χρήστης για να κάνει κράτηση

Η διαδικασία αυτή που ακολουθεί ο χρήστης από την στιγμή της επίσκεψής του στην αρχική σελίδα του συστήματός μας μέχρι και την κατοχύρωση της κράτησης του αριθμού και του είδους των δωματίων που επιθυμεί, μπορεί να χωριστεί στις παρακάτω 15 υποδιαδικασίες , όπως φαίνεται και στον πίνακα που ακολουθεί.

ID	Περιγραφή
1	Είσοδος πελάτη στην ηλεκτρονική σελίδα του ξενοδοχείου
2	Επιλογή της καρτέλας Τιμές και κρατήσεις
3	Εμφάνιση στο πελάτη της λίστας με τις κατηγορίες δωματίων και των αντίστοιχων τιμών
4	Επιλογή κατηγορίας δωματίου από το πελάτη
5	Επιλογής ημερομηνίας άφιξης και αναχώρησης για την επιλεγμένη κατηγορία δωματίου
6	Αναζήτηση των επιλογών του πελάτη στη βάση δεδομένων του ξενοδοχείου
7	Εμφάνιση αποτελεσμάτων αναζήτησης
8	Μη διαθέσιμο δωμάτιο σύμφωνα με τις επιλογές του πελάτη και επιστροφή στην αρχική σελίδα
9	Ύπαρξη διαθέσιμου δωματίου
10	Εμφάνιση φόρμας συμπλήρωσης των προσωπικών στοιχείων του πελάτη
11	Καταγραφή των προσωπικών στοιχείων του πελάτη στη φόρμα συμπλήρωσης
12	Επιβεβαίωση και ολοκλήρωση κράτησης δωματίου
13	Αποθήκευση των δεδομένων της κράτησης στη βάση δεδομένων του ξενοδοχείου
14	Ενημέρωση της βάσης δεδομένων σύμφωνα με τα δεδομένα της νέας κράτησης
15	Τερματισμός της διαδικασίας κράτησης δωματίου και επιστροφή στην αρχική σελίδα

Πίνακας 1

Με τη βοήθεια του πίνακα αναλύεται επακριβώς η διαδικασία που ακολουθείται από τους χρήστες για να κάνουν κράτηση δωματίου. Έτσι γίνεται εύκολα κατανοητή και η αρχιτεκτονική του συστήματός μας , ο τρόπος με τον οποίο θα σχεδιασθεί η βάση δεδομένων και ο τρόπος που θα υλοποιηθεί η εφαρμογή μας.

5.3 Φυσικός Σχεδιασμός και Σχεδιασμός Βάσης Δεδομένων

Όπως και στα περισσότερα συστήματα, που περιέχουν σχεδιασμό μιας βάσης δεδομένων με διάφορους πίνακες, έτσι και στην περίπτωσή μας οι τρεις κυριότεροι παράγοντες που επηρεάζουν το σχεδιασμό της βάσης του ξενοδοχείου μας, είναι οι εξής:

- Σύστημα
- Λειτουργία
- Ανάπτυξη

Όσον αφορά τον πρώτο παράγοντα, δηλαδή το σύστημα, αυτό που πρέπει να προσεχτεί είναι οι απαιτούμενες λειτουργίες του οργανισμού μας να υποστηρίζονται από τη βάση δεδομένων που θα δημιουργήσουμε και το επίπεδο απόδοσης της βάσης να είναι υψηλό, ώστε να εξυπηρετεί τον σκοπό για τον οποίο δημιουργήθηκε. Επιπλέον η συχνότητα επεξεργασίας των δεδομένων παίζει πολύ μεγάλο ρόλο, γι' αυτό και η βάση πρέπει να είναι όσο το δυνατόν πιο <<προσιτή>> στο διαχειριστή και να μην τον δυσκολεύει. Τέλος πρέπει να εξυπηρετεί και τους χρήστες με την έννοια να προστατεύει τα δεδομένα των πελατών του ξενοδοχείου.

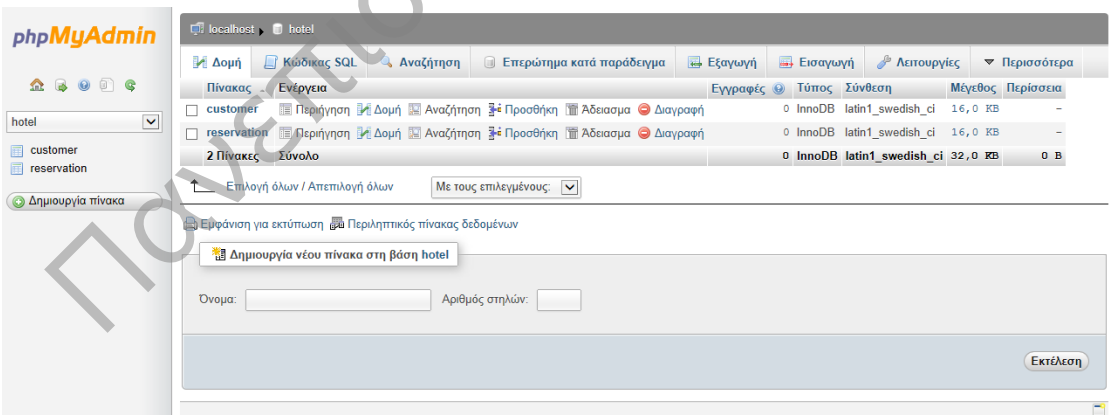
Στον παράγοντα λειτουργία αυτό που πρέπει να προσεχτεί ιδιαίτερα, είναι το κόστος λειτουργίας της βάσης, το οποίο θα πρέπει να συμπεριλαμβάνεται στον προϋπολογισμό του έργου. Σε καμία περίπτωση δεν πρέπει να υπερβαίνει τον αριθμό που δύναται να καλύψει το ξενοδοχείο, διότι σε αυτή την περίπτωση η επιχείρηση κινδυνεύει με κατάρρευση. Ακόμη οι απαιτήσεις αξιοπιστίας της βάσης παίζουν πολύ μεγάλο ρόλο, με την έννοια ότι οι πελάτες επιλέγουν τις προτιμήσεις τους από την ιστοσελίδα του ξενοδοχείου και αυτές πρέπει να μεταφέρονται με ασφάλεια στη βάση δεδομένων του συστήματός μας. Ακόμη η λειτουργία της βάσης μας έχει να κάνει και με τη συμβατότητα με άλλα συστήματα. Για παράδειγμα με πληροφοριακά συστήματα τραπεζών σχετικά με τον έλεγχο των τραπεζικών λογαριασμών των πελατών.

Τέλος όσον αφορά τον παράγοντα ανάπτυξη , πρέπει να προσεχτούν το κόστος ανάπτυξης, ο χρονικός ορίζοντας ανάπτυξης και η δυσκολία υλοποίησης. Το κόστος ανάπτυξης και ο χρονικός ορίζοντας ανάπτυξης της βάσης δεν θα πρέπει να υπερβαίνουν τα όρια που έχουν συμφωνηθεί στον προϋπολογισμό του έργου μεταξύ των υπευθύνων του έργου και της διοίκησης του ξενοδοχείου. Επιπλέον ο βαθμός δυσκολίας υλοποίησης της βάσης πρέπει να είναι τέτοιος, ώστε να μην εμπεριέχει μεγάλο κόστος και η βάση να είναι έτοιμη μέσα στα πλαίσια του χρονοδιαγράμματος. Αυτή άλλωστε είναι και η βασικότερη επιδίωξη των ανθρώπων του ξενοδοχείου προκειμένου να βελτιώσουν τη λειτουργία του και να προσελκύσουν περισσότερους πελάτες, για να παραμείνουν σε υψηλό επίπεδο ανταγωνισμού.

Η βάση δεδομένων του συστήματός μας ονομάζεται hotel. Η βάση μας αποτελείται από δύο πίνακες, τον πίνακα πελάτη(customer)και τον πίνακα κρατήσεις(reservation). Ακολουθούν ενδεικτικές εικόνες που δείχνουν τη δημιουργία της βάσης, τη δημιουργία πινάκων και τα πεδία-χαρακτηριστικά που έχει ο κάθε πίνακας.

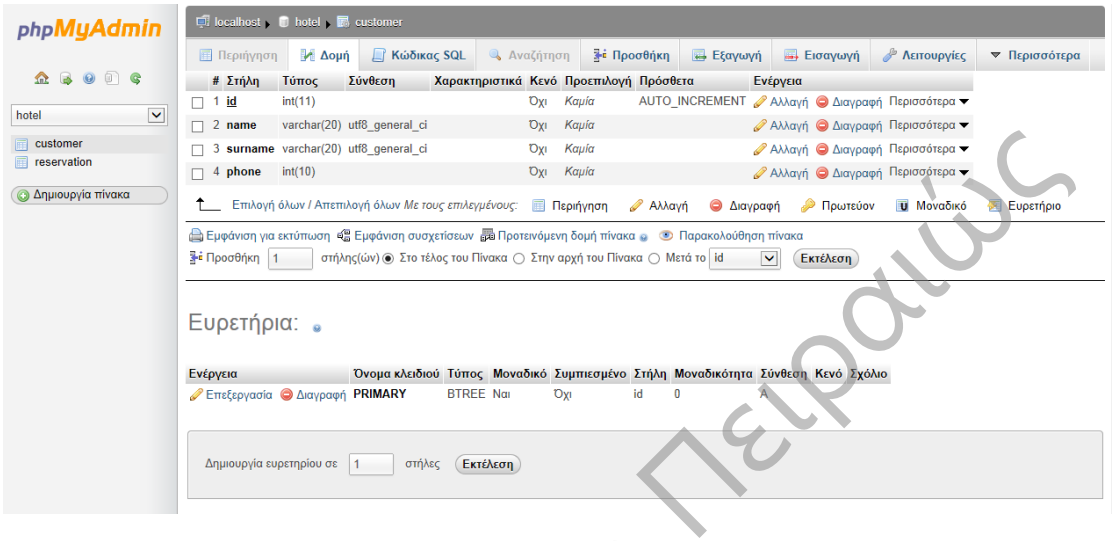
Στην εικόνα που ακολουθεί φαίνεται η επιλογή της βάσης hotel που περιέχει δύο πίνακες, τον πίνακα customer και τον πίνακα reservation.

Εικόνα 1 βάσης δεδομένων



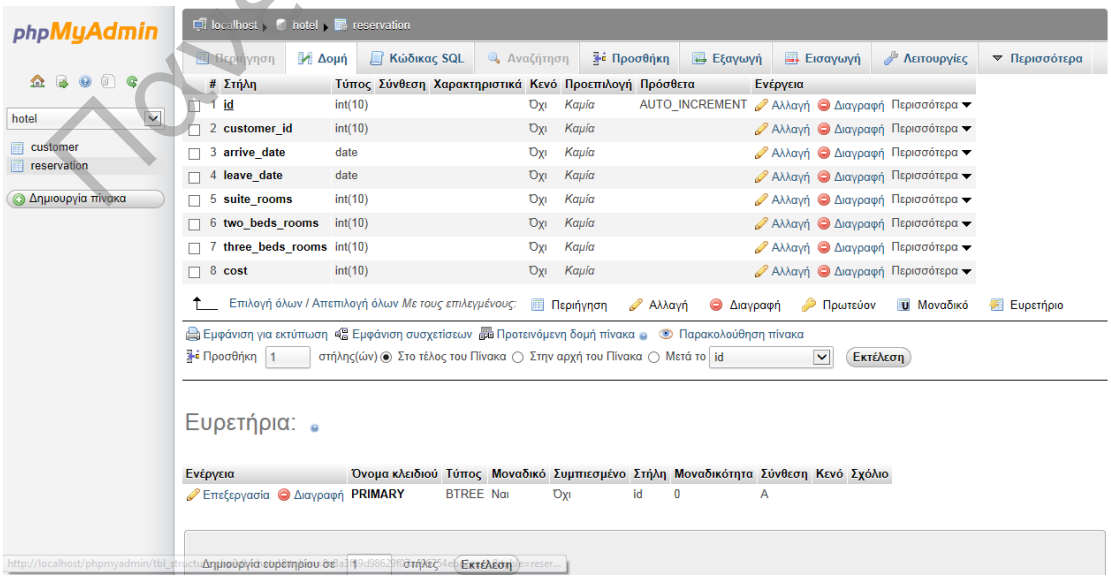
Στην εικόνα που ακολουθεί φαίνεται η δημιουργία του πίνακα customer, όπου φαίνονται ξεκάθαρα τα πεδία-χαρακτηριστικά του πίνακα, ο τύπος των δεδομένων για κάθε χαρακτηριστικό κλπ. Τα χαρακτηριστικά για τον πίνακα customer είναι: customer id, name, surname, phone.

Εικόνα 2 βάσης δεδομένων



Στην εικόνα που ακολουθεί φαίνεται η δημιουργία του πίνακα reservation, όπου φαίνονται ξεκάθαρα τα πεδία-χαρακτηριστικά του πίνακα, ο τύπος των δεδομένων για κάθε χαρακτηριστικό κλπ. Τα χαρακτηριστικά για τον πίνακα reservation είναι: id, customer id, arrive date, leave date, suite rooms, two beds rooms, three beds rooms, cost.

Εικόνα 3 βάσης δεδομένων



5.4 Ψευδοκώδικας

Ακολουθεί ένα ενδεικτικό τμήμα ψευδοκώδικα μέσω του οποίου γίνεται περισσότερο κατανοητό το σκεπτικό με το οποίο έγινε η υλοποίηση της εφαρμογής.

Διάβασε **Όνομα Πελάτη**

Διάβασε **Κατηγορία δωματίου**

Διάβασε **Ημερομηνία**

WHILE **Ημερομηνία# 0** do

If **Κατηγορία δωματίου** = suite AND διαθεσιμότητα>=1 then

Εμφάνισε ‘ Η κράτηση ολοκληρώθηκε’, **Όνομα Πελάτη**

Else

Εμφάνισε ‘ Η κράτηση ακυρώθηκε’

If else **Κατηγορία δωματίου** = two beds AND διαθεσιμότητα >=1then

Εμφάνισε ‘Η κράτηση ολοκληρώθηκε’, **Όνομα Πελάτη**

Else

Εμφάνισε ‘Η κράτηση ακυρώθηκε’

Else **Κατηγορία δωματίου=** three beds AND διαθεσιμότητα >=1then

Εμφάνισε ‘Η κράτηση ολοκληρώθηκε’, **Όνομα Πελάτη**

Else

Εμφάνισε ‘Η κράτηση ακυρώθηκε’

END WHILE

Εμφάνισε’ Παρακαλώ εισάγετε σωστά την ημερομηνία’

Ο ψευδοκώδικας περιγράφει με απλά λόγια ένα μέρος της διαδικασίας που απαιτείται για να μπορέσει ο χρήστης να κάνει κράτηση δωματίου. Εφόσον δώσει έγκυρα στοιχεία όσον αφορά το όνομά του, το είδος και τον αριθμό των δωματίων που επιθυμεί, καθώς και την ημερομηνία που θέλει για να κλείσει τα δωμάτια, το σύστημα ελέγχει αν μπορεί να καλύψει τις προδιαγραφές του χρήστη και εμφανίζει τα αντίστοιχα μηνύματα. Αν ο χρήστης δεν δώσει έγκυρες πληροφορίες όσον αφορά τα στοιχεία του παρακαλείται να συμπληρώσει σωστά την αντίστοιχη φόρμα συμπλήρωσης στοιχείων.

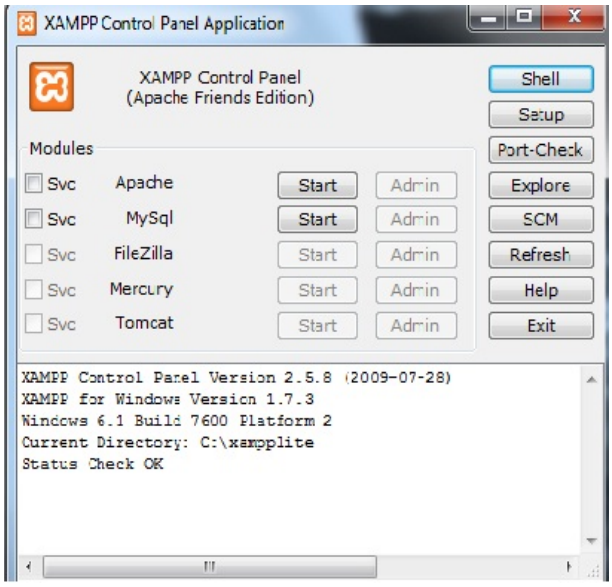
5.5 Εγχειρίδιο διαχειριστή

Στην ενότητα αυτή, μέσω της παρουσίασης του εγχειριδίου διαχειριστή για κάποιον που θέλει να τρέξει την εφαρμογή μας, παρουσιάζεται η κύρια διαδικασία για την οποία υλοποιήθηκε το συγκεκριμένο σύστημα για το ξενοδοχείο Relax Hotel. Αυτή δεν είναι άλλη από την κράτηση δωματίων μέσω Διαδικτύου από τους χρήστες.

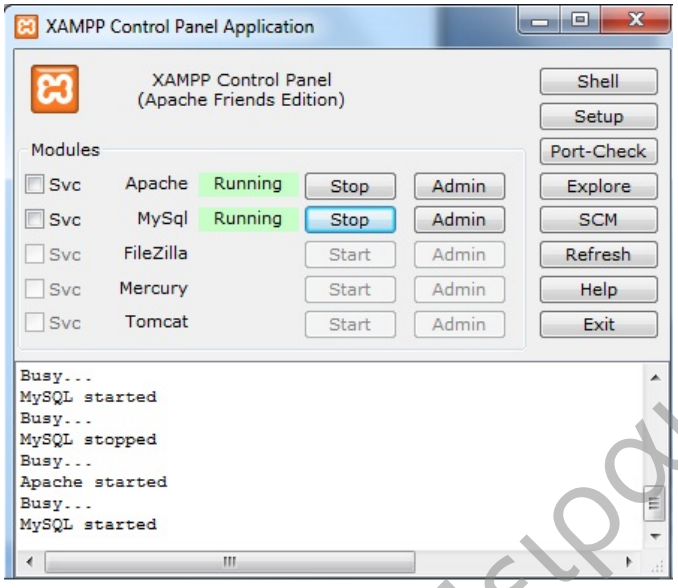
1ο βήμα Εγκατάσταση XAMPP

Εγκαταστήστε το XAMPP (εφόσον δεν υπάρχει ήδη στον υπολογιστή σας). Προτείνετε να κατεβάσετε και να εγκαταστήσετε το XAMPP από τη διεύθυνση http://www.apachefriends.org/en/xampp.html. Ακολουθήστε τις οδηγίες που σας παρέχει το site. Αφού ολοκληρώσετε την εγκατάσταση πηγαίνετε στο Έναρξη Όλα τα προγράμματα όπου βρίσκετε το XAMPP control panel και πατήστε το κουμπί Start στο Apache και MySql (Εικόνα 1) . Μετά από ελάχιστα δευτερόλεπτα θα δείτε την Εικόνα 2.

Εικόνα 1



Εικόνα 2



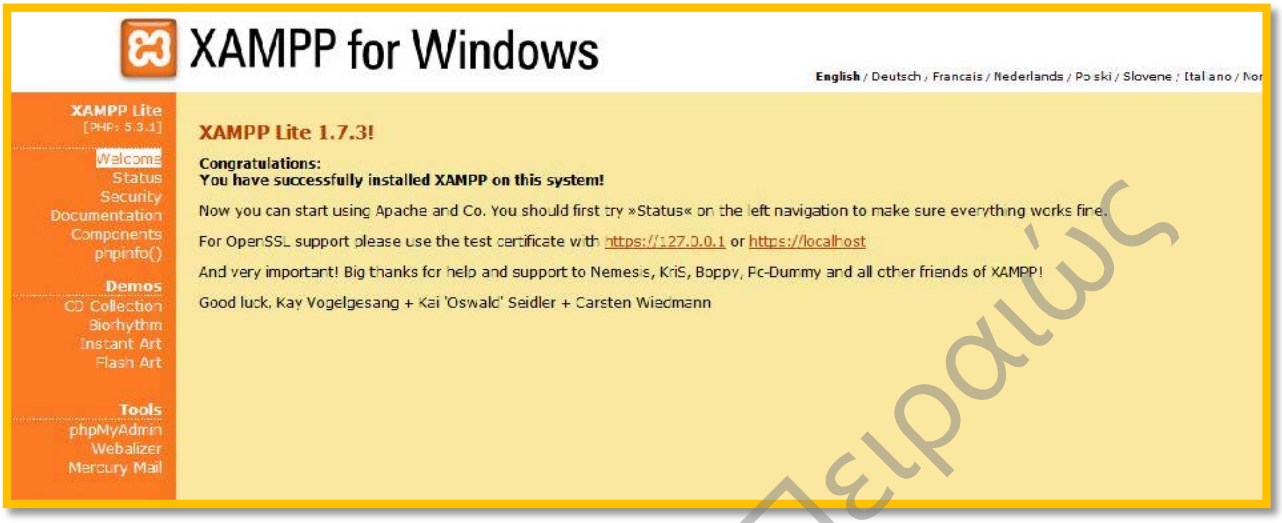
2ο Βήμα: Δημιουργία βάσης δεδομένων

Ανοίγετε τον browser που διαθέτετε (π. χ. Internet Explorer, Mozilla, ή όποιον άλλο έχετε) και πληκτρολογείτε στη γραμμή διευθύνσεων το localhost. Σας εμφανίζονται διαδοχικά οι παρακάτω εικόνες. Ακολουθείστε τις οδηγίες.

Εικόνα 3

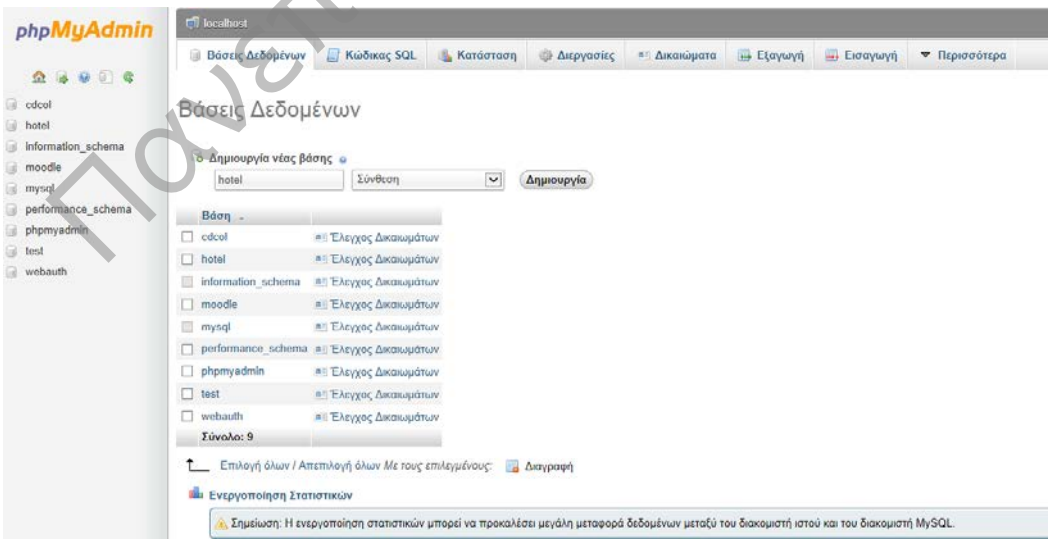
Στην εικόνα 4 που ακολουθεί για να μεταβούμε στο μενού δημιουργίας της βάσης δεδομένων του συστήματός μας επιλέγουμε από την καρτέλα Tools την επιλογή PhpMyAdmin.

Εικόνα 4



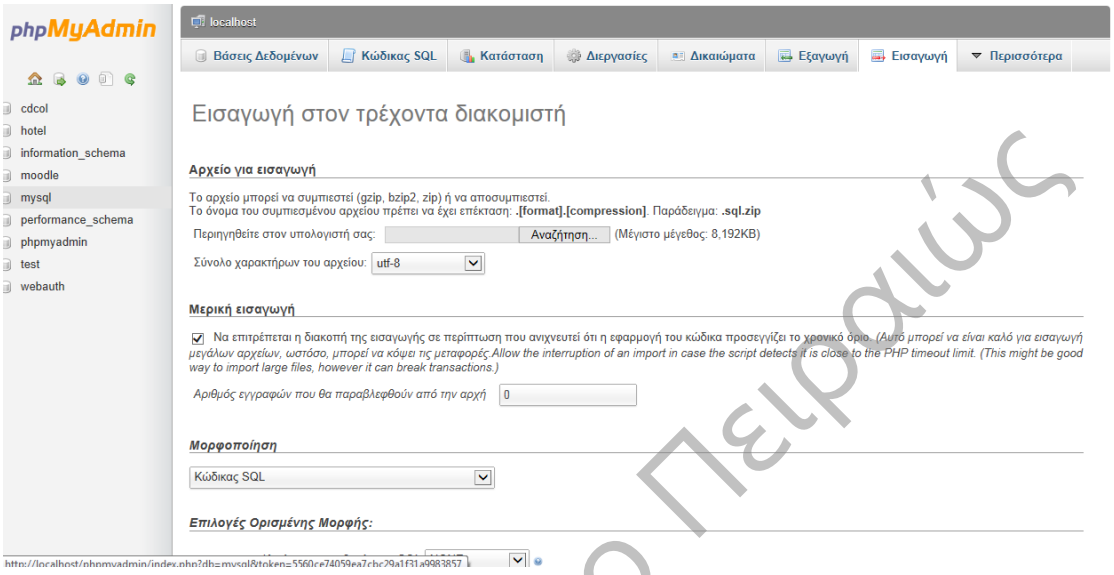
Στη συνέχεια δημιουργούμε τη βάση δεδομένων όπως φαίνεται παρακάτω:

Εικόνα 5(Διαδικασία δημιουργίας βάσης δεδομένων)



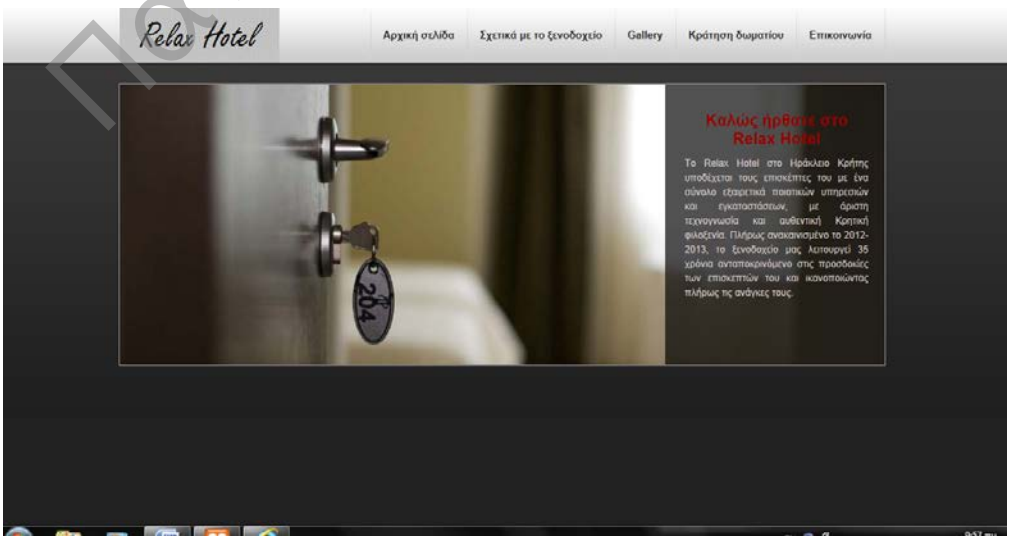
Στη συνέχεια μετά την επιτυχημένη δημιουργία της βάσης δεδομένων μας επιλέγουμε και εισάγουμε (import) τη βάση από το αρχείο SQL που έχουμε. Τον φάκελο που έχουμε με την ονομασία hotel τον βάζουμε μέσα στον φάκελο htdocs toy xampp. Στην καρτέλα αναζήτηση γίνεται η εισαγωγή του SQL αρχείου.

Εικόνα 6 (Διαδικασία δημιουργίας βάσης δεδομένων)



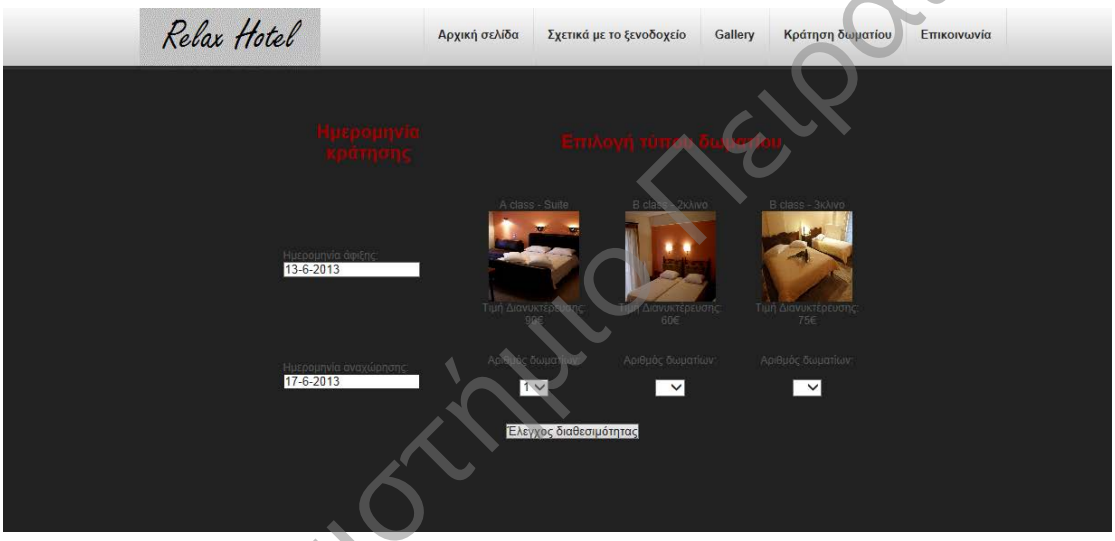
Πλέον είμαστε έτοιμοι να μπούμε στο σύστημά μας!!!!. Το μόνο που μένει είναι να ανοίξουμε το browser της αρεσκείας μας και να πληκτρολογήσουμε localhost/hotel.Η αρχική σελίδα του συστήματός μας είναι η Εικόνα 7.

Εικόνα7 (Αρχική σελίδα του συστήματός μας)

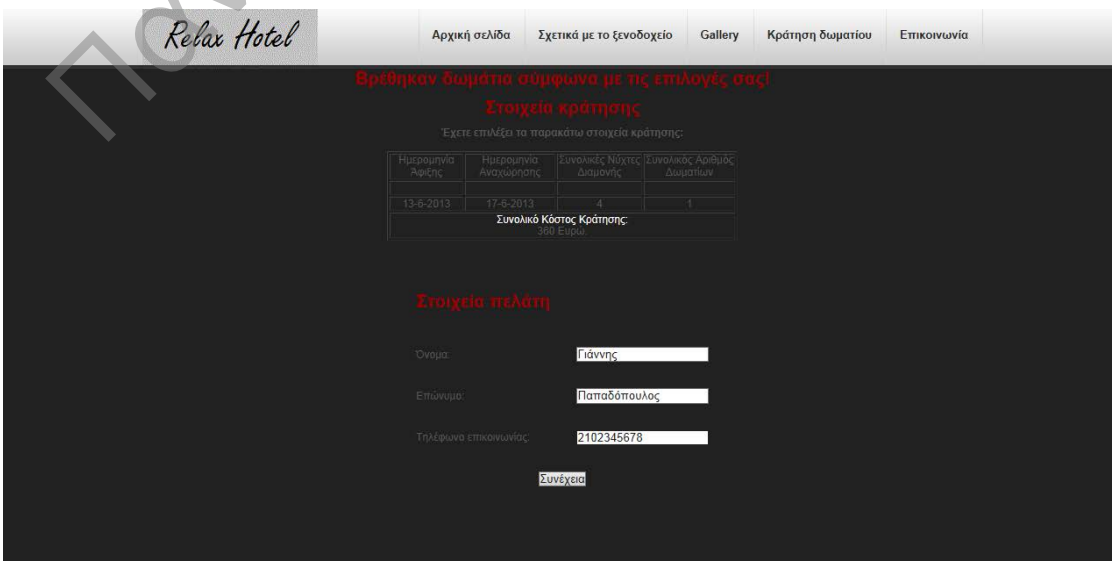


Η καρτέλα Κράτηση Δωματίου είναι αυτή που ενδιαφέρει και το διαχειριστή του συστήματος, ο οποίος θα είναι ο υπάλληλος του ξενοδοχείου. Ο διαχειριστής θα ελέγχει σε καθημερινό βαθμό τη βάση και σύμφωνα με την τελευταία εγγραφή(πιο πρόσφατη) θα αντλεί πληροφορίες για τον τύπο του δωματίου που επέλεξε ο κάθε πελάτης, τις συνολικές μέρες παραμονής κ.α. Το μόνο που κάνουν οι πελάτες είναι να δίνουν τα στοιχεία τους όπως φαίνεται στις εικόνες που ακολουθούν:

Εικόνα 8(Ημερομηνία και είδος δωματίου που επιλέγει ο χρήστης)

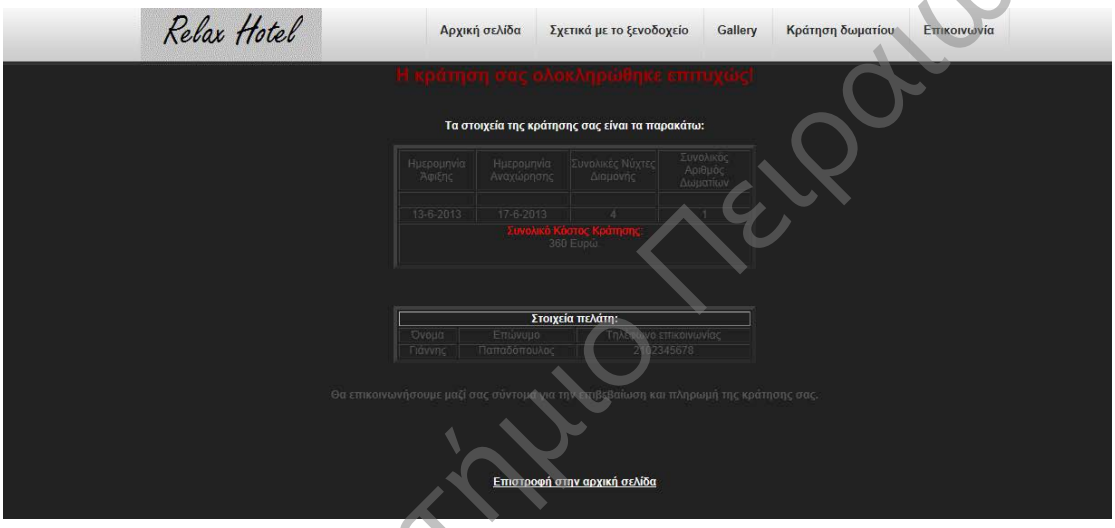


Εικόνα 9 (Συμπλήρωση των στοιχείων του χρήστη)



Οι παραπάνω 2 εικόνες απεικονίζουν μια κράτηση σουίτας για τέσσερις ημέρες από 13-6-2013 έως 17-6-2013 από κάποιον κύριο Γιάννη Παπαδόπουλο. Δώσαμε κάποια ενδεικτικά στοιχεία για να δούμε αν λειτουργεί σωστά η εφαρμογή μας. Το τηλέφωνο επικοινωνίας μαζί του είναι αυτό που έδωσε ο ίδιος στην ετικέτα Τηλέφωνο Επικοινωνίας . Πατώντας στην καρτέλα Συνέχεια η κράτηση επιβεβαιώνεται και εμφανίζεται η παρακάτω εικόνα

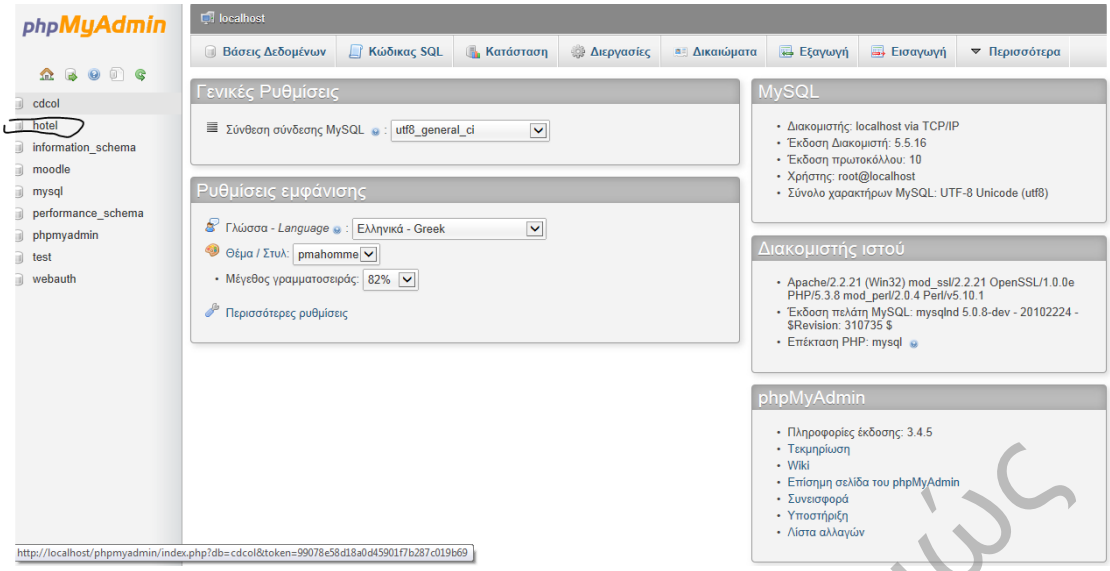
Εικόνα 10(Εμφάνιση αποτελέσματος κράτησης και λοιπές πληροφορίες)



Αυτό που πρέπει να κάνει ο διαχειριστής βλέποντας την παρακάτω εικόνα στην ιστοσελίδα του ξενοδοχείου είναι να πάει στη βάση δεδομένων και να ελέγξει, αν παράλληλα με την επιβεβαίωση καταχώρησης, δημιουργήθηκε εγγραφή στη βάση με τα παραπάνω στοιχεία, δηλαδή όνομα-επώνυμο και τηλέφωνο πελάτη καθώς και τύπο δωματίου.

Πατάμε στον browser που διαθέτουμε localhost/xampp και επιλέγουμε τη βάση hotel από τις διαθέσιμες (Εικόνα11)

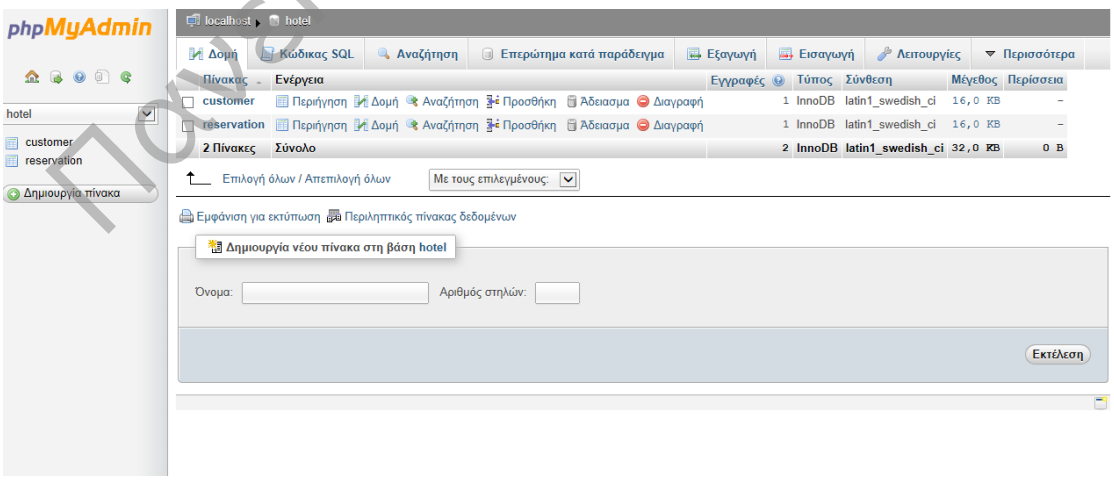
Εικόνα 11 (Έλεγχος διαχειριστή για την ενημέρωση της βάσης δεδομένων του συστήματός μας)



Στη συνέχεια ,αφού ο διαχειριστής έχει ανοίξει την βάση δεδομένων hotel έχουμε την εξής εικόνα(Εικόνα12)

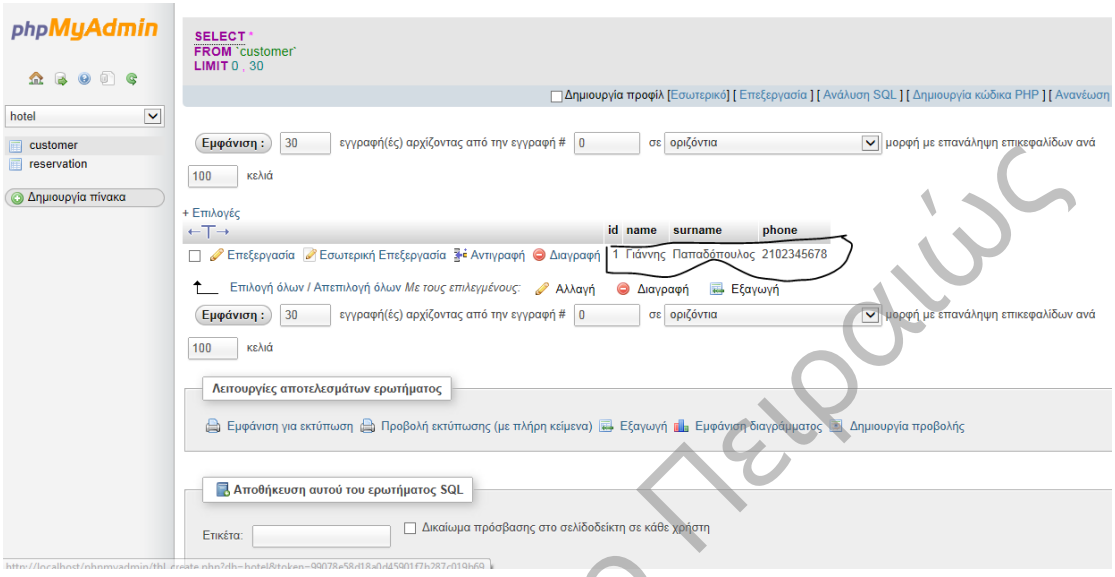
Παρατηρούμε ότι έχουμε από μια εγγραφή για κάθε πίνακα της βάσης δεδομένων του συστήματός μας.

Εικόνα 12 (Εμφάνιση πινάκων της βάσης δεδομένων του συστήματός μας και εμφάνιση εγγραφών)



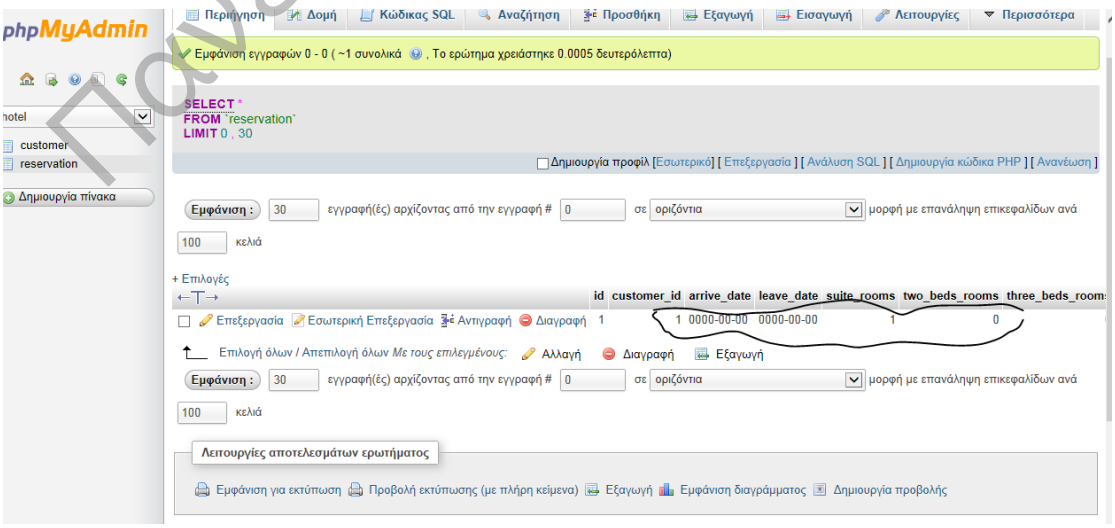
Για να δει ο διαχειριστής αναλυτικά τα στοιχεία της εγγραφής του πίνακα Customer ανοίγει τον πίνακα και βλέπει ότι η εγγραφή δημιουργήθηκε. Αυτό φαίνεται στην εικόνα 13

Εικόνα 13 (Αναλυτικά στοιχεία εγγραφής πίνακα Customer)



Για να δει ο διαχειριστής αναλυτικά τα στοιχεία της εγγραφής του πίνακα Reservations ανοίγει τον πίνακα και βλέπει ότι η εγγραφή δημιουργήθηκε. Αυτό φαίνεται στην εικόνα 14

Εικόνα 14(Αναλυτικά στοιχεία εγγραφής πίνακα Reservations)



Σημαντική σημείωση: Τα 000000 που εμφανίζονται στην ημερομηνία έχει να κάνει με παλαιότερη έκδοση του XAMPP και όχι με κάποιον λάθος ορισμό στα στοιχεία των πινάκων της βάσης δεδομένων του συστήματός μας

6. Η πλατφόρμα Google APP Engine

6.1 Γενική περιγραφή

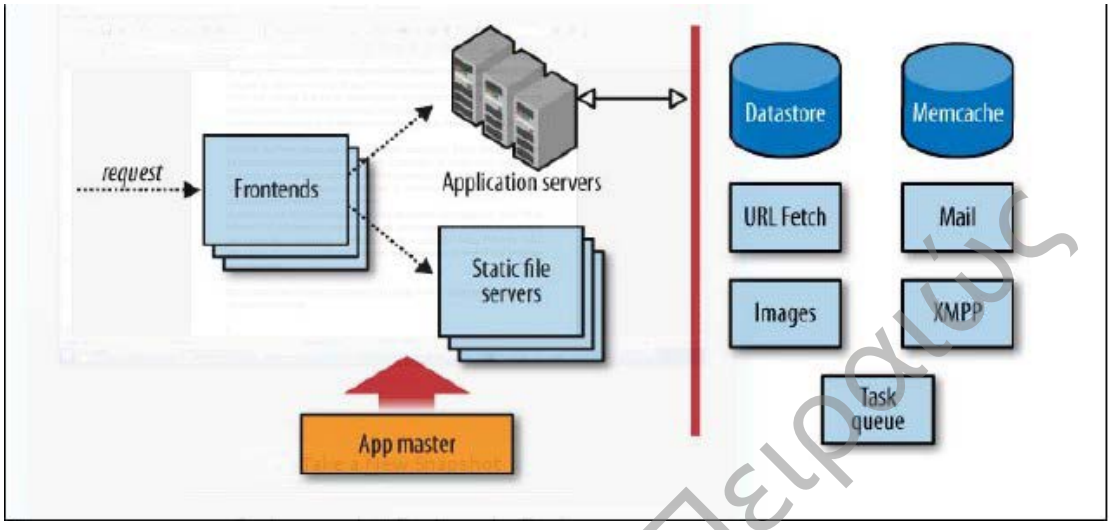
Η πλατφόρμα στην οποία επιλέξαμε να αναπτύξουμε την εφαρμογή μας είναι το AppEngine της Google. Στο κεφάλαιο αυτό, θα περιγράψουμε αναλυτικά το AppEngine και τις δυνατότητες που προσφέρει.

Το Google AppEngine (GAE) είναι μια Cloud Computing Πλατφόρμα ως Υπηρεσία (platform as a service-PaaS) για την ανάπτυξη και φιλοξενία διαδικτυακών εφαρμογών στα κέντρα δεδομένων (data centers) της Google. Αν και το AppEngine μπορεί να φιλοξενήσει οποιουδήποτε είδους διαδικτυακές εφαρμογές, το περιβάλλον του, είναι ειδικά σχεδιασμένο για την ανάπτυξη δυναμικών, πραγματικού χρόνου (real time) εφαρμογών. Πιο συγκεκριμένα, το περιβάλλον του AppEngine είναι κατάλληλα σχεδιασμένο για εφαρμογές που χρειάζεται να εξυπηρετούν ταυτόχρονα πολλούς χρήστες. Όταν μια εφαρμογή εξυπηρετεί πολλούς χρήστες ταυτόχρονα, χωρίς να μειώνεται η απόδοσή της, τότε λέμε ότι κλιμακώνεται (scales). Οι εφαρμογές που είναι γραμμένες για το AppEngine κλιμακώνονται αυτόματα. Καθώς περισσότεροι χρήστες χρησιμοποιούν την εφαρμογή, το AppEngine παρέχει περισσότερους πόρους στην εφαρμογή και φροντίζει και για τη διαχείριση αυτών των πόρων. Η εφαρμογή δεν χρειάζεται να γνωρίζει τίποτα σχετικά με τους πόρους που της έχουν διατεθεί ή σχετικά με τη χρήση αυτών των πόρων.

Σχετικά με το οικονομικό μοντέλο χρήσης, σε αντίθεση με τους <κλασσικούς> εξυπηρετητές (servers) για τη φιλοξενία ιστοσελίδων, με το Google AppEngine, μας διατίθενται δυναμικά πόροι και πληρώνουμε μόνο για τους πόρους που χρησιμοποιούμε. Οι καταναλησκούμενοι πόροι χρεώνονται με βάση τα gigabytes χρήσης και δεν υπάρχουν πάγιες μηνιαίες χρεώσεις, όπως και δεν απαιτούνται προκαταβολές. Οι πόροι που χρεώνονται είναι η χρήση CPU για υπολογισμούς, η αποθήκευση δεδομένων, το εισερχόμενο κι εξερχόμενο εύρος ζώνης του δικτύου, καθώς και διάφορες συγκεκριμένες υπηρεσίες, που παρέχει το GAE. Επίσης, αναφέρουμε, ότι μέχρι ένα συγκεκριμένο όριο στην κατανάλωση πόρων, η χρήση του AppEngine είναι χωρίς χρέωση. Έτσι, μικρές εφαρμογές, με χαμηλό φόρτο εργασίας μπορούν να φιλοξενηθούν στην υποδομή της Google χωρίς κάποιο κόστος.

Στη συνέχεια θα δώσουμε μια γενική περιγραφή της αρχιτεκτονικής του Google AppEngine.

Εικόνα 1. Εικόνα περιγραφής της αρχιτεκτονικής του (GAE)



Μια εφαρμογή του AppEngine αποκρίνεται σε αιτήματα ιστού (web requests). Ένα web request ξεκινάει όταν ένας πελάτης (client), συνήθως ο web browser κάποιου χρήστη, επικοινωνήσει με την εφαρμογή μέσω ενός HTTP request. Η πρώτη στάση για ένα εισερχόμενο request είναι το frontend του App Engine. Ένας εξισορροπητής φόρτου(load balancer) κι ένα σύστημα για την βέλτιστη κατανομή των requests σε πολλά μηχανήματα, δρομολογεί το request σε έναν από τους frontend servers. Το frontend καθορίζει, από τη διεύθυνση που υπάρχει στο request, την εφαρμογή στην οποία αναφέρεται το request. Κάθε εφαρμογή του AppEngine έχει ένα μοναδικό αναγνωριστικό (application ID) κι επίσης, με βάση το αναγνωριστικό αυτό, παίρνει ένα δικό της όνομα (domain name) στο appspot.com. Έτσι, η διεύθυνση προορισμού, που υπάρχει στα requests, έχει τη μορφή: http://app-id.appspot.com/url/path Με βάση αυτό το URL επομένως, δρομολογεί το frontend το request στον κατάλληλο server.

Το URL path μπορεί επίσης να αντιστοιχεί σε έναν διαχειριστή αιτημάτων (request handler),δηλαδή σε κώδικα εφαρμογής, που καλείται για να καθορίσει την απάντηση στο request. Στην περίπτωση αυτή, το frontend στέλνει το request σε κάποιον από τους servers εφαρμογών (app servers). Τότε, το app server ξεκινάει ένα στιγμιότυπο (instance) της εφαρμογής σε έναν server ή χρησιμοποιεί κάποιο υπάρχον στιγμιότυπο αν υπάρχει κάποιο που τρέχει από προηγούμενο request.

Αν το URL path δεν αντιστοιχεί σε τίποτα, που να είναι δηλωμένο στο configuration της εφαρμογής, τότε το frontend επιστρέφει ως απάντηση στον client ένα: HTTP 404 <Not Found> error, όπως δηλαδή θα έκανε κι οποιαδήποτε άλλη web εφαρμογή.

Οι app servers χρησιμοποιούν διάφορες στρατηγικές για την κατανομή των requests και για την εκκίνηση των στιγμιότυπων της εφαρμογής, ανάλογα με το φόρτο εργασίας και με τη χρήση των πόρων. Όλες όμως αυτές οι στρατηγικές είναι σχεδιασμένες, ώστε να λειτουργούν καλύτερα με request handlers, που επιστρέφουν γρήγορα. Για την ακρίβεια, ένα request μπορεί να κάνει μέχρι 30 δευτερόλεπτα μέχρι να επιστρέψει κάποια απάντηση στον client. Σκοπός είναι να μεγιστοποιηθεί η απόδοση (throughput) των στιγμιότυπων των εφαρμογών, ώστε να τρέχουν τόσα στιγμιότυπα, που να μπορούν να ανταποκριθούν στις τρέχουσες ανάγκες του φόρτου εργασίας. Το πλήθος των στιγμιότυπων όμως, δεν θα πρέπει να είναι τόσο μεγάλο, ώστε να μην υπάρχουν στιγμιότυπα, που να είναι αδρανή, χωρίς να εκτελούν κάποια λειτουργία. Επίσης, οι app servers είναι σχεδιασμένοι έτσι ώστε, ένας request handler να μην μπορεί να επηρεάσει τη συμπεριφορά ή την επίδοση ενός άλλου handler, που τρέχει στον ίδιο server.

Τα frontends, οι app servers και οι servers στατικών αρχείων κυβερνώνται από έναν <App master>. Ανάμεσα στις άλλες αρμοδιότητές του, ο <App Master> είναι υπεύθυνος και για την ενημέρωση των εκδόσεων του software της εφαρμογής. Οι ενημερώσεις σε μια εφαρμογή διαδίδονται γρήγορα, χωρίς όμως αυτό να σημαίνει ότι δεν υπάρχει περίπτωση να τρέχει ταυτόχρονα κώδικας δυο διαφορετικών εκδόσεων της εφαρμογής. Αν η εφαρμογή λάβει κάποια requests πριν την μετάβασή της σε νέα έκδοση, τότε η επεξεργασία των request αυτών θα γίνει με τον κώδικα της προηγούμενης έκδοσης της εφαρμογής.

Όπως είδαμε, ο τρόπος με τον οποίο τρέχει μια εφαρμογή του App Engine είναι ο εξής: το frontend δέχεται ένα request, και με βάση το URL και το configuration της εφαρμογής, το δρομολογεί σε έναν από πολλούς servers. Μόλις ο server λάβει το request, κάνει τους υπολογισμούς του κι επιστρέφει κάποια δεδομένα, ως απάντηση στο χρήστη. Ο τρόπος με τον οποίο γίνεται το configuration της εφαρμογής διαφέρει ανάλογα με το runtime περιβάλλον, που χρησιμοποιούμε για την εφαρμογή.

Τα runtime περιβάλλοντα που υποστηρίζει η πλατφόρμα GAE είναι 4: ένα της Java, της Python, της Go και της PHP.

6.2 Runtime environment

Το runtime περιβάλλον ενεργοποιείται μόλις ξεκινήσει η διαχείριση ενός request και εξαφανίζεται μόλις αυτή τελειώσει. Με το να μην διατηρεί στο runtime την κατάσταση ανάμεσα σε διαφορετικά requests, το App Engine μπορεί να κατανείμει το φόρτο σε όσους servers χρειάζεται, ώστε να συμπεριφέρεται σε όλα τα requests με τον ίδιο τρόπο, ανεξάρτητα με το φόρτο που διαχειρίζεται μια δεδομένη χρονική στιγμή.

Επίσης, ο κώδικας μιας εφαρμογής δεν μπορεί να έχει πρόσβαση στον server, στον οποίον τρέχει, με την κλασσική έννοια. Μια εφαρμογή μπορεί να διαβάζει τα δικά της αρχεία από το σύστημα αρχείων του server (filesystem), αλλά δεν μπορεί να γράφει σε αρχεία, ούτε να διαβάζει αρχεία που ανήκουν σε άλλες εφαρμογές. Επίσης, μια εφαρμογή δεν μπορεί να έχει πρόσβαση στο hardarw του server, που σχετίζεται με το δίκτυο. Επομένως, οι δικτυακές λειτουργίες γίνονται με τη χρήση υπηρεσιών του AppEngine, ειδικά σχεδιασμένων γι' αυτό το λόγο. Τέλος, μια εφαρμογή μπορεί να βλέπει τις μεταβλητές περιβάλλοντος, που έχει θέσει το App Engine, αλλά αλλαγές σε αυτές δεν διατηρούνται ανάμεσα σε διαφορετικά requests.

Εν ολίγοις, κάθε request ζει στο δικό του <sandbox>. Αυτό επιτρέπει στο App Engine να διαχειρίζεται ένα request με τον server, που κατά την εκτίμησή του, θα αποκρίνεται γρηγορότερα. Δεν υπάρχει τρόπος να εγγυηθεί κανείς ότι ο ίδιος server θα διαχειριστεί δυο διαφορετικά requests, ακόμα κι αν προέρχονται από τον ίδιο client και εντός σύντομου χρονικού διαστήματος.

Το <sandbox> αυτό επιτρέπει ακόμα στο App Engine, να τρέχει πολλές εφαρμογές στον ίδιο server, χωρίς η συμπεριφορά της μιας εφαρμογής να επηρεάζει την άλλη. Εκτός από την περιορισμένη πρόσβαση στο λειτουργικό σύστημα, το runtime περιβάλλον περιορίζει και τη χρήση της CPU και της μνήμης, που μπορεί να κάνει ένα request.

Οι βασικοί περιορισμοί που επιβάλλει το sandbox είναι:

- Μια εφαρμογή δεν μπορεί να δημιουργεί και να αναθέτει εργασίες σε επιπρόσθετες διεργασίες .Όλη η επεξεργασία ενός request πρέπει να γίνεται από τη διεργασία του request handler.

- Μια εφαρμογή δεν μπορεί να εγκαθιστά δικτυακές συνδέσεις. Ορισμένες δικτυακές δυνατότητες παρέχονται από υπηρεσίες του AppEngine, όπως είναι το URL Fetch και το Mail.

- Μια εφαρμογή μπορεί μόνο να διαβάζει από το filesystem και μπορεί να διαβάζει μόνο τον δικό της κώδικα και τα δικά της resource files. Δεν μπορεί να δημιουργεί ή να τροποποιεί αρχεία. Αντί για αρχεία, για την αποθήκευση δεδομένων, μια εφαρμογή μπορεί να χρησιμοποιεί το datastore.

- Μια εφαρμογή δεν μπορεί να δει ή να γνωρίζει οτιδήποτε για άλλες εφαρμογές ή διεργασίες που τρέχουν στο server. Το ίδιο ισχύει και για τους request handlers. Ένας request handler δεν μπορεί να δει άλλους request handlers της ίδιας εφαρμογής που μπορεί να τρέχουν παράλληλα στον ίδιο server.

6.3 Datastore

Οι περισσότερες κλιμακώσιμες web εφαρμογές χρησιμοποιούν ξεχωριστό σύστημα για τη διαχείριση των web requests και για την αποθήκευση δεδομένων. Το σύστημα διαχείρισης των requests δρομολογεί κάθε request σε ένα από πολλά μηχανήματα, καθένα από τα οποία διαχειρίζεται το request χωρις να γνωρίζει τίποτα σχετικά με τα requests που έχουν δρομολογηθεί σε άλλα μηχανήματα. Κάθε request handler συμπεριφέρεται χωρίς να διατηρεί κατάσταση, δρώντας αποκλειστικά και μόνο με βάση το περιεχόμενο του request για το οποίο έχει να δώσει απάντηση. Οι περισσότερες όμως web εφαρμογές χρειάζεται να διατηρούν κατάσταση. Για παράδειγμα, η εφαρμογή μπορεί να χρειάζεται να θυμάται ότι ο χρήστης που έκανε ένα request, είναι ο ίδιος χρήστης που έκανε ένα request νωρίτερα σε κάποιο άλλο μηχάνημα. Για το λόγο αυτό, οι request handlers πρέπει να αλληλεπιδρούν με μια κεντρική βάση δεδομένων, όπου και θα ενημερώνεται η κατάσταση της εφαρμογής.

Όπως και με τη διαχείριση των requests, το AppEngine διαχειρίζεται την κλιμάκωση και τη διατήρηση της αποθήκευσης δεδομένων αυτόματα. Η εφαρμογή αλληλεπιδρά με ένα αφηρημένο μοντέλο, που κρύβει τις λεπτομέρειες της διαχείρισης από τα data servers. Πίσω από αυτό το αφηρημένο μοντέλο, το AppEngine προσφέρει δυο επιλογές για την αποθήκευση των δεδομένων, οι οποίες διαφέρουν μεταξύ τους ως προς τη διαθεσιμότητα και τη συνέπεια των δεδομένων.

Η πρώτη επιλογή είναι το High Replication Datastore (HRD). Στο HRD, τα δεδομένα αντιγράφονται σε πολλά data centers . Αυτό παρέχει

το υψηλότερο επίπεδο διαθεσιμότητας για αναγνώσεις κι εγγραφές, με υψηλό κόστος στις εγγραφές, λόγω της διάδοσης των δεδομένων. Στο μοντέλο αυτό, δεδομένου ενός αρκούντως μεγάλου χρονικού διαστήματος, στο οποίο δεν έχουν σταλεί αλλαγές στο σύστημα, όλες οι ενημερώσεις αναμένονται να διαδοθούν τελικώς στο σύστημα κι όλα τα αντίγραφα να είναι συνεπή μεταξύ τους. Σημειώνουμε ότι το High Replication Datastore είναι η πρώτη επιλογή για μια νέα εφαρμογή του AppEngine κι ότι αυτή είναι η επιλογή που χρησιμοποιήσαμε κι εμείς για την εφαρμογή μας.

Η δεύτερη επιλογή είναι το Master/Slave Datastore. Το Master/Slave Datastore προσδιορίζει ένα data center, το οποίο διατηρεί το κύριο αντίγραφο όλων των δεδομένων. Τα δεδομένα που γράφονται στο master data centers διαδίδονται ασύγχρονα σε όλα τα υπόλοιπα (slave) data centers . Από τη στιγμή που μόνο ένα data center είναι master για εγγραφές σε μια δεδομένη χρονική στιγμή, το Master/Slave προσφέρει ισχυρή συνέπεια για όλες τις αναγνώσεις και τα ερωτήματα, με το κόστος προσωρινής μη διαθεσιμότητας, λόγω προβλημάτων των data centers .

Στη συνέχεια, θα παρουσιάσουμε το αφηρημένο μοντέλο του Datastore, της βάσης δεδομένων του Google AppEngine, με το οποίο αλληλεπιδρούν οι εφαρμογές του. Το Datastore του AppEngine είναι μια αντικειμενοστραφής βάση δεδομένων, η οποία στηρίζεται στο **Big Table** της Google, το οποίο με τη σειρά του είναι ένα σύστημα βάσεων δεδομένων που έχει αναπτυχθεί στο **Google File System** (GFS). Ένα αντικείμενο στο Datastore είναι γνωστό ως οντότητα (**entity**). Ένα entity έχει ένα κλειδί, που το αναγνωρίζει μοναδικά σε ολόκληρο το σύστημα. Τα κλειδιά μπορούν να αποθηκεύονται ως δεδομένα στα entities, ώστε να δημιουργούνται αναφορές από ένα αντικείμενο σε ένα άλλο.

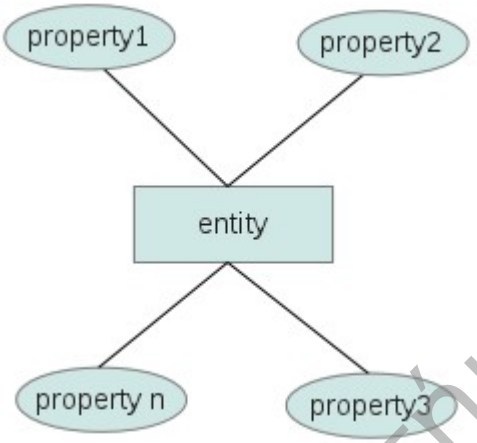
Το κλειδί ενός entity έχει διάφορα μέρη. Το πρώτο μέρος είναι το **application id**, το οποίο διαβεβαιώνει ότι δεν υπάρχει entity σε άλλη εφαρμογή με το ίδιο κλειδί. Επίσης διαβεβαιώνει ότι καμιά άλλη εφαρμογή δεν μπορεί να αποκτήσει πρόσβαση στα δεδομένα μιας συγκεκριμένης εφαρμογής.

Ένα άλλο σημαντικό μέρος του κλειδιού είναι το είδος (**kind**). Κάθε entity του Datastore ανήκει σε ένα συγκεκριμένο kind. Το kind κατηγοριοποιεί το entity για τους σκοπούς των ερωτημάτων. Για παράδειγμα, μια εφαρμογή διαχείρισης ανθρώπινου δυναμικού, ενδεχομένως να αναπαριστά κάθε πελάτη της εταιρίας ως ένα entity που ανήκει στο kind <Employee>. Η εφαρμογή είναι υπεύθυνη για να προσδιορίσει το kind ενός entity κατά τη δημιουργία του entity.

Τέλος, το κλειδί περιέχει κι ένα **entity ID**. Αυτό μπορεί να είναι μια αυθαίρετη συμβολοακολουθία (string), που καθορίζεται από την εφαρμογή ή μπορεί να είναι ένα κλειδί που δημιουργείται αυτόματα από το Datastore. Από τη στιγμή που ένα entity έχει δημιουργηθεί, το κλειδί του δεν μπορεί να αλλάξει κι αυτό ισχύει για όλα τα μέρη του κλειδιού.

Τα δεδομένα ενός entity είναι αποθηκευμένα σε μία ή περισσότερες ιδιότητες (**properties**). Ένα property έχει ένα όνομα και μια ή περισσότερες τιμές. Οι τιμές ενός entity για ένα δεδομένο property δεν είναι ανάγκη να είναι όλες του ίδιου τύπου.

Εικόνα 2. Σχηματική αναπαράσταση ενός entity με τα properties του



Τα entities στο Datastore σχηματίζουν ένα ιεραρχικά δομημένο σχήμα. Όταν δημιουργούμε ένα νέο entity, μπορούμε προαιρετικά, να δηλώσουμε ένα άλλο entity ως γονιό του. Η συσχέτιση αυτή, μεταξύ ενός entity και του γονιού του είναι μόνιμη και δεν μπορεί να αλλάξει από τη στιγμή, που το entity έχει δημιουργηθεί. Ένα entity χωρίς γονιό ονομάζεται **root** entity. Ο γονιός ενός entity , ο γονιός του γονιού ενός entity κ.ο.κ αναδρομικά, είναι οι πρόγονοι του entity. Η ακολουθία των entities ξεκινώντας από ένα root entity και πηγαίνοντας από γονιό σε παιδί, αποτελεί το μονοπάτι των προγόνων ενός entity και συμπεριλαμβάνεται στο κλειδί του. Φαίνεται επομένως ότι τα entities συνδέονται μεταξύ τους με έναν τρόπο τέτοιο ώστε να προκύπτει μια δενδρική δομή.

6.4 Υπηρεσίες

Τα βασικά συστατικά μέρη του AppEngine είναι το runtime περιβάλλον, το Datastore και οι κλιμακώσιμες υπηρεσίες που παρέχει.

Έχοντας περιγράψει τα δυο πρώτα, μας μένει να περιγράψουμε τις υπηρεσίες που παρέχει το AppEngine. Αυτές είναι οι: App Identity, Blobstore, Google Cloud Storage, Capabilities, Conversion, Channel, Images, Mail, Memcache, Prospective Search, Task Queues, URL Fetch, Users και XMPP.

- **App Identity**

Το App Identity επιτρέπει στην εφαρμογή μας να πιστοποιεί την ταυτότητά της σε άλλα εξωτερικά συστήματα.

- **Blobstore**

Το Blobstore επιτρέπει στην εφαρμογή μας να χρησιμοποιεί αντικείμενα δεδομένων, τα οποία ονομάζονται blobs, και είναι πολύ μεγαλύτερα σε μέγεθος από τα αντικείμενα του Datastore. Ένα blob δημιουργείται ανεβάζοντας ένα αρχείο μέσω ενός HTTP request. Συνήθως αυτό γίνεται με τον εξής τρόπο: η εφαρμογή παρουσιάζει μια φόρμα, η οποία έχει ένα πεδίο που καλεί το χρήστη να δώσει ένα αρχείο που θέλει να ανεβάσει. Μόλις η φόρμα υποβληθεί, το Blobstore δημιουργεί το blob από τα περιεχόμενα του αρχείου κι επιστρέφει μια αναφορά στο blob, η οποία χρησιμοποιείται στη συνέχεια για τη διαχείριση του blob.

- **Google Cloud Storage**

Το **Google Cloud Storage** είναι μια ακόμα υπηρεσία αποθήκευσης που παρέχει το AppEngine και προσφέρει ορισμένα χαρακτηριστικά που δεν έχει το Datastore ή το Blobstore, όπως Access Control Lists (ACLS) για τη διαχείριση των δεδομένων.

- **Capabilities**

Με το Capabilities η εφαρμογή μας μπορεί να ανιχνεύει διακοπές ή προγραμματισμένες περιόδους μη διαθεσιμότητας ορισμένων χαρακτηριστικών της εφαρμογής μας και να απενεργοποιεί τα χαρακτηριστικά αυτά, πριν επηρεάσουν τους χρήστες της.

- **Conversion**

Το Conversion κάνει μετατροπές μεταξύ διάφορων τύπων αρχείων. Οι τύποι αρχείων που υποστηρίζει είναι: HTML, PDF, text και διάφορα formats για εικόνες, ενώ οι μετατροπές μπορούν να γίνονται τόσο σύγχρονα όσο και ασύγχρονα.

- **Channel**

Το Channel δημιουργεί μια μόνιμη σύνδεση μεταξύ της εφαρμογής μας και των Google servers, επιτρέποντας στην εφαρμογή μας να στέλνει real time μηνύματα, σε clients. Το χαρακτηριστικό αυτό είναι ιδιαίτερα χρήσιμο για εφαρμογές που θέλουν να ενημερώνουν άμεσα τους χρήστες τους σχετικά με νέες πληροφορίες ή για εφαρμογές όπου ένας χρήστης επιθυμεί να στέλνει real time πληροφορίες σε άλλους χρήστες. Χαρακτηριστικά παραδείγματα τέτοιων εφαρμογών είναι τα παιχνίδια που απαιτούν πολλούς παίκτες και τα chat rooms.

- **Images**

Με το Images, το AppEngine παρέχει τη δυνατότητα διαχείρισης εικόνων. Με την υπηρεσία αυτή, μπορούμε να αλλάζουμε το μέγεθος, να περιστρέφουμε και να κόβουμε εικόνες.

- **Mail**

Με την υπηρεσία αυτή, οι εφαρμογές του AppEngine μπορούν να στέλνουν emails εκ μέρους των διαχειριστών της εφαρμογής καθώς κι εκ μέρους χρηστών των Google Accounts. Το Mail δίνει ακόμα τη δυνατότητα στις AppEngine εφαρμογές να λαμβάνουν emails σε διάφορες διευθύνσεις. Οι εφαρμογές, χρησιμοποιώντας το Mail, λαμβάνουν και στέλνουν emails με τη μορφή HTTP requests.

- **Memcache**

Οι κλιμακώσιμες web εφαρμογές υψηλής επίδοσης, συχνά χρησιμοποιούν για ορισμένες εργασίες μια κατανεμημένη cache μνήμη επιπλέον ή στη θέση της μόνιμης αποθήκευσης. Για το σκοπό αυτό, το AppEngine έχει το Memcache.

- **Prospective Search**

Το Prospective Search επιτρέπει σε μια εφαρμογή να δηλώσει ένα μεγάλο σύνολο ερωτημάτων που αντιστοιχίζονται αυτόματα σε μια ροή από έγγραφα εισόδου. Κάθε φορά που παρουσιάζεται ένα νέο έγγραφο, το prospective search επιστρέφει τα IDs όλων των δηλωμένων ερωτημάτων που αντιστοιχούν στο έγγραφο αυτό.

- **Task Queues**

Με τα Task Queues, οι AppEngine εφαρμογές μπορούν να κάνουν εργασίες που ξεκινάν με το request κάποιου χρήστη, αλλά μπορούν να

ολοκληρωθούν έξω από τα πλαίσια του συγκεκριμένου request. Αν κάποια εφαρμογή χρειάζεται να εκτελέσει κάποια εργασία στο background, χρειάζεται το Task Queue για να την οργανώσει σε μικρές διακριτές μονάδες εργασίας που καλούνται tasks. Στη συνέχεια η εφαρμογή βάζει τα tasks σε μια ή περισσότερες ουρές προς εκτέλεση. Η οργάνωση των tasks σε ουρές εξαρτάται, από το configuration της εφαρμογής.

- **URL Fetch**

Όπως έχουμε αναφέρει, μια AppEngine εφαρμογή ζει στο δικό της sandbox. Ένας από τους περιορισμούς που επιβάλλει το sandbox είναι ότι η εφαρμογή δεν μπορεί να δημιουργεί αυθαίρετα δικτυακές συνδέσεις με άλλες web τοποθεσίες. Οι AppEngine εφαρμογές χρησιμοποιούν το URL Fetch για να κάνουν HTTP requests και να συνδέονται σε άλλες τοποθεσίες.

- **Users**

Με το Users , οι AppEngine εφαρμογές μπορούν να πιστοποιούν χρήστες, οι οποίοι έχουν λογαριασμό στο Google Accounts, καθώς και λογαριασμούς σε κάποιο Google App domain. Επίσης, η εφαρμογή μπορεί να καταλαβαίνει αν κάποιος χρήστης έχει κάνει sign-in ή όχι, όπως και το αν είναι συνδεδεμένος ως administrator ή ως απλός χρήστης. Αυτό διευκολύνει αρκετά την ανάπτυξη περιοχών στις οποίες έχουν πρόσβαση μόνο οι διαχειριστές της εφαρμογής.

- **XMPP**

Με το XMPP, οι εφαρμογές του AppEngine μπορούν να στέλνουν και να λαμβάνουν στιγμιαία μηνύματα από και προς χρήστες XMPP συμβατών υπηρεσιών, όπως το Google Talk .

6.5 Πλεονεκτήματα GAE

Το Google AppEngine επιτρέπει στους χρήστες να εκτελέσουν εφαρμογές διαδικτύου στην υποδομή της Google. Οι AppEngine εφαρμογές, είναι εύκολο να δημιουργηθούν, να συντηρηθούν και καλύπτουν τις ανάγκες για αποθήκευση που όλο αυξάνονται. Το Google AppEngine καθιστά εύκολο το να δημιουργήσουμε μια εφαρμογή που τρέχει αξιόπιστα, ακόμη και κάτω από μεγάλο φόρτο και με μεγάλες ποσότητες δεδομένων. Περιλαμβάνει τα ακόλουθα χαρακτηριστικά:

- δυναμικό web servicing με πλήρη υποστήριξη για κοινές τεχνολογίες διαδικτύου
- μόνιμη αποθήκευση με ερωτήματα και διεργασίες
- αυτόματη κλιμάκωση και εξισορρόπηση φορτίου
- APIs για τον έλεγχο ταυτότητας των χρηστών και αποστολή e-mail χρησιμοποιώντας Google λογαριασμούς
- ένα πλήρως εξοπλισμένο περιβάλλον, τοπικά ανεπτυγμένο, που προσομοιώνει το Google AppEngine στον υπολογιστή μας
- ουρές εργασιών που στόχο έχουν την εκτέλεση εργασιών έξω από το πεδίο ενός αιτήματος web
- προγραμματισμένες εργασίες για την ενεργοποίηση συμβάντων σε συγκεκριμένο χρόνο και σε τακτά χρονικά διαστήματα.

Το AppEngine δεν κοστίζει τίποτα στο χρήστη για να ξεκινήσει. Όλες οι εφαρμογές μπορούν να χρησιμοποιήσουν μέχρι και 1 gigabyte της αποθήκευσης, αρκετή CPU και εύρος ζώνης για να υποστηρίξουν μια αποτελεσματική εφαρμογή, η οποία θα μπορεί να εξυπηρετεί 5 εκατομμύρια προβολές περίπου το μήνα, εντελώς δωρεάν. Όταν ο χρήστης ενεργοποιήσει τη χρέωση για την εφαρμογή του, τότε του επιτρέπεται να αγοράσει επιπλέον πόρους που του χρειάζονται. Τα δωρεάν όρια παραμένουν τα ίδια και πληρώνει μόνο για τους επιπλέον πόρους που χρησιμοποιεί.

6.6 Deploy και Διαχείριση εφαρμογής με το GAE

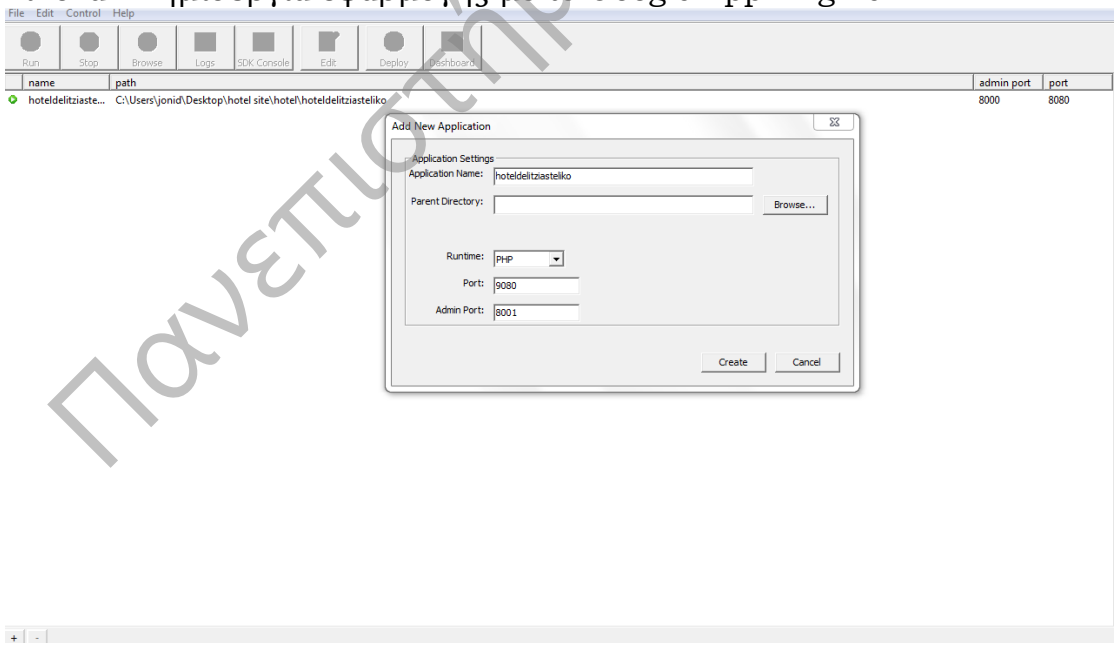
Στην ενότητα αυτή θα περιγράψουμε τις σημαντικότερες δυνατότητες που μας δίνει το GAE για να διαχειριστούμε την εφαρμογή που έχουμε κάνει deploy στην πλατφόρμα.

Προηγουμένως όμως θα δείξουμε τον τρόπο με τον οποίο κάνουμε deploy την εφαρμογή μας μέσω του Google App Engine Launcher.

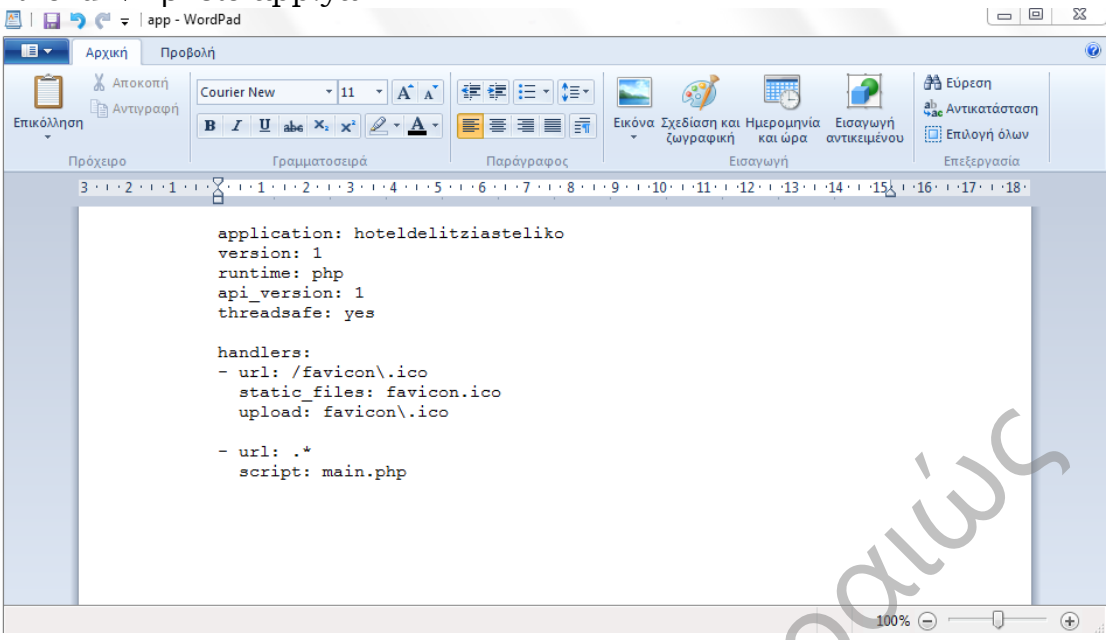
Αρχικά ανοίγουμε τον Google App Engine Launcher και στη συνέχεια πάμε **File -> Create New Application .** Εδώ χρειάζεται προσοχή να δώσουμε το όνομα στην εφαρμογή που δηλώσαμε και στην πλατφόρμα GAE όταν επιλέξαμε **Create Application** . Αποθηκεύουμε την εφαρμογή μας στο φάκελο που είναι τα αρχεία της εφαρμογής μας. Στη συνέχεια το Google App Engine Launcher δημιουργεί αυτόματα στο φάκελό μας ένα αρχείο **app.yaml** που είναι υπεύθυνο για τη διαχείριση της εφαρμογής και που το όνομά του πρέπει να είναι το hoteldelitziasteliko όπως δηλώσαμε την εφαρμογή μας. Γενικά ότι αλλαγές κάνουμε στην εφαρμογή μας π.χ εμφάνιση γίνονται απ αυτό το αρχείο . Επιπλέον το Google App Engine Launcher δημιουργεί και ένα αρχείο **main.php** που περιλαμβάνει τον κώδικα της εφαρμογής μας.

Η παραπάνω διαδικασία φαίνεται στις εικόνες που ακολουθούν

Εικόνα1 Δημιουργία εφαρμογής με το Google App Engine

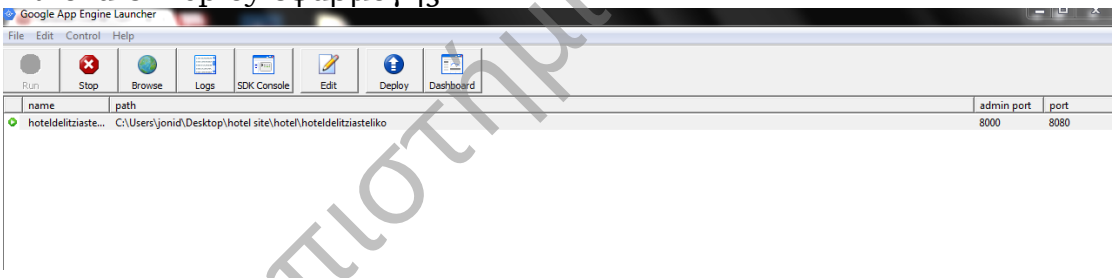


Εικόνα 2 Αρχείο app.yaml



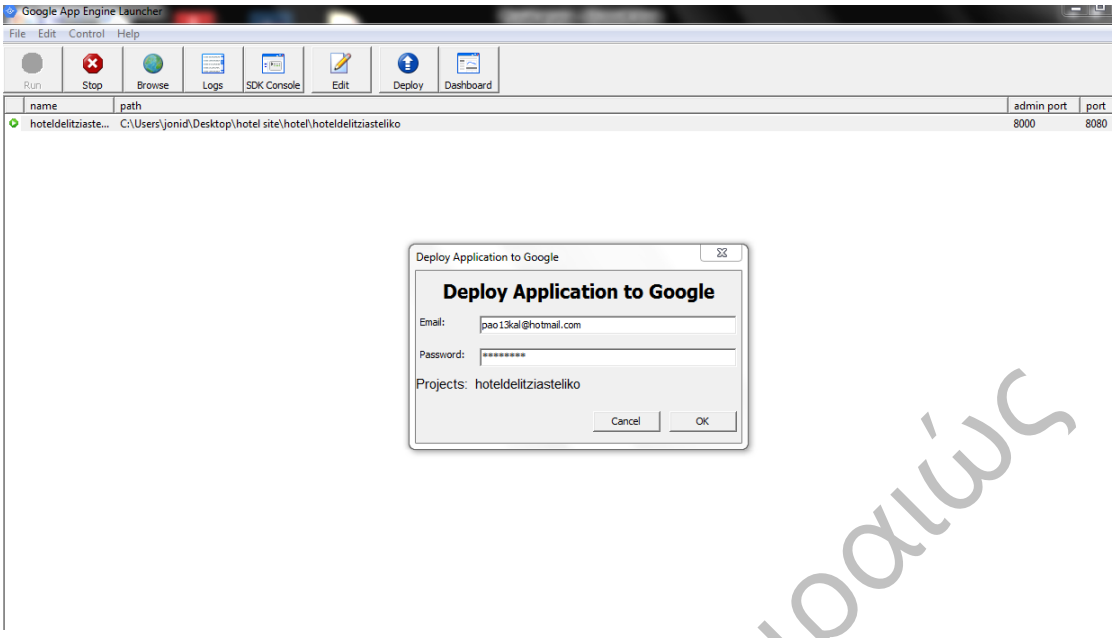
Τώρα αφού δημιουργήσαμε την εφαρμογή πρέπει να την κάνουμε **deploy** στην πλατφόρμα όπως φαίνεται στην παρακάτω εικόνα μέσω του του Google App Engine Launcher

Εικόνα 3 Deploy εφαρμογής



Στη συνέχεια στο παραθυράκι που εμφανίζεται δίνουμε το e-mail μας και τον κωδικό μας για να μπορέσει η πλατφόρμα του GAE να κάνει ταυτοποίηση των στοιχείων μας και να μπορέσουμε ν ανεβάσουμε μ επιτυχία την εφαρμογή μας όπως φαίνεται στην εικόνα 4

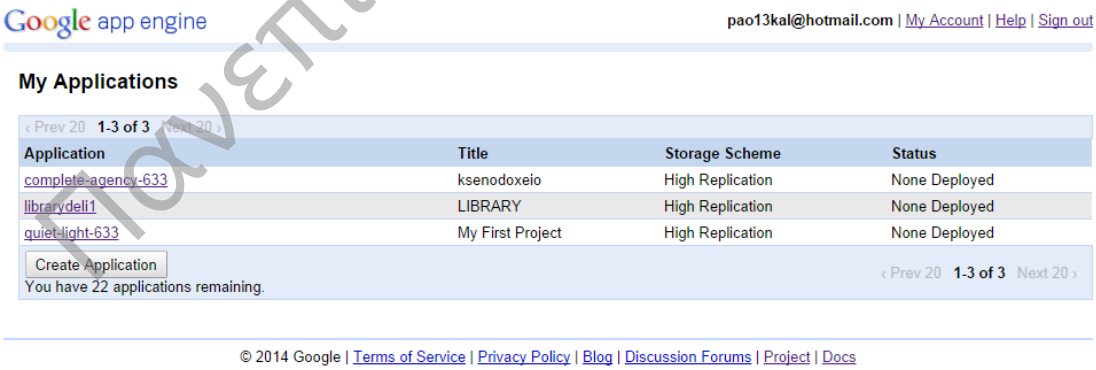
Εικόνα 4 Deploy εφαρμογής συνέχεια



Αφού πατήσουμε οκ η εφαρμογή μας έχει ανέβει με επιτυχία στην πλατφόρμα GAE.

Αρχικά μεταβαίνουμε στην εξής ιστοσελίδα: https://appengine.google.com/ και στη συνέχεια μας εμφανίζεται η εξής εικόνα(αφού δώσουμε τα στοιχεία του λογαριασμού μας αν δεν τα αποθηκεύει το σύστημα)

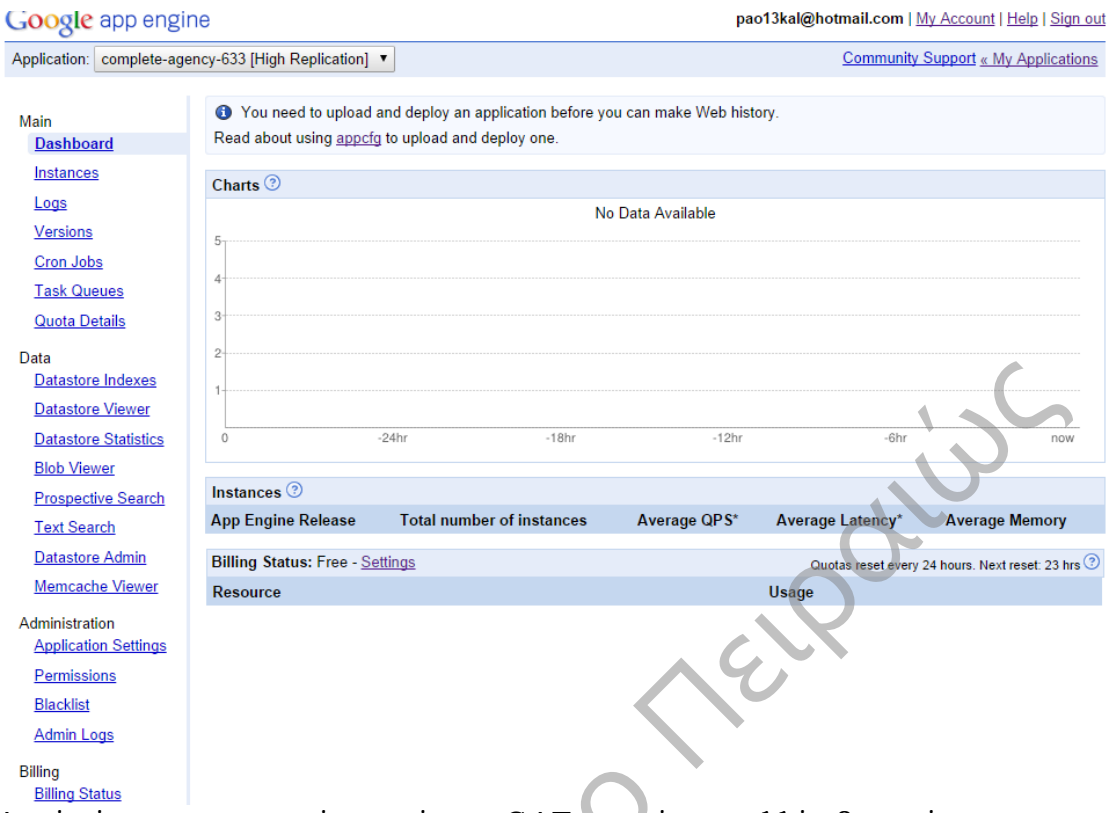
Εικόνα 1



Εδώ φαίνονται οι εφαρμογές που έχουμε ανεβάσει στην πλατφόρμα καθώς και ο σύνδεσμος <Create Application> που οδηγεί στη δημιουργία νέας εφαρμογής.

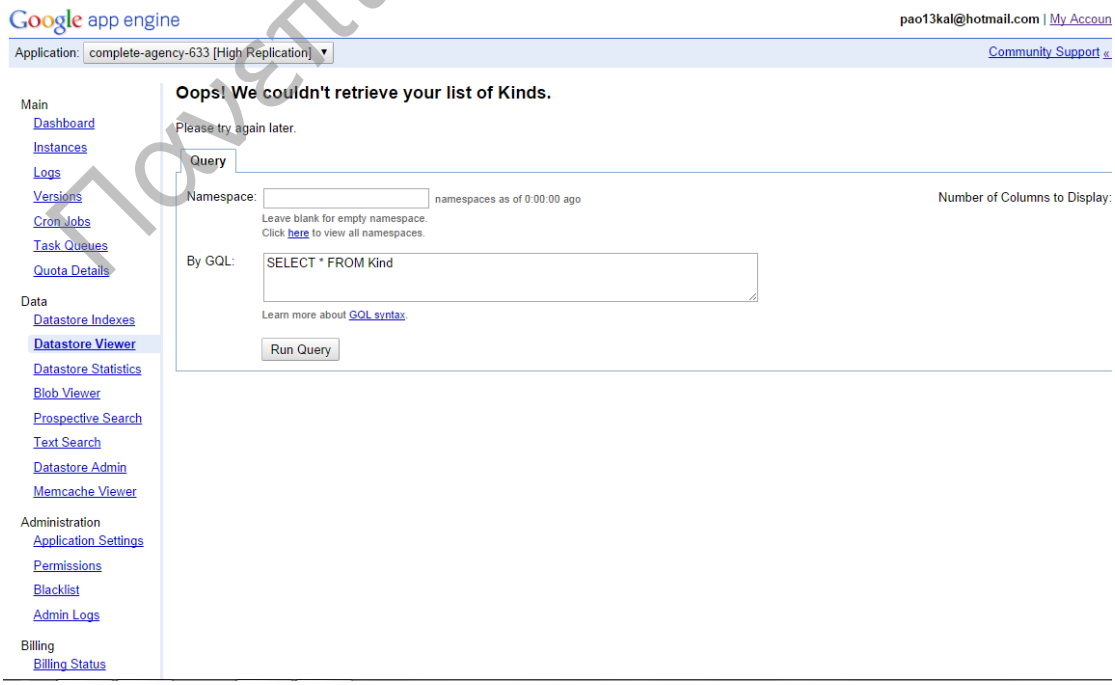
Στη συνέχεια αφού επιλέξουμε ποια εφαρμογή θέλουμε να διαχειριστούμε, π.χ την εφαρμογή με τίτλο ksenodoxeio μας εμφανίζεται η εξής εικόνα

Εικόνα 2



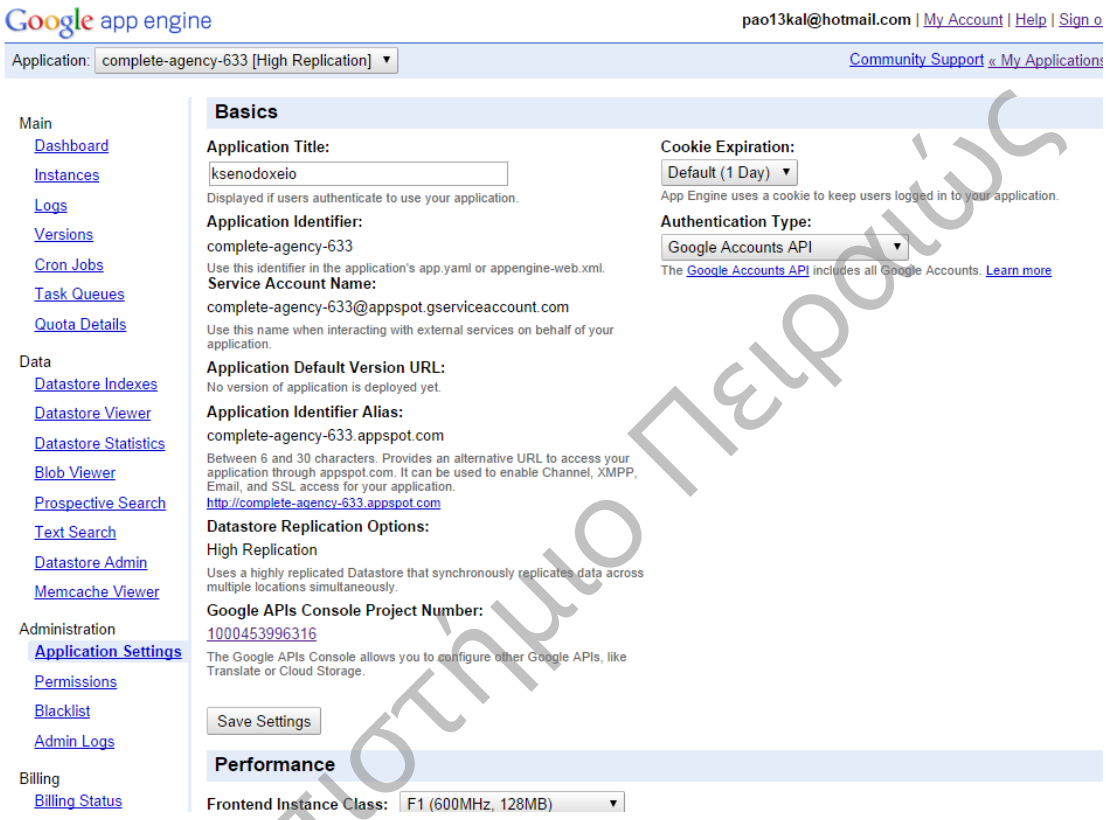
Αυτό είναι το κεντρικό μενού του GAE με πάρα πολλές δυνατότητες, όπως να δούμε την επισκεψιμότητα της εφαρμογής μας , να δούμε την ποσότητα μνήμης που χρησιμοποιούμε καθώς και το χώρο που καταλαμβάνουν οι εγγραφές μας στο Datastore.

Εικόνα 3



Στην εικόνα 3 έχουμε τον Datastore Viewer που μας βοηθά να κάνουμε ερωτήματα σχετικά με τα αντικείμενα της βάσης μας . Η γλώσσα που χρησιμοποιούμε για τα ερωτήματα μοιάζει πάρα πολύ με την SQL αλλά είναι ξεχωριστή για το GAE γι αυτό την ονομάζει GQL

Εικόνα 4



Στην εικόνα 4 φαίνονται διάφορα στοιχεία για την εφαρμογή μας , όπως ο τίτλος της , το αναγνωριστικό της στο GAE καθώς και άλλα χαρακτηριστικά αυθεντικοποίησης.

Τέλος υπάρχουν αρκετοί ακόμη σύνδεσμοι που αφορούν σε περισσότερες δυνατότητες διαχείρισης της εφαρμογής μας αλλά κρίθηκε σκόπιμο να αναλυθούν περισσότερο οι πιο σημαντικές και αυτές που θα χρησιμοποιηθούν περισσότερο από τους χρήστες που θ ανεβάζουν τις εφαρμογές τους στο GAE

Βιβλιογραφία :

1. What is Google App Engine, Google App Engine Google Code, http://code.google.com/intl/elGR/appengine/docs/whatisgoogleappengine.html

2. Developer' s guide – Google App Engine – Google Code, http://code.google.com/intl/el-GR/appengine/docs/

3. Getting Started: PHP – Google App Engine – Google Code, http://code.google.com/intl/el-GR/appengine/docs/php/gettingstarted/