Πανεπιστήμιο Πειραιώς – Τμήμα Πληροφορικής
Πρόγραμμα Μεταπτυχιακών Σπουδών
«Προηγμένα Συστήματα Πληροφορικής»

Μεταπτυχιακή Διατριβή

Τίτλος Διατριβής	**Προώθηση των περιοχών της Ελλάδας με την βοηθεια της κοινοτητας χρηστών μιας «έξυπνης» εφαρμογής διαδικτύου, η οποία χρησιμοποιεί** **-** **Community driven promotion of the areas of Greece through an intelligent web application using web services**
Ονοματεπώνυμο Φοιτητή	**Λογοθέτης Χαράλαμπος**
Πατρώνυμο	**Γεώργιος**
Αριθμός Μητρώου	**ΜΠΣΠ/15047**
Επιβλέπων	**Ευθύμιος Αλέπης, Επίκουρος Καθηγητής**
Βοηθός Επιβλέπων	**Σπυρίδων Παπαδημητρίου, Υποψήφιος Διδάκτορας**

Ημερομηνία Παράδοσης **Παρασκευή 09 Φεβρουαρίου 2018**

Τριμελής Εξεταστική Επιτροπή

(υπογραφή)	(υπογραφή)	(υπογραφή)
Ευθύμιος Αλέπης Επίκουρος Καθηγητής	Μαρία Βίρβου Καθηγήτρια	Γεώργιος Τσιχριντζής Καθηγητής

Ευχαριστίες

Για την ολοκλήρωση της παρούσας διπλωματικής εργασίας θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή κύριο Ευθύμιο Αλέπη και τον βοηθό επιβλέποντα κύριο Σπύρο Παπαδημητριου για την εμπιστοσύνη που μου έδειξαν και για τον χρόνο τους. Επίσης ιδιαίτερες ευχαριστίες θέλω να εκφράσω προς την οικογένεια μου και προς την μέλλουσα γυναίκα μου Νικολέτα για την υπομονή και την στήριξη τους.

Περίληψη

Τα τελευταία χρόνια γίνεται μια διαρκής προσπάθεια εδραίωσης της τουριστικής βιομηχανίας ως βασικό σημείο της ελληνικής οικονομίας. Παράλληλα, ενώ ο τουρισμός και η προώθηση των τουριστικών περιοχών έχουν εντυπωσιακή εξέλιξη σε σύγκριση με παλιότερα, παρατηρείται το φαινόμενο ότι μεγάλος αριθμός ανθρώπων, επιθυμούν να ξεφύγουν από την “κλασσική” τουριστική εμπειρία που υπόσχονται τα πακέτα των τουριστικών γραφείων και αναζητούν τρόπους να βιώσουν τις ιδιαίτερες ομορφιές των τόπων που επισκέπτονται.

Η παρούσα διπλωματική εργασία βασίζεται στην παραπάνω ανάγκη. Έχει ως βασικό άξονα την εναλλακτική τουριστική προβολή των περιοχών της Ελλάδας, μέσα από την οπτική ματιά των κατοίκων ή των επισκεπτών τους, και κύριος σκοπός είναι η καταγραφή και η διάδοση των ιδιαίτερων λεπτομερειών κάθε τόπου, που μόνο κάποιος που έχει ζήσει στην περιοχή μπορεί να γνωρίζει. Για αυτό τον σκοπό υλοποιήθηκε μια web εφαρμογή, μέσα από την οποία μπορούν να προβληθούν όλες οι περιοχές της Ελλάδας, καθώς δίνεται η δυνατότητα στους χρήστες να δημιουργούν ειδικευμένα άρθρα για τους τόπους που επιθυμούν και να βαθμολογούν όσα άρθρα είναι ήδη δημοσιευμένα, συμβάλλοντας έτσι στην σταδιακή δημιουργία ενός δικτύου με “ιδιαίτερες” τουριστικές πληροφορίες. Η κατηγοριοποίηση των γεωγραφικών περιοχών έγινε με βάση τα εξής τέσσερα γεωγραφικά κριτήρια: α) “Περιφέρειες”, β) “Νομοί”, γ) “Δήμοι”, δ) “Πόλεις”. Κάτω από αυτές της κατηγορίες ομαδοποιούνται όλα τα άρθρα που δημιουργούν οι χρήστες. Η εφαρμογή σχεδιάστηκε με “responsive design” έτσι ώστε να μπορεί να διατηρεί την ίδια εμπειρία χρήσης (UI/UX) όταν ο χρήστης πλοηγείται από κινητή συσκευή (smatphone/tablet). Επιπλέον, έχει υλοποιηθεί και restfull api service, με την χρήση του οποίου είναι εφικτή η δημιουργία ενός mobile app που θα έχει πρόσβαση στην ίδια βάση δεδομένων χωρίς να χρειάζεται η ανάπτυξη του business logic από την αρχή. Τέλος, λόγω της δυναμικής και του μεγάλου όγκου πληροφορίας που θα δημιουργήσει η εν λόγω εφαρμογή σε πραγματικό περιβάλλον λειτουργίας, κρίθηκε αναγκαίο και υλοποιήθηκε σύστημα “προτάσεων” προς τον χρήστη, ανάλογα με το ιστορικό των προβολών που έχει πραγματοποιήσει κατά την πλοήγησή του στην εφαρμογή.

Abstract

During the last decade, there is a constant effort to consolidate the tourism industry as a key point of the Greek economy. At the same time, while tourism and the promotion of areas with tourist interest have evolved tremendously, there is a phenomenon that a large number of people want to avoid the "classic" tourist experience that the organized holiday packages and they are looking for other ways to experience the unique beauties of the places they visit.

The current Thesis is based on the tourists' need that was previously mentioned. Its main target is the alternative promotion of the different areas of Greece, through the descriptions of the local inhabitants. For this purpose a web application was created. The users of this application will publish articles of the areas that they want to promote. Also, they will be able to rate all the articles that will have been published up to that time. The classification of the areas was based on the following four geographical criteria: a) Greater Regions, b) Regions, c) Municipalities, d) Cities. Each article belongs to one of these categories.

The application is designed with responsive behavior, so that the user will have the same "user experience" when he is browsing from a mobile device. In addition, a RESTfull api service has been implemented. It serves the requested data to JSON, and that is a very important step for a mobile application that could be developed in the future. Finally, it is worth mentioning that a recommendation system has also been implemented so as to improve even more the user interface and the user's experience.

Περιεχόμενα

1. Εισαγωγή

1.1 Ανάλυση αναγκών

Τα τελευταία χρόνια ο όγκος της ψηφιακής πληροφορίας έχει αυξηθεί δραματικά, ενώ είναι προφανές ότι αυτό θα συνεχίσει να συμβαίνει ολοένα και πιο έντονα τα επόμενα χρόνια. Η αποτελεσματική διαχείριση όλων αυτών των πληροφοριών αποτελεί μια πραγματική πρόκληση και γι' αυτό τον λόγο όσοι τομείς της πληροφορικής σχετίζονται με τα δεδομένα έχουν ιδιαίτερη πρόοδο. Για την εξαγωγή πληροφοριών υπάρχουν εξειδικευμένοι πλέον τρόποι data mining και data analytics, ενώ η μοντελοποίηση των χρηστών και η τεχνητή νοημοσύνη συντελούν στην ομαλότερη και πιο προσιτή εμπειρία χρήσης, μέσα από εξειδικευμένα συστήματα προτάσεων, προσαρμοσμένα πάνω στις συνήθειες και στις ανάγκες του κάθε χρήστη. Πλέον, ο χρόνος για την αναζήτηση οποιασδήποτε πληροφορίας είναι της τάξης των millisecond, ενώ παράλληλα κάποια συστήματα προτάσεων, βασισμένα πάνω σε ειδικούς αλγόριθμους, έχουν την δυνατότητα να σχηματίζουν σχετικά γρήγορα το προφίλ του χρήστη, να καταλαβαίνουν τις συνήθειες και τις επιθυμίες του και να προβαίνουν σε επιτυχημένες προτάσεις, γλιτώνοντας του χρόνο από πιθανές περιττές αναζητήσεις μέσα στον χαοτικά μεγάλο όγκο των πληροφοριών.

Η επιστήμη της πληροφορικής είναι εδώ για να βελτιώσει την ποιότητα ζωής των ανθρώπων καλύπτοντας τις διαρκώς αυξανόμενες καθημερινές ανάγκες. Πιο συγκεκριμένα, για την ελληνική πραγματικότητα, αν και η λεγόμενη "ψηφιακή επανάσταση" δεν φαίνεται να έχει συμβεί ακόμα, παρατηρούνται όλο και περισσότερες προσπάθειες προς αυτό το σκοπό. Διαδικτυακές διαφημίσεις, ηλεκτρονικό εμπόριο, διαδικτυακά μέσα ενημέρωσης, ηλεκτρονική διακυβέρνηση καθώς και πολλοί άλλοι τομείς συμπληρώνουν το παζλ προς την ψηφιακή εποχή. Ο κάθε άνθρωπος έχει διαθέσιμη την απάντηση σχεδόν για κάθε ερώτημά του, σε κλάσματα του δευτερολέπτου. Παρόλα αυτά, δεν έχει ελαττωθεί ούτε στο ελάχιστο η ανάγκη του για τις "δια ζώσης" εμπειρίες, τις οποίες δεν μπορεί να αντικαταστήσει η τεχνολογία επάξια.

Ο τουρισμός είναι μια από αυτές τις εμπειρίες που αναζητεί ο σύγχρονος άνθρωπος. Είναι συνυφασμένη με την περίοδο των διακοπών, όχι απαραίτητα των καλοκαιρινών, και είναι μια ιδανική ευκαιρία για ψυχαγωγία και επαφή με διαφορετικές εικόνες. Η πιο συνηθισμένη διαδικασία για κάποιον που επιθυμεί να ταξιδέψει, είναι να επιλέξει το τόπο και στην συνέχεια να αναζητήσει σχετικές πληροφορίες. Το αναμενόμενο είναι να γίνουν αρκετές αναζητήσεις στο διαδίκτυο, να αντληθούν πληροφορίες από διάφορες πηγές και στο τέλος να δημιουργηθεί ένα οργανόγραμμα, το οποίο θα περιλαμβάνει πληροφορίες για την μεταφορά, την διαμονή, την διατροφή, την διασκέδαση κλπ. Αυτό προϋποθέτει οργάνωση και αρκετό προσωπικό χρόνο, ο οποίος δεν είναι πάντα διαθέσιμος. Έτσι μεγάλος αριθμός ανθρώπων απευθύνονται σε τουριστικά γραφεία, αναζητώντας έτοιμα τουριστικά πακέτα τα οποία τους προσφέρουν μια οργανωμένη ταξιδιωτική εμπειρία με ελάχιστη προσπάθεια από την μεριά τους.

Καθώς οι τάσεις και οι συνήθεις αλλάζουν, μεταβλήθηκε με τα χρόνια και το ενδιαφέρον για τις "εύκολες" διακοπές. Δημιουργήθηκε σιγά σιγά η τάση της "εξερεύνησης", όπου ο ταξιδιώτης ζητάει να ζήσει μια διαφορετική και λιγότερο τετριμμένη εμπειρία κατά τις διακοπές του. Πολλές φορές αναζητεί την δυνατότητα να ζήσει την "καθημερινότητα" ενός τόπου, και όχι απλά ένα στημένο απρόσωπο τουριστικό προϊόν. Κάτι τέτοιο απαιτεί από τον τουρίστα να έρθει σε πιο άμεση επαφή με τους ανθρώπους που κατοικούν μόνιμα στην περιοχή που επισκέπτεται, να τους ζητήσει πληροφορίες και ίσως να προσπαθήσει να μιμηθείς τις συνήθειές τους και τους ρυθμούς της καθημερινότητάς τους. Σε αυτό το σημείο αναδύεται μια εντελώς διαφορετική ανθρώπινη ανάγκη, ένα τμήμα της οποίας θα προσπαθήσει να καλύψει η παρούσα εργασία.

1.2 Σύντομη περιγραφή της ανάπτυξης του λογισμικού

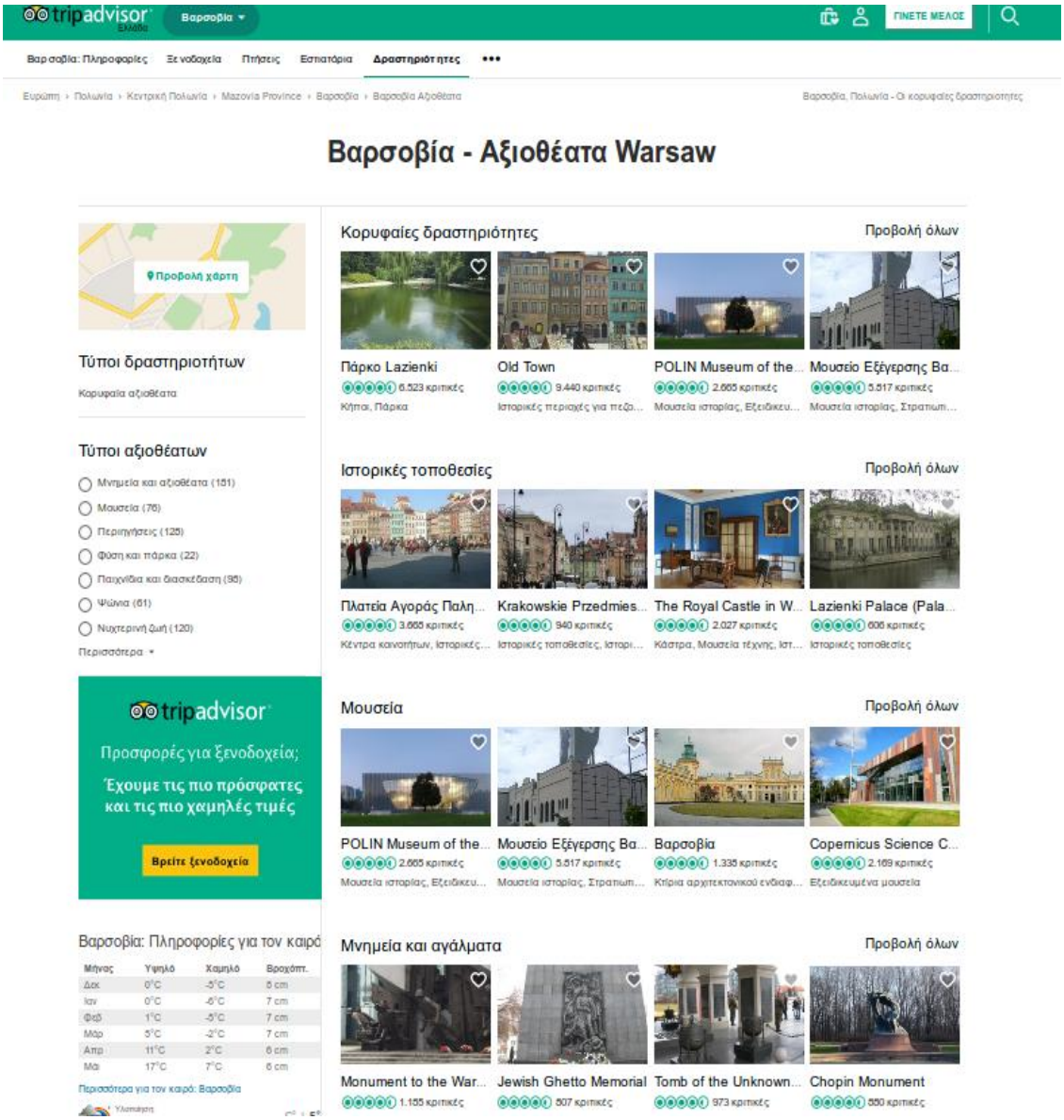
Όπως αναφέρθηκε ήδη, το πρώτο βήμα στην ανάπτυξη της εφαρμογής είναι η αναγνώριση και η καταγραφή-ανάλυση της ανθρώπινης ανάγκης που θα προσπαθήσει να καλύψει η εφαρμογή. Το δεύτερο βήμα είναι ο έλεγχος της αγοράς την τρέχουσα στιγμή ώστε να διαπιστωθεί πως καλύπτεται ως τώρα αυτή η ανάγκη από τις υπάρχουσες εφαρμογές, τις διαφορές τους, τι προσφέρουν στον τελικό χρήστη και με ποιόν τρόπο προσεγγίζουν το πρόβλημα. Αυτό το βήμα περιγράφεται αναλυτικά στην επόμενη ενότητα. Το επόμενο βήμα είναι η αναλυτική περιγραφή της εφαρμογής σε κείμενο. Χρειάζεται να καταγραφούν τι θα εμφανίζονται για παράδειγμα στην αρχική οθόνη και τι λειτουργίες θα επιτελούνται όταν γίνεται κλικ σε κάποιο κουμπί. Με αυτό τον τρόπο γίνονται ξεκάθαρες από την αρχή οι λειτουργίες που προσφέρει η εφαρμογή, και έτσι μπορούν ευκολότερα να εντοπιστούν τυχόν παραλήψεις ή λάθη.

Στην συνέχεια γίνεται η ανάλυση των σεναρίων χρήσης και ο χωρισμός τους σε μικρότερα τμήματα. Σε αυτό το σημείο αποφασίζεται με ποια τεχνολογία θα υλοποιηθεί κάθε λειτουργικότητα καθώς και το πως μπορεί να απλοποιηθεί σε λιγότερα βήματα εφόσον κάτι τέτοιο είναι εφικτό. Έπειτα ακολουθεί η τμηματική ανάπτυξη της εφαρμογής σε επίπεδο κώδικα παράλληλα με τις δοκιμές στο τέλος κάθε ενότητας ώστε να διαπιστωθούν τυχόν ελλείψεις. Αξίζει να αναφερθεί ότι στην περίπτωση που η εφαρμογή εκτελεστεί σε πραγματικό περιβάλλον διαδικτύου θα πρέπει να επιλεγούν και άλλες τεχνολογίες που αφορούν τον server ο οποίος θα φιλοξενήσει την εφαρμογή, όπως το λειτουργικό που θα χρησιμοποιεί, τα services που θα τρέχουν παράλληλα με την εφαρμογή και τα πιστοποιητικά κρυπτογράφησης που θα χρησιμοποιηθούν. Στα πλαίσια αυτής της διπλωματικής δεν αναπτύχθηκαν περαιτέρω.

1.3 Ανασκόπηση πεδίου

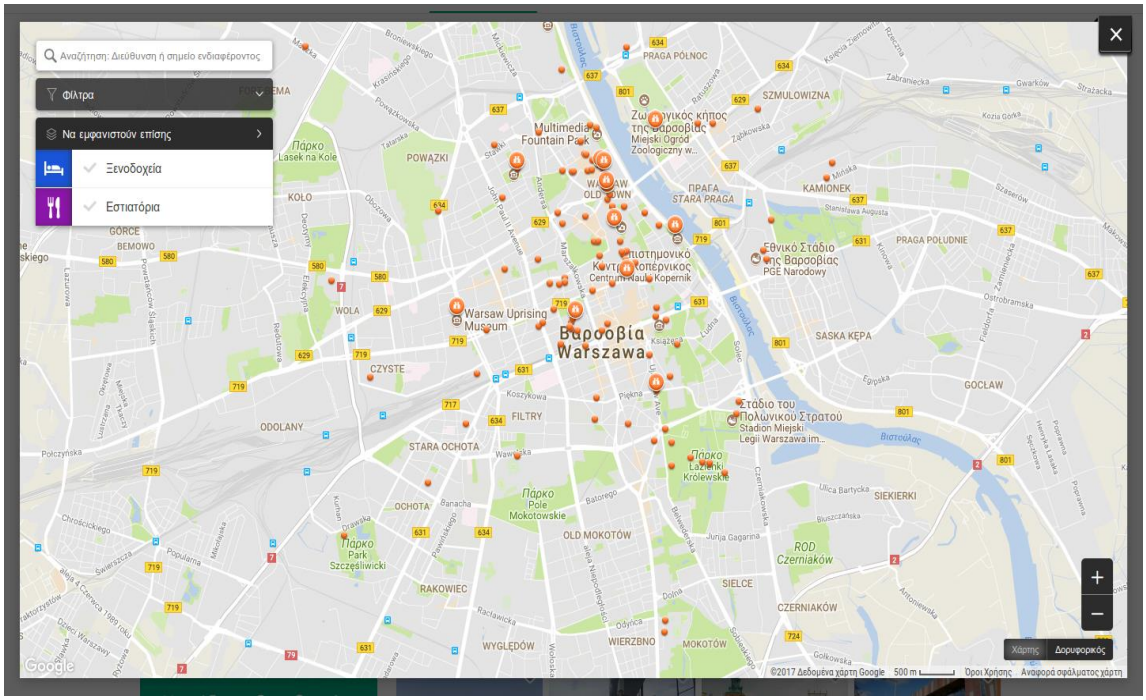
Στην συνέχεια θα παρατεθούν κάποιες από τις εμπορικές εφαρμογές, οι οποίες έχουν κοινά χαρακτηριστικά με την εφαρμογή που αναπτύχθηκε για αυτή τη διπλωματική εργασία. Οι εφαρμογές που θα παρουσιαστούν είναι διαδικτυακές, διότι το διαδίκτυο είναι το μέσο που στηρίζει και προωθεί τέτοιου είδους λογισμικά.

Ξεκινώντας, η δημοφιλέστερη εμπορική εφαρμογή που προσφέρει παρόμοια εμπειρία χρήσης, είναι η “https://www.tripadvisor.com” σύμφωνα με τις μετρήσεις του “https://www.alexa.com”. Η εφαρμογή Tripadvisor, έχει παρουσία και στην Ελλάδα, όχι απλά μεταφράζοντας το περιεχόμενο της στην ελληνική γλώσσα, αλλά έχοντας ενεργή συνεργασία με επιχειρήσεις. Όπως αναφέρεται στην ιστοσελίδα της, προσφέρει ενημέρωση για τιμές ξενοδοχείων, αεροπορικών εισιτηρίων και εστιατορίων, ενώ επίσης παρουσιάζει κριτικές και σχόλια από τους εγγεγραμμένους χρήστες της πλατφόρμας. Επιπλέον υπάρχει ένα ξεχωριστό τμήμα της εφαρμογής που αναφέρεται μόνο στα αξιοθέατα και τις δραστηριότητες της περιοχής την οποία αναζήτησε ο χρήστης. Τα αξιοθέατα είναι ομαδοποιημένα σε γενικές κατηγορίες όπως “Μουσεία”, “Συναυλίες”, “Υπαίθριες Δραστηριότητες” και πολλά άλλα τα οποία φιλτράρουν αυτόματα, και σε ελάχιστο χρονικό διάστημα, τα αποτελέσματα τα οποία επιθυμεί να δει ο χρήστης της σελίδας (Εικόνα 1). Οι κριτικές και τα σχόλια των χρηστών του Tripadvisor υπάρχουν και εδώ μαζί με μια μεσοπρόθεσμη πρόβλεψη για τον καιρό που επικρατεί στην περιοχή. Πολύ χρήσιμος είναι επίσης και ο διαδραστικός χάρτης που παρουσιάζει την ευρύτερη περιοχή, πάνω στον οποίο εμφανίζονται όλα τα σχετικά σημεία ενδιαφέροντος (Εικόνα 2).



Εικόνα 1: Tripadvisor - Σελίδα με την λίστα των δραστηριοτήτων (εμφανίζονται επίσης τα φίλτρα, ο χαρτης της περιοχής και οι πληροφορίες για τον καιρό)

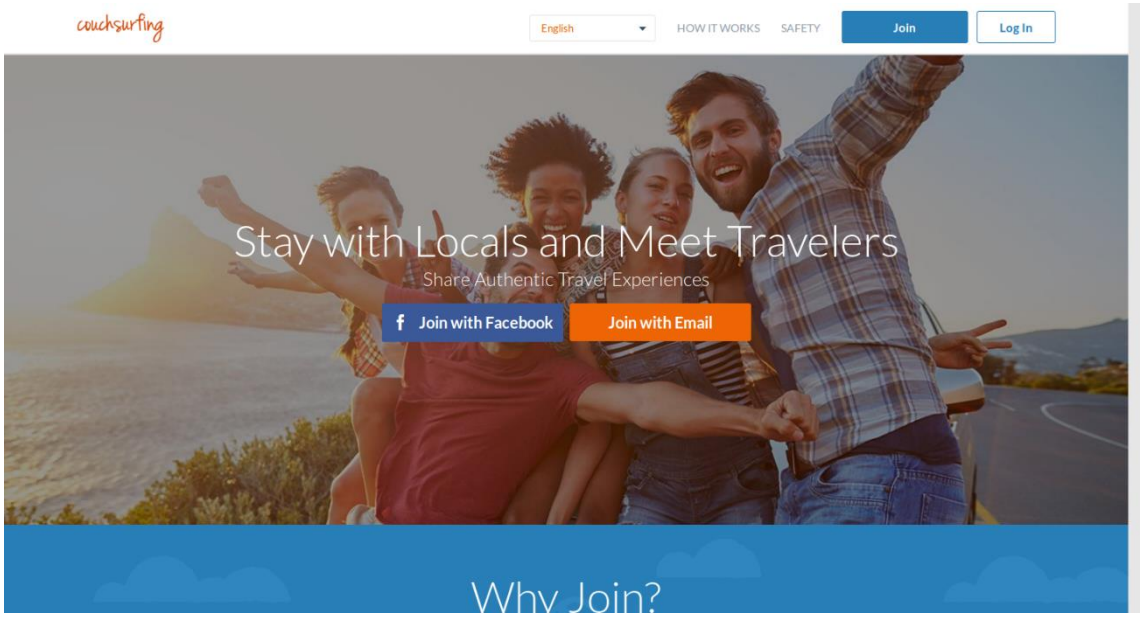
Προώθηση των περιοχών της Ελλάδας με την βοηθεια της κοινοτητας χρηστών μιας «έξυπνης» εφαρμογής, η οποία χρησιμοποιεί υπηρεσίες διαδικτύου



Εικόνα 2: Tripadvisor - Διαδραστικός χάρτης της περιοχής με τα σημεία ενδιαφέροντος

Η δεύτερη εμπορική εφαρμογή που είναι παρόμοια με την εφαρμογή της παρούσας διπλωματικής εργασίας είναι η “https://www.couchsurfing.com”. Η συγκεκριμένη εφαρμογή, είναι ένα κοινωνικό δίκτυο που έχει ως σκοπό να φέρει όσους σκοπεύουν να ταξιδέψουν σε άμεση επαφή με τους μόνιμους κάτοικους της περιοχής. Δίνει την δυνατότητα σε εγγεγραμμένους χρήστες, να αποκτήσουν νέους φίλους από διάφορες περιοχές σε όλο τον κόσμο, να συζητήσουν και να ενημερωθούν για τις δραστηριότητες που υπάρχουν στην περιοχή που σκοπεύουν να επισκεφθούν. Το ιδιαίτερο σημείο που κάνει την συγκεκριμένη εφαρμογή να ξεχωρίζει από άλλα κοινωνικά δίκτυα είναι η δυνατότητα που δίνει στους χρήστες της να φιλοξενήσουν στο σπίτι τους ή να φιλοξενηθούν από άλλους χρήστες. Μέσα από την πλατφόρμα, ο κάθε χρήστης μπορεί να δηλώσει ότι έχει την δυνατότητα να διαθέσει για συγκεκριμένες ημέρες κάποιο από τα δωμάτια του σπιτιού του ώστε να φιλοξενηθεί κάποιο από τα υπόλοιπα μέλη της πλατφόρμας. Η διαδικασία περιέχει κάποιες δικλείδες ασφαλείας ώστε να είναι καλυμμένος τόσο ο ιδιοκτήτης του σπιτιού, όσο και ο φιλοξενούμενος. Κάθε προφίλ χρήστη περιέχει αξιολογήσεις και σχόλια από άλλους χρήστες, πράγμα που διατηρεί την ασφάλεια των χρηστών καθώς και την αξιοπιστία της ίδιας της εφαρμογής. Επίσης υπάρχουν πολλές ενεργές κοινότητες που ανανεώνουν διαρκώς τις πληροφορίες που αφορούν δραστηριότητες και σημεία ενδιαφέροντος για την περιοχή τους. Επίσης οργανώνουν συναντήσεις με σκοπό να γνωρίζονται οι χρήστες από κοντά και να συντηρούν το κλίμα της “ομάδας”.

Εκτός από το στοιχείο της “άμεσης ανθρώπινης επαφής” το οποίο προστίθεται στην προσφερόμενη ταξιδιωτική εμπειρία, η εφαρμογή “https://www.couchsurfing.com” δημιουργεί (μέσω των χρηστών της) μια διαδικτυακή βάση πληροφοριών, αποτελούμενη από προσωπικές εμπειρίες και γνώμες καθημερινών ανθρώπων, και οι οποίες κατηγοριοποιούνται με γεωγραφικά κριτήρια. Έτσι μελλοντικοί χρήστες έχουν πρόσβαση σε όλες αυτές τις πληροφορίες ενώ παράλληλα συνεχίζουν να προστίθενται ολοένα και περισσότερες. Στην επόμενη εικόνα (Εικόνα



Εικόνα 3: CouchSurfing - Η αρχική σελίδα της εφαρμογής

2. Παρουσίαση και χρήση της εφαρμογής

2.1 Γενική επισκόπηση

Για την παρούσα διπλωματική διατριβή, αναπτύχθηκε μια εφαρμογή διαδικτύου η οποία για λόγους καλύτερης ανάλυσης διαχωρίστηκε σε δυο μικρότερα τμήματα. Το πρώτο υλοποιήθηκε για να καλύψει τις ανάγκες που περιγράφηκαν στο προηγούμενο κεφάλαιο , δηλαδή την πλοήγηση στις τουριστικές πληροφορίες, προσφέροντας παράλληλα και το περιβάλλον που θα χρησιμοποιεί ο τελικός χρήστης. Το δεύτερο τμήμα, είναι μια βοηθητική εφαρμογή διαχείρισης περιεχομένου. Σε αυτήν δεν έχουν πρόσβαση οι χρήστες της προηγούμενης εφαρμογής, αλλά μόνο οι administrators. Υλοποιήθηκε ως ένα εργαλείο για τους διαχειριστές, ώστε να μπορούν να προσθέτουν, επεξεργάζονται και να αφαιρούν περιεχόμενο από την βάση δεδομένων της βασικής εφαρμογής εύκολα και άμεσα. Στις ενότητες που ακολουθούν, παρουσιάζονται και οι δυο ξεχωριστά, με αναλυτικές πληροφορίες και σενάρια χρήσης.

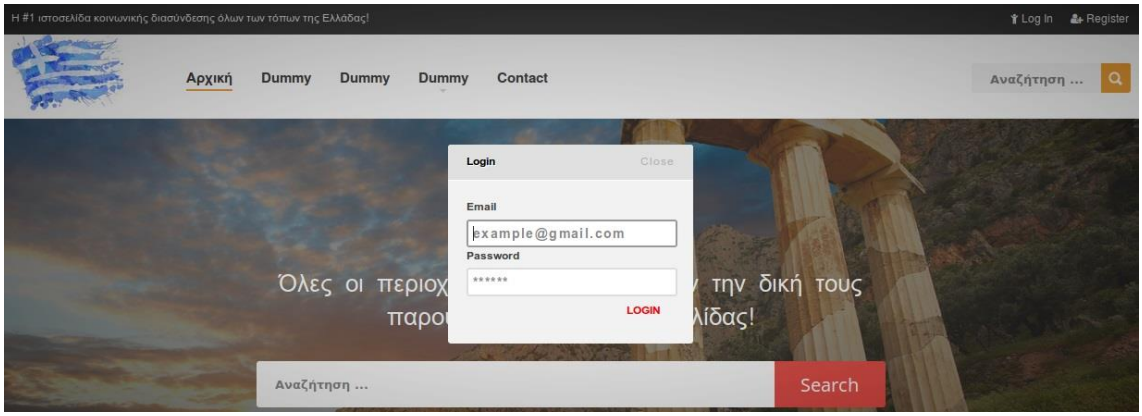
2.2 Παρουσίαση της κύριας εφαρμογής

2.2.1 Εισαγωγή

Καθώς η συγκεκριμένη εφαρμογή είναι διαδικτυακή, η βασικότερη σελίδα της είναι αυτή που βλέπει πρώτη ο χρήστης. Η αρχική σελίδα δίνει πρόσβαση σε όλες τις επιμέρους λειτουργίες της εφαρμογής και περιέχει μια συνοπτική παρουσίαση των βασικότερων τμημάτων της εφαρμογής. Ωστόσο, για να γίνει αντιληπτή συνολικά η δομή της εφαρμογής, θα παρουσιαστούν σε ξεχωριστές ενότητες όλες οι σελίδες με τις οποίες θα μπορεί να αλληλεπιδράσει ο τελικός χρήστης.

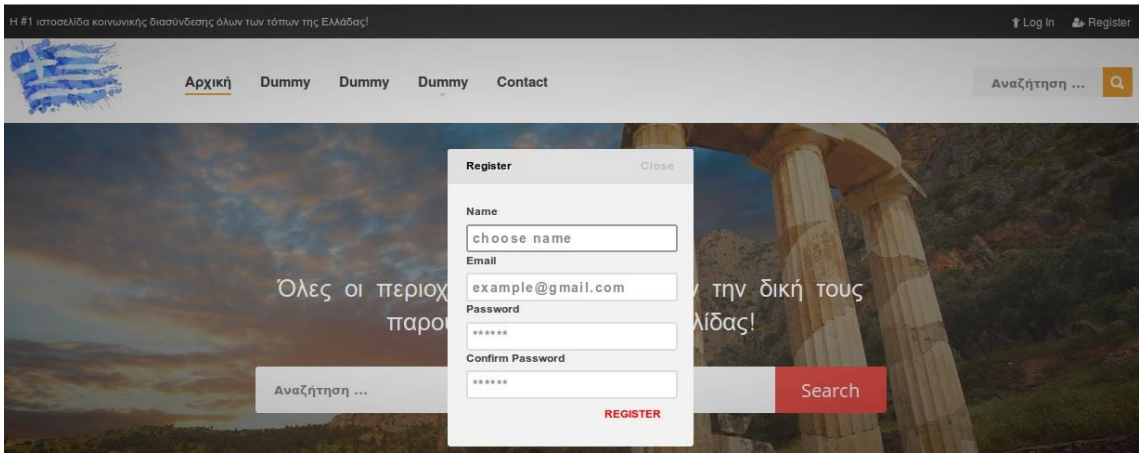
2.2.2 Αρχική σελίδα

Ξεκινώντας την ανάλυση της αρχικής σελίδας, χρειάζεται να γίνει ένας διαχωρισμός σε επιμέρους τμήματα. Ξεκινώντας από το επάνω μέρος της σελίδας, διακρίνεται ένα πλαίσιο με μαύρο φόντο, στα αριστερά του οποίου υπάρχει ένας χαιρετισμός και στα δεξιά υπάρχουν δύο σύνδεσμοι, ένας για την σύνδεση του χρήστη στην εφαρμογή με τους προσωπικούς του κωδικούς και ένας για την εγγραφή των νέων χρηστών στην εφαρμογή. Ο χρήστης, κάνοντας κλικ σε κάποια από τις δύο επιλογές, ενεργοποιεί το αντίστοιχο αναδυόμενο παράθυρο, όπου του ζητείται να εισάγει τα κατάλληλα στοιχεία ώστε να μπορέσει να συνεχίσει. Στην συνέχεια ακολουθούν εικόνες στις οποίες παρουσιάζονται οι παραπάνω περιπτώσεις. Στην εικόνα 4 διακρίνεται η λειτουργικότητα της σύνδεσης (Log In) ενός χρήστη ο οποίος έχει εγγραφεί στο παρελθόν και έχει αποκτήσει προσωπικούς κωδικούς. Αρκεί να συμπληρώσει το προσωπικό του email και τον προσωπικό του μυστικό κωδικό (password) που δήλωσε κατά την εγγραφή του, και στην συνέχεια να κάνει κλικ στο “LOGIN” που διακρίνεται με κόκκινους χαρακτήρες στο κάτω δεξιά μέρος του πλαισίου.



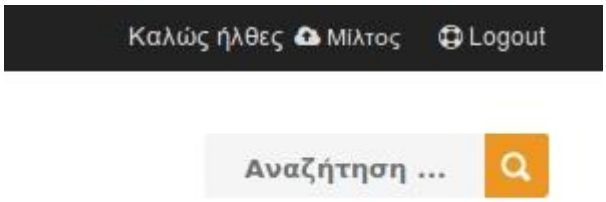
Εικόνα 4: Σύνδεση χρήστη (Log in)

Στην επόμενη εικόνα 5, παρουσιάζεται η λειτουργικότητα της εγγραφής ενός νέου χρήστη στην εφαρμογή. Σε αυτή την περίπτωση, ο χρήστης καλείται να συμπληρώσει τα απαιτούμενα πεδία και να κάνει κλικ στο “REGISTER” που διακρίνεται με κόκκινους χαρακτήρες στο κάτω δεξιά μέρος του πλαισίου.



Εικόνα 5: Εγγραφή νέου χρήστη (Register)

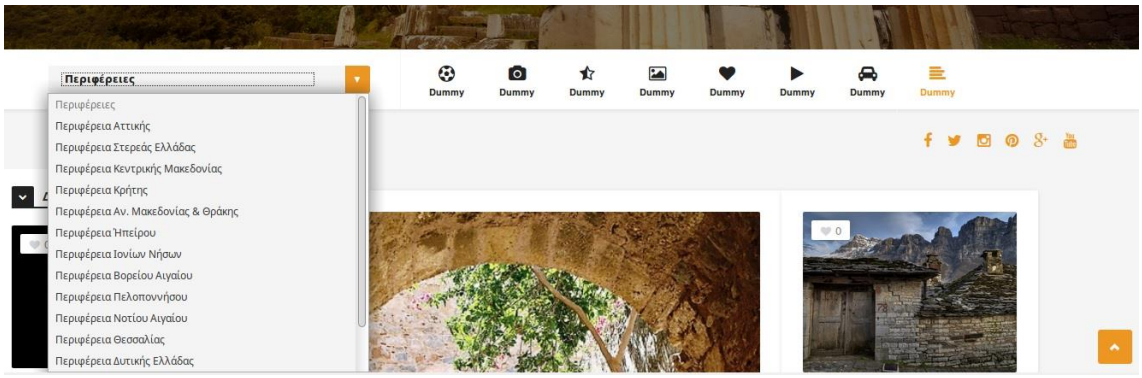
Και για τις δυο παραπάνω περιπτώσεις, όταν η διαδικασία ολοκληρωθεί επιτυχώς, ο χρήστης παρατηρεί μια επαναφόρτωση της σελίδας, ενώ πλέον στο επάνω δεξιά μέρος της σελίδας, εμφανίζεται το όνομα του και το κουμπί της αποσύνδεσης (εικόνα 6).



Εικόνα 6: Απεικόνιση κατάστασης μετά την επιτυχή σύνδεση του χρήστη

Συνεχίζοντας την ανάλυση της αρχικής σελίδα, διακρίνεται μια ελληνική σημαία, η οποία εμφανίζεται σε όλες τις σελίδες τις εφαρμογής. Ο ρόλος της είναι να ανακατευθύνει τον χρήστη στην αρχική σελίδα από οποιαδήποτε σελίδα και αν βρίσκεται ανα πάσα στιγμή. Δεξιότερα, υπάρχει το μενού της εφαρμογής, το οποίο στην παρούσα φάση έχει ενεργά μόνο δύο links, αυτό που ανακατευθύνει τον χρήστη στην αρχική σελίδα και εκείνο που τον οδηγεί στην σελίδα επικοινωνίας (contact page). Τα υπόλοιπα έχουν παραμείνει για να υποστηρίξουν πιθανές μελλοντικές λειτουργικότητες. Στο ίδιο ύψος της σελίδας,στο δεξί τμήμα καθώς επίσης και ακριβώς πιο κάτω, υπάρχει η δυνατότητα αναζήτησης της εφαρμογής, μια από τις βασικότερες πλέον λειτουργικότητες που συναντιόνται πλέον σε κάθε λογισμικό. Εδώ, ο χρήστης μπορεί να αναζητήσει κείμενο που περιέχεται σε όλα τα άρθρα που υπάρχουν αποθηκευμένα.

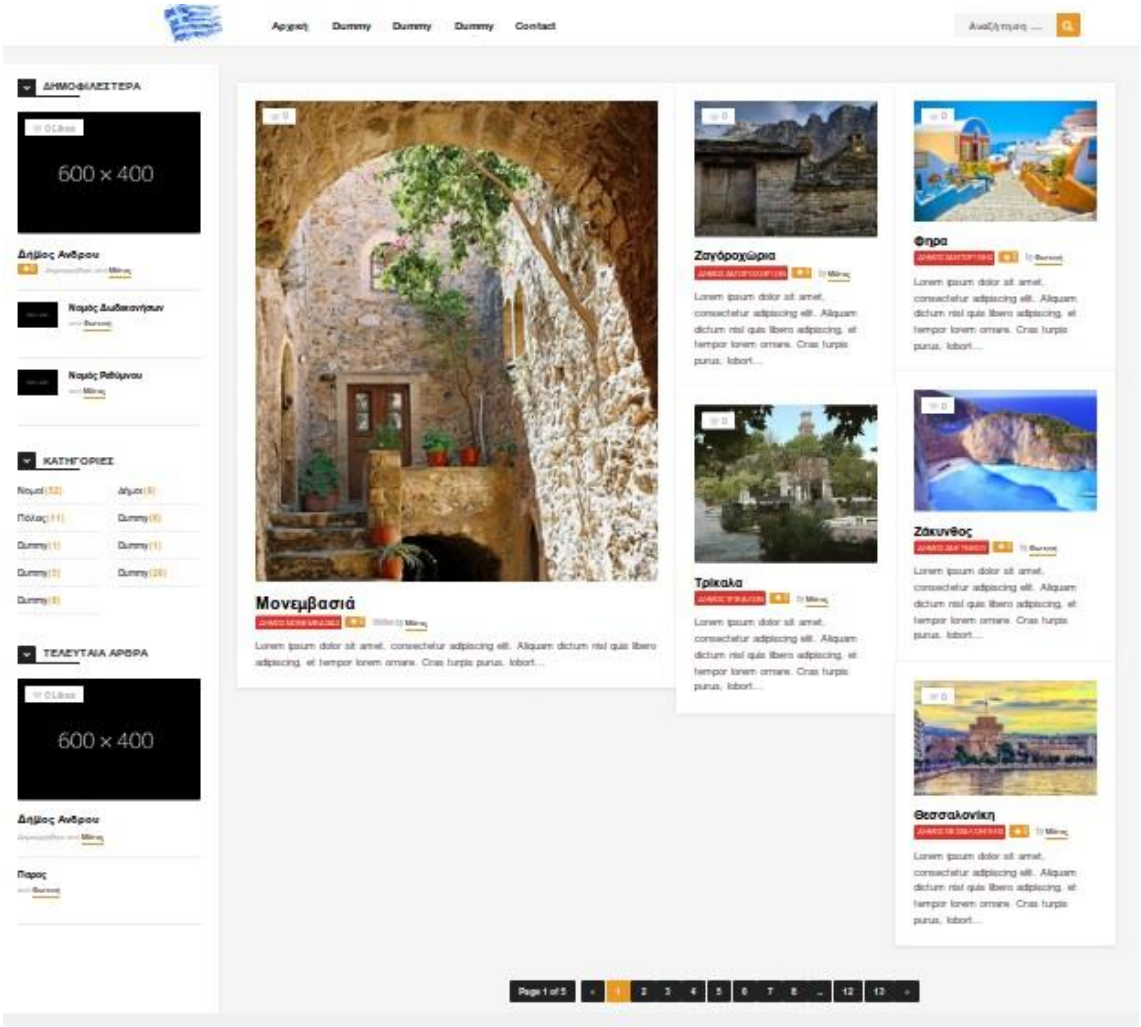
Στην συνέχεια υπάρχει ένα δευτερεύον μενού, στο δεξί μέρος του οποίου υπάρχει ένα πλαίσιο επιλογών και στο αριστερό μέρος υπάρχουν κάποια εικονίδια, ανενεργά προς το παρόν, και τα οποία έχουν παραμείνει για να υποστηρίξουν πιθανές μελλοντικές λειτουργικότητες. Το πλαίσιο επιλογών εμφανίζεται μόλις ο χρήστης κάνει κλικ επάνω σε αυτό, και εμφανίζει όλες τις διαθέσιμες “Περιφέρειες” της ελληνικής επικράτειας (εικόνα 7). Κάνοντας κλικ σε οποιαδήποτε από αυτές, φορτώνεται η αντίστοιχη σελίδα με όλα τα σχετικά άρθρα της συγκεκριμένη περιφέρειας.



Εικόνα 7: Ανοιχτό πλαίσιο επιλογών “Περιφέρειες”

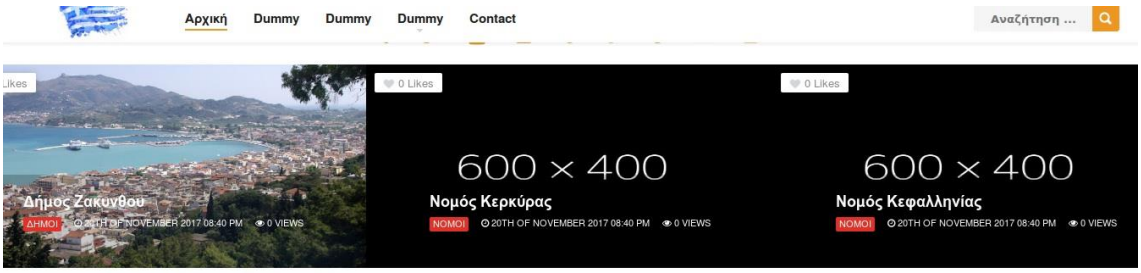
Το επόμενο τμήμα της αρχική σελίδας είναι το βασικότερο, αφού εδώ παρουσιάζονται σημαντικές πληροφορίες. Στο δεξί τμήμα, το οποίο καταλαμβάνει και τον μεγαλύτερο χώρο, εμφανίζονται έξι τυχαία άρθρα με τις βασικές εικόνες τους, τον τίτλο, ένα μικρό τμήμα του κειμένου τους, το όνομα του χρήστης που το δημοσίευσε, την αξιολόγηση και τα likes που έχουν συγκεντρώσει συνολικά. Στο κάτω μέρος υπάρχει λειτουργικότητα που υποστηρίζει σελιδοποίηση των αποτελεσμάτων. Εκεί εμφανίζεται ο συνολικό αριθμός σελίδων και η τρέχουσα σελίδα. Ο χρήστης μπορεί να αλλάζει σελίδα αποτελεσμάτων χωρίς να φεύγει από την αρχική σελίδα της εφαρμογής. Έτσι μπορεί να προβάλει σε βήματα, όλα τα αποτελέσματα των άρθρων καθώς αυτά θα εμφανίζονται ανά έξι, σε διαφορετικές όμως σελίδες. Το αριστερό τμήμα, περιλαμβάνει τρεις υποενότητες. Στην πρώτη, η οποία έχει τίτλο “ΔΗΜΟΦΙΛΕΣΤΕΡΑ”, εμφανίζονται τα τρία δημοφιλέστερα άρθρα, με κριτήριο την επισκεψιμότητά τους, Στην δεύτερη, με τίτλο “ΚΑΤΗΓΟΡΙΕΣ”, εμφανίζονται οι τρεις βασικές κατηγορίες άρθρων, δηλαδή “Νομοί”, “Δήμοι” και “Πόλεις”, ενώ έχουν μείνει ανενεργές και άλλες επιλογές για μελλοντική επέκταση της εφαρμογής. Στην τρίτη υποενότητα, με τίτλο “ΤΕΛΕΥΤΑΙΑ ΑΡΘΡΑ” εμφανίζονται τα τρία άρθρα

με την πιο πρόσφατη ημερομηνία δημοσίευσης, ανεξάρτητα από την κατηγορία στην οποία υπάγονται. Ακολουθεί εικόνα (εικόνα 8) όπου διακρίνονται όλα τα παραπάνω.



Εικόνα 8: Κεντρικό τμήμα της αρχικής σελίδας

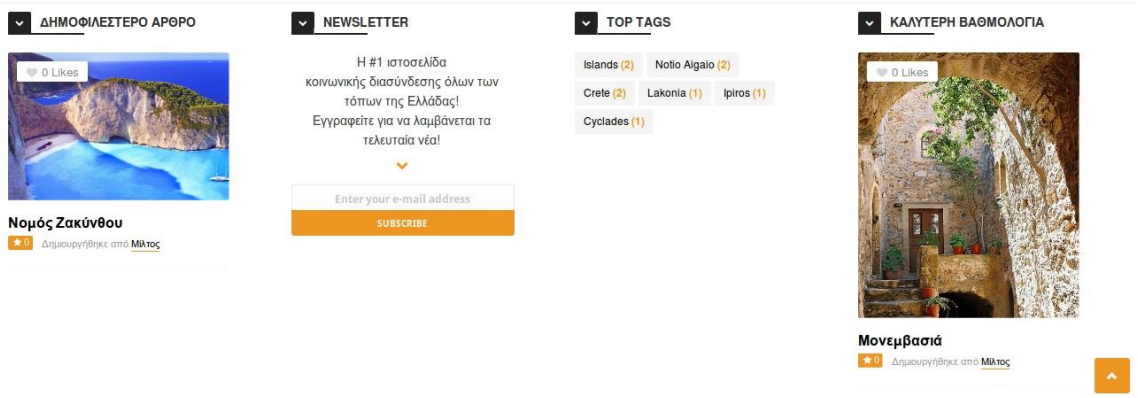
Στην συνέχεια υπάρχει ένα slider (εικόνα 9), όπου εμφανίζονται δέκα προτεινόμενα άρθρα. Σε αυτό το σημείο αξίζει να αναφερθεί ότι αν ο χρήστης δεν είναι εγγεγραμμένος και δεν έχει κάνει σύνδεση την τρέχουσα στιγμή, τότε στο συγκεκριμένο σημείο θα εμφανιστούν τυχαία άρθρα. Τα άρθρα εμφανίζονται εδώ με την βασική τους εικόνα, τον τίτλο τους, την κατηγορία τους, την ημερομηνία δημοσίευσή τους, το όνομα του χρήστη που τα δημοσίευσε και τον συνολικό αριθμό των επισκέψεων που είχαν από όλους τους χρήστες της εφαρμογής. Ο χρήστης μπορεί να σύρει με το ποντίκι τα άρθρα, είτε αριστερά είτε δεξιά, και έτσι να εμφανίσει και τα υπόλοιπα στην οθόνη του.



Εικόνα 9: Slider της αρχικής σελίδας

Προώθηση των περιοχών της Ελλάδας με την βοηθεια της κοινοτητας χρηστών μιας «έξυπνης» εφαρμογής, η οποία χρησιμοποιεί υπηρεσίες διαδικτύου

Το τελευταίο τμήμα της αρχικής σελίδα είναι αυτό που παρουσιάζεται στην εικόνα 10. Διακρίνεται σε 4 υποενότητες. Ξεκινώντας από αριστερά, υπάρχει μια ενότητα με τίτλο “ΔΗΜΟΦΙΛΕΣΤΕΡΟ ΑΡΘΡΟ” όπου παρουσιάζεται εκείνο το άρθρο που έχει τις περισσότερες προβολές. Επειδή έχει ήδη αναφερθεί προηγουμένως η ύπαρξη και μιας άλλη ενότητα με δημοφιλή άρθρα, χρειάζεται να ξεκαθαριστεί ότι το ΄δημοφιλέστερο άρθρο που εμφανίζεται εδώ δεν εμφανίζεται και στην άλλη ενότητα. Συνεχίζοντας, στην ενότητα με τίτλο “NEWSLETTER” ο χρήστης μπορεί να εγγραφεί στις λίστες ενημέρωσης της εφαρμογής καταχωρώντας απλά το email του, ώστε να λαμβάνει από τους διαχειριστές ενημερωτικά μηνύματα. Η επόμενη ενότητα με τίτλο “TOP TAGS” εμφανίζει τα δημοφιλέστερα tag τις εφαρμογής. Ως tag νοείται ένα χαρακτηριστικό που κατηγοριοποιεί παρεμφερή άρθρα, κάνοντας έτσι ευκολότερη την αναζήτηση τους. Κάνοντας κλικ σε κάποιο από αυτά τα tags φορτώνεται η σελίδα με τα άρθρα που υπάγονται σε αυτό το tag. Τελευταία υποενότητα είναι αυτή με το όνομα “ΚΑΛΥΤΕΡΗ ΒΑΘΜΟΛΟΓΙΑ”. Εδώ εμφανίζεται το άρθρο που έχει συνολικά την καλύτερη βαθμολογία. Η βαθμολογία είναι ένα χαρακτηριστικό των άρθρων που επιδρά στην κατάταξή τους, και μπορεί να επηρεαστεί μόνο από όσους χρήστες είναι συνδεδεμένοι με τους προσωπικούς τους κωδικούς. Για κάθε χρήστη καταχωρείται μόνο μία ψήφος, ενώ ένας χρήστης που έχει ήδη ψηφίσει μπορεί να αλλάξει την ψήφο του, είτε προς το καλύτερο είτε προς το χειρότερο. Περισσότερα στοιχεία για την ψηφοφορία δίνονται σε επόμενη ενότητα. Κλείνοντας, αξίζει να αναφερθεί ότι το τετράγωνο κίτρινου χρώματος στο κάτω δεξιά μέρος της εικόνας 10, που περικλείει ένα βέλος που δείχνει προς τα επάνω, εμφανίζεται σε όλες τις σελίδες της εφαρμογής. Εμφανίζεται αυτόματα μόλις ο χρήστης ξεκινήσει να περιηγείται προς τα κάτω στην εκάστοτε σελίδα. Κάνοντας κλικ επάνω του, ο χρήστης επανέρχεται άμεσα στο πάνω μέρος της σελίδας, χωρίς να χρειαστεί να χρησιμοποιήσει τις μπάρες κύλισης του browser ή την ρόδα κύλισης του ποντικιού που χειρίζεται.

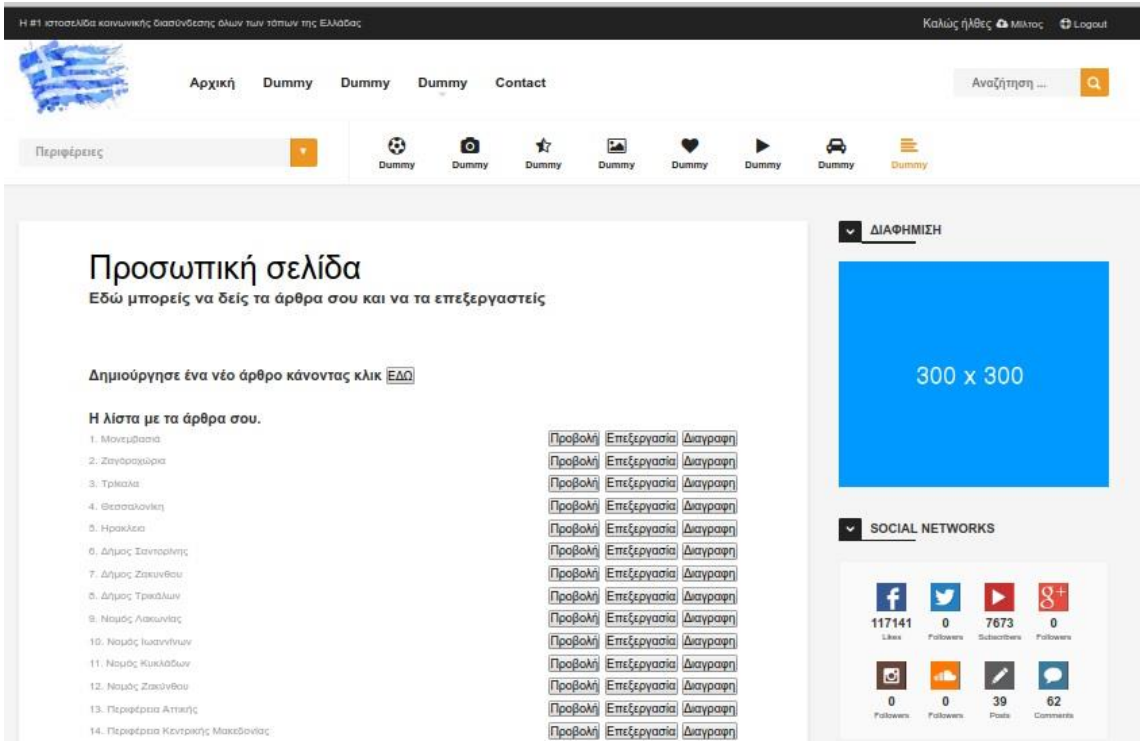


Εικόνα 10: Τελευταίο τμήμα της αρχικής σελίδας.

2.2.3 Προσωπική σελίδα

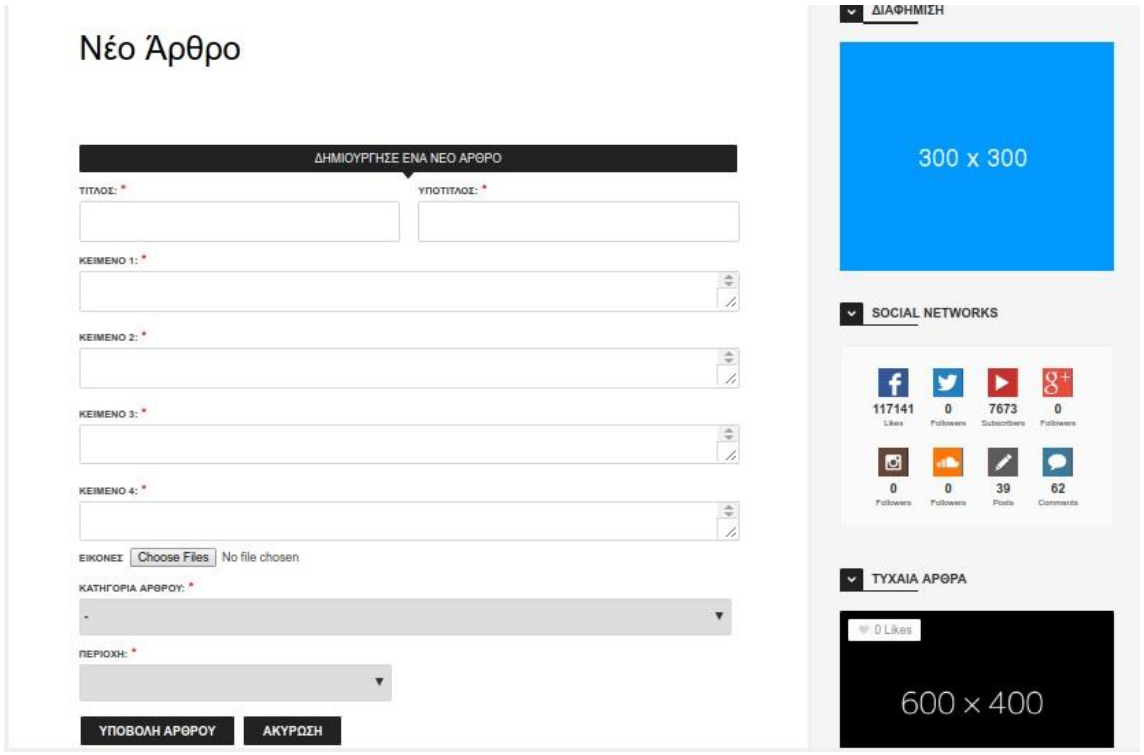
Όπως αναφέρθηκε και σε προηγούμενο σημείο, κατά την σύνδεση (login) του χρήστη με τους προσωπικούς του κωδικούς, γίνεται επαναφόρτωση της σελίδας. Αυτό έχει ως αποτέλεσμα να εμφανίζονται ένα μήνυμα που συνοδεύει το όνομα του χρήστη και το κουμπί αποσύνδεσης (εικόνα 6). Ο χρήστης σε αυτό το σημείο έχει την δυνατότητα να κάνει κλικ πάνω στο όνομα του και να κατευθυνθεί στην προσωπική του σελίδα.

Η προσωπική σελίδα του κάθε χρήστη είναι εμφανή μόνο στον ίδιο. Εδώ εμφανίζονται όλα τα άρθρα που έχει δημοσιεύσει ο χρήστης και παράλληλα του δίνεται η δυνατότητα να τα προβάλει ή και να τα επεξεργαστεί αν το επιθυμεί. Επίσης του δίνεται η δυνατότητα να τα διαγράψει κάποιο από τα άρθρα που έχει δημοσιεύσει στο παρελθόν καθώς και να δημιουργήσει νέα άρθρα.



Εικόνα 11: Προσωπική σελίδα

Εάν ο χρήστης επιλέξει να δημιουργήσει ένα νέο άρθρο κάνοντας κλικ στο αντίστοιχο σημείο, τότε ανακατευθύνεται στην σελίδα που απεικονίζεται στην εικόνα 12.



Εικόνα 12: Σελίδα δημιουργίας νέου άρθρου

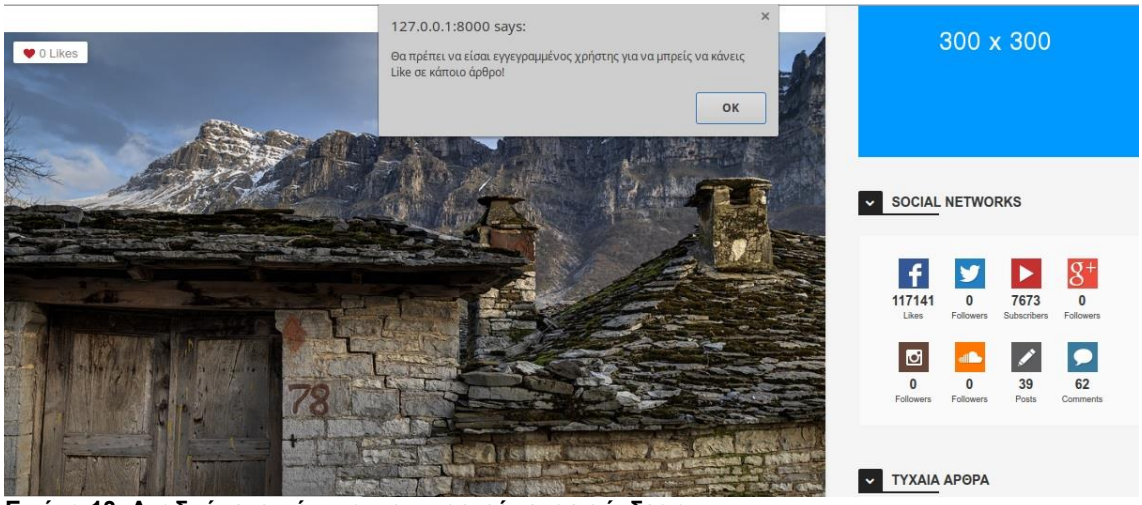
Σε αυτή την σελίδα ο χρήστης καλείται να συμπληρώσει τα απαιτούμενα πεδία και να επισυνάψει ως 5 εικόνες σχετικές με το άρθρο. Τα 4 διαφορετικά πεδία κειμένου και ο συγκεκριμένος αριθμός εικόνων εξυπηρετούν στην συγκεκριμένη απεικόνιση των άρθρων έτσι

ώστε να μην χαλάει η ομοιομορφία των συγκεκριμένων σελίδων. Αφού συμπληρωθούν όλα τα πεδία, ο χρήστης αρκεί να κάνει κλικ στο κουμπί “ΥΠΟΒΟΛΗ ΑΡΘΡΟΥ” και τότε το άρθρο θα δημοσιευτεί αυτόματα. Το νέο άρθρο θα εμφανιστεί επίσης και στην προσωπική του σελίδα την επόμενη φορά που θα την επισκεφθεί.

2.2.4 Σελίδα άρθρου

Στην σελίδα άρθρου εμφανίζονται όλες οι πληροφορίες που εισήγαγε κάποιος χρήστης όταν το δημοσίευσε. Επίσης εμφανίζονται η ημερομηνία δημοσίευσης του, το όνομα του χρήστη που το δημοσίευσε, το πλήθος των likes που έχει καθώς και την βαθμολογία που του έχουν δώσει οι χρήστες.

Σε αυτό το σημείο αξίζει να αναφερθεί η λειτουργικότητα των likes και της βαθμολογίας. Αρχικά, ενώ ο αριθμός των likes και η βαθμολογία είναι δημόσια ορατή σε όλους τους επισκέπτες της εφαρμογής, δεν έχουν όλοι δικαίωμα να κάνουν like ή να ψηφίσουν το άρθρο. Αυτό το δικαίωμα δίνεται μόνο στους εγγεγραμμένους χρήστες της εφαρμογής που έχουν συνδεθεί με τους προσωπικούς τους κωδικούς. Σε κάθε άλλη περίπτωση εμφανίζεται αναδυόμενο μήνυμα που προτρέπει σε σύνδεση (εικόνα 13).



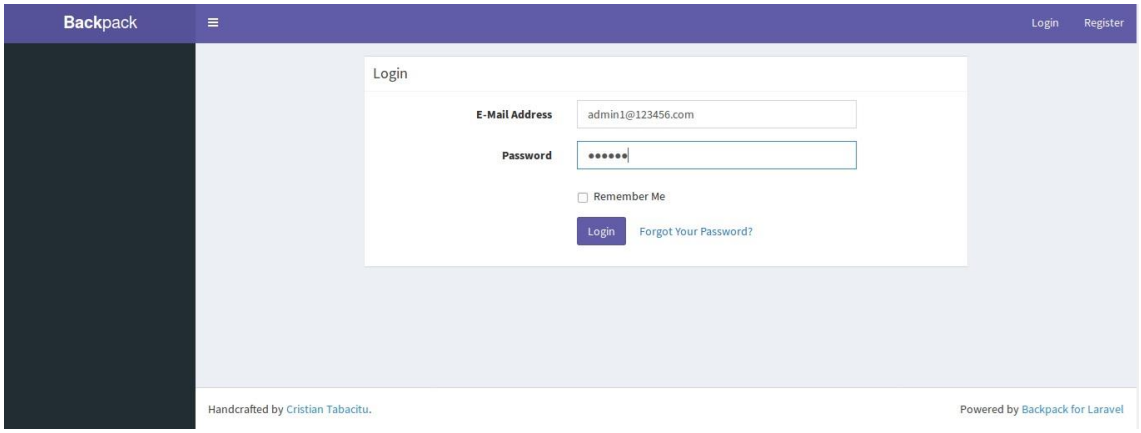
Εικόνα 13: Αναδυόμενο μήνυμα που προτρέπει σε σύνδεση

2.3 Παρουσίαση της εφαρμογής διαχείρισης περιεχομένου

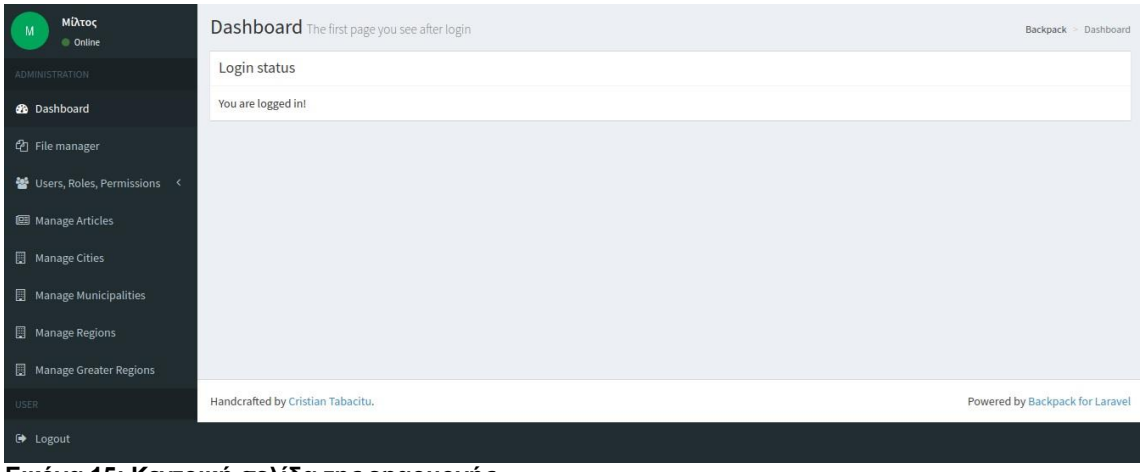
2.3.1 Εισαγωγή

Όπως έχει αναφερθεί και σε προηγούμενα σημεία, στα πλαίσια της παρούσας διπλωματικής εργασίας αναπτύχθηκε και μια δεύτερη εφαρμογή, η οποία οποία είναι βοηθητική της πρώτης. Εξυπηρετεί την βασικότερη ανάγκη ενός διαχειριστή, δηλαδή την εύκολη και άμεση διαχείριση του περιεχομένου της εφαρμογής. Πρακτικά, η παρούσα εφαρμογή είναι ένα σύστημα διαχείρισης περιεχομένου (Content Management System), και η οποία, έχοντας πρόσβαση στην ίδια βάση δεδομένων με την βασική εφαρμογή, δίνει την δυνατότητα στον διαχειριστή να δημιουργήσει, να επεξεργαστεί και να διαγράψει πληροφορίες μέσα από ένα εύχρηστο γραφικό περιβάλλον.

Η εφαρμογή είναι διαθέσιμη μόνο για χρήστες που κατέχουν κωδικούς πρόσβασης, διασφαλίζοντας με αυτό τον τρόπο, σε βασικό επίπεδο, την ακεραιότητα των αποθηκευμένων πληροφοριών. Στην επόμενη εικόνα (εικόνα 14) απεικονίζεται η σελίδα την οποία βλέπει ο χρήστης, ενώ στην εικόνα 15 απεικονίζεται η κεντρική σελίδα της εφαρμογής και η οποία εμφανίζεται μόνο όταν έχει γίνει η επαλήθευση των προσωπικών κωδικών του χρήστη.



Εικόνα 14: Εισαγωγή προσωπικών κωδικών χρήστη

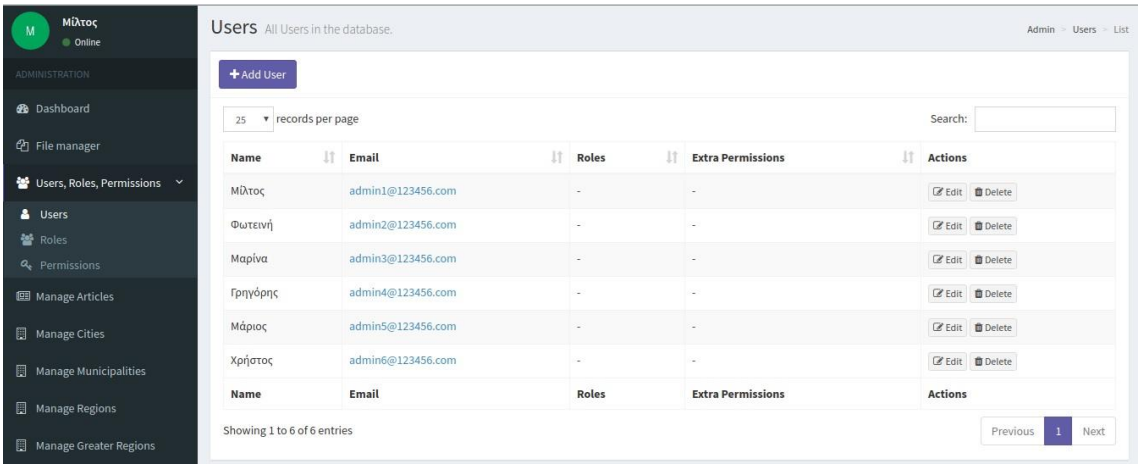


Εικόνα 15: Κεντρική σελίδα της εφαρμογής

2.3.2 Πλοήγηση μέσω του κεντρικού μενού

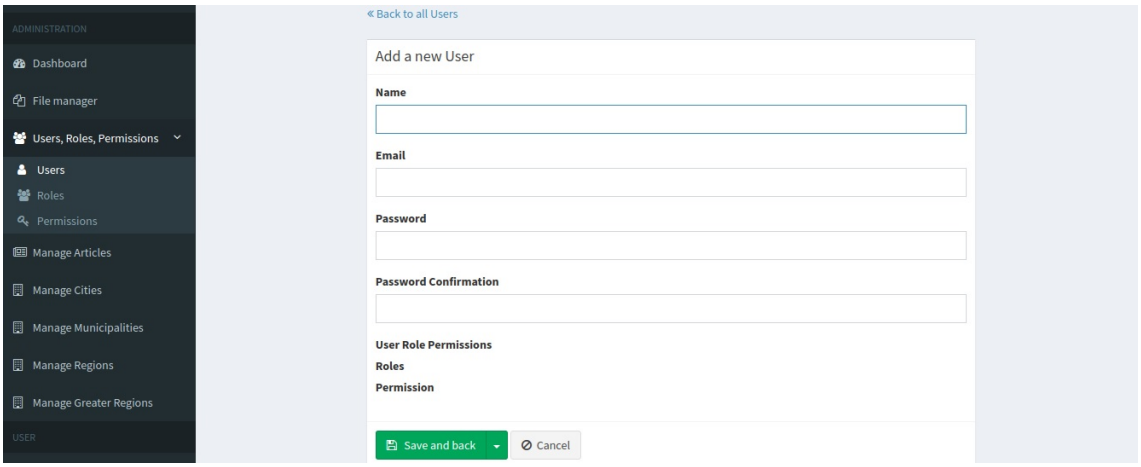
Στην εικόνα 15, στο αριστερό μέρος διακρίνεται το κεντρικό μενού της εφαρμογής, ενώ στο δεξιό τμήμα εμφανίζονται οι σχετικές πληροφορίες που επιθυμούμε. Το μενού χωρίζεται σε εννιά τμήματα, τα οποία θα επεξηγηθούν στην συνέχεια για την καλύτερη κατανόηση της εφαρμογής. Η πρώτη επιλογή με τίτλο “Dashboard” εμφανίζει πρακτικά την πρώτη σελίδα της εφαρμογής.

Στην παρούσα φάση, εκτός από το μήνυμα της επιτυχούς σύνδεσης, δεν εμφανίζει καμία άλλη χρηστική πληροφορία. Η δεύτερη επιλογή με τίτλο “File manager” δεν χρησιμοποιείται στην παρούσα εφαρμογή. Έχει παραμείνει ως υπενθύμιση για πιθανή μελλοντική λειτουργικότητα. Η τρίτη επιλογή με τίτλο “Users, Roles, Permissions” περιέχει τρεις υπό-επιλογές, όπως φαίνονται και στην εικόνα 16. Από αυτές χρησιμοποιείται μόνο η επιλογή “Users”, ενώ οι “Rules” και Permissions” δεν λαμβάνονται υπόψιν στην παρούσα φάση.



Εικόνα 16: Σελίδα εμφάνισης εγγεγραμμένων χρηστών

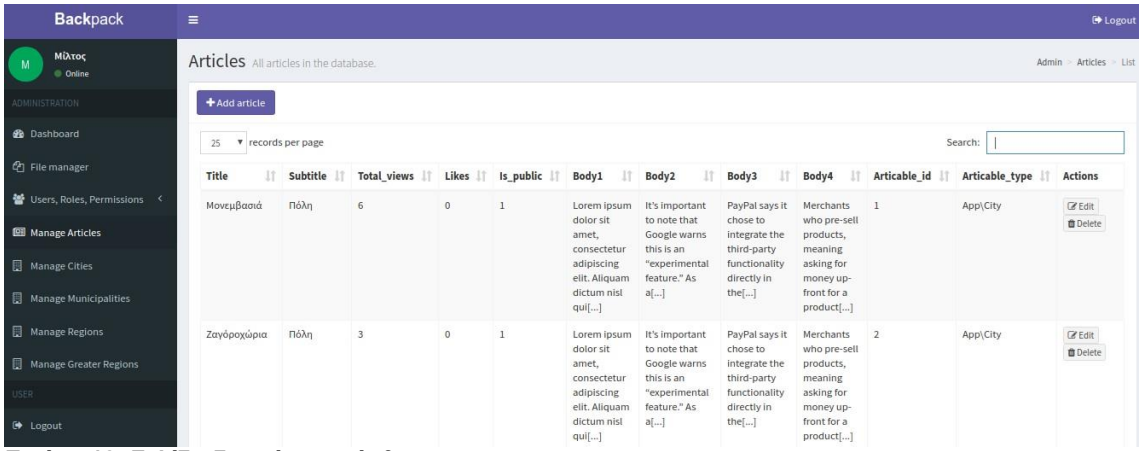
Στην εικόνα 16 παρουσιάζεται η σελίδα που εμφανίζεται όταν ο χρήστης επιλέξει το “Users”. Εδώ εμφανίζονται όλοι οι εγγεγραμμένοι χρήστες της εφαρμογής, με το όνομα και το email τους (οι στήλες Roles και Extra Permissions δεν λαμβάνονται υπόψιν), ενώ επίσης δίνεται η δυνατότητα στον διαχειριστή να τους επεξεργαστεί ή να τους διαγράψει. Στην ίδια σελίδα υπάρχει δυνατότητα αναζήτησης μέσω του πεδίου “Search” στο επάνω δεξιά τμήμα της σελίδας και η δυνατότητα προσθήκης νέου χρήστη μέσω του χρώματος μοβ κουμπιού, με τίτλο “Add User” στο επάνω αριστερά τμήμα της σελίδας. Στην αμέσως επόμενη εικόνα 17, παρουσιάζεται η σελίδα όπου ο διαχειριστής πρέπει να συμπληρώσει τα πεδία της φόρμας για να δημιουργήσει έναν νέο χρήστη. Και τα τέσσερα πεδία που παρουσιάζονται στην εικόνα είναι απαιτούμενα και πρέπει να συμπληρωθούν από τον διαχειριστή.



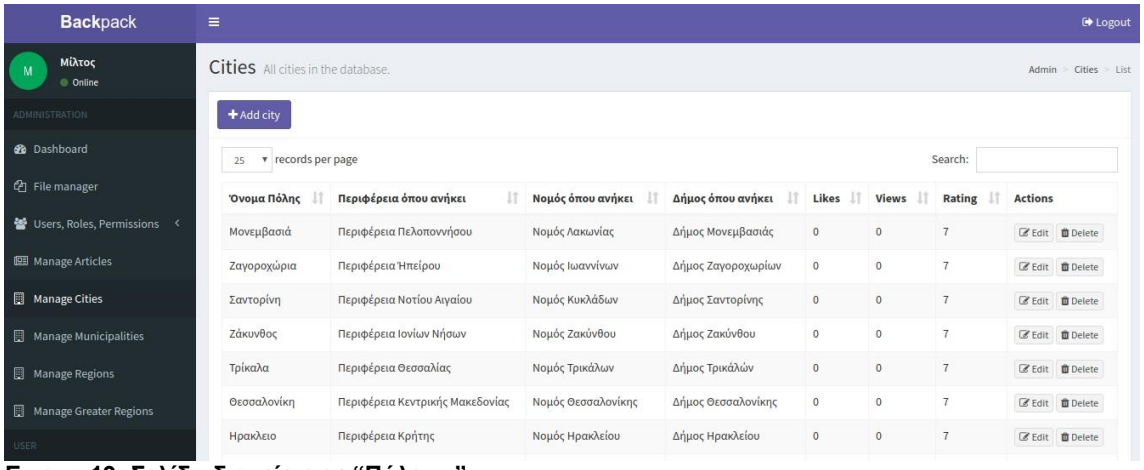
Εικόνα 17: Σελίδα δημιουργίας νέου χρήστη

Στην συνέχεια παρατίθενται εικόνες από 5 επόμενες επιλογές του μενού με τίτλους “Manage Articles” (εικόνα 18), “Manage Cities” (εικόνα 19), “Manage Municipalities”(εικόνα 20), “Manage Regions” (εικόνα 21) και “Manage Greater Regions” (εικόνα 22). Οι σελίδες που εμφανίζουν την λίστα των αποτελεσμάτων έχουν κοινή οπτική διαμόρφωση και διαθέσιμες επιλογές, όπως την δυνατότητα αναζήτησης, την δυνατότητα ταξινόμησης των τρεχόντων αποτελεσμάτων, την δυνατότητα επεξεργασίας ή διαγραφής της κάθε εγγραφής και τέλος την δυνατότητα επιλογής

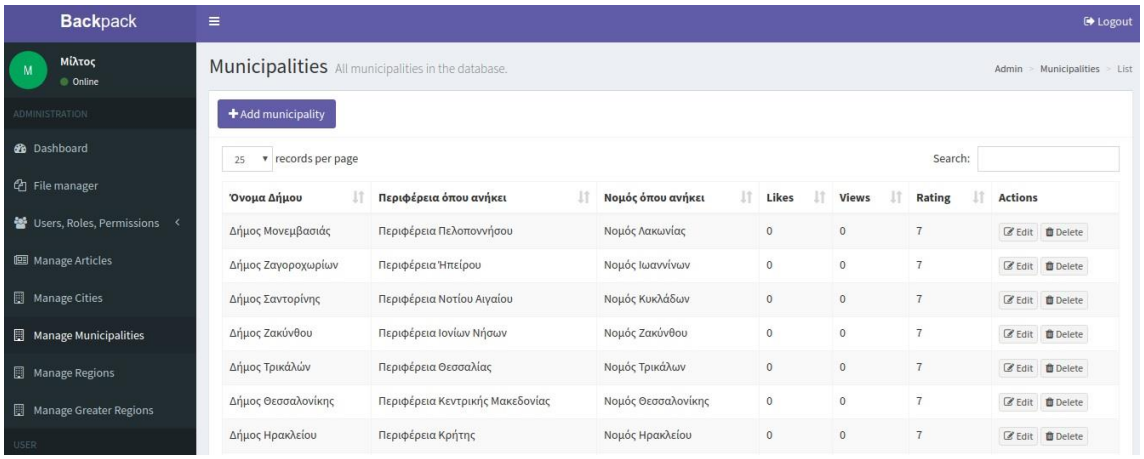
του πλήθους των αποτελεσμάτων που θα εμφανίζονται ανά σελίδα καθώς επίσης και την σελιδοποίηση των αποτελεσμάτων.



Εικόνα 18: Σελίδα διαχείρισης άρθρων

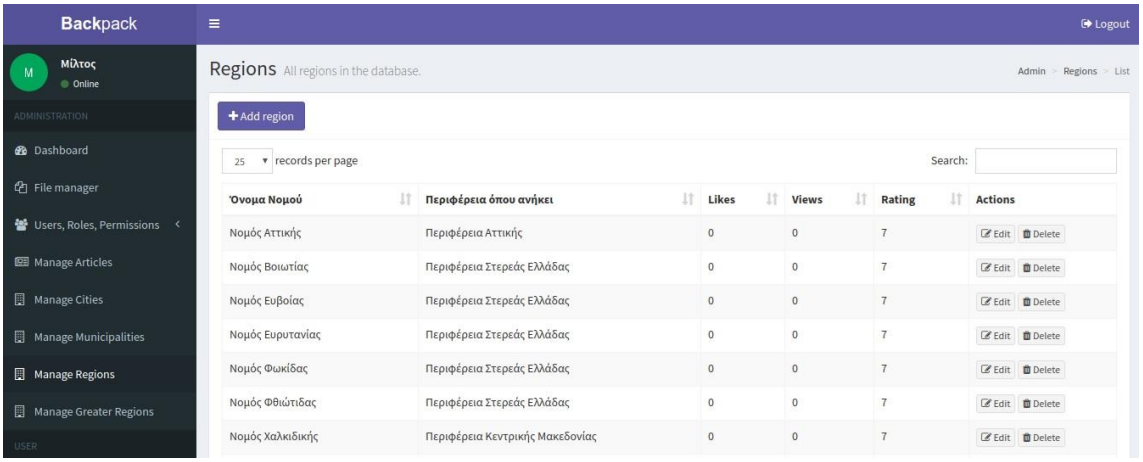


Εικονα 19: Σελίδα διαχείρισης "Πόλεων"

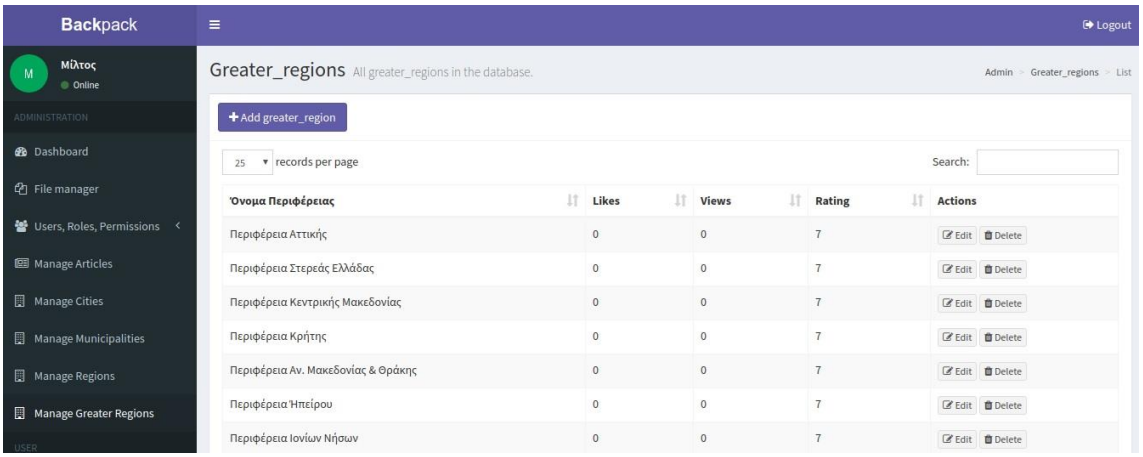


Εικόνα 20: Σελίδα διαχείρισης "Δήμων"

Προώθηση των περιοχών της Ελλάδας με την βοηθεια της κοινοτητας χρηστών μιας «έξυπνης» εφαρμογής, η οποία χρησιμοποιεί υπηρεσίες διαδικτύου

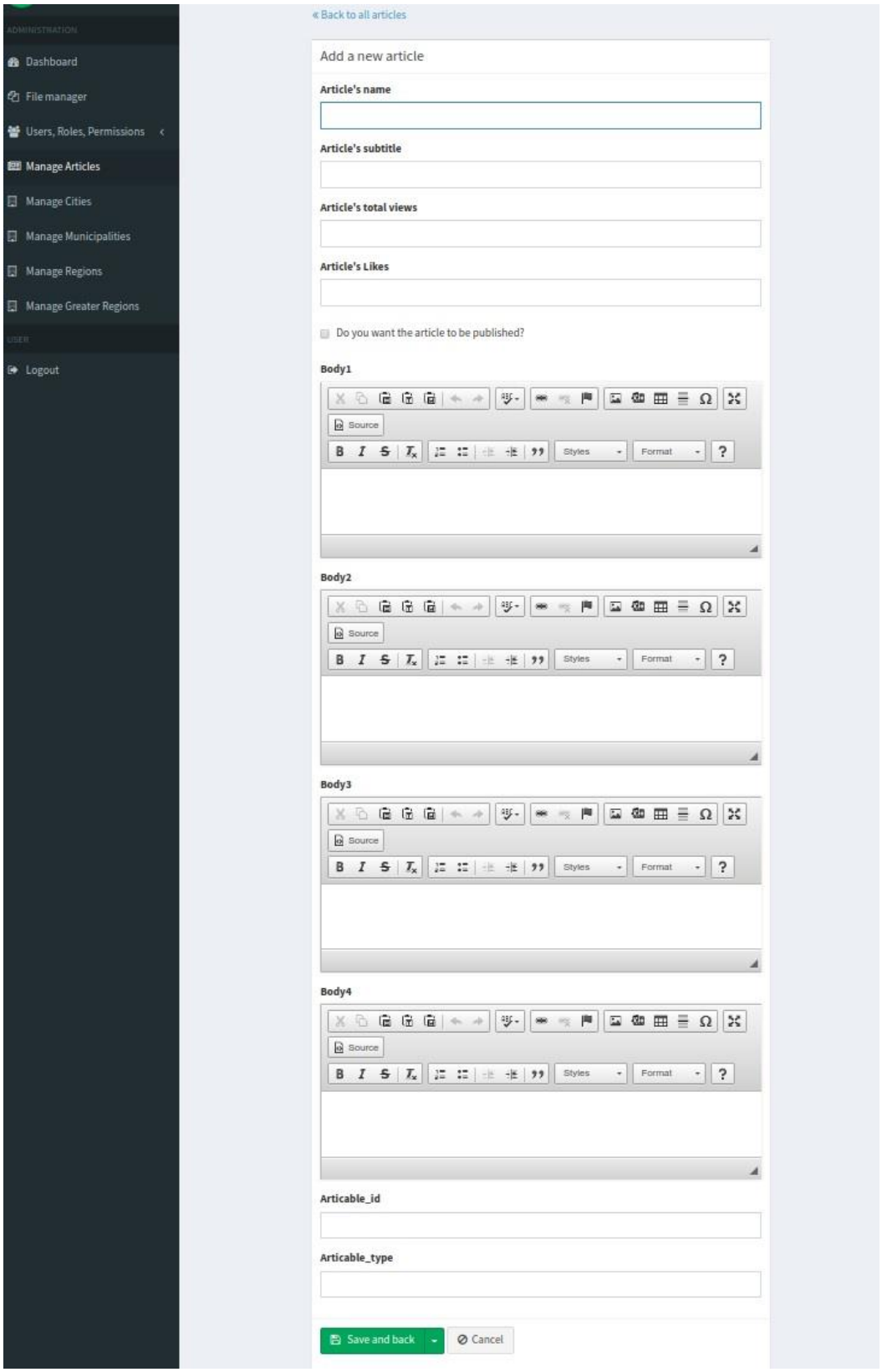


Εικόνα 21: Σελίδα διαχείρισης "Νομών"



Εικόνα 22: Σελίδα διαχείρισης "Περιφερειών"

Στην επόμενη εικόνα (εικόνα 23), παρουσιάζεται ενδεικτικά η σελίδα δημιουργίας ενός νέου άρθρου. Ξεκινώντας από το επάνω μέρος της σελίδας, διακρίνεται η επιλογή "Back to all articles" την οποία εάν πατήσει ο χρήστης, επιστρέφει στην σελίδα με την λίστα των άρθρων χωρίς να αποθηκεύσει τις πληροφορίες που τυχόν είχε εισάγει στην παρούσα σελίδα μέχρι εκείνη την στιγμή. Στην συνέχεια παρουσιάζονται όλα τα απαιτούμενα πεδία που πρέπει να συμπληρωθούν για να αποθηκευτεί ένα νέο άρθρο. Οι αντίστοιχες σελίδες των υπολοίπων επιλογών του μενού είναι παρόμοιες και γι' αυτόν τον λόγο δεν έχουν συμπεριληφθεί οι εικόνες τους. Αξίζει να αναφερθεί ότι τόσο στην παρούσα εφαρμογή,όσο και στην κύρια εφαρμογή της παρούσας διπλωματικής εργασίας, έγινε προσπάθεια ώστε η δομή των βασικών σελίδων να ακολουθεί μια βασική φόρμα, ώστε να βοηθήσει τον τελικό χρήστη να μάθει να χειρίζεται την εφαρμογής σε μικρό χρονικό διάστημα και να μην τον αναγκάζει να "θυμάται" κάθε φορά που βρίσκεται για παράδειγμα το κουμπί διαγραφής. Έτσι οι βασικές λειτουργίες (δημιουργία νέου αντικειμένου, επεξεργασία, διαγραφή, αναζήτηση) βρίσκονται σε προκαθορισμένα σημεία μέσα στις σελίδες. Κλείνοντας υπάρχει και η δυνατότητα αποσύνδεσης από την εφαρμογή (logout), όπου ο χρήστης μπορεί να κάνει κλικ στην αντίστοιχη επιλογή είτε από το μενού είτε από το επάνω δεξιά μέρος της σελίδας.



Εικόνα 23: Σελίδα δημιουργίας νέου άρθρου.

Προώθηση των περιοχών της Ελλάδας με την βοηθεια της κοινοτητας χρηστών μιας «έξυπνης» εφαρμογής, η οποία χρησιμοποιεί υπηρεσίες διαδικτύου

3. Αρχιτεκτονική συστήματος και βοηθητικά εργαλεία

3.1 Αρχική προσέγγιση

Η μελέτη και η ανάπτυξη της εφαρμογής έγιναν εξ 'ολοκλήρου από την αρχή, γεγονός που δημιούργησε σημαντικά και κρίσιμα ερωτήματα που έπρεπε να απαντηθούν, καθώς θα επηρέαζαν όλη την πορεία υλοποίησης του έργου. Για το λόγο ότι η αρχιτεκτονική της εφαρμογής ήταν ένα μείζον θέμα, αποφασίστηκε να διαιρεθεί σε μικρότερα αυτόνομα τμήματα, κάθε τμήμα να αναλυθεί ξεχωριστά από το άλλο, και στο τέλος θα γινόταν η διαδικασία της ένωσής τους. Ο βασικός διαχωρισμός της υλοποίησης των εφαρμογών διαδικτύου γίνεται σε τρεις κατηγορίες:

- Front end – Υλοποίηση του τμήματος της εφαρμογής που εκτελείται στον browser του τελικού χρήστη.
- Back end – Υλοποίηση του τμήματος της εφαρμογής που εκτελείται στον server και διαχειρίζεται όλα τα HTTP requests που στέλνονται από τους χρήστες
- Δομή της Βάσης Δεδομένων

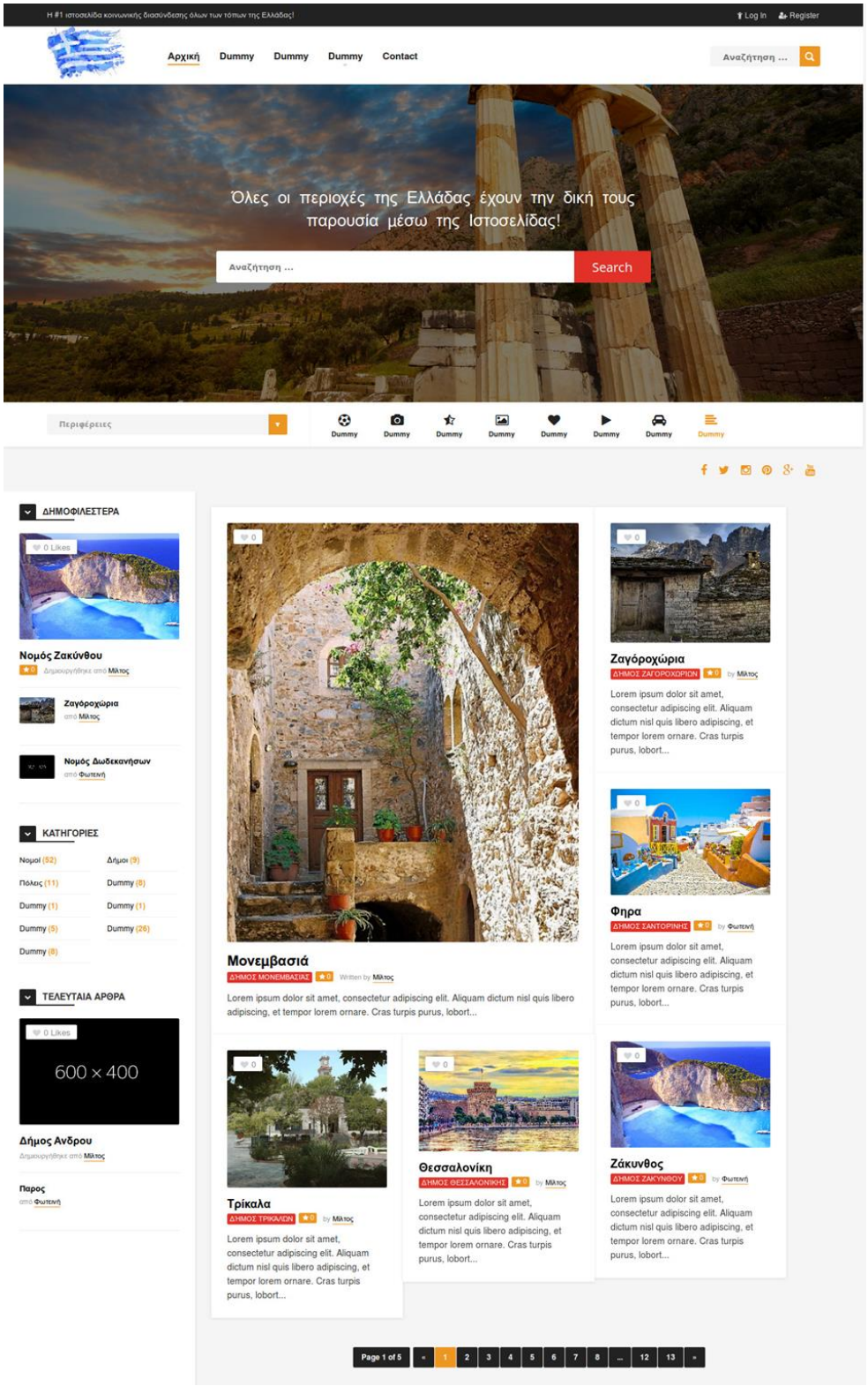
Και οι τρεις προαναφερθείσες κατηγορίες είναι εξίσου σημαντικές καθώς έχουν επίδραση στην εμπειρία χρήσης της εφαρμογής, στην απόκριση και την τεχνική συντήρηση της εφαρμογής καθώς επίσης και στην ευελιξία της σε μελλοντικές αναβαθμίσεις. Η διαδικασία ανάπτυξης της εφαρμογής ξεκίνησε με τον ορισμό των αναγκών τις οποίες θα εξυπηρετεί, την ανάλυση του τρόπου με τον οποίο θα λειτουργεί, καθώς επίσης και με ποιο τρόπο θα γίνει όσο το δυνατόν περισσότερο χρηστική. Στην συνέχεια περιγράφονται με περισσότερες λεπτομέρειες κάθε μια από τις προαναφερόμενες τρεις κατηγορίες, ενώ στην τελευταία ενότητα περιγράφεται το εργαλείο που χρησιμοποιήθηκε για τον έλεγχο πηγαίου κώδικα.

3.2 Front end υλοποίηση

Η συγκεκριμένη κατηγορία ανάπτυξης εφαρμογών διαδικτύου προσανατολίζεται στην δημιουργία του γραφικού περιβάλλοντος, το οποίο είναι το πρώτο πράγμα με το οποίο έρχεται σε επαφή ο χρήστης και διαμορφώνει την γενικότερη εμπειρία χρήσης. Βασικό στοιχείο στην σχεδίαση είναι η σχετική ομοιομορφία των τμημάτων της κάθε σελίδας, έτσι ώστε ο χρήστης να μπορεί να "μάθει" σε μικρό χρονικό διάστημα τον τρόπο λειτουργίας της εφαρμογής και να μπορεί να εντοπίζει σε σύντομο χρονικό διάστημα τις βασικότερες λειτουργίες.

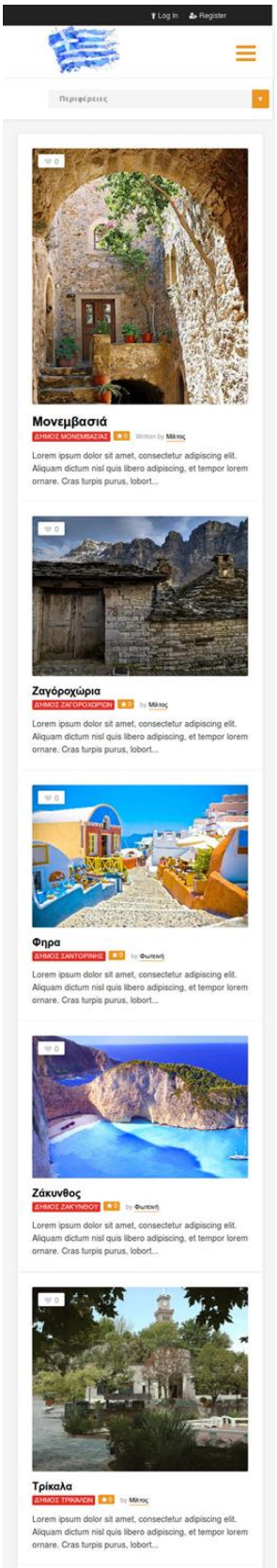
Σε τεχνικό επίπεδο, χρησιμοποιήθηκαν οι τρεις δημοφιλέστερες τεχνολογίες για web development, δηλαδή οι html 5, css3 και javascript. Η html5 είναι η τελευταία έκδοση της δημοφιλούς γλώσσας του διαδικτύου. Δημοσιεύτηκε τον Οκτώβριο του 2014 από τον οργανισμό World Wide Web Consortium, ο οποίος είναι υπεύθυνος για την τυποποίηση των τεχνολογιών στο παγκόσμιο διαδίκτυο. Υποστηρίζεται από όλους τους σύγχρονους web browsers και τις mobile συσκευές. Σε ότι αφορά την εμπειρία χρήσης σε mobile συσκευές, η html συνεργάζεται με την γλώσσα προγραμματισμού css3, η οποία διαχειρίζεται την οπτική δομή των πληροφοριών μιας ιστοσελίδας. Ο επιτυχημένος συνδυασμός των προηγούμενων δυο τεχνολογιών, η διάδοση των κινητών συσκευών κατά την τελευταία δεκαετία και η αυξημένη ανάγκη για την προβολή ιστοσελίδων σε οθόνες μικρού μεγέθους (λιγότερο από 10 ίντσες) οδήγησαν σε μια νέα τάση, το λεγόμενο "responsive design". Οι ανάγκες που καλύπτει αυτός ο τρόπος σχεδιασμού ιστοσελίδων είναι καθαρά πρακτικός, αφού οι χρήστες των web εφαρμογών χρησιμοποιεί που χρησιμοποιούν κινητές συσκευές, επιθυμούν να έχουν παρόμοια εμπειρία χρήσης με εκείνη που θα είχαν σε οθόνες μεγαλύτερου μεγέθους. Το responsive design έχει χρησιμοποιηθεί και στην παρούσα διπλωματική εργασία (και στις δυο εφαρμογές), έτσι ώστε να μεταβάλλεται δυναμικά ο τρόπος προβολής του περιεχόμενο της ιστοσελίδας, ανάλογα με της διαστάσεις της οθόνης. Στην συνέχεια παρατίθενται δυο εικόνες στις οποίες εμφανίζεται η ίδια σελίδα σε οθόνη 15.4 inches και σε οθόνη 5. και σε οθόνη 5.8 inches. Στην δεύτερη περίπτωση παρατηρείται ότι το μενού έχει εξαφανιστεί, δίνοντας την θέση του σε ένα link ώστε ο χρήστης να

το εμφανίζει όταν επιθυμεί, ενώ οι φωτογραφίες των άρθρων εμφανίζονται η μια κάτω από την άλλη. Έτσι γίνεται αξιοποίηση όλους του πλάτους της οθόνης της εκάστοτε συσκευής.



Εικόνα 24: Η αρχική σελίδα της εφαρμογής σε ανάλυση 15.4 inches

Προώθηση των περιοχών της Ελλάδας με την βοηθεια της κοινοτητας χρηστών μιας «έξυπνης» εφαρμογής, η οποία χρησιμοποιεί υπηρεσίες διαδικτύου



Εικόνα 25: Η αρχική σελίδα της εφαρμογής σε οθόνη 5.8 inches

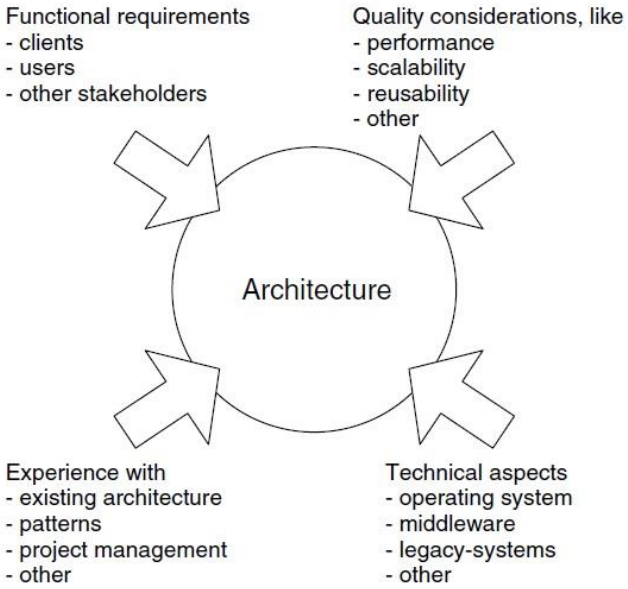
Προώθηση των περιοχών της Ελλάδας με την βοηθεια της κοινοτητας χρηστών μιας «έξυπνης» εφαρμογής, η οποία χρησιμοποιεί υπηρεσίες διαδικτύου

Σε ότι αφορά την javascript, χρησιμοποιήθηκε το πλέον διαδεδομένο library, το jquery. Σε πολλά σημεία της υλοποίησης φάνηκαν τα πλεονεκτήματα του συγκεκριμένου όπως η διαχείριση των event και το ajax. Αξίζει να αναφερθεί σε αυτό το σημείο η εφαρμογή στέλνει αιτήματα στον server μέσω ajax σε αρκετά σημεία, όπως για παράδειγμα το pop up όπου ο χρήστης μπορεί να κάνει login και register και τα σημεία που ο χρήστης βαθμολογεί τα άρθρα (θα αναλυθεί στην συνέχεια). Με την χρήση του jquery επιτεύχθηκε μείωση των γραμμών του κώδικα, ευκολότερη αναγνωσιμότητα και άρα καλύτερη συντήρηση της εφαρμογής. Επίσης βελτιώθηκε και η εμπειρία χρήσης της εφαρμογής, καθώς ελαττώθηκαν οι ανανεώσεις των σελίδων και εμφανίζονται πλέον και μηνύματα που καθοδηγούν τον χρήστη.

3.3 Back end υλοποίηση

3.3.1 Εισαγωγή

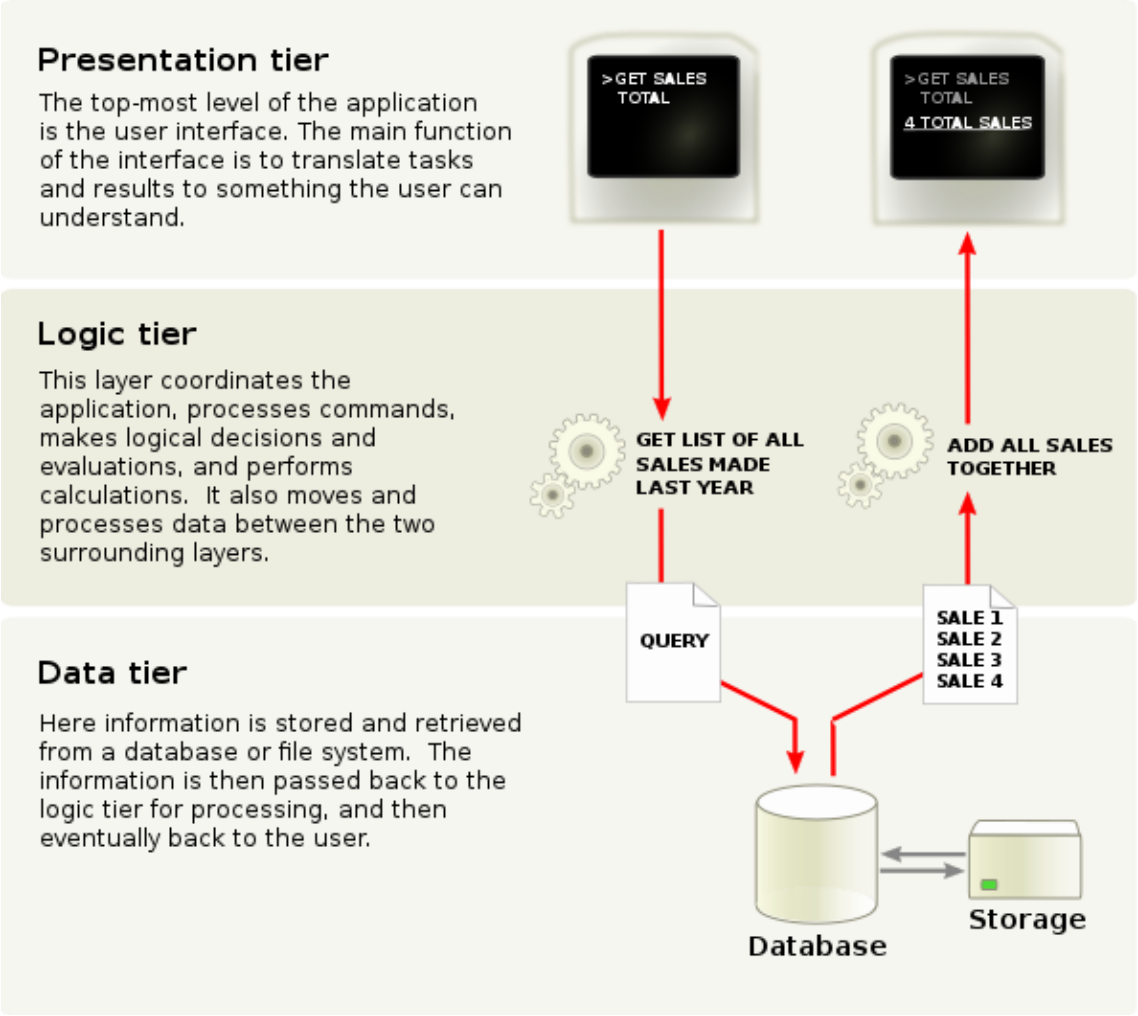
Πέρα από το γραφικό περιβάλλον της εφαρμογής, υπάρχει και το back end τμήμα το οποίο επηρεάζει εξίσου την απόδοση της εφαρμογής και την αλληλεπίδραση με τον χρήστη. Καθώς το συγκεκριμένο τμήμα είναι η ραχοκοκαλιά των εφαρμογών διαδικτύου, χρειάστηκε να γίνουν σημαντικές επιλογές σε ότι αφορά την αρχιτεκτονική της εφαρμογής. Σε αντίθεση με το front end τμήμα, εδώ οι επιλογές σε διαθέσιμες τεχνολογίες είναι περισσότερες και εξυπηρετούν τις αυξανόμενες ανάγκες που προκύπτουν στην ανάπτυξη λογισμικού για το web. Κάθε εφαρμογή έχει ιδιαίτερες απαιτήσεις και καλείται να επιλύσει διαφορετικές ανάγκες. Σημαντικά ερωτήματα που πρέπει να απαντηθούν πριν την επιλογή της αρχιτεκτονικής και των τεχνολογιών που θα χρησιμοποιηθούν είναι το πλήθος των χρηστών που αναμένεται να χρησιμοποιήσουν την εφαρμογή, το πόσο υψηλό θα είναι το επίπεδο ασφάλειας της εφαρμογής και τα δεδομένα τα οποία θα διαχειρίζεται η εφαρμογή, δηλαδή αν θα είναι είναι δεδομένα πραγματικού χρόνου καθώς και το μέγεθός τους (τα οποία συνεπάγονται και αντίστοιχη πρόβλεψη σε επίπεδο δικτύου). Επιπλέον, οι εφαρμογές web θα πρέπει να βασίζονται σε μια αρχιτεκτονική λογισμικού που θα επιτρέπει την γρήγορη συντήρηση και επεκτασιμότητα, καθώς οι ανάγκες στο διαδίκτυο μεταβάλλονται με γρήγορους ρυθμούς.



Εικόνα 26: Παράγοντες που επηρεάζουν την αρχιτεκτονική ενός λογισμικού (Kappel, G., Proll, B.

3.3.2 Αρχιτεκτονικές λογισμικού διαδικτύου

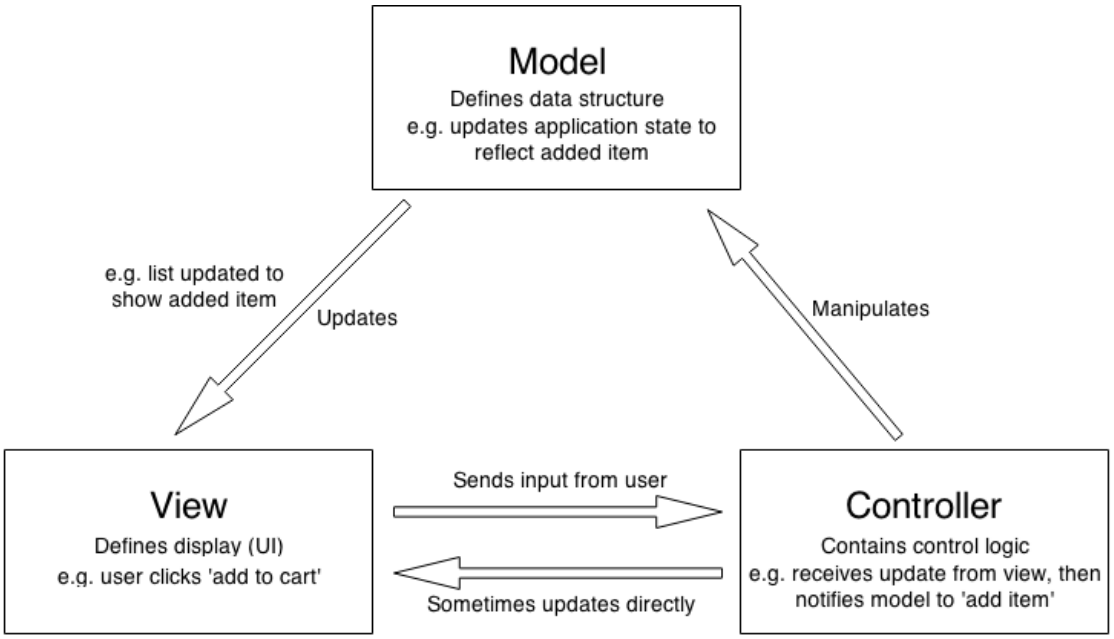
Στην ανάπτυξη λογισμικού για το διαδίκτυο χρησιμοποιούνται αρκετές διαφορετικές αρχιτεκτονικές. Αξίζει να γίνει αναφορά σε μια από τις πλέον διαδεδομένες αρχιτεκτονικές, την επονομαζόμενη “Πολλαπλών επιπέδων” (multitier architecture). Η δημοφιλέστερη παραλλαγή της είναι η αρχιτεκτονική “τριών επιπέδων” (three-tier architecture), σύμφωνα με οποία η παρουσίαση των δεδομένων στον τελικό χρήση, η “λογική” της εφαρμογής και η διαχείριση και αποθήκευση των δεδομένων γίνονται από διαφορετικά και αυτόνομα μεταξύ τους τμήματα λογισμικού. Αυτό βοηθάει στην καλύτερη συντήρηση της εφαρμογής σε βάθος χρόνου, καθώς πιθανά bugs εμφανίζονται σε κάποιο από τα τρία προηγούμενα τμήματα και επιλύονται εκεί. Επίσης η επεκτασιμότητα της εφαρμογής είναι ευκολότερη αφού οι μελλοντικά νέες λειτουργικότητες θα υλοποιηθούν μόνο σε ένα από τα προηγούμενα τμήματα, ανάλογα την ανάγκη που χρειάζεται να επιλυθεί, κάνοντας έτσι ποιο ξεκάθαρο το “τι” θα επηρεαστεί από τον καινούριο κώδικα που θα εισαχθεί.



Εικόνα 27: Ενδεικτική χρήση της αρχιτεκτονικής “τριών επιπέδων”

Η αρχιτεκτονική λογισμικού που χρησιμοποιήθηκε στην παρούσα διπλωματική εργασία ονομάζεται “MVC” από τα αρχικά των λέξεων model, view και controller τα οποία είναι και τα τρία δομικά τμήματα της αρχιτεκτονικής. Κάθε ένα από αυτά τα τρία τμήματα διαχειρίζονται σε διαφορετικό επίπεδο τα δεδομένα και τις αλληλεπιδράσεις του τελικού χρήστη με την εφαρμογή. Όπως και στο “multitier architecture” σκοπός είναι η δυνατότητα εύκολης και γρήγορης συντήρησης της εφαρμογής, και η δυνατότητα επαναχρησιμοποίησης του κώδικα. Το βασικό στοιχείο είναι το “model”. Εδώ καθορίζονται όλες οι λειτουργίες και η γενικότερη συμπεριφορά

της εφαρμογής. Είναι ανεξάρτητο από το γραφικό περιβάλλον της εφαρμογής και διαχειρίζεται τα δεδομένα και την λογική του λογισμικού. Το “view” είναι το τμήμα εκείνο που είναι υπεύθυνο για το οπτικό κομμάτι της εφαρμογής, δηλαδή εκεί που εμφανίζονται όλες οι πληροφορίες που έχει ζητήσει ο χρήστης. Τέλος, ο “controller” είναι το τμήμα που διαχειρίζεται οτιδήποτε εισάγει ο χρήστης στην εφαρμογή, και μεταφέρει αυτά τα αιτήματα (requests στην διεθνή ορολογία του web) είτε στα “views” είτε στα “models”.



Εικόνα 2.8: Αρχιτεκτονική MVC

Ιστορικά, η πρώτη αναφορά στην αρχιτεκτονική “MVC” έγινε την δεκαετία του 1970 στην γλώσσα Smalltalk-76 από τον Trygve Reenskaug στο ερευνητικό κέντρο “Xerox Palo Alto Research Center”. Τα τελευταία χρόνια έχει ευρεία διάδοση και στην ανάπτυξη web εφαρμογών μέσω των web frameworks που αναπτύχθηκαν για πολλές γλώσσες προγραμματισμού. Ως web framework νοείται ένα προκαθορισμένο προγραμματιστικό περιβάλλον, που προσφέρει συγκεκριμένες λειτουργικότητες και έχει ως σκοπό την μείωση του χρόνου και του κόστους υλοποίησης μιας εφαρμογής, ενώ παράλληλα χρησιμοποιούν όλα τα τελευταία χαρακτηριστικά των γλωσσών προγραμματισμού πάνω στις οποίες βασίζονται. Τα web frameworks είναι πλέον τόσο δημοφιλή που η υλοποίηση μιας τέτοιας εφαρμογής από το μηδέν, θεωρείται πλέον κάτι το αδιανόητο. Σχεδόν όλες οι σύγχρονες γλώσσες προγραμματισμού έχουν πλέον διαθέσιμα frameworks τα οποία προσφέρουν πλεονεκτήματα όπως την μείωση του συνολικού χρόνου υλοποίησης, την δυνατότητα επαναχρησιμοποίησης τμημάτων του κώδικα και την δυνατότητα να υπάρχει μεγάλη ευελιξία σε μελλοντικές αναβαθμίσεις της εφαρμογής. Πιο συγκεκριμένα, η PHP έχει διαθέσιμα αρκετά frameworks όπως το Symphony2, το CodeIgniter, το Laravel, το Yii2 και πολλά άλλα. Κάθε ένα από αυτά έχει βασιστεί σε διαφορετική αρχιτεκτονική λογισμικού, και κατά συνέπεια έχουν διαφορετικά πλεονεκτήματα και μειονεκτήματα. Καθέ ένα από αυτά πρέπει να χρησιμοποιούνται συνειδητά για την δημιουργία την τελικής εφαρμογής, ανάλογα με τις ανάγκες και τις ιδιαίτερες λειτουργικότητες που υπάρχουν. Για παράδειγμα τοYii2 αποτελεί μονόδρομο για εφαρμογές που η απόδοση είναι ένας κρίσιμος παράγοντας, καθώς είναι το γρηγορότερο framework για PHP, ενώ το codeigniter είναι χρήσιμο σε περιπτώσεις που η συμβατότητα με παλαιότερες εκδόσεις PHP είναι αναγκαία.

3.3.3 Laravel

Στην παρούσα εφαρμογή, χρησιμοποιήθηκε το web framework Laravel (έκδοση 5.4), ένα από τα νεότερα frameworks για την PHP, το οποίο έχει βασιστεί πάνω στην αρχιτεκτονική του MVC

βασιστεί πάνω στο Codeigniter. Ωστόσο, έχει αναπτύξει πολυάριθμη κοινότητα ποργραμματιστών η οποία το συντηρεί συνεχώς με συνέπεια να είναι το δημοφιλέστερο web framework πλέον για την γλώσσα PHP. Από το 2017 έχουν ξεκινήσει και το συνεισφέρουν στην ανάπτυξή του και εταιρείες λογισμικού όπως οι Vehikl, Tighten Co. Και Kirschbaum

Εικόνα 29: Το λογότυπο του Laravel

Στα πλεονεκτήματα του περιλαμβάνεται το γεγονός ότι χρησιμοποιεί τον package manager της PHP με το όνομα “Composer”. Αυτό πρακτικά σημαίνει ότι αναλόγως με τις τρέχουσες ανάγκες της εφαρμογής, υπάρχει δυνατότητα να εγκατασταθεί ένα από τα υπάρχοντα thrid party packages και έτσι να υπάρχει διαθέσιμη άμεσα μια νέα λειτουργικότητα. Αυτός ο αποκεντρωμένος τρόπος διαχείρισης packages, βοηθάει στην ευκολότερη συντήρηση του κάθε ενός από αυτά, καθώς και εν τέλει και του ίδιου του Laravel, αφού πλέον κάθε ομάδα από την κοινότητα προγραμματιστών ασχολείται με κάτι εξειδικευμένο. Για την υλοποίηση της εφαρμογής της παρούσας εργασίας χρησιμοποιηθηκαν τα packages “Laravel-Tagging”, “Form Collective”, “Laravel-Scout”, “Laravel-activitylog”. Τα “Laravel-activitylog” χρησιμοποιήθηκε ώστε να καταγράφει στην βάση δεδομένων τις σελίδες που επισκέπτεται ο κάθε χρήστης. Τα αποτελέσματα αυτά χρησιμοποιούνται σε διάφορα σημεία της εφαρμογής ώστε να εμφανίζονται προτάσεις με παρόμοια αποτελέσματα. Το “Laravel-Tagging” χρησιμοποιήθηκε ώστε να μπορούν να ομαδοποιηθούν κάτω από συγκεκριμένες καρτέλες (tags) τα άρθρα της εφαρμογής, έτσι ώστε να μπορούν να ανακτηθούν ευκολότερα από τους χρήστες. Το “Laravel-Scout” χρησιμοποιήθηκε για να υλοποιηθεί η λειτουργικότητα της αναζήτησης στην εφαρμογή. Ο χρήστης μπορεί να κάνει αναζήτηση μέσα από τα αντίστοιχα πεδία που υπάρχουν στο επάνω μέρος σε κάθε σελίδα, και να λαμβάνει τα αποτελέσματα σε ελάχιστο χρόνο, ακόμα και αν το σύνολο των εγγραφών μέσα στις οποίες γίνεται η αναζήτηση είναι μεγαλύτερο από μερικές δεκάδες χιλιάδες. Αυτό επιτυγχάνεται με την χρήση ενός third party service, του Algolia το οποίο κρατάει καταχωρημένες τις εγγραφές τις βάσης της εφαρμογής, και μετά από κάθε αίτημα της εφαρμογής, απαντάει με τα αποτελέσματα της αναζήτησης μέσω του API που διαθέτει, σε μορφή JSON.

Ένα άλλο χαρακτηριστικό του Laravel είναι το “Blade” ένα web template το οποίο συνδυάζει το “View” και το “Model” της αρχιτεκτονικής MVC ώστε να δημιουργεί σελίδες με δυναμικό περιεχόμενο. Έτσι σε ένα αρχείο που περιέχει κώδικα HTML μπορεί να γραφτεί κώδικας PHP που καλεί κλάσεις και συναρτήσεις από κάποιο model. Επίσης δίνεται η δυνατότητα να επαναχρησιμοποιηθούν πολλές φορές αρχεία με κώδικα html, σε διαφορετικές σημεία της εφαρμογής.

Σημαντικός είναι επίσης και ο τρόπος που λειτουργούν οι controllers. Πρακτικά, κάθε controller είναι μια μέθοδος η οποία αντιστοιχίζεται με ένα URI και αναγνωρίζει ένα συγκεκριμένο HTTP request. Όταν καλεστεί ένας controller, τότε και αυτός με την σειρά του ενεργοποιεί το αντίστοιχο model ή και view. Η πιο συνηθισμένη χρήση των controllers είναι να επιτελούν λειτουργίες που υπάγονται στο λεγόμενο CRUD (Create, Read, Update, Delete), δηλαδή ελέγχει τον κύκλο ζωής ενός “αντικειμένου”, του βασικότερου στοιχείου στον αντικειμενοστραφή προγραμματισμό. Σε αυτό το σημείο αξίζει να αναφερθεί ότι το laravel

χρησιμοποιεί την τεχνική object relation mapping (Eloquent ORM) ώστε τα αντικείμενα που δημιουργούνται από την εφαρμογή να συνδέονται άμεσα με την βάση δεδομένων χωρίς να χρειάζονται (σε προγραμματιστικό επίπεδο) να γράφονται συνεχώς ερωτήματα προς την βάση. Με αυτό τον τρόπο διευκολύνονται οι σχέσεις που υπάρχουν μεταξύ των διαφορετικών κλάσεων της εφαρμογής.

Τέλος μια ακόμα προγραμματιστική τεχνική που χρησιμοποιείται στο laravel και του δίνει σημαντικό πλεονέκτημα είναι τα “migrations”, η καταγραφή των αλλαγών της βάσης δεδομένων (π.χ. μετατροπές στηλών ή και πινάκων) σε μεμονωμένα αρχεία, ταξινομημένα με χρονολογική σειρά. Με αυτό τον τρόπο, μια εφαρμογή μπορεί άμεσα να μεταφερθεί σε άλλο server ενώ σε περίπτωση που κάποια αλλαγή δεν είναι πλέον επιθυμητή μπορεί εύκολα να αναστραφεί.

3.3.4 Ανάλυση εφαρμογής

Το γεγονός ότι η αρχιτεκτονική που χρησιμοποιήθηκε είναι το MVC βοηθάει στην υποδιαίρεση της εφαρμογής σε μικρότερα αυτόνομα τμήματα με την χρήση των models. Κατά την ανάλυση της εφαρμογής διαχωρίστηκαν τα βασικά τμήματα τις εφαρμογής, για κάθε ένα από τα οποία δημιουργήθηκε και ένα model. Τα models είναι κλάσεις που κληρονομούν συνήθως από την κλάση “Model”, η οποία είναι μια από τις θεμελιώδεις κλάσεις του Laravel. Ενδεικτικά, μερικά από τα models είναι το GreaterRegion για τις περιφέρειες της Ελλάδας, το Region για τους νομούς, το Municipality για τους δήμους και το City για τις πόλεις και τα χωριά. Το βασικότερο όμως model είναι το Article το οποίο είναι και η βασικότερη οντότητα της εφαρμογής.

Στον κώδικα του model Article διακρίνονται τα στοιχεία που το αποτελούν, δηλαδή 'title', 'subtitle', 'body1', 'body2', 'body3', 'body4', 'body5', 'image1_url', 'image2_url', 'image3_url', 'image4_url', 'image5_url','articable_id', 'articable_type'. Τα πεδία των εικόνων υπάρχουν μόνο για μελλοντική χρήση. Δεν χρησιμοποιούνται αυτή την στιγμή καθώς οι εικόνες των άρθρων ελέγχονται από το model ArticleImage με το οποίο υπάρχει relation. Σε αυτό το σημείο αξίζει να γίνει αναφορά στον τρόπο που δημιουργεί το Laravel τα relations μεταξύ των models. Στον κώδικα του model Article, οι πρώτες συναρτήσεις που έχουν γραφτεί αφορούν τις συσχετίσεις με τα άλλα models καθώς τον τρόπο της συσχέτισης. Στην συνέχεια έχουν επισυναφθεί αυτές οι συναρτήσεις οι οποίες θα επεξηγηθούν αμέσως μετά.

```
/* -- Relationship with articles (Polymorphic) -- */
    public function articable() {
        return $this->morphTo();
    }

    /* -- Inverse Relationship with User-- */
    public function user() {
        return $this->belongsTo('App\User');
    }

    /* -- Relationship with Ratings -- */
        public function ratings() {
        return $this->hasMany('App\Rating');
    }
```

```
    /* -- Relationship with Likes -- */
    public function likes() {
        return $this->hasMany('App\Like');
    }

    /* -- Relationship with ArticleImages -- */
    public function images() {
        return $this->hasMany('App\ArticleImage');
```

Το άρθρο έχει συσχέτιση με όλα τα models των περιοχών που αναφέρθηκαν πιο πάνω. Έτσι κάθε περιοχή (π.χ. κάθε Νομός) μπορεί να έχει πολλά άρθρα, αλλά κάθε άρθρο μπορεί να ανήκει μόνο σε μία περιοχή. Για να δημιουργηθεί αυτή η συσχέτιση δημιουργήθηκε η συνάρτηση articable() μέσα στη οποία χρησιμοποιήθηκε η μέθοδος morphTo(). Αυτή είναι μια μέθοδος που προσφέρει το laravel ώστε να δημιουργήσει ένα polymorphic relationship. Αυτός ο τύπος συσχέτισης, σε επίπεδο βάσης δεδομένων, καταργεί την ανάγκη ενός ενδιάμεσου πίνακα στον οποίο θα αποθηκεύονται τα FOREIGN KEYS των άλλων πινάκων (εξ' ορισμού, στο Laravel κάθε model έχει έναν πίνακα στην βάση δεδομένων). Η πληροφορία της συσχέτισης των δύο models αποθηκεύεται μέσα στον πίνακα του model που χρησιμοποιεί την μέθοδο morphTo(), του Article δηλαδή στην προκειμένη περίπτωση. Στον πίνακα του άρθρου υπάρχουν δυο επιπλέον στήλες, η articable και η articable_id. Στην πρώτη στήλη αποθηκεύεται το όνομα του model στο οποίο ανήκει το άρθρο(π.χ. City) και στην δεύτερη αποθηκεύεται το id της συγκεκριμένης περιοχής (π.χ. 2). Έτσι, το Laravel, διαβάζοντας τις πληροφορίες του άρθρου καταλαβαίνει ότι το τρέχον άρθρο "ανήκει" στην περιοχή με model "articable" και id το "articable_id". Για να γίνει περισσότερο κατανοητός ο συγκεκριμένος τρόπος συσχέτισης, χρειάζεται να αναφερθεί και και ο κώδικας από το άλλο model. Στο επόμενο παράδειγμα έχει επισυναφθεί ενδεικτικά ο κώδικας από το model "City"

```
    /* -- Relationship with articles (Polymorphic) -- */
    public function articles()
    {
        return $this->morphMany('App\Article', 'articable');
```

Εδώ γίνεται χρήσης της αντίστροφης μεθόδου, της morphMany(), η οποία είναι η συμπληρωματική της morphTo() ακριβώς επειδή όπως αναφέρθηκε και προηγουμένως, η συσχέτιση των δύο models είναι "ένα προς πολλά".

Συνεχίζοντας στον κώδικα του model Article, υπάρχουν η συνάρτηση user() μέσα στην οποία καλείται η μέθοδος belongsTo() και οι συναρτήσεις ratings(), likes() και images() μέσα στις οποίες καλείται η μέθοδος hasMany(). Είναι προφανές ότι οι συγκεκριμένες μέθοδοι που προσφέρει το Laravel, χρησιμοποιούνται για να σχηματίσουν συσχετίσεις "ένα προς πολλά" με τα αντίστοιχα models που δίνονται ως παράμετροι. Φυσικά, αυτές οι μέθοδοι χρησιμοποιούνται πάντα συμπληρωματικά και στα δυο models, ώστε η συσχέτιση να είναι αμφίδρομη. Οι επόμενες συναρτήσεις που υπάρχουν μέσα στο model Article χρησιμοποιούνται για να εμφανίζουν εικόνες και διάφορα άρθρα ανάλογα με το πόσο δημοφιλή είναι. Η τελευταία συνάρτηση με όνομα recommendedArticles() αφορά το σύστημα προτάσεων που έχει υλοποιηθεί και θα αναλυθεί ξεχωριστά στην επόμενη ενότητα.

Άλλα models που δημιουργήθηκαν μέσα στην εφαρμογή είναι το “ContactUS” το οποιο διαχειρίζεται τα πεδία της φόρμας της σελίδας επικοινωνίας, το “Like” και το “Rating” τα οποία αφορούν τις ομώνυμες λειτουργίες που υπάρχουν για τους εγγεγραμμένους χρήστες. Ιδιαίτερη αναφορά αξίζει το model “User”. Το συγκεκριμένο δημιουργείται αυτόματα από το Laravel σε κάθε νέα εφαρμογή. Το Laravel έχει ενσωματωμένο μηχανισμό για εγγραφή και σύνδεση των χρηστών και το συγκεκριμένο model χρησιμοποιείται για αυτό τον σκοπό. Για τις ανάγκες της παρούσας εφαρμογής, δημιουργήθηκαν συσχετίσεις με τα models “Rating”, “Like” και “Article” διότι χρειάζεται να καταγράφεται η πληροφορία για τον χρήστη που τα δημιουργεί. Μέσα στο model δημιουργήθηκαν επίσης και τρεις συναρτήσεις. Η πρώτη από αυτές με όνομα “publish()” είναι εκείνη που καλείται για να αποθηκεύσει ένα άρθρο και ταυτόχρονα να το αντιστοιχίσει με τον τρέχοντα χρήστη. Η δεύτερη είναι η “generateToken()” η οποία χρησιμοποιείται όταν ο χρήστης προσπαθήσει να συνδεθεί με τους κωδικούς του μέσω του API. Συγκεκριμένη συνάρτηση παράγει ένα μοναδικό αλφαριθμητικό 60 χαρακτήρων, το οποίο θα πρέπει να το χρησιμοποιήσει ο χρήστης σε κάθε αίτημα που στέλνει προς την εφαρμογή μέσω του API, έτσι ώστε να επαληθευτεί η ταυτότητα του. Η τελευταία συνάρτησα είναι η “mostViewedArticles()” η οποία χρησιμοποιείται για να επιστρέφει τα άρθρα που έχει επισκεφθεί ο τρέχον χρήστης. Χρησιμοποιείται στο σύστημα προτάσεων των άρθρων που αναλύεται στην επόμενη ενότητα.

Το δεύτερο σημαντικό στοιχείο στην αρχιτεκτονική της εφαρμογής είναι οι controllers. Στο Laravel οι controllers βρίσκονται πάντα μέσα στην διαδρομή “app/Http/Controllers”. Η ονομασία του συνήθως προκύπτει μεταφέροντας στον πληθυντικό το όνομα του αντίστοιχου model (τα ονόματα των models είναι πάντα στον ενικό) και συνήθως είναι κλάσεις που κληρονομούν από την κλάση “Controller”. Οι συναρτήσεις που περιέχουν τις περισσότερες φορές έχουν ονόματα ανάλογα με την λειτουργία CRUD που έχουν αντιστοιχηθεί (index, show, store, update, delete). Οι controllers που βρίσκονται μέσα στον φάκελο “Auth” έχουν δημιουργηθεί από το Laravel για την λειτουργία του authentication των χρηστών. Οι υπόλοιποι controllers φορτώνουν τις αντίστοιχες σελίδες εφόσον κάποιος χρήστης τις ζητήσει από την εφαρμογή. Ιδιαίτερη χρειάζεται να γίνει για τον controller των άρθρων. Ο constructor της κλάσης επιβάλλει να έχει γίνει ταυτοποίηση του χρήστη πριν καλεστούν οποιεσδήποτε μέθοδοι. Εξαιρέσεις είναι οι μέθοδοι index και show οι οποίες προβάλουν την λίστα και την σελίδα των άρθρων αντίστοιχα και έπρεπε να είναι προσβάσιμες προς όλους τους χρήστες της εφαρμογής. Οι controllers δεν φορτώνουν πάντα σελίδες της εφαρμογής. Κάποιοι από αυτούς επιτελούν και άλλες λειτουργίες όπως η επεξεργασία ή η αξιολόγηση του άρθρου. Η αξιολόγηση του άρθρου καθώς και το “like” που μπορεί να κάνει κάποιος εγγεγραμμένος χρήστης, δεν έχει κάποιο οπτικό αποτέλεσμα μέσα από τον controller. Το αίτημα στέλνεται στον controller με Ajax μέσω μιας φόρμας, και εκτελείται. Η επαναφόρτωση της σελίδας που παρατηρείται μετά της αξιολόγηση γίνεται από την Javascript, εφόσον ο server απαντήσει στον browser με μήνυμα επιτυχούς εκτέλεσης.

3.3.5 Σύστημα προτάσεων

Όπως αναφέρθηκε και πιο πάνω, στην εφαρμογή που υλοποιήθηκε για την παρούσα διπλωματική εργασία, προστέθηκε και κώδικας για σύστημα προτάσεων. Επειδή το άρθρο είναι εκείνη η οντότητα πάνω στην οποία έχει βασιστεί η εφαρμογή, επιλέχθηκε να δημιουργηθεί ένα σύστημα προτάσεων για άρθρα.

Ξεκινώντας χρειάζεται να αναφερθούν οι δυο διαφορετικοί τύποι προτάσεων που υλοποιήθηκαν στην εφαρμογή. Αρχικά, ο απλούστερος τύπος είναι εκείνος που προτείνει-παρουσιάζει τα δημοφιλέστερα άρθρα της εφαρμογής. Το μόνο που απαιτείται ώστε να υλοποιηθεί αυτό στο σύστημα, είναι να καταγράφονται το πόσες φορές έχουν προβληθεί τα άρθρα και στην συνέχεια να καλούνται εκείνα που έχουν συγκεντρώσει τις περισσότερες προβολές. Το μειονέκτημά του είναι ότι βασίζεται στην καταγραφή ενός πολύ γενικού γεγονότος, δηλαδή τον αριθμό των προβολών, πράγμα που έχει ως αποτέλεσμα όλοι οι χρήστες να βλέπουν τις ίδιες “προτάσεις”, ανεξάρτητα από το αν αυτές είναι κοντά στα ενδιαφέροντά τους. Αυτό το μειονέκτημα, στην συγκεκριμένη περίπτωση της τουριστικής εφαρμογής μάλλον εξομαλύνεται αφού οι τουριστικές “τάσεις” είναι κάτι που μεταβάλλεται συνεχώς, και τελικά ίσως να είναι χρήσιμο για τον τελικό χρήστη να γνωρίζει ποιες περιοχές είναι περισσότερο δημοφιλείς την τρέχουσα περίοδο. Για αυτόν ακριβώς τον λόγο υλοποιήθηκε και δεύτερο σύστημα

προτάσεων, το οποίο όμως βασίζεται στην βαθμολογία των άρθρων. Όπως έχει αναφερθεί και σε προηγούμενες ενότητες, οι εγγεγραμμένοι χρήστες έχουν την δυνατότητα να βαθμολογούν όλα τα δημοσιευμένα άρθρα. Όλες οι βαθμολογίες καταγράφονται, και έτσι προκύπτουν οι προτάσεις με τα καλύτερα βαθμολογημένα άρθρα. Αυτά τα δυο συστήματα προτάσεων εμφανίζουν τα αποτελέσματά τους στην αρχική σελίδα για όλους τους χρήστες, ανεξάρτητα από το αν είναι εγγεγραμμένοι ή όχι.

Ο δεύτερος τύπος προτάσεων που χρησιμοποιήθηκε είναι περισσότερο προσωποποιημένος. Απευθύνεται μόνο σε εγγεγραμμένους χρήστες, καθώς αποθηκεύει δεδομένα από την πλοήγηση τους στην εφαρμογή, ενώ τα αποτελέσματα του είναι διαφορετικά ανά χρήστη. Για να υλοποιηθεί αυτό το σύστημα προτάσεων είναι απαραίτητο ο χρήστης να έχει συνδεθεί στην εφαρμογή με τους προσωπικούς του κωδικούς. Έπειτα, στην αρχική σελίδα, στο slider που υπάρχει στο κάτω μέρος της, βλέπει όλα τα αποτελέσματα των προτεινόμενων άρθρων σύμφωνα. Τα αποτελέσματα αυτά προκύπτουν από τα άρθρα τα οποία έχει επισκεφθεί μέχρι εκείνη την χρονική στιγμή ο συγκεκριμένος χρήστης. Για να καταγραφούν τα άρθρων που επισκέπτεται ο εκάστοτε χρήστης, χρησιμοποιήθηκε ένα open source package του laravel, το “Laravel-activitylog”. Έτσι έχοντας ήδη αυτή την πληροφορία, και χρησιμοποιώντας τις συσχετίσεις του model των άρθρων με τα άλλα models, ήταν εφικτό να δημιουργηθούν προτάσεις παρόμοιων άρθρων.

Ο αλγόριθμος που χρησιμοποιήθηκε για το δεύτερο σύστημα προτάσεων επιστρέφει δέκα αποτελέσματα. Όπως αναφέρθηκε και πιο πριν, αν ο χρήστης δεν είναι εγγεγραμμένος, τότε δεν επιστρέφει προτάσεις. Επειδή όμως δεν ήταν επιθυμητό να μένει κενό το συγκεκριμένο slider στην αρχική σελίδα, αποφασίστηκε να εμφανίζει δέκα τυχαία άρθρα. Αν ο χρήστης έχει συνδεθεί με τους κωδικούς του, τότε ο κώδικας διαβάζει όλες τις εγγραφές από τον πίνακα log_activity που έχουν αντιστοίχιση με τον τρέχοντα χρήστη. Στην συνέχεια για κάθε άρθρο εντοπίζει την περιοχή με την οποία είναι αντιστοιχισμένη. Για παράδειγμα, αν ένα άρθρο είναι αντιστοιχισμένο με μια πόλη τότε, αναζητά και άλλες πόλεις που ανήκουν στον ίδιο δήμο, έπειτα επιστρέφει δήμους που ανήκουν στον ίδιο νομός και συνεχίζει κλιμακωτά προς τα πίσω έως ότου φτάσει στο επίπεδο των περιφερειών οπότε και σταματάει. Σημαντικό στοιχείο είναι ότι τα προτεινόμενα αποτελέσματα μεταβάλλονται και αυτά δυναμικά καθώς ο χρήστης περιηγείται στην εφαρμογή. Η επιβάρυνση του αλγόριθμου στην απόδοση της εφαρμογής είναι μικρή, ενώ μπορεί να βελτιωθεί ακόμα περισσότερο μελλοντικά, χρησιμοποιώντας μνήμη cache για τα προτεινόμενα αποτελέσματα η οποία θα ανανεώνεται κάθε ένα μικρό χρονικό διάστημα.

3.3.6 RESTful API

Η εφαρμογή που υλοποιήθηκε σε αυτήν την διπλωματική εργασία, προσεγγίζει χρήστες που έχουν στα ενδιαφέροντά τους τις μετακινήσεις και τον τουρισμό. Ενώ το διαδίκτυο έχει μεγάλη διάδοση τις τελευταίες δυο δεκαετίες, το ίδιο συμβαίνει και με τα ασύρματα δίκτυα και τις φορητές συσκευές. Αυτή η τάση διευρύνεται όλο και περισσότερο και γι’ αυτόν τον λόγο κρίθηκε αναγκαίο η εφαρμογή να είναι συμβατή με τις φορητές συσκευές ώστε να μπορεί να χρησιμοποιηθεί και εκεί. Στο οπτικό κομμάτι της εφαρμογής, όπως έχει ήδη αναφερθεί έχει γίνει ήδη πρόβλεψη και έχει εφαρμοστεί το λεγόμενο “responsive design”, το οποίο προσαρμόζει το γραφικό περιβάλλον της εφαρμογής σε οθόνες μικρού μεγέθους. Δεν καλύπτει όπως την περίπτωση των mobile applications, δηλαδή των εφαρμογών που είναι εξ’ αρχής δημιουργημένες να εκτελούνται στις φορητές συσκευές. Η παρούσα διπλωματική εργασία δεν καλύπτει την υλοποίηση του mobile application, αλλά υλοποιεί την διασύνδεση που θα χρειαστεί ένα mobile application ώστε να έχει πρόσβαση στα ίδια δεδομένα χωρίς να χρειάζεται να δημιουργηθεί νέα βάση δεδομένων και νέα λογική λογισμικού από το μηδέν. Έτσι γίνεται εκμετάλλευση των δομών που ήδη υπάρχουν και παράλληλα ελαχιστοποιείται το κόστος και ο χρόνος της εφαρμογής που θα υλοποιηθεί μελλοντικά για κινητές συσκευές.

Για να επιτευχθεί αυτή η συμβατότητα με οποιαδήποτε εφαρμογή δημιουργηθεί μελλοντικά για κινητές συσκευές ανεξάρτητα από την πλατφόρμα στην οποία θα εκτελείται, υλοποιήθηκε και υπηρεσία REST API. Ως REST (Representational state transfer) ορίζεται εκείνη η αρχιτεκτονική λογισμικού η οποία υλοποιεί την διαλειτουργικότητα μεταξύ διαφορετικών πληροφοριακών συστημάτων που επικοινωνούν μέσω διαδικτύου. Πρακτικά αυτό σημαίνει ότι η κύρια εφαρμογή

έχει δυνατότητα να παρέχει σε μια κινητή συσκευή, ακριβώς την ίδια πληροφορία που θα έδινε σε αντίστοιχη περίπτωση στον web browser. Εδώ δημιουργούνται δυο ερωτήματα, το πως θα αναγνωρίσει η εφαρμογή ότι την πληροφορία δε την ζητάει κάποιος web browser και σε τι μορφή θα είναι η απάντηση με τα δεδομένα που θα επιστρέψει. Τις απαντήσεις σε αυτά τα ερωτήματα τις δίνει το Laravel, το οποίο υποστηρίζει εξ' ορισμού RESTful υλοποιήσεις. Βασικό στοιχείο της αρχιτεκτονικής στην οποία έχει βασιστεί το Laravel είναι τα routes, δηλαδή η αντιστοίχιση ενός URI με κάποια ενέργεια (συνήθως κάποια μέθοδο ενός controller). Ένα απλό παράδειγμα route είναι το εξής:

Σε αυτή την περίπτωση, όταν η εφαρμογή αντιληφθεί ότι κάποιος χρήστης ζητάει τις πληροφορίες που αντιστοιχούν π.χ. στο URI *http:///localhost:3000/articles/1/* χρησιμοποιώντας την HTTP μέθοδο GET αναγνωρίζει το παραπάνω route και καλεί την μέθοδο show του controller με όνομα ArticlesController, η οποία φορτώνει στην οθόνη του χρήστη την σελίδα του άρθρου με id = 1. Τα routes δηλώνονται μέσα στο αρχείο που βρίσκεται στο routes/web.php. Για την περίπτωση δημιουργίας ενός RESTful service, το Laravel έχει διαθέσιμο το αρχείο php. Εδώ θα ανατρέξει η εφαρμογή, όταν κάποιος χρήστης ζητήσει ένα URI της μορφής οπότε γίνεται αντιληπτό ότι ο χρήστης ζητάει να του δοθούν τα δεδομένα σε διαφορετική μορφή. Και εδώ, η εφαρμογή αναγνωρίζει το URI και το αντιστοιχίζει με το route:

Route::get('articles/{article}', 'ArticleApiController@show')

Σε αυτή την περίπτωση, η εφαρμογή αφού λάβει ένα αίτημα με μέθοδο GET θα ενεργοποιήσει την μέθοδο show του ArticleApiController.

Δεν υπάρχει κάποιος τυποποιημένος τρόπος για την μορφή στην οποία θα πρέπει να δίνονται τα δεδομένα μέσω REST API. Συνηθίζεται όμως να είναι σε μία από τις επόμενες μορφές: HTML, XML, JSON. Στην παρούσα διπλωματική εργασία επιλέχθηκε η τρίτη επιλογή, δηλαδή το JSON. Το JSON (JavaScript Object Notation) είναι ένα πρότυπο κωδικοποίησης αρχείων και δεδομένων το οποίο δομεί τα δεδομένα σε πίνακες, χρησιμοποιώντας ζεύγη "κλειδιού"-"τιμής" ("attribute"-"value"). Το κλειδί είναι συνήθως μια μονολεκτική περιγραφή της πληροφορίας που περιλαμβάνεται στο πεδίο "τιμή". Αυτός ο τρόπος κωδικοποίησης είναι ανεξάρτητος από τις γλώσσες προγραμματισμού, γι' αυτό και χρησιμοποιείται ως ενδιάμεσο στάδιο για να επικοινωνήσουν συστήματα που είναι ασύμβατα μεταξύ τους. Επίσης, για να τονιστεί και η ομαδοποίηση των routes που προσφέρει το Laravel, στο αρχείο routes/api.php, κάποια routes είναι προσβάσιμα σε όλους τους χρήστες, ενώ κάποια άλλα απαιτούν να έχει γίνει εγγραφή και σύνδεση του χρήστη με τους προσωπικούς κωδικούς. Έτσι, ενώ τα URIs 'greater-regions', 'regions', municipalities' και 'cities' είναι ελεύθερα προσβάσιμα, τα URIs που αφορούν τα 'articles' έχουν ομαδοποιηθεί κάτω από το middleware με τίτλο 'auth:api', το οποίο ζητάει πριν από κάθε ενέργεια να πιστοποιηθεί ότι ο χρήστης είναι συνδεδεμένος με τους προσωπικούς του κωδικούς. Σε διαφορετική περίπτωση επιστρέφει μήνυμα λάθους. Σε περίπτωση επαλήθευσης, υπάρχουν διαθέσιμες μέθοδοι που υλοποιούν όλη την διαδικασία του CRUD.

3.4 Δομή της Βάσης Δεδομένων

Η βάση δεδομένων που επιλέχθηκε για την εφαρμογή είναι η Mysql. Το Laravel σαν framework υποστηρίζει φυσικά και άλλους τύπους βάσεων όπως Sqlite και Postgres αλλάζοντας απλά τις ρυθμίσεις της εφαρμογής στο αρχείο config/database.php. Όπως έχει αναφερθεί και σε προηγούμενο κεφάλαιο, η δομή της βάσης δεδομένων στο Laravel καθορίζεται άμεσα από τα models που υπάρχουν, ενώ η διαχείριση των πινάκων και των στηλών γίνεται μέσα από την διαδικασία των migrations, ενός συστήματος version control, όπου καταγράφονται με χρονολογική σειρά σε αρχεία όλες οι αλλαγές που γίνονται στην βάση. Το Laravel επίσης έχει μια λειτουργικότητα η οποία "γεμίζει" με προκαθορισμένα δεδομένα οποιαδήποτε βάση. Έτσι, η διαδικασία εγκατάστασης και έναρξης λειτουργίας της εφαρμογής σε έναν άλλο server απαιτεί ελάχιστες ρυθμίσεις.

Στην αρχιτεκτονική MVC έχει καθιερωθεί κάθε model να έχει τουλάχιστον έναν πίνακα στην βάση δεδομένων ώστε να αποθηκεύονται οι βασικές του πληροφορίες. Επίσης, έχει επικρατήσει η σύμβαση το όνομα του πίνακα να είναι ίδιο με το όνομα του model έτσι ώστε το laravel να κάνει άμεσα την σύνδεση χωρίς να χρειάζονται επιπλέον ρυθμίσεις και επίσης να διευκολύνεται η μελλοντική συντήρηση της εφαρμογής. Στην συνέχεια θα αναφερθούν με αλφαβητική σειρά όλοι οι πίνακες της βάσης δεδομένων καθώς και τα δεδομένα που αποθηκεύουν.

Ξεκινώντας, πρέπει να γίνει αναφορά στην κωδικοποίηση που θα χρησιμοποιεί η βάση δεδομένων. Επιλέχθηκε η κωδικοποίηση utf8 έτσι ώστε να υπάρχει συμβατότητα με τους ελληνικούς χαρακτήρες που χρησιμοποιούνται στην εφαρμογή καθώς και με ορισμένους ειδικούς χαρακτήρες που χρησιμοποιούνται πλέον στο διαδίκτυο. Ο πρώτος πίνακας είναι ο “activity_log” και χρησιμοποιείται για να καταγράφονται οι σελίδες που επισκέπτονται οι εγγεγραμμένοι χρήστες της εφαρμογής. Από εδώ, το σύστημα προτάσεων διαβάζει τα δεδομένα που χρειάζεται ώστε να προτείνει τα σχετικά άρθρα. Ο πίνακας ανήκει στα migrations του package “Laravel-activitylog” που χρησιμοποιήθηκε για να κατάγραφει τις προβολές των άρθρων. Από τις στήλες που περιλαμβάνει, χρησιμοποιούνται η subject_id, η subject_type, η causer_id και η causer_type. Στις δυο πρώτες στήλες αποθηκεύεται το άρθρου που προβλήθηκε ενώ στις δυο τελευταίες στήλες αποθηκεύεται ο χρήστης που άνοιξε την σελίδα του άρθρου. Η πληροφορία αποθηκεύεται σε δυο στήλες γιατί χρησιμοποιήθηκε ο “polymorphic” τύπος αποθήκευσης συσχετίσεων. Για παράδειγμα στην στήλη subject_id αποθηκεύεται το id που έχει η καταγεγραμμένη εγγραφή στον πίνακα της, και στην στήλη subject_type αποθηκεύεται το όνομα του model στο οποίο ανήκει η εγγραφή. Στην εικόνα 30, η πληροφορία που έχει αποθηκευτεί στην γραμμή με id=1 είναι ότι το άρθρο με id=70 προβλήθηκε από τον χρήστη με id=1. Στην γραμμή με id =6 έχει αποθηκευτεί η πληροφορία ότι το άρθρο με id=18 προβλήθηκε από τον χρήστη με id=2.

#	id	log_name	description	subject_id	subject_type	causer_id	causer_type	properties	created_at	updated_at
1	1	default	updated	70	App\Article	1	App\User	{"attribute...	2017-11-20 20:40:19	2017-11-20 20:40:19
2	2	default	updated	76	App\Article	1	App\User	{"attribute...	2017-11-20 20:41:26	2017-11-20 20:41:26
3	3	default	updated	18	App\Article	1	App\User	{"attribute...	2017-11-20 20:41:37	2017-11-20 20:41:37
4	4	default	updated	18	App\Article	1	App\User	{"attribute...	2017-11-20 20:41:41	2017-11-20 20:41:41
5	5	default	updated	18	App\Article	1	App\User	{"attribute...	2017-11-20 20:41:43	2017-11-20 20:41:43
6	6	default	updated	18	App\Article	2	App\User	{"attribute...	2017-11-20 20:41:44	2017-11-20 20:41:44
7	7	default	updated	6	App\Article	1	App\User	{"attribute...	2017-12-17 17:45:18	2017-12-17 17:45:18
8	8	default	updated	6	App\Article	1	App\User	{"attribute...	2017-12-17 17:45:31	2017-12-17 17:45:31
9	9	default	updated	1	App\Article	1	App\User	{"attribute...	2017-12-17 17:45:57	2017-12-17 17:45:57
10	10	default	updated	1	App\Article	3	App\User	{"attribute...	2017-12-17 17:48:49	2017-12-17 17:48:49
11	11	default	updated	1	App\Article	1	App\User	{"attribute...	2017-12-17 17:50:45	2017-12-17 17:50:45
12	12	default	updated	1	App\Article	3	App\User	{"attribute...	2017-12-17 17:50:57	2017-12-17 17:50:57
13	13	default	updated	1	App\Article	1	App\User	{"attribute...	2017-12-17 17:52:19	2017-12-17 17:52:19
14	14	default	updated	2	App\Article	1	App\User	{"attribute...	2017-12-17 17:52:33	2017-12-17 17:52:33
15	15	default	updated	2	App\Article	2	App\User	{"attribute...	2017-12-17 17:54:16	2017-12-17 17:54:16

Εικόνα 30: Πίνακας activity_log

Ο δεύτερος πίνακας είναι ο “article_images” (εικόνα 31) και ανήκει στο model “ArticleImage”. Εδώ αποθηκεύονται όλες οι εικόνες των άρθρων. Επιλέχθηκαν να αποθηκεύονται σε ξεχωριστό πίνακα και όχι πάνω στον ίδιο πίνακα που αποθηκεύονται τα άρθρα, έτσι ώστε να είναι ευκολότερη η συντήρησή του και η πιθανή επέκτασή του μελλοντικά. Στην στήλη “article_id” αποθηκεύεται το id του άρθρου στο οποίο ανήκει η εικόνα. Στην στήλη is_visible, η οποία είναι τύπου boolean, αποθηκεύεται η πληροφορία για το αν η εικόνα έχει επιλεγεί να είναι ορατή ή όχι. Στην στήλη “image_url” αποθηκεύεται το path που είναι αποθηκευμένη η εικόνα, ενώ η στήλη description έχειι προστεθεί για μελλοντική αναφορά και δεν χρησιμοποιείται προς το παρόν.

#	id	article_id	is_visible	image_url	description	created_at	updated_at
1	1	1	1	01/01.jpg	NULL	2017-11-20 20:40:11	NULL
2	2	1	1	01/02.jpg	NULL	2017-11-20 20:40:11	NULL
3	3	1	1	01/03.jpg	NULL	2017-11-20 20:40:11	NULL
4	4	1	1	01/04.jpg	NULL	2017-11-20 20:40:11	NULL
5	5	1	1	01/05.jpg	NULL	2017-11-20 20:40:11	NULL
6	6	2	1	02/01.jpg	NULL	2017-11-20 20:40:11	NULL
7	7	2	1	02/02.jpg	NULL	2017-11-20 20:40:11	NULL
8	8	2	1	02/03.jpg	NULL	2017-11-20 20:40:11	NULL
9	9	2	1	02/04.jpg	NULL	2017-11-20 20:40:11	NULL
10	10	2	1	02/05.jpg	NULL	2017-11-20 20:40:11	NULL
11	11	3	1	03/01.jpg	NULL	2017-11-20 20:40:11	NULL
12	12	3	1	03/02.jpg	NULL	2017-11-20 20:40:11	NULL
13	13	3	1	03/03.jpg	NULL	2017-11-20 20:40:11	NULL
14	14	3	1	03/04.jpg	NULL	2017-11-20 20:40:11	NULL
15	15	3	1	03/05.jpg	NULL	2017-11-20 20:40:11	NULL

Εικόνα 31: Πίνακας article_images

Ο επόμενος πίνακας είναι ο “articles” (εικόνα 32), ανήκει στο model “Article” και εδώ αποθηκεύονται τα άρθρα και όλες οι σημαντικές πληροφορίες που τα αφορούν. Η πρώτη βασική στήλη είναι η user_id, όπου αποθηκεύεται το id του χρήστη ο οποίος δημιούργησε το άρθρο. Στις στήλες “title”, “subtitle”, “body1”, “body2”, “body3”, “body4” αποθηκεύονται ο τίτλος και το κείμενο του άρθρου. Η στήλη “total_views” είναι ουσιαστικά ένας μετρητής που αποθηκεύεται ο συνολικός αριθμός που έχει προβληθεί το άρθρο. Οι στήλες “likes” και “rating” αποθηκεύονται αντίστοιχα τα likes και οι αξιολογήσεις των εγγεγραμμένων χρηστών. Η στήλη “rating” είναι τύπου double επειδή αποθηκεύονται ο μέσος όρος των αξιολογήσεων. Η επόμενη στήλη “is_public” είναι τύπου boolean, και δημιουργήθηκε για μελλοντική χρήση σε περίπτωση που χρειαστεί ο διαχειριστής της εφαρμογής να ελέγχει τα άρθρα που θα εμφανίζονται για παράδειγμα σε μια συγκεκριμένη σελίδα. Οι στήλες “image1_url”, “image2_url”, “image3_url”, “image4_url”, “image5_url” αρχικά δημιουργήθηκαν για να αποθηκεύουν το path των φωτογραφιών του άρθρου. Αποφασίστηκε όμως, να αποθηκεύονται τελικά σε ξεχωριστό πίνακα. Οι συγκεκριμένες στήλες έχουν παραμείνει για την περίπτωση όπου για λόγους βελτιστοποίησης της ταχύτητας της εφαρμογής, χρειαστούν οι φωτογραφίες του άρθρου χωρίς να γίνει query προς τον πίνακα των εικόνων. Οι επόμενες στήλες “articable_id” και “articable_type” αποθηκεύουν την περιοχή για την οποία έχει γραφτεί το άρθρο (δηλαδή περιφέρεια, νομό, δήμο ή πόλη). Όπως έχει αναφερθεί και σε προηγούμενες ενότητες, η πληροφορία αποθηκεύεται σε δυο στήλες διότι η συσχέτιση που έχει επιλεγεί είναι τύπου polymorphic. Έτσι στην articable_type αποθηκεύεται το όνομα του model (π.χ. “App/City) και στην άλλη στήλη αποθηκεύεται το αντίστοιχο id.

Column	Type	Default Value	Nullable	Character Set	Collation	Privileges	Extra
id	int(10) unsigned		NO			select,insert,update,references	auto_increment
user_id	int(11)		NO			select,insert,update,references	
title	varchar(255)		YES	utf8mb4	utf8mb4_unicode_ci	select,insert,update,references	
subtitle	varchar(255)		YES	utf8mb4	utf8mb4_unicode_ci	select,insert,update,references	
total_views	int(11)	0	NO			select,insert,update,references	
likes	int(11)	0	NO			select,insert,update,references	
rating	double(2,1)	0.0	NO			select,insert,update,references	
is_public	tinyint(1)	0	NO			select,insert,update,references	
body1	text		YES	utf8mb4	utf8mb4_unicode_ci	select,insert,update,references	
body2	text		YES	utf8mb4	utf8mb4_unicode_ci	select,insert,update,references	
body3	text		YES	utf8mb4	utf8mb4_unicode_ci	select,insert,update,references	
body4	text		YES	utf8mb4	utf8mb4_unicode_ci	select,insert,update,references	
image1_url	varchar(255)		YES	utf8mb4	utf8mb4_unicode_ci	select,insert,update,references	
image2_url	varchar(255)		YES	utf8mb4	utf8mb4_unicode_ci	select,insert,update,references	
image3_url	varchar(255)		YES	utf8mb4	utf8mb4_unicode_ci	select,insert,update,references	
image4_url	varchar(255)		YES	utf8mb4	utf8mb4_unicode_ci	select,insert,update,references	
image5_url	varchar(255)		YES	utf8mb4	utf8mb4_unicode_ci	select,insert,update,references	
articable_id	int(11)		NO			select,insert,update,references	
articable_type	varchar(255)		NO	utf8mb4	utf8mb4_unicode_ci	select,insert,update,references	
created_at	timestamp		YES			select,insert,update,references	
updated_at	timestamp		YES			select,insert,update,references	

Εικόνα 32: Πίνακας articles

Επόμενος πίνακα είναι ο “cities”, ο οποίος όμως θα παρουσιαστεί μαζί με τους “municipalities”, “regions” και “greater_regions” λόγω του ότι η δομή τους είναι παρόμοια. Οι προηγούμενοι πίνακες αντιστοιχίζονται με τα models “City”, “Municipality”, “Region” και “GreaterRegion”.

Column	Type	Default Value	Nullable	Character Set	Collation
id	int(10) unsigned		NO		
title	varchar(255)		YES	utf8mb4	utf8mb4_unicode_ci
greater_region_id	int(11)		YES		
region_id	int(11)		YES		
municipality_id	int(11)		YES		
likes	int(11)	0	NO		
views	int(11)	0	NO		
rating	double(2,1)	7.0	NO		
created_at	timestamp		YES		
updated_at	timestamp		YES		

Εικόνα 33: Πίνακας cities

Column	Type	Default Value	Nullable	Character Set	Collation
id	int(10) unsigned		NO		
title	varchar(255)		YES	utf8mb4	utf8mb4_unicode_ci
greater_region_id	int(11)		YES		
region_id	int(11)		YES		
likes	int(11)	0	NO		
views	int(11)	0	NO		
rating	double(2,1)	7.0	NO		
created_at	timestamp		YES		
updated_at	timestamp		YES		

Εικόνα 34: Πίνακας municipalities

Column	Type	Default Value	Nullable	Character Set	Collation
id	int(10) unsigned		NO		
title	varchar(255)		NO	utf8mb4	utf8mb4_unicode_ci
greater_region_id	int(11)		NO		
likes	int(11)	0	NO		
views	int(11)	0	NO		
rating	double(2,1)	7.0	NO		
created_at	timestamp		YES		
updated_at	timestamp		YES		

Εικόνα 35: Πίνακας regions

Column	Type	Default Value	Nullable	Character Set	Collation
id	int(10) unsigned		NO		
title	varchar(255)		NO	utf8mb4	utf8mb4_unicode_ci
likes	int(11)	0	NO		
views	int(11)	0	NO		
rating	double(2,1)	7.0	NO		
created_at	timestamp		YES		
updated_at	timestamp		YES		

Εικόνα 36: Πίνακας greater_regions

Οι κοινές στήλες που έχουν οι παραπάνω πίνακες είναι οι “title” όπου αποθηκεύεται ο τίτλος της περιοχής, οι “likes” και “views” οι οποίες είναι για μελλοντική χρήση, η στήλη “rating” όπου αποθηκεύεται ο μέσος όρος των αξιολογήσεων από όλα τα άρθρα που αφορούν την περιοχή. Οι στήλες “greater_region_id”, “region_id” και “municipality_id” δεν υπάρχουν σε όλους τους πίνακες και εξυπηρετούν στις συσχετίσεις των περιοχών π.χ. σε ποιο νομό και σε ποιόν δήμο ανήκει μια συγκεκριμένη πόλη.

Στην συνέχεια υπάρχουν κάποιοι πίνακες που θα επεξηγηθούν συνοπτικά καθώς είναι βοηθητικοί για την λειτουργία της εφαρμογής. Ο πίνακας “contactus” ανήκει στο model “ContactUS” και καταγράφει τα μηνύματα που αποστέλονται από την φόρμα της σελίδας επικοινωνίας της εφαρμογής. Συγκεκριμένα καταγράφει το όνομα του χρήστη, το email, το μήνυμα επικοινωνίας και την ημερομηνία και ώρα αποστολής. Έπειτα ακολουθεί οι πίνακας “migrations” τον οποίον τον δημιουργεί το Laravel ώστε να καταγράφει ποια migrations, δηλαδή ποιές αλλάγές στην βάση δεδομένων έχουν εκτελεστεί ώστε να μην τα εκτελέσει και την επόμενη φορά. Ο πίνακας “password_resets” έχει δημιουργηθεί επίσης από το Laravel και εδώ καταγράφονται τα tokens που δημιουργούνται αυτόματα σε περίπτωση που κάποιος χρήστης της εφαρμογής ζητήσει επαναδημιουργία του προσωπικού του κωδικού. Το ίδιο ισχύει και για τον πίνακα “users”, όπου καταγράφονται οι εγγεγραμμένοι χρήστες, με το όνομα που δήλωσαν κατά την εγγραφή τους, το email τους, τον προσωπικό τους κωδικό σε κωδικοοιημένη μορφή και τα tokens που απαιτούνται για επαλήθευση κατά την σύνδεση μέσω του API. Οι επόμενοι τρεις πινακες “permission_roles”, “permissions”, “permission_users”, “roles” και “role_users” δημιουργήθηκαν αυτόματα από το package Laravel-Backpack που χρησιμοποιήθηκε στην δεύτερη εφαρμογή της παρούσας διπλωματικής εργασίας ώστε να δημιουργηθεί ένα σύστημα διαχείρισης περιεχομένου και δεν έχουν κάποια χρησιμότητα αυτή την στιγμή. Οι τρεις τελευταίοι πίνακες είναι οι “tagging_tagged”, “tagging_tag_groups” και “tagging_tags”. Δημιουργήθηκαν και χρησιμοποιούνται από το package Laravel-Tagging, με το οποίο τα άρθρα μπορούν να ομαδοποιηθούν κάτω από “ετικέτες”, τα λεγόμενα tags, έτσι ώστε να είναι ευκολότερη η αναζήτηση τους.

Column	Type	Default Value	Nullable	Character Set	Collation
id	int(10) unsigned		NO		
tag_group_id	int(10) unsigned		YES		
slug	varchar(255)		NO	utf8mb4	utf8mb4_unicode_ci
name	varchar(255)		NO	utf8mb4	utf8mb4_unicode_ci
suggest	tinyint(1)	0	NO		
count	int(10) unsigned	0	NO		

Εικόνα 37: Πίνακας tagging_tags

Στον πίνακα “tagging_tags” (εικόνα 37) αποθηκεύονται τα tags που έχουν δημιουργηθεί με το όνομά τους στην στήλη “name”, το slug που θα εμφανίζεται στο URI τους και τον συνολικό αριθμό των άρθρων με τα οποία έχουν αντιστοιχηθεί στην στήλη “count. Η στήλη “tag_group_id” όπως και ο πίνακας “tagging_tag_groups” δεν χρησιμοποιούνται στην παρούσα φάση.

Column	Type	Default Value	Nullable	Character Set	Collation
id	int(10) unsigned		NO		
taggable_id	int(10) unsigned		NO		
taggable_type	varchar(255)		NO	utf8mb4	utf8mb4_unicode_ci
tag_name	varchar(255)		NO	utf8mb4	utf8mb4_unicode_ci
tag_slug	varchar(255)		NO	utf8mb4	utf8mb4_unicode_ci

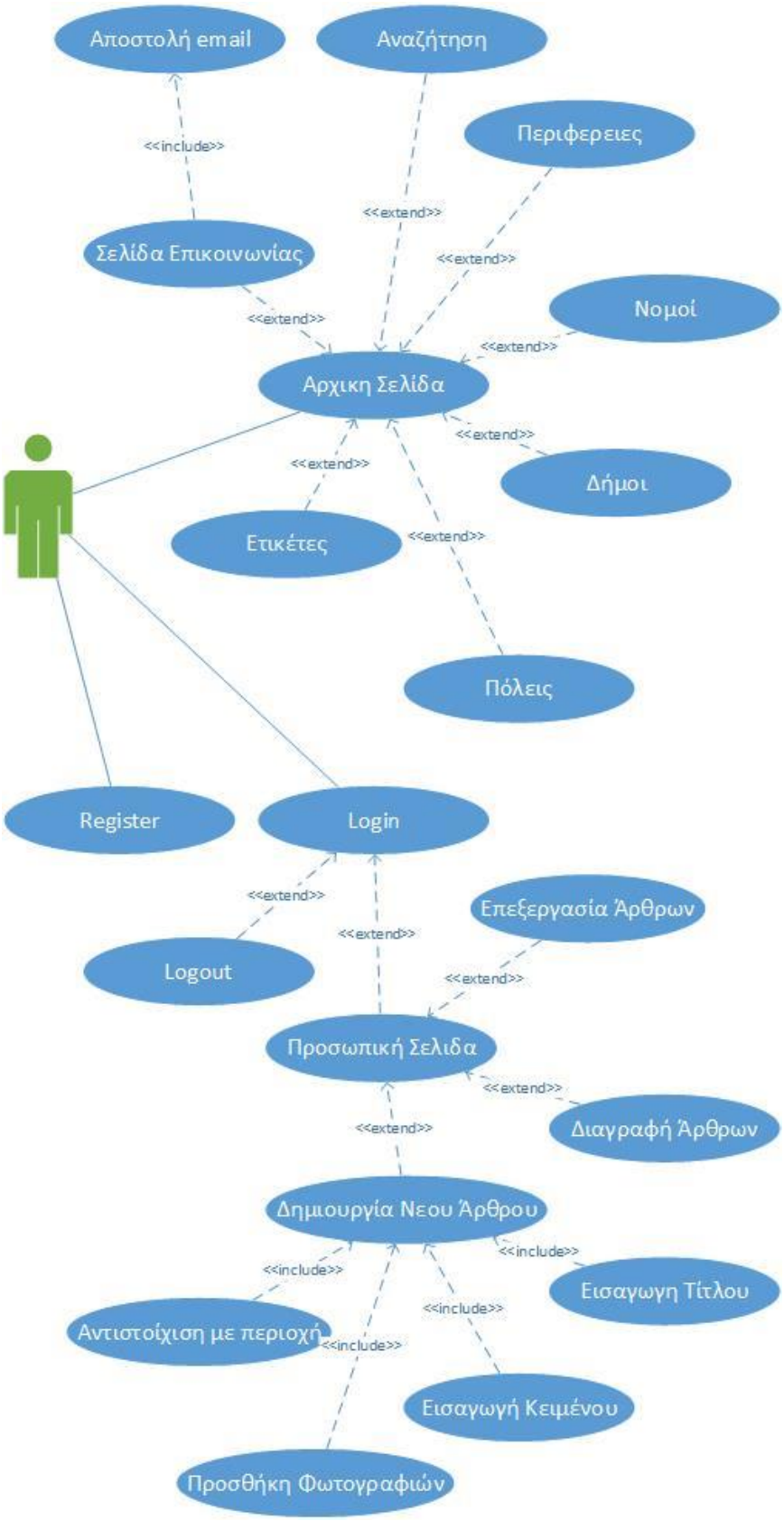
Εικόνα 38: Πίνακας tagging_tagged

Κλείνοντας, στον πίνακα “tagging_tagged” (εικόνα 38) αποθηκεύονται τα άρθρα που έχουν αντιστοιχηθεί με κάποιο tag. Η πληροφορία αυτή αποθηκεύεται στις δυο στήλες “taggable_id” και “taggable_type” με την χρήση του polymorphic relation. Στις στήλες “tag_name” και “tag_slug” αποθηκεύονται το όνομα και το slug του tag.

3.5 Διαγράμματα UML

Η γλώσσα UML (unified model language) είναι μια γλώσσα που χρησιμοποιείται στην σχεδίαση λογισμικού, ώστε να καταγράφονται, να επεξηγούνται και να τεκμηριώνονται οι προδιαγραφές ενός λογισμικού. Η χρήση της UML είναι ανεξάρτητη την γλώσσα προγραμματισμού στην οποία πρόκειται να υλοποιηθεί η εφαρμογή, βοηθάει όμως στην σχεδίαση εφαρμογών που βασίζονται σε αντικειμενοστραφείς γλώσσες προγραμματισμού και γι’ αυτόν τον λόγο η χρήση της είναι ευρέως διαδεδομένη. Τα βασικότερα στοιχεία που χρησιμοποιούνται στην σχεδίαση με την UML είναι οι οντότητες, τα διαγράμματα και οι σχέσεις. Σε αυτήν την ενότητα θα παρουσιαστούν ορισμένα διαγράμματα UML ώστε να γίνουν περισσότερο κατανοητές στον αναγνώστη της παρούσας διπλωματικής εργασίας, ορισμένες ιδιαίτερες εργασίες που επιτελούνται στην εφαρμογή. Έχει δοθεί περισσότερο βάρος στην επεξήγηση των λειτουργιών που εκμεταλλεύονται την αρχιτεκτονική MVC που χρησιμοποιήθηκε καθώς και τα βασικά της στοιχεία.

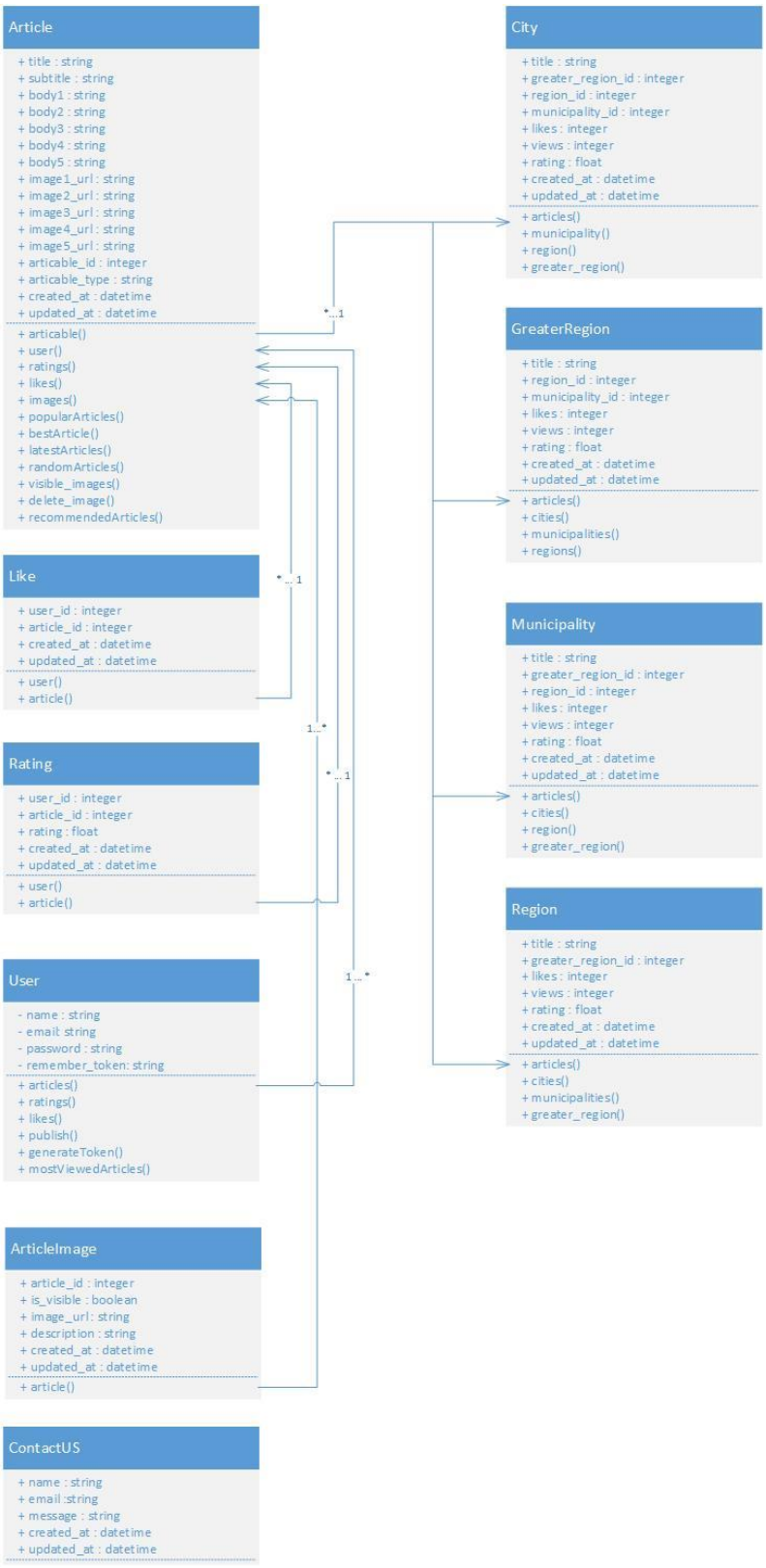
Αρχικά είναι αναγκαίο να παρουσιαστούν οι περιπτώσεις χρήσεων της εφαρμογής, έτσι ώστε να είναι σαφής ο τρόπος που μπορεί να αλληλεπιδράσει ο τελικός χρήστης με την εφαρμογή. Στην επόμενη εικόνα 39 παρουσιάζεται το διάγραμμα “use case” στο οποίο αναλύονται οι βασικές ακολουθίες κινήσεων που μπορεί να ακολουθήσει ο χρήστης καθώς και τις λειτουργίες που μπορεί να επιτελέσει άμεσα από οποιαδήποτε σελίδα της εφαρμογής. Οι λειτουργίες αυτές είναι οι login, logout και register και είναι διαθέσιμες σε κάθε σελίδα, ακριβώς επειδή θεωρούνται από τις βασικότερες. Το δεύτερο κομμάτι αφορά την πλοήγηση του χρήστη στην εφαρμογή. Θεωρήθηκε ως σημείο αναφοράς η αρχική σελίδα της εφαρμογής, από όπου ο χρήστης μπορεί να επισκεφθεί τα υπόλοιπα στοιχεία της εφαρμογής τα οποία είναι οι γεωγραφικές περιοχές της Ελλάδας (νομοί, περιφέρειες, δήμοι, πόλεις). Τέλος παρουσιάζονται και κάποιες δευτεύουσες περιπτώσεις όπως η αναζήτηση, οι σελίδα με τις ετικέτες των άρθρων και η σελίδα επικοινωνίας.



Εικόνα 39: Διάγραμμα χρήσεων

Προώθηση των περιοχών της Ελλάδας με την βοηθεια της κοινοτητας χρηστών μιας «έξυπνης» εφαρμογής, η οποία χρησιμοποιεί υπηρεσίες διαδικτύου

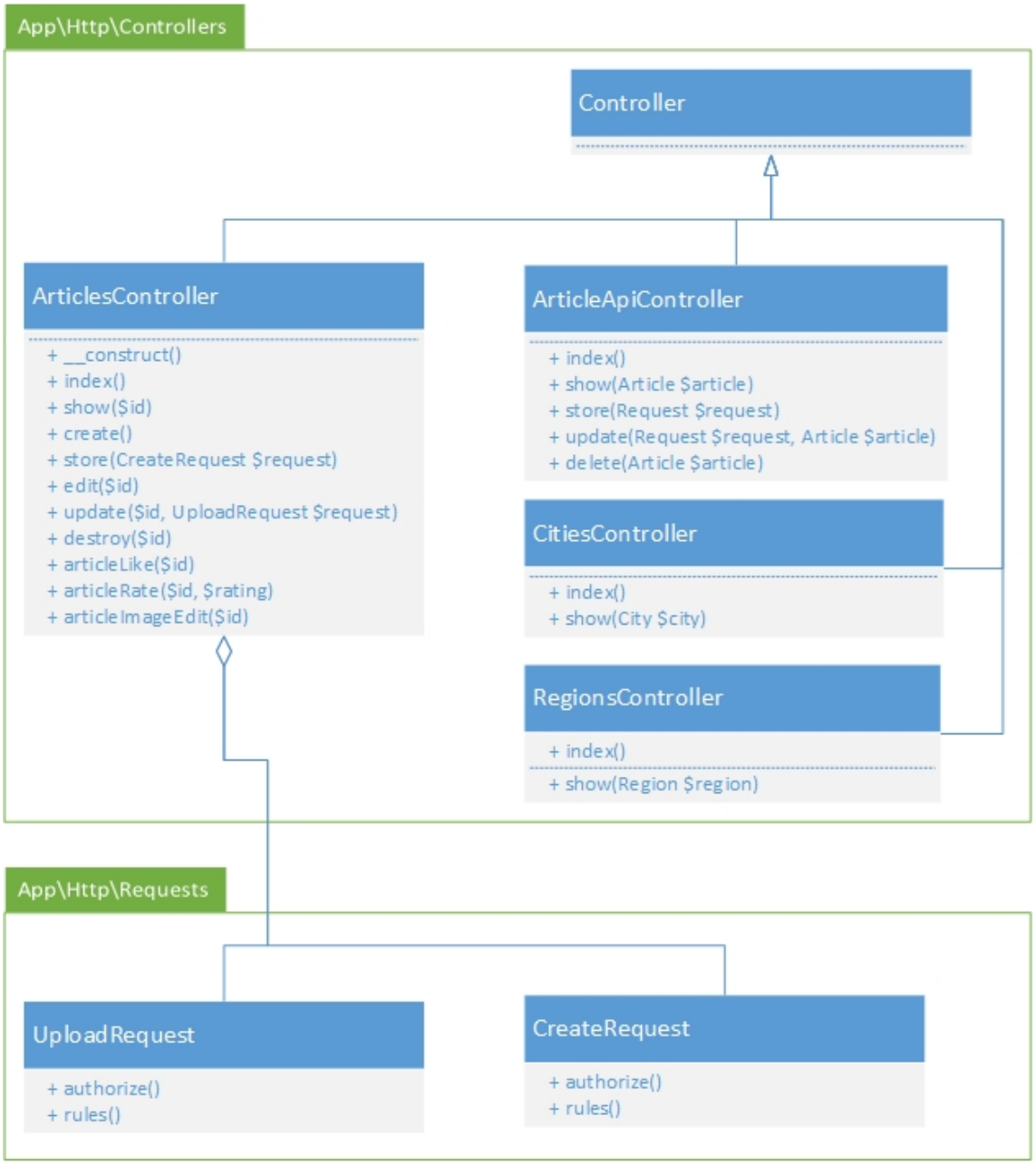
Το επόμενο διάγραμμα που παρουσιάζεται στην εικόνα 40 είναι το διάγραμμα κλάσεων όπου απεικονίζονται οι κλάσεις που δημιουργήθηκαν για τις ανάγκες της εφαρμογής.



Εικόνα 40: Διάγραμμα κλάσεων (models)

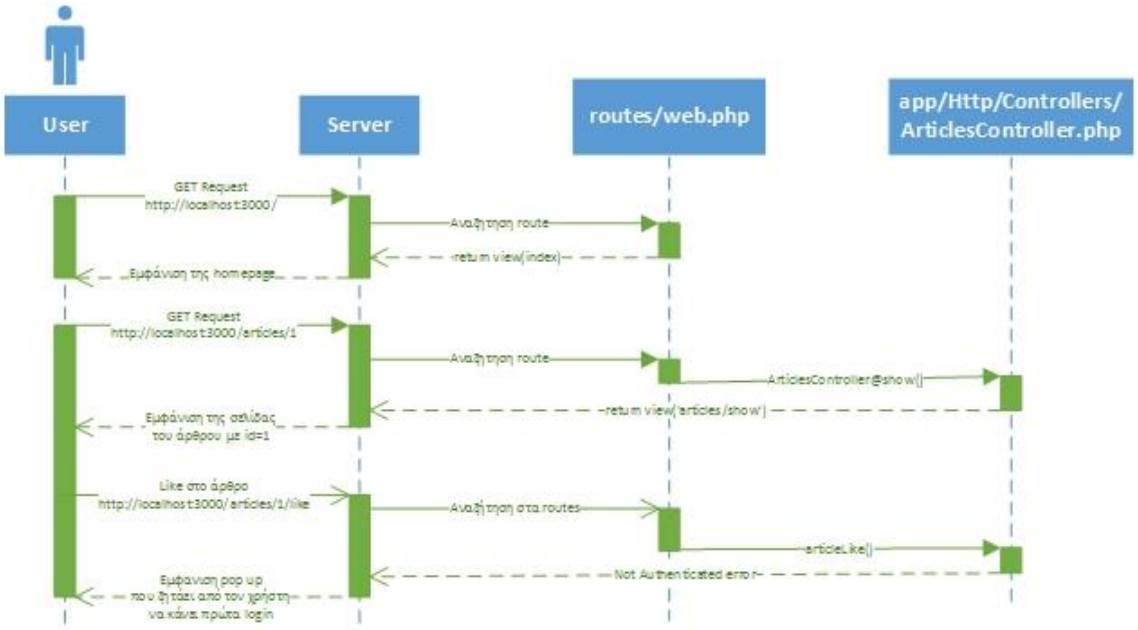
Προώθηση των περιοχών της Ελλάδας με την βοηθεια της κοινοτητας χρηστών μιας «έξυπνης» εφαρμογής, η οποία χρησιμοποιεί υπηρεσίες διαδικτύου

Οι κλάσεις που παρουσιάζονται, καλύπτουν την δομή model της αρχιτεκτονικής MVC. Όλες κληρονομούν από την κλάση “Model” η οποία υπάρχει εξ’ ορισμού μέσα στην βιβλιοθήκη του Laravel. Σε αυτό το διάγραμμα διακρίνονται επίσης και οι σχέσεις που έχουν οι κλάσεις μεταξύ τους, οι μεταβλητές τους καθώς και οι συναρτήσεις τους. Σε αυτό το σημείο χρειάζεται να αναφερθεί ότι και οι controllers, το δεύτερο στοιχείο της αρχιτεκτονικής MVC είναι κλάσεις οι οποίες και εκείνες κληρονομούν από την κλάση “Controller” του Laravel. Επειδή ο ρόλος τους είναι τα διαχειρίζονται τα requests που φτάνουν στον server από τους χρήστες της εφαρμογής, η δομή τους είναι αρκετά τυποποιημένη. Έτσι, για να γίνει αυτό αισθητό την επόμενη εικόνα 41 επιλέχθηκαν να παρουσιαστούν στο διάγραμμα κλάσεων ορισμένοι από τους controllers μαζί με τις αντίστοιχες κλάσεις που υλοποιούν την διαχείριση των requests για την δημιουργία των άρθρων και το upload των εικόνων των άρθρων από τον χρήστη.



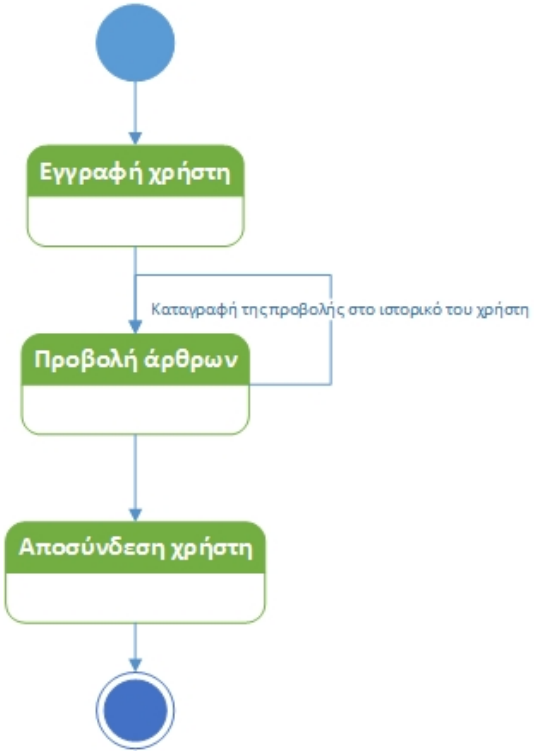
Εικόνα 41: Διάγραμμα κλάσεων για controllers και requests

Το επόμενο διάγραμμα που παρουσιάζεται στην εικόνα 42, έχει άμεση σχέση με το προηγούμενο. Είναι ένα διάγραμμα σειράς όπου παρουσιάζει την χρονική αλληλουχία των λειτουργιών της εφαρμογής, όταν κάποιος χρήστης της εφαρμογής προσπαθήσει να κάνει “Like” σε κάποιο άρθρο χωρίς να έχει προηγουμένως κάνει login.



Εικόνα 42: Διάγραμμα σειράς για την λειτουργια “like” ενός άρθρου

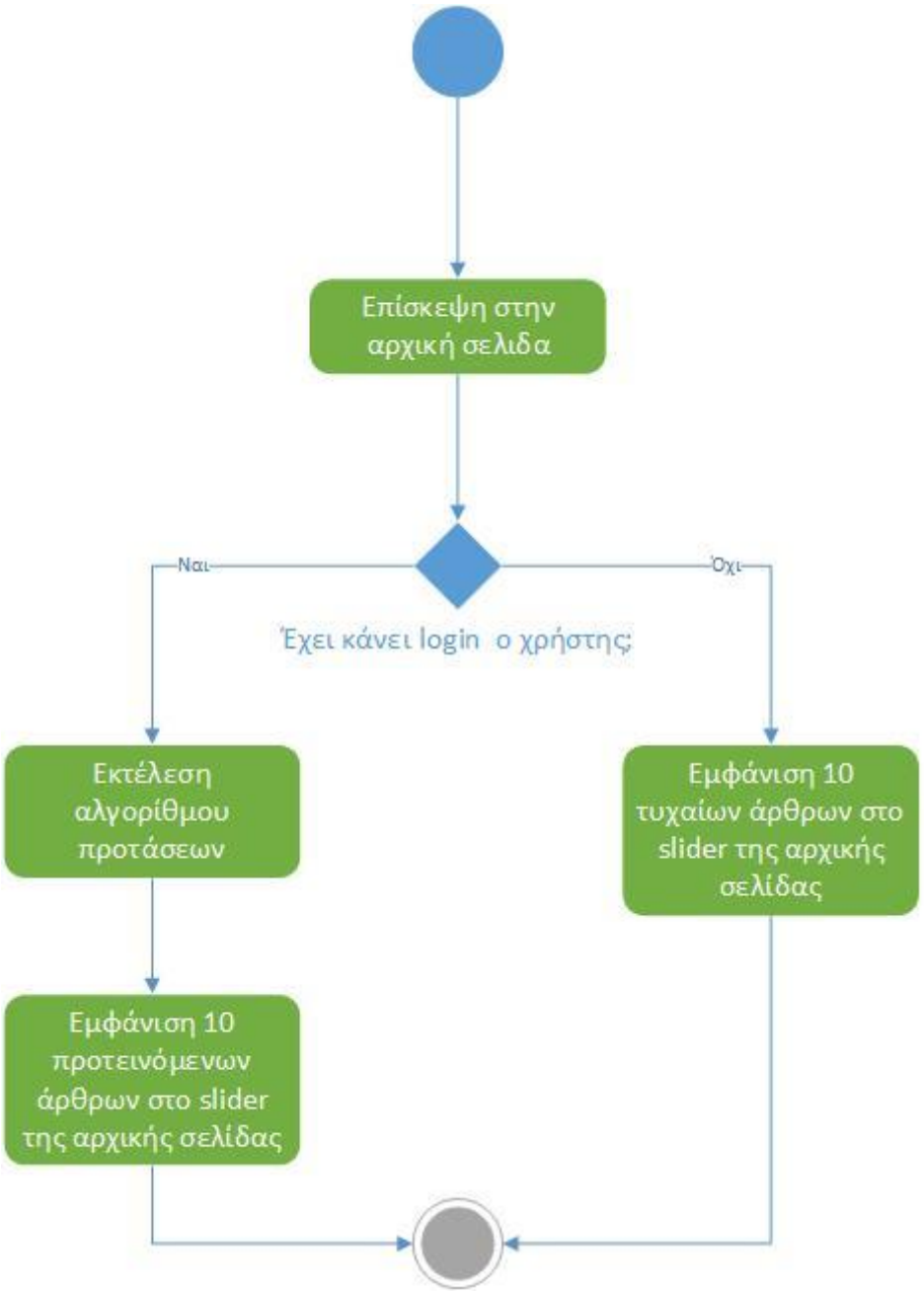
Τα δυο επόμενα διαγράμματα αφορούν το σύστημα προτάσεων που υλοποιήθηκε. Το πρώτο (εικόνα 43) είναι ένα διάγραμμα δραστηριοτήτων στο οποίο παρουσιάζεται το στιγμιότυπο καταγραφής των σελίδων που επισκέπτεται ο χρήστης κατά την περιήγησή του. Τα δεδομένα που προκύπτουν από αυτή την καταγραφή χρησιμοποιούνται στην συνέχεια από τον αλγόριθμο προτάσεων ώστε να εμφανιστούν οι προτάσεις που αφορούν τον τρέχοντα χρήστη.



Εικόνα 43: Διάγραμμα καταστάσεων για την καταγραφή του ιστορικού του χρήστη

Προώθηση των περιοχών της Ελλάδας με την βοηθεια της κοινοτητας χρηστών μιας «έξυπνης» εφαρμογής, η οποία χρησιμοποιεί υπηρεσίες διαδικτύου

Το δεύτερο (εικόνα 44) είναι ένα διάγραμμα δραστηριοτήτων. Αφορά την εκτέλεση του αλγορίθμου προτάσεων, η οποία προϋποθέτει από τον χρήση να έχει συνδεθεί με τους προσωπικούς του κωδικούς. Αυτό συμβαίνει επειδή ο αλγόριθμος λειτουργεί με τέτοιο τρόπο ώστε να δίνει εξατομικευμένες προτάσεις για άρθρα που πιθανόν να ενδιαφέρουν περισσότερο τον κάθε χρήστη. Σε αντίθετη περίπτωση το αποτέλεσμα που επιστρέφει είναι τυχαίο. Σε κάθε περίπτωση, έχει επιλεγεί να επιστρέφονται συνολικά ως 10 αποτελέσματα για λόγους ευχρηστίας.



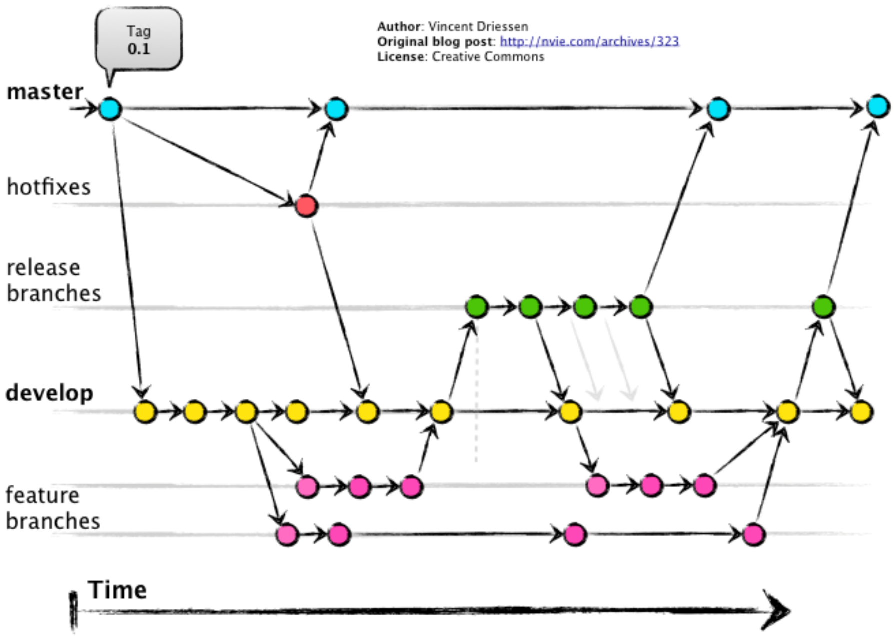
Εικόνα 44: Διάγραμμα δραστηριοτήτων για την εκτέλεση του αλγορίθμου προτάσεων

3.6 Σύστημα ελέγχου πηγαίου κώδικα

Δεδομένης της πολυπλοκότητας της εφαρμογής που αναπτύχθηκε, δημιουργήθηκε η ανάγκη για τον έλεγχο των ενδιάμεσων εκδόσεών της κατά της διάρκεια της ανάπτυξης. Για να καλυφθεί αυτή η ανάγκη χρησιμοποιήθηκε ένα εργαλείο ελέγχου πηγαίου κώδικα (version control system) και συγκεκριμένα το “GIT” το οποίο είναι λογισμικό ανοιχτού κώδικα. Αναπτύχθηκε από τον δημιουργό του linux, τον Linus Torvalds το 2005 για την ανάπτυξη του πυρήνα του λειτουργικού συστήματος. Από τότε έχει βελτιωθεί και η χρήση του θεωρείται ως μια καλή πρακτική για την ανάπτυξη και την συντήρηση ενός λογισμικού.

Εικόνα 40: Το λογότυπο του git

Η λογική στην οποία βασίζεται είναι η καταγραφή των σταδίων ανάπτυξης του λογισμικού, με τέτοιο τρόπο ώστε ο προγραμματιστής να έχει πρόσβαση σε όλη την ιστορικότητα της ανάπτυξης, και να έχει μια ακριβή εικόνα για τις αλλαγές που έγιναν στον κώδικα, ποια αρχεία επηρεάστηκαν, την χρονική που έγιναν και από ποιόν προγραμματιστή σε περίπτωση που η ανάπτυξη γίνεται από ομάδα με περισσότερα από ένα άτομα. Έτσι υπάρχει η δυνατότητα να ανιχνευτούν αμεσότερα τα bugs της εφαρμογής και κατά συνέπεια να διορθωθούν. Επίσης σε περίπτωση που πολλοί προγραμματιστές δουλεύουν πάνω στην ίδια εφαρμογή, υπάρχει σαφήνεια κατά την διάρκεια της υλοποίησης και ο κάθε ένας μπορεί να εργάζεται ανεξάρτητα από τον άλλον.



Εικόνα 41: Χρο

Ένα ακόμα σημαντικό πλεονέκτημα του GIT είναι η δυνατότητα που δίνει στους προγραμματιστές να δουλεύουν κάνοντας αλλαγές στον κώδικα, χωρίς να επηρεάζουν τα αρχεία

που γνωρίζουν ότι λειτουργούν σωστά μέχρι εκείνη την στιγμή. Δεν χρειάζεται επίσης να θυμούνται τι αλλαγές έχουν γίνει σε περίπτωση που κάτι δεν δουλεύει σωστά. Αρκεί να επανέλθουν στον τελευταίο χρονικό σημείο που έχει αποθηκευτεί και έτσι είναι απολύτως σίγουρο ότι η εφαρμογή δεν έχει επηρεαστεί καθόλου. Αυτό βοηθάει πολύ σε περίπτωση που η εφαρμογή βρίσκεται σε περιβάλλον παραγωγής και δεν υπάρχει δυνατότητα για λάθη. Μια από τις βασικές έννοιες του GIT είναι το branch. Ο προγραμματιστής δουλεύει στα αρχεία της εφαρμογής, τα αποθηκεύει και όταν ολοκληρώσει την εργασία του αποθηκεύει το γενικότερο στιγμιότυπο της εφαρμογής, δηλαδή το commit, μέσω του git. Αυτό εμφανίζεται στην εικόνα 41 ως κουκίδες, ενώ οι οριζόντιες γραμμές είναι τα branches. Το κάθε branch είναι αυτόνομο και δεν επηρεάζει τα υπόλοιπα, εκτός από την περίπτωση που ο προγραμματιστής επιλέξει να το ενώσει με κάποιο άλλο. Όπως φαίνεται και στην εικόνα, κάθε branch έχει τα διαφορετικά commits, ενώ σε κάποια σημεία έχουν ενωθεί με το κεντρικό branch, δηλαδή εκείνο με το όνομα “develop”.

Κλείνοντας, το git σαν λογισμικό υποστηρίζεται από όλα τα λειτουργικά συστήματα και μπορεί να εγκατασταθεί παντού. Τα τελευταία χρόνια, αρκετές εταιρείες το έχουν εξελίξει ακόμα περισσότερο προσφέροντας την δυνατότητα στους προγραμματιστές να αποθηκεύουν τα branches σε απομακρυσμένους servers. Έτσι η εφαρμογή είναι αποθηκευμένη στο cloud, που συνεπάγεται ότι είναι προσβάσιμη από κάθε σημείο στον κόσμο. Επίσης αυτές οι εταιρείες προσφέρουν εργαλεία για αυτόματη εκτέλεση test κάθε φορά που αποθηκεύεται ένα νέο commit, συμβάλλοντας έτσι στην καλύτερη αξιοπιστία του κώδικα και μειώνοντας την πιθανότητα για bugs.

4. Συμπεράσματα και μελλοντικές βελτιώσεις

Όπως αναφέρθηκε και στο πρώτο κεφάλαιο την εργασίας, η παρούσα εφαρμογή υλοποιήθηκε ώστε να εκτελεστεί σε περιβάλλον διαδικτύου και να τροφοδοτείται με περιεχόμενο από τους ίδιους του χρήστες της. Επίσης αναπτύχθηκε και ένα σύστημα προτάσεων που απευθύνεται μόνο στους εγγεγραμμένους χρήστες. Σε αυτό το κεφάλαιο θα παρατεθούν ορισμένες βελτιώσεις και λειτουργικότητες οι οποίες θα μπορούσαν να προστεθούν μελλοντικά μέσα στην εφαρμογή.

Αρχικά θα μπορούσε να γίνει βελτίωση του αλγορίθμου που δημιουργεί τα προτεινόμενα άρθρα. Θα μπορούσε να επιστρέφει αποτελέσματα με περισσότερη ακρίβεια σε ότι αφορά την γεωγραφική τοποθεσία. Αυτή την στιγμή λαμβάνεται υπ’ όψιν μόνο η ευρύτερη γεωγραφική περιοχή. Για παράδειγμά, για μια συγκεκριμένη πόλη προτείνονται άρθρα από άλλες πόλεις που υπάρχουν στον ίδιο δήμο και στον ίδιο νομό. Εδώ υπάρχει η δυνατότητα ο αλγόριθμος να γίνει ακόμα πιο συγκεκριμένος με την πρόσθεση και άλλων κριτηρίων. Ένα κριτήριο που θα μπορούσε να χρησιμοποιηθεί είναι τα tags, τα οποία ομαδοποιούν τα άρθρα, έτσι ώστε να προτείνονται άρθρα που ανήκουν σε γειτονική γεωγραφική περιοχή και έχουν κοινά tags. Ένα δεύτερο κριτήριο που θα μπορούσε να ληφθεί υπ’ όψιν από τον αλγόριθμο προτάσεων είναι το σύστημα αξιολόγησης και likes για τους εγγεγραμμένους χρήστες. Χρησιμοποιώντας την πληροφορία που ήδη αποθηκεύεται από την εφαρμογή στην βάση δεδομένων, μπορούν να δημιουργηθούν προτάσεις με δημοφιλείς περιοχές με καλές κριτικές, που βρίσκονται κοντά στις περιοχές που ενδιαφέρουν τον χρήστη. Επίσης υπάρχει η δυνατότητα να βελτιωθεί ο τρόπος με τον οποίο η εφαρμογή “καταλαβαίνει” ποιες περιοχές αρέσουν στον χρήστη. Αυτή την στιγμή το μόνο κριτήριο είναι το πόσες φορές έχει επισκεφθεί ένα χρήστης το άρθρο. Οι αξιολογήσεις και τα likes που έχει κάνει ο χρήστης είναι ακόμα δυο κριτήρια που δείχνουν τις προτιμήσεις του και έχουν την δυνατότητα να δώσουν αξιόλογες προτάσεις. Σε αυτή την λογική, μπορεί να προστεθεί στην σελίδα των άρθρων μια φόρμα όπου οι χρήστες μπορούν να αφήνουν τα προσωπικά τους σχόλια και κριτικές. Αυτό, εκτός από βελτίωση του συστήματος προτάσεων, αυξάνει την διαδραστικότητα της εφαρμογής και κάνει πιο ενδιαφέρουσα την εμπειρία χρήσης του τελικού χρήστη.

Το επόμενο στάδιο βελτίωσης του αλγορίθμου είναι η προσθήκη δυνατότητας ομαδοποίησης των χρηστών. Αυτή την στιγμή το σύστημα προτάσεων βασίζεται καθαρά επάνω στο ιστορικό που έχει καταγραφεί για τον ίδιο τον χρήστη. Με την ομαδοποίηση των χρηστών, ο αλγόριθμος θα γίνει πιο “έξυπνος” σε ότι αφορά την επεξεργασία των δεδομένων που έχουν ήδη καταγραφεί

στην βάση δεδομένων, καθώς θα έχει την δυνατότητα να εντοπίζει χρήστες που έχουν παρόμοιες καταγραφές στο ιστορικό τους και να τους ομαδοποιεί. Στην συνέχεια θα επεξεργάζεται τα δεδομένα από όλους τους χρήστες και οι προτάσεις στις οποίες θα καταλήγει θα είναι περισσότερο διευρυμένες. Έτσι, με αυτό τον τρόπο, θα έχει προστεθεί αυτόματα περισσότερη ευελιξία στο σύστημα προτάσεων, το οποίο θα παραμένει εξατομικευμένο, αλλά ταυτόχρονα θα εμπλουτίζεται συνεχώς με περισσότερα δεδομένα.

Σε ότι αφορά την απόδοση της εφαρμογής, στο άμεσο μέλλον θα βοηθούσε η προσθήκη cache στις σελίδες των άρθρων. Τμήματα που επαναχρησιμοποιούνται, όπως το slider που υπάρχει στο κάτω μέρος των σελίδων, μπορεί να αποθηκεύεται στην μνήμη έτσι ώστε να μην δημιουργείται κάθε φορά που φορτώνεται μια σελίδα, γιατί αυτό συνεπάγεται άσκοπα ερωτήματα προς την βάση δεδομένων και κατανάλωση πόρων του server. Τα τμήματα των σελίδων που θα επιλεγούν να αποθηκεύονται στην μνήμη cache μπορούν να ρυθμιστούν ώστε να ανανεώνεται το περιεχόμενο τους αφού περάσει κάποιο συγκεκριμένο χρονικό διάστημα. Με αυτό τον τρόπο θα επιτευχθεί γρηγορότερη απόκριση της εφαρμογής καθώς τα δεδομένα θα αντλούνται από την μνήμη και όχι από την βάση δεδομένων.

Επίσης, από την στιγμή που το περιεχόμενο το προσθέτουν οι χρήστες της εφαρμογής τίθεται το ερώτημα το πως θα φιλτράρεται η καταλληλότητα του περιεχόμενο και των εικόνων και των άρθρων που επιθυμούν να δημοσιεύσουν οι χρήστες. Αυτόματα, δημιουργείται η ανάγκη να υπάρχουν διαβαθμίσεις στους χρήστες με ανάλογα δικαιώματα επεξεργασίας και δημοσίευσης. Από την εφαρμογή διαχείρισης περιεχομένου, υπάρχει ήδη η δυνατότητα, αν και είναι ανενεργή προς το παρόν, να δημιουργηθούν “ρόλοι” που θα αντιστοιχηθούν με τον κάθε χρήστη. Σε κάθε ρόλο θα έχουν δοθεί συγκεκριμένα δικαιώματα. Έτσι, για παράδειγμα ένας χρήστης θα μπορεί να εγκρίνει άρθρα ώστε να προβληθούν στην κύρια εφαρμογή, κάποιο άλλος θα μπορεί να επεξεργάζεται άρθρα και εικόνες και κάποιος άλλος θα μπορεί να δημιουργεί ή και να διαγράφει χρήστες. Με αυτό τον τρόπο, κατανέμοντας τους ρόλους σε διαφορετικά άτομα, η γενικότερη διαχείριση της εφαρμογής γίνεται πιο αποδοτική.

Κλείνοντας, θα πρέπει να ληφθεί υπ’ όψιν ότι η παρούσα εφαρμογή βασίζεται σε δυναμικά δεδομένα, τα οποία εισάγονται από τους χρήστες. Ακριβώς λόγω αυτής της δυναμικότητας που έχει η παρούσα εφαρμογή, υπάρχουν πολλές πιθανότητες να προκύψουν ανάγκες οι οποίες δεν έχουν προβλεφθεί. Για να αντιμετωπιστεί αυτό το θέμα, χρειάζεται διαρκής ανατροφοδότησα από τους ίδιους τους χρήστες της εφαρμογής, μέσω της φόρμας επικοινωνίας και πιθανότατα μέσω της υλοποίησης μια νέας λειτουργικότητας που θα επιτρέπει να βαθμολογούν την ίδια την εφαρμογή, μέσω μιας προκαθορισμένης φόρμας, με τρόπο άμεσο και γρήγορο ώστε να μην παρεμποδίζεται η πραγματική ανάγκη του χρήστη, η οποία είναι να χρησιμοποιήσει την εφαρμογή.

5. Πηγές

13. Architectural Styles and the Design of Network-based Software Architectures, Roy Thomas Fielding, 2000
14. Web Engineering, Gerti Kappel, Birgit Proll, Siegfried Reich, Werner Retschitzegger, 2003
15. Web Application Architecture, Leon Shklar, Richard Rosen, 2003