ΥΛΟΠΟΙΗΣΗ ΥΒΡΙΔΙΚΗΣ ΕΦΑΡΜΟΓΗΣ ΓΙΑ ΚΙΝΗΤΕΣ ΣΥΣΚΕΥΕΣ

F FOR FITNESS

ΟΝΤΙΣΕ ΤΣΑΚΑΙ

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΤΕ

Πίνακας περιεχομένων

Ευχαριστίες

Η παρούσα εργασία αποτελεί διπλωματική εργασία στα πλαίσια του προπτυχιακού προγράμματος του τμήματος Μηχανικών Πληροφορικής ΤΕ, της Σχολής Τεχνολογικών Εφαρμογών του ΤΕΙ Ηπείρου. Πριν την παρουσίαση της παρούσας διπλωματικής εργασίας, αισθάνομαι την υποχρέωση να ευχαριστήσω ορισμένους από τους ανθρώπους που γνώρισα, συνεργάστηκα μαζί τους και συνέβαλαν στην υλοποίησή της. Πρώτο από όλους θέλω να ευχαριστήσω τον επιβλέποντα καθηγητή της διπλωματικής εργασίας, επίκουρο Καθηγητή Ιωάννη Τσούλο για την πολύτιμη καθοδήγηση του, την εμπιστοσύνη και εκτίμηση που μου έδειξε. Τις ευχαριστίες μου εκφράζω και στους καθηγητές Αλέξανδρο Τζάλλα (Καθηγητής Εφαρμογών) και Νικόλαο Γιαννακέα (Πανεπιστημιακό Υπότροφο) που δέχτηκαν να είναι μέλη της τριμελής επιτροπής αξιολόγησης της διπλωματικής εργασίας. Τέλος, ιδιαίτερες ευχαριστίες θέλω να απευθύνω στο Νικόλαο Κατερτσίδη (CEO της Εταιρείας UNIVEYE) για την καθοριστική του συμβολή, ο οποίος στάθηκε σημαντικός αρωγός στην προσπάθειά μου και με υποστήριξε σε κάθε φάση της εργασίας μου (καθώς σημαντικό μέρος της εργασίας μου αποτέλεσε αντικείμενο της πρακτική άσκησης).

Περίληψη

Η παρούσα διπλωματική εργασία έχει ως αντικείμενο την καταγραφή των διαθέσιμων αρχιτεκτονικών προσεγγίσεων που αφορούν την υλοποίηση εφαρμογών για κινητές συσκευές. Αρχικά παρουσιάζονται τα πλεονεκτήματα και μειονεκτήματα των διαφορετικών αρχιτεκτονικών προσεγγίσεων, με παράλληλη καταγραφή των βασικών τεχνολογιών που χρησιμοποιούνται στην εκάστοτε προσέγγιση. Επιπρόσθετα, στα πλαίσια της εργασίας αναπτύχθηκε μια νέα και πρωτοποριακή υβριδική εφαρμογή για κινητές συσκευές με βασικό αντικείμενο την καταγραφή των σημαντικότερων καθημερινών δραστηριοτήτων των χρηστών όπως το περπάτημα, η ακριβής μέτρηση της κατανάλωσης αλκοόλ και τροφίμων. Χαρακτηριστικά αυτής της εφαρμογής αποτελούν:

- η βάση δεδομένων με σχεδόν όλες τις διαθέσιμες τροφές και τα αλκοολούχα ποτά που μπορεί να επιλέξει ο χρήστης και ο υπολογισμός του συνόλου των θερμίδων του.
- Η καταγραφή και η μέτρηση της κίνησης του χρήστη (περπάτημα) μέσω της χρήσης αισθητήρα ποδομέτρου.
- Ο υπολογισμός του δείκτη βάρους του χρήστη ο οποίος προσδιορίζει κατά πόσο το βάρος του είναι φυσιολογικό.
- Η δημιουργία καθημερινού, εβδομαδιαίου αλλά και μηνιαίου ιστορικού για την κίνηση, την κατανάλωση τροφίμων και αλκοόλ του χρήστη.

Στο σημείο αυτό θα πρέπει να σημειωθεί ότι συμπεριλαμβάνεται DVD με τον πηγαίο κώδικα της εφαρμογής.

Κεφάλαιο 1. Εισαγωγή

Στην σημερινή εποχή η ιδιοκτησία μια κινητής συσκευής θεωρείται δεδομένη και αναγκαία με συνέπεια οι πωλήσεις να αυξάνονται εκθετικά με τον χρόνο. Το φαινόμενο οφείλεται στις γιγάντιες δυνατότητες των smartphone που σχεδόν αντικαθιστούν την ανάγκη προσωπικού υπολογιστή λόγου της φορητότητας, των τεχνικών χαρακτηριστικών, του μεγάλου εύρος των εφαρμογών και την φύση του ανθρώπου που τον παρακινεί να επικοινωνεί και να δικτυώνεται.

1.1 Ιστορική αναδρομή κινητών συσκευών

Η ονομασία κινητή συσκευή προκύπτει από τα τεχνικά χαρακτηριστικά που συνδικάζουν την ανεπάρκεια καλωδιακής σύνδεσης και την παροχή από δίκτυο κινητής τηλεφωνίας. Τα κινητά τηλέφωνα χρησιμοποιούν την τεχνολογία cell και εκπέμπουν σε υψηλές συχνότητες για την αποφυγή των παρεμβολών με ηλεκτρονικές συσκευές και με ραδιοφωνικούς σταθμούς. Η πρώτη υλοποίηση κινητού τηλέφωνου ήταν το Motorola DynaTAC 8000X, δημιουργήθηκε στις 3 Απρίλιου 1973 από την πολυεθνική εταιρεία τηλεπικοινωνίων Motorola και προχώρησε σε μαζική παραγωγή του κινητού το 1983 (Εικόνα 1). Το τηλέφωνο είχε τα έξεις χαρακτηριστικά διαστάσεις 228.6x127x44.4mm και βάρος 1.1KG, πρόσφερε μέχρι και 30 λεπτά ομιλίας ή 6 ώρες αναμονής μέχρι την επόμενη επαναφόρτιση που διαρκούσε 10 ώρες, η συσκευή είχε κόστος $3995. Τα χαρακτηριστικά αυτά έκανα την δύσκολη την απόκτηση και φορητότητα των κινητών και έτσι οι μεγαλύτερες εταιρείες τηλεπικοινωνίων επιθυμούσαν να καλύψουν τις ανάγκες αυτές και προσθέτοντας δυνατότητες για γραπτή επικοινωνία (SMS, MMS) [1].

Εικόνα 1. Motorola DynaTAC 8000X είναι το πρώτο κινητό τηλέφωνο που κυκλοφόρησε το 3/4/1973

Η ραγδαία εξέλιξη των κινητών οφείλετε στον ανταγωνισμό των εταιριών δηλαδή στην ζήτηση του προϊόντος, φέρνοντας καινοτόμες υλοποιήσεις στο τεχνικό, λειτουργικό τομέα Έτσι η Ericsson το 1973 δημιούργησε το μοντέλο GS 88 Penelope που

χαρακτηρίστηκε ως έξυπνο κινητό (Smartphone).Η διάκριση των έξυπνων και των απλών κινητών δεν είναι σαφής και δεν υπάρχει επίσημος ορισμός οποίος να διευκρινίζεις τις διάφορες μεταξύ τους, αλλά μια από τις σημαντικές διάφορες είναι ότι έχουν προηγμένες διεπαφές προγραμματισμού εφαρμογών (API) που κάνουν την εγκατάσταση τρίτων εφαρμογών και σύνδεση με το hardware εύκολη. Επίσης το hardware στα smartphones είναι περίπλοκο και παρέχει περισσότερες δυνατότητες. Με το πέρασμα του χρόνου τρόπος σκέψεις των ανθρώπων για την επικοινωνία και για την διασκέδαση (παιχνίδια)είχε αλλοιωθεί και αυξήθηκε η ζήτηση των συσκευών, με αποτέλεσμα να πιεστούν οι εταιρίες και περισσότερες καινοτόμες ιδέες στο τεχνικό κομμάτι και στο λογισμικό να προσχωρήσουν στην υλοποίηση. Η βασικές υλοποιήσεις ήταν η δημιουργία οθονών με μεγαλύτερη ευκρίνεια, διάσταση ώστε να ενσωματώσουν και να υποστηρίξουν την προσπέλαση των διαδικτυακών ιστοσελίδων που είχαν μεγαλύτερες απαιτήσεις ισχύος (Hardware) [2].

Η πλατφόρμα λειτουργικού σύστημα Android [3] για κινητές συσκευές παρουσιάστηκε για πρώτη φορά στις 5 Νοεμβρίου 2007, βασιζόμενο σε πυρήνα Linux σχεδιασμένο κυρίως για Tablet,Phablet και Smartphone τα οποία θα αναλυθούν λεπτομερειακά στο κεφάλαιο 2. Το Android υλοποιήθηκε αρχικά από την Android inc. η οποία αγοράστηκε από την Google και έπειτα από την Open Handset Alliance. Το λογισμικό αυτό επιτρέπει στους προγραμματιστές να υλοποιήσουν εφαρμογές στην αντικειμενοστραφής γλωσσά προγραμματισμού Java, η συγκεκριμένη OOP (Object-oriented programming) γλώσσα είναι πολύ γνωστή για τις εφαρμογές υπολογιστών μέχρι και υλοποίησης δυναμικών ιστοσελίδων (Java EE [4]).

1.2 Αντικείμενο της διπλωματικής εργασίας

Το αντικείμενο της παρούσας διπλωματικής εργασίας είναι η υλοποίηση μιας υβριδικής εφαρμογής(Hybrid Application) στηριζόμενο στο Framework Apache Cordova με θέμα την καθοδήγηση και την παρακολούθηση της υγείας. Η αίτια που χρησιμοποιήθηκε το Cordova είναι ότι υποστηρίζεται από όλες τις πλατφόρμες λογισμικών. Πιο συγκεκριμένα, το αντικείμενο την εφαρμογής καλύπτει τα έξεις θέματα, την μέτρηση των βημάτων του χρήστη και υπολογισμός καύσης θερμίδων. Επίσης θα προσφέρει στον χρήστη την δυνατότητα να επιλεγεί τα προϊόντα που κατανάλωσε ανά γεύμα έπειτα θα του παρουσιάζονται όλα τα συστατικά ανά ποσότητα. Μια άλλη θέση που καλύπτεται είναι η πλήρης η παρακολούθηση και έλεγχος της προσλήψεις νερού και αλκοολούχων ποτών. Στα θέματα αυτά παρέχονται στον χρήστη η ικανότητα να παρακολουθεί το πλήρης ιστορικό μεμονωμένα. Επίσης θα παρέχεται η αποστολή όλων τον δεδομένων σε έναν κεντρικό διακομιστή σε των χρηστών ανά λογαριασμό.

Κεφάλαιο 2. Ανάπτυξη εφαρμογών για κινητές συσκευές

Η επιτυχία ενός μεταγενέστερου μοντέλου κινητικής συσκευής είναι ο συνδυασμός τριών κατηγοριών πλατφόρμας λογισμικού, εξαρτήματα υλικού και υποστήριξη εφαρμογών σύμφωνα με της δυο παραπάνω κατηγορίες. Οι δυνατότητες που προσφέρουν συνήθως είναι μετρητής βημάτων (pedometer), υψηλές ανάλυσης κάμερες, βιντεοκάμερες, μονάδα πλοηγείς (GPS) για εντοπισμό της συσκευής. Καθώς οι μεταγενέστεροι browsers των κινητών προσφέρουν δυνατότητες όπως είναι η αναγνώριση ιστοσελίδων για κινητά και υπολογιστών, υποστήριξη στην αποστολή δεδομένων P2P (Peer to Peer) μέσω των WebShockets. Μια τεχνολογία που βασίζεται στην αποστολή P2P δεδομένων είναι το WebRTC [5] (Real Time communication),που έχει υλοποιήσεις όπως αποστολή βίντεο, Instant messaging(γραπτά κείμενα), αποστολή αρχείου χωρίς την παρέμβαση διακομιστή(Server) και απολυτή κρυπτογράφηση των δεδομένων μέσω HTTPS πρωτοκόλλου.

2.1 Είδη κινητών συσκευών

Η υλοποίηση μια εφαρμογής είναι μια περιπλοκή διαδικασία και διαφέρει στο τεχνικό κομμάτι από την πλατφόρμα λογισμικού που υποστηρίζεται. Κάθε λογισμικό επιτρέπει στους προγραμματιστές εφαρμογών να συγγράψουν κώδικα σε μια ξεχωριστή αντικειμενοστραφούς γλώσσα, ωστόσο οι εφαρμογές έχουν ένα κοινό ότι προορίζονται για εγκατάσταση στις κινητές συσκευές. Εν συσκευές αυτές διαφέρουν από το μέγεθος, τεχνικά χαρακτηριστικά, δυνατότητες λογισμικού και σύμφωνα με αυτά τα χαρακτηριστικά κατηγοριοποιούνται σε Smartphone, Tablet, Phablet.

2.1.1 Smartphone

Στην εποχή μας, όταν αναφερόμαστε στην λέξη κινητό εννοούμε Smartphone γιατί έχουν κυριαρχήσει πλέον την αγορά σε σύγκριση με τα απλά κινητά (Εικόνα 2).

Εικόνα 2. Smartphone συσκευή

Τα κινητά τηλεφωνά (Smartphone) έχουν την ικανότητα να στέλνουν και να λαμβάνουν σήμα μέσω υψηλών συχνοτήτων, η επικοινωνία γίνεται στο κοντινό κέντρο κινητής τηλεφωνίας. Αρχικά τα κινητά είχαν τις λειτουργίες ενός απλού κινητού(πχ κλήσης, SMS, MMS) και ενός προσωπικού ψηφιακού βοηθού (PDA) (Εικόνα 3), ενώ τώρα τα κινητά τηλεφωνά χαρακτηρίζονται ως ένα πολυμήχανά αφού σε αρκετές περιπτώσεις έχουν περισσότερες δυνατότητες και από έναν υπολογιστή. Το PDA είναι μια συσκευή που προσφέρει την δυνατότητα σύνδεσης στο διαδίκτυο, αντί για πληκτρολόγιο χρησιμοποιείται ένας ειδικός στύλος, το λειτουργικό σύστημα προσφέρει ευκολία στον χρήστη με το ιδικό μενού που είναι λίστες. Η κυρίως χρήση των PDA είναι για επιχειρήσεις αφού βοηθάει τον πελάτη-υπάλληλο να καταχώρηση, ανάκτηση πληροφορίες με ένα πάτημα [6].

Εικόνα 3. Personal Digital Assistant

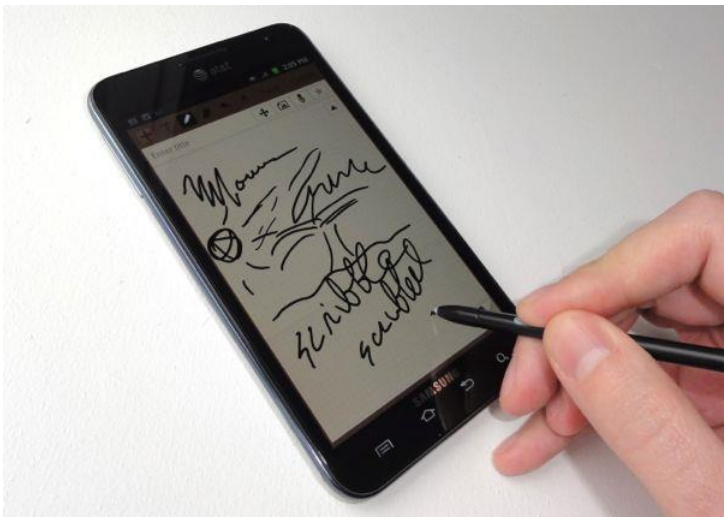
2.1.2 Tablet

Tablet υπολογιστής συχνά αναφερόμενο και ως απλά Tablet (Εικόνα 4), είναι μια επίπεδη λεπτή κινητή συσκευή με μεγάλη οθόνη αφής που συνήθως ξεπερνάει τις 7-inch. Πιο συγκεκριμένα τα Tablet έχουν τις ιδίες ομοιότητες με ένα Smartphone εκτός από την αδυναμία σύνεσης σε κέντρο κινητής τηλεφωνίας μέσω κεραίας εκπομπής συχνοτήτων, δηλαδή έλλειψη δυνατότητα υποστήριξης εφαρμογών για επικοινωνία στο πρότυπο αυτό. Η χρήση των Tablet γίνεται μέσω εικονικού πληκτρολογίου και πάτημα στην οθόνη αφής αλλά υποστηρίζουν εξωτερικά πληκτρολόγια, ποντίκι ακόμα και trackpad. Οι χρήστες συνήθως τα χρησιμοποιούν για την ψυχαγωγία όπως για παιχνίδια, πλοήγηση στον διαδίκτυο για τη ανάγνωση ηλεκτρονικών βιβλίων λόγο της τεράστιας οθόνης και της επαναφορτιζόμενης μπαταρίας που έχει μεγαλύτερη διάρκεια από ένα μέσω κινητό τηλέφωνο [7].

Εικόνα 2. Tablet υπολογιστής

2.1.3 Phablet

Ο ορός Phablet προέρχεται από τον συνδυασμό των λέξεων Phone, Tablet και προσφέρει τις δυνατότητες και των δυο συσκευών δηλαδή σύνδεση σε κέντρο κινητής τηλεφωνίας και εξωτερικές συσκευές (εξαρτήματά). Προσφέρονται σε διαστάσεις οθόνης από 5.11 μέχρι 6.99 inches και ενσωματωμένο στυλό που επιτρέπει την συγγραφή με το πάτημα στην οθόνη. Τα Phablet έγιναν γνωστά το 2011 όταν η Samsung κυκλοφόρησε σε μαζική παραγωγή τη συσκευή Galaxy Note (Εικόνα 5). Ένα χρόνο αργότερα πολλές εταιρίες κατασκευής Smartphone ένταξαν στην παραγωγή τους Phablet, έτσι σημειώθηκε το ποσό των 25.6 εκατομμύριων πωλήσεων. Λόγο του μεγέθους της αγοράς το 2012 ονομάστηκε «Η χρονιά των Phablet» από την ειδησεογραφικό πρακτορείο Reuters, δυο χρονιά αργότερα οι πωλήσεις των Phablet εκτοξευτήκαν στα ύψη σημειώνοντας ρεκόρ αφού ξεπέρασαν τις πωλήσεις λάπτοπ και desktop [8].



Εικόνα 3. Το Samsung Galaxy Note είναι το πρώτο Phablet που βγήκε σε μαζική παραγωγή το 2012 και έκανε ρεκόρ πωλήσεων

2.2 Είδη εφαρμογών για κινητές συσκευές

Το πρώτο βήμα για τις επιχειρήσεις εφαρμογών συσκευών είναι να παρθεί η πιο σημαντική απόφαση που θα έχει τεράστιο αντίκτυπο στο αποτέλεσμα του έργου, είναι η τεχνική με την οποία θα υλοποιήσουν την εφαρμογή (Mobile Application), συμφώνα με τις κατηγορίες Native, hybrid και Ιστοσελίδα. Η απόφαση για καλύτερη τεχνική είναι αδύνατη για κάθε έργο γιατί υπάρχουν πολλές παράμετροι την καθορίζουν όπως ο προϋπολογισμός που διατίθεται, το χρονοδιάγραμμα του έργου, την πελατεία που προσπαθεί να καλύψει και την λειτουργικότητα της εφαρμογής. Η λειτουργικότητα της εφαρμογής έχει τον μεγαλύτερο ρολό στης παραπάνω παραμέτρους παράδειγμα για την κατηγορία προτύπου ιστοσελίδα έχει ένα μεγάλο μειονέκτημα. Το μειονέκτημα αυτό είναι ανεπάρκεια πρόσβαση στις APIs (Application Programming interface) που αναφέρονται στους αισθητήρες, χαρακτηριστικά λογισμικού και σε εξαρτήματα του κινητού. Μια πρωτοφανής σκέψη είναι ότι δεν μπορεί να δημιουργηθεί εφαρμογή Web(ιστοσελίδας) που να έχει πρόσβαση στον αποθηκευτικό χώρο και να έχεις μεγάλες απαιτήσεις γραφικών.

2.2.1 Μητρικές Εφαρμογές (Native Applications)

Μητρικές εφαρμογές θεωρούνται οι εφαρμογές που έχουν συγγραφεί στην αντικειμενοστραφούς γλώσσα στην οποία προδιαγράφεται από την πλατφόρμα του λογισμικού της συσκευής. Για την πλατφόρμα Android προδιαγράφεται σαν μητρική γλώσσα η Java όπως εμφανίζεται στον παρακάτω πίνακα. Οι μητρικές εφαρμογές έχουν δυαδικά εκτελέσιμα αρχεία τα οποία αναγνωρίζονται από την εικονική μηχανή (Virtual Machine) συσκευή και αποθηκεύονται τοπικά. Ο χρήστης μπορεί να εκτελέσει την εγκατάσταση και να αποδεχτεί τους ορούς και τα δικαιώματα που απαιτεί η εφαρμογή, ωστόσο μπορεί να εκτελεστεί από την εταιρία που έχει δημιουργήσει την κινητή συσκευή. Ο πιο διάσημος και εύκολος τρόπος λήψεις μητρικών εφαρμογών από τα app stores, όπως Google Market- Play Store (Android), App Store Market (IOS), BlackBerry App World (BlackBerry OS) και Windows Phone Marketplace (Windows Phone) (Πίνακας 1). Σε περίπτωσή που δεν εκτελεστεί η εγκατάσταση από τα app stores, η εφαρμογή δεν ενημερώνεται ποτέ και η εταιρία που έχει δημιουργήσει το λογισμικό δεν καλύπτει σε θέματα ασφάλειας γιατί η εφαρμογή μπορεί να έχει αλλοιωθεί από τέταρτες εταιρίες ή πρόσωπα.

Πίνακας 1. Συνοπτικές πληροφορίες των τεσσάρων πιο διάσημων λειτουργικών συστημάτων για κινητές συσκευές

	Apple IOS	**Android**	**BlackBerry OS**	**Windows Phone**
Γλώσσα	Objective-C, Swift και για Cross Compiled C#, C++, JavaScript	Java και για Cross Compiled C#, C++, JavaScript	Java	C#, VB.NET και για Cross Compiled C++, JavaScript
IDE	Xcode	Android Studio, NetBeans	BB Java Eclipse Plug-in	Visual Studio, Windows Phone development tools
Μορφή Πακέτου	.app	.apk	.cod	.xap

App Store	Apple App Store-Google Market	Google Store	BlackBerry App World	Windows Phone Marketplace

Μόλις η εφαρμογή εγκατασταθεί στην συσκευή ο χρήστης, μπορεί να την εκτελέσει από το μενού του λογισμικού σαν όλες τις άλλες εφαρμογές ή υπηρεσίες που η συσκευή προσφέρει. Κατά τη αρχικοποίηση η μητρική εφαρμογή διασυνδέεται άμεσα με το λειτουργικό σύστημα της συσκευής χωρίς την παρέμβαση ενδιάμεσου παραγώγου. Επίσης επιτρέπεται η πρόσβαση σε όλα τα APIs (Διεπαφές προγραμματισμού εφαρμογών) που είναι διαθέσιμα από το λειτουργικό σύστημα της συσκευής. Οι διεπαφές αυτές είναι συναρτήσεις, πρωτοκολλά και εργαλεία που διευκολύνουν τους προγραμματιστές εφαρμογών να αλληλοεπιδράσουν με τα εξαρτήματα της συσκευής ή ακόμα και να εμφανίσουν απλά γραφικά στην οθόνη.

Η υλοποίηση μιας μητρικής εφαρμογής έχει την απαίτηση από τους προγραμματιστές, να συγγράψουν προγραμματιστικό κώδικα στην απαιτούμενη γλώσσα του λογισμικού και να δημιουργήσουν επιπλέον πόρους για την εφαρμογή όπως εικόνες, ήχους και διαφορά αρχεία (Resources).Οι πόροι αυτοί θα προδιαγράφουν κάποιες λειτουργίες του λογισμικού (theme, xml names) ή της διεπαφή του χρήστη (πχ xml Layout για Android συσκευές) σύμφωνα με το λογισμικό αυτό. Ο προμηθευτής του λογισμικού προσφέρει εργαλεία ονομαζόμενα SDK (Software Development Kit) που τα εκμεταλλεύονται τα IDEs (Integrated Development Environment) για την συγγραφή της εφαρμογής. Με αποτέλεσμα την δημιουργία δυαδικό εκτελέσιμου αρχείου το οποίο θα εμπεριέχει και τα resources αρχεία, ώστε να είναι έτοιμο να κυκλοφορήσει στην αγορά.

Οι δυο κατηγορίες API που διαθέσιμα από το λειτουργό σύστημα είναι, το χαμηλού επιπέδου και το υψηλού επιπέδου API. Το χαμηλό επιπέδου API έχει άμεση επαφή με υλικό κομμάτι της συσκευής και επιτρέπει την πρόσβαση από τις εφαρμογές για να αλληλεπιδράσεις άμεσα με οθόνη, πληκτρολόγιο , εμφάνιση των γραφικών, σύνδεση στο διαδίκτυο, λήψη πληροφοριών από το GPS και το μικρόφωνο. Ενώ τα υψηλού επιπέδου API έχουν άμεση επαφή με το λογισμικό και τις εφαρμογές που έχουν δημιουργηθεί από τα χαμηλού επιπέδου API,όπως πλοήγηση στο διαδίκτυο, διαχείριση του ημερολογίου, επαφών, τηλεφωνικών κλήσεων και αποστολή SMS [9].

2.2.2 Διαδικτυακές εφαρμογές (Mobile-Web Applications) και ιστοσελίδες για κινητές συσκευές (Web Site for Mobiles)

Οι σύγχρονες κινητές συσκευές εμπεριέχουν ισχυρά προγράμματα περιήγησης, οι οποίοι υποστηρίζουν τις περισσότερες καινούριες δυνατότητες HTML (HyperText Markup Language), CSS3(Cascading Style Sheets 3) και JavaScript. Οι τεχνολογίες αυτές χρησιμοποιούνται για την εμφάνιση πλουσίων χρωμάτων στις ιστοσελίδες και λειτουργίες που εξηγούν αναλυτικά στο Κεφάλαιο 4 για την υλοποίηση της υβριδικής εφαρμογής μας.

Μερικά πλεονέκτημα που έχουν οι διαδικτυακές εφαρμογές σε σύγκριση με τις ιστοσελίδες για κινητές συσκευές είναι ότι περιλαμβάνουν προηγμένα στοιχεία περιβάλλον χρήστη δηλαδή Touch-friendly και διασπαστικό περιβάλλον στον χρήστη, πρόσβαση εκτός σύνδεσης των αρχείων αφού βρίσκονται τοπικά στην συσκευή

(Πίνακας 2). Ένα άλλο προτέρημα είναι η εκτέλεση της εφαρμογής από το εγκαταστημένο εικονίδιο σαν τις μητρικές εφαρμογές.

Πίνακας 2. Διαδικτυακές εφαρμογές (Mobile Web Applications) σε σύγκριση με ιστοσελίδες για κινητές συσκευές (Web Site for Mobiles)

Χαρακτηριστικά	Mobile Web Apps	Mobile Websites
Εργαλεία και γνώση	Γραμμένα εντελώς σε HTML, CSS και JavaScript	Γραμμένα εντελώς σε HTML, CSS και JavaScript
Εκτελέσει	Εκτέλεση από το εγκατεστημένο εικονίδιο σαν μητρική εφαρμογή	Πλοήγηση από στον περιηγητή στην συγκεκριμένα σελίδα URL
Εμπειρία χρήστη	Touch-friendly, διασπαστικό περιβάλλον στον χρήστη	Πλοήγηση περιβάλλον χρήστη μεταξύ σελίδων με στατικά δεδομένα
Επίδοσης	Τα αρχεία βρίσκονται τοπικά οπότε καλύτερη ανταπόκριση και προσπέλαση offline	Τα αρχεία βρίσκονται διακομιστή και πρόσβαση γίνεται μόνο με σύνδεση στο Internet

Υπάρχουν τεράστιες υλοποιήσεις που προσφέρουν διαδικτυακές εφαρμογές και οι ιστοσελίδες για κινητές συσκευές που αναγνωρίζουν ότι την συσκευή του χρήστη. Οι κινητές διαδικτυακές εφαρμογές είναι μια αυστηρά υποσχόμενη μόδα. Εκμεταλλευτήκαν αυτή την μόδα και για διευκολύνουν τους προγραμμάτισες να υλοποιήσουν διεπαφές χρήστη στην μεριά τους πελάτη(Browser), έχουν δημιουργηθεί πολλές εργαλειοθήκες σε JavaScript. Οι πιο γνώστες εργαλειοθήκες- βιβλιοθήκες είναι Bootstrap mobile, dojox.mobile, Sencha Touch and jQuery Mobile οι οποίες παράγουν διεπαφές χρήστη που είναι παροιμίες με τις μητρικές εφαρμογές και είναι αδύνατον στο γυμνό διακριθούν οι διαφορές. Και οι δυο κατηγόριες εκτελούνται αποκλείστηκα στον προεπιλεγμένο μηχανή περιηγητή του κινητού και υποστηρίζει τις τελευταίες έκδοσης JavaScript,HTML,CSS που είναι διαθέσιμες στην έκδοση αυτή.

Ένα τεράστιο ωφέλημα των διαδικτυακών εφαρμογών είναι το χαμηλό κόστος λογού ότι ο ίδιος κώδικας τρέχει σε όλες τις πλατφόρμες λογισμικού («Write once, run anywhere»). Οι περισσότερες συσκευές χρησιμοποιούν την ιδιά μηχανή εμφάνισης στον περιηγητή τον WebKit. Ο WebKit έχει ενσωματωθεί από τους περιηγητές Google Chrome, Mozilla Firefox και Safari, αυτό συνεπάγεται ότι υποστηρίζεται από όλες τις κινητές συσκευές ανεξάρτητος μοντέλου και πλατφόρμας λογισμικού.

Σε αντίθεση με της μητρικές εφαρμογές ,οι οποίες είναι ανεξάρτητες εκτελέσεις που επικοινωνούν απευθείας με το λογισμικό, ενώ οι διαδικτυακές εφαρμογές εκτελούνται από τον περιηγητή. Όμως ο περιηγητής είναι από μονός του μια μητρική εφαρμογή που επικοινωνεί άμεσα με τα APIs του λειτουργικού. Μόνο μερικά από αυτά τα APIs είναι εκτεθειμένα στις διαδικτυακές εφαρμογές σε σύγκριση με τις μητρικές εφαρμογές εχουν πρόσβαση σε όλα τα APIs του συστήματος. Αυτό αναμένεται να αλλάξει στον μέλλον όταν αναπτυχθεί περισσότερο η HTML για δυνατότητες που δεν είναι τώρα εφικτές στον WebKit.

2.2.3 Υβριδικές Εφαρμογές (Hybrid Applications)

Ο υβριδικός σχεδιασμός συνδικάζει τον μητρικό προγραμματισμό και την διαδικτυακή τεχνολογία (ιστοσελίδα). Χρησιμοποιώντας αυτή την προσέγγιση, οι προγραμματιστές συγγράφουν κώδικα σε web(HTML, JavaScript, CSS) που εκτελείται το ίδιο σε όλες τις πλατφόρμες λειτουργικού και παρέχεται άμεση πρόσβαση σε μητρικά APIs.. Δηλαδή η εφαρμογή μπορεί να επικαλεστεί πληροφορίες, δεδομένα από τα APIs την στιγμή που τα έχει ανάγκη και όχι μόνο στη αρχή δημιουργίας της Οθόνη(Intent).

Ο κώδικας που είναι εγγεγραμμένος στη μητρική γλωσσά χρησιμοποιεί τα APIs του λειτουργικού συστήματος για την δημιουργία της μηχανής εμφάνισης HTML η οποία χρησιμεύει ως γέφυρα αναμεσά στον περιηγητή και στα APIs της συσκευής. Η γέφυρα αυτή επιτρέπει στην υβριδική προσέγγιση πλήρης πρόσβαση στα χαρακτηριστικά, εξαρτήματα της μοντέρνας συσκευής. Ο κώδικας αυτός μπορεί να γραφτεί από τους προγραμματιστές από την αρχή (scratch) είτε να χρησιμοποιηθεί βιβλιοθήκη που παρέχει διεπαφές σε JavaScript που καλούν τα APIs και παρέχουν όλες τις δυνατότητες του συστήματος όπως το Apache Cordova [10] (Εικόνα 4). Το Apache Cordova είναι μια βιβλιοθήκη-Framework που επιτρέπει την πρόσβαση σε όλα APIs του λειτουργικού συστήματος με την βοήθεια τον επεκτάσεων που επεκτείνονται σε αυτό και η επικοινωνία με τις μητρικές κλάσης γίνεται μέσω το JavaScript αρχείων που εκτελούν την αντίστοιχη μέθοδο. Το Framework αυτό θα χρησιμοποιεί για τη δημιουργία της υβριδικής εφαρμογής στην παρούσα διπλωματική και θα αναλυθεί λεπτομερειακά στο επόμενο κεφάλαιο.

Εικόνα 4. Το λογότυπο του Apache Cordova

Η διαδικτυακή υλοποίηση της εφαρμογής μπορεί να είναι αποτελείται από HTML, JavaScript, CSS αρχεία που έχουν αποθηκευτεί σε έναν απομακρυσμένο διακομιστή είτε τοπικά στην συσκευή ή βρίσκονται στο κώδικα της εφαρμογής. Όταν ο κώδικας της εφαρμογής βρίσκεται σε απομακρυσμένο διακομιστή επιτρέπει στους προγραμματιστές να τον τροποποιούνε και να προσθέσουν αρχεία χωρίς να χρειάζεται να υποβάλουν την καινούρια έκδοση στα τοπικά Market των συσκευών και να περιμένουν να εγκριθεί η έκδοση αυτή. Αλλά δυστυχώς η προσέγγιση αυτή δεν επιτρέπει την δυνατότητα offline πρόσβασης στα περιεχόμενα της εφαρμογής όταν δεν υπάρχει σύνδεση στο διαδίκτυο. Από την άλλη μεριά, όταν περιέχει η εφαρμογή στο

κώδικα της τα HTML εκτελείται γρηγορότερα γιατί δεν αναμένει να ληφθούν τα αρχεία αλλά δεν υποστηρίζεται απομακρυσμένη αναβάθμιση στην τεχνική αυτή. Η καλύτερη λύση είναι ο συνδυασμός των δυο πρότυπων, δηλαδή όταν εκτελεστεί η εφαρμογή και υπάρχει σύνδεση στο διαδίκτυο η εφαρμογή να ελέγχει στον απομακρυσμένο διακομιστή εάν υπάρχει νεότερη έκδοση και να την αποθηκεύσει τοπικά. Όταν ληφθεί η έκδοση αυτή θα εκτελείτε κανονικά σαν να ήταν ο κώδικας HTML μέσα στα πακέτα της εφαρμογής, αφού τα αρχεία αυτά φορτώνονται στην ενδιάμεση μνήμη (Cache) για μεγαλύτερες αποδόσεις (ταχύτητα εκτέλεσης) [11] .

2.2.4 Μεταγλωττισμένες εφαρμογές (Cross Compiled Applications)

Την παρούσα χρονική στιγμή υπάρχουν εργαλεία που επιτρέπουν στους προγραμματιστές κινητών συσκευών να δημιουργήσουν μητρικές εφαρμογές με γλώσσες προγραμματισμού που δεν τις αναγνωρίζει το λειτουργικό σύστημα. Τα εργαλεία αυτά έχουν ως στόχο να δημιουργήσουν μια μητρική εφαρμογή που χρησιμοποιεί τα όλα τρέχων GUIs, APIs της συσκευής που είναι διαθέσιμα από το λειτουργικό.

Η προσέγγιση αυτή είναι όμοια με την υβριδική, δηλαδή υλοποιούνται εφαρμογές που είναι συμβατές με όλες τις συσκευές ανεξάρτητος λογισμικού. Αλλά όπως ειπώθηκε παραπάνω, η προσέγγιση αυτή για να λειτουργήσει επιτρέπει στα JS αρχεία να επικοινωνούν με μητρικές καλαΐσεις και η διεπαφή χρήστη είναι απλά ένα Web View το οποίο προσπαθεί να μιμηθεί τα GUI του συστήματος σε αντίθεση με την μεταγλωττισμένη προσέγγιση που υλοποιεί μητρικές κλάσης του συστήματος χρησιμοποιώντας όλες τις δυνατότητες και χαρακτηριστικά της συσκευής. Για να κατασκευαστεί μια μεταγλωττισμένη εφαρμογή θα πρέπει να γίνει η εκμάθηση κάποιου ειδικού Framework που μεταφράζει τον κώδικα που συγγράφεται στον compiler χρησιμοποιώντας το SDK(System Development Kit) του framework, μετατρέπεται(μεταγλώττιση) σε κλάσης είτε σε αρχεία XML σύμφωνα με το XMLVM [12] compiler. Η μετατροπή αυτή έχει ως αποτέλεσμα αρχεία με διαφοροποίηση ανάλογα το λειτουργικό συστήματα που προσφέρεται , για να μπορέσει το VM(Virtual Machine) να τα αναγνωρίσει και να τα εκτελέσει.

Τα εργαλεία που κυριαρχούν στη αγορά είναι το Xamarin [13], Titanium appcelerator [14]. Σύμφωνα με τα λεγόμενα του Xamarin είναι το μόνο IDE που επιτρέπει την δημιουργία μητρικών εφαρμογών(μητρικές κλάσης) Android, IOS, Windows στο Visual Studio [15] και επιτρέπει στους προγραμματιστές να συγγράψουν τον κώδικα της εφαρμογής σε C#. Επιπλέον το Xamarin είναι εγκατεστημένο στο Visual Studio και έχει επεκτάσεις που υποστηρίζουν για την δημιουργία των εκλέξιμων αρχείων, debugging (εύρεση λαθών) μέσω συσκευών προσομοιωτών που εκτελούν την εφαρμογή στην αντίστοιχη πλατφόρμα λογισμικού. Επίσης το πακέτο που προκύπτει από την μεταγλώττιση είναι σε μητρική προγραμματιστή γλωσσά ώστε να δίνει την δυνατότητα στον προγραμματιστή να τροποποίηση τον πηγαίο κώδικα. Σε αντίθεση το appcelerator επιτρέπει στους προγραμματιστές να συγγράψουν κώδικα σε JavaScript και η μεταγλώττιση έχει ως τελικό στάδιο τη δημιουργία εκλέξιμων αρχείων με μη προσπελάσιμο και κατανοήσιμο κώδικα αφού υλοποιείται με τις προδιάγραφες του Titanium SDK.

2.2.5 Σύγκριση αρχιτεκτονικών προσεγγίσεων

Παρατηρείται ότι οι μητρικές εφαρμογές έχουν μεγαλύτερο κόστος υλοποίησης και συντήρησης αφού κυκλοφορήσουν στην αγορά αλλά έχουν άμεση πρόσβαση στα APIs που είναι διαθέσιμα από το συγκεκριμένοι λειτουργικό και μπορούν να επεξεργαστούν απαιτητικά γραφικά και δεδομένα (Πίνακας 3). Ενώ οι διαδικτυακές εφαρμογές είναι εύκολες στην υλοποίηση, συντήρηση και προσφέρουν δυνατότητα εκτέλεσης σε όλες τις συσκευές ανεξάρτητος λειτουργού συστήματος, αλλά εκτελούν άμεσα τα διαθέσιμα APIs του WebKit και έχοντας ως διεπαφή χρήστη το WebView του περίτμητη που τις καθιστά πολύ αργές ,δηλαδή δεν μπορούν να υποστηρίξουν εφαρμογές με απαιτητικά γραφικά. Από την άλλη, η υβριδική προσέγγιση προσφέρει χαμηλό κόστος υλοποίησης, συντριβής και εκτελείται σε όλες τις πλατφόρμες λογισμικού αλλά απαιτεί την εκμάθηση κάποιου Framework που κάνει διαθέσιμα τα APIs του συστήματος μέσω τον μητρικών κλάσεων, επίσης υπάρχει η ανάγκη ενός επαγγελματία προγραμματιστή JavaScript που κατανοεί εις βάθους την αρχιτεκτονική του HTML5, HTTP ώστε για αυξήσει την ταχύτητα εκτέλεσης των αρχείων .html. Η τελευταία κατηγορία, η cross-compiled έχει τα ιδιά προτερήματα με την υβριδική προσέγγιση αλλά έχει καλύτερες αποδόσεις, το μειονέκτημα είναι ότι χρειάζεται την εκμάθηση κάποιου συγκείμενου IDE και τον τρόπο λειτουργίας compiler. Τα IDEs αυτά είναι πολύ ακριβά στην απόκτηση τους και κάνουν τον προγραμματιστή απολυτά εξαρτημένο σε αυτά και σε περίπτωση που ένα χαρακτηριστικό δεν δουλεύει σωστά, η διόρθωση του είναι πολύ δύσκολη και ο χρόνος εξέλιξης είναι τεράστιος.

Πίνακας 3. Σύγκριση όλων τον αρχιτεκτονικών προσεγγίσεων μεταξύ τους.

Χαρακτηριστικά	**Μητρικές εφαρμογές**	**Διαδικτυακές εφαρμογές**	**Υβριδικές εφαρμογές**	**Μεταγλωττισμένες εφαρμογές**
Γλώσσα προγραμματισμού	Μητρική	HTML, JavaScript, CSS	Μητρική, HTML, JavaScript,	JavaScript, C++, C#
Συντήρηση και μεταφορά σε άλλες πλατφόρμες λογισμικού	Δύσκολη	Πολύ εύκολη	Μέτρια	Μέτρια
Πρόσβαση στα APIs και αλλά χαρακτηριστικά της συσκευής	Άμεση και εύκολη	Έμμεση και πρόσβαση σε όσα είναι διαθέσιμα από τον περιηγητή	Έμμεση και πρόσβαση σε όσα είναι διαθέσιμα APIs από το Framework	Άμεση και μέτριας δυσκολίας η πρόσβαση, ανάλογα το SDK του Framework

Ταχύτητα στην διαχείριση υψηλών γραφικών	Πολύ γρήγορη	Μέτρια	Μέτρια	Γρηγόρη
Μόχθος για την εκμάθηση λειτουργίας	Λίγος	Λίγος	Μέτριος	Πολύ δύσκολος
Εξήγηση εφαρμογής	Δύσκολη (από το app stores)	Πολύ εύκολη (από server είτε από app store)	Μέτρια (συνήθως από app store)	Δύσκολη (από το app stores)
Τρόπος εκτέλεσης	Από το app menu	Περιηγητή εάν είναι web-site αλλιώς από το app menu	Από το app menu	Από το app menu

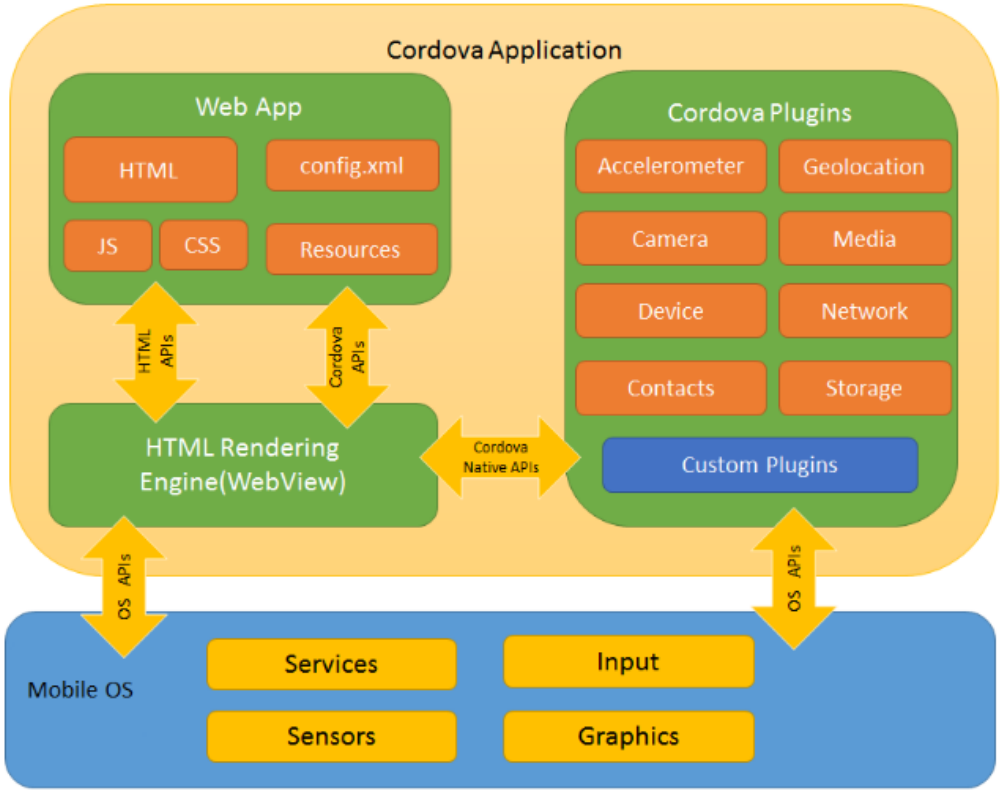
Κεφάλαιο 3. Εργαλεία ανάπτυξης για την δημιουργία Υβριδικής εφαρμογής

Στο κεφάλαιο αυτό θα αναλυθεί πλήρως η αρχιτεκτονική του Apache Cordova και πως μπορεί να αξιοποιηθεί η γνώση αυτή για να υλοποιηθεί μια πλήρως ολοκληρωμένη εφαρμογή που θα προσεγγίζει τις αποδόσεις τις μητρικής. Θα χρησιμοποιεί το Cordova Framework για την δημιουργία του project από τη αρχή προσθέτοντας όλες τις πλατφόρμες λογισμικού και επεκτάσεις για την πλήρης λειτουργικότητα. Επίσης, για την σωστή λειτουργικότητα της εφαρμογής θα εγκατασταθεί το IDE Android Studio [16] συμβατός με το Android SDK για την αποσφαλματοποίηση σε Java και για την δημιουργία του εκτελέσιμου αρχείου. Επίσης θα χρησιμοποιεί το NetBeans IDE για αποσφαλματοποίηση και συγγραφή κώδικα σε HTML, JavaScript, CSS, διότι οι τεχνολογίες αυτές δεν υποστηρίζονται πλήρως από το Android Studio.

3.1 Ανάλυση της αρχιτεκτονικής Framework Apache Cordova

Τα προβλήματα αρχίσαν όταν τα HTML αρχεία δεν είχαν καμία πρόσβαση στα APIs και χαρακτηριστικά των συσκευών. Από την άλλη μεριά, ούτε ο μητρικός κώδικας δεν μπορούσε να επικοινωνήσει με διαδικτυακό , οφείλεται στο περιηγητή που έχει πρόσβαση σε λίγα APIs του συστήματος. Έτσι το 2008 σε ένα iPhoneDevCamp δημιουργήθηκε ένα λογισμικό(Framework) με την ονομασία PhoneGap, υστέρα εξαγοράστηκε από τη Adobe η οποία το έκανε δωρεά στο Apache. Το Apache Software Foundation ήταν η εταιρία που ονόμασε το λογισμικό Cordova και επέκτεινε την λειτουργικότητα του ώστε να υποστηρίζει σχεδόν όλες τις πλατφόρμες λογισμικού και υλοποίησε επεκτάσεις που έχουν την δυνατότητα προσβάσεις σε όλα τα APIs της συσκευής. Το Cordova είναι ένα πολύπλοκο λογισμικό αλλά παρέχονται έγραφα προγραμματιστών και ο πηγαίος κώδικας, ώστε να δίνεται η δυνατότητα στους προγραμματιστής εφαρμογών να τροποποιήσουν τον προγραμματιστικό κώδικα στις ανάγκες της εφαρμογής.

Βασικά το Cordova παράγει μια υβριδική εφαρμογή που περιέχει HTML, JavaScript αρχεία και τον μητρικό κώδικα που αναγνωρίζεται από την πλατφόρμα λογισμικού της συσκευής. Ο μητρικός κώδικάς είναι ο ενδιάμεσος αναμεσά στα APIs του συστήματος και στα JavaScript αρχεία ώστε να προσφέρει την πρόσβαση σε όλες σχεδόν τις δυνατότητες της συσκευής όπως προβάλλεται στην Εικόνα 5 (διάγραμμα αρχιτεκτονικής). Όλες οι διεπαφές χρήστη εμφανίζονται και εκτελούνται από στο Web View του περιηγητή (WebKit engine), όμοια με ιστοσελίδες διαδίκτυού. Επίσης όλες οι συνάρτησης που αφορούν την άμεση λειτουργικότητα της συσκευής είναι εγγεγραμμένες μόνο σε JavaScript, ώστε ο περιηγητής να παράξει το επιθυμητό αποτέλεσμα αφού η προγραμματιστή γλωσσά που αναγνωρίζει είναι η JavaScript.



Εικόνα 5. Η Αρχιτεκτονική του Apache Cordova σε μια υψηλή επιπέδου όψη μιας Cordova εφαρμογής.

Η διαδικασία με την οποία ανταλλάσσονται τα δεδομένα στο Cordova αναμεσά στον μητρικό κώδικα και τον JavaScript είναι μέσω ασύγχρονων XHR (XMLHttpRequest) κλήσεων. Συνεπάγεται ότι το Cordova συμπεριφέρεται σαν κανονικός διακομιστής ιστοσελίδων που ανταλλάσσει πληροφορίες με τους πελάτες. Η συμπεριφορά αυτή είναι η αίτια που οι υβριδικές εφαρμογές είναι πιο λιγότερο αποδοτικές από τις μητρικές γιατί καταναλώνουν πόρους του συστήματος έμμεσα με παρέμβαση διακομιστή. Ο διακομιστής καταναλώνει μεγάλη υπολογιστική ισχύ όταν το μέγεθος δεδομένων είναι τεράστιο και οι κλήσεις που υποστηρίζει είναι πολλές. Πιο συγκεκριμένα, ο τρόπος των ασύγχρονων κλήσεων(Ajax Calls) γίνεται ως εξής [17].

- Το JavaScript(στον κώδικα .js) κάνει μια ασύγχρονη XHR κλήση όταν θέλει να χρησιμοποιήσει ένα API της συσκευής
- Ο διακομιστής κρατάει online την σύνδεση μέχρι τα δεδομένα που ζητούνται να είναι διαθέσιμα
- Ο διακομιστής στέλνει τα δεδομένα πίσω στον πελάτη-browser και κλείνει την σύνδεση για κλήση αυτή
- Ο διακομιστής αρχίζει απευθείας να υπακούει σε μια άλλη XHR κλήση που προέρχεται από τον πελάτη
- Ο πελάτης λαμβάνει XHR απάντηση και επεξεργάζεται ως να αναγνωρίσει αν χρειάζεται να μεταβεί στο αίτημα
- Ο πελάτης κάνει ένα καινούριο ασύγχρονο XHR αίτημα για να λάβει τα δεδομένα που ζητάει

Η βά5ση των υβριδικών εφαρμογών στηρίζονται στην ιδέα των ασύγχρονων κλήσεων για να εναλλάσσουν πληροφορίες με τα μητρικά APIs. Έτσι στο Cordova εχουν στηρίξει όλη την αρχιτεκτονική σε αυτή την ιδέα και για αυτό οι προγραμματιστές του λογισμικού δημιουργήσαν μητρικές κλάσης με σκοπό υπερκαλύπτονται από επεκτάσεις, για να εξυπηρετήσουν τους προγραμματιστές εφαρμογών. Για να δημιουργηθεί μια πλήρως λειτουργική επέκταση στην Cordova, παράδειγμα για πλατφόρμα Android πρέπει να δημιουργεί μια Java κλάση που να επεκτείνει την κλάση «CordovaPlugin» και να υπερκαλύπτει την μέθοδο «execute…» όπως διακρίνεται στην Εικόνα 6. Η μέθοδος execute κάνει ένα ασύγχρονο αίτημα στον διακομιστή κάθε φορά που καλείται από την JavaScript αρχεία. Ωστόσο για να υπάρχει πρόσβαση από τα JavaScript αρχεία της εφαρμογής θα πρέπει να δηλωθεί συνάρτηση με το ίδιο όνομα ακριβώς που δηλώθηκε στο String στην Java κλάση «module.exports.HighToTheSky(success,error,args){…}».

```
public class CordovaExample extends CordovaPlugin {
    @Override
    public boolean execute(String action, JSONArray args, CallbackContext callbackContext) throws JSONException {
        if (action.equals("HighToTheSky")) {
            String message = args.getString(0);
            this.echo(message, callbackContext);
            return true;
        }
        return false;
    }

    private void echo(String message, CallbackContext callbackContext) {
        if (message != null && message.length() > 0) {
            callbackContext.success(message);
        } else {
            callbackContext.error("Expected one non-empty string argument.");
        }
    }
}
```

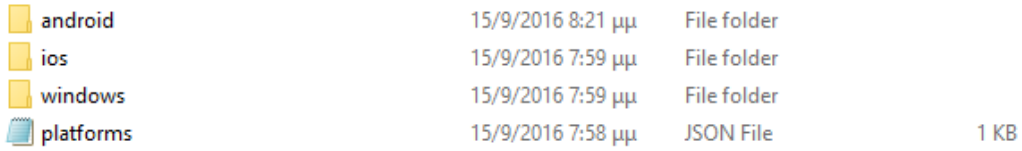
Εικόνα 6. Παράδειγμα Java αρχείου που υπερκαλύπτει την μέθοδο execute της μητρικής κλάσης για την διμιουργία επέκτασης (plugin).

3.2. Δημιουργία Cordova Project

Η δημιουργία των επεκτάσεων (plugins) που αναγνωρίζονται από τις πλατφόρμες λογισμικών των συσκευών είναι πολύπλοκη διαδικασία που απαιτεί γνώση του αντικειμένου, ειδικά στην περίπτωση που συμπεριληφθούν αρχεία για να υπερκαλύψουν τα WebViews και να επικοινωνούν με τοπικό διακομιστή. Το Cordova προσφέρει λειτουργίες, όπως την δημιουργία ενός επεκτάσιμου project που αναγνωρίζει όλες τις πλατφόρμες λογισμικών και φτιάχνει (build) το project ώστε να συνδεθεί με SDK. Σύμφωνα με το πρότυπο αυτό θα δημιουργηθεί το εκτελέσιμο αρχείου που θα εγκατασταθεί και θα εκτελεστεί από την συσκευή. Συγκεκριμένα η λειτουργία της δημιουργίας του Project μέσου του project γίνεται ως έξεις

- Μέσου του Node.js [18] θα εγκατασταθεί το λογισμικού Cordova για την υποστήριξη πρόσβασης στις λειτουργίες του «npm install -g cordova»

- Υστέρα για την δημιουργία του επεκτάσιμου Project που θα περιέχει τα XML αρχεία με χαρακτηριστικά της εφαρμογής (πχ ονομασία πακέτου στην εγκύστωση της εφαρμογής) «cordova create Fitness com.example.Univeye Fitness».
- Για την υποστήριξη των λειτουργικών συστημάτων είναι αναγκαία η επιλογή του φακέλου και η συγγραφή των έξεις εντολών «cordova platform add windows», «cordova platform add android» και «cordova platform add ios» (Εικόνα 7).
- Ως τελικό βήμα είναι η δημιουργία του εκτελέσιμου αρχείου που θα αναγνωρίζεται από την πλατφόρμα λογισμικού που θα εγκατασταθεί και εκτελεστεί. Όμως είναι αναγκαία η εγκατάσταση του συγκεκριμένου SDK λογισμικού πχ για την δημιουργία του .apk στο Android με την εντολή «Cordova build android» είναι αναγκαία η παρουσία του JDK(Java Development Kit) και Android SDK.



Εικόνα 7. Φάκελοι για τις πλατφόρμες λογισμικού που θα αναγνωρίζεται η εφαρμογή

3.2.1 Επιλογή των Plugin

Η επιλογή των επεκτάσεων είναι μια χρονοβόρα διαδικασία και πρέπει να γίνει στην αρχή του Project πριν την συγγραφή κώδικα της εφαρμογή, η αιτία είναι πως όταν στο Cordova γίνεται εγκατάσταση μιας καινούριας επέκτασης διαγραφεί τον κώδικα που έχει συγγραφεί και στην θέση εκχωρεί τον προκαθορισμένο. Επίσης μια άλλη αίτια, είναι ότι χαρακτηρίζεται ως κακή τεχνική λογού ότι ο διακομιστής στέλνει τα δεδομένα οπότε αυτά είναι διαθέσιμα από την συσκευή και σύμφωνα με την ροή του προγράμματος που τα επικαλείται, με σκοπό να αλλάζει όλη η δομή του κώδικα και να μην εκτελείται έτσι όπως θα ήταν επιθυμητό με συνέπεια ότι θα πρέπει να τροποποιηθεί όλος κώδικάς εάν χρειάζεται μια καινούρια λειτουργία που απαιτεί την παρέμβαση του μητρικού κώδικα. Από την Άλλη μεριά, η περιληπτική συγγραφή των ξεχωριστών απαιτήσεων λογισμικού για κάθε διαφορετική οθόνη χρήστη(page με διαφορετικές λειτουργίες) θα εξιχνιάσει το μυστήριο.

Στην πρώτη οθόνη(index.html) ο χρήστης θα εισάγει το όνομα χρήστη και τον κωδικό για να συνδεθεί στην εφαρμογή, η αυθεντικοποίηση θα γίνεται από ένα web service στο διαδίκτυο και υστέρα τα στοιχεία αυτά θα γραφτούν σε ένα αρχείο , για να μην υπάρχει η ανάγκη κάθε φορά που εκτελείται η εφαρμογή να ζητιέται από τον χρήστη η συγγραφή των στοιχείων του. Θα προστεθεί η επέκταση με την εντολή «cordova plugin add https.//github.com/edelworksgithub/SharedPreferences.git» [19] και θα προσφέρει την δυνατότητες όπως δημιουργία, αναγνώριση περιεχόμενων από αρχείο και υψηλές ταχύτητες συναλλαγών πληροφοριών αφού τα SharedPreferences είναι διαθέσιμα όταν εκτελείται η εφαρμογή.

Στην δεύτερη οθόνη(frontPage.html) που είναι η κεντρική, θα δίνει την δυνατότητα στον χρήστη να παρακολουθήσει την κίνηση του όλης της ημέρας σε ένα διάγραμμα. Η μέτρηση των βημάτων γίνεται με τους έξεις τρόπους GPS, με έναν αλγόριθμο που θα μετράει την έξοδό του Accelerometer ή το Pedometer το οποίο έχει την μικρότερη απόκλιση λάθους και το μεγαλύτερο κόστος απόκτησης. Οι τρεις κατηγορίες αυτές εχουν μια ομοιότητα, ότι θα είναι υπηρεσίες που εκτελούνται συνέχεια από τον συσκευή του χρήστη ακόμα και εάν η εφαρμογή είναι κλειστή. Για την εφαρμογή αυτή θα χρησιμοποιηθεί η τεχνολογία Pedometer που έχει και την λιγότερη κατανάλωση μπαταρίας, θα εκτελεστεί η εντολή «cordova plugin add https://github.com/texh/cordova-plugin-stepcounter.git» [20]. Τα δεδομένα που συλλέγει το service τα γραφεί σε SharedPreferences για να υπάρχει η δυνατότητα πρόσβασης.

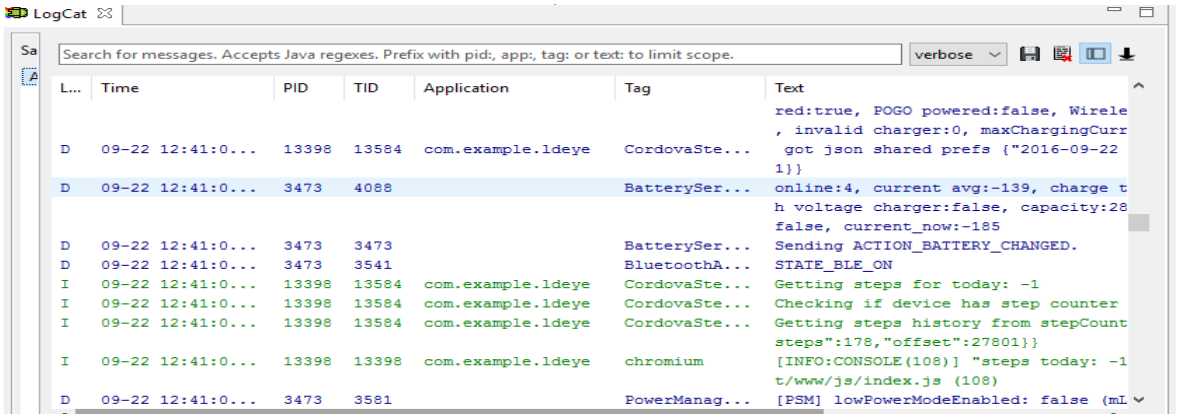
Στη οθόνη της κατανάλωσης αλκοόλ (alcoholMain.html) η εφαρμογή θα δίνει την δυνατότητα στο χρήστη να παρακολουθεί την ποσότητα και της θερμίδες που κατανάλωσε. Η λειτουργία αυτή προϋποθέτει πως τα δεδομένα αυτά είναι αποθηκευμένα, οι μέθοδοι που μπορούν να αποθηκευτούν τοπικά δεδομένα είναι με αρχεία ή με βάση δεδομένων SQLite. Το πρότυπο που θα ακολουθηθεί είναι το SQLite λόγο ότι το μέγεθος των δεδομένων που θα καταχωρούνται καθημερινά είναι μεγάλο και την σχέση εξάρτησης των οντοτήτων. Το πρότυπο αυτό θα ακολουθηθεί και από τις άλλες οθόνες για την καταχώρηση και εμφάνιση των δεδομένων. Η πρόσθεση του επέκτασης γίνεται ως έξεις «cordova plugin add cordova-sqlite-storage»

3.3. Μεταγλωττιστές και Editor για την δημιουργία Υβριδικής εφαρμογής

Όπως ειπώθηκε παραπάνω για την αποσφαλματοποίηση και σωστή λειτουργία της εφαρμογής δημιουργήθηκε η ανάγκη εγκατάσταση λογισμικών από τρίτες εταιρίες. Τα λογισμικά αυτά προσφέρουν μεγάλη ευκολία στην αποσφαλματοποίηση, στον έλεγχος και στην υπογράμμιση των συντακτικών λαθών στην προγραμματιστική γλώσσα Java. Επίσης είναι αναγκαία η εγκατάσταση λογισμικού που θα προσφέρει την δυνατότητα αποσφαλματοποίηση και εκτέλεσης μόνο τις ιστοσελίδας ώστε να τροποποιηθεί ο κώδικας με σκοπό να τονίζει την καλύτερη εμπειρία στον χρήστη.

3.3.1 Μεταγλωττιστές για την δημιουργία Android εφαρμογή

Το Android Studio είναι ένα ολοκληρωμένο προγραμματιστικό περιβάλλον (IDE) που προσφέρεται δωρεάν από την Google για την ανάπτυξη εφαρμογών σε Android. Το λογισμικό αυτό προσφέρει εργαλεία για την εύρεση των συντακτικών λαθών, ακόμα και για την ανάδειξη των λογικών λαθών που σε πολλές περιπτώσεις η εξιχνίαση τους παίρνει ακόμα και μέρες. Προσφέρεται η δυνατότητα δημιουργίας και εκτέλεσης της εφαρμογής με ένα πάτημα. Η εκτέλεση αυτή μπορεί να γίνει σε έναν προσομοιωτή είτε σε μια πραγματική συσκευή Android και τα αποτελέσματα της εκτέλεσης σε εμφανίζονται πραγματικό χρόνο. Οι κατηγορίες των αποτελεσμάτων είναι, μέτρηση μέγεθος χρήσης της εφαρμογής στην RAM, υπολογιστική ισχύει που καταναλώνεται σε όλα τα στάδια των οθονών και μηνύματα αλληλεπίδρασης της συσκευής με την εφαρμογή (Εικόνα 8).

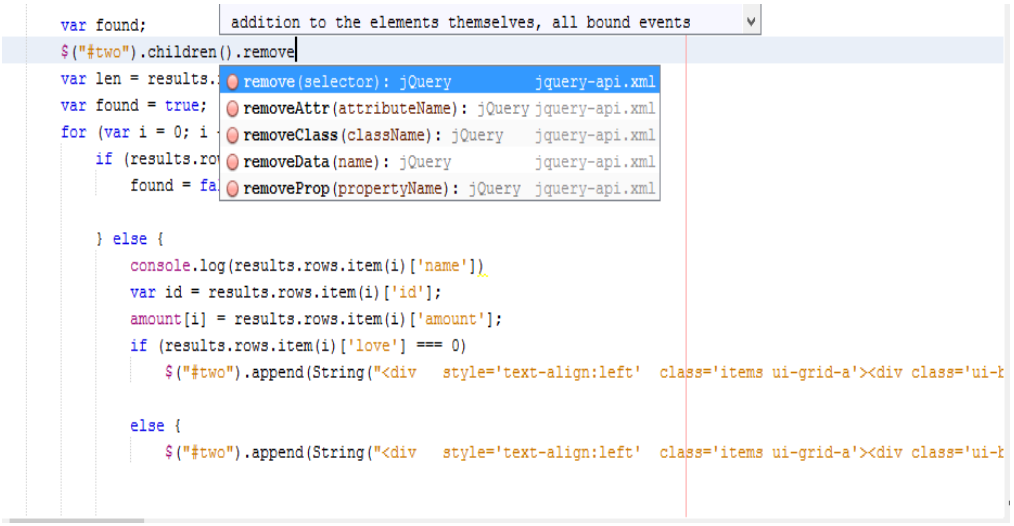


Εικόνα 8: Android studio μνήματα της εφαρμογής και του συστήματος.

3.3.2 Editor για την υλοποίηση ιστοσελίδας και debug

Οι τεχνολογίες διαδίκτυο που χρησιμοποιούνται για την υλοποίηση της εφαρμογής και έχουν άμεση αλληλεπίδραση με τον πελάτη είναι διερμηνευόμενες σε σύγκριση με τις μητρικές που είναι μεταγλωττισμένες. Από τη άλλη μεριά, το Android Studio δεν παρέχει την δυνατότητα αναγνώρισης των τεχνολογιών αυτών και οι επεκτάσεις από τρίτες εταιρίες δεν είναι ολοκληρωμένες. Οπότε υπάρχει η ανάγκη εγκατάστασης ενός λογισμικού που θα προσφέρει ένα διακομιστή για την εκτέλεση και τροποποίηση των αρχείων. Τα αποτελέσματα θα φαίνονται σε πραγματικό χρόνο (on the fly) και θα περιέχει έναν assistant editor που θα αναγνωρίζει τα τις τεχνολογίες αυτές. Το λογισμικό που θα χρησιμοποιεί για την παρούσα διπλωματική είναι το NetBeans IDE για διαδικτυακές τεχνολογίες καλύπτει όλες τις προαπαιτούμενος συνθήκες (Εικόνα 9). Η αναγκαιότητα του λογισμικού είναι τεράστια λογού ότι σε περιπτώσεις που έχει γίνει λάθος στην συγγραφή του κώδικα JavaScript δεν πρόκειται να εκτελεστεί η οθόνη με την οποία το JavaScript αρχείο σχετίζεται.

Το NetBeans παρέχει ορθογραφικό έλεγχο (code-editor assistant) σε πραγματικό χρόνο, όταν γίνεται η συγγραφή κώδικα σε JavaScript,HTML και CSS. Το code-assistant συμπληρώνει εντολές αυτόματα που συγγράφονται και σε περίπτωση λάθους υπογραμμίζεται η εντολή και αναδεικνύεται η σωστή λύση. Επίσης παρέχεται HTTP διακομιστής που εκτελεί τα HTML αρχεία στον περιηγητή ώστε ο προγραμματισμός και διαμόρφωση των διεπαφών χρήστη να είναι πιο διαδραστική αφού η εκτέλεση γίνεται στον ίδιο χρόνο με τον προγραμματισμό. Η τακτική αυτή ακολουθείται παγκόσμιος από τον σύγχρονο προγραμματισμό για να δώσει μεγαλύτερη βαρύτητα στον πελάτη και να διευκολύνει τον προγραμματιστή.

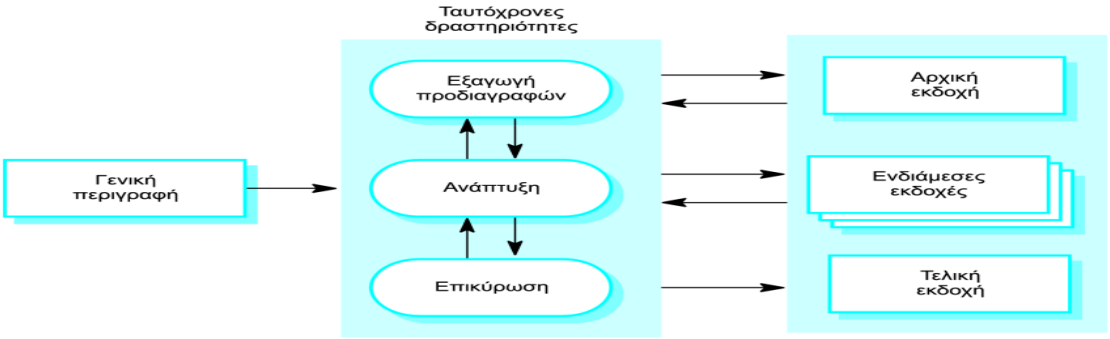


Εικόνα 9:NetBeans Code Assistant στην συγγραφή κώδικα JavaScript

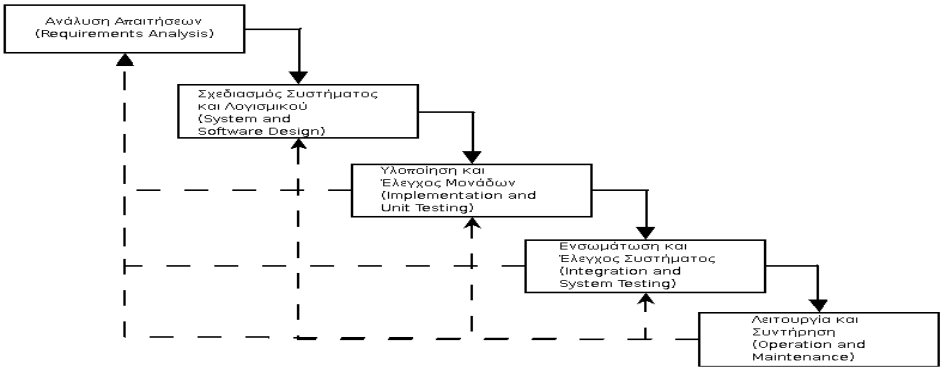
4. Ανάπτυξη Λογισμικού

Στα πλαίσια της διπλωματικής εργασίας αναπτύχθηκε υβριδική εφαρμογή για κινητές συσκευές με σκοπό να βοηθήσει και να παρακινήσει τους χρήστες να κρατηθούν σε φόρμα και να είναι υγιής. Ο χρήστης έχει την δυνατότητα να παρακολουθεί τις θερμίδες που καταναλώνει και καίει καθημερινά, ωστόσο να παρακολουθεί την πρόοδο του στον χρόνο.

Στο κεφάλαιο αυτό θα αναλυθούν λεπτομερειακά τα στάδια της ανάπτυξης λογισμικού. Σημειώνεται πως στην καταγραφή των απαιτήσεων χρησιμοποιείται ο ορός πελάτης, οπού ο πελάτης είναι ένας φανταστικός χαρακτήρας που δημιουργήθηκε για εξυπηρέτηση και διεκπεραίωση του έργου. Η ανάπτυξη λογισμικού είναι μια πολύπλοκη και χρονοβόρα διαδικασία που ακολουθείται για την δημιουργία λογισμικό και εφαρμογών, προσφέρει βέλτιστο αποτέλεσμα και κέρδος των πελατών αλλά και των επιχρίσεων. Η μεθοδολογία που θα ακολουθηθεί στα πλαίσια της διπλωματικής είναι η Εξελικτική Ανάπτυξη [21] (Εικόνα 10), η διαδικασία αυτή έχει ως πρώτο βήμα την καταγραφή των μη λειτουργικών απαιτήσεων ενώ η συγγραφή των λειτουργικών απαιτήσεων γίνεται σταδιακά ανά κομμάτι της εφαρμογής. Έπειτα ακολουθεί η ανάπτυξη λογισμικού ανά κομμάτι και παρουσιάζεται για την επιβεβαίωση από τον πελάτη για έγκριση. Οι διαδικασίες αυτές επαναλαμβάνονται μέχρι να ολοκληρωθεί όλο το λογισμικό, σε σύγκριση με την μέθοδο Καταρράκτη [22] που δεν υπάρχει αλληλεπίδραση από τον πελάτη αλλά μόνο όταν αναπτυχθεί όλο το λογισμικό (Εικόνα 11).



Εικόνα 10. Εξελικτική ανάπτυξη λογισμικού [23]



Εικόνα 11. Μοντέλο καταρράκτη

4.1 Αναλυτική καταγραφή μη λειτουργικών απαιτήσεων

Η αναλυτική καταγραφή των μη λειτουργικών απαιτήσεων πριν την ανάπτυξη λογισμικού είναι αναγκαία διαδικασία λογού ότι οι απαιτήσεις αυτές αφορούν την άμεση λειτουργικότητα της εφαρμογής, δηλαδή ορίζουν τους περιορισμούς στις υπηρεσίες ή στις λειτουργίες που προφέρει το σύστημα. Συνήθως αφορούν την αξιοπιστία , τις τεχνολογίες και τα πρότυπα που πρέπει να χρησιμοποιηθούν κατά το στάδιο ανάπτυξη της εφαρμογής ωστόσο σε πολλές περιπτώσεις η διαχώριση των δυο κατηγοριών απαιτήσεων είναι μια λεπτή γραμμή . Οι μη λειτουργικές απαιτήσεις που καταγράφηκαν σύμφωνα με τις επιθυμίες του πελάτη είναι

1) **Αξιοπιστία**
   - Το σύστημα θα πρέπει να είναι διαθέσιμό για την λειτουργία του λογισμικού όταν το απαιτεί ο χρήστης.
   - Το λογισμικό θα πρέπει να εγγυάται στον χρήστη ότι θα συμβαίνουν μηδαμινές αποτυχίες όταν το λογισμικό θα εκτελείται από αυτόν.
   - Η εφαρμογή θα πρέπει να αποθηκεύει τα δεδομένα που θα εισάγονται από τον χρήστη και σε καμία περίπτωση δεν πρέπει να διαγράφονται, μόνο με την συγκατάθεση του χρήστη
2) **Απόδοση**
   - Η εφαρμογή θα πρέπει να ανταποκρίνεται σε λιγότερο από ένα δευτερόλεπτο όταν εκτελείται από τον χρήστη και όταν γίνεται μετάβαση σε διαφορετικές οθόνες
   - Το σύστημα δεν πρέπει να καταλαμβάνει μεγάλο μέρος του αποθηκευτικού χώρου της συσκευής
3) **Ευχρηστία**
   - Η εφαρμογή θα διαθέτει φιλικό περιβάλλον προ τον τελικό χρήστη. Η κυριά οθόνη θα περιέχει εικονίδια και διαγράμματα προόδου του χρήστη. Ώστε να εξογκωθεί με το λογισμικό γρηγορά και ευκολά
   - Η εφαρμογή για κάθε οθόνη θα διαθέτει διαφορετικό χρώμα, όπως και στην κεντρική οθόνη θα περιέχοντα τα αντίστοιχα χρώματα για την πλοήγηση. Ώστε να είναι εύκολο στον χρήστη να ξεχωρίζει τις οθόνες και μόνο από τα χρώματα

- Η εφαρμογή θα διαθέτει διαγράμματα σε κάθε οθόνη που θα υποδηλώνουν την αντίστοιχη κατηγορία προόδου του χρήστη, όταν αυτός θα πλοηγείτε σε αυτές
- Η εφαρμογή θα στέλνει μια φορά την ημέρα τα δεδομένα του χρήση για αποθήκευση στον κεντρικό διακομιστή, είτε κάθε φορά που επιθυμεί ο χρήστης να συγχρονιστούν τα δεδομένα αυτά

4) **Εξελισιμότητα**
   - Το λογισμικό θα πρέπει να έχει την δυνατότητα εξέλιξης μέσω από τα αντίστοιχα market της συσκευής του χρήστη
   - Η εφαρμογή θα πρέπει να κρατάει τα προηγούμενα δεδομένα και την πρόοδο του χρήστη σε περίπτωση αναβάθμισης
5) **Βάση δεδομένων**
   - Το λογισμικό θα αποθηκεύει την πρόοδο του χρήστη σε βάση δεδομένων SQLite, επίσης οι οντότητες δεν θα ξεπερνάτε σχέσεις εξάρτησης των δυο επίπεδων

6) **Εγκατάσταση**
   - Η εγκατάσταση του λογισμικού θα πρέπει να γίνεται ευκολά και γρηγορά στην συσκευή του χρήστη
   - Επίσης η απεγκατάσταση του λογισμικού θα πρέπει να γίνεται γρήγορα και να μην αφήνει κανέναν ίχνος ότι η εφαρμογή αυτή ήταν εγκαταστημένη στη συσκευή του χρήστη
7) **Συμβατότητα**
   - Το λογισμικό θα διάθετέ για αρχικά για την πλατφόρμα λογισμικού Android και έπειτα για IOS, Widows. Επίσης σε όλες τις πλατφόρμες λογισμικών η εφαρμογή θα έχει το ιδιά διεπαφή χρήστη
8) **Λειτουργία**
   - Η εφαρμογή θα πρέπει να συγκρίνει πληροφορίες για την κίνηση(βήματα) του χρήστη ακόμα και όταν αυτός δεν την εκτελεί
9) **Κόστος**
   - Η υλοποίηση του συστήματος δεν πρέπει αν υπερβαίνει τις 4.000 ευρώ συνολικά
   - Το κόστος συντριβής του συστήματος δεν πρέπει να υπερβαίνει τα 200 ευρώ μηναία

Σύμφωνα με τις παραπάνω απαιτήσεις προκύπτει ότι θα πρέπει να δημιουργηθεί μια υβριδική εφαρμογή που έχει μικρό κόστος κατασκευής και έχει την δυνατότητα εκτελέσεις από όλες τις πλατφόρμες λογισμικών για κινητές συσκευές. Επίσης λόγο χωρητικότητας πρέπει αν χρησιμοποιούν βιβλιοθήκες JavaScript μικρού μεγέθους και εικόνες με μικρές διαστάσεις για να μην καταλαμβάνουν μεγάλο αποθηκευτικό χώρο στην συσκευή του χρήστη.

4.2 Καταγραφή των λειτουργικών απαιτήσεων και υλοποίηση της εφαρμογής

Οι λειτουργικές απαιτήσεις, είναι οι απατήσεις του πελάτη που αφορούν άμεσα την λειτουργικότητα της εφαρμογής και την εμφάνιση της. Σύμφωνα με την διαδικασία του εξελικτικού προγραμματισμού που ακολουθείται, αρχικά θα γράφονται αναλυτικά οι απαιτήσεις λογισμικού , έπειτα θα συγγράφεται ο κώδικας και μετά γίνεται η παρουσίαση στον πελάτη. Στην πλαίσια της διπλωματικής θα θεωρηθεί ότι η παρουσιάζεται κάθε ολοκληρωμένη οθόνη ξεχωριστά στον πελάτη αντί για τμήματα

οθονών της εφαρμογής όπως προδιαγράφεται από τον εξελικτικό προγραμματισμό και θα γίνεται η ανάλυση μόνο του κεντρικού προγραμματιστικού κώδικα για αποφυγή πολυλογίας.

Οι τεχνολογίες (βιβλιοθήκες) που χρησιμοποιήθηκαν για την ανάπτυξη σε κώδικα JavaScript και HTML για όλες τις οθόνες χρήστη, είναι το Dev Express Chart [24] για την δημιουργία διαγράμματος που θα επιτρέπει στον χρήστη να παρακολουθεί την πρόοδο αι τα δεδομένα του. Το JQuery [25] για λειτουργία μεθόδων που η συγγραφή τους θα ήταν επίπονη και χρονοβόρα διαδικασία. Το JQuery Mobile [26] για την αναλογική τροποποίηση των αντικειμένων ώστε να μην εχουν καμία διαφορά σε όλα τα μεγέθη οθονών, επίσης χρησιμοποιήθηκαν χαρακτηριστικά και αντικείμενα της βιβλιοθήκης αυτής για να προσομοιώσουν τις γραφικές διεπαφές χρήστη των μητρικών εφαρμογών. Η βιβλιοθήκη Bootstrap [27] για την σχεδίαση γραφικών διεπαφών ώστε να είναι λείες και φιλικές προς τον χρήστη. Τελευταία, για την πλοήγηση στο ιστορικό του χρήστη χρησιμοποιήθηκε η βιβλιοθήκη Sly [28], ώστε να γίνεται ευχάριστη η προβολή του ιστορικού και εύκολη προς τον χρήστη.

4.2.1 Κεντρική οθόνη

Οι λειτουργικές απαιτήσεις που καταγράφτηκαν σύμφωνα με τις επιθυμίες του πελάτη για την κεντρική οθόνη είναι

- Το σύστημα θα δίνει την δυνατότητα στον χρήστη να βλέπει την ημερήσια του κίνηση(βήματα) μέσα από ένα διάγραμμά.
- Η εφαρμογή θα επιτρέπει στον χρήστη να προηγηθεί σε όλες τις οθόνες της εφαρμογής. Το κάθε κουμπί πλοηγήσεις θα αποτελείται από μια εικόνα που θα περιλαμβάνει το χρώμα της οθόνης που αντιστοιχεί και το όνομα της αντίστοιχα
- Το λογισμικό θα επιτρέπει να στον χρήστη να βλέπει τα στοιχεία ταυτότητας του, αλλά ο χρήστης δεν θα έχει την δυνατότητα τροποποίησης ονόματος, ύψους και ημερομηνία γέννησης
- Η εφαρμογή θα επιτρέπει στον χρήστη να αυξάνει τον μετρητή ποτηριών νερού
- Το λογισμικό θα επιτρέπει στον χρήστη να αποσυνδεθεί από την εφαρμογή και να διαγράψει όλα τα δεδομένα που έχουν σχέση με αυτόν. Η διαγραφή θα γίνεται μόνο με την συγκατάθεση του χρήστη
- Η εφαρμογή θα συγχρονίζει όλα τα δεδομένα του χρήστη με τον κέντρο διακομιστεί όταν εκτελείται, χωρίς την συγκαταθέσεις του χρήστη. Επίσης το λογισμικό θα δίνει την δυνατότητα στον χρήστη να συγκλονίζει τα δεδομένα του οπότε επιθυμεί με το πάτημα ενός κουμπιού

Σύμφωνα με τις παραπάνω απαιτήσεις, εγκαταστάθηκε plugin στο Cordova όπως αναφέρθηκε στο προηγούμενο κεφάλαιο και τροποποιήθηκε έτσι ώστε τα δεδομένα που συλλέγονται να αποθηκεύονται σε SharedPreference (.xml μορφή) για διευκόλυνση της πρόσβαση των δεδομένων προσφέροντας μεγάλες ταχύτητες εγγραφής των δεδομένων. Είναι σημαντικό ο τρόπος και ο τύπος που θα γράφονται τα δεδομένα από το Service, λόγο ότι θα εκτελείται συνέχεια από την συσκευή του χρήστη, έτσι πρέπει η λειτουργία να είναι απλή ώστε να μη καταναλώνεται μεγάλη υπολογιστική ισχύ και να επιβραδύνει όλες τις υπόλοιπες λειτουργίες.

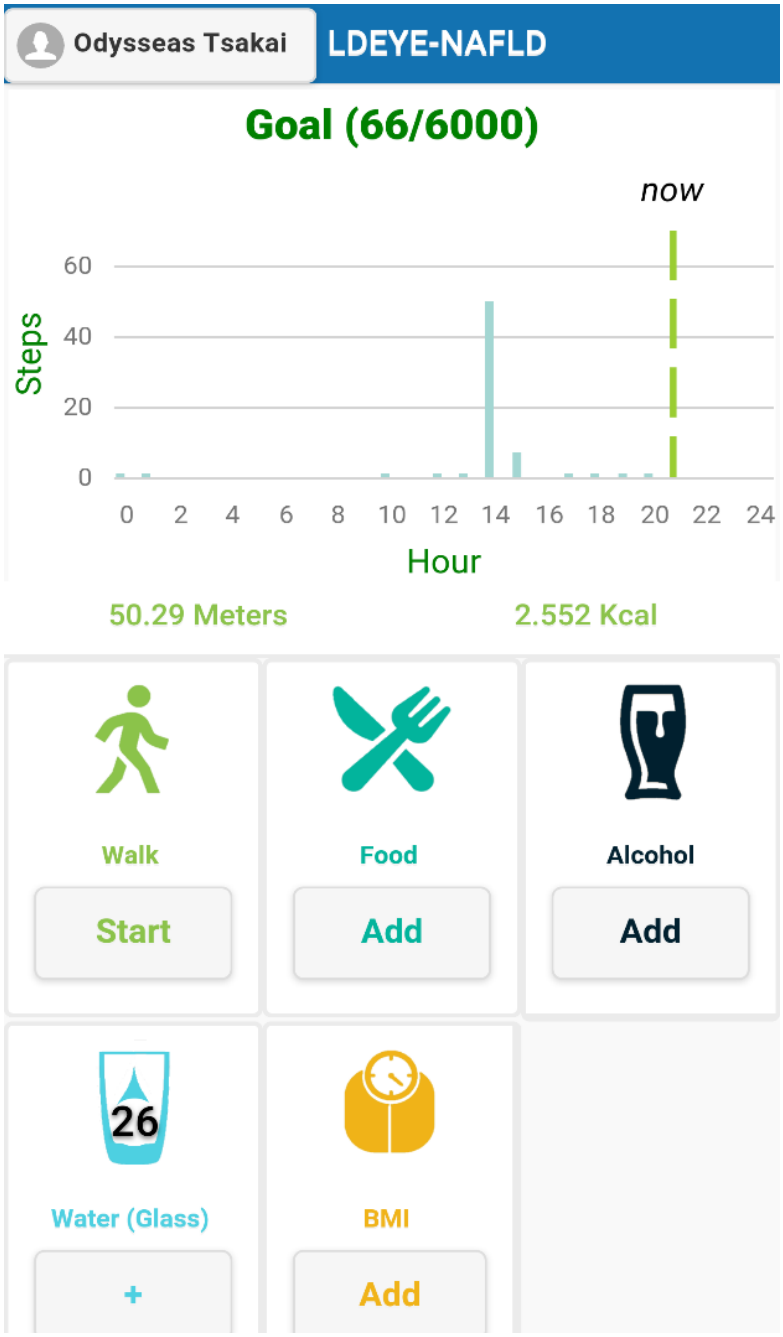
Το κύριο σημείο της κεντρικής οθόνης που χρειάστηκε την προσοχή και περισσοτέρους πόρους του συσκευής του χρήστη είναι ο συγχρονισμός των δεδομένων, ώστε να μην

χάσει την πρόοδο του. Όπως φαίνεται στην παρακάτω εικόνα (Εικόνα 12) εξετάζεται λεπτομερειακά ποτέ έγινε η τελευταία εισαγωγή ή τροποποίηση των κατηγοριών ώστε όλα τα δεδομένα που έχουν εισαχθεί από τον χρήστη είτε έχουν καταγράφει από το Service να σταλθούν στον κεντρικό διακομιστή. Όταν ολοκληρωθεί η αποστολή των δεδομένων σημειώνεται η μέρα σε SharedPreference και τροποποιούνται η στήλες όλων των οντοτήτων ώστε να μην ξανασταλθούν εάν δεν τροποποιηθούν. Η διαδικασία αυτή ακολουθείται ώστε να μην ξανασταλθούνε τα ιδιά δεδομένα καθημερινά και να μην εκτελείται η λειτουργία αυτή αυτόματα εάν δεν υπάρχει αλλαγή. Η λειτουργία αυτή είναι χρονοβόρα και χρειάζεται μεγάλη υπολογιστική ισχύ και σύνδεση στο διαδίκτυο.

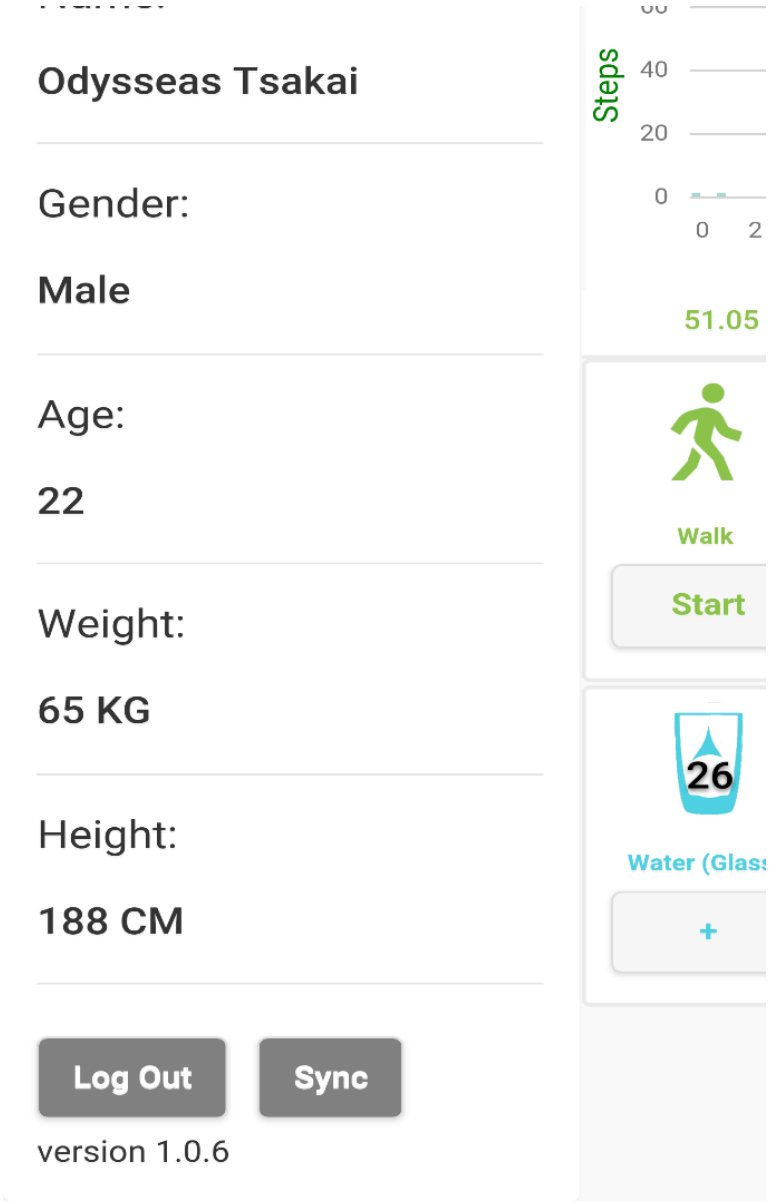
```
function getLastSyncData(tx){
tx.executeSql("select MIN(date) as da from bmi where sync=0",[],function(tx, results){console.log(results.rows.length+"bmi dat
tx.executeSql("select MIN(date) as da from food where sync=0 and date <>'1994-1-23' ",[],function(tx, results){console.log(res
tx.executeSql("select MIN(daily) as da from water where sync=0",[],function(tx, results){console.log(results.rows.length+" wat
tx.executeSql("select MIN(date) as da from alcohol where sync=0 and date <>'1994-1-23' ",[],function(tx, results){console.log(
}
function collectAllData(w){
console.log(historyData);
    var lastUpdate;
db.transaction(getLastSyncData,function(error){console.log(error.message)},function(){console.log('success')});
 getValue("lastUpdate","string",function(val){lastUpdate=Date.create(val); console.log('found LastUpdate'+lastUpdate);
 console.log(lastUpdate.getTime()+" aaaaa"+Date.create("yesterday").getTime()+" aaa"+w)
  if(lastUpdate.getTime()<=Date.create("yesterday").getTime()||w==='now')
  res()
  });
 function res(){
  var day;
  if(w===undefined)
      day=Date.create('yesterday');
      else{...}

      var oldest="";
      if(lastUpdate.getTime()<lastSyncDataDate.getTime())
      oldest=Date.create(lastUpdate);
      else
      oldest=Date.create(lastSyncDataDate);
      window.copyOldest=Date.create(oldest);
var fitness={date:forger(oldest),walking:[],bmi:[],food:[],water:0};
    for(var i=0;oldest<=day;i++){

        fitness={date:forger(oldest),walking:[],food:[],alcohol:[]};
        getFeetByDate(oldest);
        getFoodByDate(oldest);
         getBMIByDate(oldest);
          getAlcoholDay(oldest);
```

Εικόνα 12. Κώδικα σε JavaScript για την συλλογή των δεόμενων του χρήστη και συγχρονισμός με τον κεντρικό διακομιστή

Το αποτέλεσμα τις κεντρικής οθόνης που παρουσιάστηκε στον πελάτη φαίνεται στις παρακάτω εικόνες (Εικόνες 13 και 14). Ο πελάτης έμεινε ευχαριστημένος και επέμεινε να συνεχιστεί το έργο και για τις υπόλοιπες λειτουργίες της εφαρμογής.



Εικόνα 4. Κεντρική οθόνη



Εικόνα 14. Στοιχεία χρήστη

4.2.2 Πεδόμετρο και ιστορικό χρήστη

Οι λειτουργικές απαιτήσεις που καταγραφτήκαν για την παρούσα οθόνη σύμφωνα με τις επιθυμίες του πελάτη, που θα περιέχει τα αποτελέσματα από το Services ποδομέτρου και το ιστορικό όλων των ημέρων είναι

- Το σύστημα θα δίνει την δυνατότητα στον χρήστη να παρακολουθεί τα βήματα του ημερήσια βάση αναλυτικά ανά ώρα
- Η εφαρμογή θα επιτρέπει στον χρήστη να παρακολουθεί τα βήματα που έχει διανύσει σε όλη την διάρκεια που η εφαρμογή έχει μαζέψει δεδομένα(ιστορικό χρήστη)
- Το λογισμικό θα υπολογίζει από τα βήματα που έχει διανύσει ο χρήστης σε μετρά και συμφώνα με τα στοιχεία που έχουν καταχωρηθεί, και πόσες θερμίδες

έχει κάψει σε Kcal. Τα αποτελέσματα αυτά θα επιτρέπονται να τα βλέπει ο χρήστης

Σύμφωνα με τα παραπάνω απαιτήσεις παρατηρείται ότι δεν υπάρχει μεγάλη δίφορα από την κεντρική οθόνη πάρα στο Ιστορικό χρήστη και την φόρμουλα που θα υπολογίζει από τα αριθμό των βημάτων σύμφωνα με το βάρος και το ύψος του χρήστη. Η φόρμουλα για τον υπολογισμό των μέτρων που έχει διανύσει ο χρήστης είναι

(allSteps*0.762)/1.609344*(height/1.65)* 0.621371192

Ενώ για την καταμέτρηση θερμίδων η φόρμουλα είναι

((weight/0.45359237)*0.57)*(distance/1000);

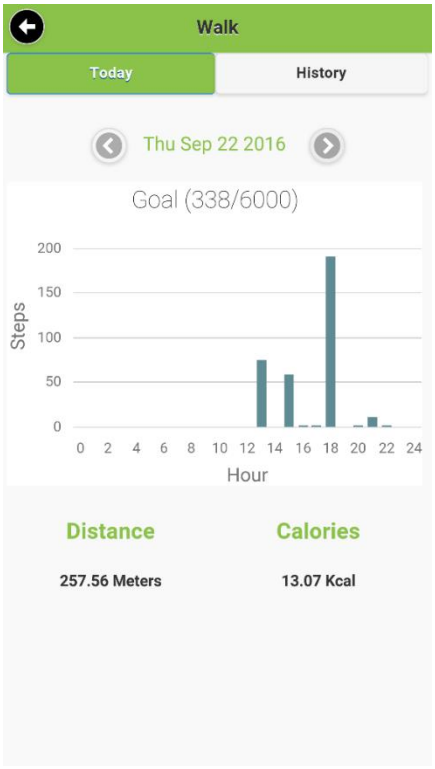
Οπού distance είναι τα μετρά που έχει διανύσει ο χρήστης [29]

Το σημαντικότερο κομμάτι της οθόνη είναι το ιστορικό του χρήστη στο οποίο θα πρέπει δυναμικά να αντλούνται η πληροφορίες από τα SharedPreference ποδομέτρου. Η αίτια που πρέπει η διαδικασία αυτή να γίνεται δυναμικά είναι, σε περίπτωση που αποθηκευόντουσαν τα δεδομένα στην RAM θα έπαιρνε αρκετό χρόνο να ολοκληρωθεί και θα κατανάλωνε μεγάλη υπολογιστική ισχύ, με συνέπεια η εφαρμογή να μην ανταποκρίνεται. Η λειτουργία αυτή γίνεται ως εξής, ο χρήστης μετακινείται στη ημερομηνία που θέλει και το λογισμικό βρίσκει για τις πέντε μέρες τα αποτελέσματα και ανανεώνει το διάγραμμά με τις καινούριες μετρήσεις (Εικόνα 15).

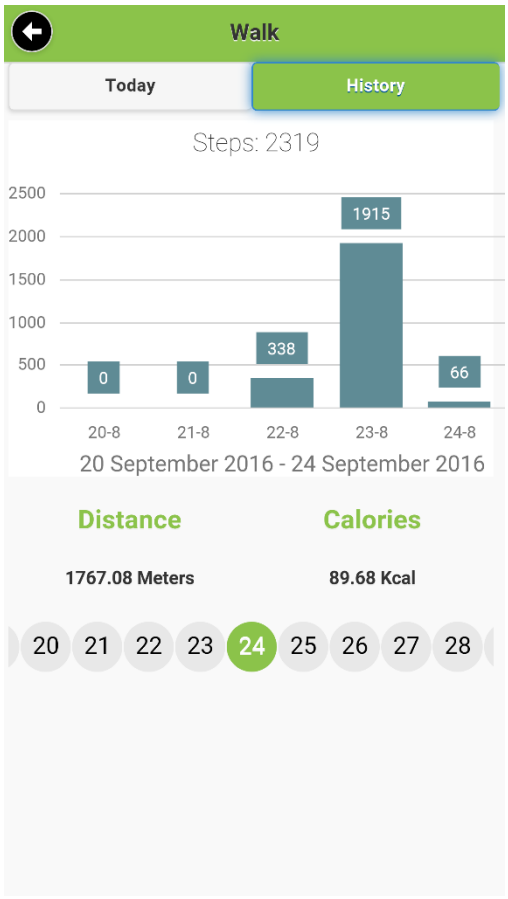
```
function handlePrefFootManyDays(date1,page){
dataSourceDays=[];
for(var k=4;0<=k;k--){
var todayStep=0;
var date=new Date.create(date1);
date.addDays(-k);
var da=new Date.create(date);
date=dateSimpleDataFormat(date);
for(var i=0;i<24;i++){
var j=i;
if(i<10)
j="0"+i;
var z=date+"  "+j;
if (typeof(historyData[z]) !== 'undefined'){
todayStep+=historyData[z].steps;
console.log(historyData[z].steps);
}
}
    dataSourceDays.push({day:(da.getDate()+"-"+da.getMonth()),steps:todayStep});

}
                var x=0;
  for(var i=0;i<dataSourceDays.length;i++){
                    console.log(dataSourceDays[i].day+"steps:"+dataSourceDays[i].steps);
                    x+=dataSourceDays[i].steps;}
                chartDays.dxChart({dataSource:dataSourceDays});
                 chartDays.dxChart({argumentAxis:{title:titleDays}});
                 chartDays.dxChart({ title: {
                                                text:"Steps: "+x,font:{size:18}}});
                 var w=0;
console.log("calc"+page);

                 getValue("weight","int",function(val){console.log("calculate "+window.page);
                  if(window.page===2)
```

Εικόνα 15. Κώδικας για την εξαγωγή των μετρήσεων και ανανέωσης του διαγράμματος στην οθόνη ποδομέτρου και ιστορικό χρήστη

Το αποτέλεσμα που παρουσιάστηκε στον πελάτη φαίνεται στις παρακάτω εικόνες (Εικόνες 16 και 17). Ο πελάτης έμεινε ευχαριστημένος και επίμενε να συνεχιστεί το έργο και για τις υπόλοιπες λειτουργίες της εφαρμογής.



Εικόνα 16. Ημερήσια βήματα χρήστη



Εικόνα 17. ιστορικό βημάτων χρήστη

4.2.3 BMI μετρητής και ιστορικό χρήστη

Οι λειτουργικές απαιτήσεις που καταγραφτήκαν σύμφωνα με τις επιθυμίες του πελάτη για οθόνη του BMI (δείκτης μάζας σώματος) μετρητή και ιστορικό χρήστη είναι

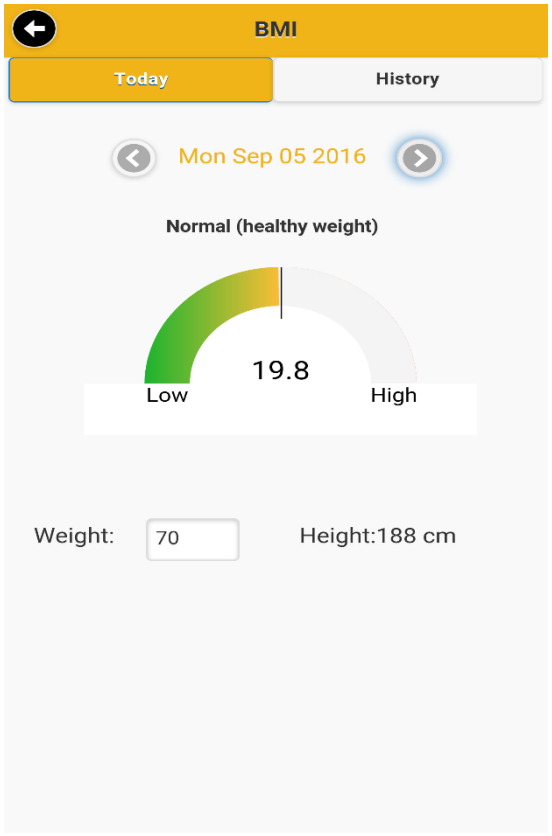
- Το σύστημα θα δίνει την δυνατότητα στο χρήστη να τροποποιεί το βάρος του όσες φορές επιθυμεί και την ημέρα που το επιθυμεί
- Το σύστημα θα εμφανίζει γραφικά στον χρήστη το BMI του όταν θα βρίσκεται στην σελίδα αυτή
- Το σύστημα θα παρέχει στον χρήστη ιστορικό για όλες τις μέτρησης BMI που έχουν γίνει στην εφαρμογή από αυτόν(ιστορικό χρήστη)

Συμφώνα με τις παραπάνω απαιτήσεις, συνεπάγεται πως για το ιστορικό χρήστη θα χρησιμοποιεί ο ίδιος τον αλγόριθμος από το ιστορικό στην οθόνη ποδομέτρου. Το κυριότερο κομμάτι που δόθηκε έμφαση είναι το γραφικό στοιχείο που διαμορφώνεται με το βάρος του χρήστη. Η διαδικασία που ακολουθείται είναι η εξής, στη αρχή εκτέλεσης της σελίδας εξάγεται η πιο πρόσφατη εισαγωγή BMI από τον πίνακα βάσης. Έπειτα αφού ολοκληρωθεί η διαδικασία αυτή γίνεται το Rendering και δίνεται η δυνατότητα στον χρήστη να δει το αποτέλεσμα. Σε περίπτωση μετατόπισης της ημερομηνίας ακολουθεί η ιδιά τακτική (Εικόνα 18).

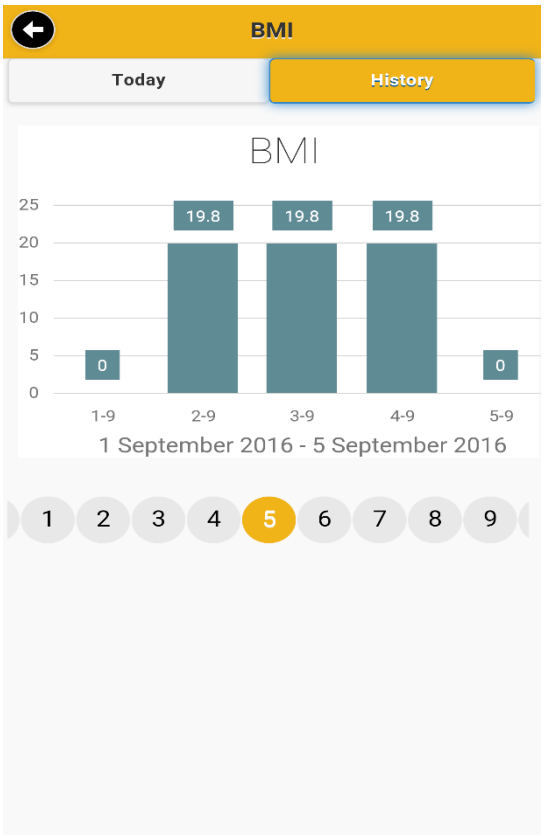
```
function selectTodayDb(){


    db.transaction(getTodayBMI,function(error){console.log("bmi error"+error.message)},function(){console.log("success bmi")});
    function getTodayBMI(tx){


//tx.executeSql("delete from bmi");
console.log("select * from bmi where date='"+dateForger(when)+"'");
tx.executeSql("select * from bmi where date='"+dateForger(when)+"'",[],selectTodayBMIDb,function(error){console.log("food error sel

function selectTodayBMIDb(tx, results) {
  var len = results.rows.length;
var found=true;
console.log(" today on bmi");
  if(len===0){
              //  window.height=172;
                // window.weight=0;
console.log("nothing today on bmi");
                  }
                  for (var i = 0; i < len;i++) {
               window.height=Number(results.rows.item(i)['height']);
                console.log("window.height"+Number(results.rows.item(i)['height']));
                  window.weight=Number(results.rows.item(i)['weight']);
          }
  ///animation BMI
  taging();

          }
      }
      }
```

Εικόνα 18. Κώδικας για την εξαγωγή των μετρήσεων και ανανέωσης animation BMI στην οθόνη BMI και ιστορικό χρήστη

Το αποτέλεσμα που παρουσιάστηκε στον πελάτη φαίνεται στις παρακάτω εικόνες (Εικόνες 19 και 20). Ο πελάτης έμεινε ευχαριστημένος και επίμενε να συνεχιστεί το έργο και για τις υπόλοιπες λειτουργίες της εφαρμογής.



Εικόνα 19. Ημερήσια μέτρηση BMI



Εικόνα 5. Ιστορικό χρήστη για BMI μέτρησης

4.2.4 Θερμιδομετρητής φαγητών από Online βάση δεδομένων και ιστορικό χρήση

Οι λειτουργικές απαιτήσεις που καταγράφτηκαν για την παρούσα οθόνη σύμφωνα με τις επιθυμίες του πελάτη είναι

- Το σύστημα θα δίνει την δυνατότητα στον χρήστη να παρακολουθεί τις θερμίδες των φαγητών που κατανάλωσε συνολικά όλη την ημέρα
- Η εφαρμογή θα περιέχει ιστορικό χρήστη που θα υλοποιείται με διάγραμμα για όλες τις καταχωρήσεις φαγητών που έχει εισάγει ο χρήστης
- Το λογισμικό θα επιτρέπεις στον χρήστη να επιλεγεί για την εισαγωγή των φαγητών από τις ακόλουθες κατηγορίες πρωινό, μεσημεριανό, βραδινό, πρωινό σνακ, απογευματινό σνακ και βραδινό σνακ
- Η εφαρμογή θα περιέχει αναλυτική λίστα των φαγητών που κατανάλωσε ο χρήστης και θα του δίνεται η δυνατότητα οξοποίησης της ποσότητας η ακόμα και διαγραφή
- Το λογισμικό θα επιτρέπει στον χρήστη να διαλέγει το φαγητό που επιθυμεί από την βάση δεδομένων FatSecret [30] και να επιλεγεί το είδος την μερίδας αλλά και τις ποσότητες που κατανάλωσε
- Το λογισμικό θα επιτρέπει στον χρήστη να επιλεγεί αγαπημένα φαγητά ώστε να διευκολύνεται. Τα αγαπημένα φαγητά του χρήστη θα είναι ανά κατηγορία γεύματος και θα είναι διαθέσιμα ακόμα και όταν δεν υπάρχει σύνδεση στον διαδίκτυο

Σύμφωνα με τις παραπάνω απαιτήσεις υλοποιήθηκε η παρούσα οθόνη με πλήρης λειτουργικότητα και φιλικό περιβάλλον προς τον χρήστη. Τα κυριότερα σημεία της οθόνης είναι η εξαγωγή πληροφοριών από την βάση δεδομένων FatSecret και η δημιουργία λίστας με όλα τα φαγητά που κατανάλωσε κατά την διάρκεια της ημέρας.

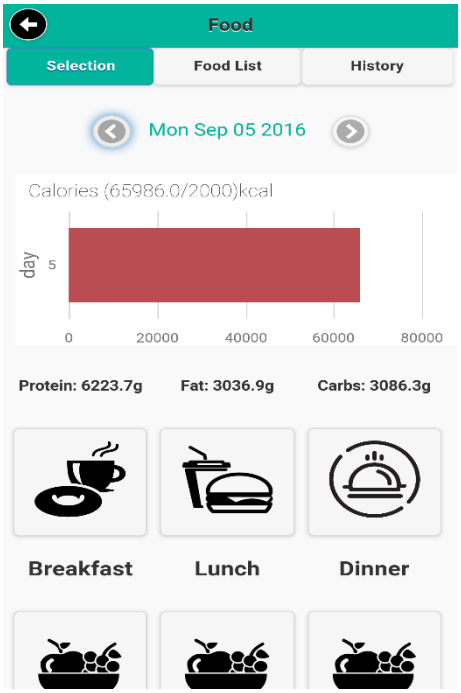
Το FatSecret παρέχει διασύνδεση με την βάση δεδομένων τους, με τα Rest APIs που είναι υλοποιημένα για γλώσσες προγραμματισμού, που εκτελούνται από την μεριά του διακομιστή όπως PHP,Java, ASP κτλ. Επίσης οι προγραμματιστές τους παρέχουν τα JavaScript APIs που αλλά είναι ολοκληρωμένα και οι πληροφορίες που αποστέλνονται από την πλατφόρμα τους δεν είναι σε JSON(JavaScript Object Notation) ή XML μορφή μέσω AJAX κλήσεων, στέλνονται αρχεία HTML. Έτσι αυτή η διαδικασία είναι επίπονη γιατί πρέπει να αποκρυπτογραφηθεί όλη η σελίδα που έχει αποσταλεί ώστε να παρθούν οι κατάλληλες πληροφορίες που ζητήθηκαν.

Η δημιουργία της λίστας των φαγητών που κατανάλωσε σε ημερήσια βάση με πλήρης λειτουργικότητα στο κάθε αντικείμενο όπως ζητήθηκε από τον πελάτη, είναι δυναμική και εξάγονται πληροφορίες από δυο πίνακες των Food και τον api. Ο api είναι ο πίνακας που έχει όλες τις πληροφορίες συστατικών και είδος μερίδας για κάθε φαγητό που εισάγεται από το FatSecret έτσι ώστε να υπάρχει δυνατότητα εισαγωγής δεδομένων ακόμα και όταν δεν υπάρχει σύνδεση στον διαδίκτυο, εφόσον την έχει επιλέξει μια φορά ο χρήστης.

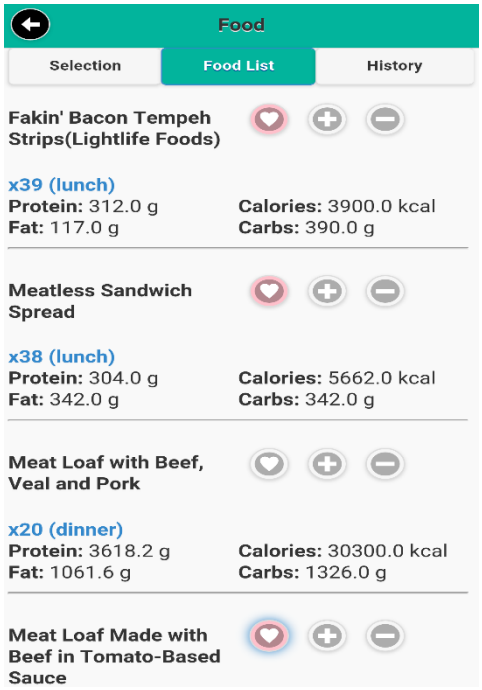
```
function userFunctions(foodOb,action){
foodOb=foodAr[foodOb];
console.log(JSON.stringify(foodOb)+action )
db.transaction(actionDb,function(error){console.log("error"+error.message)},function(){console.log('success')});
function actionDb(tx){
if(action==="minus"){
if(foodOb.amount==1){

tx.executeSql("update food set date='1994-1-23' where id="+foodOb.id)
selectFoodListDb();
}
else{
foodOb.amount--;
tx.executeSql("update food set amount="+(foodOb.amount)+" where id="+foodOb.id);
$(".amount"+foodOb.id).text("x"+foodOb.amount+" ("+cateOb[foodOb.category]+")");
db.transaction(function(){$("."+foodOb.api.id+foodOb.id).children().remove();
 $("."+foodOb.api.id+foodOb.id).append("<div class='ui-grid-a' style='text-align: left'><div class='ui-block-a'><strong> Protein:</
},null,null);
}
asyncDay('food');
}
if(action==='plus'){
foodOb.amount++;
tx.executeSql("update food set amount="+(foodOb.amount)+" where id="+foodOb.id);
$(".amount"+foodOb.id).text("x"+foodOb.amount+" ("+cateOb[foodOb.category]+")");
db.transaction(function(){$("."+foodOb.api.id+foodOb.id).children().remove();
 $("."+foodOb.api.id+foodOb.id).append("<div class='ui-grid-a' style='text-align: left'><div class='ui-block-a'><strong> Protein:</
},null,null);
asyncDay('food');
}
if(action==='love'){
if(foodOb.love===1){
$('#f'+foodOb.id+" .ui-icon-heart").css("background-color","white");
console.log("no love");
foodOb.love=0;
```

Εικόνα 21. Κώδικας για την πλήρης λειτουργικότητα του κάθε τρόφιμου που κατανάλωσε ο χρήστης

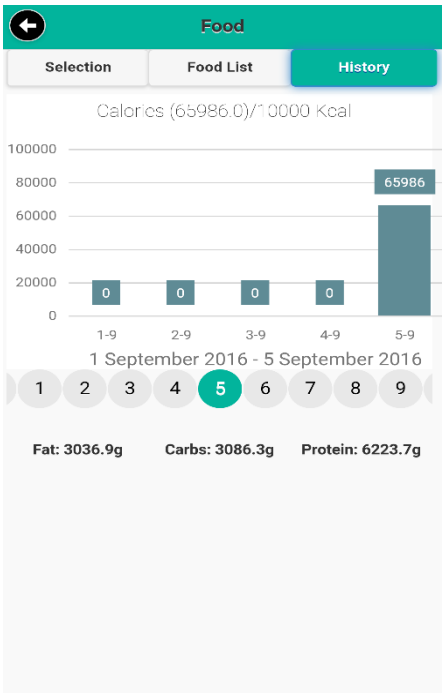
Το αποτέλεσμα που παρουσιάστηκε στον πελάτη φαίνεται στις παρακάτω εικόνες (Εικόνες 22-27). Ο πελάτης έμεινε ευχαριστημένος και επίμενε να συνεχιστεί το έργο και για τις υπόλοιπες λειτουργίες της εφαρμογής.



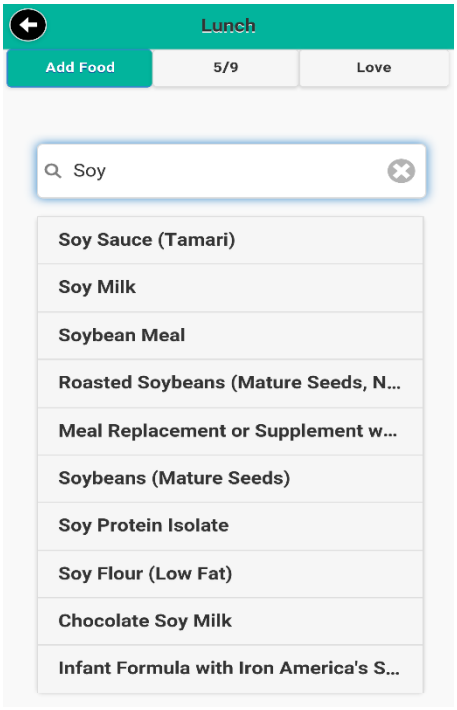
Εικόνα 6. Ημερήσια κατανάλωση φαγητού



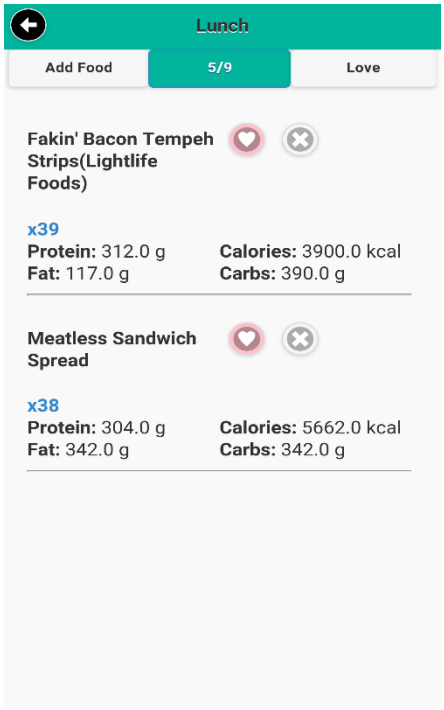
Εικόνα 7. Λίστα φαγητών που καταναλωθήκαν ημερήσια



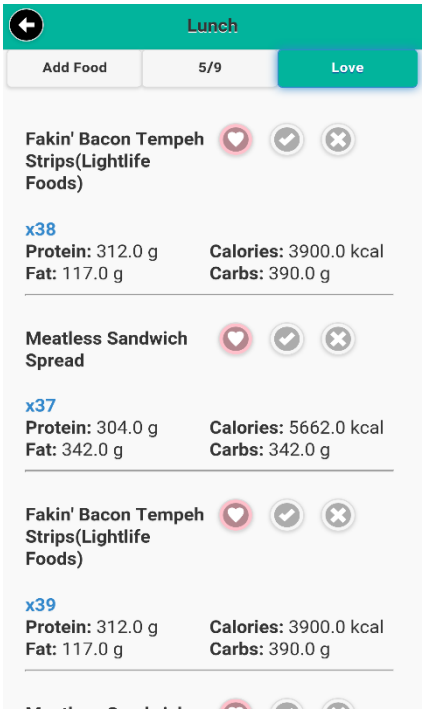
Εικόνα 24. Ιστορικό χρήστη



Εικόνα 25. Επιλογή φαγητών από το FatSecret



Εικόνα 26. Λίστα φαγητών που καταναλωθήκαν



Εικόνα 27.Λιστα Αγαπημένων φαγητών

4.2.4 Μέτρηση κατανάλωση αλκοολούχων ποτών και ιστορικό χρηστή

Οι λειτουργικές απαιτήσεις που καταγράφτηκαν για την παρούσα οθόνη σύμφωνα με τις επιθυμίες του πελάτη είναι

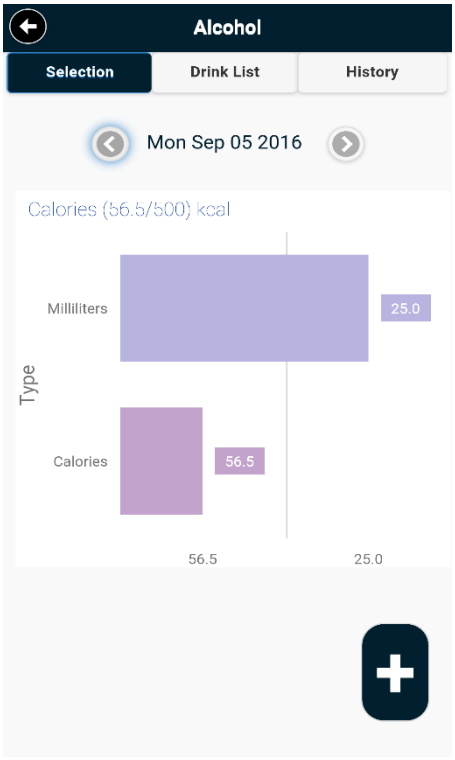
- Το λογισμικό θα δίνει την δυνατότητα στον χρήστη να παρακολουθεί τις θερμίδες και την ποσότητα των αλκοολούχων ποτών που κατανάλωσε συνολικά σε όλη την ημέρα
- Η εφαρμογή θα δίνει την δυνατότητα στον χρήστη να παρακολουθεί το ιστορικό όλων των αλκοολούχων ποτών που έχει καταναλώσει
- Το σύστημα θα επιτρέπει στον χρήστη να επεξεργαστεί τα αλκοολούχα ποτά που έχει καταναλώσει στην μέρα, σε μια κατακόρυφη λίστα
- Το λογισμικό θα επιτρέπει στον χρήστη να επιλεγεί μέσα από έξι κατηγορίες αλκοολούχων ποτών beer , wine, spirits, alcopop, cider και champagne, έπειτα να επιλέγει το είδος του αντικειμένου που το κατανάλωσε, την ποσότητα και το ποσοστό αλκοόλ
- Το λογισμικό θα επιτρέπει στον χρήστη να επιλέγει για εκχώρηση αγαπημένα αλκοολούχα ποτά ώστε να γίνεται γρηγορά η εισαγωγή για διευκόλυνση του πελάτη

Σύμφωνα με τις παραπάνω απαιτήσεις υλοποιήθηκε η παρούσα οθόνη με πλήρης λειτουργικότητα και φιλικό περιβάλλον προς τον χρήστη. Τα κυριότερο προγραμματιστικό κομμάτι ήταν η δημιουργία αντικειμένων σε JavaScript που θα παριστάνουν τις κατηγορίες και επιλογές που θα είναι διαθέσιμες για τον χρήστη. Τα JavaScript αντικείμενα δημιουργήθηκαν συμφώνα με τις επιθυμίες του πελάτη και προβολή τους στον χρήστη γίνεται δυναμικά σύμφωνα με έναν αλγόριθμο που επεξεργάζεται τα αντικείμενα αυτά (Εικόνα 28).

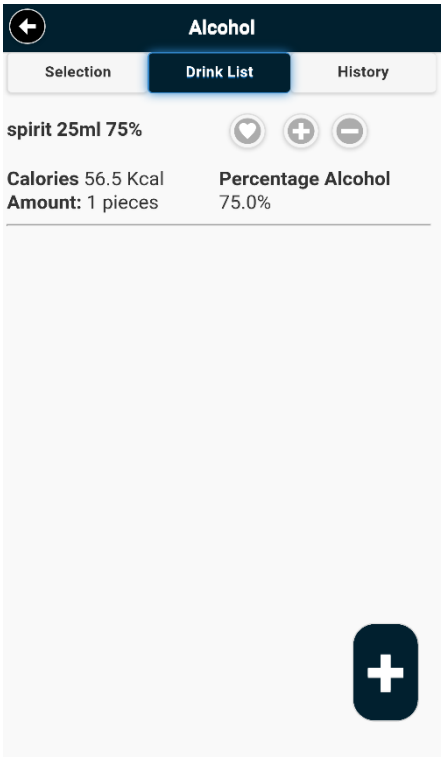
```
    var Beer={name:"Beer",icon:"beer",instances:[
{size:284,name:'Half Pint 284ml',img:'1'},
{size:300,name:'Bottle 300ml',img:'2'},
{size:440,name:'Can 440ml',img:'3'},
{size:500,name:'Large Can 500ml',img:'4'},
{size:568,name:'Pint 568ml',img:'5'},
{size:750,name:'Bottle 750ml',img:'6'},
{size:1000,name:'Large Bottle 1000ml',img:'6'}],
percentage:{min:3,max:9,step:0.5},
calories:0.32};
    var Wine={name:"Wine",icon:"wine",instances:[
{size:125,name:'Small Glass 125ml',img:'1'},
{size:175,name:'Normal Glass 175ml',img:'1'},
{size:187,name:'Small Bottle 187ml',img:'4'},
{size:250,name:'Large Glass 250ml',img:'3'},
{size:750,name:'Bottle 750ml',img:'5'}],
percentage:{min:9,max:16,step:0.5},
calories:0.74};

   var Spirit={name:"Spirits",icon:"spirit",instances:[
{size:25,name:'Single 25ml',img:'1'},
{size:35,name:'Large Single 35ml',img:'2'},
{size:50,name:'Double 50ml',img:'2'},
{size:500,name:'Bottle 750ml',img:'4'}],
percentage:{min:25,max:95,step:2.5},
calories:2.26};

   var Cider={name:"Cider",icon:"cider",instances:[
{size:284,name:'Half Pint 284ml',img:'1'},
{size:300,name:'Bottle 300ml',img:'2'},
{size:440,name:'Can 440ml',img:'3'},
{size:500,name:'Large Can 500ml',img:'4'},
{size:568,name:'Pint 568ml',img:'4'},
{size:750,name:'Bottle 750ml',img:'6'},
{size:1000,name:'Large Bottle 1000ml',img:'6'},
```

Εικόνα 28. κώδικας σε JavaScript για την δημιουργία των αλκοολούχο ποτών και των χαρακτηριστικών τους για να τα επιλέξει ο χρήστης

Το αποτέλεσμα που παρουσιάστηκε στον πελάτη φαίνεται στις παρακάτω εικόνες (Εικόνες 29-34). Ο πελάτης έμεινε ευχαριστημένος και επίμενε να συνεχιστεί το έργο και για τις υπόλοιπες πλατφόρμες λογισμικού.

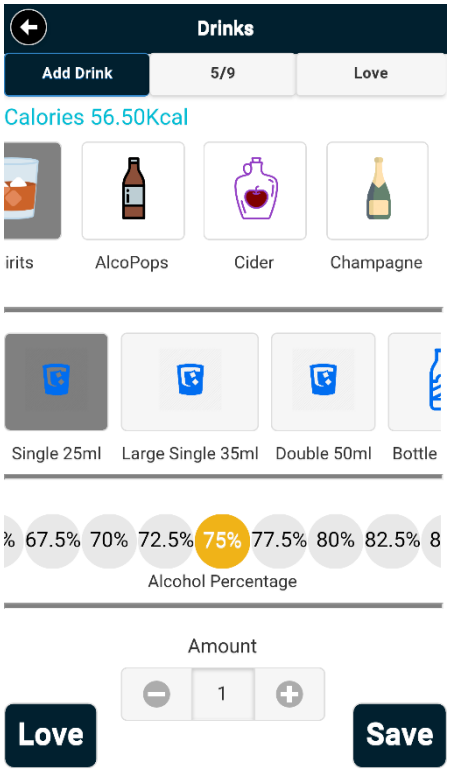


Εικόνα 29. Ημερήσια κατανάλωση αλκοόλ

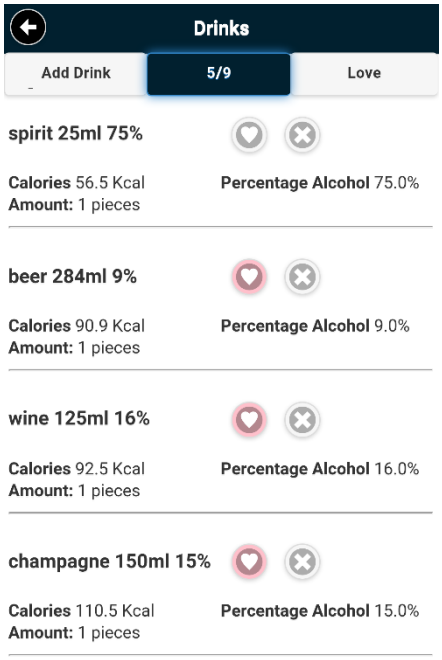


Εικόνα 30. Λίστα ποτών που έχουν καταναλωθεί

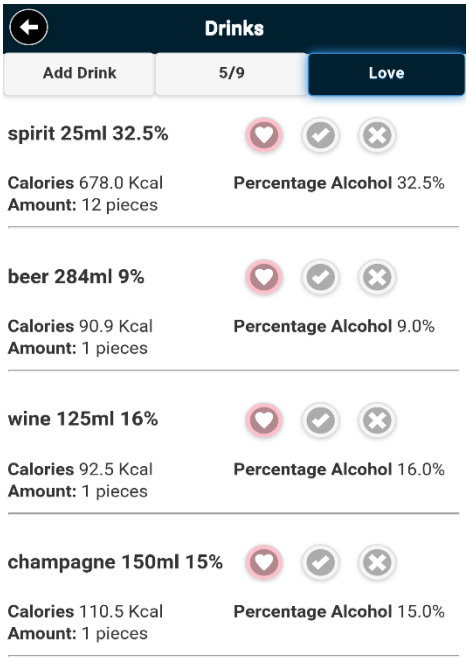
Εικόνα 31. Ιστορικό ποτών χρήστη



Εικόνα 32. Κατηγορίες επίλογων ποτών



Εικόνα 33 . Ημερήσια λίστα καταναλωμένων ποτών



Εικόνα 34. Αγαπημένα ποτά χρήστη

5. Συμπεράσματα και μελλοντικές επεκτάσεις

5.1 Συμπεράσματα

Το συμπέρασμα που προέκυψε στη παρούσα εργασία είναι ότι υβριδικές εφαρμογές υπερισχύουν έναντι των μητρικών όταν η εφαρμογή δεν χρειάζεται να επεξεργαστεί απαιτητικά γραφικά (παιχνίδια) . Η αίτια είναι ότι οι τεχνολογίες διαδίκτυού είναι πιο διευρυμένες και έχουν δημιουργηθεί βιβλιοθήκες και Frameworks που κάνουν την συγγραφή κώδικα σε JavaScript πιο γρήγορη και εύκολη. Επίσης εχουν δημιουργηθεί γραφικά αντικείμενα που δεν απέχουν από τα μητρικά, με συνέπεια η συγγραφή κώδικα γίνεται πιο γρηγορά και το αποτέλεσμα είναι ίδιο με λιγότερους πόρους. Έπειτα, οι υβριδικές εφαρμογές υποστηρίζονται από όλες τις πλατφόρμες λογισμικών χάρις το Cordova, ενώ για τις μητρικές πρέπει να υλοποιηθεί ξεχωριστή εφαρμογή για κάθε διαφορετικό λειτουργικό. Τελευταίο, σε περίπτωση αναβάθμισης στις υβριδικές μπορεί να εκτελείται από το αντίστοιχό Market του λογισμικού είτε από έναν διακομιστή που θα ελέγχει την εφαρμογή ενώ στις μητρικές είναι απαραίτητο η αναβάθμιση να εκτελείται μεσώ Market.

5.2 Μελλοντικές επεκτάσεις

Μελλοντικές επεκτάσεις που θα προχωρήσουν στην υλοποίηση είναι η υποστήριξη της εφαρμογής και από άλλες πλατφόρμες λογισμικών IOS, Windows και Firefox αφού οι πόροι που χρειάζονται είναι λίγοι λόγο της αρχιτεκτονικής Cordova. Επίσης, η εφαρμογή θα συνδέεται με άλλες συσκευές χειρός των πιο διάσημων μοντέλων που προσφέρουν δεδομένα όπως pedometer, καρδιακοί παλμοί κτλ. Τα δεδομένα αυτά θα στέλνονται στην συσκευή του χρήστη θα αποκωδικοποιούνται από την εφαρμογή για να δίνει την δυνατότητα στον χρήστη να παρακολουθεί με ακρίβεια την υγειά του. Τελευταία, η εφαρμογή θα υλοποιηθεί σαν ιστοσελίδα στο διαδίκτυο και θα συνδέεται με τον λογαριασμό του στην συσκευή που έχει εγκαταστήσει την εφαρμογή, ώστε να διευκολύνεται και να έχει πλήρη έλεγχο της υγείας του οπότε και οπού επιθυμεί.

Αναφορές

[1] Know Your Mobile, «The History of Mobile Phones From 1973 To 2008: The Handsets That Made It ALL Happen,» 2015. [Ηλεκτρονικό]. Available: http://www.knowyourmobile.com/nokia/nokia-3310/19848/history-mobile-phones-1973-2008-handsets-made-it-all-happen. [Πρόσβαση 1 Αυγουστος 2016].

[2] Wikipedia, «Κινητό τηλέφωνο,» 2014. [Ηλεκτρονικό]. Available: https://el.wikipedia.org/wiki/%CE%9A%CE%B9%CE%BD%CE%B7%CF%84%CF%8C_%CF%84%CE%B7%CE%BB%CE%AD%CF%86%CF%89%CE%BD%CE%BF. [Πρόσβαση 1 Αύγουστος 2016].

[3] Android (operating system)-Wiki, «Android (operating system),» 2012. [Ηλεκτρονικό]. Available: https://el.wikipedia.org/wiki/Android. [Πρόσβαση 1 Αυγουστος 2016].

[4] Oracle, «JAVA EE,» Oracle, [Ηλεκτρονικό]. Available: http://www.oracle.com/technetwork/java/javaee/overview/index.html. [Πρόσβαση 1 August 2016].

[5] WebRTC, «WebRTC,» [Ηλεκτρονικό]. Available: https://webrtc.org/. [Πρόσβαση 2 August 2016].

[6] Spiros790, «Κινητό τηλέφωνο,» 2014. [Ηλεκτρονικό]. Available: https://el.wikipedia.org/wiki/%CE%9A%CE%B9%CE%BD%CE%B7%CF%84%CF%8C_%CF%84%CE%B7%CE%BB%CE%AD%CF%86%CF%89%CE%BD%CE%BF. [Πρόσβαση 1 Αύγουστος 2016].

[7] Wikipedia, «Tablet computer,» 2011. [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/Tablet_computer. [Πρόσβαση 2 August 2016].

[8] Wikipedia, «Phablet,» 2011. [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/Phablet. [Πρόσβαση 2 August 2016].

[9] appcelerator, «Native vs. HTML5,» appcelerator, 2012. [Ηλεκτρονικό]. Available: https://nhlearninggroup.com/Portals/18/White%20Papers%20and%20eBooks/White%20Paper%20Native%20vs%20Html5.pdf. [Πρόσβαση 2 August 2016].

[10] Apache Cordova, «Overview,» Apache Software Foundation, 2012. [Ηλεκτρονικό]. Available: https://cordova.apache.org/docs/en/latest/guide/overview/index.html. [Πρόσβαση 6 August 2016].

[11] IBM, «Native, web or hybrid,» IBM, 2012. [Ηλεκτρονικό]. Available: ftp://public.dhe.ibm.com/software/pdf/mobile-enterprise/WSW14182USEN.pdf. [Πρόσβαση 3 August 2016].

[12] XMLVM, «XMLVM,» XMLVM, 2011. [Ηλεκτρονικό]. Available: http://xmlvm.org/overview/. [Πρόσβαση 5 August 2016].

[13] Wikipedia, «Xamarin,» Wikipedia, 2011. [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/Xamarin. [Πρόσβαση 5 August].

[14] Appcelerator, «Appcelerator,» Appcelerator, [Ηλεκτρονικό]. Available: http://www.appcelerator.com/mobile-app-development-products/. [Πρόσβαση 5 August 2016].

[15] Microsoft, «Visual Studio,» Visual Studio, 2016. [Ηλεκτρονικό]. Available: https://www.visualstudio.com/. [Πρόσβαση 5 August 2016].

[16] Android Studio, «Android Studio,» Google, [Ηλεκτρονικό]. Available: https://developer.android.com/studio/index.html. [Πρόσβαση 5 August 2016].

[17] WordPress, «Deep Dive into Apache Cordova with Android,» 13 December 2012. [Ηλεκτρονικό]. Available: http://ishanaba.com/blog/tag/how-cordova-works/. [Πρόσβαση 6 August 2016].

[18] Node.js Foundation, «Node,» Node.js Foundation, 2016. [Ηλεκτρονικό]. Available: https://nodejs.org/en/. [Πρόσβαση 5 August 2016].

[19] Author: R. Pramod Nair, «GitHub,» EdelWorks edelworks.in, [Ηλεκτρονικό]. Available: https://github.com/edelworksgithub/SharedPreferences. [Πρόσβαση 5 August 2016].

[20] GitHub, «cordova-plugin-stepcounter,» GitHub, 2016. [Ηλεκτρονικό]. Available: https://github.com/texh/cordova-plugin-stepcounter. [Πρόσβαση 05 August 2016].

[21] Hewlett-Packard, «The Evolutionary Development Model for Software,» Hewlett-Packard, 1996. [Ηλεκτρονικό]. Available: http://www.hpl.hp.com/hpjournal/96aug/aug96a4.pdf. [Πρόσβαση 7 August 2016].

[22] Π. Λουρίδας, «Η Διεργασία Ανάπτυξης Λογισμικού,» [Ηλεκτρονικό]. Available: http://www0.dmst.aueb.gr/louridas/lectures/dais/process/process.html. [Πρόσβαση 7 August 2016].

[23] Μ. Νικολαϊδου, «Ανάλυση Συστημάτων και Τεχνολογία Λογισμικού,» [Ηλεκτρονικό]. Available: https://eclass.hua.gr/modules/document/file.php/DIT187/%CE%A0%CE%91%CE%A1%CE%9F%CE%A5%CE%A3%CE%99%CE%91%CE%A3%CE%95%CE%99%CE%A3/se01_introduction.pdf. [Πρόσβαση 7 August 2016].

[24] DevExpress, «DevExtreme,» [Ηλεκτρονικό]. Available: https://js.devexpress.com/WebDevelopment/. [Πρόσβαση 7 August 2016].

[25] JQuery , «JQuery,» 2016. [Ηλεκτρονικό]. Available: https://jquery.com/. [Πρόσβαση 7 August 2016].

[26] JQuery, «JQuery mobile,» 2016. [Ηλεκτρονικό]. Available: https://jquerymobile.com/. [Πρόσβαση 7 August 2016].

[27] Boostrap, MIT, «Boostrap,» 2016. [Ηλεκτρονικό]. Available: http://getbootstrap.com/. [Πρόσβαση 7 August 2016].

[28] Sly, «Sly,» [Ηλεκτρονικό]. Available: http://darsa.in/sly/. [Πρόσβαση 7 August 2016].

[29] T. BANAS, «How to Convert Pedometer Steps to Calories,» 2014. [Ηλεκτρονικό]. Available: http://www.livestrong.com/article/238020-how-to-convert-pedometer-steps-to-calories/. [Πρόσβαση 07 August 2016].

[30] FatSecret, «The FatSecret Platform API,» [Ηλεκτρονικό]. Available: https://platform.fatsecret.com/api/. [Πρόσβαση 7 August 2016].