



ΧΑΡΟΚΟΠΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΜΑΤΙΚΗΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

**Δημιουργία εφαρμογής πωλήσεων και δημοπρασιών
προϊόντων γεωργικής παραγωγής**

Στέφανος Φαφαλιός

Επιβλέποντες Καθηγητές:

**Γ. Δημητρακόπουλος,
Δ. Αναγνωστόπουλος,
Η. Βαρλάμης**

**ΑΘΗΝΑ
ΣΕΠΤΕΜΒΡΙΟΣ 2016**

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

**Δημιουργία εφαρμογής πωλήσεων και δημοπρασιών
προϊόντων γεωργικής παραγωγής**

Στέφανος Φαφαλιός

AM:it21135

Επιβλέποντες Καθηγητές:

**Γεώργιος Δημητρακόπουλος (Επικουρος Καθηγητής),
Δημοσθένης Αναγνωστόπουλος (Καθηγητής)**

Περίληψη

Τη σημερινή εποχή της κρίσης είναι ολοένα και πιο δύσκολο για έναν ελεύθερο επαγγελματία να ξεκινήσει μια καινούρια επιχείρηση. Πέρα από τα κόστη που απαιτούνται για την ίδρυση και λειτουργία της επιχείρησης, απαιτείται και μεγάλη επένδυση για την προώθηση της επιχείρησης και τη διαφήμισή της.

Ειδικότερα στις περιπτώσεις χονδρικής ή λιανικής πώλησης πολλοί επαγγελματίες επιλέγουν να επενδύσουν στις νέες τεχνολογίες ώστε να δημιουργήσουν νέα κανάλια προώθησης των προϊόντων τους. Επενδύουν έτσι σε δικτυακούς ιστοτόπους και καταστήματα τα οποία τους επιτρέπουν να τοποθετήσουν τον κατάλογο προϊόντων τους, να βρουν αγοραστές και να κλείσουν αγοραστικές συμφωνίες. Με την διάδοση των έξυπνων κινητών όμως, που διαθέτουν τις κατάλληλες τεχνολογίες για πρόσβαση στο διαδίκτυο, δημιουργούνται ολοένα και περισσότερες εφαρμογές που βασίζονται σε αυτό το χαρακτηριστικό για την εύκολη πρόσβαση του χρήστη σε κάθε είδους πληροφορία. Αυτός είναι και ο κυριότερος λόγος, που ο αριθμός e-shop είχε εκρηκτική άνοδο τα τελευταία χρόνια.

Η ραγδαία βελτίωση της τεχνολογίας και ο πολλαπλασιασμός των χρηστών smartphone τα κατέστησε μια ανερχόμενη πλατφόρμα στο χώρο των αγορών, παρόλα αυτά μέχρι σήμερα δεν υπήρξε αντίστοιχη πρόοδος στο χώρο των πωλήσεων. Ενώ υπάρχουν πολλές εφαρμογές για αγορά προϊόντων από δικτυακά καταστήματα, οι εφαρμογές αυτές έχουν αναπτυχθεί με χρήματα των επιχειρήσεων και περιέχουν αποκλειστικά τα προϊόντα τους.

Το όραμα πίσω από αυτή την εργασία, είναι η δημιουργία μιας εφαρμογής που θα απευθύνεται σε παραγωγούς και πωλητές προϊόντων και θα τους δίνει τη δυνατότητα να διατηρούν και να διαχειρίζονται ένα e-shop από τη smartphone συσκευή τους. Μιας εφαρμογής-μεσάζοντα η οποία θα διευκολύνει ανεξάρτητους παραγωγούς (π.χ., αγροτικών προϊόντων) στην εύρεση πελατών και στην διάθεση των προϊόντων τους, χωρίς να τους επιβαρύνει οικονομικά με μια αρχική επένδυση για την κατασκευή της.

Για το σκοπό αυτό αναπτύχθηκε ως έργο Ανοιχτού Λογισμικού και σχεδιάστηκε με τη λογική της πλατφόρμας client-server, ώστε να μπορεί να υποστηρίξει περισσότερους από ένα πωλητές και αγοραστές με μία μόνο εγκατάσταση. Με τον τρόπο αυτό στοχεύουν να είναι δυνατή η χρήση της εφαρμογής, από το μεγαλύτερο δυνατό ποσοστό χρηστών κινητών συσκευών, οποιαδήποτε στιγμή, και χωρίς να απαιτούνται ιδιαίτερες γνώσεις.

Abstract

Nowadays due to the economic crisis, it is increasingly difficult for freelancers and self-employed workers to start a new business. Apart from the costs required for the establishment and operation of a business, a major investment for the promotion of enterprise and advertisement is also required. Especially in the cases of wholesale or retail, many professionals choose to invest in new technologies to create new promotional channels for their products. As a result, they invest on web sites and stores that allow them to place their product catalog, find buyers and close purchasing deals. With the proliferation of smartphone, who have the appropriate technology to access the internet, more and more applications are created based on this feature for easy user access to any information. This is the main reason that the number of e-shops had explosive growth in the recent years.

The rapid improvement of technology and the proliferation of smartphone users made them a rising platform in the marketplace, however up to date there had not been made any corresponding progress in sales. While there are many applications for shopping at web stores, these applications have been developed with the business' money and contain only their products.

The vision behind this project is to create an application that is aimed at producers and sellers and enables them to maintain and manage an e-shop from their smartphone device. An application-intermediary which will enable independent producers (eg of agricultural products) to finding clients and marketing their products, without burdening them financially with an initial investment for construction.

For this purpose it was developed as an Open Source project and designed with the logic of the client-server platform, so that it can support more than one sellers and buyers with a single installation. Thus aim to enable the use of the application by the largest percentage of users of mobile devices possible, anytime, and without the requirement of any special knowledge.

Περιεχόμενα

Περίληψη.....	5
Abstract.....	7
Περιεχόμενα.....	9
Κεφάλαιο 1.....	11
Εισαγωγή.....	11
Κεφάλαιο 2.....	13
Θεωρητικό υπόβαθρο.....	13
2.1 Δημοπρασία/Πλειστηριασμός.....	13
2.2 Hybrid mobile app.....	15
2.3 JSON.....	16
2.4 Σχεσιακή βάση δεδομένων(Relational database)	19
2.5 API.....	20
2.6 MVC Framework.....	23
2.7 SDK.....	24
2.8 Ιδεατές μηχανές (Virtual Machines).....	25
Κεφάλαιο 3.....	26
ΑΠΑΙΤΗΣΕΙΣ.....	26
3.1 Λειτουργικές Απαιτήσεις	26
3.2 Μη λειτουργικές απαιτήσεις	28
3.3 Αρχιτεκτονική συστήματος	28
Κεφάλαιο 4	30
Υλοποίηση	30
4.1 Εργαλία ανάπτυξης και διαχείρισης λογισμικού	30
4.2 Frameworks	36
Κεφάλαιο 5	42
Παρουσίαση εφαρμογής.....	42
5.1 Back-End.....	42
5.2 Front-End.....	58
Κεφάλαιο 6	69
Συμπεράσματα και πιθανές επεκτάσεις	69
Βιβλιογραφία.....	70
Πίνακας εικόνων	72

Κεφάλαιο 1

ΕΙΣΑΓΩΓΗ

Σκοπός της εργασίας είναι η δημιουργία μιας εφαρμογής που θα απευθύνεται σε παραγωγούς και πωλητές προϊόντων και θα τους δίνει τη δυνατότητα, με τεράστια ευκολία, να διατηρούν και να διαχειρίζονται ένα e-shop από τη smartphone συσκευή τους.

Η εργασία υπήρχε σαν ιδέα από την ομάδα Ελεύθερου Λογισμικού / Λογισμικού Ανοικτού Κώδικα (ΕΛ/ΛΑΚ) του Χαροκοπείου Πανεπιστημίου, και συγκεκριμένα ξεκίνησε να υλοποιείται από τους Χρήστο Σαρδιανό και Γιάννη Κατάκη. Στα πλαίσια, της πτυχιακής μου εργασίας αποφάσισα να προχωρήσω την υλοποίηση της λειτουργικότητας που είχε αρχικά σχεδιαστεί, εστιάζοντας στις υπηρεσίες προς τον "διαθέτη/πωλητή" προϊόντων.

Σκοπός της εργασίας είναι να αποτελέσει μια ανοιχτή πλατφόρμα προώθησης αγροτικών προϊόντων για ανεξάρτητους παραγωγούς, λιανέμπορους, συνεταιρισμούς, σωματεία και πρωτίστως μικρομεσαίες αγροτικές επιχειρήσεις, οι οποίες δεν έχουν τη δυνατότητα να επενδύσουν χρήματα για τη διάθεση των προϊόντων τους ηλεκτρονικά αλλά επιθυμούν να πραγματοποιήσουν μόνοι τους τη διαχείριση και την προώθηση τους. Η επιχειρησιακή στρατηγική των εφαρμογών βασίζεται στην ενεργοποίηση εμπορών και πελατών με απώτερο σκοπό την επίτευξη της καλύτερης δυνατής τιμής αγοράς για τους τελικούς καταναλωτές καθώς και την διευκόλυνση της παροχής των προϊόντων από τους παραγωγούς, ειδικά για εκείνους που οι τρόποι παροχής των προϊόντων τους είναι περιορισμένοι.

Η εργασία επικεντρώθηκε στην υλοποίηση για συσκευές με λειτουργικό Android και iOS αλλά μπορεί με κάποιες τροποποιήσεις να χρησιμοποιηθεί και σε άλλα δημοφιλή λειτουργικά συστήματα καθώς χρησιμοποιήθηκε ένα cross-platform πλαίσιο ανάπτυξης για κινητές συσκευές, το ionic framework, που υποστηρίζει όλα τα βασικά λειτουργικά συστήματα κινητών (iOS, Android, Windows Mobile) και επιτρέπει την πραγματοποίηση της παραπάνω εργασίας. Η σχεδίαση της πλατφόρμας έγινε με τρόπο που να διευκολύνει την επεκτασιμότητά της και την ανεξάρτητη ανάπτυξη των εφαρμογών πωλητή και αγοραστή. Η εν' λόγω εργασία, για την διευκόλυνση των χρηστών της, αποφασίστηκε να χωριστεί σε δύο εφαρμογές, μια για τον πελάτη (client) και μια για τον παραγωγό (producer), οι οποίες θα επικοινωνούν μέσω της βάσης δεδομένων. Πιο συγκεκριμένα δημιουργήθηκε μία RESTful υπηρεσία API για την εύκολη πρόσβαση των δύο εφαρμογών στα δεδομένα της βάσης.

Παρακάτω θα γίνει η ανάλυση και η παρουσίαση της εφαρμογής για τον παραγωγό (producer) την οποία υλοποίησα. Ο κώδικας της εργασίας τόσο της restful υπηρεσίας αλλά και της κινητής εφαρμογής είναι δημόσιος και διαθέσιμος online [1][2].

Κεφάλαιο 2

Σε αυτό το κεφάλαιο αναλύονται οι έννοιες στις οποίες βασίζονται οι τεχνολογίες που χρησιμοποιήθηκαν για την ολοκλήρωση της εργασίας.

Θεωρητικό υπόβαθρο

Η εφαρμογή FarmIT απευθύνεται κυρίως σε ανεξάρτητους παραγωγούς και μικρομεσαίες αγροτικές επιχειρήσεις οι οποίες επιθυμούν να πραγματοποιήσουν μόνοι τους την προώθηση των προϊόντων τους μέσω του διαδικτύου. Το FarmIT απευθύνεται επίσης σε καταναλωτές που επιθυμούν να αγοράσουν αγροτικά προϊόντα σε όσο το δυνατόν καλύτερες τιμές απευθείας από τους παραγωγούς που τα παράγουν, χωρίς την ύπαρξη μεσαζόντων. Ο καλύτερος τρόπος για να επιτευχθεί η βέλτιστη τιμή είναι η δημιουργία δημοπρασιών.

2.1 Έννοιες Δημοπρασιών

2.1.1 ΓΕΝΙΚΑ

Πλειστηριασμός είναι μια κατάσταση κατά την οποία ένα αντικείμενο βγαίνει προς πώληση με έναν ιδιαίτερο τρόπο [3]. Αυτός ο ιδιαίτερος τρόπος ακολουθεί τα εξής βήματα:

1. Το αντικείμενο έχει μία αρχική τιμή εκκίνησης, απευθύνεται σε συγκεκριμένους ενδιαφερόμενους και λήγει σε περιορισμένο χρονικό διάστημα.
2. Ο κάθε ενδιαφερόμενος μπορεί να κάνει μία προσφορά σε τιμή μεγαλύτερη της αρχικής.
3. Έτσι το αντικείμενο αυτό αποκτάει καινούρια τιμή: τη μεγαλύτερη της προσφοράς που έκανε κάποιος.
4. Έπειτα, έχει δικαίωμα κάποιος άλλος με δεδομένη πλέον την καινούρια τιμή, να κάνει μεγαλύτερη προσφορά, και πάει λέγοντας.
5. Έτσι η τιμή του αντικειμένου ανεβαίνει συνέχεια βάσει των προσφορών που έχουν κάνει αυτοί που συμμετέχουν στον πλειστηριασμό αυτόν.
6. Στο τέλος κατά τη λήξη του, αυτός που έχει κάνει τη μεγαλύτερη προσφορά κερδίζει το αντικείμενο αυτό, καταβάλλοντας και αυτό το ποσό.

Αυτό το αντικείμενο μπορεί να είναι από κάτι μικρό (πχ μία "σπάνια" καρφίτσα), έως και κάτι μεγάλο (πχ αυτοκίνητο, σπίτι, κλπ).

Μια εναλλακτική ονομασία για αυτή τη διαδικασία πώλησης αγαθών, χωρίς προσδιορισμό της μέγιστης ή ελάχιστης τιμής είναι η **δημοπρασία** [4].

1. Η διαδικασία αυτή είναι δημόσια, δηλαδή ανοιχτή σε όλους και γίνεται με την μέθοδο του πλειστηριασμού.
2. Ουσιαστικά γίνονται προσφορές για τα αγαθά που δημοπρατούνται και η καλύτερη προσφορά κερδίζει.
3. Με απλά λόγια, αυτός που κάνει την καλύτερη προσφορά, δηλαδή προσφέρει τα περισσότερα χρήματα αποκτά το αγαθό πληρώνοντας το αντίτιμο.
4. Η δημοπρασία γίνεται συνήθως για περιουσιακά στοιχεία και αγαθά μεγάλης αξίας.
5. Μια άλλη περίπτωση που γίνεται δημοπρασία είναι για κάποιο έργο του δημοσίου. Μόλις προκηρυχθεί το έργο, οι ενδιαφερόμενοι στέλνουν φάκελο με την προσφορά τους και αυτός που κάνει την καλύτερη προσφορά αναλαμβάνει το έργο.

2.1.2 ΕΙΔΗ ΔΗΜΟΠΡΑΣΙΩΝ [5]

1. Απόλυτη Δημοπρασία (ή δημοπρασία χωρίς επιφύλαξη)

Το αντικείμενο πωλείται στον υψηλότερο πλειοδότη, ανεξάρτητα από την τιμή. Δεδομένου ότι μια πώληση έχει εξασφαλιστεί, ο ενθουσιασμός από μεριάς του αγοραστή και η συμμετοχή είναι αυξημένη. Δημιουργεί μέγιστη ανταπόκριση από την αγορά. Πολλοί πωλητές, συμπεριλαμβανομένων των χρηματοπιστωτικών ιδρυμάτων και κυβερνητικών υπηρεσιών έχουν αρχίσει να χρησιμοποιούν αυτή τη μέθοδο πιο συχνά.

2. Δημοπρασία Ελάχιστης προσφοράς

Ο διοργανωτής της δημοπρασίας θα δεχθεί προσφορές σε ή πάνω από μια δημοσιευμένη ελάχιστη τιμή. Αυτή η ελάχιστη τιμή είναι πάντα αναφερόμενη στα φυλλάδια και τις διαφημίσεις και ανακοινώνεται και στη δημοπρασία. Υπάρχει μειωμένος κίνδυνος για τον πωλητή, από την στιγμή που η τιμή πώλησης θα πρέπει να είναι πάνω από ένα ελάχιστο αποδεκτό επίπεδο. Οι αγοραστές γνωρίζουν ότι θα είναι σε θέση να αγοράσουν στην τιμή του ελάχιστου και πάνω. Ο πωλητής, ωστόσο, να περιορίζει το ενδιαφέρον στη δημοπρασία, μόνο σε εκείνους τους αγοραστές που είναι πρόθυμοι να πληρώσουν την ελάχιστη τιμή προσφοράς. Ως εκ τούτου, η ελάχιστη τιμή θα πρέπει να είναι αρκετά μικρή για να λειτουργήσει ως κίνητρο και όχι ως εμπόδιο.

3. Αποθεματική Δημοπρασία (μια δημοπρασία που υπόκειται σε επιβεβαίωση)

Σε αυτό το σενάριο, η βαρύτητα της υψηλότερης προσφοράς μειώνεται.

Η ελάχιστη προσφορά δεν δημοσιεύεται, και ο πωλητής διατηρεί το δικαίωμα να δεχθεί ή να απορρίψει την υψηλότερη προσφορά εντός καθορισμένου χρόνου - οποιαδήποτε στιγμή από αμέσως μετά τη δημοπρασία έως και 72 ώρες μετά την λήξη της.

Οι πωλητές προκαθορίζουν την τιμή στην οποία θα πωληθεί το αντικείμενο και δεν είναι υποχρεωμένοι να επιβεβαιώσουν μια πώληση, εκτός αν γίνει σε τιμή που είναι απόλυτα αποδεκτή σε αυτούς.

Το κύριο μειονέκτημα της Αποθεματικής Δημοπρασίας είναι ότι οι εν' δυνάμει αγοραστές μπορεί να μην επιθυμούν να επενδύσουν χρόνο και χρήματα, όταν δεν υπάρχει καμία βεβαιότητα ότι θα είναι σε θέση να αγοράσουν, ακόμη και αν έχουν κάνει την υψηλότερη προσφορά.

4. Αντίστροφη Δημοπρασία

Μια αντίστροφη δημοπρασία είναι ένα είδος πλειστηριασμού στον οποίο έχουν αντιστραφεί οι ρόλοι του αγοραστή και του πωλητή. Σε μια συνηθισμένη δημοπρασία, οι αγοραστές ανταγωνίζονται για την απόκτηση αγαθών ή υπηρεσιών, προσφέροντας όλο και υψηλότερες τιμές.

Σε μια αντίστροφη δημοπρασία, οι πωλητές ανταγωνίζονται για την απόκτηση αγαθών ή υπηρεσιών από τον αγοραστή και οι τιμές μειώνονται κατά κανόνα όσο οι πωλητές μειοδοτούν ο ένας τον άλλον.

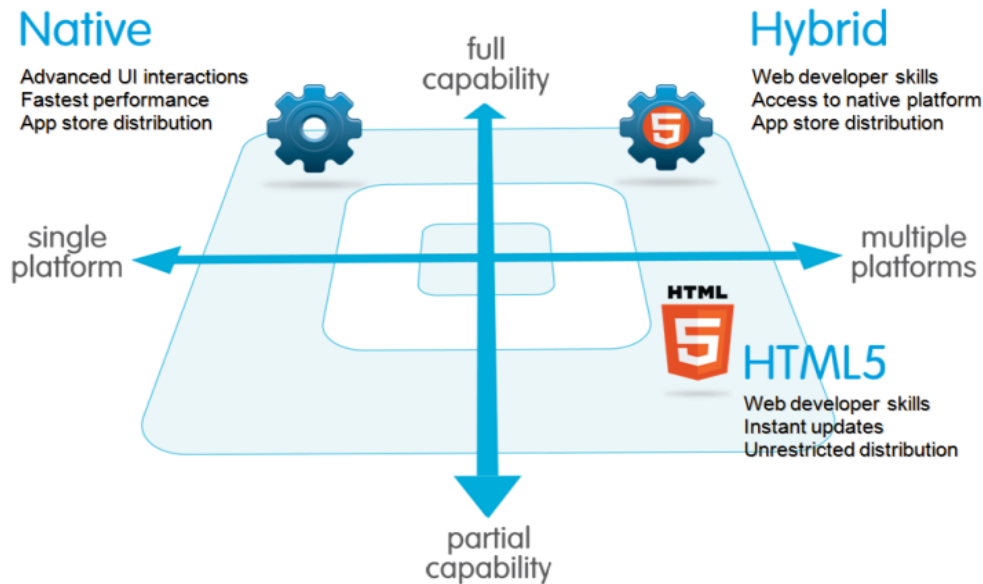
Στην δική μας περίπτωση, το είδος δημοπρασίας που χρησιμοποιείται είναι, αντίστροφη δημοπρασία μέγιστης προσφοράς.

Στη δική μας περίπτωση υποστηρίζονται α) η δημοπρασία ελάχιστης προσφοράς όταν ο πωλητής καθορίζει την κατώτερη τιμή πώλησης των προϊόντων του και οι αγοραστές πλειοδοτούν μέχρι την κατακύρωση, β) η αντίστροφη δημοπρασία όταν ένας ή περισσότεροι πωλητές αποφασίσουν να μειοδοτήσουν για να καλύψουν το αίτημα ενός αγοραστή. Για να αποφύγουμε την περίπτωση της Αποθεματικής Δημοπρασίας κάθε ποσότητα που τίθεται σε δημοπρασία δεσμεύεται προσωρινά μέχρι να κλείσει η δημοπρασία και δεν μπορεί να χρησιμοποιηθεί σε άλλη δημοπρασία. Είναι ευθύνη του πωλητή να προσθέσει ανά πάσα στιγμή πραγματικό απόθεμα στην εφαρμογή και να διασφαλίσει τη διάθεσή του, διαφορετικά θα αξιολογηθεί αρνητικά να δεν μπορεί να προσφέρει την κατοχυρωμένη ποσότητα. Αντίστοιχα και για τον αγοραστή που δεν θα προχωρήσει σε κάποια αγορά που έχει δεσμεύσει.

2.2 Τεχνολογικό υπόβαθρο

Η εφαρμογή που αναπτύχθηκε έχει δύο βασικά τμήματα: α) το server (με τη ΒΔ όπου καταχωρούνται τα δεδομένα της εφαρμογής) όπου συντονίζει όλη τη διαδικασία αποθεμάτων και πωλήσεων, β) τον client μια εφαρμογή για κινητά τηλέφωνα που επικοινωνεί με το server για οποιαδήποτε ενέργεια. Η εφαρμογή client έχει υλοποιηθεί ως υβριδική εφαρμογή κινητού που θα μπορούσε να είναι διαθέσιμη και μέσω browser.

Ορίζουμε υβριδική (hybrid), μια διαδικτυακή εφαρμογή η οποία, κατά κύριο λόγο κατασκευάστηκε με τη χρήση HTML5 και JavaScript, τρέχει σε περιβάλλον browser μέσα σε μία εφαρμογή, και παράλληλα έχει πρόσβαση στη πλατφόρμα και τις λειτουργίες μιας έξυπνης συσκευής. Το ionic είναι ένα από τα πιο δημοφιλή παραδείγματα framework για τη δημιουργία υβριδικών εφαρμογών για κινητά.



Εικόνα 1: Native vs Hybrid

Η δημιουργία μιας τέτοιου είδους εφαρμογής έχει αρκετά πλεονεκτήματα σε σχέση με την ανάπτυξη εφαρμογών στοχευμένων σε συγκεκριμένα λειτουργικά συστήματα καθώς κερδίζουμε τόσο σε χρόνο ανάπτυξης όσο και υποστήριξης αφού μπορεί με κάποιες τροποποιήσεις να χρησιμοποιηθεί σε διαφορετικές πλατφόρμες, όπως είναι οι ευρέως διαδεδομένες android, windows phone και iOS. Επιπλέον το κόστος ελαχιστοποιείται αφού για παράδειγμα αν θέλουμε να υλοποιήσουμε μία εφαρμογή για να καλύψουμε δύο λειτουργικά συστήματα όπως το Android και το iOS χρειάζεται μία ομάδα προγραμματιστών. Ακόμη συνδυάζει τις τεχνολογίες του διαδικτύου με τις τεχνολογίες των συσκευών.

Το μειονέκτημα μίας υβριδικής εφαρμογής σε σύγκριση με μία native είναι ο χρόνος απόκρισης. Μία υβριδική εφαρμογή επικοινωνεί με τον browser της συσκευής και ο browser με το λειτουργικό, ενώ μία native εφαρμογή επικοινωνεί απευθείας με το λειτουργικό. Έτσι στις υβριδικές εφαρμογές παρεμβαίνει ένα επιπλέον στάδιο, καθιστώντας τις πιο αργές .

Συμπερασματικά, η επιλογή δημιουργίας μιας εφαρμογής ως native είναι καλύτερη όταν θέλουμε να δημιουργήσουμε μία πολύπλοκη εφαρμογή, έχουμε ως στόχο την επίδοση και ο χρόνος και το κόστος δεν μας ενδιαφέρουν. Αντίθετα η επιλογή δημιουργίας μίας εφαρμογής ως υβριδική είναι προτιμότερη όταν θέλουμε γρήγορη ανάπτυξη του λογισμικού, ιδιαίτερα αν επιθυμούμε παράλληλη ανάπτυξη της εφαρμογής σε περισσότερες από μία πλατφόρμες, εύκολη συντήρηση και περιορισμούς στο κόστος.

Αν και δεν έχει ερευνηθεί το ποσοστό των υβριδικών εφαρμογών στην αγορά, το ενδιαφέρον για τα απαραίτητα frameworks για την υλοποίησή τους συνεχώς αυξάνεται. Το 2013 η Gartner Inc. έκανε την πρόβλεψη ότι μέχρι το 2016 περισσότερες από το 50% των εφαρμογών θα είναι υβριδικές [6].

2.3 JSON

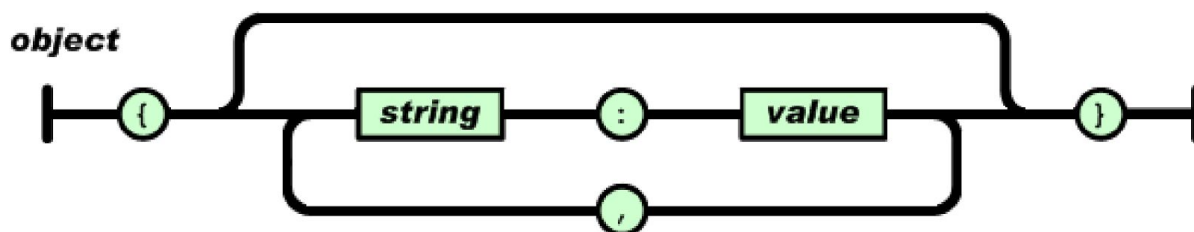
JSON (JavaScript Notation Object) είναι ένα ελαφρύ πρότυπο ανταλλαγής δεδομένων. Είναι εύκολο για τους ανθρώπους να διαβάσουν και να γράψουν. Είναι εύκολο για τις μηχανές να το αναλύσουν και να το δημιουργήσουν. Βασίζεται σε ένα υποσύνολο της γλώσσας προγραμματισμού JavaScript. JSON είναι μια μορφή κειμένου που είναι εντελώς ανεξάρτητη από τις γλώσσες προγραμματισμού, αλλά χρησιμοποιεί κανόνες(conventions) τους οποίους γνωρίζουν προγραμματιστές της C-οικογένειας γλωσσών, συμπεριλαμβανομένων των C , C ++, C #, Java, JavaScript, Perl, Python, και πολλοί άλλοι. Αυτές οι ιδιότητες καθιστούν το JSON μια ιδανική γλώσσα δεδομένων ανταλλαγή [7].

Το JSON είναι χτισμένο σε δύο δομές:

- Μια συλλογή από ζεύγη ονόματος μεταβλητής / τιμής. Σε πολλές γλώσσες, αυτό υλοποιείται ως ένα αντικείμενο, καταχώριση, δομή, λεξικό, πίνακα hash, λίστα κλειδιών, ή προσαρμοστικού πίνακα (object, record, struct, hash table, keyed list or associative array).
- Μια ταξινομημένη λίστα τιμών. Στις περισσότερες γλώσσες προγραμματισμού, αυτή πραγματοποιείται ως μια σειρά, διάνυσμα, λίστα, ή ακολουθία.

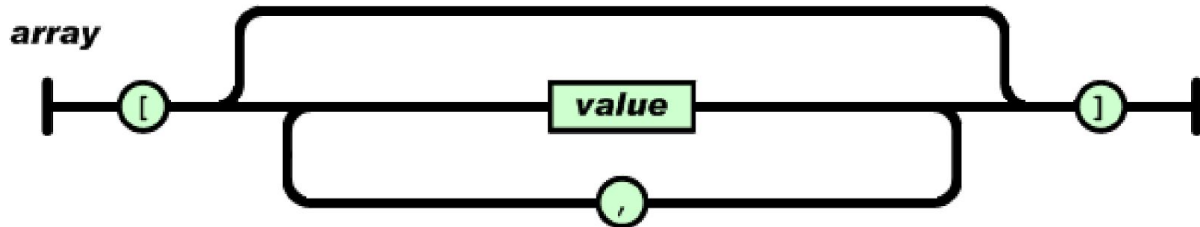
Αυτές είναι καθολικές δομές δεδομένων. Σχεδόν όλες οι σύγχρονες γλώσσες προγραμματισμού υποστηρίζουν τουλάχιστον μια από τις παραπάνω μορφές. Είναι λογικό μια μορφή δεδομένων που είναι ανταλλάξιμη σε γλώσσες προγραμματισμού να βασίζεται επίσης σε αυτές τις δομές. Στο JSON οι οντότητες για τις οποίες κρατάμε πληροφορίες παίρνουν αυτές τις μορφές:

- Ένα αντικείμενο είναι μια μη διατεταγμένη σειρά από ζεύγη ονόματος / τιμής. Ένα αντικείμενο ξεκινά με {(αριστερό άγκιστρο) και τελειώνει με} (δεξιό άγκιστρο). Κάθε όνομα ακολουθείται από: (άνω κάτω τελεία) και στην συνέχεια γράφεται η τιμή του. Τα ζεύγη ονόματος / τιμής χωρίζονται από, (κόμμα).



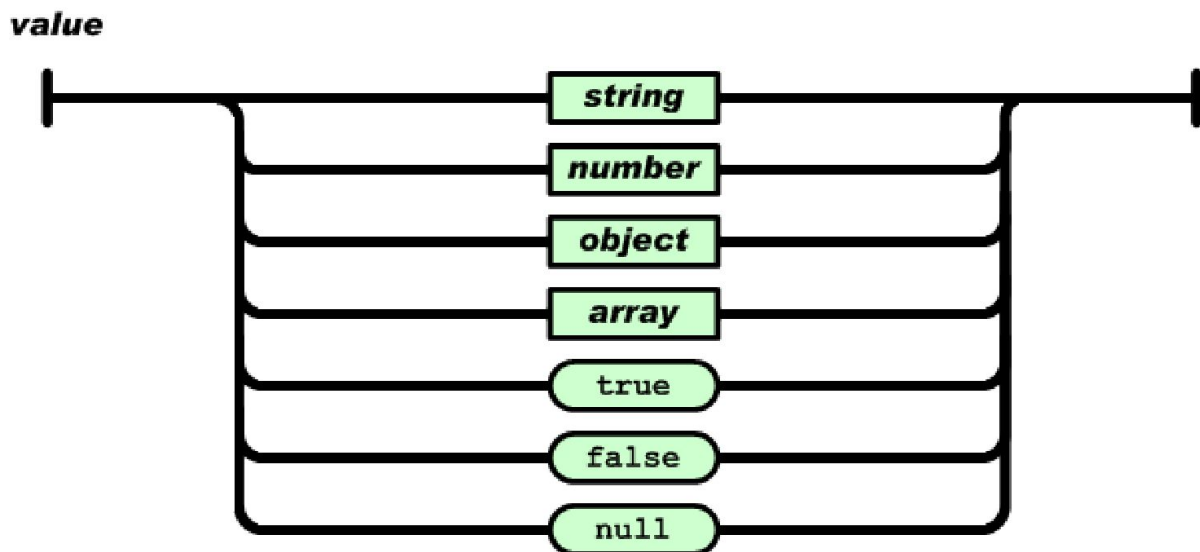
Εικόνα 2: Json string

- Ένας πίνακας είναι μια διατεταγμένη συλλογή τιμών. Μια σειρά αρχίζει με [(αριστερή αγκύλη) και τελειώνει με] (δεξιά αγκύλη). Οι τιμές χωρίζονται με, (κόμμα).



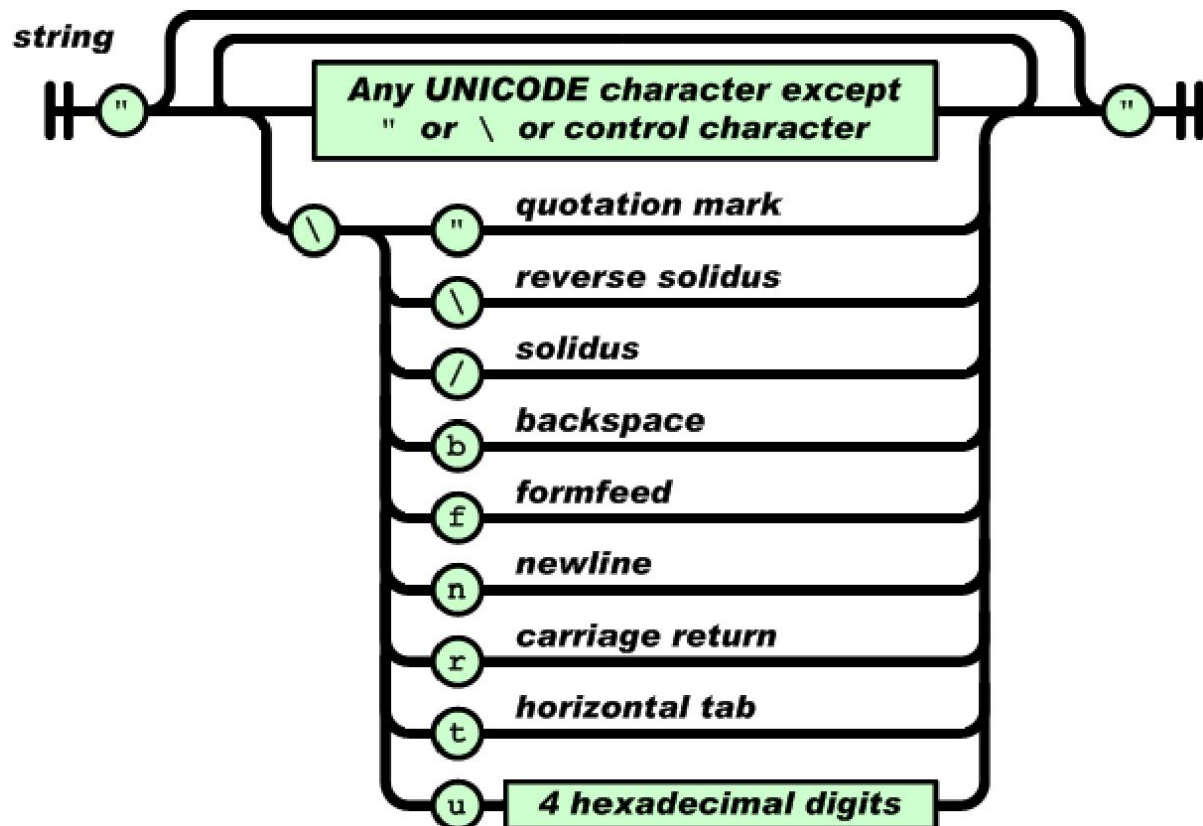
Εικόνα 3: Json array

- Η τιμή μπορεί να είναι ένα σύνολο χαρακτήρων (string) σε διπλά εισαγωγικά, ή ένας αριθμός, ή συνθήκη αλήθειας/ψεύδους, ή ένα αντικείμενο, ή μια ακολουθία, ή μηδενική(null). Αυτές οι δομές μπορεί να είναι εμφολευμένες (nested).



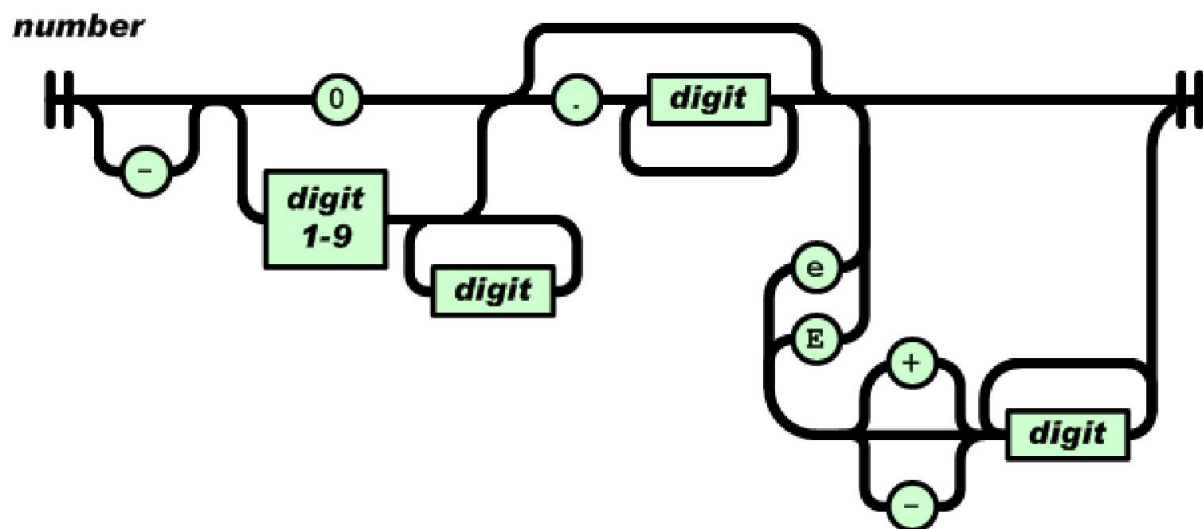
εικόνα 4: Json values

- Μια συμβολοσειρά(string) είναι μια ακολουθία από μηδέν ή περισσότερους χαρακτήρες Unicode, μέσα σε διπλά εισαγωγικά, χρησιμοποιώντας backslash για αποδράσεις. Ένας χαρακτήρας αντιπροσωπεύεται ως μια συμβολοσειρά με ένα χαρακτήρα(single character string). Μια συμβολοσειρά μοιάζει πολύ με μια C ή Java συμβολοσειρά.



εικόνα 5: Json escapes

- Ένας αριθμός μοιάζει πολύ με ένα C ή Java αριθμό, εκτός από το ότι δεν χρησιμοποιούνται οι οκταδικές και οι δεκαεξαδικές μορφές.



εικόνα 6: Json numbers

- Τα κενά (Whitespaces) μπορούν να εισαχθούν μεταξύ οποιουδήποτε ζεύγους αντικειμένων (token). Με εξαίρεση μερικών λεπτομερειών κωδικοποίησης (encoding), αυτό περιγράφει γενικότερα την γλώσσα (προγραμματισμού).

2.4 Σχεσιακή βάση δεδομένων(Relational database)

2.4.1 Γενικά

Με τον όρο **σχεσιακή βάση δεδομένων [8]** εννοείται μία συλλογή δεδομένων οργανωμένη σε συσχετισμένους πίνακες που παρέχει ταυτόχρονα δυνατότητα ανάγνωσης, εγγραφής, τροποποίησης ή και πιο πολύπλοκων διαδικασιών πάνω στα δεδομένα. Ο σκοπός μιας βάσης δεδομένων είναι η οργανωμένη αποθήκευση πληροφορίας και η δυνατότητα εξαγωγής της πληροφορίας αυτής, ιδίως σε πιο οργανωμένη μορφή, σύμφωνα με ερωτήματα που τίθενται στη σχεσιακή βάση δεδομένων. Τα δεδομένα είναι δυνατόν να αναδιοργανώνονται με πολλούς διαφορετικούς τρόπους, σε νοητούς πίνακες, χωρίς να είναι απαραίτητη η αναδιοργάνωση των φυσικών πινάκων που τα αποθηκεύουν. Οι ερωτήσεις, είτε από το χρήστη είτε από λογισμικό, προς τη βάση δεδομένων, γίνονται συνήθως μέσω της διαδεδομένης διαλογικής γλώσσας SQL(Structured Query Language). Εκτελώντας ερωτήματα ο χρήστης (ή το λογισμικό που εκπροσωπεί το χρήστη) είναι δυνατόν, ανάλογα με τα δικαιώματά του, να δημιουργήσει, να μεταβάλλει και να διαγράψει δεδομένα στη βάση, ή να ανασύρει πληροφορίες μέσω σύνθετων κριτηρίων αναζήτησης.

2.4.2 MYSQL

Η **MySQL[9]** είναι ένα σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων ανοιχτού κώδικα. Είναι δημοφιλής βάση δεδομένων για διαδικτυακά προγράμματα και ιστοσελίδες.

MySQL είναι ένα κεντρικό συστατικό στοιχείο του "LAMP": Στοιβα λογισμικού εφαρμογών web ανοιχτού κώδικα(και άλλων στοιβών "AMP"). LAMP είναι ένα αρκτικόλεξο για το "Linux, Apache, MySQL, Perl / PHP / Python". Εφαρμογές που χρησιμοποιούν τη βάση δεδομένων MySQL περιλαμβάνουν: TYPO3, MODx, Joomla, WordPress, phpBB, τη δύναμη, και Drupal. Επίσης η MySQL χρησιμοποιείται σε κάποιες από τις πιο διαδεδομένες διαδικτυακές υπηρεσίες, όπως το Flickr, το Youtube, το Wikipedia, το Google, το Facebook και το Twitter.

2.5 API

2.5.1 Γενικά

Η Διεπαφή Προγραμματισμού Εφαρμογών[10] (αγγλ. API, από το Application Programming Interface), γνωστή και ως Διασύνδεση Προγραμματισμού Εφαρμογών (για συντομία, *διεπαφή* ή *διασύνδεση*), είναι η διεπαφή των προγραμματιστικών διαδικασιών που παρέχει ένα λειτουργικό σύστημα, βιβλιοθήκη ή εφαρμογή, προκειμένου να επιτρέπει να γίνονται προς αυτά αιτήσεις από άλλα προγράμματα ή/και ανταλλαγή δεδομένων.

2.5.2 Σκοπός

Ακριβώς όπως μια γραφική διεπαφή χρήστη καθιστά ευκολότερο για τους ανθρώπους να χρησιμοποιούν προγράμματα, διεπαφές προγραμματισμού εφαρμογών (api) καθιστούν ευκολότερο για τους προγραμματιστές να χρησιμοποιούν ορισμένες τεχνολογίες σε προγραμματιστικές εφαρμογές. Αφαιρώντας την υποκείμενη υλοποίηση και εκθέτοντας μόνο τα αντικείμενα ή τις ενέργειες που χρειάζεται ο προγραμματιστής, ένα API μειώνει το γνωστικό φορτίο σε έναν προγραμματιστή.

2.5.3 Remote APIs

Τα Remote APIs επιτρέπουν στους προγραμματιστές να επεξεργαστούν απομακρυσμένους πόρους μέσω πρωτοκόλλων, ειδικές προδιαγραφές για την επικοινωνία που επιτρέπουν σε διαφορετικές τεχνολογίες να εργαστούν μαζί, ανεξάρτητα από τη γλώσσα ή την πλατφόρμα. Για παράδειγμα, το Java Database Connectivity API επιτρέπει στους προγραμματιστές να κάνουν ερωτήματα σε πολλούς διαφορετικούς τύπους βάσεων δεδομένων με το ίδιο σύνολο λειτουργιών, ενώ το Java remote method invocation API χρησιμοποιεί το Java Remote Method Protocol για να επιτρέψει την κλήση λειτουργιών εξ αποστάσεως, οι οποίες φαίνονται τοπικές στον προγραμματιστή. Ως εκ τούτου, τα remote APIs είναι χρήσιμα στη διατήρηση της αφαίρεσης αντικειμένων(object abstraction) στον αντικειμενοστραφή προγραμματισμό.

2.5.4 Web APIs

Web APIs είναι οι καθορισμένες διεπαφές μέσω των οποίων συμβαίνουν αλληλεπιδράσεις μεταξύ μιας επιχείρησης και εφαρμογών που χρησιμοποιούν λειτουργίες της. Μια προσέγγιση API είναι μια αρχιτεκτονική προσέγγιση που περιστρέφεται γύρω από την παροχή προγραμματιζόμενων διασυνδέσεων σε ένα σύνολο υπηρεσιών σε διαφορετικές εφαρμογές που εξυπηρετούν διαφορετικούς τύπους καταναλωτών.

Όταν χρησιμοποιείται στο πλαίσιο της ανάπτυξης ιστοσελίδων, ένα API συνήθως ορίζεται ως ένα σύνολο πρωτοκόλλων μεταφοράς υπερκειμένου (HTTP) μηνυμάτων αιτήσεων(request), καθώς επίσης και ένας ορισμός της δομής των μηνυμάτων απάντησης(response messages), των οποίων η μορφή είναι συνήθως σε Extensible Markup Language (XML) ή σε μορφή JSON. Όταν χρησιμοποιείται στο πλαίσιο της ανάπτυξης ιστοσελίδων, ένα API συνήθως ορίζεται ως ένα σύνολο πρωτοκόλλων μεταφοράς υπερκειμένου (HTTP) μηνυμάτων αιτήσεων(request), καθώς επίσης και ένας ορισμός της δομής των μηνυμάτων απάντησης(response messages), των οποίων η μορφή είναι συνήθως σε Extensible Markup Language (XML) ή σε μορφή JSON.

Ενώ τα "web API" ιστορικά είναι σχεδόν συνώνυμα με τα web services, η πρόσφατη τάση (το λεγόμενο Web 2.0) απομακρύνεται από τη χρήση Simple Object Access Protocol (SOAP) υπηρεσιών που βασίζονται στην web και service-oriented αρχιτεκτονική (SOA), προς υπηρεσίες που βασίζονται στο πρωτόκολλο REST(REpresentational State Transfer). Τα Web APIs επιτρέπουν το συνδυασμό πολλαπλών APIs σε νέες εφαρμογές γνωστές ως mashups. Στον χώρο των κοινωνικών μέσων(social media), τα web APIs έχουν κάνει εφικτό διαδικτυακές κοινότητες να μπορούν να στεγάσουν ανταλλαγές περιεχομένου και δεδομένων μεταξύ των κοινοτήτων και των

εφαρμογών. Με αυτόν τον τρόπο, το περιεχόμενο που δημιουργείται σε ένα μέρος μπορεί να αναρτηθεί δυναμικά και να ενημερώνονται σε οποιαδήποτε τοποθεσίας στον παγκόσμιο ιστό.

- Server side

Ένα server-side web API είναι μια προγραμματική διεπαφή αποτελούμενη από ένα ή περισσότερα δημόσια άκρα(public endpoints) σε ένα καθορισμένο σύστημα μηνυμάτων αίτησης-απάντησης, συνήθως εκφράζεται σε μορφή JSON ή XML, το οποίο εκτίθεται μέσω του web, συνήθως μέσω ενός HTTP-based web server. Mashups είναι web based εφαρμογές που συνδυάζουν τη χρήση πολλαπλών server-side web APIs.

- Client side

Ένα client-side web API είναι μια προγραμματική διεπαφή για να την επέκταση της λειτουργικότητας σε ένα πρόγραμμα περιήγησης web ή κάποιο άλλο HTTP client. Αρχικά εμφανίζονταν πιο συχνά με τη μορφή nativeplug-in επεκτάσεων του προγράμματος περιήγησης, ωστόσο κατά κύριο λόγο, τα νεότερα στοχεύουν τυποποιημένες συνδέσεις JavaScript.

2.5.5 Restful APIs

Στην πληροφορική, representational state transfer (REST) είναι μια αρχιτεκτονική που χρησιμοποιείται στο web development. Συστήματα και sites που είναι σχεδιασμένα να χρησιμοποιούν αυτή την αρχιτεκτονική στοχεύουν στην γρήγορη απόδοση, την αξιοπιστία και στη δυνατότητα να αναβαθμίσουν (να μπορούν να αναπτυχθούν και να υποστηρίξουν εύκολα επιπλέον χρήστες). Για την επίτευξη αυτών των στόχων, οι προγραμματιστές εργάζονται με επαναχρησιμοποιήσιμα συστατικά (components) που μπορούν να μεταβάλλονται και να ενημερώνονται χωρίς να επηρεάζεται το σύστημα στο σύνολό του, ενώ αυτό βρίσκεται σε λειτουργία.

Όταν συστήματα συμμορφώνονται με τους περιορισμούς της REST μπορούν να ονομάζονται RESTful. RESTful συστήματα συνήθως, αλλά όχι πάντα, επικοινωνούν μέσω Hypertext Transfer Protocol (HTTP) με ίδιες διαδικασίες HTTP (GET, POST, PUT, DELETE, κλπ) που τα προγράμματα περιήγησης χρησιμοποιούν για να ανακτήσουν ιστοσελίδες και να στείλουν δεδομένα σε απομακρυσμένους διακομιστές.

Συστήματα διεπαφής REST με εξωτερικά συστήματα δικτυακών πόρων προσδιορίζονται από Uniform Resource Identifiers (URIs), για παράδειγμα /people/tom, το οποίο μπορεί να λειτουργήσει κατά τη χρήση τυποποιημένων διαδικασιών όπως GET /people/tom

Τα APIs υπηρεσιών Web(Web service APIs) που συμμορφώνονται με την REST αρχιτεκτονική περιορισμών ονομάζονται RESTful APIs. Τα HTTP-based RESTful APIs ορίζονται με τα ακόλουθα χαρακτηριστικά:

- base URL, πχ <http://example.com/resources/>
- Ένα τύπο διαδικτυακού μέσου που καθορίζει τα στοιχεία δεδομένων μεταβατικής κατάστασης (π.χ., Atom, microformats, application/vnd.collection+json κ.α.) Η τρέχουσα εκπροσώπηση(current representation) λέει στον πελάτη(client) πώς να συνθέσει όλες τις μεταβάσεις στην επόμενη κατάσταση της εφαρμογής. Αυτό θα μπορούσε να είναι τόσο

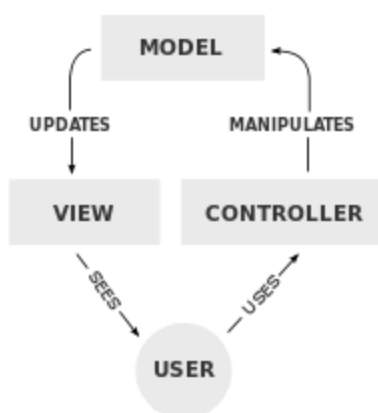
απλό όσο μια διεύθυνση URL ή τόσο περίπλοκο όπως μια βοηθητική εφαρμογή Java(java applet)

- Πρότυπες μεθόδους HTTP (π.χ., OPTIONS, GET, PUT, POST, και DELETE)

2.6 MVC Framework

Model-View-Controller (MVC)[11] είναι ένα λογισμικό αρχιτεκτονικό πρότυπο για την δημιουργία διεπαφών χρήστη σε υπολογιστές. Χωρίζει μια εφαρμογή λογισμικού σε τρία αλληλοσυνδεδεμένα μέρη, έτσι ώστε να διαχωριστούν οι εσωτερικές αναπαραστάσεις πληροφοριών από τους τρόπους που οι πληροφορίες παρουσιάζονται ή γίνονται αποδεκτές από το χρήστη.

Παραδοσιακά χρησιμοποιείται για επιφάνειες εργασίας γραφικών διεπαφών χρήστη (GUIs), αυτή η αρχιτεκτονική έχει γίνει δημοφιλής για το σχεδιασμό εφαρμογών web.



εικόνα 7: MVC

Δομικά στοιχεία

Το κεντρικό στοιχείο του MVC, το model, συλλαμβάνει τη συμπεριφορά της εφαρμογής από την άποψη του τομέα του προβλημάτων της, ανεξάρτητα από το περιβάλλον εργασίας χρήστη.

Το model διαχειρίζεται άμεσα τα δεδομένα, τη λογική και τους κανόνες της εφαρμογής.

Ένα view μπορεί να είναι οποιαδήποτε αναπαράσταση εξόδου των πληροφοριών, όπως ένα γράφημα ή διάγραμμα. Πολλαπλά views ίδιων πληροφοριών είναι δυνατά, όπως ένα ιστόγραμμα για τη διαχείριση και ένα view πινάκων για τους λογιστές.

Το τρίτο μέρος, ο controller, δέχεται είσοδο και το μετατρέπει σε εντολές για το model ή το view.

Αλληλεπιδράσεις

Εκτός από τη διαίρεση της εφαρμογής σε τρία είδη των στοιχείων, ο σχεδιασμός model-view-controller καθορίζει τις αλληλεπιδράσεις μεταξύ τους.

Ένα model αποθηκεύει δεδομένα που ανακτώνται σύμφωνα με εντολές από τον controller και εμφανίζονται στο view.

Ένα view δημιουργεί νέα έξοδο προς το χρήστη με βάση τις αλλαγές που έγιναν στο model.

Ένας controller μπορεί να στείλει εντολές στο model για την ενημέρωση της κατάστασης του. (πχ. επεξεργασία ενός εγγράφου). Μπορεί επίσης να στείλει εντολές σε ένα view του για να αλλάξει την παρουσίαση του των δεδομένων του model που προβάλλει το view(π.χ. με κύλιση σε ένα έγγραφο).

Χρήση σε εφαρμογές web

Αν και αρχικά αναπτύχθηκε για προγραμματισμό desktop, το model-View-Controller έχει υιοθετηθεί ευρέως ως μια αρχιτεκτονική για World Wide Web εφαρμογές σε μεγάλες γλώσσες προγραμματισμού. Αρκετά εμπορικά και μη web frameworks έχουν δημιουργηθεί που επιβάλλουν αυτό το μοτίβο. Αυτά τα frameworks διαφέρουν ως προς τις ερμηνείες τους, κυρίως στον τρόπο με τον οποίο οι MVC ευθύνες κατανέμονται ανάμεσα στον πελάτη(client) και στον διακομιστή(server).

Πρόωρα web MVC frameworks πήραν μια λεπτή προσέγγιση προς τον client που τοποθέτησε σχεδόν εξ ολοκλήρου το model, το view και τον controller στο server. Αυτό εξακολουθεί να αντανακλάται σε δημοφιλή πλαίσια όπως Ruby on Rails, Django, ASP.NET MVC. Στην προσέγγιση αυτή, ο client στέλνει είτε hyperlink αιτήματα ή είσοδο φόρμας(form) στον controller και στη συνέχεια λαμβάνει μια πλήρη και ενημερωμένη ιστοσελίδα (ή άλλο έγγραφο) από την view: το model υπάρχει εξ ολοκλήρου στο διακομιστή. Καθώς οι τεχνολογίες πελάτη έχουν ωριμάσει, frameworks, όπως AngularJS, EmberJS, JavaScriptMVC και Backbone έχουν δημιουργηθεί που επιτρέπουν τα συστατικά MVC να εκτελεστούν εν μέρει από τον client(όπως Ajax).

2.7 SDK

Ένα software development kit (SDK ή "devkit") είναι συνήθως ένα σύνολο εργαλείων ανάπτυξης λογισμικού που επιτρέπει τη δημιουργία εφαρμογών για ένα συγκεκριμένο πακέτο λογισμικού, framework λογισμικού, hardware πλατφόρμα, το σύστημα του υπολογιστή, κονσόλα βίντεο παιχνιδιών, λειτουργικό σύστημα, ή παρόμοια πλατφόρμα ανάπτυξης. Για τη δημιουργία εφαρμογών, θα πρέπει να κατεβάσετε ένα ειδικό SDK. Για παράδειγμα, η ανάπτυξη μιας εφαρμογής Android απαιτεί SDK με Java, για iOS apps ένα SDK iOS με Swift, και για το MS

Windows το .NET SDK Framework με .NET. Υπάρχουν, επίσης, τα SDK που είναι εγκατεστημένα σε εφαρμογές για την παροχή στατιστικών και δεδομένων σχετικών με τη δραστηριότητα. Χαρακτηριστικά παραδείγματα περιλαμβάνουν το Google και το Facebook.

Μπορεί να είναι κάτι τόσο απλό όσο η δημιουργία ενός ή περισσότερων διεπαφών προγραμματισμού εφαρμογών (API) με τη μορφή κάποιων βιβλιοθηκών για τη διασύνδεση σε μια συγκεκριμένη γλώσσα προγραμματισμού ή να περιλαμβάνουν εξελιγμένο υλικό που μπορεί να επικοινωνεί με κάποιο άλλο ενσωματωμένο σύστημα. Κοινά εργαλεία περιλαμβάνουν εγκαταστάσεις εντοπισμού σφαλμάτων και άλλα βοηθητικά προγράμματα, που παρουσιάζονται συχνά σε ένα ολοκληρωμένο περιβάλλον ανάπτυξης (IDE). Τα SDK περιλαμβάνουν επίσης συχνά δείγμα κώδικα και την υποστηρικτικές τεχνικές σημειώσεις ή άλλο βοηθητικό υλικό για να συμβάλει στην αποσαφήνιση σημείων από το πρωτογενές υλικό αναφοράς.

2.8 Ιδεατές μηχανές (Virtual Machines)

Μια ιδεατή μηχανή είναι ένα λογισμικό υπολογιστή που, όπως ένας φυσικός υπολογιστής, τρέχει ένα λειτουργικό σύστημα και εφαρμογές. Η ιδεατή μηχανή αποτελείται από ένα σύνολο αρχείων συστήματος που υποστηρίζονται από τους φυσικούς πόρους του περιβάλλοντος υπολογιστή (host). Κάθε ιδεατή μηχανή έχει ιδεατές συσκευές που παρέχουν την ίδια λειτουργικότητα με αυτή που θα είχε μια φυσική συσκευή και έχουν πρόσθετα οφέλη από την άποψη της φορτικότητας, διαχειρισιμότητας και ασφάλειας.

Μια ιδεατή μηχανή αποτελείται από διάφορους τύπους αρχείων που αποθηκεύονται σε μια συσκευή αποθήκευσης(storage device) που μπορεί να τα υποστηρίξει. Τα βασικά αρχεία που απαρτίζουν μια εικονική μηχανή είναι το αρχείο ρυθμίσεων, το ιδεατό αρχείο δίσκου, το αρχείο ρύθμισης NVRAM, και το αρχείο καταγραφής(log).

Κεφάλαιο 3

ΑΠΑΙΤΗΣΕΙΣ

Η σχεδίαση της εφαρμογής βασίστηκε πάνω σε μια αρχική καταγραφή των λειτουργικών και μη λειτουργικών απαιτήσεων από αυτή. Καθώς δεν υπήρξε ομάδα πραγματικών χρηστών, κατά τη σχεδίαση της εφαρμογής, αξιοποιήθηκαν προσωπικές εμπειρίες και ιδέες των αρχικών εμπνευστών και εμού μέσα από την εμπειρία που είχαμε στη σχεδίαση εταιρικών site.

3.1 Λειτουργικές Απαιτήσεις

3.1.1 Εγγραφή χρηστών

Η εφαρμογή δίνει τη δυνατότητα στους χρήστες, να εγγράφονται και να δημιουργούν το εταιρικό τους προφίλ. Πιο συγκεκριμένα, στην εφαρμογή των παραγωγών ζητούνται επιπλέον στοιχεία απ' ότι στην εφαρμογή των πελατών τα οποία θα βεβαιώνουν ότι πληρούν τις απαιτούμενες προϋποθέσεις για τη νόμιμη άσκηση του επαγγέλματος. Επιπλέον, το σύστημα ελέγχει την ορθότητα των δεδομένων που εισάγουν οι προς εγγραφή χρήστες και να τους ενημερώνει για τυχόν λάθη ή παραλείψεις.

3.1.2 Προβολή και ενημέρωση του προφίλ των χρηστών

Οι χρήστες της εφαρμογής έχουν ένα δημόσιο προφίλ, το οποίο περιέχει βασικές πληροφορίες για αυτούς, ούτως ώστε να είναι εύκολα αναγνωρίσιμοι από τους υπόλοιπους χρήστες. Επιπλέον οι παραγωγοί έχουν την δυνατότητα δημιουργίας του προφίλ της εταιρίας κάτι αναγκαίο ώστε να μπορούν να καταχωρίσουν τα προϊόντα τους στην εφαρμογή. Τέλος, όλοι οι χρήστες έχουν την δυνατότητα ενημέρωσης των στοιχείων του προφίλ τους.

3.1.3 Δημιουργία, προβολή και ενημέρωση προφίλ εταιρίας παραγωγού

Οι παραγωγοί έχουν την δυνατότητα δημιουργίας και τροποποίησης του προφίλ της εταιρίας τους κάτι αναγκαίο ώστε να μπορούν να καταχωρίσουν τα προϊόντα τους στην εφαρμογή.

3.1.4 Καταχώρηση προϊόντων των παραγωγών

Οι εγγεγραμμένοι παραγωγοί από την στιγμή που έχουν δημιουργήσει το προφίλ της εταιρίας τους μπορούν να καταχωρούν στο προφίλ τους τα είδη και προϊόντα που διαθέτουν προς πώληση. Για να το πετύχουν αυτό, κάθε προϊόν που θα εισάγεται, να περιέχει τουλάχιστον μία φωτογραφία του προϊόντος, πληροφορίες για τον τόπο και τρόπο παραγωγής του, την τιμή του, καθώς και την διαθέσιμη ποσότητα.

3.1.5 Καταχώρηση προσφορών από τους παραγωγούς

Κατά την είσοδό τους στην εφαρμογή οι παραγωγοί μπορούν να δουν στην αρχική τους σελίδα τις ενεργές αγγελίες που έχουν καταχωρήσει οι πελάτες σε μία λίστα. Συγκεκριμένα, ο για να είναι

ορατή μία αγγελία από έναν παραγωγό, θα πρέπει τουλάχιστον ένα από τα προϊόντα της αγγελίας να περιέχεται στα είδη-προϊόντα τα οποία διαθέτει. Με αυτό τον τρόπο θα εξασφαλίζεται ότι μπορεί σε κάθε περίπτωση να εξυπηρετήσει την εκάστοτε αγγελία. Επιπλέον ο παραγωγός έχει την δυνατότητα αναζήτησης και προβολής αγγελιών με βάση την κατηγορία του προϊόντος ζήτησης.

3.1.6 Ιστορικό προσφορών

Ο παραγωγός έχει την δυνατότητα προβολής του ιστορικού προσφορών του, στο οποίο μπορεί να δει στοιχεία για όλες τις προσφορές που έχει κάνει (πχ σε ποια δημοπρασία αναφέρεται η προσφορά, αν είναι νικητήρια κλπ)

3.1.7 Άμεση αγορά

Δύνεται η δυνατότητα σε ένα πελάτη να δημιουργήσει αίτηση αγοράς για κάποιο συγκεκριμένο προϊόν άμεσα (παρακάμπτοντας την διαδικασία της δημοπρασίας). Με την σειρά του ο παραγωγός/ιδιοκτήτης του προαναφερθέντος προϊόντος έχει την δυνατότητα αποδοχής ή απόρριψης των αιτήσεων άμεσης αγοράς που λαμβάνει.

3.1.8 Συνομιλία ανάμεσα σε πελάτη και παραγωγό

Σε περίπτωση που λήξει μια δημοπρασία και υπάρχει νικητής, ή σε περίπτωση που υπάρχει αποδοχή άμεσης αγοράς, τότε δημιουργείται η δυνατότητα συνομιλίας ανάμεσα στον πελάτη (δημιουργό της δημοπρασίας ή της αίτησης άμεσης αγοράς) και στον παραγωγό αντίστοιχο παραγωγό, για την τακτοποίηση τυχών λεπτομερειών συναλλαγής (πχ ημερομηνία, τοποθεσία).

3.2 Μη λειτουργικές απαιτήσεις

3.2.1 Απόδοση εφαρμογής

Ο χρόνος απόκρισης της εφαρμογής δεν θα πρέπει να υπερβαίνει τα 4 δεύτερα. Επίσης, η εφαρμογή θα πρέπει να λειτουργεί αδιάλειπτα και να διασφαλίζει ότι ακόμα και σε περίπτωση κάποια απρόσμενης διακοπής λειτουργίας, θα διαθέτει μηχανισμούς ανάκαμψης που θα της επιτρέπουν να επανέλθει στην προηγούμενη λειτουργική της κατάσταση. Τέλος, η εφαρμογή θα πρέπει να διαθέτει μηχανισμούς που θα εγγυώνται την συνέπεια και ακεραιότητα των δεδομένων της.

3.2.2 Χρηστικότητα εφαρμογής

Καθώς η εφαρμογή απευθύνεται σε ευρύ κοινό χρηστών, θα πρέπει να περιέχει γραφικά συστατικά των οποίων η λειτουργία να είναι εύκολα αντιληπτή και ξεκάθαρη. Επιπλέον, τα γραφικά συστατικά που θα δημιουργηθούν για τις ανάγκες της εφαρμογής θα πρέπει να συνοδεύονται από χρωματικούς συνδυασμούς που δεν θα αποπροσανατολίζουν τους χρήστες.

3.2.3 Ασφάλεια και ιδιωτικότητα των δεδομένων της εφαρμογής

Με γνώμονα την συμμόρφωση με όλους τους κανόνες προστασίας των προσωπικών δεδομένων και διασφάλισης του προσωπικού απορρήτου, η εφαρμογή θα πρέπει να εξασφαλίζει ότι κάθε χρήστης έχει τον αποκλειστικό έλεγχο των προσωπικών του δεδομένων. Επίσης, η εφαρμογή με τη χρήση μηχανισμών ασφαλείας θα πρέπει να αποτρέπει την μη εξουσιοδοτημένη προβολή των ιδιωτικών δεδομένων και να διασφαλίζει την ακεραιότητα των ευαίσθητων δεδομένων. Τέλος, η εφαρμογή θα πρέπει να διατίθεται με άδεια που θα καθορίζει ρητά ότι τα δεδομένα της εφαρμογής δεν μπορούν να χρησιμοποιηθούν για άλλους σκοπούς (πχ., διαφήμιση).

3.3 Αρχιτεκτονική συστήματος

Η παρούσα εργασία αποτελείται από δύο μέρη, το front-end (υβριδική εφαρμογή στο κινητό του χρήστη) και το back-end (εφαρμογή διακομιστή και βάση δεδομένων). Τα δύο αυτά μέρη βρίσκονται σε διαρκή επικοινωνία μεταξύ τους ανταλλάσσοντας πληροφορίες σε μορφή **JSON**.

3.3.1 Front-End

Στο front-end έχουμε την υβριδική εφαρμογή για smart-phones ανεπτυγμένη με το codeigniter framework, η οποία και τρέχει στην συσκευή του χρήστη. Είναι κυρίως υπεύθυνο για την προβολή και ανανέωση δεδομένων του χρήστη (πχ προβολή δημοπρασιών σε πραγματικό χρόνο). Για την ανάπτυξή του χρησιμοποιήθηκαν τα frameworks το ionic και cordova τα οποία αναφέρονται στο επόμενο κεφάλαιο.

3.3.2 Back-End

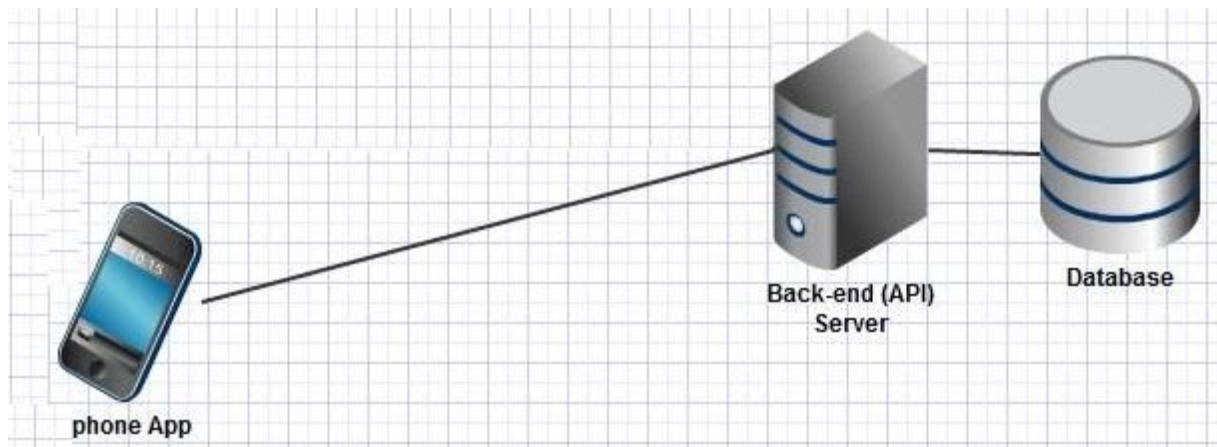
Για το Back-End χρησιμοποιήθηκε το Oracle Virtual Box μέσω της οποίας στήθηκε τοπικά ένας ιδεατός διακομιστής (virtual server). Πιο συγκεκριμένα, χρησιμοποιείται Ubuntu server 14.04.4 LTS.

Σε αυτόν εγκαταστάθηκαν τα εξής:

- apache 2.4.7
- php 5.5.9
- mysql 14.14
- codeigniter 3.0.0

Το codeigniter framework χρησιμοποιήθηκε για την ανάπτυξη της backend εφαρμογής, η οποία επικοινωνεί με το front-end (εφαρμογή πελάτη) και είναι υπεύθυνη για τη συλλογή και αποθήκευση δεδομένων στην βάση.

Η βάση δεδομένων που χρησιμοποιήθηκε υπάρχει στην διεύθυνση gianta.xyz στο port 3306 και σκοπός της είναι η αποθήκευση και ανάκτηση δεδομένων.



εικόνα 8: επικοινωνία front-end/back-end

Όπως φαίνεται και στην εικόνα, το front-end (phone Application) επικοινωνεί με τον server μέσω Internet ο οποίος με την σειρά του, εκτελεί τις εντολές που χρειάζονται ώστε να γίνουν οι κατάλληλες ενέργειες(όπως ανάκτηση ή τροποποίηση δεδομένων) στην βάση δεδομένων(database).

Κεφάλαιο 4

Υλοποίηση

Στο παρών κεφάλαιο αναλύονται όλες οι τεχνολογίες που χρειάστηκαν για την ανάπτυξη της εφαρμογής, τόσο στο front-end όσο και στο back-end.

4.1 Εργαλεία ανάπτυξης και διαχείρισης λογισμικού

4.1.1 Oracle Virtual Box

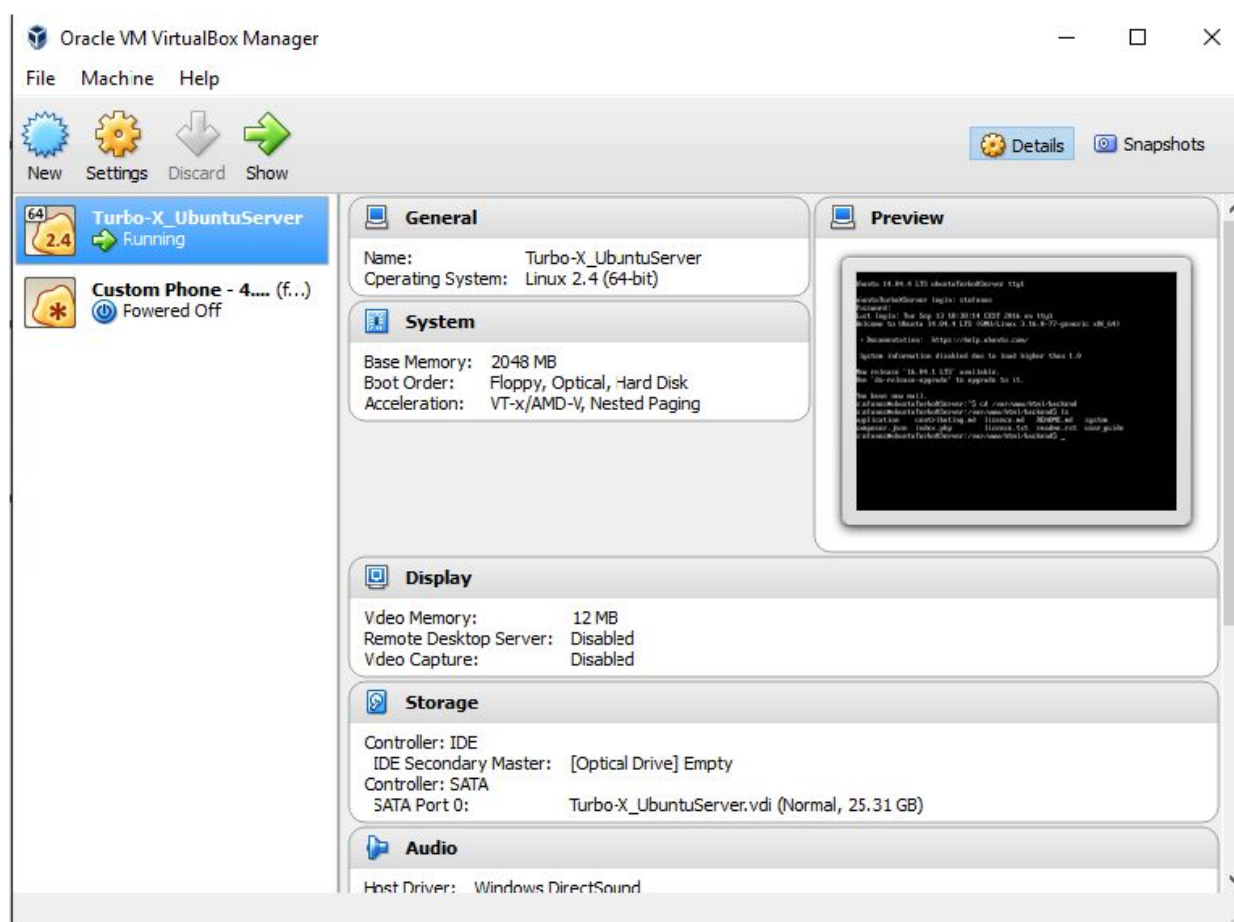
Κατά την περίοδο ανάπτυξης της παρούσας εργασίας παρουσιάστηκε δυσκολία στην εύρεση χώρου σε server για την “φιλοξενία” του back-end μέρους της πτυχιακής. Για εμένα την λύση σε αυτό το πρόβλημα έδωσε το Oracle Virtual Box. Μέσω του οποίου έστησα ένα ιδεατό server τοπικά στον προσωπικό μου υπολογιστή.



VirtualBox[14] είναι ένα ισχυρό x86 και AMD64 / Intel64 προϊόν δημιουργίας ιδεατών μηχανών για επιχειρήσεις, καθώς και για απλούς χρήστες. Το VirtualBox όχι μόνο είναι ένα προϊόν υψηλής απόδοσης με εξαιρετικά πλούσια χαρακτηριστικά για πελάτες επιχειρήσεων, είναι επίσης η μόνη επαγγελματική λύση που είναι ελεύθερα διαθέσιμη ως Λογισμικό Ανοικτού Κώδικα σύμφωνα με τους όρους της GNU General Public License (GPL) έκδοση 2.

Επί του παρόντος, το VirtualBox τρέχει σε Windows, Linux, Macintosh, και Solaris και υποστηρίζει ένα μεγάλο αριθμό λειτουργικών συστημάτων guest που περιλαμβάνουν αλλά δεν περιορίζονται σε Windows (NT 4.0, 2000, XP, Server 2003, Vista, Windows 7, Windows 8, τα Windows 10), DOS / Windows 3.x, Linux (2.4, 2.6, 3.x και 4.x), Solaris και OpenSolaris, OS / 2, και το OpenBSD.

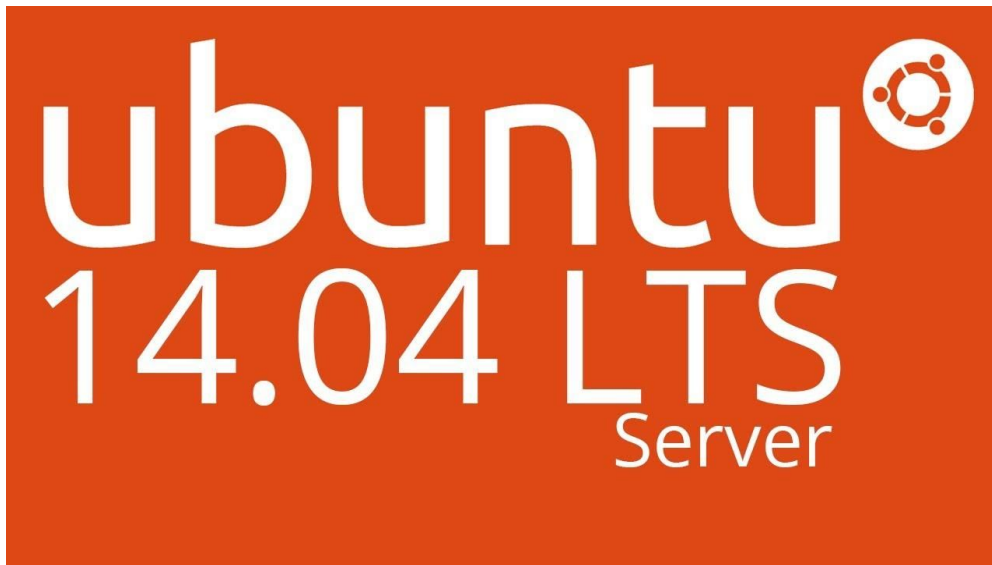
Το VirtualBox αναπτύσσεται ενεργά με συχνές κυκλοφορίες και έχει μια συνεχώς αυξανόμενη λίστα χαρακτηριστικών, υποστηριζόμενων λειτουργικών συστημάτων και πλατφορμών.



εικόνα 9: παράδειγμα χρήσης virtualBox

4.1.2 Ubuntu Server

Το περιβάλλον διακομιστή (server) που επέλεξα να εγκαταστήσω στο VirtualBox ήταν το Ubuntu Server 14.04.4 LTS[15]



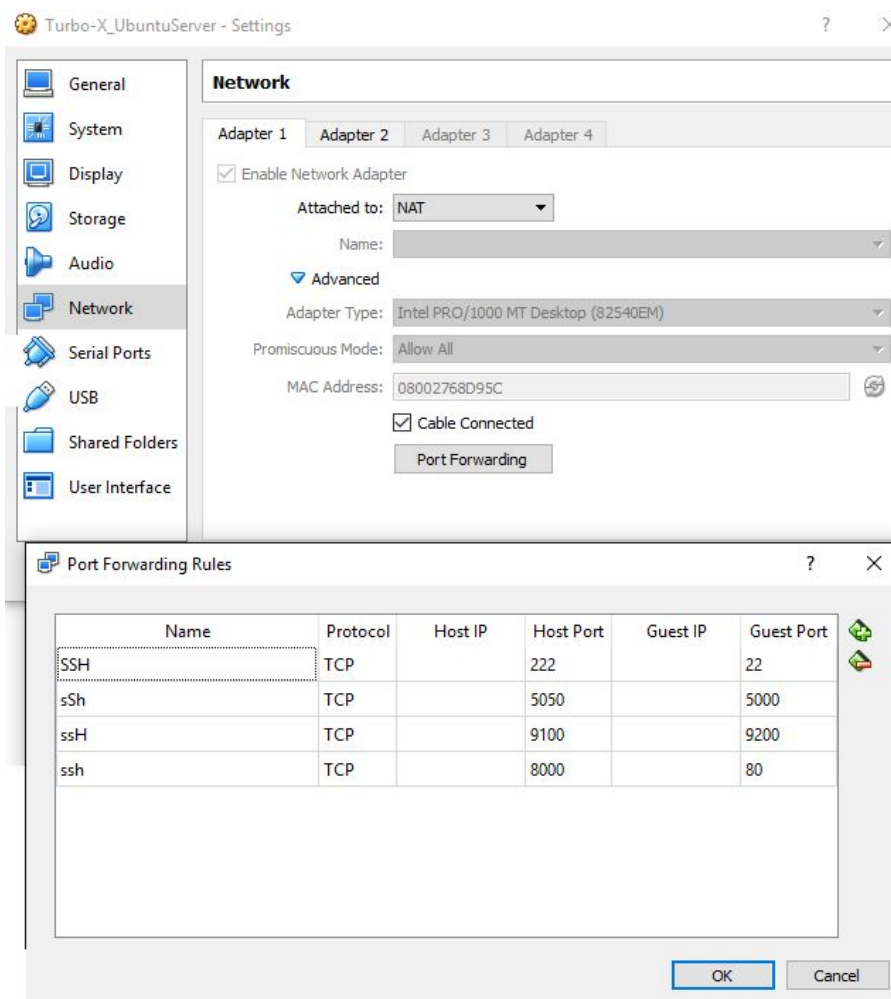
εικόνα 10: ubuntu 14.04 LTS Server

Σε αυτό το ιδεατό μηχανήμα, μπορώ να έχω πρόσβαση μόνο αν το έχω ενεργοποιήσει μέσω του Virtual Box, και λειτουργεί με τον ίδιο ακριβώς τρόπο που θα λειτουργούσε ένα οποιοδήποτε μηχανήμα με περιβάλλον Ubuntu, χωρίς γραφικό περιβάλλον χρήστη.

The image is a screenshot of a VirtualBox window titled "Turbo-X_UbuntuServer [Running] - Oracle VM VirtualBox". The window has a menu bar with "File", "Machine", "View", "Input", "Devices", and "Help". The main area is a terminal window with a black background and white text. The terminal shows a user named "stefanos" at a prompt "stefanos@ubuntuTurboXServer:~\$". The user enters "cd ~/var/www/html/backend", "ls", and "who". The output of "ls" shows a directory listing: "application", "contributing.md", "licence.md", "README.md", "system", "composer.json", "index.php", "license.txt", "readme.rst", and "user_guide". The terminal window has a standard Linux-style window control bar at the bottom with minimize, maximize, and close buttons. The bottom of the screenshot shows the VirtualBox toolbar with various icons and a "Right Control" button.

εικόνα 11: virtual server

Το μηχάνημα αυτό είναι επίσης προσβάσιμο από τον προσωπικό μου υπολογιστή μέσω της τοποθεσίας : <http://localhost> στο port 8000 ή μέσω ssh στο port 222. Αυτό επιτεύχθηκε μέσω κάποιων τροποποιήσεων στους κανόνες προσβασιμότητας του VirtualBox.



εικόνα 12: server networking

4.1.3 ExpanDrive

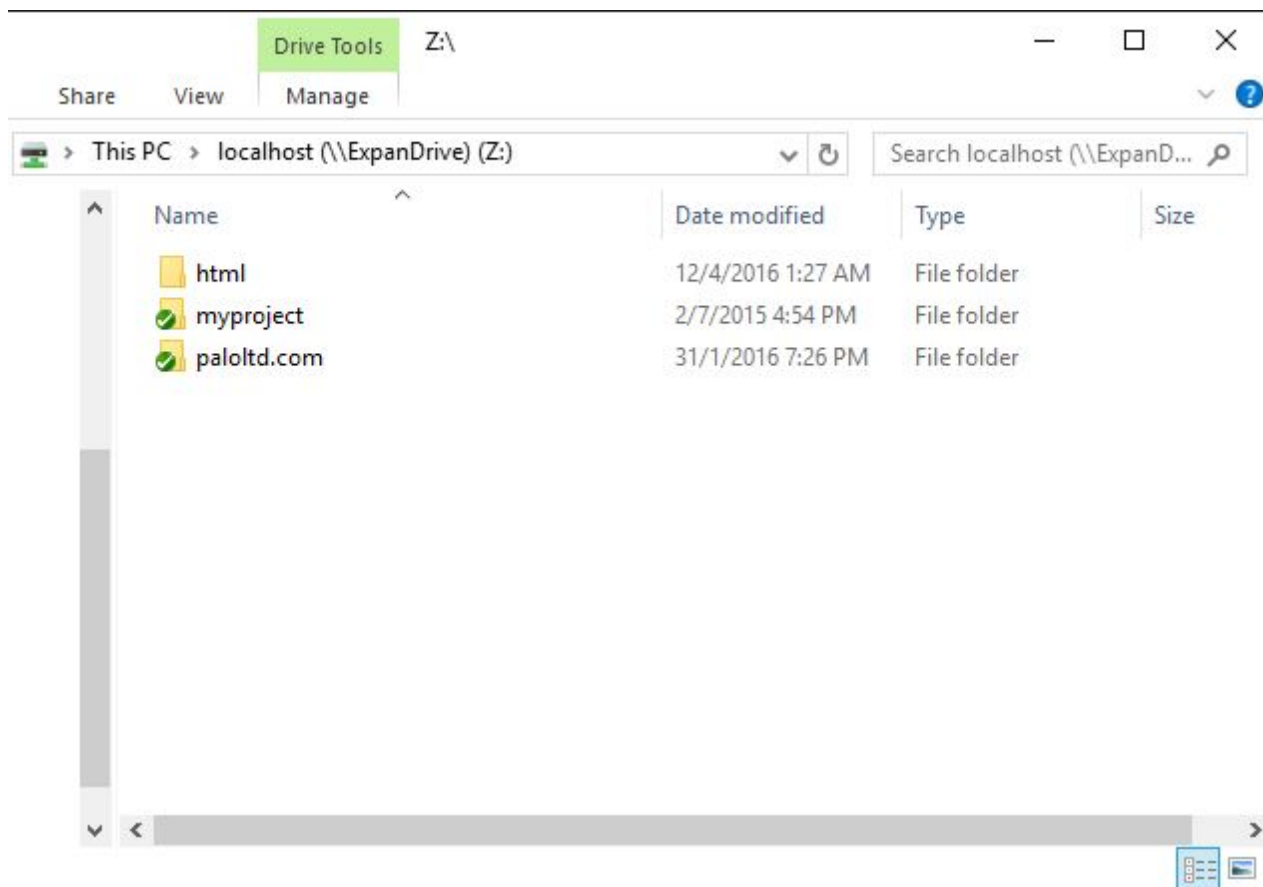
Επειδή χωρίς γραφικό περιβάλλον χρήστη ήταν δύσκολο να δημιουργήσω και να τροποποιήσω αρχεία, χρησιμοποίησα το ExpanDrive[16], ώστε να μπορώ να διαχειριστώ τα αρχεία της ιδεατής μηχανής σαν να ήταν αρχεία του φυσικού μου συστήματος. Αυτό το πρόγραμμα χρησιμοποιείται επίσης και για προβολή και τροποποίηση αρχείων εξ' αποστάσεως.



ExpanDrive είναι ένα client καταμεμημένο σύστημα αρχείων για Mac OS X και Microsoft Windows που διευκολύνει τη χαρτογράφηση του τοπικών φακέλων σε πολλούς διαφορετικούς τύπους Cloud Storage. Όταν σε ένα διακομιστή έχει χρησιμοποιηθεί το ExpanDrive τότε οποιοδήποτε πρόγραμμα μπορεί να διαβάσει, να γράψει και να διαχειριστεί απομακρυσμένα αρχεία (δηλαδή, τα αρχεία που υπάρχουν μόνο στο server) σαν να ήταν αποθηκευμένα τοπικά. Είναι διαφορετικό από τα περισσότερα προγράμματα μεταφοράς αρχείων, επειδή η ενσωμάτωση γίνεται σε όλες τις εφαρμογές του λειτουργικού συστήματος.

The screenshot shows the ExpanDrive application window. On the left is a sidebar with icons for "New Drive", "Drives", "Settings", and "Feedback". The "Drives" section is active, showing a list of drives: "localhost" (localhost - /var/www) and "10.100.51.100" (10.100.51.100). The main area contains configuration fields for a drive. The "drive type" is set to "SSH (SFTP)". The "server" is "localhost", "username" is "stefanos", and "password" is masked with asterisks. There is a checkbox for "save password" and a link for "fewer options". The "authentication" is set to "Password", "port" is "222", "nickname" is "localhost", and "remote path" is "/var/www". The "drive letter" is set to "Z:", and there is an unchecked checkbox for "reconnect at login". At the bottom right are "Delete" and "Connect" buttons.

εικόνα 13: expandrive variables



εικόνα 14: παράδειγμα χρήσης expandrive

Με τις ρυθμίσεις που έχω κάνει στο ExpanDrive, όταν συνδέομαι στον server, δημιουργείται ο φάκελος (Z:) στο φυσικό μου μηχάνημα, μέσω του οποίου έχω πρόσβαση σε όλους τους φακέλους και υποφακέλους που βρίσκονται στο server στο μονοπάτι /var/www.

4.1.4 MySql WorkBench



Το πρόγραμμα MySQL Workbench[17] χρησιμοποιήθηκε για την δημιουργία και συντήρηση της βάσης δεδομένων που χρησιμοποιήθηκε στην εργασία.

Είναι ένα εργαλείο σχεδιασμού βάσεων δεδομένων που ενσωματώνει ανάπτυξη SQL, διοίκηση, σχεδιασμό βάσεων δεδομένων, δημιουργία και συντήρηση σε ένα ενιαίο ολοκληρωμένο περιβάλλον ανάπτυξης για το σύστημα της βάσης δεδομένων MySQL. Είναι ο διάδοχος DBDesigner 4 από fabFORCE.net, και αντικαθιστά το προηγούμενο πακέτο λογισμικού, MySQL GUI Tools Bundle.

4.1.4 Brackets



Για την ανάπτυξη κώδικα, τόσο στο front-end, όσο και στο back-end, χρησιμοποίησα το Brackets[18] με το οποίο είμαι πολύ οικείος. Είναι ένα μοντέρνο εργαλείο ανάπτυξης κώδικα κυρίως για web-based εφαρμογές το οποίο μέσω της οργάνωσης αρχείων που παρέχει με βοήθησε αρκετά στην παράλληλη ανάπτυξη του front-end μέρους της εργασίας με αυτό του back-end.



εικόνα 15: παράδειγμα χρήσης brackets

4.2 Frameworks

4.2.1 Codeigniter

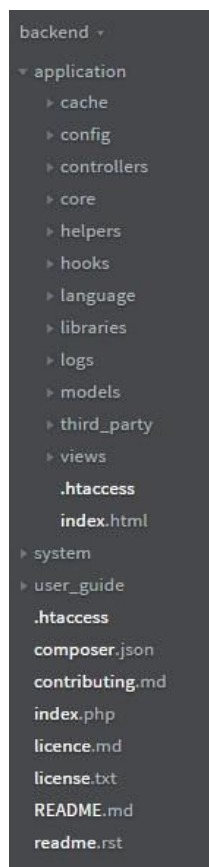
Το CodeIgniter[19] είναι ένα ισχυρό PHP framework, που χρησιμοποιεί την αρχιτεκτονική MVC

και δημιουργήθηκε για προγραμματιστές που χρειάζονται ένα απλό και κομψό πακέτο εργαλείων για την δημιουργία εφαρμογών web πολλών δυνατοτήτων.



Το CodeIgniter είναι ένα framework για Ανάπτυξη Εφαρμογών - ένα σύνολο εργαλείων - για προγραμματιστές που χτίζουν ιστοσελίδες χρησιμοποιώντας PHP. Στόχος του είναι να δώσει τη δυνατότητα να αναπτυχθούν projects πολύ πιο γρήγορα από ότι θα αναπτύσσονταν αν γραφόταν ο κώδικας από το μηδέν, παρέχοντας ένα πλούσιο σύνολο βιβλιοθηκών για συχνά απαιτούμενες εργασίες, καθώς και ένα απλό interface και λογική δομή για την πρόσβαση σε αυτές τις βιβλιοθήκες.

Παρακάτω η δομή των αρχείων του Codeigniter.



εικόνα 16: δομή αρχείων codeigniter

Όπως φαίνεται στην εικόνα, οι τρεις κύριοι φάκελοι είναι οι application, system και user_guide.

- user_guide

Ο φάκελος user_guide περιέχει ολόκληρο τον Οδηγό χρήστη Codeigniter. Αν χρειαστεί, σε οποιοδήποτε σημείο, είναι δυνατό να προβληθεί το τοπικό αντίγραφο του οδηγού στο http://localhost/codeigniterProject/user_guide. Επίσης ο (πάντα up-to-date) ακριβώς ίδιος οδηγός είναι επίσης διαθέσιμος στην ηλεκτρονική διεύθυνση <http://ellislab.com/codeigniter/user-guide>

- system

Ο φάκελος system περιέχει όλη την εσωτερική λειτουργία της Codeigniter. Όλα τα αρχεία που θα βοηθήσουν τροφοδοτηθεί η web εφαρμογή μας περιέχονται μέσα σε αυτόν το φάκελο.

- application

Ο φάκελος application είναι αυτός με τον οποίο ασχολείται κανείς περισσότερο. Είναι το μέρος όπου το σύνολο του κώδικα ειδικά για την εφαρμογή μας θα "ζήσει", συμπεριλαμβανομένων όλων των models, controllers, και views.

4.2.2 AngularJs



AngularJS[20] (που συνήθως αναφέρεται ως "Angular" ή "Angular.js") είναι ένα πλήρες, βασισμένο σε JavaScript, front-end framework ανοιχτού κώδικα για web εφαρμογές, το οποίο διατηρείται κυρίως από την Google και από μια κοινότητα ατόμων και επιχειρήσεων για την αντιμετώπιση των πολλών προκλήσεων που εμφανίζονται με την ανάπτυξη εφαρμογών μίας σελίδας. Έχει ως στόχο να απλοποιήσει τόσο την ανάπτυξη όσο και τη δοκιμή αυτών των εφαρμογών με την παροχή ενός πλαισίου για αρχιτεκτονικές front-end, model-View-Controller (MVC) και model-view-viewmodel (MVVM), μαζί με συστατικά(components) που

χρησιμοποιούνται συνήθως σε πλούσιες σε χαρακτηριστικά εφαρμογές Διαδικτύου.

Το AngularJS framework λειτουργεί κάνοντας πρώτα ανάγνωση της σελίδας HTML, στην οποία έχουν ενσωματώσει πρόσθετα προσαρμοσμένα tags. Η Angular ερμηνεύει αυτά τα χαρακτηριστικά, ως οδηγίες για δέσμευση εισόδου ή εξόδου μερών της σελίδας σε ένα μοντέλο(model) που αντιπροσωπεύεται από το πρότυπο μεταβλητών JavaScript. Οι τιμές αυτών των μεταβλητών JavaScript μπορούν να ρυθμιστούν χειροκίνητα μέσα από τον κώδικα, ή να ανακτηθούν από στατικούς ή δυναμικούς πόρους JSON.

Σύμφωνα με την JavaScript analytics υπηρεσία Libscore, η AngularJS χρησιμοποιείται στις ιστοσελίδες του Wolfram άλφα, NBC, Walgreens, Intel, Sprint, ABC News, και περίπου 8.400 άλλους δικτυακούς τόπους από 1 εκατομμύριο sites τον Ιούλιο του 2015.

Η AngularJS είναι το frontend μέρος της στοίβας MEAN , που αποτελείται από τη βάση δεδομένων MongoDB, Express.js web application server framework, την ίδια την Angular.js και το Node.js runtime περιβάλλον.

Παρακάτω κάποιες βασικές γενικές έννοιες και εργαλεία της AngularJs:

Scope

Η Angular χρησιμοποιεί τον όρο "scope" με ένα τρόπο παρόμοιο με τις βασικές αρχές της επιστήμης των υπολογιστών.

Το Scope στην επιστήμη των υπολογιστών περιγράφει, όταν σε ένα πρόγραμμα μια συγκεκριμένη δέσμευση είναι έγκυρη. Η προδιαγραφή ECMA-262 ορίζει το scope ως: ένα λεξικογραφικό περιβάλλον στο οποίο ένα αντικείμενο λειτουργίας(function) εκτελείται σε σενάρια web στο front-end.

Ως μέρος της αρχιτεκτονικής "MVC", το scope αποτελεί το "Model", και όλες οι μεταβλητές που ορίζονται στο scope μπορούν να είναι προσβάσιμες από το "View", καθώς και από τον "Controller". Το Scope συμπεριφέρεται σαν κόλλα και ενώνει το "View" και τον "Controller".

Bootstrap

Τα καθήκοντα που εκτελούνται από το AngularJS bootstrapper να συμβαίνουν σε τρεις φάσεις, αφού ο DOM έχει φορτωθεί:

1. Δημιουργία νέου Injector.
2. Η μεταγλώττιση των οδηγιών(directives) που κοσμούν το DOM.
3. Η σύνδεση όλων των directives στο scope.

Τα AngularJS directives επιτρέπουν στον προγραμματιστή να καθορίσει προσαρμοσμένα και επαναχρησιμοποιήσιμα HTML-like στοιχεία και χαρακτηριστικά που καθορίζουν τις συνδέσεις δεδομένων και τη συμπεριφορά των στοιχείων παρουσίασης.

Αμφίδρομη σύνδεση δεδομένων

Η «αμφίδρομη σύνδεση δεδομένων της AngularJS είναι το πιο αξιοσημείωτο χαρακτηριστικό της, αποφορτίζοντας σε μεγάλο βαθμό τις ευθύνες του backend διακομιστή. Αντ' αυτού, τα πρότυπα(templates) παρέχονται σε μορφή απλού HTML σύμφωνα με τα στοιχεία που περιέχονται στο scope που ορίζεται στο model. Η \$scope υπηρεσία της AngularJS ανιχνεύει αλλαγές στο model και τροποποιεί εκφράσεις HTML στο view μέσω ενός controller. Ομοίως, οι τυχόν αλλαγές στο view αντικατοπτρίζονται στο model. Αυτό παρακάμπτει την ανάγκη ενεργής χειραγώγησης του DOM και ενθαρρύνει ταχείς προτυποποιήσεις εφαρμογών web. Η AngularJS ανιχνεύει αλλαγές στα models συγκρίνοντας τις τρέχουσες τιμές με τις τιμές που αποθηκεύτηκαν νωρίτερα σε μια διαδικασία "βρώμικου ελέγχου"(dirty-checking).

4.2.3 Node.js



Το Node.js[21] είναι ένα ανοιχτού κώδικα, cross-platform JavaScript runtime περιβάλλον για την ανάπτυξη ενός μεγάλου φάσματος εργαλείων και εφαρμογών. Παρά το γεγονός ότι Node.js δεν είναι ένα JavaScript framework, πολλά από τα βασικά εργαλεία του είναι γραμμένα σε JavaScript, και οι προγραμματιστές μπορούν να γράψουν νέα εργαλεία σε JavaScript. Το περιβάλλον εκτέλεσης ερμηνεύει το JavaScript χρησιμοποιώντας το engine της Google V8 JavaScript.

Η αρχιτεκτονική του Node.js είναι καθοδηγούμενη από γεγονότα ικανή για ασύγχρονη I/O. Αυτές οι επιλογές σχεδιασμού στοχεύουν στη βελτιστοποίηση απόδοσης και επεκτασιμότητας σε εφαρμογές Web με πολλές λειτουργίες εισόδου / εξόδου, καθώς και σε εφαρμογές Web πραγματικού χρόνου(πχ., προγράμματα επικοινωνίας και παιχνίδια του προγράμματος περιήγησης πραγματικού χρόνου) .

Εταιρικοί χρήστες του λογισμικού Node.js περιλαμβάνουν GoDaddy, Groupon, IBM, LinkedIn, Microsoft, Netflix, PayPal, Rakuten, SAP, Voxer, Walmart, Yahoo!, και Cisco Systems.

4.2.4 Apache Cordova



Το Apache Cordova[22] (πρώην PhoneGap) είναι ένα δημοφιλές framework ανάπτυξης κινητών εφαρμογών που δημιουργήθηκε αρχικά από Nitobi. Η Adobe Systems αγόρασε το Nitobi το 2011, μετονομάστηκε ως PhoneGap, και αργότερα κυκλοφόρησε μια έκδοση ανοιχτού κώδικα του λογισμικού που ονομάζεται Apache Cordova. Το Apache Cordova επιτρέπει στους προγραμματιστές λογισμικού να δημιουργήσουν εφαρμογές για φορητές συσκευές που χρησιμοποιούν CSS3, HTML5 και JavaScript αντί να βασίζονται σε APIs συγκεκριμένης πλατφόρμας, όπως αυτά των Android, iOS ή Windows Phone. Επιτρέπει την "συσκευασία"(wrapping-up) του CSS, HTML και JavaScript κώδικα ανάλογα με την πλατφόρμα της συσκευής. Επεκτείνει τις δυνατότητες της HTML και JavaScript για να λειτουργούν με τη συσκευή. Οι προκύπτουσες εφαρμογές είναι υβριδικές, που σημαίνει ότι δεν είναι ούτε πραγματικά native εφαρμογές για κινητά (επειδή όλα τα rendering διάταξης γίνονται μέσω Web views, αντί να γίνονται στο τοπικό UI framework της πλατφόρμας), ούτε εντελώς Web-based (επειδή δεν είναι μόνο Web εφαρμογές, αλλά είναι συσκευασμένα σαν εφαρμογές για τη διανομή, και να έχουν πρόσβαση στα τοπικά APIs της συσκευής). Η ανάμιξη τοπικών και υβριδικών "κομματιών" κώδικα κατέστη δυνατή από την έκδοση 1.9.

Το λογισμικό στο παρελθόν ονομαζόταν απλά PhoneGap", μετά "Apache Callback. Ως λογισμικό ανοικτού κώδικα, το Apache Cordova επιτρέπει περιτυλίγματα(wrappers) γύρω από αυτό, όπως το Appery.io ή το Intel XDK.

Όπως το PhoneGap, πολλά άλλα εργαλεία και frameworks, επίσης, χτισμένα πάνω στην Cordova, συμπεριλαμβανομένων των Ionic, Monaca, TACO, η Intel XDK και το Telerik Platform. Τα εργαλεία αυτά χρησιμοποιούν Cordova, και όχι PhoneGap για τις βασικές λειτουργίες τους.

Χρηματοδότες του έργου Apache Cordova περιλαμβάνουν Adobe, BlackBerry, Google, IBM, Intel, Microsoft, Mozilla, και άλλους.

4.2.5 IONIC



Το Ionic[23] είναι ένα πλήρες SDK ανοιχτού κώδικα για υβριδική ανάπτυξη εφαρμογών για κινητά. Χτισμένο πάνω σε AngularJS και Apache Cordova, το Ionic παρέχει εργαλεία και υπηρεσίες για την ανάπτυξη υβριδικών κινητών εφαρμογών με χρήση τεχνολογιών Web όπως CSS, HTML5, και Sass. Οι εφαρμογές μπορούν να κατασκευαστούν με αυτές τις τεχνολογίες Web και στη συνέχεια διανέμονται μέσω τοπικών καταστημάτων app για να εγκατασταθούν στις συσκευές. Το Ionic δημιουργήθηκε από τον Max Lynch, Ben Sperry, και Adam Bradley της Drifty Co. Το 2013.

Κεφάλαιο 5

Παρουσίαση εφαρμογής

Στο παρόν κεφάλαιο αναλύονται οι εφαρμογές και τα συστήματα που δημιουργήθηκαν στα πλαίσια της παρούσας εργασίας.

5.1 Back-End

Στο κεφάλαιο 5.1 αναλύονται οι υλοποιήσεις για την βάση δεδομένων όπως επίσης και η εφαρμογή διακομιστή codeigniter.

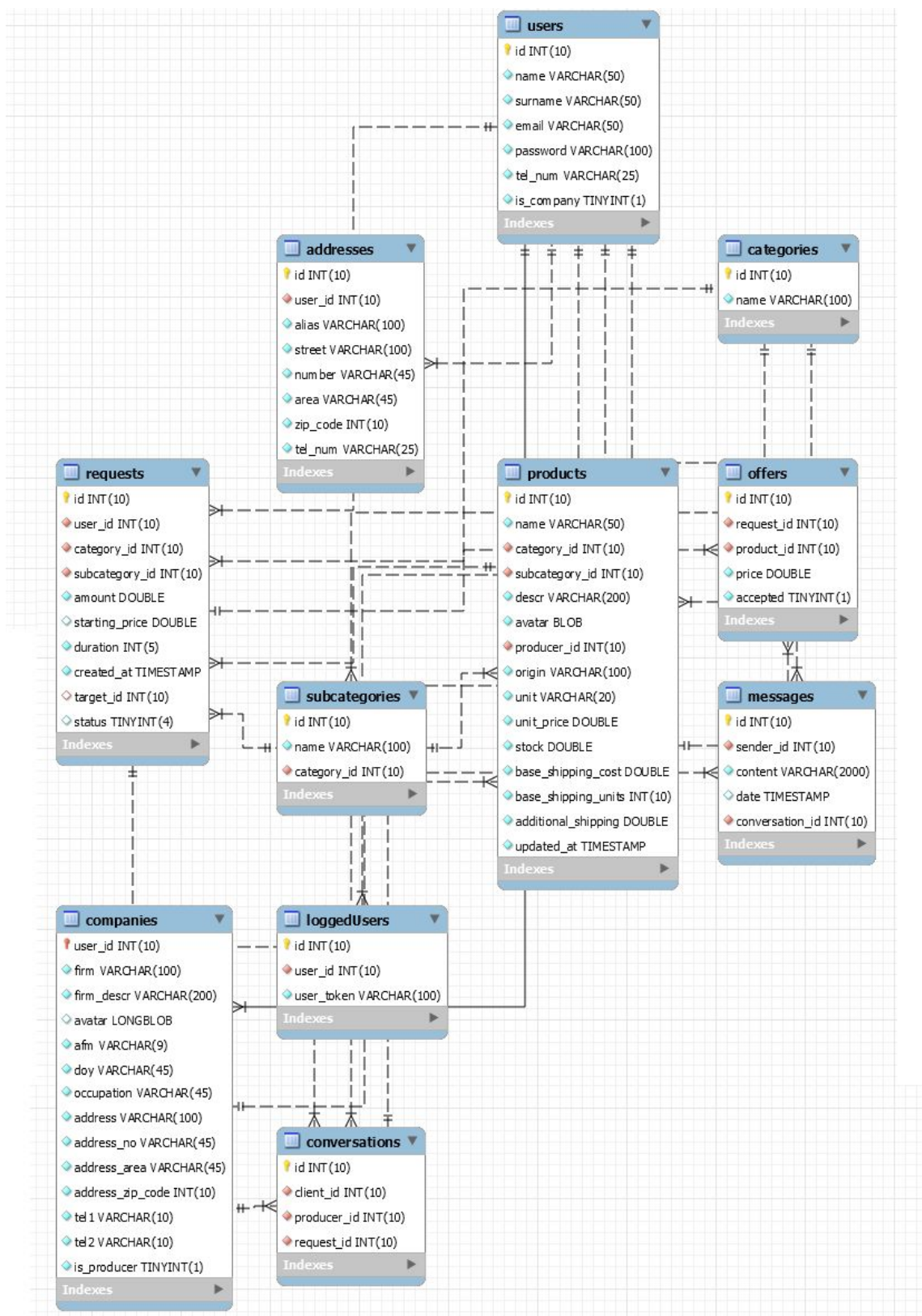
5.1.1 Βάση Δεδομένων

Η βάση δεδομένων είναι εγκατεστημένη στο gianta.xyz στο port 3306 και ονομάζεται farmit_dev. Είναι στημένη σε MySQL και έχω άμεση πρόσβαση σε αυτή μέσω του MySQL Workbench.

Παρακάτω το σχεσιακό διάγραμμα οντοτήτων της βάσης δεδομένων.

Αναλυτικά:

- Το table users είναι το table που περιέχει τα βασικά στοιχεία ενός χρήστη.
- Το table companies περιέχει τα στοιχεία εταιρίας ενός παραγωγού.
- Τα tables categories και subcategories περιέχουν τα ονόματα των κατηγοριών και υποκατηγοριών αντίστοιχα.
- Το table products περιέχει τα στοιχεία των προϊόντων των παραγωγών.
- Το table loggedUsers περιέχει στοιχεία για τους χρήστες που είναι συνδεδεμένοι αυτή τη στιγμή.
- Το table requests περιέχει στοιχεία για τις δημοπρασίες ή τις άμεσες αγορές που έχουν δημιουργηθεί (αν σε ένα request η μεταβλητή target_id είναι διαφορετική του null, τότε πρόκειται για άμεση αγορά και όχι για δημοπρασία. Το πεδίο status υποδεικνύει εάν μια αίτηση για άμεση πώληση έχει γίνει αποδεκτή από τον παραγωγό ή έχει απορριφθεί).
- Το table offers περιέχει στοιχεία για τις προσφορές που έχουν κάνει οι παραγωγοί για κάποιο request.
- Το table addresses περιέχει στοιχεία για τον προτιμώμενο τόπο συναλλαγών ενός χρήστη.
- Το table conversations περιέχει στοιχεία για τις συνομιλίες που έχουν δημιουργηθεί ανάμεσα στους παραγωγούς και στους απλούς χρήστες.
- Το table messages περιέχει όλα τα μηνύματα που έχουν σταλεί για μια συγκεκριμένη συνομιλία καθώς και στοιχεία για αυτά(πχ., ημερομηνία αποστολής).



εικόνα 17: EER diagram

Όλα τα δεδομένα των tables(με εξαίρεση τα categories και subcategories των οποίων τα δεδομένα είναι περασμένα χειροκίνητα) αποθηκεύονται/τροποποιούνται/διαγράφονται μέσω της back-end

codeigniter εφαρμογής η οποία υπάρχει στον διακομιστή.

5.1.2 Εφαρμογή Διακομιστή

Για την εκτέλεση των εργασιών που πρέπει να γίνουν στην βάση δεδομένων και για την αποστολή τους στο front-end δημιουργήθηκε μια restful εφαρμογή στον διακομιστή με την χρήση του Codeigniter MVC framework.

Η εφαρμογή που δημιουργήθηκε στον διακομιστή είναι επίσης υπεύθυνη για την δημιουργία, αναζήτηση, τροποποίηση, και διαγραφή δεδομένων από την βάση δεδομένων καθώς και για την αποστολή κατάλληλης “απάντησης” στην εφαρμογή front-end. Όλα τα δεδομένα που στέλνονται σαν “αίτηση”(request) ή απάντηση(response) είναι σε μορφή JSON. Οι controllers της εφαρμογής περιέχουν τις διεργασίες στις οποίες μπορώ να έχω πρόσβαση μέσω ενός μιας αίτησης http. Οι controllers με την σειρά τους κάνουν τις απαραίτητες τροποποιήσεις που χρειάζεται στα models(τα οποία αντιπροσωπεύουν τα tables της βάσης). Τέλος ένα view σε μορφή JSON με όλα τα απαραίτητα δεδομένα(πχ., success, error, response data κλπ) στέλνεται σαν απάντηση στο front-end.

Μετά από κάποιες τροποποιήσεις που χρειάστηκαν στα αρχεία συστήματος του ιδεατού server που χρησιμοποίησα καθώς και σε κάποια αρχεία της εφαρμογής, μπορεί κανείς να έχει πρόσβαση στις λειτουργίες αυτής της εφαρμογής με τον παρακάτω σύνδεσμο:
<http://localhost:8000/backend/index.php/api/όνομα-controller/όνομα-διεργασία>

5.1.2.1 Controllers

Σε περίπτωση που κάτι πάει στραβά κατά την εκτέλεση κάποιας συνάρτησης σε ένα controller, τότε στέλνεται μια απάντηση στο front-end με το κατάλληλο μήνυμα λάθους. Στις συναρτήσεις που χρειάζονται πιστοποίηση, μαζί με τα υπόλοιπα δεδομένα εισόδου στέλνεται επίσης και ένα “κλειδί”(token) για το οποίο θα μιλήσουμε αργότερα.

Οι controllers που δημιούργησα για τις ανάγκες της εφαρμογής αναλυτικά παρακάτω(με αλφαβητική σειρά) :

Addresses

Ο addresses controller είναι υπεύθυνος για όλες τις λειτουργίες που αφορούν την επιθυμητή τοποθεσία συναλλαγών των χρηστών.

Συναρτήσεις:

- **get_user_address_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο το email του χρήστη, και επιστρέφει ένα JSON αντικείμενο με τα στοιχεία επιθυμητής τοποθεσίας συναλλαγών που υπάρχουν στην βάση για αυτό το χρήστη.

- **new_address_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο τα απαραίτητα στοιχεία για την δημιουργία καινούριας εγγραφής επιθυμητής τοποθεσίας συναλλαγών στην βάση και καλεί τις απαραίτητες διεργασίες στο κατάλληλο model για την δημιουργία της. Τέλος, σε περίπτωση που όλα πάνε καλά(εάν δεν

προκύπτει κάποιο σφάλμα) επιστρέφεται μήνυμα επιτυχίας.

- **edit_address_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο τα απαραίτητα στοιχεία για την τροποποίηση κάποιας υπάρχουσας εγγραφής επιθυμητής τοποθεσίας συναλλαγών στην βάση και εκτελεί τις καλεί διεργασίες στο κατάλληλο model για την τροποποίησή της. Τέλος, σε περίπτωση που όλα πάνε καλά επιστρέφεται μήνυμα επιτυχίας.

Conversations

Ο conversations controller είναι υπεύθυνος για όλες τις λειτουργίες που αφορούν την συνομιλία ανάμεσα σε δύο χρήστες.

Συναρτήσεις:

- **get_user_conversations_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο το email ενός χρήστη, και επιστρέφει στοιχεία για όλες τις συνομιλίες που αφορούν αυτό το χρήστη με το email αυτό.

- **get_messages_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο ένα αναγνωριστικό(id) συνομιλίας, και επιστρέφει στοιχεία για όλα τα μηνύματα που έχουν σταλεί για αυτή τη συνομιλία.

- **send_message_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο ένα αναγνωριστικό συνομιλίας καθώς επίσης και το περιεχόμενο του μηνύματος που πρέπει να σταλεί, και καλεί την κατάλληλη λειτουργία model, ώστε να δημιουργηθεί μια καινούρια εγγραφή μηνύματος για την συγκεκριμένη συνομιλία.

General

Ο general controller είναι υπεύθυνος για όλες τις λειτουργίες που πρέπει να γίνουν οι οποίες δεν χρειάζονται κάποια ταυτοποίηση για να εκτελεσθούν.

Συναρτήσεις:

- **get_categories_post**

Αυτή η συνάρτηση επιστρέφει στοιχεία για όλες τις κατηγορίες που έχουν καταχωρηθεί στην βάση.

- **get_subcategories_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο ένα αναγνωριστικό κατηγορίας επιστρέφει στοιχεία για όλες τις υποκατηγορίες που υπάρχουν σε αυτή την κατηγορία.

- **get_server_date_time_post**

Αυτή η συνάρτηση επιστρέφει ένα timestamp με την ώρα που έχει αυτή τη στιγμή η βάση δεδομένων.(Χρειάζεται ώστε στο front-end να είναι δυνατό να υπολογιστεί η διαφορά ώρας

ανάμεσα στο server και τον χρήστη)

Offers

Ο offers controller είναι υπεύθυνος για όλες τις λειτουργίες που έχουν να κάνουν με τις προσφορές των παραγωγών για κάποια δημοπρασία.

Συναρτήσεις:

- **get_offers_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο ένα αναγνωριστικό δημοπρασίας, και επιστρέφει στοιχεία για όλες τις δέκα καλύτερες προσφορές που έχουν γίνει για αυτή τη δημοπρασία.

- **place_offers_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο τα απαραίτητα στοιχεία που χρειάζονται για την δημιουργία καινούριας προσφοράς για μια συγκεκριμένη δημοπρασία. Ελέγχει αν η δημοπρασία είναι ακόμα ενεργή(αν ο υπολειπόμενος χρόνος της δημοπρασίας είναι μεγαλύτερος του μηδέν), και αν ο παραγωγός έχει αρκετό απόθεμα για να εκπληρώσει την δημοπρασία σε περίπτωση που κερδίσει. Σε περίπτωση που και οι δύο έλεγχοι είναι θετικοί τότε καλείται η κατάλληλη μέθοδος στο model ώστε να δημιουργηθεί στην βάση μια καινούρια εγγραφή για τη συγκεκριμένη δημοπρασία και σε περίπτωση που όλα πάνε καλά, επιστρέφεται μήνυμα επιτυχίας.

- **get_user_offers_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο το email ενός χρήστη, και επιστρέφει στοιχεία για το ιστορικό καλύτερων προσφορών που έχει κάνει, για όλες τις δημοπρασίες στις οποίες έχει λάβει μέρος.

Products

Ο products controller είναι υπεύθυνος για όλες τις λειτουργίες που έχουν να κάνουν με τα προϊόντα των παραγωγών.

Συναρτήσεις:

- **new_product_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο όλα τα απαραίτητα στοιχεία για την δημιουργία καινούριου προϊόντος για κάποιον παραγωγό και καλεί την μέθοδο του κατάλληλου model ώστε να δημιουργηθεί η εγγραφή προϊόντος στην βάση δεδομένων. Τέλος, επιστρέφεται μήνυμα επιτυχίας μαζί με το αναγνωριστικό του καινούριου προϊόντος που δημιουργήθηκε.

- **edit_product_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο τα απαραίτητα στοιχεία που χρειάζονται για την δημιουργία καινούριας προσφοράς για μια συγκεκριμένη δημοπρασία. Ελέγχει αν η δημοπρασία είναι ακόμα ενεργή(αν ο υπολειπόμενος χρόνος της δημοπρασίας είναι μεγαλύτερος του μηδέν), και αν ο παραγωγός έχει αρκετό απόθεμα για να εκπληρώσει την δημοπρασία σε περίπτωση που κερδίσει. Σε περίπτωση που και οι δύο έλεγχοι είναι θετικοί τότε καλείται η κατάλληλη μέθοδος στο model ώστε να δημιουργηθεί στην βάση μια καινούρια εγγραφή για τη συγκεκριμένη δημοπρασία και σε περίπτωση που όλα πάνε καλά, επιστρέφεται μήνυμα επιτυχίας.

- **delete_product_post**

Αυτή η λειτουργία παίρνει σαν είσοδο το αναγνωριστικό ενός προϊόντος, έπειτα καλώντας την κατάλληλη μέθοδο model ελέγχει αν το προϊόν είναι ενεργό σε κάποια δημοπρασία (αν δηλαδή υπάρχει κάποια δημοπρασία, της οποίας η καλύτερη προσφορά να αφορά το παρών προϊόν). Σε αυτή την περίπτωση επιστρέφεται κατάλληλο μήνυμα. Διαφορετικά το προϊόν διαγράφεται και επιστρέφεται μήνυμα επιτυχίας.

- **change_picture_post**

Αυτή η λειτουργία παίρνει σαν είσοδο το αναγνωριστικό ενός προϊόντος και ένα αρχείο φωτογραφίας, και έπειτα καλώντας την κατάλληλη μέθοδο, αποθηκεύει τροποποιεί το συγκεκριμένο προϊόν αλλάζοντας την φωτογραφία του. Τέλος η καινούρια φωτογραφία που αποθηκεύτηκε, μετατρέπεται σε base64 μορφή και επιστρέφεται στο front-end, μαζί με ένα μήνυμα επιτυχίας.

- **get_category_products_post**

Αυτή η λειτουργία παίρνει σαν είσοδο το αναγνωριστικό μιας κατηγορίας και μιας υποκατηγορίας και το email του χρήστη, έπειτα καλώντας την κατάλληλη μέθοδο, επιστρέφει τα στοιχεία των προϊόντων που έχει ο χρήστης για τη συγκεκριμένη κατηγορία και υποκατηγορία αφού πρώτα έχει μετατρέψει το αρχείο εικόνας του κάθε προϊόντος από blob σε μορφή base64.

- **get_products_post**

Αυτή η λειτουργία παίρνει σαν είσοδο το email του χρήστη, και επιστρέφει στοιχεία για όλα τα προϊόντα που έχει ο χρήστης, αφού πρώτα έχει μετατρέψει το αρχείο εικόνας του κάθε προϊόντος από blob σε μορφή base64.

- **get_product_post**

Αυτή η λειτουργία παίρνει σαν είσοδο το αναγνωριστικό ενός προϊόντος, έπειτα καλώντας την κατάλληλη μέθοδο, επιστρέφει τα στοιχεία του προϊόντος αυτού, αφού πρώτα έχει μετατρέψει το αρχείο εικόνας του προϊόντος από blob σε μορφή base64.

Requests

Ο requests controller είναι υπεύθυνος για όλες τις λειτουργίες που αφορούν τις δημοπρασίες και τις άμεσες πωλήσεις.

Συναρτήσεις:

- **home_requests_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο τις παραμέτρους ταξινόμησης που έχει εισάγει ο χρήστης(αν υπάρχουν), και έπειτα επιστρέφει τις δέκα πρώτες δημοπρασίες που προκύπτουν από την ταξινόμηση αυτή.

- **search_requests_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο τις παραμέτρους ταξινόμησης που έχει εισάγει ο χρήστης(αν υπάρχουν) καθώς επίσης και αναγνωριστικό κατηγορίας και υποκατηγορίας, και έπειτα επιστρέφει τις δέκα πρώτες δημοπρασίες που προκύπτουν από την ταξινόμηση αυτή και ανήκουν στην

κατηγορία και στην υποκατηγορία που εισήχθη.

- **suitable_requests_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο τις παραμέτρους ταξινόμησης που έχει εισάγει ο χρήστης(αν υπάρχουν) καθώς επίσης και το email του χρήστη, και έπειτα επιστρέφει τις δέκα πρώτες δημοπρασίες των οποίων η κατηγορία και υποκατηγορία είναι ίδιες με την κατηγορία και υποκατηγορία των προϊόντων του χρήστη και έχουν προκύψει από την παραπάνω ταξινόμηση.

- **instant_requests_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο το email του χρήστη, και έπειτα χρησιμοποιώντας την κατάλληλη μέθοδο, επιστρέφει τα requests που αναφέρονται σε άμεσες πωλήσεις τα οποία δεν έχουν διευθετηθεί ακόμα από τον παραγωγό(το πεδίο status είναι null).

- **decide_instant_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο το αναγνωριστικό ενός request το οποίο πρόκειται για άμεση πώληση και την απόφαση του παραγωγού(αν το αποδέχθηκε ή το απέρριψε), και έπειτα αλλάζει το πεδίο status με βάση την απόφαση του χρήστη. Επιπλέον αν ο χρήστης έχει αποδεχθεί αυτή την άμεση πώληση, τότε αφαιρείται το ποσό που ζητήθηκε από το απόθεμα του παραγωγού και δημιουργείται μία καινούρια συνομιλία ανάμεσα στον χρήστη που έκανε την αίτηση για άμεση αγορά και τον παραγωγό, για την συγκεκριμένη αίτηση αγοράς. Τέλος επιστρέφεται μήνυμα επιτυχίας.

- **get_request_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο το αναγνωριστικό μιας δημοπρασίας και επιστρέφει τα απαραίτητα δεδομένα που χρειάζονται στο front-end.

- **get_complete_request_post**

Όμοια με την get_request_post μόνο που τώρα μαζί με τα δεδομένα του request, επιστρέφεται επίσης και η τιμή της καλύτερης προσφοράς που έγινε για αυτό το request.

Users

Ο users controller είναι υπεύθυνος για όλες τις λειτουργίες που έχουν να κάνουν με το προφίλ των χρηστών και το προφίλ εταιρίας των παραγωγών

Συναρτήσεις:

- **register_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο τα στοιχεία που χρειάζονται για την εγγραφή ενός καινούριου χρήστη, και αν όλα πάνε καλά (πχ., δεν υπάρχει ήδη άλλος χρήστης με το ίδιο email) τότε ο καινούριος χρήστης εισάγεται στην βάση δεδομένων(ο κωδικός του χρήστη αποθηκεύεται σε μορφή hash για μεγαλύτερη ασφάλεια), δημιουργείται ένα κλειδί(token) και αποθηκεύεται στο table loggedUsers το email του χρήστη και το token του. Τέλος, επιστρέφεται το token και το session(κάποια στοιχεία του χρήστη που θα αποθηκεύονται και τοπικά στην συσκευή του χρήστη).

- **login_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο το email του χρήστη και τον κωδικό του, και ελέγχει αν υπάρχει χρήστης με αυτό το συνδυασμό email και κωδικού. αν βρεθεί δημιουργείται ένα

κλειδί(token) και αποθηκεύεται στο table loggedUsers το email του χρήστη και το token του. Τέλος, επιστρέφεται το token και το session(κάποια στοιχεία του χρήστη που θα αποθηκεύονται και τοπικά στην συσκευή του χρήστη).

- **edit_acount_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο τα καινούρια στοιχεία του χρήστη και έπειτα τροποποιεί τα παλιά του δεδομένα από την βάση ανάλογα με τα καινούρια. Τέλος επιστρέφεται το session.

- **logout_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο το email του χρήστη, και διαγράφει την εγγραφή με το συγκεκριμένο email από το loggedUsers table.

- **check_password_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο το email του χρήστη και ένα password, και ελέγχει αν το password που έχει εισάγει ο χρήστης είναι σωστό, και επιστρέφεται κατάλληλο μήνυμα ανάλογα με το αποτέλεσμα του ελέγχου(Αυτή η συνάρτηση χρησιμοποιείται στην περίπτωση που ο χρήστης θέλει να αλλάξει τον κωδικό του. Για να του επιτραπεί να τον αλλάξει, πρέπει να έχει βάλει πρώτα σωστά τον παλιό του κωδικό).

- **change_password_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο το email του χρήστη και ένα password, αλλάζει το password που υπάρχει αποθηκευμένο στην βάση δεδομένων με το παρών password. Τέλος αν όλα πάνε καλά επιστρέφεται μήνυμα επιτυχίας.

- **add_company_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο τα στοιχεία που χρειάζονται για να δημιουργήσει ένας χρήστης προφίλ για την εταιρία του. Αν όλα πάνε καλά, επιστρέφεται μήνυμα επιτυχίας.

- **edit_company_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο τα στοιχεία που χρειάζονται για να τροποποιήσει ένας χρήστης το προφίλ της εταιρίας του. Αν όλα πάνε καλά(και τροποποιηθεί η εγγραφή της βάσης επιτυχώς), επιστρέφεται μήνυμα επιτυχίας.

- **get_company_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο το email του χρήστη και επιστρέφει τα στοιχεία της εταιρίας του.

- **get_picture_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο το email του χρήστη και επιστρέφει τη φωτογραφία που είναι αποθηκευμένη για την εταιρία του συγκεκριμένου χρήστη σε μορφή base64.

- **change_picture_post**

Αυτή η συνάρτηση παίρνει σαν είσοδο το email του χρήστη και ένα αρχείο φωτογραφίας και αλλάζει την φωτογραφία που είναι αποθηκευμένη στην βάση με την φωτογραφία εισόδου. Τέλος επιστρέφει την φωτογραφία που μόλις αποθηκεύτηκε σε μορφή base64(αυτό συμβαίνει για να σιγουρευτεί ο χρήστης ότι η φωτογραφία αποθηκεύτηκε σωστά και μπορεί να φανεί κανονικά).

ServerCommands

Όλοι οι controllers που ανέφερα παραπάνω είναι μέρος της rest υπηρεσίας. Υπάρχει ένας ακόμα controller, ο serverCommands, ο οποίος δεν είναι μέρος γι' αυτήν. Χρησιμοποιείται για να τρέχει τις εντολές που χρειάζεται να εκτελεστούν έτσι και αλλιώς, χωρίς δηλαδή να περιμένουν κάποια αίτηση από το front-end.

Συναρτήσεις:

- **check_for_completed_requests**

Αυτή μόνη τέτοια συνάρτηση που χρειάστηκε να δημιουργήσω στα πλαίσια της εργασίας. Δεν παίρνει τίποτα σαν είσοδο. Χρησιμοποιώντας μεθόδους από τα κατάλληλα models, βρίσκει όλες τις δημοπρασίες οι οποίες έχουν λήξει, έχουν γίνει προσφορές γ' αυτές, αλλά δεν έχει οριστεί κάποια προσφορά ως νικητήρια. Έπειτα για κάθε τέτοια δημοπρασία, θέτεται η καλύτερη προσφορά σε αυτήν ως νικητήρια(το πεδίο accepted στο συγκεκριμένο offer που νίκησε γίνεται 1), αφαιρείται το ποσό που ζητήθηκε στην δημοπρασία από το απόθεμα του παραγωγού που κέρδισε, και τέλος δημιουργείται μια καινούρια εγγραφή συνομιλίας ανάμεσα στους δύο χρήστες(αυτόν που δημιούργησε την δημοπρασία και αυτόν που την κέρδισε) για την συγκεκριμένη δημοπρασία.

Cron - Χρονοπρογραμματισμός εργασιών

Το λογισμικό Cron είναι ένας χρονοπρογραμματιστής σε Unix ή παρόμοια λειτουργικά συστήματα ηλεκτρονικών υπολογιστών. Οι άνθρωποι που δημιουργούν και συντηρούν περιβάλλοντα λογισμικού χρησιμοποιούν cron για να προγραμματίσουν εργασίες (εντολές ή κάποια scripts) να τρέχουν περιοδικά σε συγκεκριμένες ώρες, ημερομηνίες ή χρονικά διαστήματα.

Το cron job που δημιούργησα για να τρέχει η παραπάνω συνάρτηση στον server, είναι το εξής:

```
*/1 * * * * php /var/www/backend/index.php serverCommands check_for_completed_requests
```

Το `*/1 * * * *` σημαίνει ότι η συγκεκριμένη εντολή θα εκτελείται κάθε 1 λεπτό.

Php να εκτελεστεί ο php κώδικας του αρχείου που ακολουθεί.

`/var/www/backend/index.php` το μονοπάτι για το αρχείο `index.php` της codeigniter. Μέσω αυτού του αρχείου γίνονται οι απαραίτητες ενέργειες για να λειτουργήσει το framework.

ServerCommands η πρώτη παράμετρος του αρχείου `index.php` ή αλλιώς ο controller που θέλω να χρησιμοποιηθεί.

`check_for_completed_requests` η δεύτερη παράμετρος του `index.php` ή αλλιώς, η συνάρτηση του controller που θέλω να εκτελεστεί.

Σημείωση: Το κλειδί(token) είναι ένα σύνολο από τυχαίους χαρακτήρες το οποίο δημιουργείται στο διακομιστή όταν ένας χρήστης συνδέεται στην εφαρμογή. Το token μαζί με το email του χρήστη αποθηκεύονται στο loggedUsers table και ταυτόχρονα το token στέλνεται και στον χρήστη ώστε να μπορεί να το αποθηκεύσει τοπικά στην συσκευή του. Κάθε φορά που ο χρήστης θέλει να αποκτήσει πρόσβαση σε κάποια λειτουργία στην οποία δεν πρέπει να έχει πρόσβαση ο οποιοσδήποτε(πχ., edit_profile), τότε μαζί με τα υπόλοιπα δεδομένα που θα στέλνονταν κανονικά, για να εκτελεστεί η λειτουργία αυτή, πρέπει να σταλεί επίσης το email του χρήστη και το token που έχει αποθηκευμένο ο χρήστης στην συσκευή του. Κατόπιν ελέγχεται αν το token που είναι αποθηκευμένο στο loggedUsers table της βάσης δεδομένων, είναι ίδιο με αυτό που έστειλε ο χρήστης. Αν ναι, τότε ο χρήστης έχει πρόσβαση στην λειτουργία, αλλιώς, η πρόσβαση του απαγορεύεται.

5.1.2.2 Models

Τα models αντικατοπτρίζουν τα στοιχεία της βάσης δεδομένων. Τα models που χρειάστηκε να δημιουργήσω στα πλαίσια της codeigniter εφαρμογής είναι τα εξής:

Address_model

Το address_model, περιέχει όλες τις μεθόδους που χρειάστηκα για να κάνω αλλαγές στο addresses table.

Συναρτήσεις:

- **getUserAddress**

Παίρνει σαν είσοδο το αναγνωριστικό ενός χρήστη και επιστρέφει στοιχεία για την επιθυμητή διεύθυνση συναλλαγών του.

- **edit_address**

Αλλάζει τα στοιχεία επιθυμητής διεύθυνσης ενός χρήστη.

- **new_address**

Δημιουργεί μια καινούρια εγγραφή διεύθυνσης για ένα συγκεκριμένο χρήστη.

Categories_model

Το categories_model, περιέχει όλες τις μεθόδους που χρειάστηκα για για την προβολή κατηγοριών και υποκατηγοριών.

Συναρτήσεις:

- **getAllCategories και getSubcategories**

Η getAllCategories επιστρέφει όλα τα στοιχεία που υπάρχουν για τις κατηγορίες, ενώ η

getSubcategories παίρνει σαν είσοδο το αναγνωριστικό μιας κατηγορίας και επιστρέφει στοιχεία που υπάρχουν για τις υποκατηγορίες που υπάγονται στην κατηγορία που εισήχθη.

Conversations_model

Το conversations_model, περιέχει όλες τις μεθόδους που χρειάστηκα για την δημιουργία συνομιλιών και αποστολή μηνυμάτων.

Συναρτήσεις:

- **get_user_conversations**

Παίρνει σαν είσοδο το αναγνωριστικό ενός χρήστη και επιστρέφει στοιχεία για όλες τις συνομιλίες που αφορούν αυτό το χρήστη.

- **new_conversation**

Παίρνει σαν είσοδο τα δεδομένα που χρειάζονται για την δημιουργία μιας καινούριας συνομιλίας, και την δημιουργεί.

- **get_conversation_data_instant_request**

Παίρνει σαν είσοδο το αναγνωριστικό μιας άμεσης πώλησης και επιστρέφει τα στοιχεία που χρειάζονται για να δημιουργηθεί μια συνομιλία ανάμεσα στους χρήστες που τους αφορά, με τέτοιο τρόπο, ώστε αν τα δεδομένα εισαχθούν όπως επιστραφούν στην new_conversation, θα δημιουργηθεί η συνομιλία που θέλουμε χωρίς να χρειαστούν επιπλέον τροποποιήσεις.

- **get_conversation_data_request**

Παρομοίως με την παραπάνω συνάρτηση, μόνο που τώρα η είσοδος είναι αναγνωριστικό δημοπρασίας, και όχι άμεσης πώλησης.

- **get_conversation_messages**

Παίρνει σαν είσοδο το αναγνωριστικό μιας συνομιλίας, και επιστρέφει στοιχεία για όλα τα μηνύματα που έχουν σταλεί στα πλαίσια της συγκεκριμένης συνομιλίας.

- **send_message**

Παίρνει σαν είσοδο τα στοιχεία που χρειάζονται για την δημιουργία καινούριας εγγραφής στο messages table, και την δημιουργεί.

Logged_users_model

Το logged_users_model, περιέχει όλες τις μεθόδους που χρειάστηκα για να αποθηκεύω κάποια στοιχεία για τους χρήστες που είναι online.

Συναρτήσεις:

- **add**

Παίρνει σαν είσοδο το αναγνωριστικό ενός χρήστη και το token του, και δημιουργεί μια καινούρια εγγραφή συνδεδεμένου χρήστη στο loggedUsers table.

- **remove**

Παίρνει σαν είσοδο το αναγνωριστικό ενός χρήστη και διαγράφει την εγγραφή συνδεδεμένου χρήστη για αυτό το χρήστη.

- **getToken**

Παίρνει σαν είσοδο το αναγνωριστικό ενός χρήστη και επιστρέφει το token του έχει αποθηκευτεί.

Offers_model

Το Offers_model, περιέχει όλες τις μεθόδους που χρειάστηκα για την διαχείριση των προσφορών των χρηστών.

Συναρτήσεις:

- **get_offers**

Παίρνει σαν είσοδο το αναγνωριστικό μιας δημοπρασίας και επιστρέφει στοιχεία για τις δέκα καλύτερες προσφορές που έχουν γίνει για αυτή.

- **get_user_offers**

Παίρνει σαν είσοδο το αναγνωριστικό ενός χρήστη και επιστρέφει στοιχεία για την καλύτερη προσφορά που έχει κάνει σε κάθε δημοπρασία που έχει πάρει μέρος.

- **place_bid**

Παίρνει σαν είσοδο το αναγνωριστικό τα στοιχεία που χρειάζονται για να δημιουργηθεί μια καινούρια προσφορά για κάποια δημοπρασία, και την δημιουργεί.

- **accept_offer_request**

Αυτή η συνάρτηση παίρνει σαν είσοδο το αναγνωριστικό μιας δημοπρασίας, και έπειτα θέτει την καλύτερη προσφορά που έχει γίνει για αυτή τη δημοπρασία ως νικητήρια και τέλος αφαιρεί το ποσό που ζητήθηκε από το απόθεμα που έχει ο παραγωγός που κέρδισε για το προϊόν που ζητήθηκε.

Products_model

Το Products_model, περιέχει όλες τις μεθόδους που χρειάστηκα για την διαχείριση των προϊόντων των χρηστών.

Συναρτήσεις:

- **add_new_product**

Παίρνει σαν είσοδο τα στοιχεία που χρειάζονται για την δημιουργία ενός καινούριου προϊόντος για κάποιο παραγωγό, και το δημιουργεί.

- **check_prod_stock**

Παίρνει σαν είσοδο το αναγνωριστικό ενός προϊόντος και το αναγνωριστικό μιας δημοπρασίας(ή άμεσης πώλησης) και ελέγχει αν το απόθεμα του συγκεκριμένου προϊόντος είναι περισσότερο από το ποσό που έχει ζητηθεί. Αν ισχύει αυτή η συνθήκη επιστρέφεται αλήθεια.

- **get_all_products**

Παίρνει σαν είσοδο το αναγνωριστικό ενός χρήστη και επιστρέφει στοιχεία για όλα τα προϊόντα που έχει καταχωρίσει.

- **get_category_products**

Αυτή η συνάρτηση παίρνει σαν είσοδο το αναγνωριστικό ενός χρήστη, καθώς και το όνομα μιας κατηγορίας και υποκατηγορίας και επιστρέφει στοιχεία για όλα τα προϊόντα που έχει ο χρήστης για αυτή την κατηγορία και υποκατηγορία.

- **get_product**

Αυτή η συνάρτηση παίρνει σαν είσοδο το αναγνωριστικό ενός προϊόντος και επιστρέφει στοιχεία για αυτό.

- **change_product_avatar**

Αυτή η συνάρτηση παίρνει σαν είσοδο το αναγνωριστικό ενός προϊόντος, καθώς τα δεδομένα μιας φωτογραφίας σε μορφή JSON(δηλαδή { 'avatar' => 'avatar_data' }) και αντικαθιστά την φωτογραφία που υπάρχει είδη για αυτό το προϊόν, με την καινούρια φωτογραφία.

- **get_product_avatar**

Αυτή η συνάρτηση παίρνει σαν είσοδο το αναγνωριστικό ενός προϊόντος, και επιστρέφει την φωτογραφία του.

- **edit_product**

Αυτή η συνάρτηση παίρνει σαν είσοδο το αναγνωριστικό ενός προϊόντος, καθώς τα δεδομένα που θέλουμε να τροποποιήσουμε και αντικαθιστά τα δεδομένα που υπάρχουν στην βάση για αυτό το προϊόν με τα καινούρια δεδομένα.

- **remove_product**

Αυτή η συνάρτηση παίρνει σαν είσοδο το αναγνωριστικό ενός προϊόντος και το διαγράφει από την βάση δεδομένων.

- **check_for_removal**

Αυτή η συνάρτηση παίρνει σαν είσοδο το αναγνωριστικό ενός προϊόντος, και ελέγχει αν το προϊόν αυτό συμμετέχει ενεργά σε κάποια δημοπρασία(αν δηλαδή η καλύτερη προσφορά που έχει γίνει για κάποια δημοπρασία της οποίας ο χρόνος δεν έχει τελειώσει, αναφέρεται στο συγκεκριμένο προϊόν). Τέλος επιστρέφεται κατάλληλο μήνυμα ανάλογα με το αποτέλεσμα.

Requests_model

Το Requests_model, περιέχει όλες τις μεθόδους που χρειάστηκα για την διαχείριση των προϊόντων των χρηστών.

Συναρτήσεις:

- **get_home_requests**

Αυτή η συνάρτηση παίρνει σαν είσοδο τις παραμέτρους ταξινόμησης που έχει εισάγει ο χρήστης(αν υπάρχουν), και έπειτα επιστρέφει στοιχεία τις δέκα πρώτες δημοπρασίες που προκύπτουν από την ταξινόμηση αυτή.

- **get_complete_request**

Παίρνει σαν είσοδο το αναγνωριστικό μιας δημοπρασίας που έχει λήξει και επιστρέφει τα κατάλληλα στοιχεία που αφορούν την δημοπρασία αυτή.

- **get_request**

Παίρνει σαν είσοδο το αναγνωριστικό μιας ενεργής δημοπρασίας και επιστρέφει τα κατάλληλα στοιχεία που αφορούν την δημοπρασία αυτή.

- **request_has_finished**

Παίρνει σαν είσοδο το αναγνωριστικό μιας δημοπρασίας και επιστρέφει σωστό ή λάθος, ανάλογα με το αν ο χρόνος της δημοπρασίας έχει τελειώσει ή όχι.

- **get_suitable_requests**

Αυτή η συνάρτηση παίρνει σαν είσοδο τις παραμέτρους ταξινόμησης καθώς και το αναγνωριστικό ενός χρήστη και επιστρέφει και έπειτα επιστρέφει στοιχεία τις δέκα πρώτες δημοπρασίες που προκύπτουν από την ταξινόμηση αυτή και αφορούν τα προϊόντα που έχει εισάγει ο χρήστης στην βάση.

- **get_new_instant_requests**

Αυτή η συνάρτηση παίρνει σαν είσοδο το αναγνωριστικό ενός παραγωγού και επιστρέφει όλες τις καινούριες αιτήσεις για άμεση πώληση που του έχουν γίνει (αυτές δηλαδή που δεν έχουν αποδεχτεί από τον παραγωγό, ούτε έχουν απορριφθεί).

- **decide_instant**

Αυτή η συνάρτηση παίρνει σαν είσοδο το αναγνωριστικό μιας αίτησης για άμεση πώληση και την απόφαση του χρήστη για αυτή (αν την απέρριψε ή την αποδέχτηκε) και τροποποιεί την άμεση πώληση ανάλογα με την απόφαση, και στην συνέχεια σε περίπτωση που η απόφαση ήταν αποδοχή, αφαιρείται το ποσό που ζητήθηκε από το απόθεμα του παραγωγού.

- **get_search_requests**

Αυτή η συνάρτηση παίρνει σαν είσοδο τις παραμέτρους ταξινόμησης που έχει εισάγει ο χρήστης(αν υπάρχουν) καθώς επίσης και αναγνωριστικό κατηγορίας και υποκατηγορίας, και έπειτα επιστρέφει τις δέκα πρώτες δημοπρασίες που προκύπτουν από την ταξινόμηση αυτή και ανήκουν στην κατηγορία και στην υποκατηγορία που εισήχθη.

- **get_completed_requests**

Επιστρέφει στοιχεία για όλες τις δημοπρασίες των οποίων ο χρόνος έχει τελειώσει και η καλύτερη προσφορά που έχει γίνει για αυτές, δεν είναι αποδεκτή.

Users_model

Το Users_model, περιέχει όλες τις μεθόδους που χρειάστηκα για την δημιουργία του προφίλ των χρηστών και της εταιρίας τους.

Συναρτήσεις:

- **get_DB_time**

Επιστρέφει ένα timestamp με την ώρα που έχει η βάση δεδομένων αυτή την στιγμή.

- **get_user_id_by_email**

Παίρνει σαν είσοδο το email ενός χρήστη και επιστρέφει το αναγνωριστικό του.

- **add_new_user**

Παίρνει σαν είσοδο τα δεδομένα που χρειάζονται για την δημιουργία καινούριου χρήστη και τον δημιουργεί.

- **get_user_pass_by_email**

Παίρνει σαν είσοδο το email ενός χρήστη και επιστρέφει τον κωδικό του.

- **get_user_id_pass_by_email**

Παίρνει σαν είσοδο το email ενός χρήστη και επιστρέφει τον κωδικό του και το αναγνωριστικό του, σε μορφή JSON.

- **get_user_session**

Παίρνει σαν είσοδο το αναγνωριστικό ενός χρήστη και τα στοιχεία του λογαριασμού του, τα οποία πρέπει να αποθηκευτούν στην συσκευή του χρήστη καθ' όλη την διάρκεια που ο χρήστης θα είναι συνδεδεμένος στην εφαρμογή.

- **update_user**

Παίρνει σαν είσοδο το αναγνωριστικό ενός χρήστη και τα καινούρια στοιχεία του λογαριασμού του και αντικαθιστά τα παλιά στοιχεία που υπάρχουν στην βάση δεδομένων με τα καινούρια.

- **add_new_company**

Παίρνει σαν είσοδο το αναγνωριστικό ενός χρήστη και τα στοιχεία που χρειάζονται για την δημιουργία καινούριας εταιρίας, και την δημιουργεί.

- **get_company**

Παίρνει σαν είσοδο το αναγνωριστικό ενός χρήστη και επιστρέφει στοιχεία για τον λογαριασμό εταιρίας του.

- **edit_company**

Παίρνει σαν είσοδο το αναγνωριστικό ενός χρήστη και τα καινούρια στοιχεία του λογαριασμού της εταιρίας του και αντικαθιστά τα παλιά στοιχεία που υπάρχουν στην βάση δεδομένων με τα καινούρια.

- **change_user_avatar**

Παίρνει σαν είσοδο το αναγνωριστικό ενός χρήστη και την καινούρια φωτογραφία και αντικαθιστά

την παλιά φωτογραφία εταιρίας του χρήστη με την καινούρια.

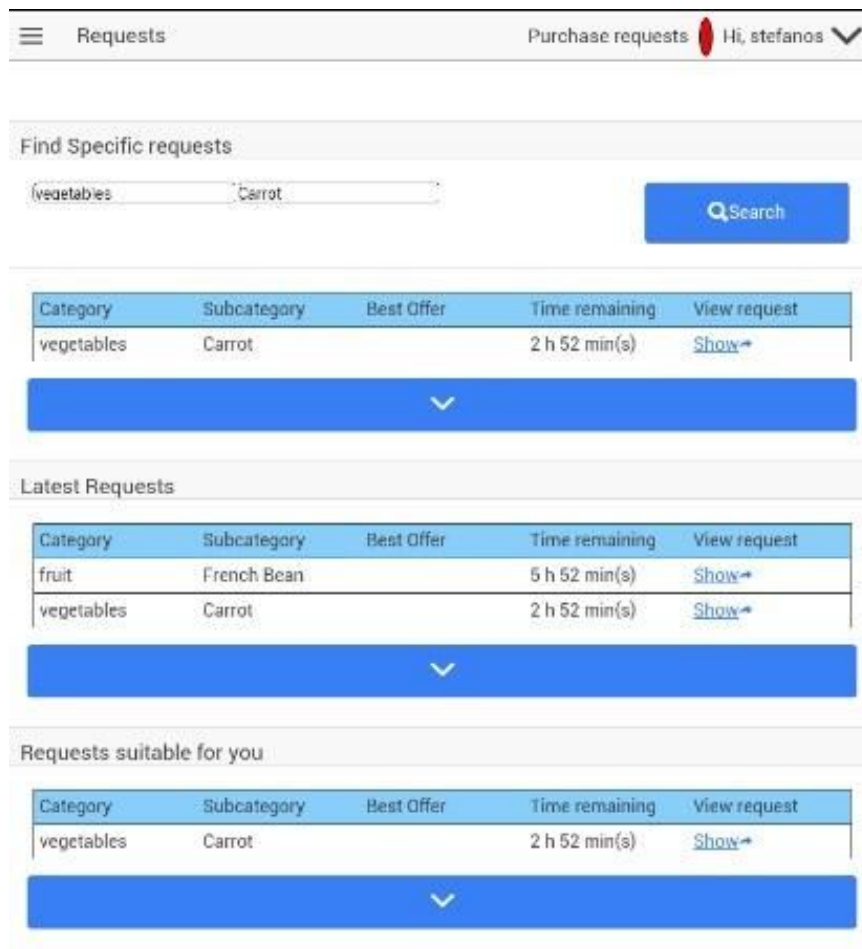
- **get_user_avatar**

Παίρνει σαν είσοδο το αναγνωριστικό ενός χρήστη και επιστρέφει την φωτογραφία της εταιρίας του.

5.2 Front-End

Η εφαρμογή front-end που δουλεύει στο κινητό του χρήστη, αποφασίστηκε να είναι υβριδική ώστε να μπορεί να απευθυνθεί σε μεγαλύτερο εύρος χρηστών και αποφασίστηκε να είναι συμβατή συγκεκριμένα με τις πλατφόρμες android και ios οι οποίοι αποτελούν τα μεγαλύτερα ποσοστά χρηστών στον κόσμο. Το framework που χρησιμοποιήθηκε είναι το Ionic MVC framework.

5.2.1 Παραδείγματα Εκτέλεσης



εικόνα 18: αρχική σελίδα εφαρμογής

Ο χρήστης μόλις συνδεθεί, μπορεί να δει ενεργές δημοπρασίες στην αρχική του σελίδα. Πατώντας σε οποιοδήποτε πεδίο σε κάποιο πίνακα, τότε θα αλλάξει ο αλγόριθμος ταξινόμησης, με βάση το πεδίο που πάτησε. Το κόκκινο εικονίδιο στην κορυφή της οθόνης λειτουργεί σαν ειδοποίηση στον χρήστη ότι υπάρχουν αιτήσεις για άμεσες αγορές που πρέπει να αποδεχτεί ή να απορρίψει.

Όλοι οι πίνακες εμφανίζουν το πολύ 10 δημοπρασίες. Για να μειωθούν τα δεδομένα που χρησιμοποιούνται και να μείνει ο χρόνος απόκρισης στο ελάχιστο δυνατό. Οι χρήστες μπορούν να “κατευθυνθούν” στις δημοπρασίες και να δουν τις 10 επόμενες ή προηγούμενες, πατώντας το βελάκι προς τα κάτω ή το βελάκι προς τα πάνω αντίστοιχα.

Ο πρώτος πίνακας εμφανίζει δημοπρασίες με βάση την αναζήτηση που έχει κάνει ο χρήστης.

Ο δεύτερος πίνακας εμφανίζει τις δημοπρασίες με τον λιγότερο διαθέσιμο χρόνο.

Ο τρίτος πίνακας εμφανίζει μόνο τις δημοπρασίες τις οποίες ο χρήστης μας μπορεί να εξυπηρετήσει με τα προϊόντα που έχει(αν δηλαδή ο χρήστης έχει μόνο καρότα, τότε σε αυτό τον πίνακα θα του εμφανίζονται μόνο οι δημοπρασίες που ζητούν καρότα).

Left Screenshot: Edit Company (My Products tab)

Field	Value
Name	qwertyuiop4
Description	qwerty
AFM	qwerty
D.O.Y.	qwerty
Occupation	qwerty
Address	qwerty
Address area	qwerty
Street number	qwerty
Postal Code	12345
Company phone number	qwerty
Personal phone number	qwerty

Right Screenshot: Edit Company (Purchase requests tab)

Field	Value
Address	qwerty
Address area	qwerty
Street number	qwerty
Postal Code	12345
Company phone number	qwerty
Personal phone number	qwerty

εικόνες 19-20: σελίδα εταιρίας χρήση

Αν ο χρήστης κατευθυνθεί στην σελίδα τροποποίησης της εταιρίας, τότε εκτός από τα στοιχεία της εταιρίας του, ο χρήστης μπορεί να δει και να τροποποιήσει την επιθυμητή τοποθεσία συναλλαγών που έχει βάλει.

Edit CompanyPurchase requestsHi, stefanos

Exchange LocationClose

AliasSaxi

Street nameOnavrou

Street number9

AreaTavros

Zip code17778

Telephone number697746994

Submit

Edit CompanyPurchase requestsHi, stefanos

Addressαε

Address areaαε

Street numberαεααα

Postal Code12345

Company phone numberαεα

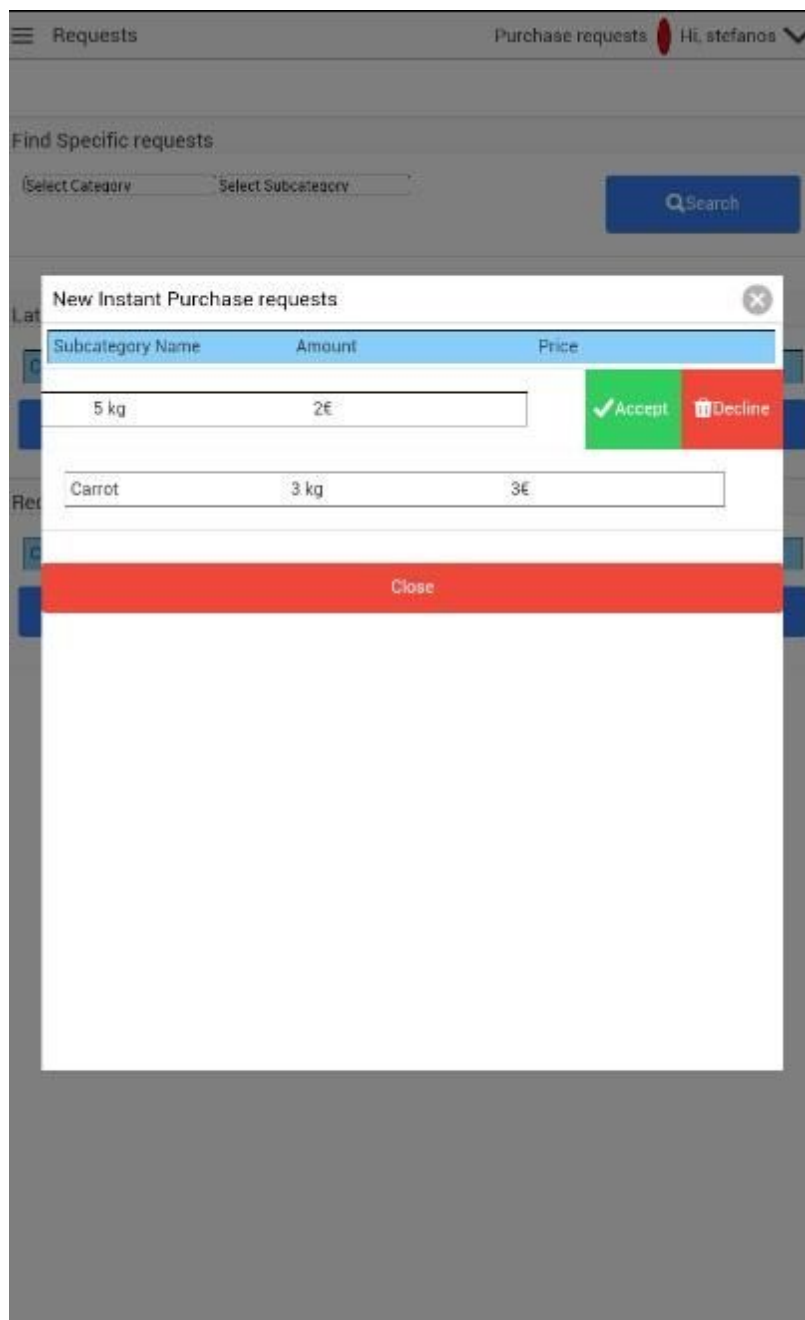
Personal phone numberαεα

Submit

Your exchange location

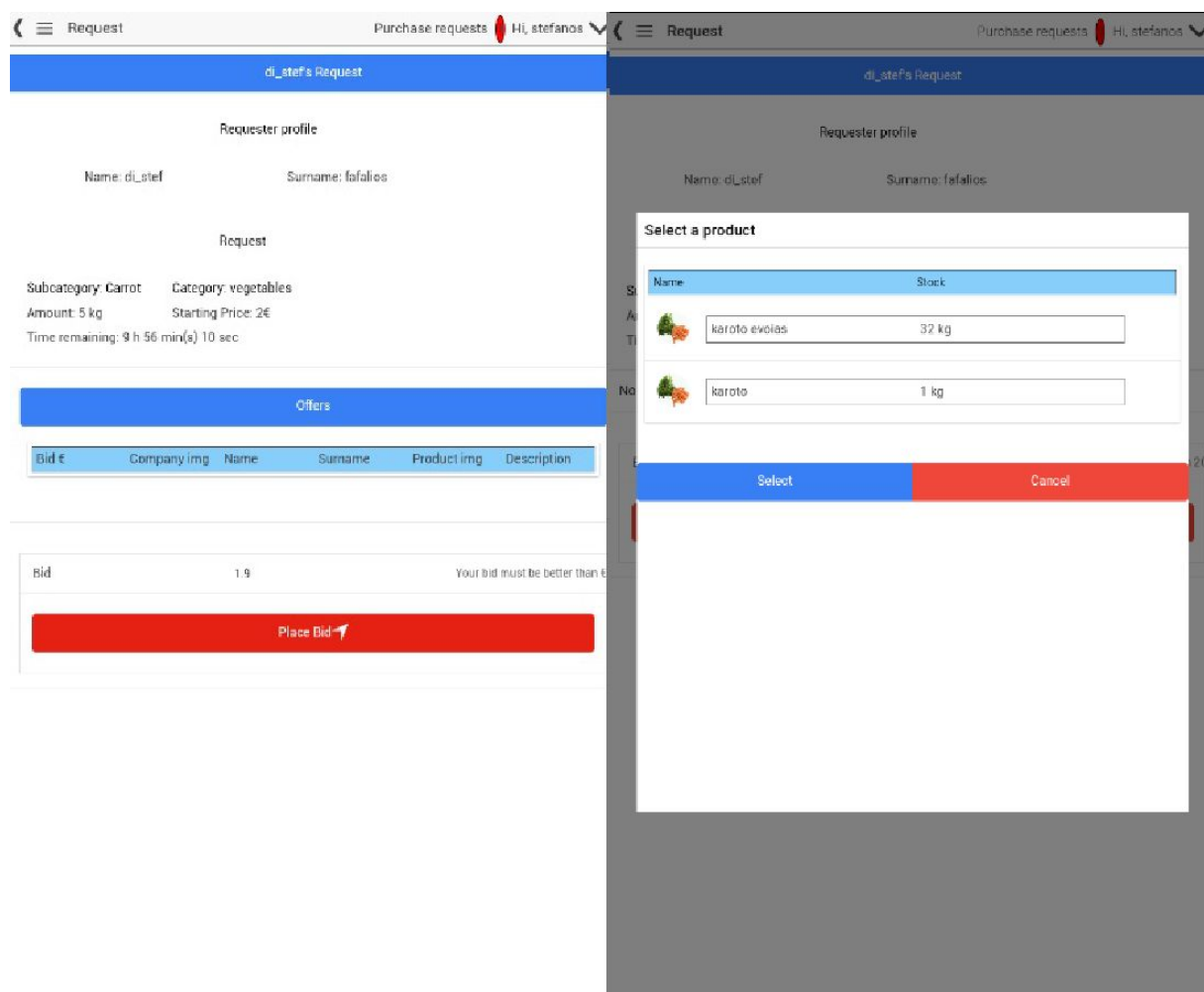


εικόνες 21-22: αλλαγή τοποθεσίας και προβολή της στο χάρτη



εικόνα 23: άμεσες αγορές

Σε περίπτωση που ο χρήστης πατήσει πάνω στο κόκκινο εικονίδιο, τότε μπορεί να δει όλα τις αιτήσεις για άμεση αγορά που του έχουν γίνει και δεν έχουν απαντηθεί. Ο χρήστης στην συνέχεια σύροντας προς τα αριστερά την αίτηση που επιθυμεί, μπορεί να δει τις επιλογές και να αποφασίσει αν θα την αποδεχθεί ή αν θα την απορρίψει.



εικόνα 24-25: τοποθέτηση προσφοράς και επιλογή προσφερόμενου προϊόντος

Request

Purchase requests

Hi, stefanos

di_stef's Request

Requester profile

Name: di_stef

Surname: fafalios

Request

Subcategory: Carrot

Category: vegetables

Amount: 5 kg

Starting Price: 2€

Time remaining: 9 h 39 min(s) 11 sec

Offers

Bid €	Company img	Name	Surname	Product img	Description
1.9		stefanos	fafalios		poly kalo karoto

Bid

Your bid must be better than 1.9€

Place Bid

εικόνα 26: εμφάνιση καινούριας προσφοράς

Όταν ένας χρήστης επιλέγει να κάνει μια προσφορά σε κάποια δημοπρασία, τότε ένα popup του εμφανίζεται, με τα προϊόντα που μπορεί να προσφέρει για την δημοπρασία, και ο χρήστης μπορεί να διαλέξει ποιο θα προσφέρει. Όταν ο χρήστης πατήσει “select”, τότε η προσφορά τοποθετείται στην δημοπρασία. Στην τρίτη οθόνη μπορούμε να δούμε ότι η προσφορά έγινε με επιτυχία, αφού εμφανίστηκε στον πίνακα με τις προσφορές που έχουν γίνει για την συγκεκριμένη δημοπρασία.

Left

Home

My offers

My products

Conversations

Conversation with Eve Naive

Conversation with stefanos fafali

Conversation with eva zafiri

Conversation with 123 123

Conversation with dl stef fafali

Conversation with George Vasalos

Conversation with dl stef fafali

Conversation with 123 123

Requests

Find Specific requests

Select Category

Select Subcategory

Latest Requests

Category	Subcategory	Best Offer	Time
vegetables	banana		1 h
vegetables	Carrot	1.9	9 h
fruit	apple		10 h

Requests suitable for you

Category	Subcategory	Best Offer	Time
vegetables	Carrot	1.9	9 h

Conversation with fafali

Purchase requests

Hi, stefanos

Request

Carrot 5.5Kg Price: 0.5

Created at: 2016-8-17 17:7:44

You wrote:

When and where would you like to do the transaction?

at: 2016-9-20 18:23:14

You wrote:

Hi

at: 2016-9-20 18:22:58

write your message...

Send

εικόνες 27-28: διαθέσιμες συνομιλίες και αποστολή μηνύματος

Όταν μία δημοπρασία λήξει, ή όταν μια αίτηση για άμεση προσφορά γίνει αποδεκτή, τότε δημιουργείται μια συνομιλία ανάμεσα στους δύο χρήστες. Αν ο χρήστης κατευθυνθεί στο πλαϊνό μενού τότε εκτός από τις υπόλοιπες επιλογές του, μπορεί και να δει τις συνομιλίες του ονομαστικά. Όταν πατήσει μια από αυτές τις συνομιλίες, τότε θα κατευθυνθεί στην σελίδα της συγκεκριμένης συνομιλίας και θα μπορεί να δει το ιστορικό της συνομιλίας για την συγκεκριμένη δημοπρασία και να στείλει καινούριο μήνυμα.

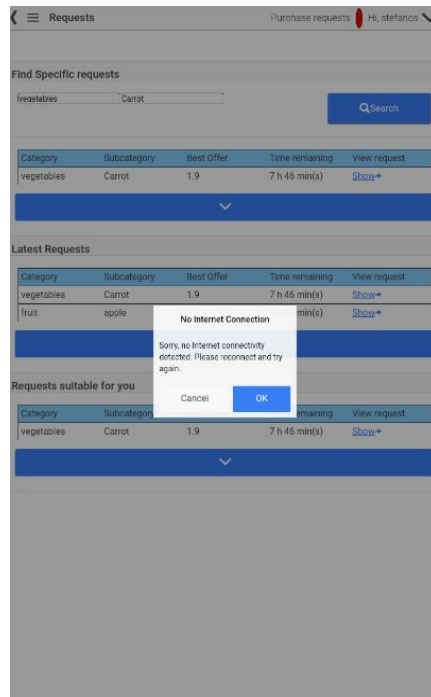
My Offers				Purchase requests	Hi, stefanos
	karoto	Amount: 5.5 kg	Bid: 0.5€	Expired	✓
	karoto evoias	Amount: 5 kg	Bid: 0.6€	Expired	✓
	karoto evoias	Amount: 5 kg	Bid: 1.9€	9 h 1 min(s)	⚙
	karoto evoias	Amount: 14 kg	Bid: 6€	Expired	✓
	Portokalia Ilias	Amount: 5 kg	Bid: 600€	Expired	✓

εικόνα 29: ιστορικό προσφορών

Αν ο χρήστης κατευθυνθεί στο My offers, μπορεί να δει το ιστορικό από τις καλύτερες προσφορές που έχει κάνει για κάθε δημοπρασία που έχει πάρει μέρος (αν δηλαδή κάνει πολλές προσφορές για κάποια δημοπρασία, τότε στο ιστορικό θα εμφανίζεται μόνο η προσφορά με την καλύτερη τιμή για εκείνη την δημοπρασία).

Με πράσινο οι προσφορές που έχει κάνει και έχουν κερδίσει την δημοπρασία. Με κίτρινο οι προσφορές οι οποίες αφορούν δημοπρασία η οποία δεν έχει λήξει ακόμα, ενώ με κόκκινο οι προσφορές που έχουν γίνει για κάποια δημοπρασία που έχει λήξει αλλά δεν κέρδισαν.

Σε περίπτωση που ο χρήστης πατήσει την εικόνα του προϊόντος, τότε να ανακατευθυνθεί στην σελίδα συγκεκριμένου προϊόντος, ενώ αν πατήσει στα στοιχεία κάποιας δημοπρασίας, τότε θα ανακατευθυνθεί στην σελίδα της συγκεκριμένης δημοπρασίας.



εικόνα 30 : μήνυμα σε περίπτωση που δεν υπάρχει internet

Σε περίπτωση που η σύνδεση στο Internet διακοπεί ή δεν υπάρχει εξ' αρχής, τότε εμφανίζεται το κατάλληλο μήνυμα στον χρήστη που τον ενημερώνει για την κατάσταση και τον προτρέπει να επανασυνδεθεί.

5.2.2 Controllers

- **appCtrl**

Ο βασικός controller. Σε αυτό τον controller υπάρχουν οι συναρτήσεις που είναι υπεύθυνες για το login του χρήστη, το logout, το register, την διαχείριση των άμεσων αιτήσεων και την προβολή του δεξιού μενού επιλογών.

- **ProfileCtrl**

Αυτός ο controller είναι υπεύθυνος για όλες τις λειτουργίες που έχουν να κάνουν με το προφίλ του χρήστη, όπως προβολή προφίλ, αλλαγές στο προφίλ, αλλαγή password.

- **CompanyCtrl**

Αυτός ο controller είναι υπεύθυνος για όλες τις λειτουργίες που έχουν να κάνουν με το προφίλ εταιρίας του χρήστη, και την επιθυμητή τοποθεσία συναλλαγών του χρήστη.

- **ProductsCtrl**

Αυτός ο controller είναι υπεύθυνος για όλες τις λειτουργίες που έχουν να κάνουν με την σελίδα προϊόντων του χρήστη(προβολή όλων των προϊόντων, δημιουργία καινούριου προϊόντος, τροποποίηση προϊόντος, διαγραφή προϊόντος).

- **RequestsCtrl**

Αυτός ο controller είναι υπεύθυνος για την προβολή των δημοπρασιών στην αρχική σελίδα, στέλνει τα δεδομένα ταξινόμησης και αναζήτησης στον σέρβερ και προβάλλει τον υπολειπόμενο χρόνο σε

μορφή που μπορεί να διαβάσει άνθρωπος. Επίσης κάθε λεπτό ανανεώνει τα στοιχεία των δημοπρασιών.

- **RequestCtrl**

Αυτός ο controller είναι υπεύθυνος για την προβολή των στοιχείων μιας δημοπρασίας στον χρήστη, τις προσφορές που έγιναν καθώς και την δημιουργία νέας προσφοράς.

- **OffersCtrl**

Αυτός ο controller είναι υπεύθυνος για την προβολή του ιστορικού προσφορών στο χρήστη, τις ανακατευθύνσεις στην σωστή δημοπρασία ή στο σωστό προϊόν καθώς και για την προβολή του υπολειπόμενου χρόνου σε μορφή που μπορεί να διαβάσει άνθρωπος.

- **ConversationCtrl**

Αυτός ο controller είναι υπεύθυνος για την συνομιλία ανάμεσα στον παραγωγό και στον πελάτη. Προβάλλει στοιχεία για την δημοπρασία στην οποία αναφέρεται η συνομιλία, και το ιστορικό των μηνυμάτων. Επίσης μετατρέπει τις ημερομηνίες από τον ώρα της βάσης δεδομένων στην τοπική ώρα του χρήστη. (δηλαδή αν η βάση δεδομένων είναι στο GMT+ 0 και ο χρήστης βρίσκεται στο GMT + 2, τότε στις ημερομηνίες που προέρχονται από την βάση δεδομένων θα αφαιρούνται 2 ώρες πριν να προβληθούν στον χρήστη).

- **.run**

Στο αρχείο με τους controller μου, εκτός από τους controllers τους ίδιους, υπάρχει και ένα κομμάτι κώδικα που εκτελείται με το που θα γίνει load το αρχείο αυτό. Μέσα σε αυτό το κομμάτι αρχικοποιώ διάφορες συναρτήσεις στις οποίες θέλω να μπορούν να έχουν πρόσβαση όλοι οι controllers. Όπως: Η συνάρτηση που εμφανίζει το κατάλληλο μήνυμα σε περίπτωση που κάτι πάει στραβά, Η συνάρτηση που ελέγχει αν υπάρχει σύνδεση στο internet και εμφανίζει κατάλληλο μήνυμα σε περίπτωση που δεν υπάρχει, Η συνάρτηση που είναι υπεύθυνη για την προβολή των συνομιλιών στο πλαϊνό μενού, η συνάρτηση τερματισμού της εφαρμογής, η συνάρτηση αποθήκευσης της διαφοράς ώρας ανάμεσα στην βάση και τον χρήστη, η συνάρτηση αποθήκευσης του session και η συνάρτηση διαγραφής του, συναρτήσεις ελέγχου φορμών, και λοιπές καθολικές μεταβλητές.

5.2.3 Plugins

Τα cordova plugins που χρησιμοποίησα για αυτή την εργασία είναι τα εξής:

- **compat**

Αυτό το plugin καθιστά εφικτή την συμβατότητα με προηγούμενες εκδόσεις της Cordova

- **compat**

Αυτό το plugin έχει ως στόχο να εξασφαλίσει ότι το console.log () είναι όσο χρήσιμο είναι δυνατό να είναι. Προσθέτει πρόσθετη λειτουργικότητα για το iOS, το Ubuntu, Windows Phone 8 και Windows.

- **device**

Αυτό το plugin ορίζει ένα καθολικό αντικείμενο συσκευής, το οποίο περιγράφει το υλικό και το λογισμικό της συσκευής. Παρά το γεγονός ότι το αντικείμενο είναι στην καθολική εμβέλεια, δεν είναι διαθέσιμο μέχρι μετά το deviceready event.

- **File και file-transfer**

Αυτά τα plugin επιτρέπουν το ανέβασμα, τη λήψη και τη διαχείριση αρχείων.

- **geolocation**

Αυτό το plugin παρέχει πληροφορίες σχετικά με τη θέση της συσκευής, όπως γεωγραφικό πλάτος και μήκος. Κοινές πηγές πληροφοριών τοποθεσίας περιλαμβάνουν Global Positioning System (GPS) και η τοποθεσία που προκύπτει από τα σήματα του δικτύου, όπως η διεύθυνση IP, RFID, WiFi και Bluetooth διευθύνσεις MAC, και τα αναγνωριστικά κυττάρων GSM / CDMA.

- **Network Information**

Παρέχει πληροφορίες σχετικά με την κινητή(cellular) και Wi-Fi σύνδεση της συσκευής, και αν η συσκευή έχει σύνδεση στο Internet.

- **ImagePicker**

Plugin για επιλογή εικόνων - για iOS και Android 4.0 και πάνω.

- **Splashscreen**

Αυτό το plugin χρειάζεται για την οθόνη εκκίνησης. Εμφανίζει και κρύβει μια οθόνη splash κατά την εκκίνηση της εφαρμογής.

- **Statusbar**

Το Status Bar παρέχει κάποιες λειτουργίες για την προσαρμογή του iOS και του Android Status Bar.

- **Whitelist**

Αυτό το plugin χρησιμοποιείται για την πλοήγηση στο webview της εφαρμογής σε Cordova 4.0

- **Keyboard**

Το plugin Keyboard παρέχει λειτουργίες που κάνουν την αλληλεπίδραση με το πληκτρολόγιο πιο εύκολη και ενεργοποιεί γεγονότα(events) για να δείξει ότι το πληκτρολόγιο θα εμφανιστεί/κρυφτεί.

Κεφάλαιο 6

Συμπεράσματα και πιθανές επεκτάσεις

Με το πέρας της εργασίας έβγαλα το εξής συμπέρασμα : Αν και ήταν δύσκολο να στηθεί το περιβάλλον εργασίας (κυρίως λόγω ασυμβατοτήτων ανάμεσα σε versions των τεχνολογιών που χρησιμοποίησα), η χρήση framework μείωσε σημαντικά τον χρόνο δημιουργίας μιας τέτοιας εφαρμογής.

Πιθανές επεκτάσεις: Η εργασία με λίγο επιπλέον κώδικα θα μπορούσε να εξελιχθεί σε κάτι πιο γενικό. Θα μπορεί δηλαδή να εξελιχθεί σε μια εφαρμογή που αφορά κάθε είδος αγαθών και όχι μόνο αγροτικά προϊόντα. Επίσης η εφαρμογή να διαχειρίζεται τις αγορές και τις πληρωμές των χρηστών μέσω Paypal. Επιπλέον θα μπορούσε να υπάρχει ένα σύστημα επαλήθευσης από χρήστη σε χρήστη με reviews που θα μπορούσε να κάνει ένας χρήστης σε κάποιον άλλο.

Βιβλιογραφία

- [1] «Front-End» [Ηλεκτρονικό]. Available: <https://bitbucket.org/farm-IT/mobileapp-provider>
- [2] «Back-End» [Ηλεκτρονικό]. Available: <https://bitbucket.org/farm-IT/backend>
- [3] «Δημοπρασία» [Ηλεκτρονικό]. Available: <http://ti-einai.gr/dimoprasia/>
- [4] «Πλειστηριασμούς» [Ηλεκτρονικό]. Available: <http://ti-einai.gr/pleistiriasmos/>
- [5] «Είδη δημοπρασίας» [Ηλεκτρονικό]. Available: <http://www.realtor.org/auction/types-of-auctions>
- [6] «Auction Types» [Ηλεκτρονικό]. Available: <http://www.realtor.org/auction/types-of-auctions>
- [7] Gartner Inc Available: <http://www.gartner.com/technology/home.jsp>
- [8] «Relational Database» [Ηλεκτρονικό]. Available: <http://searchsqlserver.techtarget.com/definition/relational-database>
- [9] «MySQL» [Ηλεκτρονικό]. Available: <http://www.mysql.com/>
- [10] «Api» [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/Application_programming_interface
- [11] «MVC framework» [Ηλεκτρονικό]. Available: <http://whatis.techtarget.com/definition/model-view-controller-MVC>
- [12] «SDK» [Ηλεκτρονικό]. Available: <http://www.webopedia.com/TERM/S/SDK.html>
- [13] «Virtual Machines» [Ηλεκτρονικό]. Available: <http://searchservvirtualization.techtarget.com/definition/virtual-machine>
- [14] «Oracle VirtualBox» [Ηλεκτρονικό]. Available: <https://www.virtualbox.org/>
- [15] «Ubuntu server» [Ηλεκτρονικό]. Available: <http://www.ubuntu.com/download/server>
- [16] «ExpanDrive» [Ηλεκτρονικό]. Available: <https://www.expandrive.com/>
- [17] «MySQL Workbench» [Ηλεκτρονικό]. Available: <http://www.mysql.com/products/workbench/>
- [18] «Brackets» [Ηλεκτρονικό]. Available: <http://brackets.io/>
- [19] «Codeigniter» [Ηλεκτρονικό]. Available: <https://www.codeigniter.com>
- [20] «AngularJs» [Ηλεκτρονικό]. Available: <https://angularjs.org/>
- [21] «Node.js» [Ηλεκτρονικό]. Available: <https://nodejs.org>

[22] «Apache Cordova» [Ηλεκτρονικό]. Available: <https://cordova.apache.org/>

[23] «Ionic» [Ηλεκτρονικό]. Available: <http://ionicframework.com/>

Πίνακας εικόνων

Εικόνα 1: Native vs Hybrid.....	16
Εικόνα 2: Json string	18
Εικόνα 3: Json array	18
Εικόνα 4: Json values	18
εικόνα 5: Json escapes	19
εικόνα 6: Json numbers	19
Εικόνα 7: MVC	23
εικόνα 8: επικοινωνία front-end/back-end	29
εικόνα 9: παράδειγμα χρήσης virtualBox	31
εικόνα 10: ubuntu 14.04 LTS Server	32
εικόνα 11: virtual server	32
εικόνα 12: server networking	33
εικόνα 13: expandrive variables	34
εικόνα 14: παράδειγμα χρήσης expandrive	35
εικόνα 15: παράδειγμα χρήσης brackets	36
εικόνα 16: δομή αρχείων codeigniter	37
εικόνα 17: EER diagram	44
εικόνα 18: αρχική σελίδα εφαρμογής	58
εικόνες 19-20: σελίδα εταιρίας χρήστη	59
εικόνες 21-22: αλλαγή τοποθεσίας και προβολή της στο χάρτη	60
εικόνα 23: άμεσες αγορές	61
εικόνα 24-25: τοποθέτηση προσφοράς και επιλογή προσφερόμενου προϊόντος	62
εικόνα 26: εμφάνιση καινούριας προσφοράς	63
εικόνες 27-28: διαθέσιμες συνομιλίες και αποστολή μηνύματος	64
εικόνα 29: ιστορικό προσφορών	65
εικόνα 30 : μήνυμα σε περίπτωση που δεν υπάρχει internet.....	66