Πτυχιακή Εργασία

Τμήμα Μηχανικών Πληροφορικής

Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Τίτλος Πτυχιακής Εργασίας

Υλοποίηση εφαρμογής για αγορά προϊόντων από κατάστημα με τη χρήση κινητού τηλεφώνου τύπου Android

Φοιτητές:

Βογιατζάκης Παναγιώτης ΑΜ:1842

Πατεράκης Μηνάς ΑΜ:2010

Εισηγητής:

Παναγιωτάκης Σπύρος

1 Περιεχόμενα

Πίνακας Εικόνων

Σύνοψη

Η παρούσα πτυχιακή εργασία έχει ως στόχο την μελέτη της τεχνολογίας του κινητού πορτοφολιού (mobile wallet) και την ανάπτυξη μια εφαρμογής σε λειτουργικό σύστημα Android η οποία θα λειτουργεί σαν ένα κινητό πορτοφόλι για ένα υποτιθέμενο κατάστημα. Η εφαρμογή θα χωρίζεται σε δύο μέρη. Η πλευρά του χρήστη-πελάτη θα μπορεί να σαρώνει barcodes ή/και qr codes των προϊόντων του καταστήματος μέσα από την κάμερα του κινητού τους τηλεφώνου και να τα προσθέτει ένα ένα στο καλάθι αγορών του, ενημερώνοντας ταυτόχρονα και την βάση δεδομένων που θα βρίσκεται σε online server. Για την πληρωμή του ο χρήστης-πελάτης θα συνδέεται με το PayPal όπου θα ολοκληρώνει τη συναλλαγή του χωρίς να χρειάζεται να χρησιμοποιήσει χρήματα ή πιστωτική κάρτα και μετά θα μεταφέρεται πάλι στην εφαρμογή όπου θα εμφανίζεται η απόδειξη πληρωμής. Αργότερα, ο χρήστης-πελάτης θα μπορεί να δει το ιστορικό των παραγγελιών του και να αλλάξει διάφορες ρυθμίσεις για τις συναλλαγές του με το κατάστημα. Οι διαχειριστές του καταστήματος θα μπορούν να προσθέτουν νέα προϊόντα και να τα επεξεργάζονται αλλάζοντας φωτογραφία, τιμή, τεμάχια στα διάφορα προϊόντα.

Abstract

Through this thesis we aim to study the mobile wallet technology and to develop a mobile wallet based Android application. This application will consist of two parts. The users-clients will be able to login inside the application and scan a product's barcode and/or qr code through their mobile phone's camera and add it to their shopping cart, updating the shop's online database in the process. When the users-clients want to complete the payment, they will get transferred to a PayPal environment to initiate the money transaction without having to physically use money or a credit card. When the transaction is completed the user will get transferred back to the application where they can view the payment's invoice along with other payment information. The users will have other options as well such as viewing their payment history and changing payment/user settings, amongst others. The administrators of this application will have the ability to add/edit/delete products and edit their settings which include uploading a product's photograph, editing a product's name, price, barcode and quantity.

1. Στόχος της πτυχιακής εργασίας

Το αντικείμενο της πτυχιακής αυτής εργασίας είναι η υλοποίηση μιας εφαρμογής που να μπορούν οι χρήστες να αγοράζουν προϊόντα μέσω σάρωσης barcodes με το κινητό τους τηλέφωνο σαν να βρίσκονται σε ένα κατάστημα τύπου σούπερμαρκετ.

Αρχικά, θα περιγράψουμε την τεχνολογία του κινητού πορτοφολιού (digital wallet) πάνω στην οποία βασίζεται και η εφαρμογή μας. Θα μάθουμε πως ξεκίνησαν, σε ποιο σημείο βρίσκονται τώρα και το σημαντικότερο, το μέλλον των κινητών πορτοφολιών καθώς και τους σημαντικότερους παρόχους αυτής της σύγχρονης υπηρεσίας.

Αργότερα, περιγράφεται αρχικά η πλατφόρμα Android. Δίνονται πληροφορίες για την ιστορία του λειτουργικού αυτού συστήματος, την εξέλιξή του, τα χαρακτηριστικά του, τις λειτουργίες του και τις δυνατότητες του.

Ανοίγοντας την εφαρμογή βλέπουμε το λογότυπό της και σέρνοντας το βελάκι προς τα επάνω εμφανίζονται τα πεδία του ονόματος (username) και του κωδικού (password). Ανάλογα με το ποιος τα συμπληρώνει εμφανίζει και το κατάλληλο μενού στη συνέχεια.

Αν τα πεδία τα συμπληρώσει ο διαχειριστής (admin) της εφαρμογής, με το κατάλληλο όνομα και κωδικό, θα εμφανιστεί ένα μενού από το οποίο θα μπορεί να προσθέσει ή να αφαιρέσει ένα προϊόν, να αλλάξει το όνομα, τον κωδικό και την περιγραφή των προϊόντων και γενικότερα να διαχειριστεί όπως θέλει τα προϊόντα. Επίσης, υπάρχει και η επιλογή αποσύνδεσης από το λογαριασμό του (logout).

Αν τα πεδία τα συμπληρώσει ένας χρήστης (user) της εφαρμογής θα εμφανιστεί ένα μενού από το οποίο θα μπορεί να πληροφορηθεί για τη λειτουργία της εφαρμογής και να κατατοπιστεί βήμα-βήμα τι μπορεί να κάνει με την εφαρμογή που έχει στα χέρια του. Θα μπορεί να αλλάξει κάποια χαρακτηριστικά του προφίλ του, να δει το ιστορικό των αγορών του, να σκανάρει το κωδικό από το προϊόν που επιθυμεί και να το προσθέσει στο καλάθι, να μπει σε αυτό από όπου θα μπορεί να προχωρήσει στην πληρωμή των προϊόντων με κάποιο ηλεκτρονικό λογαριασμό Paypal . Τέλος, υπάρχει κουμπί που αποσυνδέεται ο χρήστης (logout).

Το κίνητρο αυτής της πτυχιακής εργασίας είναι να βοηθήσει τον χρήστη της συγκεκριμένης εφαρμογής στις αγορές του σε ένα κατάστημα, δίνοντάς του τη δυνατότητα να αγοράσει προϊόντα με τη χρήση κάποιου Paypal ηλεκτρονικού λογαριασμού χωρίς να χρειάζεται να περιμένει σε ουρές ταμείων και χωρίς να κρατάει μαζί του χρήματα ή χρεωστικές κάρτες. Αυτό βοηθάει επίσης και τον ιδιοκτήτη του καταστήματος να απασχολεί λιγότερο προσωπικό στην επιχείρησή του άρα έχει λιγότερα λειτουργικά έξοδα! Επίσης, πιστεύουμε ότι ο έμπορος θα έχει περισσότερους πελάτες στο κατάστημα του παρέχοντας τους μια αξιόπιστη υπηρεσία πληρωμής και ασφαλέστερων συναλλαγών.

Επίσης, στόχος της πτυχιακής αυτής είναι να εξοικειωθούμε όσο το δυνατό περισσότερο με τη γλώσσα προγραμματισμού Android, την ανάπτυξη και σχεδίαση ενός user interface για μια Android εφαρμογή, την ανάπτυξη και παραμετροποίηση μιας βάσης δεδομένων, την χρησιμοποίηση της κάμερας του κινητού τηλεφώνου για ανάγνωση barcodes και την ένωση της εφαρμογής με τη βιβλιοθήκη του PayPal για την προσομοίωση των συναλλαγών.

Πιο συγκεκριμένα, στο κεφάλαιο 2 περιγράφουμε τις τεχνολογίες του κινητού πορτοφολιού (mobile wallet).

Στο κεφάλαιο 3 κάνουμε μια εισαγωγή στο Android λειτουργικό σύστημα περιγράφοντας διάφορα χαρακτηριστικά, ιστορικά στοιχεία και (προγραμματιστικές) έννοιες του.

Στο κεφάλαιο 4 αναφέρουμε το σχέδιο δράσης μαζί με το state of the art, το χρονοδιάγραμμα της πτυχιακής εργασίας, το λογισμικό που χρησιμοποιήθηκε καθώς και τις εισαγωγικές διαδικασίες για την έναρξη υλοποίησης της εφαρμογής.

Στο κεφάλαιο 5 αναλύουμε της δημιουργία και υλοποίηση της βάσης δεδομένων με μια εκτενής παρουσίαση κάθε πίνακα και των χαρακτηριστικών του καθώς και με ένα μοντέλο οντοτήτων-συσχετίσεων.

Στο κεφάλαιο 6 αναλύουμε την υλοποίηση της εφαρμογής που δημιουργήσαμε παρέχοντας εικόνες μέσα από την εφαρμογή και αποκόμματα πηγαίου κώδικα όπου χρειάζεται.

2. Mobile Wallets

2.1. Εισαγωγή στα Mobile wallets

Το mobile wallet ή αλλιώς κινητό πορτοφόλι είναι ένα εικονικό πορτοφόλι που αποθηκεύει διάφορα στοιχεία πληρωμών τα οποία μας επιτρέπουν να κάνουμε online αγορές ή αγορές (μέσα) σε ένα κατάστημα χωρίς να χρησιμοποιήσουμε την «φυσική» πιστωτική μας κάρτα.

Παρ'ότι διαφορετικά mobile wallets έχουν βγει στην αγορά τα τελευταία χρόνια, πολλά απ'αυτά βρίσκονται στο αρχικό στάδιο της λειτουργίας τους. Αλλά αυτό δεν σημαίνει ότι δεν υπάρχουν ήδη αρκετές υπηρεσίες που μπορούν οι χρήστες τους να επωφεληθούν.

Αρχικά, ενώ η διείσδυση των mobile wallets ήταν επικεντρωμένη στη διανομή κουπονιών-προσφορών και εκπτωτικών καρτών, τώρα τελευταία τα mobile wallet παρουσιάζουν ποικίλες δυνατότητες εκτός των προαναφερθέντων όπως δυνατότητες για διαχείριση οικονομικών υπηρεσιών, επιλογές πληρωμών και mobile banking.
Ας αναφέρουμε μερικά βασικά χαρακτηριστικά ενός mobile wallet:

- Διαχείριση από τον καταναλωτή ενός διευρυμένου χαρτοφυλακίου των κινητών υπηρεσιών πληρωμών από διαφορετικούς παρόχους
- Διευκόλυνση των πληρωμών (επιλογή και έλεγχος ταυτότητας) για τα αγαθά, τις υπηρεσίες ή τις πληρωμές πρόσωπο με πρόσωπο.
- Αποθήκευση των εισιτηρίων, καρτών επιβίβασης που μπορεί να παρουσιαστούν σε ένα σημείο ελέγχου.
- Υπηρεσίες ενιαίου χώρου αποθήκευσης για προγράμματα πιστότητας και κουπονιών.
- Αποθήκευση διαπιστευτηρίων για την εύκολη και βολική αναγνώριση και ταυτοποίηση σε κάποιο σημείο πληρωμής.
- Αποθήκευση προσωπικών πληροφοριών, όπως τη διεύθυνση παράδοσης για να διευκολυνθούν οι online αγορές ενός καταναλωτή.

Εικόνα 1.1: Χρήση κινητού πορτοφολιού σε ένα supermarket

2.2. Σημεία Πώλησης

Σημείο πώλησης ή αλλιώς POS (Point of Sale), είναι ο τόπος όπου ολοκληρώνεται η συναλλαγή σε ένα κατάστημα λιανικής. Είναι το σημείο στο οποίο ένας πελάτης κάνει μια πληρωμή στον έμπορο, σε αντάλλαγμα για τα αγαθά ή τις υπηρεσίες. Στο σημείο πώλησης ο λιανοπωλητής θα υπολογίζει το ποσό που οφείλεται από τον πελάτη και παρέχονται οι ανάλογες επιλογές για τον πελάτη να καταβάλει την πληρωμή. Ο έμπορος θα πρέπει επίσης κατά κανόνα να εκδίδει απόδειξη για τη συναλλαγή.

Το εκάστοτε POS χρησιμοποιεί, σε διάφορες βιομηχανίες λιανικής, προσαρμοσμένο υλικό λογισμικό σύμφωνα με τις απαιτήσεις του κάθε εμπόρου - καταστήματος. Οι έμποροι λιανικής πώλησης ενδέχεται να χρησιμοποιούν ζυγαριές, scanners, ηλεκτρονικά μέσα και χειροκίνητες ταμειακές μηχανές, τερματικά EFTPOS, οθόνες αφής και οποιαδήποτε άλλη ποικιλία του υλικού και λογισμικού που διατίθενται για χρήση με POS **[εικόνα 1.2].** Για παράδειγμα, ένα μανάβικο χρησιμοποιεί μία ζυγαριά στο σημείο πώλησης, ενώ τα μπαρ και τα εστιατόρια χρησιμοποιούν ειδικά προσαρμοσμένο λογισμικό για να παραχθεί η ανάλογη υπηρεσία όταν ένα πελάτης παραγγέλνει ένα γεύμα ή ποτό.

Τα σύγχρονα σημεία πώλησης αναφέρονται συχνά ως σημεία υπηρεσιών γιατί δεν είναι μόνο σημεία πώλησης αλλά και σημεία επιστροφής προϊόντων ή παραγγελιών κάποιου πελάτη.

Επίσης, περιλαμβάνουν προηγμένες υπηρεσίες για να ανταποκριθούν στις διάφορες λειτουργίες μια επιχείρησης όπως διαχείριση οικονομικών στοιχείων και υπολογισμών, διαχείρηση αποθεμάτων, επεξεργασία στοιχείων σε προϊόντα-υπηρεσίες και άλλα. Παλιότερα γινόταν όλη η καταχώρηση των πληροφοριών χειροκίνητα το οποίο είχε ως συνέπεια διάφορα λάθη στις καταχωρήσεις.

Εικόνα 1.2: Τυπικός εξοπλισμός ενός σημείου πώλησης

Ένα σύστημα ενός σημείου λιανικής πώλησης συνήθως περιλαμβάνει μια ταμειακή μηχανή (η οποία τα τελευταία χρόνια περιλαμβάνει έναν υπολογιστή , οθόνη (αφής) , συρτάρι μετρητών , εκτυπωτή παραλαβής , οθόνη πελάτη και barcode scanner) και η πλειοψηφία των συστημάτων περιλαμβάνουν επίσης έναν αναγνώστη χρεωστικής / πιστωτικής κάρτας. Ένα τέτοιο σύστημα ενδέχεται να περιλαμβάνει ένα μεταφορικό ιμάντα, κλίμακα βάρους, ολοκληρωμένο σύστημα επεξεργασίας πιστωτικών καρτών, μία συσκευή καταγραφής υπογραφής και μια συσκευή pin pad (για τον κωδικό pin) για τον πελάτη.

Όλο και περισσότερες οθόνες σημείων πώλησης χρησιμοποιούν τεχνολογία οθόνης αφής για ευκολία στη χρήση. Το λογισμικό του συστήματος ενός σύγχρονου σημείου

πώλησης μπορεί τυπικά να χειριστεί τον όγκο των πελατών με βάση τις λειτουργίες, όπως οι πωλήσεις, επιστροφές, ανταλλαγές, κάρτες δώρων, προγράμματα πιστότητας πελατών, προσφορές, εκπτώσεις και άλλα. Αυτό το λογισμικό μπορεί να επιτρέψει λειτουργίες διακίνησης συναλλάγματος και πολλαπλούς τύπους πληρωμών .

Για σημεία πώλησης σε επιχειρήσεις όπως ξενοδοχεία, εστιατόρια και μπαρ παρέχονται ειδικά λογισμικά συστήματα που παρέχουν υπηρεσίες μισθοδοσίας και τήρηση βιβλίων. Ακόμα, τυπικά λογισμικά εστιατορίων είναι σε θέση να δημιουργήσουν και να εκτυπώσουν εντολές εκτύπωσης προετοιμασίας παραγγελιών στην κουζίνα ή το μπαρ και αναφορές-στατιστικά αυτών.

2.3. Ασύρματο σύστημα πώλησης-υπηρεσιών

Μια πιο πρόσφατη καινοτομία είναι το ασύρματο σύστημα υπηρεσιών (Wireless POS) που δείχνει την χρήση ασύρματων συσκευών για την διευκόλυνση των υπηρεσιών σε μια επιχείρηση. Το ασύρματο σύστημα πώλησης, τυπικά, έχει τη βάση του σε ένα κεντρικό σταθμό-διακομιστή που εκεί συνδέονται άμεσα μέσω του δικτύου μία ή περισσότερες φορητές συσκευές για την ασύρματη επικοινωνία τους.

Εικόνα 1.3: Ασύρματο σημείο πώλησης σε εστιατόριο

Το ασύρματο σύστημα υπηρεσιών μπορεί να κάνει ότι και ένα κανονικό σύστημα πώλησης-υπηρεσιών δηλαδή καταχώρηση προϊόντων, παραγγελίες πελατών, διαχείρηση πιστωτικών καρτών, διαχείρηση αποθεμάτων και άλλα.

Σαν τεχνολογία, μπορεί να μειώσει δραματικά το κόστος με:

- Εξάληψη χρόνου αναμονής για πωλήσεις-υπηρεσίες.
- Αύξηση παραγωγικότητας των εργαζομένων της εκάστοτε επιχείρησης.
- Μείωση κόστους εγκατάστασης σε σχέση με παλαιότερες εγκαταστάσεις τέτοιων συστημάτων.
- Ανταλλαγή υπηρεσιών on the go.
- Αποδοχή πληρωμών με smartphone ή και tablet.
- Εξαργύρωση προσφορών χωρίς συνεχής χρήση χαρτιού και κουπονιών.
- Προσέλκηση νέων πελατών λόγω κορυφαίας τεχνολογίας.

2.4. Εισαγωγή στη τεχνολογία NFC

Το NFC είναι συντομογραφία των λέξεων Near Field Communications. Μια έννοια που έχει πάρει μεγάλες διαστάσεις στον τομέα των mobile πληρωμών. Το NFC είναι μια νεότερη ασύρματη τεχνολογία που επιτρέπει στις συσκευές να επικοινωνούν μεταξύ τους σε μικρές αποστάσεις (συνήθως εώς και 10 εκατοστά).

Εικόνα 1.4: Χρησιμοποίηση NFC τεχνολογίας σε μικρή απόσταση

Η επικοινωνία δεν χρειάζεται να συμβεί όμως μεταξύ των δύο φορητών συσκευών όπως τα δύο τηλέφωνα. Μπορεί να λειτουργήσει με μια κινητή συσκευή και έναν στόχο κάποιας μορφής - για παράδειγμα, ένα σύστημα πώλησης (point-of-sale) στο ταμείο ενός καταστήματος ή ακόμα και κάτι τόσο απλό όσο μια ετικέτα, αφίσα, αυτοκόλλητο ή αλλιώς μια κάρτα με ένα τσιπ NFC ενσωματωμένο. Στην περίπτωση των απλών αυτών στόχων, δεν απαιτούνται μπαταρίες για να τροφοδοτήθούν τα NFC τσιπς. Αντ 'αυτού, τα τσιπ είναι σε μια παθητική κατάσταση, όπου περιμένουν να ενεργοποιηθούν από μια άλλη συσκευή που μπορεί να δημιουργήσει μια RF (ραδιοσυχνότητα).

Εικόνα 1.5: Τομείς τεχνολογίας χρησιμοποίησης του NFC

Παρά το γεγονός ότι η χρήση του όρου είναι σχετικά νέα, το NFC παράγεται από μια τεχνολογία του έτους 1940 που ονομάζεται RFID (Radio Frequency Identification). Και οι δύο αυτές τεχνολογίες είναι σχεδιασμένες ως αυτόνομα microchips που μπορούν να μεταδίδουν πληροφορίες ασύρματα σε περιορισμένες αποστάσεις. Το RFID είχε σχεδιαστεί για να παρέχει πληροφορίες σε απόσταση λίγων μέτρων, ενώ το microchip NFC έχει σχεδιαστεί για απόστασεις μόλις λίγων εκατοστών.

Το NFC και το RFID μοιράζονται την ίδια εννοιολογική αρχή αλλά κατασκευάστηκαν για διαφορετικούς σκοπούς. Το RFID εξελίχθηκε για να διαμοιράζει πληροφορίες στην εκάστοτε τεχνολογία που το χρησιμοποιεί. ΤΟ NFC παρέχει ως επί το πλείστον πληροφορίες στο άτομο που του τις ζητάει.

Πιο αναλυτικά, το RFID χρησιμοποιείται σε εφαρμογές όπως παρακολούθηση αποθεμάτων, ετικέτες διοδίων, ελεγκτές αποσκευών, ανάγνωση μετρητών και επισήμανση προϊόντων. Από την άλλη, το NFC χρησιμοποιείται σε εφαρμογές όπως κάρτες πρόσβασης σε ένα κτίριο, πιστωτικές κάρτες, ταυτότητες, κάρτες υγείας, τα εισητήρια μεταφορών, προωθητικές ενέργειες μάρκετινγκ και συντομεύσεις URL.

Το σύνολο των καταναλωτών έμαθε για τη τεχνολογία NFC το 2005, όταν άρχισαν οι τράπεζες να ενσωματώνουν τσιπάκια στις πιστωτικές κάρτες. Τα τσιπάκια αυτά σχεδιάστηκαν για να επιταχύνουν την διεργασία της πληρωμής και να μειώσουν τον κίνδυνο της κλωνοποίησης καρτών, ένα πρόβλημα που αντιμετώπιζαν πιστωτικές κάρτες με μαγνητικές λωρίδες. Οι πιστωτικές κάρτες με NFC τεχνολογία σχεδιάστηκαν για να επικοινωνούν ασύρματα με νέας γενιάς τερματικά πληρωμών.

2.4.1. Τύποι NFC chips

Μία κατηγορία είναι τα παθητικά microchips. Αυτά τα τσιπ μεταδίδουν πληροφορίες αποκλειστικά όταν διεγείρονται από ένα εξωτερικό NFC reader. Αυτά τα παθητικά microchips δεν απαιτούν ενσωματωμένη πηγή ενέργειας, επειδή δημιουργούν τη δική τους ενέργεια, χρησιμοποιώντας τα ραδιοκύματα που παράγονται από το εκάστοτε NFC reader. Τα παθητικά microchips έχουν προγραμματιστεί για να έχουν πρόσβαση αυτόματα στην ROM (read-only memory) και μετά να μεταδώσουν ROM περιεχόμενα στον αντίστοιχο reader.

Μια άλλη κατηγορία είναι τα ενεργά microchips. Αυτά τα τσιπ μπορούν να έχουν μια αμφίδρομη επικοινωνία με εξωτερικές συσκευές ανάγνωσης-εγγραφής NFC. Η διαφορά τους με τα παθητικά τσιπς είναι ότι μπορούν να λειτουργήσουν είτε ως πομπός είτε ως δέκτης πληροφορίας. Τα ενεργά τσιπς είναι στην ουσία ασύρματα συστήματα (υπολογιστές).

2.4.2. Μελλοντική εξέλιξη για NFC chips

Λόγω της ραγδαίας εξέλιξης, η χρήση των συστημάτων NFC εκτιμάται ότι θα φτάσει στο 1.2 εκατομμύρια μέχρι το 2015. Οι λιανοπωλητές ξεκίνησαν πιλοτικά προγράμματα μέσα στο 2012 όπου ενσωμάτωσαν έξυπνες αφίσες στις πινακίδες των μαγαζιών τους και ανέπτυξαν διάφορες στρατηγικές διαφήμισης εξωτερικού χώρου. Επίσης μια άλλη εξέλιξη είναι οι πληρωμές μέσω κινητού τηλεφώνου. Μέσα σε 4 χρόνια, παραπάνω από το 50% των «έξυπνων κινητών» θα υποστηρίζουν το συγκεκριμένο σύστημα NFC.

Εικόνα 1.6: Χρήση NFC τεχνολογίας στο παιχνίδι «Angry Birds Magic».

Μια καινοτομία που πρόκειται να παρουσιαστεί είναι στο καινούργιο παιχνίδι "angry birds magic" όπου θα ενσωματώσει τις τεχνολογίες του NFC και GPS. Το παιχνίδι ξεκινά ως συνήθως με πέντε επίπεδα, αλλά η χρήση NFC ξεκινά μετά την ολοκλήρωση αυτών. Θα μπορείτε να ξεκλειδώσετε επιπλέον επίπεδα και να συνεχίσετε σε αυτά το παιχνίδι, μόνο αν βρείτε ένα τηλέφωνο με το NFC ενεργοποιημένο και «ενώσετε» τις συσκευές μαζί.

Άλλες εφαρμογές που συζητούνται είναι εφαρμογές που αφορούν την παρακολούθηση της υγείας, όπως επίσης και τρόπους ώστε να εφοδιάσουν τα «έξυπνα σπίτια» με πόρτες που όχι μόνο ξεκλειδώνουν μέσω ενός NFC συστήματος, αλλά μπορούν επίσης να επικοινωνήσουν με άλλα αντικείμενα στο σπίτι.

2.5. Πληρωμές μέσω κινητού τηλεφώνου (Mobile Payments)

Η επιθυμία του ανθρώπου να επικοινωνήσει είναι τόσο παλιά όσο ο ίδιος ο άνθρωπος. Η εφεύρεση των συσκευών επικοινωνίας, όπως ο τηλέγραφος , το τηλέφωνο , και ιδιαίτερα το κινητό τηλέφωνο έχουν επηρεάσει το πώς οι άνθρωποι ζουν με τρόπους που σχεδόν κανείς δεν μπορούσε να προβλέψει .

Μια άλλη απαίτηση σχεδόν τόσο παλιά όσο και η ανάγκη επικοινωνίας είναι η ανάγκη οι άνθρωποι να ανταλλάξουν πράγματα με αξία.

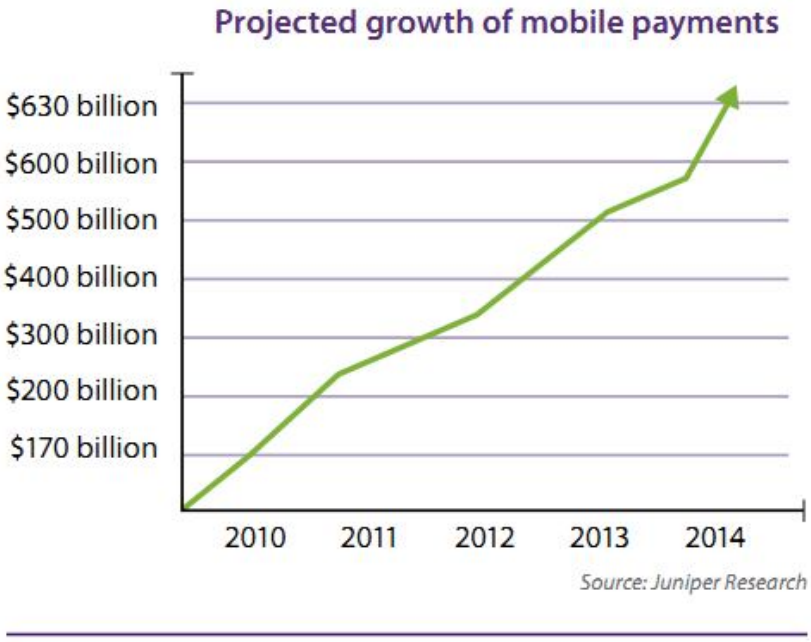
Ακριβώς όπως οι μέθοδοι της επικοινωνίας έχουν εξελιχθεί με την πάροδο του χρόνου , έτσι έχουν αναπτυχθεί και οι μεθόδοι και τα εργαλεία του εμπορίου , από την ανάπτυξη κοινώς αποδεκτών νομισμάτων και τη δημιουργία των τραπεζών , εώς την εξέλιξη των πολύπλοκων και ισχυρών δικτύων πληρωμής που μπορούν να επικυρώσουν και να επιτρέψουν συναλλαγές σε απόσταση χιλιάδων χιλιομέτρων μέσα σε λίγα δευτερόλεπτα .

Οι επικοινωνίες και το εμπόριο βιομηχανιών με τις τεχνολογίες υποστήριξής τους, έχουν εξελιχθεί δραματικά με την πάροδο του περασμένου αιώνα, κατάσταση που αλλάζει πιο ραγδαία τα τελευταία 40 χρόνια. Συνδυάζοντας τις πιο πρόσφατες τεχνολογίες από τις βιομηχανίες κινητών τηλεφώνων και του ηλεκτρονικού εμπορίου, παίρνουμε μια υπηρεσία που παρέχει νέες δυνατότητες, όπως και την ελευθερία να πραγματοποιούνται συναλλαγές με τρόπους που διαφορετικά δεν θα γινόντουσαν. Αυτό η υπηρεσία είναι γνωστή ως πληρωμές μέσω κινητού τηλεφώνου.

Τα άλματα της τεχνολογίας που έχουν γίνει την τελευταία δεκαετία έχουν μετατρέψει τα κινητά τηλέφωνα από απλά τηλέφωνα σε ισχυρά υπολογιστικά εργαλεία. Σήμερα, οι

χρήστες των συσκευών αυτών μπορούν να τις χρησιμοποιήσουν για πολλές εφαρμογές στην καθημερινότητα τους όπως να βρουν ένα κοντινό κατάστημα ή ένα βενζινάδικο.

Οι πληρωμές μέσω κινητού τηλεφώνου έφτασαν τα 170 δισεκατομμύρια δολλάρια παγκοσμίως το 2010. Μέσα στο 2014, υπολογίζεται ότι αυτός ο αριθμός θα φτάσει τα 630 δισεκατομμύρια δολλάρια, σύμφωνα με την Juniper Research, έναν πάροχο επιχειρησιακών πληροφοριών στην Ευρώπη. Στις Ηνωμένες Πολιτείες Αμερικής οι πληρωμές μέσω κινητού τηλεφώνου θα φτάσουν τα 214 δισεκατομμύρια δολλάρια το 2015, με το 77 τις εκατό να είναι πληρωμές σε καταστήματα, σύμφωνα με την εταιρία έρευνας Aite Group στη Βοστώνη. Σαν συμπέρασμα, οι επιλογές στον κόσμο των πληρωμών μέσω κινητού τηλεφώνου είναι πολλές και (θα) αλλάζουν με πολλούς παράγοντες.**[Βιβλιογραφία: 18]**



Εικόνα 1.7: Γράφημα εκτίμησης πληρωμών μέσω κινητού τηλεφώνου

2.6. Χαρακτηριστικά πληρωμών μέσω κινητού τηλεφώνου

Μια υπηρεσία για πληρωμές μέσω κινητού τηλεφώνου προκειμένου να γίνει αποδεκτή στην αγορά ως ένα τρόπος πληρωμών πρέπει να πληρεί κάποιες προϋποθέσεις όπως:

Απλότητα και ευχρηστία: Η υπηρεσία της πληρωμής αυτής πρέπει να είναι απολύτως φιλική προς το χρήστη και η εκάστοτε υπηρεσία να μπορεί να προσαρμόζεται στις απαιτήσεις των χρηστών της.

Παγκοσμιότητα: Η υπηρεσία πρέπει να μπορεί να παρέχει συναλλαγές μεταξύ πελάτη σε πελάτη (customer to customer), επιχείρηση σε πελάτη (business to customer) και επιχείρηση σε επιχείρηση (business to business). Η κάλυψη της υπηρεσίας θα πρέπει να είναι εγχώρια, περιφερειακή ή ακόμα καλύτερα και παγκόσμια.

Διαλειτουργικότητα: Η ανάπτυξη της υπηρεσίας θα πρέπει να βασίζεται σε πρότυπα και ανοιχτές τεχνολογίες για να μπορεί η υπηρεσία αυτή να αλληλεπιδρά πιο εύκολα με άλλα παρόμοια συστήματα κινητών πληρωμών.

Ασφάλεια και προστασία απορρήτου: Ο πελάτης θα πρέπει να μπορεί να εμπιστεύεται την εκάστοτε υπηρεσία για να κάνει τις συναλλαγές του και να διασφαλίζεται ότι δεν καταγράφονται στοιχεία ιδιωτικής ζωής των πελατών που δεν θα έπρεπε. Οι πληρωμές μέσω κινητού τηλεφώνου πρέπει να είναι όσο το δυνατότερο ανώνυμες για λόγους πρόληψης υποκλοπών σε συναλλαγές. Το σύστημα της υπηρεσίας πρέπει να έχει προστασία από επιθέσεις χάκερς ή άλλων κακοποιών.

Κόστος: Μια υπηρεσία κινητών πληρωμών δεν πρέπει να είναι ακριβότερη στην υλοποίηση, εγκατάσταση και εξοπλισμό από τις ήδη υπάρχουσες.

Ταχύτητα: Μια υπηρεσία κινητών πληρωμών πρέπει να παρέχει μια συνετή ταχύτητα στις πληρωμές που γίνονται για να ικανοποιήσει τον όγκο των συναλλαγών μεταξύ πελατών και εμπόρων.

2.7. Mobile Banking

Το Mobile Banking είναι ένα σύστημα που επιτρέπει στους πελάτες που είναι καταχωρημένοι σε μια τράπεζα, δηλαδή έχουν λογαριασμό στη τράπεζα αυτή, να προβούν σε συναλλαγές μέσω μιας κινητής συσκευής-τηλεφώνου.

2.7.1. Interactive Voice Response (IVR)

Αν καλέσατε ποτέ τον εκδότη της πιστωτικής σας κάρτας και σας έλεγε «Για τα Αγγλικά, πιέστε 1», «για τα στοιχεία του λογαριασμού, πατήστε 2» κτλ τότε καταλαβαίνετε τι είναι η υπηρεσία της φωνητικής απόκρισης. Οι πελάτες καλούν έναν IVR αριθμό μέσω των κινητών τηλεφώνων τους και χαιρετίζονται από ένα αποθηκευμένο ηλεκτρονικό μήνυμα που ακολουθείται από ένα μενού επιλογών.

Οι πελάτες αργότερα καλούνται να επιλέξουν κάτι πατώντας τον αντίστοιχο αριθμό στο πληκτρολόγιο τους. Ένα λογισμικό διαβάζει τις πληροφορίες με χρήση τεχνολογιών text-to-speech. Το IVR δεν χρειάζεται πάντα ένα κινητό τηλέφωνο για να λειτουργήσει γι'αυτό δεν παρέχει και πολύ προηγμένες υπηρεσίες αλλά χρησιμοποιείται κυρίως για άντληση καταχωρημένων πληροφοριών.

2.7.2. Υπηρεσίες Short Message Service (SMS Banking)

Το SMS Banking είναι ένας τύπος Mobile Banking και παρέχει υπηρεσίες της τράπεζας προς τους πελάτες της, επιτρέποντας τους να διαχειριστούν συγκεκριμένες τραπεζικές υπηρεσίες μέσω κινητών τηλεφώνων με τη χρήση μηνυμάτων (SMS).

Η υπηρεσία αυτή βασίζεται σε ένα σύστημα pull-push μηνυμάτων. Δηλαδή, τα push μηνύματα είναι εκείνα που η τράπεζα διαλέγει να στείλει στον πελάτη, χωρίς εκείνος να κάνει αίτηση για να πάρει την πληροφορία.

Συνήθως, αυτού του είδους τα SMS είναι τύπου μάρκετινγκ ή μηνύματα που ειδοποιούν ένα συμβάν που καταγράφηκε στον λογαριασμό τράπεζας του πελάτη. Το συμβάν αυτό μπορεί να είναι μια μεγάλη λήψη χρημάτων από κάποιο ATM ή κάποια άλλη απότομα μεγάλη συναλλαγή που έγινε με την πιστωτική κάρτα του εκάστοτε πελάτη. Ακόμα, μπορεί να είναι ένα συμβάν ενεργοποίησης κάποιας υπηρεσίας με την αποστολή κάποιου κωδικού ενεργοποίησης.

Τα μηνύματα τύπου pull είναι εκείνα που γίνονται από τον πελάτη προς τον λογαριασμό του στην τράπεζα, χρησιμοποιώντας πάντα το κινητό του τηλέφωνο. Συνήθως τέτοιου τύπου μηνύματα είναι αίτηση παγώματος μιας πιστωτικής κάρτας και ερωτήματα όπως την κατάσταση του λογαριασμού του πελάτη ή ερώτηση για τα επιτόκια καταθέσεων.

Τα πλεονεκτήματα του SMS Banking είναι ότι δουλεύει στην πλειοψηφία των κινητών τηλεφώνων ανεξάρτητα από μοντέλο ή κατασκευαστή. Η αποστολή μηνυμάτων κειμένου είναι σχετικά αποδοτική αφού κοστίζει λιγότερο με διάφορα πακέτα που παρέχουν οι τράπεζες για αποστολή μηνυμάτων κειμένου. Από την άλλη μεριά τα μειονεκτήματα αυτής της υπηρεσίας δεν είναι ενθαρρυντικά. Η υποκλοπή δεδομένων μέσω μηνυμάτων κειμένου είναι σχετικά εύκολη αφού τα μηνύματα κειμένου (SMS) χρησιμοποιούν ανεπαρκή κρυπτογράφηση.

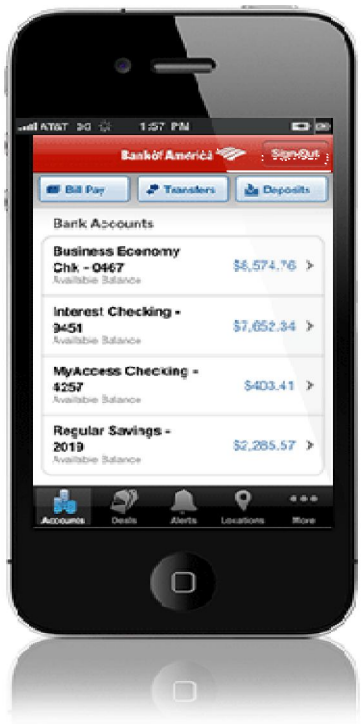
2.7.3. Πάροχοι για Mobile Banking

Οι πάροχοι γίνονται ολοένα και περισσότεροι αλλά εμείς θα αναφέρουμε τους μεγαλύτερους παγκόσμια παρόχους που είναι ευρύτερα αναγνωρισμένοι και καλύτερα ανεπτυγμένοι σε σχέση με άλλους.

2.7.4. Mobile Banking από την Τράπεζα της Αμερικής

Η τράπεζα της Αμερικής (Bank of America) παρέχει υπηρεσίες όπου οποιοδήποτε κινητό τηλέφωνο με πρόσβαση στο Internet μπορεί να χρησιμοποιήσει χωρίς να κατεβάσει επιπλέον λογισμικό. Μερικές από τις υπηρεσίες τους περιλαμβάνουν πρόσβαση στις αποταμιεύσεις, πιστωτικές κάρτες, στοιχεία, δάνεια και άλλα. Επίσης παρέχονται υπηρεσίες εύρεσης καταστημάτων ή ATM με χάρτες και οδηγίες.

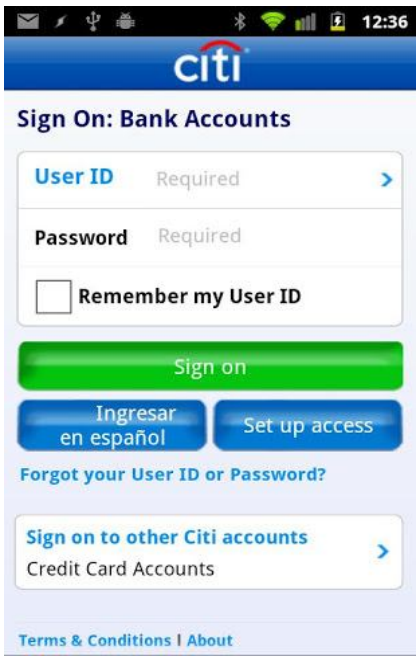
Οι χρήστες των υπηρεσιών αυτών μπορούν επίσης να δημιουργήσουν ειδοποιήσεις που στέλνονται με e-mail ή άλλου τύπου μηνύματα κειμένου για να ειδοποιήσουν τον πελάτη για κάποια λήξη αποπληρωμής ή ενημέρωση για το υπόλοιπο του λογαριασμού του.



Εικόνα 1.8: Εφαρμογή Mobile Banking από την Τράπεζα της Αμερικής

Η Τράπεζα της Αμερικής διαφημίζει τις υπηρεσίες αυτές ως δωρεάν. Αλλά ίσως να υπάρχουν χρεώσεις ανάλογα με την εταιρεία κινητής τηλεφωνίας που χρησιμοποιείται από τον πελάτη.

2.7.5. Citi Mobile από την Citibank



Εικόνα 1.9: Αρχική όψη της εφαρμογής Citi Mobile για Android

Για το Citi Mobile πρέπει ο πελάτης να κατεβάσει την αντίστοιχη εφαρμογή στο τηλέφωνο του. Η διαδικασία αρχίζει με τη σύνδεση των χρηστών στους online λογαριασμούς τους όπου αργότερα στέλνεται ένα μήνυμα στο κινητό τους για το πώς να κατεβάσουν την εφαρμογή και πώς να χρησιμοποιήσουν το κλειδί ενεργοποίησης που τους δόθηκε.

Αφού τα κάνουν αυτά, η εφαρμογή προσαρμόζεται στο κινητό του κάθε πελάτη και παρέχει πληροφορίες συναλλαγών, ιστορικού, και τοποθεσίες που βρίσκονται καταστήματα σχετικά με την Citi. Το Citi Mobile χρησιμοποιεί κρυπτογράφηση 128-bit, όπως όταν κάνουμε συναλλαγές online.

Επίσης, η δωρεάν αυτή υπηρεσία υποστηρίζει κινητά όπως iPhone, iPod touch, Windows Mobile, BlackBerry και φυσικά Android. Πιο αναλυτικά, τα κύρια χαρακτηριστικά που προσφέρει το City Mobile είναι:

- Έλεγχος υπολοίπου στο λογαριασμό
- Μεταφορά χρημάτων μεταξύ των CitiBank λογαριασμών του πελάτη
- Προβολή πρόσφατων συναλλαγών
- Εύρεση του πλησιέστερου ATM / υποκαταστήματος

- Προβολή διαθέσιμων πιστώσεων

2.8. Τύποι πληρωμών μέσω κινητού σε φυσικά καταστήματα

Έχουμε δύο τύπους για πληρωμές μέσω κινητού σε φυσικά καταστήματα, την κοντινή πληρωμή (proximity) και την απομακρυσμένη (remote).

Εικόνα 1.10: Πληρωμή μέσω κινητού τηλεφώνου

2.8.1. Remote Mobile Payments

Απομακρυσμένη (remote) πληρωμή μέσω κινητού είναι όταν χρησιμοποιείται το κινητό τηλέφωνο για τη διεξαγωγή μιας συναλλαγής για ένα αγαθό ή μια υπηρεσία. Υπάρχουν μερικοί διαφορετικοί τρόποι με την οποία η remote πληρωμή μπορεί να πραγματοποιηθεί.

Ένας τρόπος επιτρέπει στους συνδρομητές της κινητής τηλεφωνίας να αγοράσουν κάτι μέσω τηλεφώνου ή μέσω SMS, όπως μια εφαρμογή, ringtone, ή βίντεο, και να χρεώνονται στο λογαριασμό του κινητού τους. Ένας άλλος τρόπος είναι οι συνδρομητές κινητής τηλεφωνίας να πληρώσουν άμεσα, μέσω μιας ιστοσελίδας-εφαρμογής κινητού τηλεφώνου με τη χρήση μεθόδων όπως το PayPal, πιστωτικών ή χρεωστικών καρτών.

2.8.2. Proximity Mobile Payments

Η κοντινή (Proximity) πληρωμή μέσω κινητού βασίζεται στην τεχνολογία (NFC) και επιτρέπει στους καταναλωτές να αγοράζουν αγαθά και υπηρεσίες άμεσα στο σημείο πώλησης κάποιου καταστήματος χρησιμοποιώντας το κινητό τους τηλέφωνο. Οι καταναλωτές απλά φέρνουν το κινητό τους τηλέφωνο κοντά σε έναν αναγνώστη (NFC Reader), και το ποσό που έχουν αγοράσει αφαιρείται από έναν προπληρωμένο λογαριασμό, έναν λογαριασμό του κινητού, ή ένα τραπεζικό λογαριασμό.

2.8.3. NFC Mobile Payments

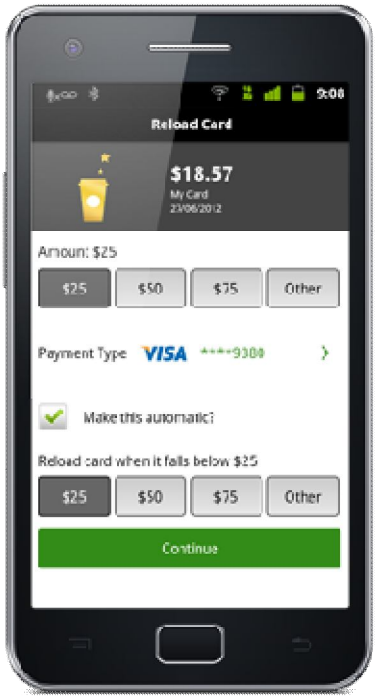
Με την χρήση της τεχνολογίας NFC ο εκάστοτε πελάτης χρησιμοποιεί ένα ειδικό κινητό τηλέφωνο που έχει ενσωματωμένη μια «έξυπνη» κάρτα (smartcard) που μπορεί να χρησιμοποιήσει κοντά σε έναν αναγνώστη τέτοιας συσκευής. Οι περισσότερες συναλλαγές δεν χρειάζονται αναγνώριση αλλά μερικές απαιτούν έλεγχο ταυτοποίησης με χρήση κωδικού PIN πριν από την ολοκλήρωση της συναλλαγής.

Η Ericsson και η Aconite είναι παραδείγματα εταιρειών που καθιστούν δυνατή για τις τράπεζες να δημιουργήσουν καταναλωτικές εφαρμογές κινητών πληρωμών που επωφελούνται από την τεχνολογία NFC. Οι NFC πωλητές στην Ιαπωνία συσχετίζονται με τα δίκτυα των μέσων μαζικής μεταφοράς , όπως το Mobile Suica που χρησιμοποιείται στο σιδηροδρομικό δίκτυο JR East. Στην Ευρώπη πολλοί πωλητές χρησιμοποιούν contactless τύπου πληρωμές μέσω κινητού τηλεφώνου σε περιοχές πάρκινγκ off-street. Έτσι επωφελούνται οι χρήστες με την ευκολία να μπορούν να πληρώσουν μέσα από το αυτοκίνητο τους με το κινητό τους τηλέφωνο. Αυτή η χρήση του συστήματος παρατηρείται σε σκανδιναβικές χώρες και στην Εσθονία.

2.9. Πληρωμές μέσω Barcodes

Οι πληρωμές μέσω barcodes γίνεται ένας όλο και πιο δημοφιλής τρόπος για να συνδεθούν οι χρήστες των κινητών τηλεφώνων με πληροφορίες που παρέχει ο έμπορος ή τα προϊόντα ενός καταστήματος.

Πιθανώς, ένα καλό παράδειγμα για να δείξουμε πόσο ισχυρές μπορούν να γίνουν οι πληρωμές κινητών τηλεφώνων μέσω barcodes είναι η εφαρμογή Starbucks Card Mobile που φέρεται να έχει επεξεργαστεί πάνω από 42 εκατομμύρια πληρωμές με τη χρήση barcode scanning έχοντας ουσιαστικά αντικαταστήσει με το κινητό τηλέφωνο την πληρωμή με το πορτοφόλι. Για την επιτυχία της εφαρμογής των Starbucks δεν είναι μόνο ότι επικεντρώθηκε στις πληρωμές μέσω κινητών τηλεφώνων αλλά ότι έχτισε εμπιστοσύνη με τους καταναλωτές του.



Εικόνα 1.11: Μια όψη της εφαρμογής Starbucks για Android

Επίσης, υπάρχει και η εφαρμογή LevelUp η οποία παίρνει την πιστωτική κάρτα που καταχωρεί ο χρήστης και δημιουργεί έναν μοναδικό QR Code που μπορεί να χρησιμοποιηθεί σε κάποιο σημείο πώλησης που έχει το ειδικό barcode reader της LevelUp, το οποίο διανέμει δωρεάν στους εμπόρους.

Το βασικό πλεονέκτημα των barcode πληρωμών είναι ότι προσφέρει άνεση στον καταναλωτή και στον έμπορο χωρίς το κόστος να γίνεται ένα πρόβλημα για την υπηρεσία αυτή.

Εικόνα 1.12: Η εφαρμογή LevelUp

2.10. Τεχνολογίες του mobile wallet

Ο απώτερος σκοπός του mobile wallet είναι να πραγματοποιήσει μια πληρωμή με τη χρήση κινητού τηλεφώνου. Ας δούμε λίγο τις τεχνολογίες που χρησιμοποιεί για να επιτύχει αυτό το σκοπό:

NFC ελεγκτή και κεραία: Ενεργοποιεί τη δυνατότητα στις κινητές συσκευές να στέλνουν τις πληροφορίες των λογαριασμών των χρηστών με ασφαλή τρόπο στο εκάστοτε contactless payment reader στο σημείο πώλησης.

Secure Element : Είναι μια ασφαλής έξυπνη κάρτα-τσιπ (smart card) στο εσωτερικό του τηλεφώνο που αποθηκεύει ή στέλνει τα στοιχεία πληρωμής του χρήστη. Είναι σε ξεχωριστό σημείο από τη μνήμη της συσκευής, δηλαδή μακρυά από φωτογραφίες, εφαρμογές ή επαφές. Για την πρόσβαση σε προσωπικές πληροφορίες υπάρχει προστασία από διάφορες στρώσεις ασφαλείας (security layers). Όταν χρειαστεί δίνει πρόσβαση στην εξουσιοδοτημένη εφαρμογή που θα το ζητήσει.

Trusted Service Manager (TSM) - Η TSM συνδέει τις κάρτες πληρωμής στο mobile wallet με ασφάλεια. Το TSM επιτρέπει στο χρήστη να εισάγει τον αριθμό λογαριασμού του στο mobile wallet που αργότερα πιστοποιεί με τη τράπεζα για να μπορέσει να χρησιμοποιηθεί από την εφαρμογή του mobile wallet του χρήστη.

2.11. Μια ματιά στα digital wallets της αγοράς

2.11.1. PayPal και PayPal App

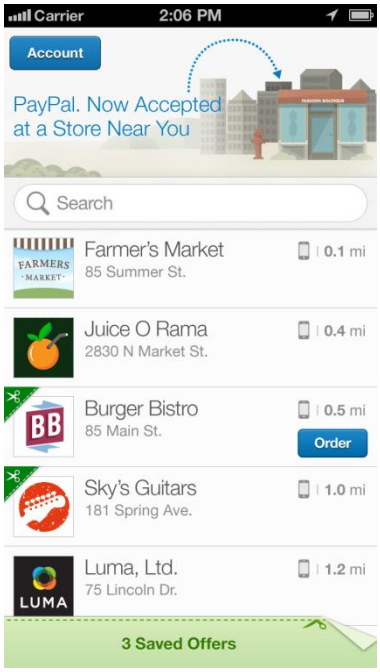
Το PayPal επιτρέπει σε κάθε επιχείρηση ή άτομο με μια διεύθυνση ηλεκτρονικού ταχυδρομείου την ασφαλή , εύκολη και αποτελεσματική (από άποψη κόστους) λήψη και αποστολή πληρωμών. Το δίκτυό της PayPal βασίζεται στην υπάρχουσα οικονομική υποδομή των τραπεζικών λογαριασμών και των πιστωτικών καρτών με σκοπό να δημιουργήσει μια παγκόσμια λύση πληρωμών σε πραγματικό χρόνο. Το PayPal χρησιμεύει ως μια ηλεκτρονική εναλλακτική λύση στις παραδοσιακές μεθόδους, όπως οι επιταγές και οι εντολές πληρωμών.

Ένας λογαριασμός PayPal (που συνδέεται πάντα με μια διεύθυνση e-mail) μπορεί να χρηματοδοτηθεί με ηλεκτρονική πίστωση από ένα τραπεζικό λογαριασμό ή από μια πιστωτική κάρτα. Το PayPal είναι ένα παράδειγμα μιας πληρωμής σε υπηρεσίες διαμεσολαβήσεως, που διευκολύνει τον κόσμο στις καθημερινές διαδικτυακές συναλλαγές του.

Ακόμη, εκτελεί την επεξεργασία των πληρωμών για online πωλήσεις, δημοπρασίες χώρων, καθώς και άλλους εμπορικούς χρήστες, για την οποία χρεώνει ένα τέλος πληρωμής δηλαδή ένα ποσοστό του ποσού που απέστειλε συν ένα πρόσθετο σταθερό ποσό για την κάθε συναλλαγή. Το επίπεδο των τελών εξαρτάται από το χρησιμοποιούμενο νόμισμα, την επιλογή πληρωμής που χρησιμοποιείται, τη χώρα του αποστολέα, τη χώρα του δικαιούχου, το ποσό που αποστέλλεται και τον τύπο του λογαριασμού του δικαιούχου.

Επιπλέον, το eBay σε αγορές που γίνονται με πιστωτική κάρτα μέσω PayPal μπορεί να αναλάβει μια αλλαγή του νομίσματος της συναλλαγής, αν ο πωλητής βρίσκεται σε άλλη χώρα. Έτσι ενημερώνονται αυτόματα και οι εκδότες πιστωτικών καρτών για τη χώρα προέλευσης του πωλητή.

Συνοψίζοντας, το PayPal χρησιμοποιώντας ένα δυνατό λογισμικό κρυπτογράφησης που επιτρέπει στους καταναλωτές να κάνουν τις μεταφορές πιστώσεων μεταξύ υπολογιστών και κινητών συσκευών. Αυτή η απλή ιδέα έχει μετατραπεί σε μία από τις πρωταρχικές μεθόδους στον κόσμο της online πληρωμής . Παρά την κατά καιρούς ταραγμένη ιστορία του , συμπεριλαμβανομένης της απάτης και αγωγής από μηχανισμούς κυβερνήσεων, το PayPal διαθέτει πλέον πάνω από 100 εκατομμύρια ενεργούς λογαριασμούς σε εκατοντάδες αγορές σε όλο τον κόσμο.



Εικόνα 1.13: Προβολή κοντινών καταστημάτων μέσα στο PayPal app

Η εφαρμογή του PayPal υποστηρίζεται φυσικά σε Android και iOS και είναι ιδιαίτερα δημοφιλής στους χρήστες των mobile wallets. Με την εφαρμογή αυτή μπορούμε να πληρώσουμε έχοντας έναν PayPal λογαριασμό, μια πιστωτική-χρεωστική κάρτα, ένα τραπεζικό λογαριασμό, ενώ μπορούμε να στείλουμε χρήματα σε κάποιον άλλο που έχει PayPal λογαριασμό.

Όπως και σε άλλες εφαρμογές για digital wallet το PayPal app ενσωματώνει προσφορές και ψηφιακά κουπόνια που δημιουργούνται αυτόματα με κάποια συναλλαγή-πληρωμή. Επίσης, μπορούμε μέσα από το μενού της εφαρμογής να βρούμε κοντινά καταστήματα που υποστηρίζουν την εν λόγω εφαρμογή και να δούμε τις προσφορές που μπορεί να έχουν αυτά τα καταστήματα.

Η ταυτοποίηση γίνεται με τεχνικές PIN κωδικού και εικόνας (όπως μια πολιτική ταυτότητα) κάνοντας έτσι ευκολότερη τη συναλλαγή και ταυτόχρονα διευρύνωντας το φάσμα των χρηστών που την χρησιμοποιούν. Εκτός από τις γνωστές υπηρεσίες προβολής υπολοίπου του λογαριασμού μας και άλλες υπηρεσίες PayPal αποδείξεων συναλλαγών, το PayPal app υποστηρίζει και την υπηρεσία Bill Me Later. Η υπηρεσία αυτή επιτρέπει στους καταναλωτές να αγοράσουν κάτι σε ένα κατάστημα και να το πληρώσουν αργότερα. Έτσι, οι καταναλωτές μπορούν να πληρώσουν το ποσό της συναλλαγής και σε δόσεις μέσα σε κάποιο χρονικό διάστημα, πριν χρεωθούν επιπλέον έξοδα.

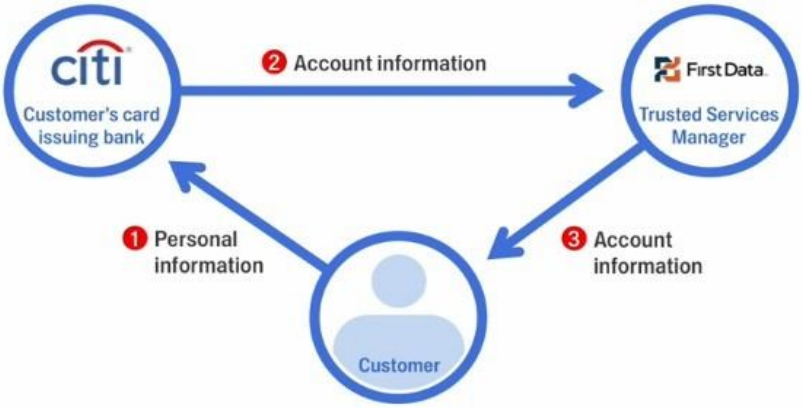
2.11.2. Google Wallet

Τον Σεπτέμβριο του 2011 βγήκε στην αγορά των Ηνωμένων Πολιτειών Αμερικής το Google Wallet. Η υπηρεσία επιτρέπει στους χρήστες της να αποθηκεύσουν πιστωτικές κάρτες, δωροεπιταγές, κάρτες πιστότητας και άλλες τεχνικές προώθησης πωλήσεων στο κινητό τους τηλέφωνο. Το Google Wallet σαν υπηρεσία έχει μια σημαντικότητα γιατί κατάφερε να ενώσει διάφορες συνιστώσες του εμπορίου σε μία, κρατώντας παράλληλα την «αξία» του καταναλωτή και του εμπόρου.

Εικόνα 1.14: Το λογότυπο του Google Wallet

Επίσης, το Google Wallet έγινε πράξη με την σύμπραξη και άλλων εταιρειών εκτός τη Google όπως την First data, την τράπεζα Citi και τη MasterCard.

- Citigroup: Η Citigroup είναι ο πρώτος εκδότης καρτών που ενσωματώνει τους λογαριασμούς των καρτών στο Google Wallet. Η Citi θα επικυρώνει της κάρτες όταν οι καταναλωτές μπάινουν στο Google Wallet και θα επικοινωνούν με την υπηρεσία της First Data (First Data's Trusted Service Manager) για να ενεργοποιήσει με ασφάλεια τους λογαριασμούς στο Google Wallet.
- First Data: Είναι ο Trusted Server Manager για το Google Wallet και η κύρια δραστηριότητα της First Data περιλαμβάνει την ασφαλή διανομή και εφοδιασμό αλλά και την διαχείρηση των ανέπαφων (contactless) πληρωμών και των εκδοτών πιστωτικών καρτών.
- Sprint: Είναι η πρώτη εταιρία κινητής τηλεφωνίας που γίνεται έταιρος με τη Google Wallet. Ο ρόλος της είναι να εμπορεύεται και να διανέμει συμβατές συσκευές με το Google Wallet.



Εικόνα 1.15: Πως δουλεύει το Google Wallet

2.11.3. Apple PassBook

Εικόνα 1.16: Εισαγωγική όψη στης εφαρμογής Passbook για το iOS 7

Το PassBook της Apple εισήχθη στο iOS 6 και στηρίζεται στη σάρωση 2D barcodes για να βοηθήσει στη διαχείρηση ταινιών, συναυλιών και αεροπορικών εισιτηρίων, καθώς και loyalty καρτών και κουπονιών για επιλεγμένους εμπόρους. Ακόμα, μπορούμε να προσθέσουμε «passes» (άδειες εισόδου) μέσα από εφαρμογές που υποστηρίζουν το PassBook.

Έτσι, αντί να φέρνουμε μαζί μας κουπόνια και εκπτωτικές κάρτες, τα αποθηκεύουμε στο PassBook. Αντίθετα με το Google Wallet, δεν μπορούμε να χρησιμοποιήσουμε το PassBook για συναλλαγές μέσα σε ένα κατάστημα (αν και η Apple επισημαίνει ότι σύντομα θα υποστηρίζεται και αυτή η λειτουργία) , αλλά υπάρχει το «BillGuard» που μπορούμε να δούμε πληροφορίες για τραπεζικούς λογαριασμούς μέσα από το δικό μας iPhone.

2.11.4. Lemon Wallet

Εικόνα 1.17: Επιλογές μέσα από την εφαρμογή Lemon Wallet

Το Lemon Wallet είναι μια εφαρμογή που μας επιτρέπει να αποθηκεύσουμε και να χρησιμοποιήσουμε χρεωστικές-πιστωτικές κάρτες αλλά και ασφαλιστικές κάρτες. Το δυνατό της σημείο είναι ότι μετατρέπει όλες αυτές τις πληροφορίες σε ένα barcode που μπορεί να σαρωθεί από τους εμπόρους. Μπορούμε επίσης με αυτήν την εφαρμογή να συνδέσουμε κάρτες πληρωμών στην εφαρμογή έχοντας πρόσβαση στις συναλλαγές μας.

2.11.5. MasterCard MasterPass

Εικόνα 1.18: Πληρωμές από διαφορετικά σημεία χρησιμοποιώντας την MasterCard MasterPass

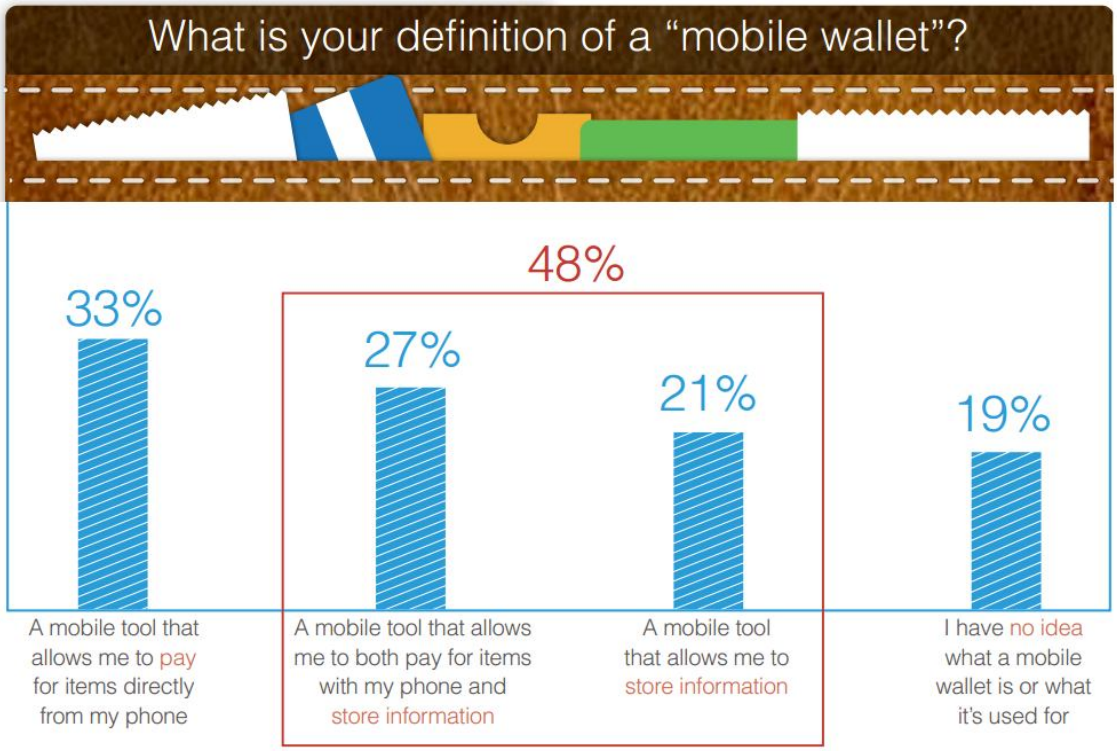
Την άνοιξη του 2012 η MasterCard έφερε στην αγορά την εφαρμογή της για digital wallet το PayPass επιτρέποντας στους χρήστες να δημιουργήσουν δωρεάν λογαριασμούς και να πληρώνουν για τις αγορές σε διάφορους διαδικτυακούς τόπους (εταίρους με τη MasterCard) με τις πιστωτικές κάρτες, και άλλες πληροφορίες που είχαν αποθηκεύσει στο PayPass κινητό πορτοφόλι τους.

Αργότερα, μέσα στο 2013 η MasterCard ανακοίνωσε μια επανασχεδίαση του PayPass μετονομάζοντας το σε MasterPass που πρώτα το έβγαλε στην αγορά στον Καναδά και την Αυστραλία και αργότερα στις Ηνωμένες Πολιτείες και σε χώρες της Ευρώπης όπως την Γαλλία και την Ιταλία. Ας δούμε μερικά από τα χαρακτηριστικά του:

- Για συναλλαγές μέσα σε ένα κατάστημα (in-store) υποστηρίζει τη χρήση NFC, QR Codes, κινητές συσκευές που θα χρησιμοποιούνται στο εκάστοτε σημείο πώλησης.
- Οι καταναλωτές μπορούν να αποθηκεύσουν πληροφορίες των πιστωτικών τους καρτών με ασφάλεια μέσα σε cloud servers. Επίσης υποστηρίζονται και άλλες επώνυμες πιστωτικές, χρεωστικές ή προπληρωμένες κάρτες.
- Μετά το τέλος της εκάστοτε συναλλαγής-παραγγελίας οι χρήστες θα έχουν πρόσβαση σε περαιτέρω πληροφορίες και υπηρεσίες όπως υπόλοιπο λογαριασμού, ειδοποιήσεις σε πραγματικό χρόνο και διάφορες προσφορές.

2.12. Πως αντιμετωπίζει η αγορά τα mobile wallets

Συνεχίζεται να υπάρχει σύγχυση στην αγορά σε σχέση με το τί μπορεί να προσφέρει ένα mobile wallet. Ενώ με μια πρώτη σκέψη θα λέγαμε ότι το mobile wallet μπορεί να χρησιμοποιηθεί ως μια εναλλακτική συσκευή πληρωμών, η έρευνα του Vibes δείχνει ότι το 48% καταλαβαίνει και τις δυνατότητες που έχει το mobile wallet και δεν αφορούν πληρωμές μέσω κινητού τηλεφώνου. Το ένα τρίτο (33%) νομίζει ότι το mobile wallet χρησιμοποιείται μόνο για πληρωμές και ένα ποσοστό 19% αναφέρει ότι δεν γνωρίζει τί είναι το mobile wallet **[Εικόνα 1.19]**. (Βιβλιογραφία [7])

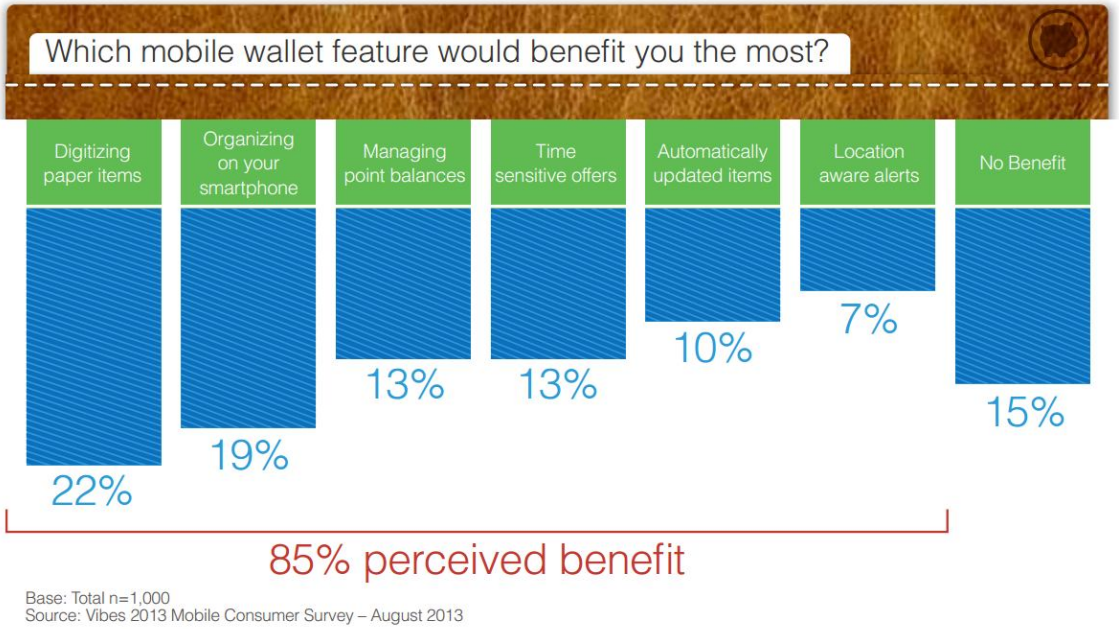


Εικόνα 1.19: Η έρευνα του Vibes για τον ορισμό του mobile wallet

2.13. Όταν το mobile wallet δεν χρησιμοποιείται για πληρωμές

Η ζήτηση για τη λειτουργία του mobile wallet όταν αυτό δεν χρησιμοποιείται για πληρωμές είναι υψηλή αφού το 85% των καταναλωτών δηλώνουν ότι θα έχουν κάποιο ώφελος από την αποθήκευση καρτών και κουπονιών στα κινητά τους τηλέφωνα. Η ψηφιοποίηση αντικειμένων όπως είναι ένα χάρτινο κουπόνι προσφορών δείχνει να ελκύει τους καταναλωτές **[εικόνα 1.20]**. (Βιβλιογραφία [7])

Η ένωση του χρόνου, τόπου και αλληλεπίδρασης είναι στοιχεία που διαφοροποιούν το mobile wallet αφού το σωστό περιεχόμενο την κατάλληλη στιγμή στην ιδανική τοποθεσία μπορούν να φέρουν πίσω τους καταναλωτές σε ένα κατάστημα με διάφορα τεχνάσματα προώθησης προϊόντων.

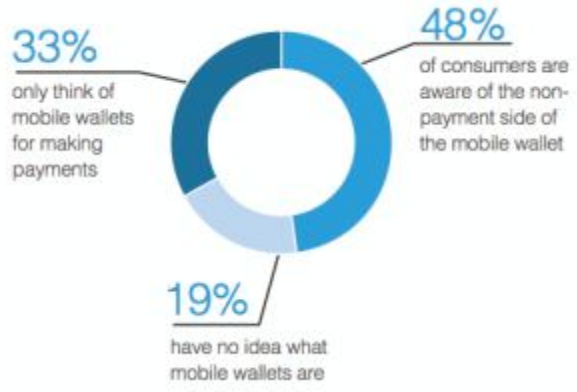


Εικόνα 1.20: Έρευνα του Vibes (Αύγουστος 2013) για τη χρήση λειτουργιών του mobile wallet

Ενδιαφέρον έχει και η «εμπειρία» που προσφέρει το mobile wallet στους καταναλωτές. Ένα ποσοστό 39% χαρακτήρισε την «εμπειρία» του mobile wallet άκρως θετική. Σαν τεχνολογία είναι πολύ νέα προς τον μέσο καταναλωτή και η εμπειρία που μπορεί να έχει κάποιος εξαργυρώνοντας ένα χάρτινο κουπόνι προσφορών μπορεί ακόμα να φαίνεται μπερδεμένη. Αυτό οφείλεται και στο γεγονός ότι τα καταστήματα δεν είναι ακόμα πλήρως εφοδιασμένα και προετοιμασμένα για την τεχνολογία του mobile wallet.

2.14. Το μέλλον για τα mobile wallets

Μέχρι το 2020, τα κινητά πορτοφόλια θα έχουν γίνει η προτιμώμενη μέθοδος των in-store ψηφιακών πληρωμών, σύμφωνα με μια μελέτη που εκπονήθηκε από το Internet & American Life Project και το Πανεπιστήμιο Elon. Οι υποστηρικτές της τεχνολογίας smartphone τονίζουν ότι αυτός ο τρόπος πληρωμής θα απλοποιήσει πραγματικά τις συναλλαγές.

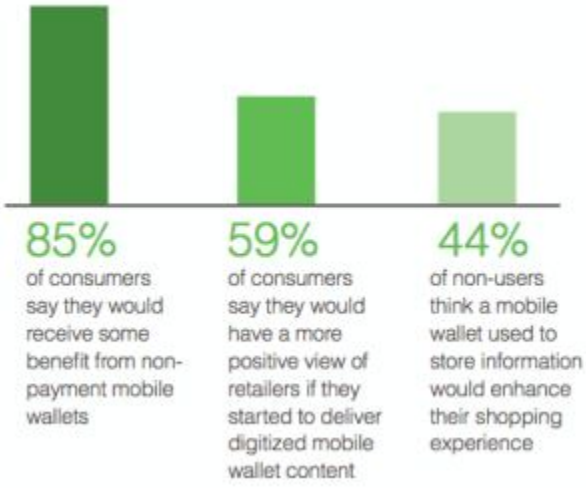


Εικόνα 1.21: Έρευνα για το mobile wallet (Βιβλιογραφία [8])

Οι πληρωμές μέσω κινητού τηλεφώνου έχουν τη δύναμη να αυξήσουν τον αριθμό των ηλεκτρονικών συναλλαγών και να παρέχουν αυξημένες δυνατότητες διαχείρησης συναλλαγών για τους χρήστες. Αυτό θα έχει πολύ θετικά αποτελέσματα σε όλη την αλυσίδα των συναλλαγών και πιο συγκεκριμένα στην μείωση των παράνομων ή/και μη ανισχεύσιμων συναλλαγών, μείωση των μετρητών χρημάτων και μείωση σε κόστη διοίκησης, αυξάνοντας την ικανοποίηση των πελατών.

Ανεξάρτητα από το πώς γίνεται η συναλλαγή (QR Codes ή NFC τεχνολογίες), νέες προοπτικές που δεν χρειάζονται τον χρήστη να ελέγχει την πιστωτική του κάρτα θα δημιουργήσουν νέα καταναλωτικά μοντέλα για τις πληρωμές μέσω κινητών τηλεφώνων.

Υπάρχουν ισχυρές ενδείξεις ότι τα mobile wallets θα αναγνωριστούν σταδιακά ανάμεσα στους καταναλωτές, κάτι που θα επιταχύνει την διείσδυση νέων εταιριών στον χώρο και θα βοηθήσει στην σύμπραξη με τις τράπεζες. Ακόμα, η τεχνολογία αυτή θα βοηθήσει στο να μειωθεί το φράγμα μεταξύ εμπόρων-καταστημάτων και καταναλωτών αφού μέσα από το κάθε κινητό πορτοφόλι κάθε σημαντική πληροφορία θα είναι άμεσα διαθέσιμη στην οθόνη του χρήστη.



Εικόνα 1.22: Έρευνα του Vibes για χρήση προσφορών στο ηλεκτρονικό πορτοφόλι

Ας αναφέρουμε μερικές από τις αλλαγές που περιμένουμε από τον χώρο των συναλλαγών μέσω κινητών τηλεφώνων και ηλεκτρονικών πορτοφολιών:

- **Νέες δυνατότητες για το Google Wallet:** Το Google Wallet επιτρέπει στους χρήστες του να αποθηκεύουν χρεωστικές και πιστωτικές κάρτες καθώς και κάρτες δώρων μεταξύ άλλων κάνει πιο εύκολη και γρήγορη μια πληρωμή μέσω κινητών τηλεφώνων με τεχνολογίες NFC. Πρόσφατα όμως η Google ανακοίνωσε μια ενημερωμένη εφαρμογή για το Google Wallet που θα επιτρέπει στους χρήστες να αποθηκεύουν περιεχόμενα όπως κάρτες πιστότητας και προσφορές στο Google Wallet. Με αυτές τις αναβαθμίσεις η Google δίνει τη δυνατότητα σε περισσότερους εμπόρους να επωφεληθούν από το Google Wallet δίνοντας έτσι πρόσβαση σε πιο ευρή κοινό καταναλωτών και εμπόρων.
- **Passbook Scanner:** Το πιο ενδιαφέρον χαρακτηριστικό της Apple για το Passbook προστέθηκε στην iOS 7 έκδοση του λειτουργικού. Με το Passbook Scanner δίνεται ένα χαρακτηριστικό σάρωσης που επιτρέπει στους καταναλωτές να σαρώσουν QR Codes και να λαμβάνουν πληροφορίες μέσα στο ηλεκτρονικό τους πορτοφόλι.

- **Εξατομικευμένη χρήση:** Η κινητή τεχνολογία παρέχει εξατομικευμένη ενημέρωση που είναι μοναδική για τον καταναλωτή που τη χρησιμοποιεί. Έτσι, γίνεται σημαντικότερο και για τους εμπόρους να δημιουργήσουν εξατομικευμένο περιεχόμενο βασισμένο στις ανάγκες του κάθε πελάτη. Με τη χρήση του κινητού πορτοφολιού θα είναι εφικτό να ειδοποιούνται οι καταναλωτές για ειδικές προσφορές που έχουν αποθηκευμένες στο κινητό πορτοφόλι τους την χρονική στιγμή που γίνονται διαθέσιμες, παρέχοντας έτσι πιο δυναμικές ευκαιρίες στους εμπόρους-εταιρείες για να κερδίσουν την εμπιστοσύνη των πελατών τους με νέες τεχνικές και στρατηγικές που θα εφαρμόσουν με τη βοήθεια του μοντέλου του ηλεκτρονικού πορτοφολιού.

3. Εισαγωγή στο λειτουργικό σύστημα Android και στον προγραμματισμό Android Εφαρμογών

3.1.Εισαγωγή στο Android

Εικόνα 1.23: Το λογότυπο του Android

Το **Android** είναι λειτουργικό σύστημα για συσκευές κινητής τηλεφωνίας το οποίο τρέχει τον πυρήνα του λειτουργικού Linux. Αρχικά αναπτύχθηκε από την Google και αργότερα από την Open Handset Alliance. Επιτρέπει στους κατασκευαστές λογισμικού να συνθέτουν κώδικα με την χρήση της γλώσσας προγραμματισμού Java, ελέγχοντας την συσκευή μέσω βιβλιοθηκών λογισμικού ανεπτυγμένων αρχικά από την Google.

Η πρώτη παρουσίαση της πλατφόρμας Android έγινε στις 5 Νοεμβρίου 2007, παράλληλα με την ανακοίνωση της ίδρυσης του οργανισμού Open Handset Alliance, μιας κοινοπραξίας 48 τηλεπικοινωνιακών εταιριών, εταιριών λογισμικού καθώς και κατασκευής hardware, οι οποίες είναι αφιερωμένες στην ανάπτυξη και εξέλιξη ανοιχτών προτύπων στις συσκευές κινητής τηλεφωνίας. Η Google δημοσίευσε το μεγαλύτερο μέρος του κώδικα του Android υπό τους όρους της Apache License, μιας ελεύθερης άδειας λογισμικού. Το λογότυπο για το λειτουργικό σύστημα Android είναι ένα ρομπότ σε χρώμα πράσινο μήλο και σχεδιάστηκε από τη γραφίστρια Irina Blok.

3.2.Χαρακτηριστικά

Στις επόμενες ενότητες θα αναλύσουμε διάφορα χαρακτηριστικά του Android που το έχουν κάνει να ξεχωρίσει με τις πρωτοποριακές τεχνολογίες που διαθέτει.

Εικόνα 1.24: Ολογράφως το λογότυπο του Android λειτουργικού

3.2.1. Λειτουργίες οθόνης

Η πλατφόρμα είναι προσαρμόσιμη σε μεγαλύτερη ανάλυση (VGA), δισδιάστατες ψηφιακές γραφικές βιβλιοθήκες, τρισδιάστατα γραφικά βασισμένα στην OpenGL ES 1.0 έκδοση χαρακτηριστικών, καθώς και παραδοσιακές απεικονίσεις οθόνης "έξυπνων" συσκευών κινητής τηλεφωνίας.

3.2.2. Αποθήκευση δεδομένων

Για τις ανάγκες αποθήκευσης δεδομένων το Android χρησιμοποιεί τη βάση δεδομένων SQLite, μια σχεσιακή (και ελαφριά στη χρήση) βάση δεδομένων. Εκτός αν χρησιμοποιήσουμε άλλες μεθόδους για τη χρήση βάσεων δεδομένων μορφής MySQL όπως θα δούμε παρακάτω.

3.2.3. Συνδεσιμότητα

Το Android υποστηρίζει τεχνολογίες συνδεσιμότητας συμπεριλαμβανομένου GSM/EDGE, CDMA, EV-DO, UMTS, Bluetooth, και Wi-Fi.

3.2.4. Αποστολή μηνυμάτων

Τα μηνύματα που μπορούν να σταλούν είναι SMS για αποστολή μηνυμάτων κειμένου, MMS για αποστολή μηνυμάτων πολυμεσικού περιεχομένου και τώρα το Android Cloud To Device Messaging (C2DM) είναι μέρος του Android Push Messaging service.

3.2.5. Περιήγηση στον ιστό

Για την περιήγηση στον ιστό το Android διαθέτει φυλλομετρητή (browser) βασισμένο στην ανοιχτή τεχνολογία Webkit, σε συνδιασμό με τη V8 μηχανή του Javascript του Chrome. Ο browser επιτυγχάνει 100% score στο τεστ επιδόσεων Acid3 στο Android 4.0.

Το WebView είναι μια κλάση που μπορεί να εμφανίσει online περιεχόμενα μέσα σε μια οθόνη μιας Android εφαρμογής. Περιέχει τεχνικές για προβολή ιστορικού σελίδων, zoom in και zoom out, αναζήτηση κειμένου και άλλα.

3.2.6. Υποστήριξη Java

Λογισμικό γραμμένο στην Java είναι δυνατόν να μεταγλωττιστεί και να εκτελεστεί στην εικονική μηχανή Dalvik, η οποία αποτελεί εξειδικευμένη υλοποίηση εικονικής μηχανής, σχεδιασμένης για χρήση σε φορητές συσκευές, παρόλο που δεν είναι πρότυπη εικονική μηχανή Java.

3.2.7. Υποστήριξη πολυμέσων

Το λειτουργικό Android υποστηρίζει τις ακόλουθα μορφές ήχου, στατικής και κινούμενης εικόνας: H.263, H.264 (σε 3GP ή MP4 container), MPEG- SP, AMR, AMR-WB, AAC, HE-AAC, MP3, MIDI, OG Vorbis, WAV,JPEG, PNG, GIF, BMP.

3.2.8. Επιπλέον υποστήριξη υλικού

Το λειτουργικό Android μπορεί να συνεργαστεί με κάμερες στατικής ή κινούμενης εικόνας, οθόνες αφής, GPS, αισθητήρες επιτάχυνσης, μαγνητόμετρα, δισδιάστατους καθώς και τρισδιάστατους επιταχυντές γραφικών.

3.2.9. Περιβάλλον ανάπτυξης λογισμικού

Το περιβάλλον ανάπτυξης λογισμικού περιλαμβάνει ένα προσομοιωτή συσκευής, εργαλεία για διόρθωση σφαλμάτων, μνήμη και εργαλεία ανάλυσης της απόδοσης του εκτελέσιμου λογισμικού καθώς και ένα επιπρόσθετο για το Eclipse IDE.

3.2.10. Αγορά και εγκατάσταση εφαρμογών

Η αγορά του Android είναι παρόμοια με το App Store του iPhone OS, το Android Market είναι ένας κατάλογος εφαρμογών που μπορούν να μεταφορτωθούν και εγκατασταθούν στην συσκευή άμεσα μέσω ασύρματων καναλιών, χωρίς την χρήση υπολογιστή. Αρχικά μόνο δωρεάν εφαρμογές ήταν δυνατόν να εγκατασταθούν. Εφαρμογές επί πληρωμή ήταν μετέπειτα διαθέσιμες στο Android Market στις ΗΠΑ ύστερα από τις 19 Φεβρουαρίου 2009.

3.2.11. Οθόνη αφής πολλαπλών σημείων

Το λειτουργικό Android είχε εξ ορισμού υποστήριξη για οθόνες πολλαπλών σημείων αλλά η δυνατότητα αυτή έχει κλειδωθεί σε επίπεδο πυρήνα (πιθανόν για αποφυγή παραβιάσεων των πατεντών λογισμικού της Apple στις τεχνολογίες οθονών αφής). Κυκλοφορεί μια ανεπίσημη τροποποίηση (mod) που έχει αναπτυχθεί για να υποστηρίζει πολλαπλή επαφή (multi-touch), αλλά απαιτεί δικαιώματα πρόσβασης υπερχρήστη (superuser) στη συσκευή για να γραφεί στη μνήμη flash ένας πυρήνας που να μην είναι υπογεγραμμένος (unsigned kernel).

3.2.12. Ιστορικό ενημερώσεων

Παρόλο που το Android είναι ένα προϊόν ελεύθερου λογισμικού, ένα κομμάτι της ανάπτυξης του λογισμικού συνεχίζεται σε ιδιωτικό παρακλάδι. Για να έρθει αυτό το λογισμικό σε κοινή θέαση δημιουργήθηκε ένα παρακλάδι του μόνο ανάγνωσης, εν ονόματι "Cupcake". Το Cupcake συνήθως συγχέεται με τον τίτλο μιας ενημέρωσης, σε αντίθεση με όσα δηλώνει η ίδια η Google στην ιστοσελίδα ανάπτυξης του Android: "το Cupcake αποτελεί ακόμη ένα έργο σε εξέλιξη, όχι μια επίσημη έκδοση." Αξιοσημείωτες αλλαγές στο λειτουργικό Android θα παρουσιαστούν στο cupcake και περιλαμβάνουν αλλαγές στο σύστημα διαχείρισης των μεταφορτώσεων (download manager), το framework, Bluetooth, το λογισμικό συστήματος, το ραδιόφωνο και το σύστημα τηλεφωνίας, εργαλεία προγραμματισμού, το κυρίως σύστημα και διάφορες εφαρμογές, καθώς και πληθώρες διορθώσεις σφαλμάτων.

Στις 30 Απριλίου 2009, κυκλοφόρησε η επίσημη ενημέρωση έκδοσης 1.5 για το Android. Αποτελείται από πολλά νέα χαρακτηριστικά και βελτιώσεις στο γραφικό περιβάλλον:

- Δυνατότητα καταγραφής κινούμενης εικόνας με την χρήση της αντίστοιχης λειτουργίας του τηλεφώνου.
- Μεταφόρτωση αρχείων βίντεο στο YouTube και εικόνων στο Picasa κατευθείαν από το τηλέφωνο.
- Επανασχεδιασμένο λογισμικό πληκτρολογίου με λειτουργία αυτόματης συμπλήρωσης κειμένου.
- Δυνατότητα αυτόματης σύνδεσης ασύρματης συσκευής ακουστικού Bluetooth εφόσον εντοπιστεί σε μια συγκεκριμένη απόσταση.
- Νέα widgets και φάκελοι που μπορούν να τοποθετηθούν στην επιφάνεια εργασίας.
- Εφέ αλλαγής οθονών και μενού.
- Διευρυμένη λειτουργία αντιγραφής/επικόλλησης για να περιλαμβάνει δικτυακές διευθύνσεις..

Μετά από αυτό έρχεται η έκδοση 1.6 επονομαζόμενη Donut και στη συνέχεια παρουσιάζεται η σειρά 2 με πρώτη έκδοση την Eclair μετά την 2.2 Froyo και περνάμε στις πρόσφατες εκδόσεις.

3.3.Gingerbread

Εικόνα 1.25: Το σήμα της Gingerbread

Η 2.3 έκδοση (Gingerbread) βελτίωσε τη διεπαφή χρήστη, βελτίωσε το πληκτρολόγιο αφής και τις δυνατότητες αντιγραφής-επικόλλησης, βελτίωσε τις επιδόσεις στα παιχνίδια ,πρόσθεσε SIP υποστήριξη (κλήσεις VOIP) και πρόσθεσε υποστήριξη για την επικοινωνία κοντινού πεδίου (Near Field Communication).

3.4.Honeycomb

Εικόνα 1.26: Το σήμα της Honeycomb

Η 3.0 έκδοση (Honeycomb) δημιουργήθηκε κυρίως για χρήση σε tablet. Η Honeycomb υποστηρίζει συσκευές με μεγαλύτερες οθόνες και μεγαλύτερες αναλύσεις και εισάγει πολλά νέα χαρακτηριστικά διεπαφής χρήστη, υποστηρίζει επεξεργαστές πολλών πυρήνων (multi-core processors), υποστηρίζει επιτάχυνση υλικού γραφικών για βελτιωμένες επιδόσεις του λειτουργικού συστήματος και των εφέ που χρησιμοποιεί και τέλος έχει υποστήριξη πλήρους συστήματος κρυπτογράφησης. Η πρώτη συσκευή που χρησιμοποίησε αυτήν την έκδοση ήταν το Motorola Xoom tablet και διατέθηκε προς πώληση τον Φεβρουάριο του 2011.

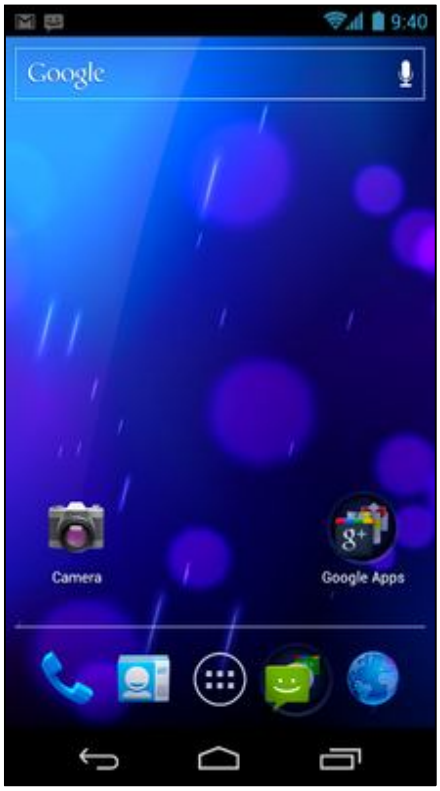
1. Η έκδοση 3.1 κυκλοφόρησε το Μάιο του 2011 και πρόσθεσε υποστήριξη για περισσότερες συσκευές εισόδου, λειτουργία USB για τη μεταφορά δεδομένων απευθείας από φωτογραφικές μηχανές και άλλες συσκευές, καθώς και υποστήριξη για Google Movies και Books Apps.
2. Η έκδοση 3.2 που κυκλοφόρησε τον Ιούλιο του 2011 πρόσθεσε βελτιστοποίηση για ένα ευρύτερο φάσμα μεγεθών οθόνης, νέο "zoom-to-fill" χαρακτηριστικό έτσι ώστε να μεγεθύνεται η οθόνη σε Full screen χωρίς να υπάρχουν σφάλματα στην εικόνα ή προβλήματα συμβατότητας, πρόσθεσε υποστήριξη για φόρτωση αρχείων πολυμέσων απευθείας από την κάρτα SD και ένα εκτεταμένο API υποστήριξης οθόνης.Το HuaweiMediaPad ήταν το πρώτο tablet 7 ιντσών που χρησιμοποίησε αυτήν την έκδοση.

3.5. Ice Cream Sandwich

Εικόνα 1.27: Το σήμα του Ice Cream Sandwich

Η 4.0 έκδοση (Ice Cream Sandwich) ανακοινώθηκε στις 19 Οκτωβρίου 2011, έφερε τα χαρακτηριστικά της Honeycomb στα smartphones και έχει προσθέσει νέα χαρακτηριστικά όπως αναγνώριση προσώπων μέσω της κάμερας, παρακολούθηση και έλεγχο των δεδομένων

και της χρήσης του δικτύου, ενιαίες κοινωνικές επαφές δικτύωσης, βελτιώσεις των φωτογραφιών, αναζήτηση e-mail ακόμα και σε offline κατάσταση, φακέλους για τα προγράμματα και διαμοιρασμό πληροφοριών με χρήση της τεχνολογίας NFC (NearFieldCommunication). Η έκδοση 4.0.3 Ice Cream Sandwich είναι η πιο πρόσφατη έκδοση του Android που είναι διαθέσιμη σε κινητά τηλέφωνα. Ο πηγαίος κώδικας του Android 4.0.1 κυκλοφόρησε στις 14 Νοεμβρίου 2011.



Εικόνα 1.28: Η επιφάνεια εργασίας της έκδοσης Ice Cream Sandwich στο Samsung Galaxy Nexus

3.6. Πίνακας εκδόσεων

Έκδοση	Κωδική ονομασία	Ημερομηνία	API level	Διανομή (4 Μαρτίου 2013)
4.2.x	*Jelly Bean*	13 Νοεμβρίου 2012	17	1.6%
4.1.x	*Jelly Bean*	9 Ιουλίου 2012	16	14.9%
4.0.x	*Ice Cream Sandwich*	16 Δεκεμβρίου 2011	15	28.6%
3.2	*Honeycomb*	15 Ιουλίου 2011	13	0.9%
3.1	*Honeycomb*	10 Μαΐου 2011	12	0.3%
2.3.3-2.3.7	*Gingerbread*	9 Φεβρουαρίου 2011	10	44%
2.3-2.3.2	*Gingerbread*	6 Δεκεμβρίου 2010	9	0.2%
2.2	*Froyo*	20 Μαΐου 2010	8	7.6%
2.0-2.1	*Eclair*	26 Οκτωβρίου 2009	7	1.9%
1.6	*Donut*	15 Σεπτεμβρίου 2009	4	0.2%

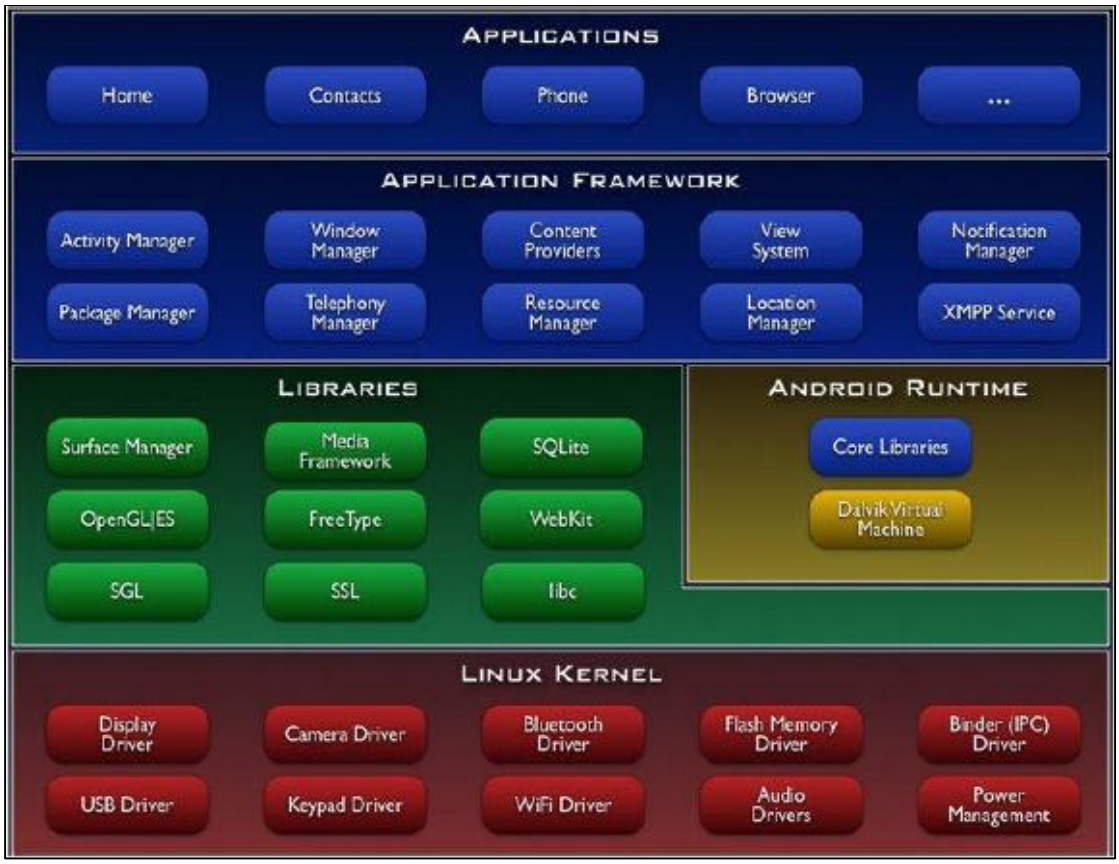
Εικόνα 1.29: Οι εκδόσεις και τα στατιστικά των προηγούμενων εκδόσεων του Android

3.7. Σχεδίαση

Το Android αποτελείται από ένα πυρήνα που βασίζεται στον πυρήνα Linux με middleware, βιβλιοθήκες και APIs γραμμένα σε γλώσσα C και λογισμικό εφαρμογών που λειτουργούν σε ένα πλαίσιο εφαρμογών που περιλαμβάνει συμβατές βιβλιοθήκες Java βασισμένες στο projectApacheHarmony. Το Android χρησιμοποιεί την εικονική μηχανή Dalvik με δυνατότητα "just-in-timecompilation" για να μπορεί να τρέχει DalvikDEX-κώδικα (Εκτελέσιμο από Dalvik), που συνήθως μεταφράζεται από Javabyte κώδικα.

Η βασική πλατφόρμα hardware για το Anrdoid είναι η αρχιτεκτονική ARM. Υπάρχει υποστήριξη για x86 (32 bit) από το projectAndroidx86, και η GoogleTV χρησιμοποιεί μια ειδική x86 έκδοση του Android.

3.8.Πυρήνας Linux του Android



Εικόνα 1.30: Το διάγραμμα αρχιτεκτονικής του Android

Ο πυρήνας του Android βασίζεται στον πυρήνα του Linux και φέρει περαιτέρω αλλαγές αρχιτεκτονικής από την Google , διαφορετικές σε σχέση με τον τυπικό κύκλο ανάπτυξης του πυρήνα του Linux . Το Android δεν έχει ένα ενσωματωμένο σύστημα X Window (X Window System), ούτε υποστηρίζει το σύνολο των GNU βιβλιοθηκών, και αυτό κάνει δύσκολο το να μεταφέρουν υπάρχουσες εφαρμογές ή βιβλιοθήκες του Linux για το Android.

Ορισμένα χαρακτηριστικά που η Google συνείσφερε στον πυρήνα του Linux και κυρίως η λειτουργία διαχείρισης ισχύος που ονομάζεται wake locks, απορρίφθηκαν από τους επικεφαλής προγραμματιστές του πυρήνα, εν μέρει επειδή οι συντηρητές του πυρήνα θεώρησαν ότι η Google δεν έδειξε καμία πρόθεση να διατηρήσει το δικό τους κώδικα. Παρόλο που η Google ανακοίνωσε τον Απρίλιο του 2010 ότι θα προσλάβει δύο υπαλλήλους για να συνεργαστεί με την κοινότητα του Linux kernel, ο Greg Kroah-Hartman, τρέχων συντηρητής του πυρήνα Linux για την stable έκδοση, είπε τον Δεκέμβριο του 2010, ότι ήταν ανήσυχος πως η Google δεν προσπαθεί πλέον να εντάξει τις αλλαγές του κώδικα της στο Linux. Μερικοί προγραμματιστές του Android της Google άφησαν να εννοηθεί ότι ¨η ομάδα του Android έχει αρχίσει να βαριέται τη διαδικασία¨, επειδή ήταν μια μικρή ομάδα καιείχαν πιο επείγουσες εργασίες να κάνουν με το Android.

Ωστόσο, το Σεπτέμβριο του 2010, ο προγραμματιστής πυρήνα του Linux,RafaelJ. Wysocki, πρόσθεσε ένα patch που βελτίωσε το κύριο πλαίσιο εκδηλώσεων αφύπνισης Linux. Είπε ότι οι οδηγοί συσκευών Android που χρησιμοποιούν wakelocks μπορούν τώρα εύκολα να συγχωνευθούν στο κύριο Linux, άλλα ότι οι ευκαιριακές ανασταλτικές λειτουργίες του Android δεν θα έπρεπε να συμπεριληφθούν στον κυρίως πυρήνα. Το2011, ο Linus Torvalds, δήλωσε ότι ¨τελικά το Android και το Linux θα επανέλθουν σε ένα κοινό πυρήνα, αλλά δεν θα είναι πιθανότατα για τέσσερα έως πέντε χρόνια¨.

Τον Δεκέμβριο του 2011, ο Greg Kroah-Hartman ανακοίνωσε την έναρξη του έργου Main lining του Android , το οποίο έχει ως στόχο να θέσει μερικούς Android οδηγούς, μπαλώματα (patches) και χαρακτηριστικά πίσω στον πυρήνα του Linux , αρχής γενομένης από το Linux 3.3. Περαιτέρω ολοκλήρωση αναμένεται στο Linux Kernel 3.4.

3.9.Εφαρμογές

Οι εφαρμογές αναπτύσσονται συνήθως σε γλώσσα προγραμματισμού JAVA χρησιμοποιώντας το λογισμικό Android Development Kit, όμως υπάρχουν και άλλα εργαλεία που είναι διαθέσιμα για τη δημιουργία και την ανάπτυξη των εφαρμογών όπως το ενσωματωμένο Development Kit για εφαρμογές ή επεκτάσεις τους σε γλώσσα C ή C++.

Επίσης το Google App Inventor, ένα γραφικό περιβάλλον που παρέχει τη δυνατότητα για δημιουργία προγράμματος για το Android ακόμα και σε αρχάριους προγραμματιστές.

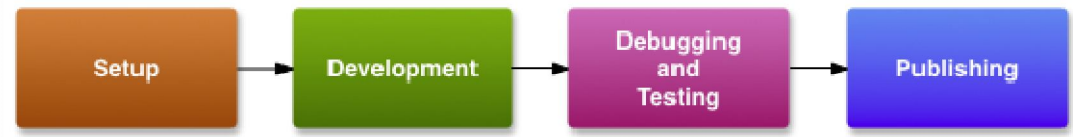
3.10. Google Play

Εικόνα 1.31: Το λογότυπο του Google Play

Το Android Market είναι το ηλεκτρονικό κατάστημα που δημιουργήθηκε από την Google για την εξυπηρέτηση των συσκευών Android. Υπάρχει ένα πρόγραμμα που είναι διαθέσιμο και εγκατεστημένο σε όλες τις συσκευές Android, το Market ή Play Store πλέον και επιτρέπει στους χρήστες να κατεβάσουν εφαρμογές που έχουν δημιουργηθεί από διάφορους προγραμματιστές. Τον Οκτώβριο του 2011 ήταν περισσότερες από 300.000 εφαρμογές διαθέσιμες και ο αριθμός εγκατεστημένων ξεπέρασε τα 10 δισεκατομμύρια, ενώ το λειτουργικό σύστημα ήταν εγκατεστημένο σε 130 εκατομμύρια συσκευές. Μόνο οι εφαρμογές που συμμορφώνονται με τις απαιτήσεις συμβατότητας της Google μπορούν να εισέλθουν στο Play Store. Το Play Store περιέχει φίλτρα που παρουσιάζουν μόνο τις εφαρμογές που είναι συμβατές με τη συσκευή του χρήστη έτσι ώστε να αποφευχθούν προβλήματα συμβατότητας. Επίσης οι προγραμματιστές των εφαρμογών έχουν την δυνατότητα να περιορίζουν τις εφαρμογές που έχουν δημιουργήσει σε συγκεκριμένες χώρες για επαγγελματικούς λόγους.

Η Google έχει συμμετάσχει στο κατάστημα προσφέροντας διάφορες εφαρμογές όπως την υπηρεσία Google Voice, Sky Map για την παρακολούθηση των άστρων, Google Translate, το Finance για την παρακολούθηση των οικονομικών, το Places Directory για την τοπική αναζήτηση , το Google Goggles που ψάχνει με βάση μια εικόνα, το Gestures Search για την αναζήτηση μέσω γραμμάτων και αριθμών που έχουν ¨ζωγραφιστεί¨ από το χρήστη των περιεχομένων του τηλεφώνου, το Google Shopper, το Google Maps και πολλά άλλα. Η Google τον Αύγουστο του 2010 δημιούργησε το Actions Voice for Android όπου η εφαρμογή αυτή επιτρέπει στους χρήστες να αναζητούν, να γράφουν μηνύματα και να πραγματοποιούν κλήσεις μέσω φωνής. Εναλλακτικά οι χρήστες μπορούν να εγκαταστήσουν μια εφαρμογή απευθείας στη συσκευή και χωρίς τη χρήση του Google Play έχοντας το εκτελέσιμο αρχείο APK ή μέσω τρίτων όπως το Amazon App store.

3.11. Google Play Developers



Εικόνα 1.32: Βασική διαδικασία ανάπτυξης και καταχώρησης Android εφαρμογής

Είναι πια πολύ εύκολο για τους προγραμματιστές εφαρμογών Android να ανεβάσουν ή και να προωθήσουν την εφαρμογή τους στο Google Play. Η διαδικασία για να ανεβάσει ένα προγραμματιστής την εφαρμογή του είναι σχετικά απλή και χρειάζεται ένα ενδεικτικό ποσό καταβολής για μία μόνον φορά. Το Google Play δίνει διάφορες δυνατότητες στους προγραμματιστές των εφαρμογών αλλά και των χρηστών παρέχοντας ένα σύστημα αξιολόγησης της εφαρμογής, εικόνες της εφαρμογής και άλλα στοιχεία όπως την έκδοση της εφαρμογής και το μέγεθος της. Επίσης, υπάρχουν πολλά στατιστικά όπως π.χ πόσοι χρήστες κατέβασαν την εφαρμογή σε συγκεκριμένο χρονικό διάστημα.

Αναλυτικότερα, για να κάνουμε publish την εφαρμογή μας στο Google Play θα πρέπει η εφαρμογή μας να τηρεί κάποιους βασικούς κανόνες που έχει συμβουλέψει η Google τους προγραμματιστές.

- Η εφαρμογή μας, δηλαδή το apk αρχείο δεν πρέπει να ξεπερνάει τα 50 Megabyte (όριο που έχει βάλει η Google)
- Πρέπει να διαγράψουμε τυχών Log κλήσεις από τον κώδικα μας και να μην υπάρχει το χαρακτηριστικό android: debuggable στο manifest αρχείο.
- Να παρέχουμε τιμές για τις μεταβλητές android:versionCode και android:versionName στο manifest αρχείο μας.

- Να κάνουμε έναν ενδελεχή έλεγχο της εφαρμογής σε τουλάχιστον μία κινητή συσκευή πριν την ανεβάσουμε στο Google Play.
- Να βεβαιωθούμε ότι δουλεύουν τα pathμας και οι κλήσεις που κάνει η εφαρμογή μας σεκάποιο απομακρυσμένο server.
- Να βεβαιωθούμε ότι χρησιμοποιούμε τα σωστά API κλειδιά (π.χ για Google Maps) στο κώδικα μας.

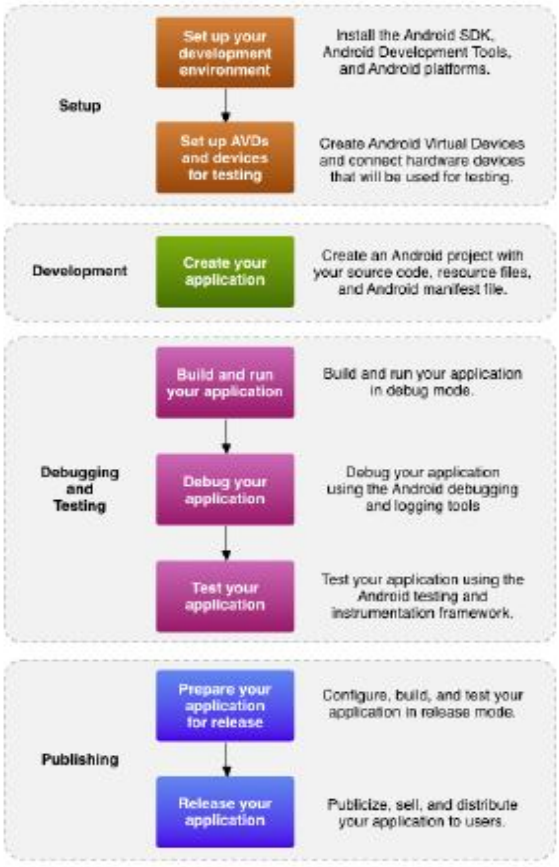
Αν θέλουμε να διαδώσουμε την εφαρμογή μας μέσω ξεχωριστής ιστοσελίδας μπορούμε να το κάνουμε αλλά δεν συνίσταται από την Google. Αν π.χ η εφαρμογή μας έχει κάποιο σύστημα συναλλαγών ή χρειάζεται πληρωμή για κατέβασμα, θα πρέπει να φτιάξουμε κάποιο δικό μας σύστημα πληρωμών, οπότε δεν θα μπορούμε να χρησιμοποιήσουμε την in-appυπηρεσία πληρωμών που παρέχει η Google (Google in-app billing). Επίσης, δεν θα μπορούμε να έχουμε την υπηρεσία δικαιωμάτων της Google που απαγορεύει παράνομη χρήση της εφαρμογής, ειδικά όταν θέλουμε να επιβαιώσουμε ότι μια συναλλαγή μέσα από την εφαρμογή μας έγινε επιτυχώς και σωστά μέσα από τις επίσημες υπηρεσίες του Google Play.

3.12. Android Tools

Η ανάπτυξη εφαρμογών για τις συσκευές Android διευκολύνεται από μια ομάδα εργαλείων που παρέχονται στο SDK (Software Development Kit). Μπορείτε να αποκτήσετε πρόσβαση σε αυτά τα εγαλεία μέσω ενός plugin που ονομάζεται Eclipse ADT (Android Development Tools) ή από τη γραμμή εντολών. Η ανάπτυξη με το Eclipse είναι η προτιμώμενη μέθοδος, διότι μπορεί να επικαλεστεί άμεσα τα εργαλεία που χρειάζεστε, ενώ αναπτύσσετε τις εφαρμογές.

Ωστόσο, μπορείτε να επιλέξετε να αναπτύξετε με άλλο IDE (Integrated Development Environment) ή έναν απλό επεξεργαστή κειμένου και να επικαλεστείτε τα εργαλεία μέσω της γραμμής εντολών ή με scripts. Αυτός ο τρόπος ανάπτυξης είναι λιγότερο εναρμονισμένος, επειδή μερικές φορές θα πρέπει να καλέσετε τα εργαλεία της γραμμής εντολών χειροκίνητα, αλλά θα πρέπει να έχετε πρόσβαση στον ίδιο αριθμό χαρακτηριστικών που θα είχατε και στο Eclipse.

Τα βασικά βήματα για την ανάπτυξη εφαρμογών (με ή χωρίς Eclipse) φαίνονται στην παρακάτω εικόνα. Τα βήματα περιλαμβάνουν την ανάπτυξη τέσσερις φάσεις ανάπτυξης, οι οποίες περιλαμβάνουν εγκατάσταση, ανάπτυξη, δοκιμή και δημοσίευση.



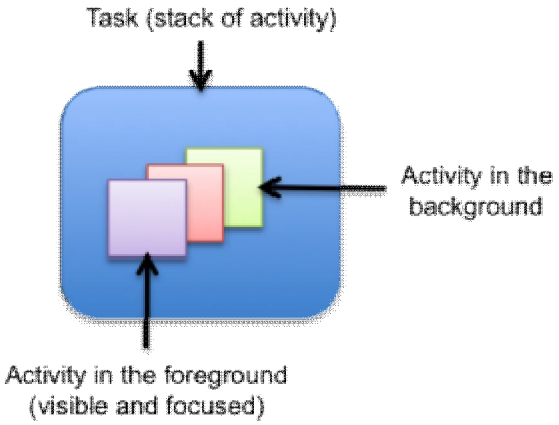
Εικόνα 1.33: Η πορεία ανάπτυξης μιας εφαρμογής Android

3.13. Activities

Ένα Activity δίνει τη δυνατότητα στο χρήστη να επικοινωνεί και να αλληλεπιδρά με την εφαρμογή. Με την Activity κλάση δημιουργείται στην ουσία ένα παράθυρο που μπορούν οι προγραμματιστές να ενσωματώσουν το εκάστοτε User Interface μέσω του setContentView(view). Οι Activities τις περισσότερες φορές παρουσιάζονται στο χρήστη σε full-screen μορφή αλλά άλλες φορές μπορούμε να τις δούμε σε floating windows ή Activity μέσα σε Activity με τη χρήση Activity Group.

Όταν τρέξει για πρώτη φορά ένα Activity πηγαίνει στο παρασκήνιο του λειτουργικού συστήματος και αργότερα εμφανίζεται στην οθόνη του χρήστη. Κατά τη διάρκεια αυτής της επεξεργασίας το Android λειτουργικό καλεί μια σειρά από μεθόδους του κύκλου ζωής ενός Activity (lifecycle methods) . Αν ο χρήστης επιλέξει να μεταβεί σε ένα άλλο Activity, τότε, το σύστημα καλεί μια άλλη σειρά από μεθόδους κύκλου ζωής στο Activity όσο εκείνο μεταφέρεται στο παρασκήνιο όπου το Activity αυτό δεν είναι πια ορατό αλλά το στιγμιότυπο του και η κατάσταση του παραμένουν ανεπηρέαστα.

Μέσα από την ανάκληση των μεθόδων του κύκλου ζωής μπορούμε να ορίσουμε πώς θα συμπεριφέρεται το Activity μας, όταν ας πούμε ο χρήστης φύγει από αυτό το Activity και ξαναμπεί αργότερα. Για παράδειγμα αν έχουμε μια streaming αναπαραγωγή πολυμέσων θα μπορούσαμε να κάνουμε pause το βίντεο και να κλείσουμε την επικοινωνία με τον σέρβερ όταν ο χρήστης ανοίξει μια άλλη διαφορετική εφαρμογή. Όταν ο χρήστης επιστρέψει στην δική μας εφαρμογή – Activity θα ξανασυνδέεται με τον σέρβερ και θα συνεχίσει να παίζει το βίντεο από το σημείο που το άφησε.



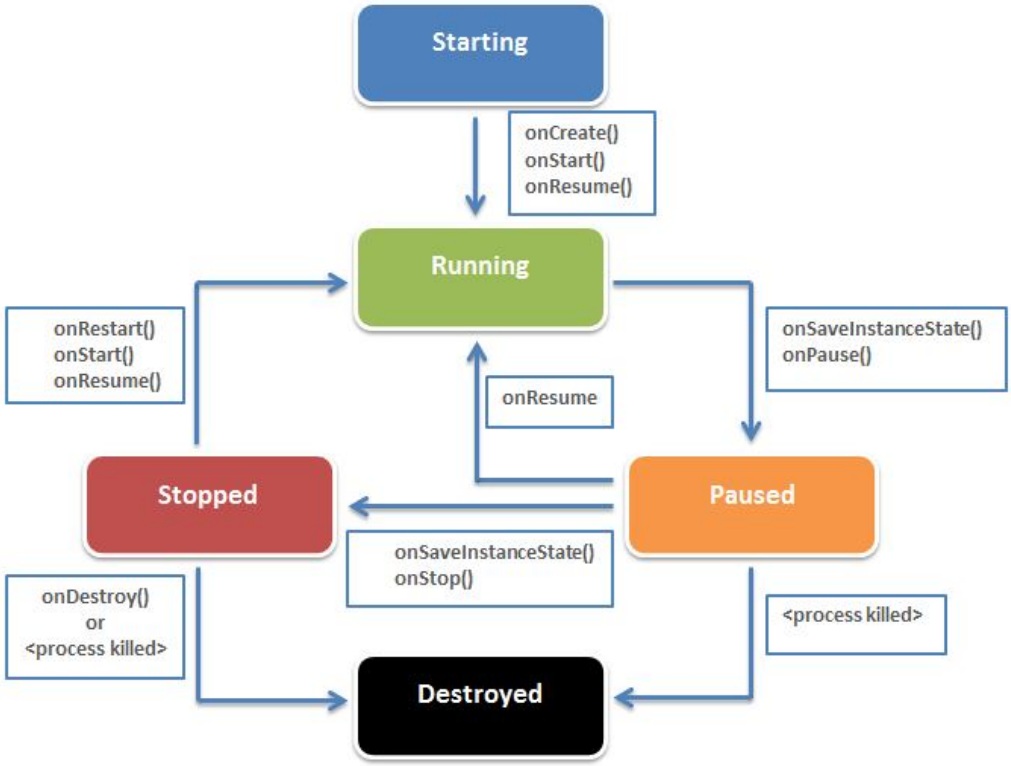
Εικόνα 1.34: Μια στοίβα από Activities

3.14. Κύκλος ζωής ενός Activity

Κατά τη διάρκεια μιας Activity, το σύστημα καλεί ένα σύνολο του κύκλου ζωής των μεθόδων σε μια ακολουθία παρόμοια με μια πυραμίδα. Δηλαδή, κάθε στάδιο του κύκλου ζωής του Activity είναι και ένα ξεχωριστό βήμα στην πυραμίδα. Δεδομένου ότι το σύστημα δημιουργεί ένα νέο instance Activity, κάθε μέθοδος επανάκλησης κινεί το Activity ένα βήμα προς την κορυφή. Η κορυφή της πυραμίδας είναι το σημείο στο οποίο η Activity εκτελείται στο προσκήνιο και ο χρήστης μπορεί να αλληλεπιδράσει με αυτό.

Καθώς ο χρήστης φεύγει από το Activity, το σύστημα καλεί άλλες μεθόδους που κινούνται στο πίσω μέρος του Activity κάτω από την πυραμίδα, προκειμένου να διαλύσει το Activity. Σε ορισμένες περιπτώσεις, το Activity θα κινείται μόνο κατά ένα μέρος κάτω από την πυραμίδα και να περιμένει, όπως όταν ο χρήστης μεταβαίνει σε μια άλλη εφαρμογή.

Αργότερα, η Activity μπορεί να κινηθεί πίσω στην κορυφή (εάν ο χρήστης επιστρέψει στην Activity) και να ξαναξεκινήσει την λειτουργία της από το οποίο σημείο που σταμάτησε.



Εικόνα 1.35: Κύκλος ζωής των δραστηριοτήτων (Activities)

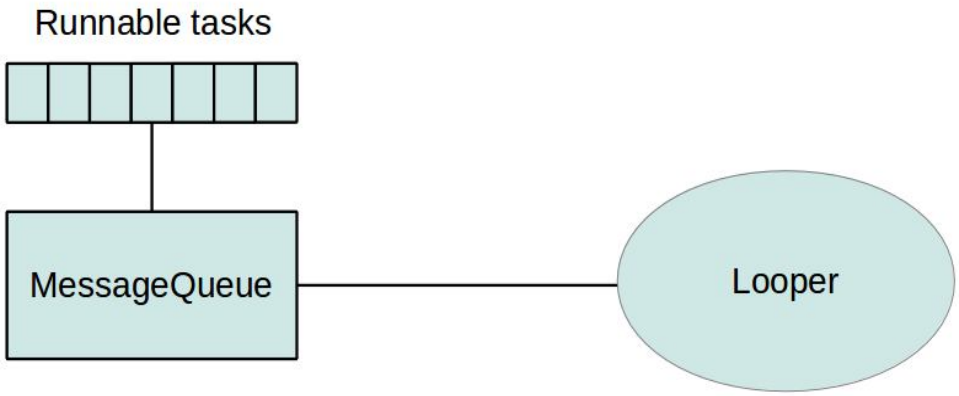
Οι περισσότερες εφαρμογές περιλαμβάνουν πολλές διαφορετικές δραστηριότητες (Activities) που επιτρέπουν στο χρήστη να εκτελέσει διάφορες ενέργειες. Π.χ το αν μια δραστηριότητα είναι η κύρια δραστηριότητα που δημιουργείται όταν ο χρήστης κάνει κλικ στο εικονίδιο της εφαρμογής.

Εμείς σαν προγραμματιστές θα πρέπει να εφαρμόσουμε την onCreate() μέθοδο για να πραγματοποιηθεί η βασική εκκίνηση της εφαρμογής που πρέπει να συμβεί μόνο μία φορά για όλη τη διάρκεια της δραστηριότητας. Για παράδειγμα, όταν η εφαρμογή μας φτάσει στην onCreate() θα πρέπει να καθοριστεί η διεπαφή χρήστη και ενδεχομένως να αρχικοποιηθούν κάποιες μεταβλητές από κλάσεις.

3.15. Android Threads

Θα νόμιζε κανείς ότι όταν καλέσουμε π.χ την setText() σε ένα TextView, η οθόνη αναβαθμίζεται με το κείμενο που ορίσαμε αμέσως τότε και εκεί. Αλλά στην ουσία, η διαδικασία γίνεται αλλιώς. Οτιδήποτε αναβαθμίζει το widget-based UI περνάει μέσα από μια ουρά μηνυμάτων. Η κλήση της setText() δεν αναβαθμίζει την οθόνη μας. Απλά αποβάλλει ένα μήνυμα από την ουρά λέγοντας στο λειτουργικό σύστημα να αναβαθμίσει την οθόνη. Το λειτουργικό σύστημα πετάει τα μηνύμα έξω από την ουρά και κάνει ότι του στείλουμε. Η ουρά αυτή επεργάζεται από ένα thread που ονομάζεται το main application thread και UI Thread. Όσο αυτό το thread μπορεί και επεξεργάζεται τα μηνύματα η οθόνη μας θα αναβαθμίζεται με την σειρά που εμείς έχουμε ορίσει.

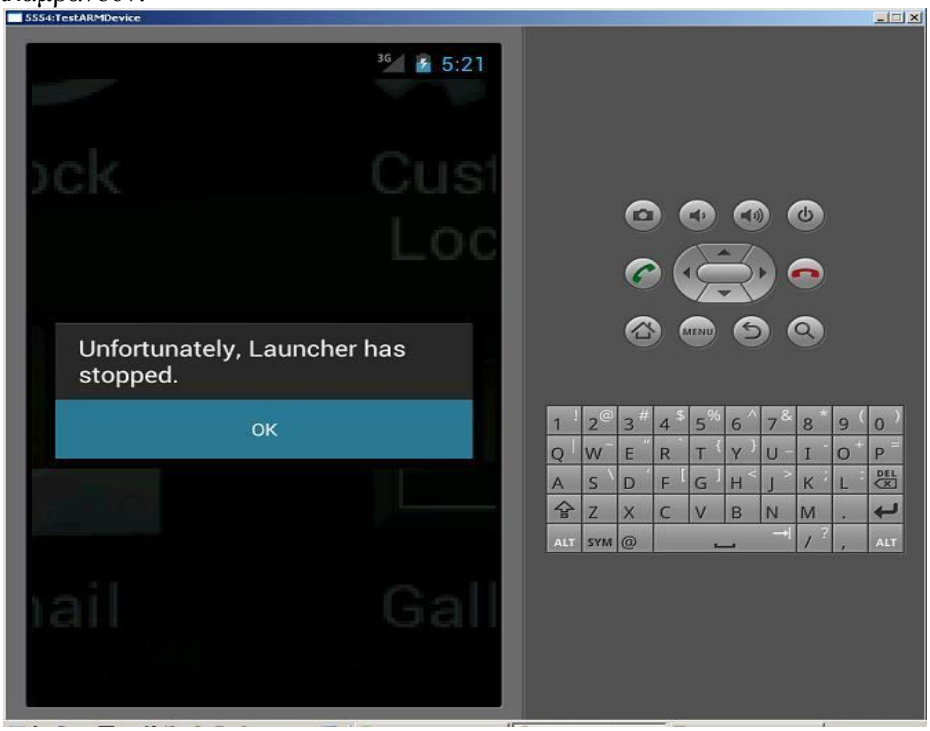
Το Android μετά, συλλέγει όλα τα γεγονότα στην ουρά και εκτελείται ένα instance της κλάσης Looper που εκτελεί την ουρά μηνυμάτων. Με άλλα λόγια η κλάση Looper είναι υπεύθυνη για να δημιουργήσει την ουρά των μηνυμάτων στο thread.



Εικόνα 1.36: Ουρά μηνυμάτων και η κλάση Looper

Αλλά, το κακό βρίσκεται στο ότι το UI Thread χρησιμοποιείται για σχεδόν όλα τα callback που κάνουμε στο Activity. Όπως την onCreate(), onClick(), onListItemClick() και άλλες παρόμοιες μεθόδους που καλούνται όλες από το main application Thread. Όσο ο κώδικας μας εκτελεί αυτές τις μεθόδους το Android ταυτόχρονα δεν επεξεργάζεται σωστά τα μηνύματα στην ουρά που κάνει την οθόνη μας να μην αναβαθμίζεται και η ροή δεδομένων στον χρήστη να μην γίνεται ομαλά.

Αυτό είναι ένα κακό στοιχείο που δεν πρέπει να υπάρχει στην εφαρμογή μας. Γεγονός είναι ότι αν η διεργασία που εκτελεί το main application Thread πάρει πάνω από μερικά δευτερόλεπτα (5 δευτερόλεπτα αναμονής θεωρούνται αρκετά για ένα Activity) τότε πολύ πιθανό είναι να δούμε στην οθόνη μας το περίφημο «Application not responding» (ANR) error που θα κάνει το Activity μας να σταματήσει . Γι'αυτό πρέπει να σιγουρευτούμε ότι οι διεργασίες που εκτελούνται στο main application Thread γίνονται γρήγορα. Αυτό πρακτικά σημαίνει ότι οποιαδήποτε διεργασία αργεί να εκτελεστεί πρέπει να μπει σε ένα background thread που είναι διαφορετικό από το main application Thread. Τέτοιες διεργασίες συμπεριλαμβάνουν:



Εικόνα 1.37: Το «Application has stopped» error

- Πρόσβαση σε webserver / Internet / web service, αποστολή δεδομένων ή κατέβασμα μιας εικόνας.
- Διεργασίες αρχείων, flash storage που μπορεί να γίνει πολύ αργό.
- Γενικά, πολύπλοκες πράξεις και μετατροπές στον κώδικα μας.

Αφού τρέξει το background Thread μας, αυτό από μόνο του δεν μπορεί να αναβαθμίσει το UI μας. Αυτό μπορεί να το κάνει μόνο το main application Thread. Παρακάτω θα αναφέρουμε μερικά εργαλεία που μπορούν να ενώσουν την αναβάθμιση του UI με το background Thread.

3.16. Η κλάση Handler, ο σκοπός της και το στιγμιότυπο της

Η κλάση Handler μπορεί να χρησιμοποιηθεί για να δηλώσει ένα thread και να δημιουργήσει ένα κανάλι επικοινωνίας όπου θα στέλνονται δεδομένα σε αυτό το thread.

Υπάρχουν δύο βασικές χρήσεις για την κλάση Handler. Πρώτον, για να προγραμματίσουμε μηνύματα που θα εκτελεστούν σε κάποιο σημείο στο μέλλον της εφαρμογής και δεύτερον για να προσθέσουμε στην ουρά μια ενέργεια που θα εκτελεστεί σε διαφορετικό thread από εκείνο που ήδη βρίσκεται η εφαρμογή εκείνη τη στιγμή.

Ο προγραμματισμός των μηνυμάτων επιτυχνάνεται με διάφορες μεθόδος όπως τις post(Runnable), postAtTime(Runnable, long), postDelayed(Runnable, long). Αυτές οι μεθόδοι μας επιτρέπουν να προσθέτουμε στην ουρά αντικείμενα τύπου Runnable για να επεξεργαστούν και να εκτελεστούν.

Παρακάτω βλέπουμε ένα παράδειγμα δημιουργίας και εκτέλεσης (με την εντολή start) ενός νέου Thread.

```
Thread thread = new Thread()
{
   @Override
   public void run() {
      try {
         while(true) {
            sleep(1000);
            handler.post(r);
         }
      } catch (InterruptedException e) {
         e.printStackTrace();
      }
   }
};
```

thread.start();

3.17. Asynctask

Η AsyncTask επιτρέπει την σωστή και εύκολη χρήση του UI Thread. Η κλάση αυτή μας επιτρέπει να εκτελέσουμε λειτουργίες στο παρασκήνιο και να δημοσιεύονται τα αποτελέσματα στο UI Thread χωρίς να χρειάζεται να τα χειριστούν τα threads ή / και οι handlers.

Η AsyncTask έχει σχεδιαστεί για να είναι μια κλάση αρωγός γύρω από το Thread και τον Handler και δεν αποτελεί ένα γενικό πλαίσιο threading. Τα AsyncTasks θα πρέπει ιδανικά να χρησιμοποιηθούν για μικρής διάρκειας χειρισμούς (λίγα δευτερόλεπτα το πολύ.) Εάν πρέπει να κρατήσει το thread για μεγάλο χρονικό διάστημα, συνιστάται να χρησιμοποιήσουμε τα διάφορα APIs που παρέχονται από την java.util.concurrentpackage όπως Thread Pool Executor και Future Task. Στην δική μας εφαρμογή η AsyncTask ήταν αρκετή για τις διεργασίες που εκτελούμε.

Ο σκοπός δηλαδή της κλάσης AsyncTask είναι να εμπεριέχει τη δημιουργία μιας διεργασίας στο παρασκήνιο (background process) και να συγχρονίζεται με το main Thread. Επίσης όπως θα αναφέρουμε παρακάτω, υποστηρίζει και αναφορά της προόδου των διεργασιών που εκτελούνται.

3.17.1. Τα στάδια της AsyncTask

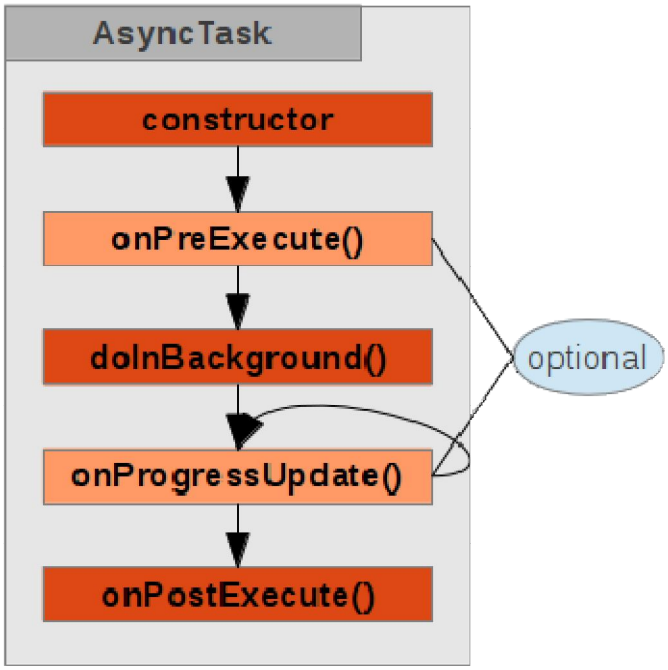
Η AsyncTask έχει τέσσερα στάδια:

onPreExecute: Αυτή η μέθοδος εκτελείται πριν την doInBackground.

doInBackground: Η doInBackground καλείται αμέσως μετά την onPreExecute(), αφότου δηλαδή η onPreExecute() τελειώσει την εκτέλεση της. Η doInBackground χρησιμοποιείται για υπολογισμούς που παίρνουν ώρα να εκτελεστούν, δηλαδή είναι το βασικό βήμα για να εκτελέσουμε τον κώδικα που θέλουμε. Οι παράμετροι της AsyncTask παίρνονται από αυτό το στάδιο και το αποτέλεσμα επιστρέφεται στο επόμενο βήμα που αναλύουμε παρακάτω.

onPostExecute: Αυτή η μέθοδος εκτελείται αφότου η doInBackground ολοκληρώσει την κλήση της. Το αποτέλεσμα από την doInBackround περνιέται σε αυτή τη μέθοδο.

onProgressUpdate: Αυτή η μέθοδος καλείται καλώντας την publishProgress ανά πάσα στιγμή ενώ εκτελείται η doInBackground. Συνήθως η onProgressUpdate χρησιμοποιείται για να ενημερώσουμε το UI Thread με κάποια μπάρα προόδου ενώ εκτελείται η doInBackground.



Εικόνα 1.38: Μεθόδοι της AsyncTask

3.17.2. Το βασικό πλεονέκτημα της AsyncTask

Το λειτουργικό Android εφαρμόζει ένα μοντέλου ενιαίου Thread και οποτεδήποτε μια Android εφαρμογή ξεκινάει, δημιουργείται και ένα αντίστοιχο Thread. Τώρα, υποθέτοντας ότι εκτελούμε μια λειτουργία δικτύου με το πάτημα ενός κουμπιού στην εφαρμογή μας. Όταν πατήσουμε το κουμπί θα γίνει μια αίτηση στον σέρβερ και έτσι πρέπει να περιμένουμε την απάντηση του σέρβερ. Εξαιτίας του ενιαίου μοντέλου στο Thread, όση ώρα περιμένουμε την απάντηση του σέρβερ, η εφαρμογή μας δεν θα ανταποκρίνεται. Έτσι χρησιμοποιούμε την AsyncTask για να αποφύγουμε λειτουργίες που παίρνουν αρκετό σχετικά χρόνο να εκτελεστούν και κάνουν το user interface της εφαρμογής μας να μην ανταποκρίνεται.

Έτσι δημιουργούμε ένα νέο Thread και τρέχουμε εκεί τη λειτουργία μας κάνοντας το βασικό Thread της εφαρμογής να παραμένει ανταποκρίσιμο στο χρήστη. Η κλάση AsyncTask περιέχει δηλαδή μεθόδους για να τρέξουμε μια λειτουργία που απαιτεί κάποιο χρόνο, στο παρασκήνιο, αλλά ταυτόχρονα μπορεί να επικοινωνεί και με το Thread της εφαρμογής και να συγχρονίζεται με αυτό ή και να το ενημερώνει όταν χρειάζεται.

3.17.3. Χρησιμοποιώντας την κλάση AsyncTask

Στον κώδικα μας χειριζόμαστε την AsyncTask ώς μια υποκλάση. Η AsyncTask χρησιμοποιεί varargs και οι παραμέτρους της είναι AsyncTask<TypeOfVarArgParams , ProgressValue , ResultValue>.

```
protected String doInBackground(String... args) {}
```

Οι τρεις τελείες λέγονται αλλιώς και εκκλείψεις. Τα varargs πρέπει να είναι πάντα η τελευταία παράμετρο στην μέθοδο μας.

Ένα AsyncTask ξεκινάει μέσω της execute() μεθόδου. Η μέθοδος execute() καλεί την doInBackground() και την onPostExecute() μέθοδο. Η TypeOfVarArgParams περνάει στην doInBackground() μέθοδο ως παράμετρος ενώ η ProgressValue χρησιμοποιείται για πληροφορίες προόδου. Η παράμετρος ResultValue πρέπει να επιστραφεί από την doInBackground μέθοδο και να περαστεί στην onPostExecute() σαν παράμετρος.

3.18. ProgressBars

Είναι συνετό όταν τρέχουμε στην εφαρμογή μας ένα background thread να ενημερώνουμε το χρήστη ότι η εφαρμογή μας εκτελεί μια διεργασία εώς ότου εκείνη ολοκληρωθεί. Μια συνήθης προσέγγιση αυτού του ζητήματος επιτυγχάνεται με το progressBar. Το progressBar ορίζεται από έναν integer αρχίζοντας από το 0 (προεπιλεγμένη τιμή). Μπορούμε να ορίσουμε εμείς το εύρος της μέγιστης τιμής δηλαδή σε ποια τιμή η πρόοδος έχει ολοκληρωθεί, με την setMax().

Στον κώδικα μας μπορούμε να βάλουμε χειροκίνητα μια θετική τιμή για να δείξουμε την πρόοδο που έχει γίνει (setProgress()) ή να ανεβάζουμε σταδιακά την πρόοδο με την incrementProgressBy(). Με την getProgress() μπορούμε να πάρουμε την θετική τιμή της προόδου που έχει γίνει.

3.19. ProgressDialog

Τα progressDialog είναι πολύ χρήσιμα στις εφαρμογές Android γιατί ειδοποιούν το χρήστη ότι μια διεργασία εκτελείται κάνοντας τα ένα βασικό εργαλείο στην δημιουργία μιας εφαρμογής και στην σωστή διεπαφή με το χρήστη. Ειδικά όταν μια διεργασία αργεί να εκτελεστεί, το progressDialog κάνει το χρήστη να περιμένει και να μην νομίζει ότι η εφαρμογή έχει παγώσει.

Τα progressDialog πρέπει να οριστούν μέσα στο Activity που χρησιμοποιούνται. Ένα progressDialog μπορεί να χρησιμοποιηθεί περισσότερες από μια φορές. Βλέπουμε παρακάτω πως μπορούμε να δείξουμε ένα απλό progressDialog στην οθόνη της εφαρμογής.

```
pDialog = newProgressDialog(Activity.this);
                        pDialog.setMessage("Αποθήκευση Αλλαγών ...");
                        pDialog.setIndeterminate(false);
                        pDialog.setCancelable(true);
                        pDialog.show();
```

Ας δούμε λίγο τις πιο χρησιμοποιούμενες μεθόδους για το progressDialog που έχουμε και στην εφαρμογή μας.

Show() : Αυτή η μέθοδος χρησιμοποιείται για να δείξει στην οθόνη μας ένα progressDialog.

dismiss(); : Με αυτή τη μέθοδο σταματάει η προβολή του progressDialog στην οθόνη μας. Το progressDialog όμως μένει στην προσωρινή μνήμη του Activity. Έτσι όταν ξανακαλέσουμε το ίδιο progressDialog θα φανεί ξανά στην οθόνη η cached εκδοχή αυτού.

Cancel() : Περίπου ίδια με την dismiss. Η βασική διαφορά είναι ότι καλείται και το DialogInterface.OnCancelListener αν είναι δηλωμένο.

Εικόνα 1.39: Το progressDialog.

3.20. Κατανόηση JAVA SE και το Dalvik Virtual Machine

Το Android runtime περιβάλλον παρέχει ένα σύνολο βασικών βιβλιοθηκών της λειτουργίας του συστήματος που μπορεί να παρέχει πρόσβαση μέσω Java και XML. Αυτές οι λειτουργίες μας δίνουν πρόσβαση στα χαρακτηριστικά μιας συσκευής Android και στο χαμηλότερο επίπεδο του λειτουργικού συστήματος Android, για να μην χρειαζόμαστε να κάνουμε όλο τον "low-level" προγραμματισμό εμείς οι developers. Εμείς απλά πρέπει να καταχωρήσουμε τις σωστές βιβλιοθήκες (libraries) στην εφαρμογή μας και μετά μέσω των built-in δυνατοτήτων του Android θα τις εκμεταλλευτεί εκείνο κατάλληλα.

Για να τρέξουμε κώδικα JavaSE, το Android χρησιμοποιεί μια λειτουργία που λέγεται Dalvik Virtual Machine (DVM). Το DVM είναι ένας μηχανισμός βελτιστοποίησης με τεχνολογίες που επιτρέπει στον κώδικα των εφαρμογών να είναι όσο τον δυνατόν περισσότερο βελτιστοποιημένα για την χρήση τους σε περιβάλλον κυρίως κινητών τηλεφώνων και συσκευών.

Επίσης υπάρχουν και τα ενσωματωμένα (embedded) περιβάλλοντα που στην ουσία ένα υπολογιστικό σύστημα είναι ενσωματωμένο μέσα στην συσκευή του χρήστη, όπως smartphones, tablets, e-book readers , iTV sets iDVD players και άλλα.

Όταν ξεκινάμε μια εφαρμογή Android, δημιουργείται μια διαδικασία που χρησιμοποιεί μνήμη και πόρους απο τον επεξεργαστή (CPU) και χρησιμοποιεί την κατάλληλη ποσότητα που απαιτείται για να λειτουργήσει. Η DVM παίρνει τις οδηγίες από τη γλώσσα JAVA μαζί με σχετικές γραμμές σχεδιασμού (σε μορφή XML) και τις συνδυάζει με οποιεσδήποτε εξωτερικές πηγές (εικόνες, αρχεία ήχου ή βίντεο 3D και ούτω καθεξής) και μεταφράζει όλα αυτά σε έναν βελτιστοποιημένο χαμηλού επιπέδου δυαδικό κώδικα που πηγαίνει στη μνήμη της συσκευής του Android και τελικά στον επεξεργαστή.

Ποιό είναι λοιπόν το όφελος της DVM; Με την DVM μας επιτρέπεται να τρέχουμε πολλές εφαρμογές σε περιορισμένους πόρους μνήμης (1GB και 2GB) και επεξεργαστική ισχύ (1GHz εώς 2GHz) και αποτρέπει τις εφαρμογές να παρεμβάλλονται μεταξύ τους. Έτσι

η διακοπή λειτουργίας μιας εφαρμογής (crash) δεν θα αναγκάσει να διακοπεί όλη η λειτουργία του λειτουργικού συστήματος και αυτό θεωρείται ένα μεγάλο πλεονέκτημα.

3.21. Η δομή των φακέλων σε ένα Android Project

Το Android προσπαθεί όσο το δυνατό περισσότερο να διαχωρίσει τα στοιχεία της εφαρμογής που δεν χρειάζεται να βρίσκονται στον κώδικα της Java. Το πραγματοποιεί αυτό χρησιμοποιώντας μια απλούστερη μορφή XML σήμανσης για να ορίσει το περιβάλλον του χρήστη (User Interface) που αλλιώς θα έπρεπε να οριστεί ξεχωριστά στον κώδικα μας. Αυτή η διευκόλυνση μας βοηθάει.

Το Android είναι πολύ συγκεκριμένο στην θέση που πρέπει να βρίσκονται τα αρχεία (assets) του πρότζεκτ μας. Κάθε φορά που τρέχουμε την εφαρμογή (compile) το Android ψάχνει να εντοπίσει αν υπάρχει η σωστή δομή στους φακέλους του πρότζεκτ μας και περιμένει να είναι όλα στη σωστή τους θέση για να λειτουργήσει σωστά η εφαρμογή. Τα assets της εφαρμογής συμπεριλαμβάνουν τον κώδικα Java, XML ορισμούς και την διασύνδεση μεταξύ τους.

Ο κώδικας Java που καθοδηγεί την εφαρμογή βρίσκεται στον /src φάκελο του πρότζεκτ μας μαζί με άλλους υποφακέλους που ίσως είναι ορισμένοι στον κώδικα μας.
Άλλα assets βρίσκονται στον /res (resources) υποφάκελο. Είναι σημαντικό για τον μεταγλωτιστή Android που χρησιμοποιούμε να μην βρίσκονται αρχεία στον /res φάκελο, παρά μόνο υποφάκελοι αλλιώς θα υπάρξει σφάλμα στη μεταγλώττιση.
Οι πιο κοινοί υποφάκελοι στην /res περιοχή είναι:

- **Layout**

Εδώ έχουμε τα λεγόμενα UI Screen Layouts (διατάξεις οθόνης) που περιέχουν XML τύπου και κώδικα,αρχεία που ορίζουμε τις διατάξεις της εκάστοτε οθόνης της εφαρμογής μας.

- **Drawable-hdpi**

Περιέχονται υψηλής ανάλυσης εικόνες σε μορφή PNG (το οποίο η Google προτιμά) ή η μορφή JPEG (αποδεκτή, αλλά δεν προτείνεται από την Google) πάει στο / res / drawable-hdpi (υψηλή ανάλυση οθόνης εικόνες, συνήθως 800 με 480 pixels αναλογία.

- **drawable-ldpi**

Περιέχονται χαμηλής ανάλυσης εικόνες σε μορφή PNG (το οποίο η Google προτιμά) ή η μορφή JPEG (αποδεκτή, αλλά δεν προτείνεται από την Google) πάει στο / res / drawable- ldpi (υψηλή ανάλυση οθόνης εικόνες, συνήθως 320 με 240 pixels αναλογία).

- **drawable-mdpi**

Περιέχονται μέτριας ανάλυσης εικόνες σε μορφή PNG (το οποίο η Google προτιμά) ή η μορφή JPEG (αποδεκτή, αλλά δεν προτείνεται από την Google) πάει στο / res / drawable-mdpi (υψηλή ανάλυση οθόνης εικόνες, συνήθως 480με 320 pixels αναλογία).

- **drawable-xhdpi**

Περιέχονται υπερυψηλής ανάλυσης εικόνες σε μορφή PNG (το οποίο η Google προτιμά) ή σε μορφή JPEG (αποδεκτή, αλλά δεν προτείνεται από την Google) πάει στο / res / drawable-xhdpi (υψηλή ανάλυση οθόνης εικόνες, συνήθως 1280με 720 pixels αναλογία).

- **Values**

Περιέχονται XML αρχεία που καθορίζουν σταθερές τιμές σε διάφορες μεταβλητές στο πρότζεκτ μας (res/values) υποφάκελος.

- **values-v11**

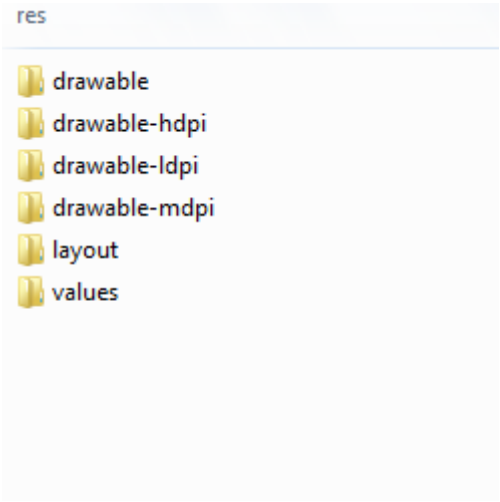
Περιέχονται honeycomb theme XML αρχεία που καθορίζουν UI theme values ((API Level 11 έως 13, επίσης γνωστή ως Android 3.x ή 3.0, 3.1 και 3.2)

- **values-v14**

Περιέχονται Ice Cream Sandwich theme XML αρχεία που καθορίζουν UI theme values (API Level 14 έως 16, γνωστή ως Android 4.xή 4.0, 4.0.3 και 4.1)

- **menu**

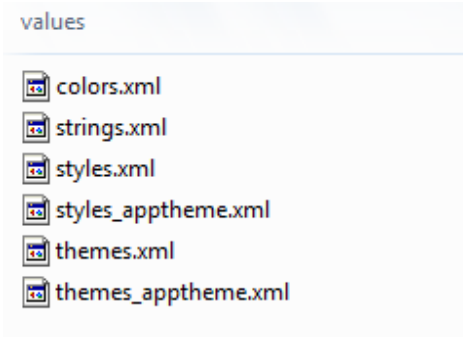
Περιέχονται XML αρχεία που καθορίζουν διατάξεις του μενού UI theme values.



Εικόνα 1.40: Μια τυπική δομή του φακέλου res σε ένα Android Project

Ο φάκελος Values

Εδώ μπορούμε να τοποθετήσουμε προκαθορισμένες τιμές της εφαρμογής με τη μορφή των αρχείων XML που καθορίζουν τα ονόματα των μεταβλητών (x ή y, για παράδειγμα) και τις τιμές τους, τα οποία στη συνέχεια αναφέρονται στον δικό μας κώδικα Java. Για παράδειγμα αυτές οι τιμές θα μπορούσαν να είναι strings (χαρακτήρες κειμένου) ή σταθερές που χρησιμοποιούμε στον κώδικα μας.



Εικόνα 1.41: Μια τυπική δομή του φακέλου values σε ένα Android Project

colors.xml:

Ένα αρχείο XML που θα καθορίσει τις τιμές των χρωμάτων που πρέπει να χρησιμοποιούνται στην εφαρμογή.Αυτές μας επιτρέπουν να τυποποιήσουμε το UI. Για παράδειγμα, μπορούμε να ορίσουμε το χρώμα του φόντου.

dimens.xml:

Ένα αρχείο XML που καθορίζει τις τιμές διαστάσεων, όπως τα τυποποιημένα ύψη και μεγέθη γραμματοσειράς για το UI μας.

arrays.xml:

Ένα αρχείο XML που ορίζει μια σειρά τιμών που πρέπει να χρησιμοποιούνται σε συνδυασμό (γνωστό και ως array). Για παράδειγμα, αυτό θα μπορούσε να είναι μια λίστα των αρχείων εικονιδίων ή μια λίστα επιλογών που θα εμφανίζονται στο χρήστη.

strings.xml:

Ένα αρχείο XML που καθορίζει strings (κείμενο) που θα χρησιμοποιηθούν στην εφαρμογή. Για παράδειγμα, θα μπορούσαμε να τοποθετήσουμε οποιοδήποτε τίτλο να εμφανίζεται στη οθόνη ή το όνομα της εφαρμογής. Εάν πρέπει να αλλάξουμε αυτά τα στοιχεία, μπορούμε να το κάνουμε απλά εδώ και όχι απαραίτητα στον κώδικά μας.

styles.xml:

Ένα αρχείο XML που καθορίζει τα styles που πρέπει να χρησιμοποιούνται στην εφαρμογή. Αυτά τα styles εφαρμόζονται στη συνέχεια στα UI στοιχεία που τους χρειάζονται, έτσι ώστε να διαχωριστεί η εμφάνιση της εφαρμογής μας από τη διάταξη και τη λειτουργικότητα.

3.22. Το AndroidManifest.xml αρχείο

Τα διάφορα components και οι ρυθμίσεις μιας Android εφαρμογής περιγράφονται στο αρχείο AndroidManifest.xml. Όλες οι δραστηριότητες (activities), οι υπηρεσίες και άλλα στοιχεία παρόχων περιεχομένου της εφαρμογής πρέπει να δηλώνονται στατικά σε αυτό το αρχείο. Το manifest αρχείο θα πρέπει επίσης να περιέχει τα απαιτούμενα δικαιώματα για την εφαρμογή. Για παράδειγμα, αν η εφαρμογή απαιτεί πρόσβαση στο δίκτυο, θα πρέπει να προσδιορίζονται μέσα σε αυτό το αρχείο. Με τη χρήση του intent filter μέσα στο AndroidManifest.xml αρχείο μιας εφαρμογής λέμε στο Android runtime να καταχωρηθεί η δραστηριότητα που δηλώσαμε ως σημείο εισόδου (δηλαδή η αρχική οθόνη) μιας εφαρμογής. Παρακάτω βλέπουμε πως μπορούμε να βάλουμε ένα intent filter στη δήλωση μιας activity για να την κάνουμε να τρέχει πρώτη όταν ανοίγουμε την εφαρμογή μας.

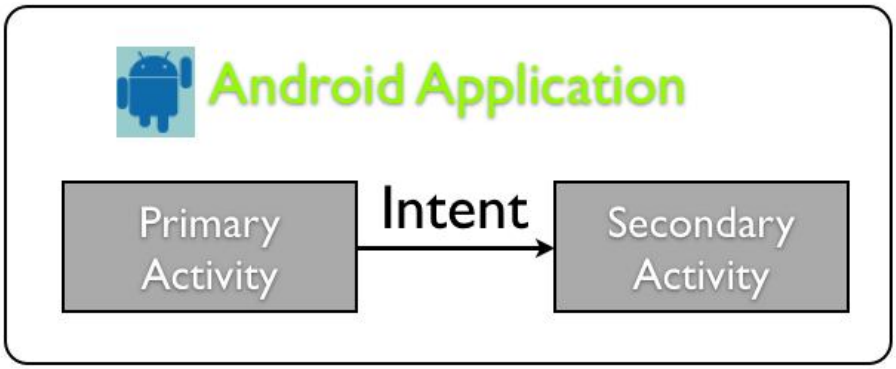
```
<intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
```

3.23. Android Layouts

Το layout είναι μια διάταξη που ορίζει την οπτική δομή για ένα user interface (περιβάλλον διεπαφής χρήστη), όπως το περιβάλλον για μια δραστηριότητα μιας εφαρμογής. Για να δηλώσουμε ένα UI χρησιμοποιούμε αρχεία και γλώσσα προγραμματισμού XML. Μεσα σε αυτά τα XML αρχεία μπορούμε να δηλώσουμε τα λεγόμενα Views (όψεις) και Viewgroups. Μέσα σε αυτά τα Views προσθέτουμε αργότερα διάφορα αντικείμενα εμφάνισης με τις αντίστοιχες ιδιότητες τους. Ας δούμε τι είδουν όψεις μπορούμε να χρησιμοποιήσουμε για το σχεδιασμό ενός Android User Interface:

- **LinearLayout**: Τοποθετεί τις όψεις σε γραμμές ή στήλες ανάλογα με το επιλεγμένο orientation. Μπορούμε να θέσουμε βάρος (weight) για την απόσταση που θα έχει κάθε γονικό στοιχείο με τα υπόλοιπα.
- **TableLayout**: Τοποθετεί τις όψεις για κάθε στοιχείο σε μια μορφή πλέγματος γραμμών και στηλών. Κάθε γραμμή σε ένα πίνακα περιέχει ένα TableRow, το οποίο περιέχει μέσα του άλλες όψεις – αντικείμενα που θα ορίσουμε για το εκάστοτε κελί.
- **FrameLayout**: Ο σκοπός του FrameLayout είναι να δεσμεύσει μια περιοχή της οθόνης για να εμφανίσει κυρίως μια ενιαία όψη (view). Αν προσθέσουμε όψεις-παιδιά αυτά θα εμφανίζονται από προεπιλογή το ένα πάνω από το άλλο. Αλλιώς μπορούμε να ορίσουμε την επιλογή του gravity που μεταφέρει το layout εκεί που θέλουμε δηλαδή το android:gravity="center_vertical" θα μεταφέρει τη διάταξη κάθετα στο κέντρο του layout.
- **RelativeLayout**: Η διάταξη αυτή είναι πιο ευέλικτη από τις άλλες γιατί επιτρέπει στις όψεις-παιδιά να τοποθετούνται σε αποστάσεις σχετικές μεταξύ τους και με το parent view. Έτσι μπορούμε να ευθυγραμμίσουμε δύο στοιχεία δεξιά ή να βάλουμε το ένα κάτω από το άλλο, στο κέντρο της οθόνης, στο κέντρο αριστερά και ούτω καθεξής. Μερικές σημαντικές παραμέτροι είναι:
  android: layout_alignParentTop
  Αν είναι "true", καθιστά το άνω άκρο αυτής της όψης να ταιριάζει με το άνω άκρο της μητρικής όψης (parent view).
  android: layout_centerVertical
  Αν είναι " true ", κεντράρει την όψη κάθετα εντός της μητρικής της όψης.
- **GridLayout**: Το GridLayout είναι μια σχετικά νέα διάταξη που εισήχθη στην έκδοση Android 4.0. Μια όψη με GridLayout χωρίζεται από αόρατες γραμμές, διαιρώντας τα μέλη της σε ένα πλέγμα που περιέχει σειρές και στήλες. Μπορούμε να ρυθμίσουμε τα κελιά να εμφανίζονται κάθετα ή οριζόντια επιτρέπωντας έτσι ένα ευρύ φάσμα επιλογών και ευκολία στη σχεδίαση για μια όψη που παλαιότερα θα φάνταζε πιο δύσκολη να υλοποιηθεί.

3.24. Android Intents



Εικόνα 1.42: Ένα Intent που συνδέει δύο δραστηριότητες μεταξύ τους

Τα Intents είναι ασύγχρονα μηνύματα που επιτρέπουν τα στοιχεία της εφαρμογής να ζητήσουν τη λειτουργικότητα από άλλα Android components. Σημαντικότερη χρήση των Intents είναι κατά την έναρξη των δραστηριοτήτων, όπου μπορεί να θεωρηθεί ως ο συνδετικός κρίκος μεταξύ των δραστηριοτήτων μιας εφαρμογής **[εικόνα 1.42]** .

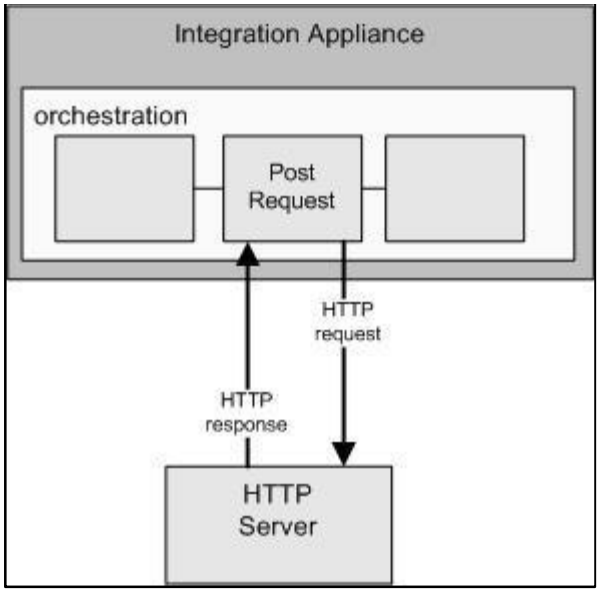
Τα Intents είναι αντικείμενα του τύπου android.content.Intent. Μέσω της μεθόδου startActivity(), ορίζουμε ότι το Intent θα πρέπει να χρησιμοποιηθεί για να μεταβούμε σε μια άλλη δραστηριότητα. Ακόμα, ένα Intent μπορεί να περιέχει δεδομένα μέσω ενός Bundle, μεταβλητές-παραμέτρους δηλαδή που μπορούμε να τις περάσουμε από Activity σε Activity.

3.25. SessionManager και SharedPreferences

Το SessionManager λειτουργεί για να αποθηκεύσουμε global μεταβλητές στην εφαρμογή μας, δηλαδή μεταβλητές που μπορούμε να τις πάρουμε μέσα από οποιαδήποτε κλάση-δραστηριότητα βρισκόμαστε

Τα Shared Preferences είναι ένας τύπος αποθήκευσης μεταβλητών που μπορεί το Android λειτουργικό να κρατήσει αφού κλείσουμε την εφαρμογή (logout). Χρησιμεύει κυρίως στην αποθήκευση του ονόματος χρήστη και του κωδικού για να μην χρειάζεται ο χρήστης να τα ξαναεισάγει κάθε φορά που μπαίνει στην εφαρμογή. Στην εφαρμογή μας το χρησιμοποιήσαμε όταν γίνεται το login του χρήστη σαν checkbox και αποθηκεύουμε το όνομα χρήστη (username) και κωδικό χρήστη(password) όταν ο χρήστης το επιλέξει.

3.26. HttpRequest και οι POST-GET μεθόδοι

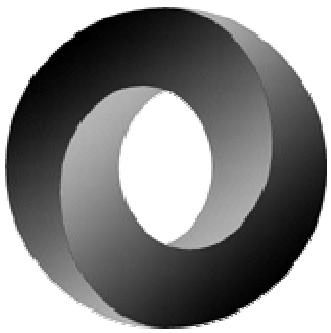


Εικόνα 1.43: Επικοινωνία με τον webserver μέσω HttpRequest

Το HttpRequest χρησιμοποιείται για να εκτελέσει μια αίτηση τύπου HTTP σε έναν (web) server ανοίγοντας έτσι ένα κανάλι επικοινωνίας **[εικόνα 1.43]** . Μας χρησίμευσε κάθε φορά που θέλουμε να στείλουμε και να πάρουμε δεδομένα από την βάση δεδομένων που έχουμε στον web server της εφαρμογής.

Χρησιμοποιήσαμε POST μέθοδο για την προώθηση την των δεδομένων στο webserver. Η POST μέθοδος έχει κάποια βασικά πλεονέκτημα έναντι της GET μεθόδου. Μερικά απ'αυτά είναι ότι η POST μέθοδο δεν έχει όριο μέγιστων χαρακτήρων, και τα δεδομένα δεν φαίνονται στο URL, προσθέτωντας έτσι περισσότερη ασφάλεια για την επικοινωνία με τη βάση δεδομένων.

3.27. Το πρότυπο JSON



Εικόνα 1.44: Το λογότυπο JSON

Ο πρώτος που έμελλε να προσδιορίσει και να διαδώσει το JSON ήταν ο Douglas Crockford τον απρίλιο του 2001 και πρωτοχρησιμοποιήθηκε στην εταιρία του όταν ήθελε να δημιουργήσει μια αμφίδρομη σύνδεση μεταξύ δύο HTTP συνδέσεων. Σε ένα πρότζεκτ για το Cartoon Network δημιούργησαν ένα plug-in που μπορούσε να διαχειριστεί DHTML στοιχεία,

μαζί με αρχικές AJAX λειτουργίες, περνώντας πληροφορίες στην εφαρμογή χωρίς ανανέωση των περιεχομένων της σελίδας, χρησιμοποιώντας τεχνολογίες HTTP, HTML και Javascript σε παλιές εκδόσεις browsers όπως ο Netscape 4.0 και Internet Explorer 5. Ο Douglas Crockford βρήκε τότε ότι η Javascript μπορεί να χρησιμοποιηθεί για μετάδοση πληροφοριών με μορφή αντικειμένων-κειμένου στην γλώσσα αυτή.

Αργότερα το σύστημα αυτό πουλήθηκε στην εταιρία Sun Microsystems και Amazon.com. Μετά το 2005 η Yahoo! άρχισε να παρέχει διαδικτυακές υπηρεσίες που βασιζόταν στο JSON και μετά ακολούθησε η Google το Δεκέμβριο του 2006 που άρχισε να το χρησιμοποιεί για το GData διαδικτυακό προτόκολλο. Τώρα πια, το JSON χρησιμοποιείται από πάρα πολλές εταιρίες πληροφορικής σε όλο το κόσμο.

3.27.1. Χαρακτηριστικά του JSON

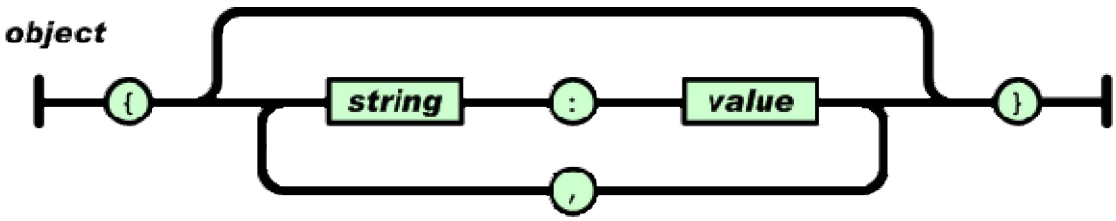
Το JSON το οποίο είναι συντομογραφία του όρου Javascript Object Notation είναι στην ουσία ένα ανοιχτό πρότυποπου χρησιμοποιεί εύκολο και αναγνώσιμο κείμενο για μετάδοση αντικειμένων δεδομένων που αποτελείται από ζεύγη χαρακτηριστικών-τιμών. Χρησιμοποιείται κυρίως για τη μετάδοση δεδομένων μεταξύ εφαρμογών και webservices και πολλές φορές έχει χαρακτηριστεί μια εναλλακτική λύση για την XML γλώσσα προγραμματισμού. Μάλιστα, προτείνεται να χρησιμοποιούμε JSON αντί XML,όπου μπορούμε, γιατί:

- Το JSON είναι πιο «ελαφρύ» σε δεδομένα και κώδικα.
- Το JSON δεν περιέχει συγκεκριμένες-προκαταχωρημένες λέξεις, κάνοντας το πιο ανθρώπινα προσιτό.
- Το JSON έχει πίνακες

Παρά το γεγονός ότι το όνομα του JSON περιέχει τη λέξη Javascript, δεν είναι απαραίτητα σχετικό με τη γλώσσα αυτή. Το JSON δηλαδή μπορεί να χρησιμοποιηθεί σε ένα μεγάλο εύρος γλωσσών προγραμματισμού.

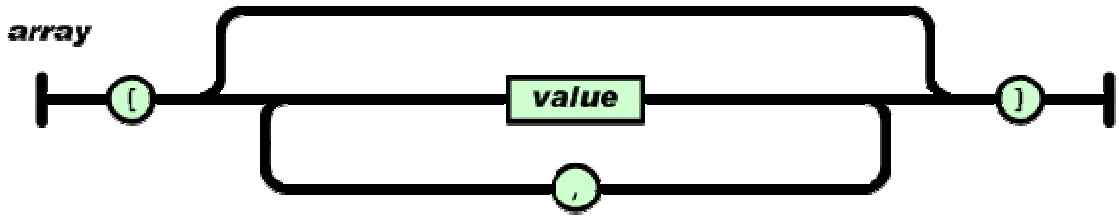
Στο ζεύγος όνομα-τιμή το όνομα πρέπει πάντα να είναι τύπου stringενώ η τιμή (value) μπορεί να είναι τύπου: string, Boolean, αριθμός, αντικείμενο, πίνακας, άδεια τιμή(null).

Ένα αντικείμενο τύπου JSON είναι μια σειρά από ζεύγη ονομάτων-τιμών. Το αντικείμενο ξεκινάει με ένα αριστερό υποστήριγμα (brace) { και τελειώνει με ένα δεξί υποστήριγμα } . Κάθε όνομα στο ζεύγος ακολουθείται από μία άνω κάτω τελεία : και τα ζεύγη ονομάτων τιμών χωρίζονται με , (κόμμα).



Εικόνα 1.45: Μορφή ενός αντικειμένου τύπου JSON

Ένας πίνακας τύπου JSON είναι μια συλλογή τιμών. Ο πίνακας αυτός ξεκινάει με μια αριστερή αγκύλη και τελειώνει με μια δεξιά αγκύλη. Οι τιμές χωρίζονται με , (κόμμα).



Εικόνα 1.46: Επεξηγηματική μορφή ενός πίνακα τύπου JSON

4. Σχέδιο Δράσης για την επόνηση της πτυχιακής εργασίας

4.1.State of the Art

Οι διαδικτυακές αγορές έχουν αρχίσει να κάνουν αισθητή την παρουσία τους καθώς όλο και περισσότεροι άνθρωποι και χρήστες της τεχνολογίας κάνουν αγορές από υπολογιστές και κινητές τηλεφωνικές συσκευές (smart phones).

Ένα smartphone με κινητό πορτοφόλι μπορεί να βοηθήσει με το να πληρώσει ο πελάτης-χρήστης για πράγματα που άλλοτε δεν θα μπορούσε, όπως χρέωση των εισιτηρίων για συναυλίες, εισιτήρια λεωφορείων, για το μετρό ή ακόμα και σε ένα σουπερμάρκετ, το οποίο πλησιάζει πιο κοντά στην εφαρμοφή που υλοποιήσαμε.

Πιο συγκεκριμένα, το μοντέλο του digital wallet δηλαδή ηλεκτρονικό πορτοφόλι έχει αναπτυχθεί ευρέως τα τελευταία χρόνια ιδίως στην Ιαπωνία. Η εφαρμογή μας βασίστηκε σε αυτό το μοντέλο του digital wallet και πιο συγκεκριμένα σε server-side digital wallet μιας και η βάση δεδομένων διατηρήθηκε σε online server- domain και δεν αποθηκεύονται οι (βασικές) πληροφορίες στη συσκευή που χρησιμοποιεί ο εκάστοτε χρήστης.

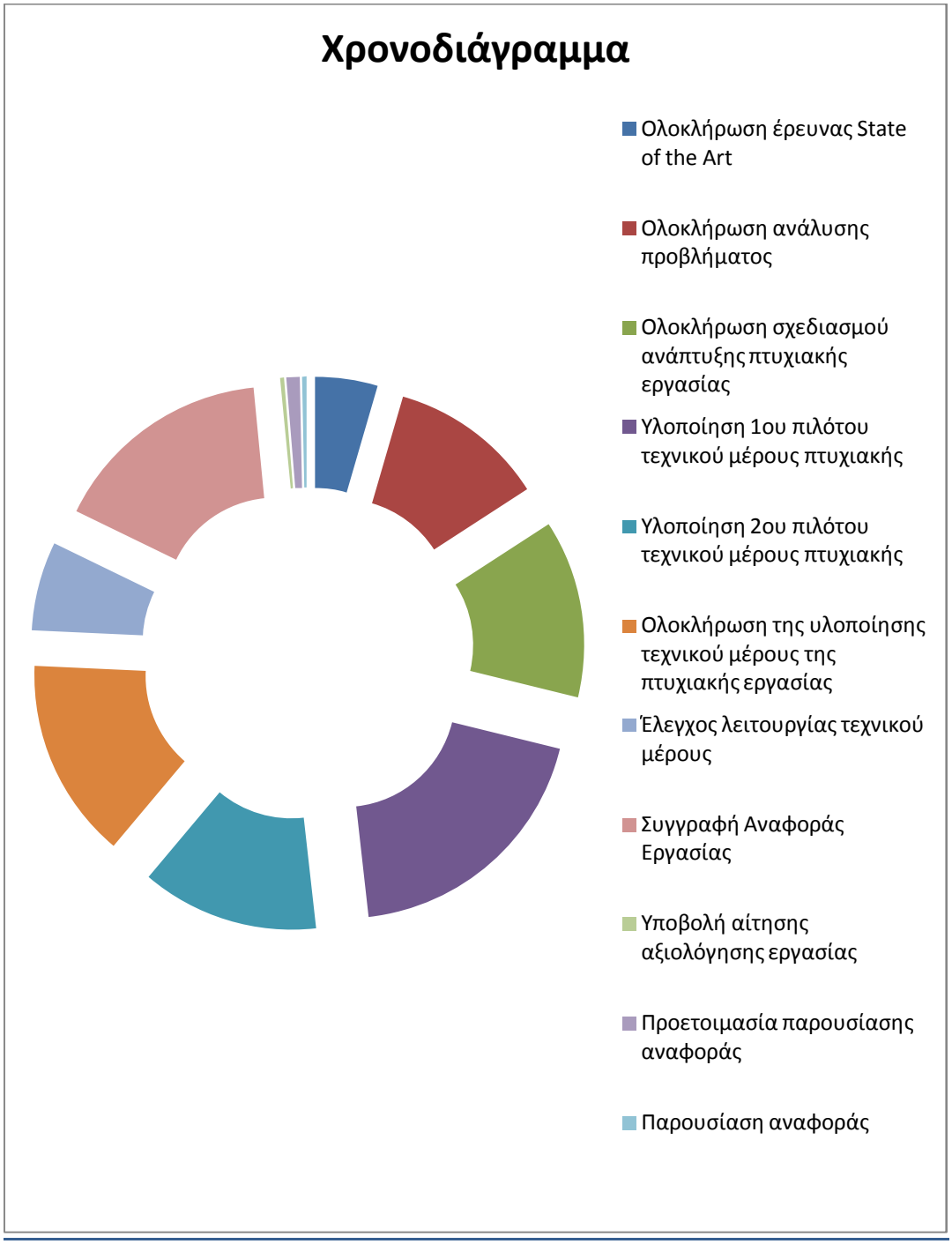
Η Google, η Apple και άλλες εταιρίες παγκόσμιου βεληνεκούς έχουν αναπτύξει εφαρμογές ψηφιακού πορτοφολιού οι οποίες παρέχουν πολλές υπηρεσίες και καταχωρούν δεδομένα των χρηστών στους αντίστοιχους server τους.

Το μοντέλο όμως του ψηφιακού πορτοφολιού δεν θα είχε σωστή βάση αν δεν στηριζόταν σε μια ασφαλή μεταφορά των χρημάτων από τον πελάτη στο κατάστημα. Γι'αυτό και στην εφαρμογή μας χρησιμοποιήσαμε μια από τις ασφαλότερες τεχνολογίες στο διαδικτυακό εμπόριο, το PayPal. Με την χρήση του PayPal οι συναλλαγές έχουν αποδεδειγμένα πάρα πολύ υψηλή ασφάλεια στους κρυπτογραφημένους αλγόριθμους που βασίζεται κάνοντας το ένα ιδανικό εργαλείο για την δημιουργία μιας εφαρμογής που βασίζεται σε συναλλαγές χρημάτων.

Στην εφαρμογή μας συνδέουμε το χρήστη με το PayPal όπου χρησιμοποιεί τον λογαριασμό που έχει στην υπηρεσία , δηλαδή βάζοντας την διεύθυνση ηλεκτρονικού ταχυδρομείου και τον κωδικό του και βλέποντας τα προϊόντα που έχει αγοράσει από το κατάστημα, πατώντας απλά ένα κουμπί επιβεβαίωσης η παραγγελία του ολοκληρώνεται.

4.2.Χρονοδιάγραμμα Πτυχιακής Εργασίας

Παρακάτω δείχνουμε ένα γράφημα με τα στάδια υλοποίησης της πτυχιακής μας εργασίας σε σχέση με τον χρόνο που δαπανήθηκε για κάθε στάδιο.



Εικόνα 1.47: Το σχεδιάγραμμα της πτυχιακής εργασίας

4.3.Περιβάλλον Σχεδίασης - Λογισμικό

Χρησιμοποιήσαμε λειτουργικό σύστημα Windows 7 Professional Edition 64 bit και οι συσκευές που χρησιμοποιήσαμε για την υλοποίηση αλλά και τις δοκιμές της εφαρμογής μας ήταν:

- Ηλεκτρονικός Υπολογιστής με χρήση Android εξομοιωτή
- Δύο Android smartphone συσκεύες για καλύτερο έλεγχο συμβατότητας.

Το λογισμικό που χρησιμοποιήσαμε:

- MySQL
- PhpMyAdmin
- Wamp Server x64 version 2.2
- Eclipse 3.7.2
- GitBash (για τη συνεργασία στον κώδικα μας)
- Aptana Studio 3, build: 3.2.2.201208201020 (για τον PHP κώδικα)

Οι γλώσσες προγραμματισμού που χρειαστήκαμε για την υλοποίηση της εφαρμογής ήταν:

- Java (Android)
- PHP
- SQL

4.4.Eclipse IDE

Εικόνα 1.48: Το λογότυπο του Eclipse IDE

Το Eclipse είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) προγραμματισμού. Περιέχει μια βάση εργασίας (workspace) και ένα σύστημα για επεκτασιμότητα (plugin-ins). Με το Eclipse μπορούμε να αναπτύξουμε εφαρμογές σε γλώσσες προγραμματισμού όπως : ADA, C, C++, COBOL, Fortran, Haskell, JavaScript, Lasso, Perl, PHP, Python, R, Ruby, Scala, Clojure, Groovy και Erlang. Το Eclipse SDK (Software Development Kit) περιέχει εργαλεία ανάπτυξης σε γλώσσα Java.

Ακόμα, το Eclipse SDK είναι δωρεάν και λογισμικό ανοιχτού κώδικα το οποίο ξεκίνησε σαν ένα πρότζεκτ της εταιρίας IBM. Με την εξαίρεση ενός μικρού runtime πυρήνα, τα πάντα στο Eclipse είναι ένα plug-in. Το Eclipse παρέχει plug-ins για μια ευρεία ποικιλία χαρακτηριστικών όπως για UML διαγράμματα, plug-in για τον DB Explorer και πολλά άλλα.

4.5.Wamp Server

Εικόνα 1.49: WampServer

Το Wampserver είναι ένα περιβάλλον για ανάπτυξη διαδικτυακών εφαρμογών. Μας βοηθάει στο να δημιουργήσουμε το δικό μας σέρβερ στο τοπικό μας δίκτυο με τη χρήση Apache2, PHP και βάσεων δεδομένων MySQL, Phpmyadmin, SQLBuddy. Πιο αναλυτικά το wampserver παρέχει:

- Διαχείρηση Apache και MySQL υπηρεσιών
- Διαχείρηση ρυθμίσεων εικονικού σέρβερ.
- Πρόσβαση σε logs.
- Εύκολη αναβάθμιση σε καινούργιες εκδόσεις για Apache & MySQL

4.6.MySQL

Εικόνα 1.50: Το λογότυπο της MySQL

Η MySQL είναι ένα -ευρέως διαδεδομένο ανοιχτού κώδικα- σύστημα διαχείρησης σχεσιακών βάσεων δεδομένων. Πολλές διαδικτυακές εφαρμογές χρησιμοποιούν τη MySQL, όπως η Wikipedia και η Google. Τα δεδομένα στη MySQL αποθηκεύονται σε πίνακες. Κάθε πίνακας είναι μια συλλογή από δεδομένα σχετικά μεταξύ τους που περιέχονται στις στήλες και τις γραμμές του. Συνήθως η MySQL χρησιμοποιείται για να

δεχτεί ερωτήματα (queries) βάσεων δεδομένων και να επιστρέψει με ταχύτητα τα αποτελέσματα.

Μερικές συχνές συναρτήσεις PHP που χρησιμοποιήσαμε για να πέρνουμε δεδομένα από την MySQL βάση δεδομένων είναι:

- mysql_query() : εκτελεί ένα ερώτημα (query) σε μια MySQL βάση δεδομένων.
- mysql_fetch_array() : μας επιστρέφει έναν πίνακα (με τις στήλες και τις τιμές τους) για κάθε γραμμή που βρίσκεται στο sql query. Επιστρέφει FALSE όταν τελειώσουν οι γραμμές του query.
- mysql_num_rows : επιστρέφει τον αριθμό των εγγραφών σε ένα sql query.
- mysql_data_seek() : μετακινεί το δείκτη σε μια γραμμή, δηλαδή μετακινεί την τρέχουσα θέση σειράς για ένα αποτέλεσμα που επιστρέφεται από την mysql_query()

4.7.Aptana Studio

Εικόνα 1.51: Aptana Studio

Το Aptana είναι ένα εργαλείο ανάπτυξη διαδικτυακών εφαρμογών, είναι τύπου ανοικτού κώδικα και υποστηρίζει γλώσσες προγραμματισμού όπως HTML5, CSS3, JavaScript, Ruby, Rails, PHP και Python. Επίσης υποστηρίζει FTP, SFTP προτόκολλα μεταφοράς αρχείων. Στην εφαρμογή μας το χρησιμοποιήσαμε για τον PHP κώδικα.

4.8.Git

Εικόνα 1.52: Το λογότυπο του Git Λογισμικού

Ένα από τα βασικά χαρακτηριστικά της ανάπτυξης λογισμικού είναι ότι γράφουμε τον πηγαίο κώδικα σε στάδια. Ξεκινούμε αρχικά με απλό κώδικα και δοκιμάζοντας διάφορα, προσθέτουμε, διορθώνουμε και γράφουμε περισσότερο κώδικα. Ιδίως όταν μαθαίνουμε προγραμματισμό, νιώθουμε την ανάγκη να έχουμε ένα σύστημα που να καταγράφει τις διαδοχικές εκδόσεις του κώδικά μας. Και όταν το πρόγραμμα μας γίνει μεγαλύτερο, ή όταν συνεργαζόμαστε με φίλους προγραμματιστές, τότε είναι επιτακτικό να έχουμε ένα σύστημα για τη διαχείριση του κώδικά μας.
Repository: Το φυσικό σημείο στο οποίο βρίσκονται αποθηκευμένα τα αρχεία μαζί με όλες τις πληροφορίες του RCS (local ή remote).
Commit: Ένα σύνολο λογικά συσχετιζόμενων αλλαγών στα παρακολουθούμενα αρχεία (edit, add, delete) μαζί με πληροφορίες για το ποιός τις έκανε,πότε και γιατί.
Branch: Όταν υπάρχει ανάγκη για απόσπαση από το "κεντρικό σώμα" του κώδικα (π.χ. για ένα experimental feature ή bug fix), ένα branch κρατάει τα νέα commits χωρίς να επηρεάζει το σώμα του κώδικα.
Tag: Δείκτης σε μια συγκεκριμένη έκδοση του κώδικα.
Merge: Η διαδικασία ενσωμάτωσης ενός branch σε ένα άλλο.
Κάποια από τα χαρακτηριστικά του Git είναι:

- Εύκολο branching
- Εύκολο merging
- Έλεγχος εγκυρότητας των δεδομένων (SHA1)
- Παρακολουθεί το περιεχόμενο συνολικά και όχι ανά αρχείο
- Πολύ γρήγορο.

4.9.PhpMyAdmin

Εικόνα 1.53: Λογότυπο του λογισμικού phpMyAdmin

Το PhpMyAdmin είναι ένα δωρεάν λογισμικό γραμμένο σε γλώσσα προγραμματισμού PHP που έχει ως σκοπό να διαχειριστεί τις MySQL βάσεις δεδομένων που έχουμε στον δίσκο μας είτε τοπικά είτε σε απομακρυσμένο σέρβερ στο ίντερνετ.

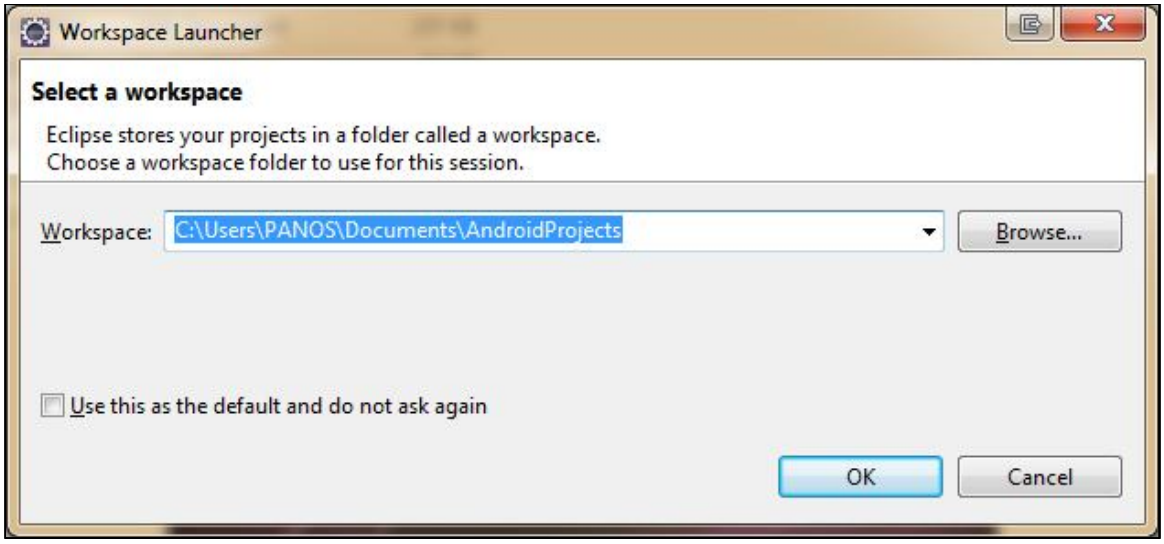
Μερικές από τις βασικές λειτουργίες του phpMyAdmin είναι:

- Πλήρης διαχείριση μια βάσης δεδομένων.
- Περιήγηση και διαγραφή βάσεων δεδομένων, πινάκων, όψεων και στηλών μέσα σε αυτές.
- Προβολή αποτελεσμάτων στη χρησιμοποίηση queries στην εκάστοτε βάση δεδομένων.
- Εισαγωγή και εξαγωγή αρχείων τύπου μοντέλου βάσης δεδομένων και αποθήκευση της στο σέρβερ μας.
- Αναζήτηση μέσα στη βάση και δημιουργία pdf αρχείου με γραφική παρουσίαση της εκάστοτε βάσης δεδομένων μας.

4.10. Έναρξη του Eclipse IDE

Το Eclipse IDE Indigo το κατεβάζουμε από την επίσημη σελίδα του Eclipse επιλέγοντας το πακέτο Eclipse IDE for Java EE Developers

Αφού το εγκαταστήσουμε το τρέχουμε από τον φάκελο ή το εικονίδιο που δημιουργείται

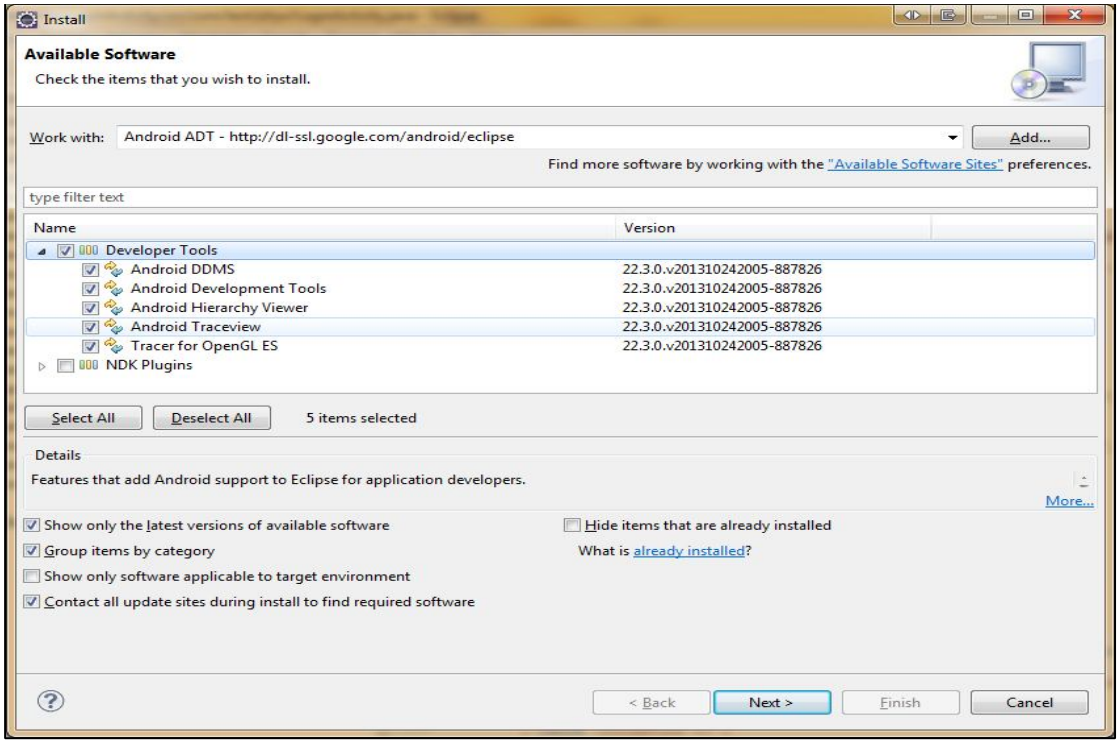


Εικόνα 1.54: Το path του φακέλου της εφαρμογής μας

Διαλέγουμε την τοποθεσία που θέλουμε να αποθηκευτεί το πρότζεκτ μας, δηλαδή η Android εφαρμογή μας και μετά πατάμε OK.

Αφού ανοίξει το πρόγραμμα Eclipse IDE, πηγαίνουμε στο Help > Install new Software, πατάμε το Add κουμπί και στα πεδία που βλέπουμε πατάμε όνομα (Name:)Android ADT και Location: https://dl-ssl.google.com/android/eclipse/

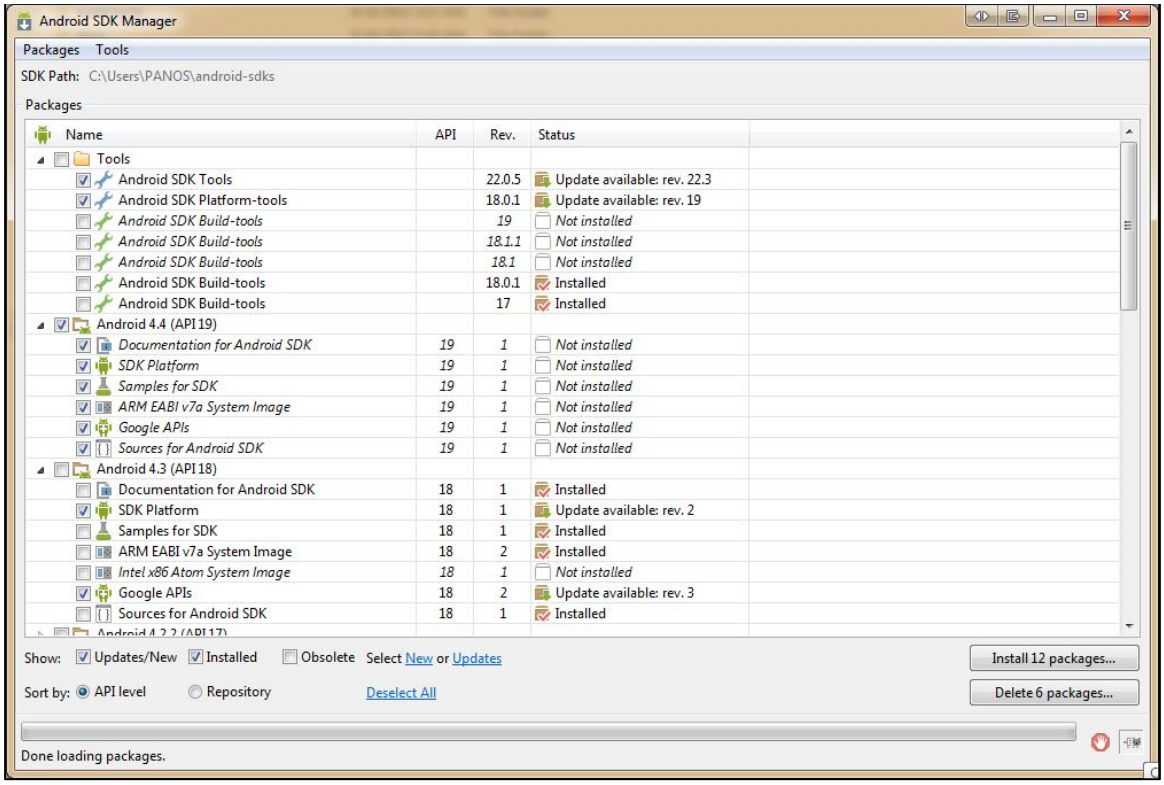
Επιλέγουμε τα developer tools και πατάμε next για να ξεκινήσει η εγκατάσταση του Android Development Tools (ADT) Plugin.



Εικόνα 1.55: Εγκατάσταση Developer Tools

4.11. Έναρξη του Android SDK

Το πρώτο βήμα είναι να κατεβάσουμε το Android Software Development Kit (SDK) που θα μας επιτρέψει να εξομοιώσουμε τη λειτουργία του Android στον υπολογιστή μας.

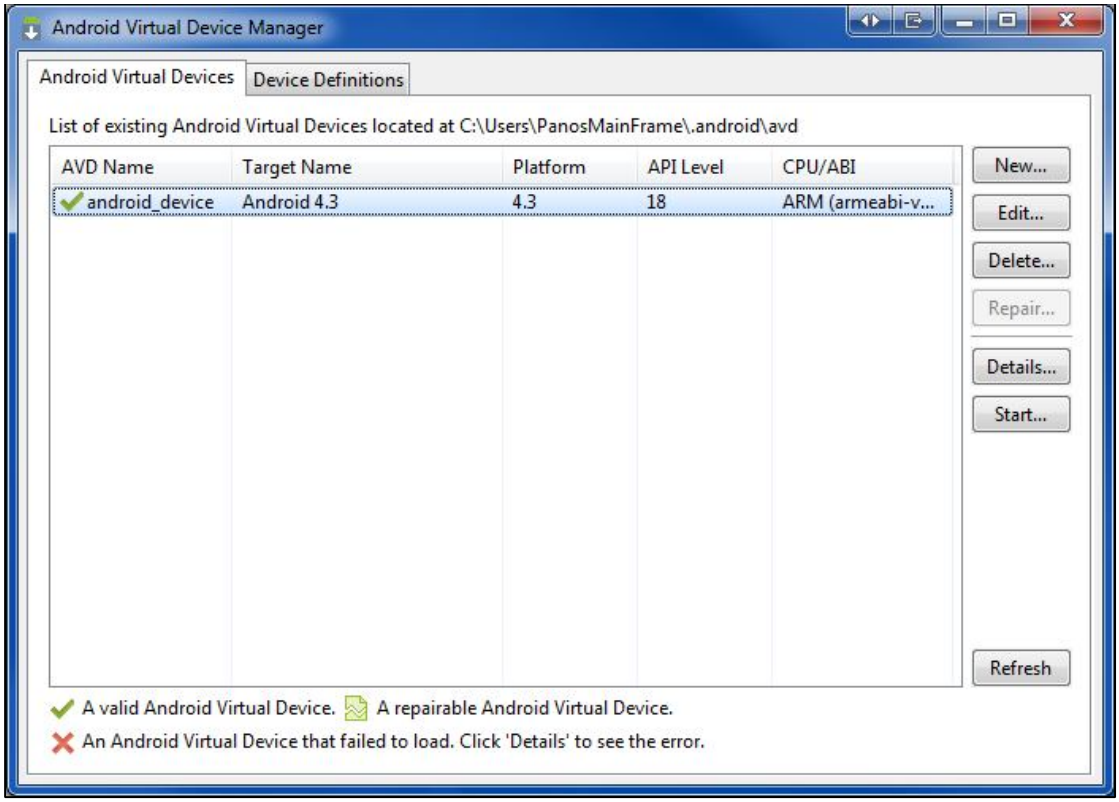


Εικόνα 1.56: Το Android SDK Manager

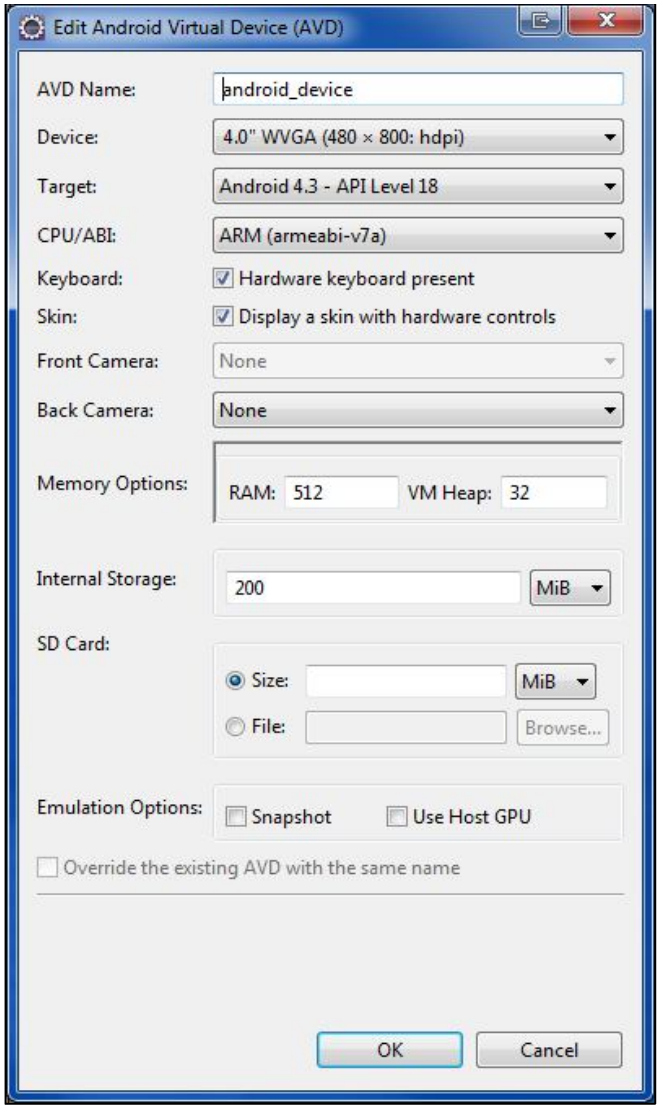
Το Android SDK Manager μας επιτρέπει να εγκαταστήσουμε πακέτα βιβλιοθήκες (API) , εργαλεία και γενικά ότι χρειάζεται ένας Android προγραμματιστής για να δημιουργήσει και να εξομοιώσει εφαρμογές στο Android.

Μέσα από τον Android SDK Manager βεβαιωνόμαστε ότι έχουμε κατεβάσει το Android (API)

4.12. Δημιουργία ενός Android Virtual Device



Εικόνα 1.57: Προσθήκη εικονικής Android συσκευής



Εικόνα 1.58: Ρυθμίσεις δημιουργίας μια εικονικής συσκευής Android

Ένα Android Virtual Device είναι ένας εξομοιωτής μιας κινητής συσκεύης Android. Για να δημιουργήσουμε μια εικονική κινητή συσκευή επιλέγουμε από την πάνω μπάρα των βασικών λειτουργιών τις επιλογές **Window > Android Virtual Device Manager** και πατάμε το κουμπί **New…** όπως φαίνεται στην εικόνα **[εικόνα 1.57]** . Αν θέλουμε να αλλάξουμε τις ρυθμίσεις της εικονικής συσκευής που δημιουργήσαμε πατάμε πάνω στην συσκευή μας και πατάμε το κουμπί **Edit...** για να επεξεργαστούμε τις επιλογές όπως φαίνεται στην εικόνα **[εικόνα 1.58]**.

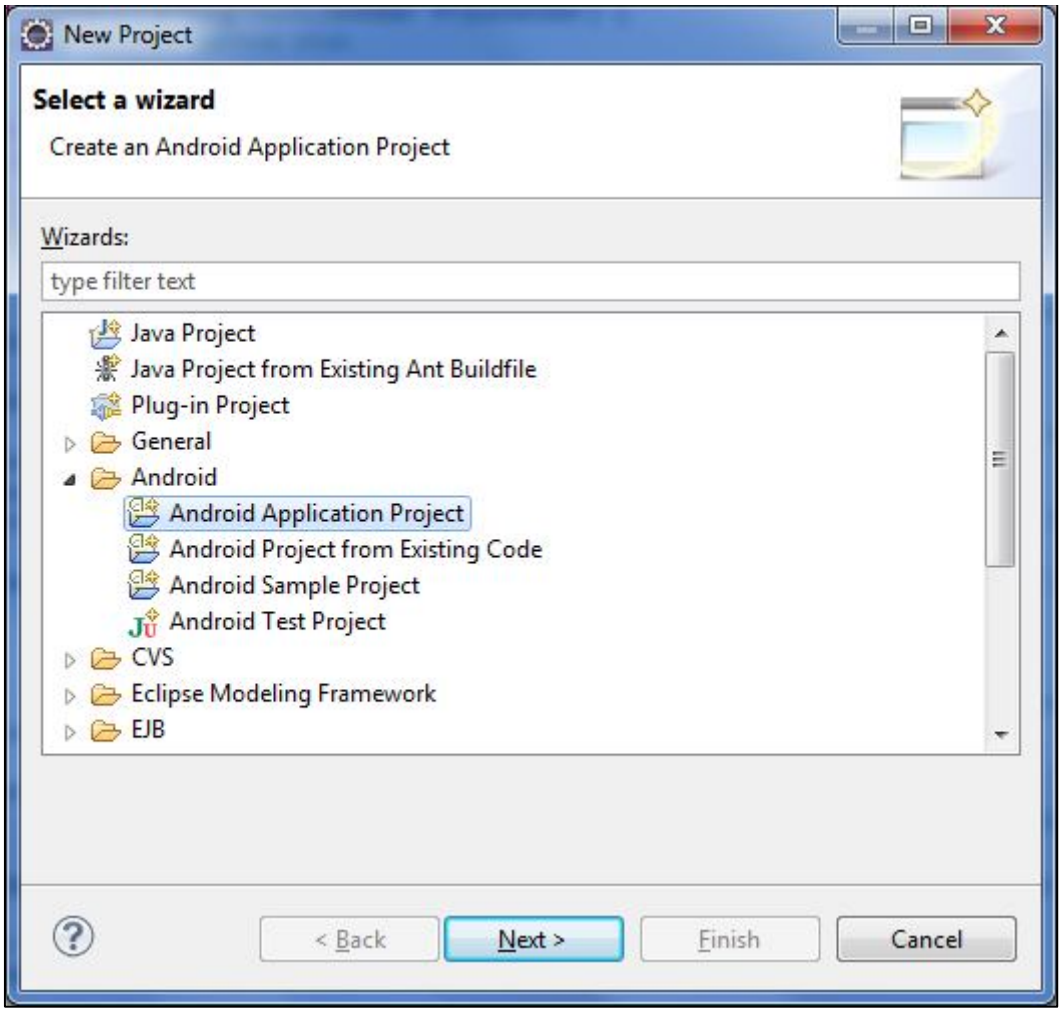
Οι ρυθμίσεις που επιλέξαμε για τη δημιουργία της εικονικής συσκευής μας είναι :

- AVD Name: android_device
- Device: 4.0” WVGA (480 x 800: hdpi)
- Target: Android 4.3 – API Level 18
- CPU/ABI: ARM (armeabi-v7a)

Τις υπόλοιπες επιλογές τις αφήσαμε στην προεπιλεγμένη μορφή τους.

4.13. Δημιουργία ενός Android Project

Πηγαίνουμε στο **New->Project** και στο παράθυρο που εμφανίζεται επιλέγουμε το φάκελο **Android->Android Application Project [εικόνα 1.59]** όπου πληκτρολογούμε το όνομα του project που θέλουμε να έχει η εφαρμογή μας.



Εικόνα 1.59: Έναρξη-δημιουργία νέου Android Project

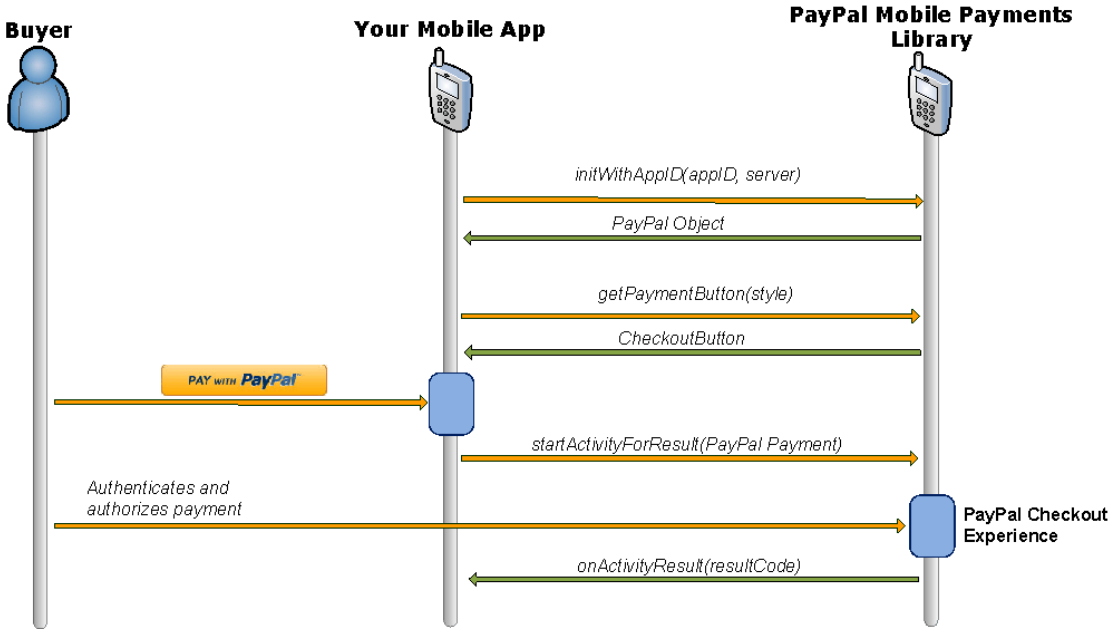
4.14. Η βιβλιοθήκη Mobile PayPal Library

Η βιβλιοθήκη Mobile PayPal Library (MPL) παρέχει μια ασφαλή και εκτενής λειτουργία στη δυνατότητα των Android (αλλά και iOS) εφαρμογών να προσθέσουν το μοντέλο πληρωμών που διαθέτει το PayPal. Με αυτή τη βιβλιοθήκη οι πελάτες-χρήστες της εφαρμογής ολοκληρώνουν τις συναλλαγές τους σε μία εφαρμογή με το πάτημα ενός κουμπιού πολύ εύκολα, χωρίς να χρειαστεί να μεταφερθούν σε άλλο παράθυρο, κάνοντας έτσι την ροή των πληρωμών πολύ προσιτή και ενιαία.

Η βιβλιοθήκη αυτή γενικά δουλεύει ως εξής:

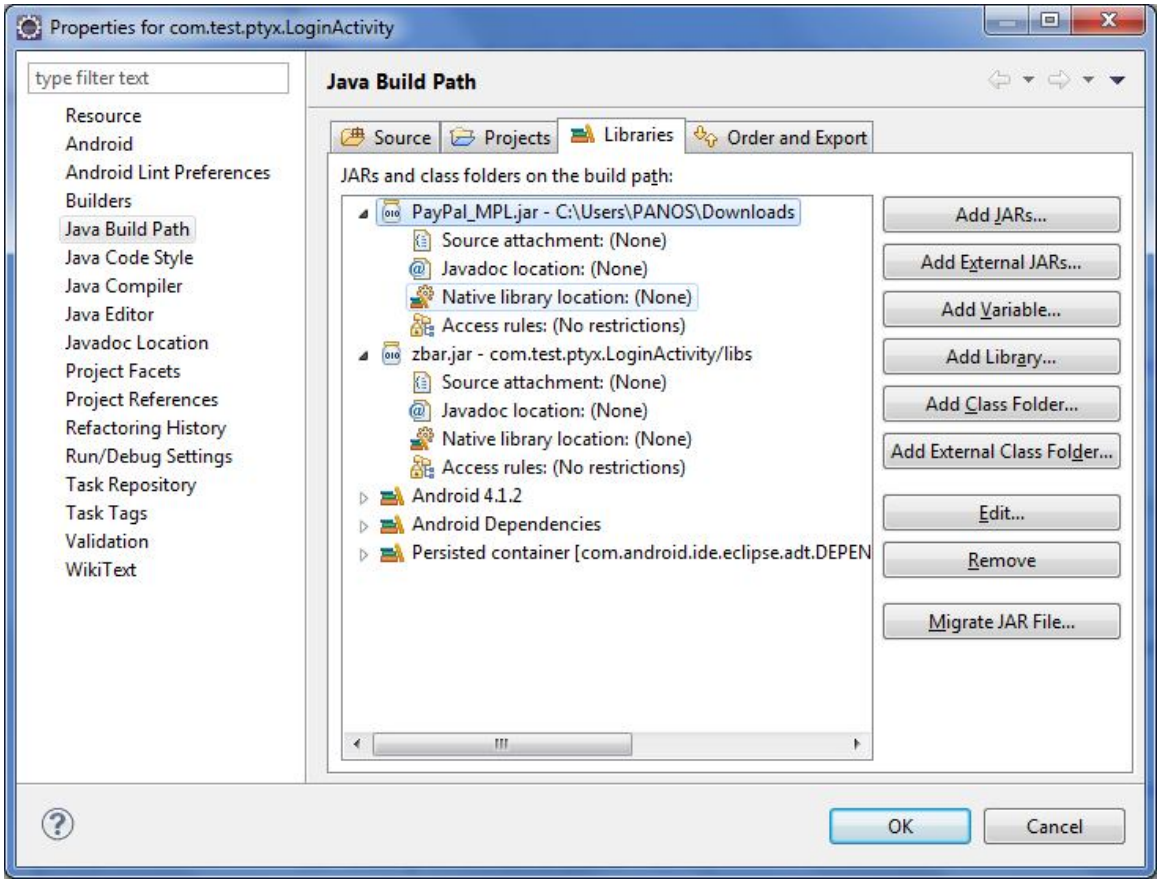
1. Η εφαρμογή μας αρχικοποιεί τη βιβλιοθήκη
2. Η βιβλιοθήκη δημιουργεί ένα κουμπί πληρωμής (Pay with PayPal Button)

3. Μέσω της εφαρμογής μας αρχικοποιούμε τις παραμέτρους όπως τα προϊόντα, την ποσότητα και την τιμή κάθε προϊόντων που επέλεξε ο χρήστης,
4. Όταν ο χρήστης πατήσει την επιλογή Pay with PayPal ξεκινάει η ροή της συναλλαγής μέσω PayPal. Εμφανίζεται δηλαδή ένα νέο παράθυρο στην εφαρμογή του χρήστη, πάνω από το όποιοδήποτε παράθυρο της εφαρμογής έχει ανοιχτό.
5. Όταν ο χρήστης ολοκληρώσει την πληρωμή του, το PayPal επιστρέφει ένα μοναδικό αριθμό συναλλαγής και διάφορα άλλα στοιχεία για την πληρωμή όπως αν πραγματοποιήθηκε επιτυχώς, ημερομηνία πληρωμής, email παραλήπτη και άλλα.
6. Το PayPal επιστρέφει τον έλεγχο στην εφαρμογή μας όπου εμείς με τα στοιχεία που μας δίνει το PayPal κάνουμε τις ανάλογες ενέργειες στην εφαρμογή μας.



Εικόνα 1.60: Σχεδιάγραμμα της βιβλιοθήκης Mobile Paypal Library για Android

4.15. Προσθήκη βιβλιοθήκης Mobile PayPal Library



Εικόνα 1.61: Προσθήκη βιβλιοθήκης Mobile PayPal Library

Για να εγκαταστήσουμε την βιβλιοθήκη Mobile PayPal Library, αφού έχουμε ανοιχτό το Eclipse, πατάμε δεξί κλικ στο όνομα του project της εφαρμογής μας, **Properties -> Java Build Path->Add external JARs** και επιλέγουμε από τον υπολογιστή μας το αρχείο της βιβλιοθήκης που έχουμε κατεβάσει. **[Εικόνα 1.61]**

4.16. Η βιβλιοθήκη Zbar για barcode scanning

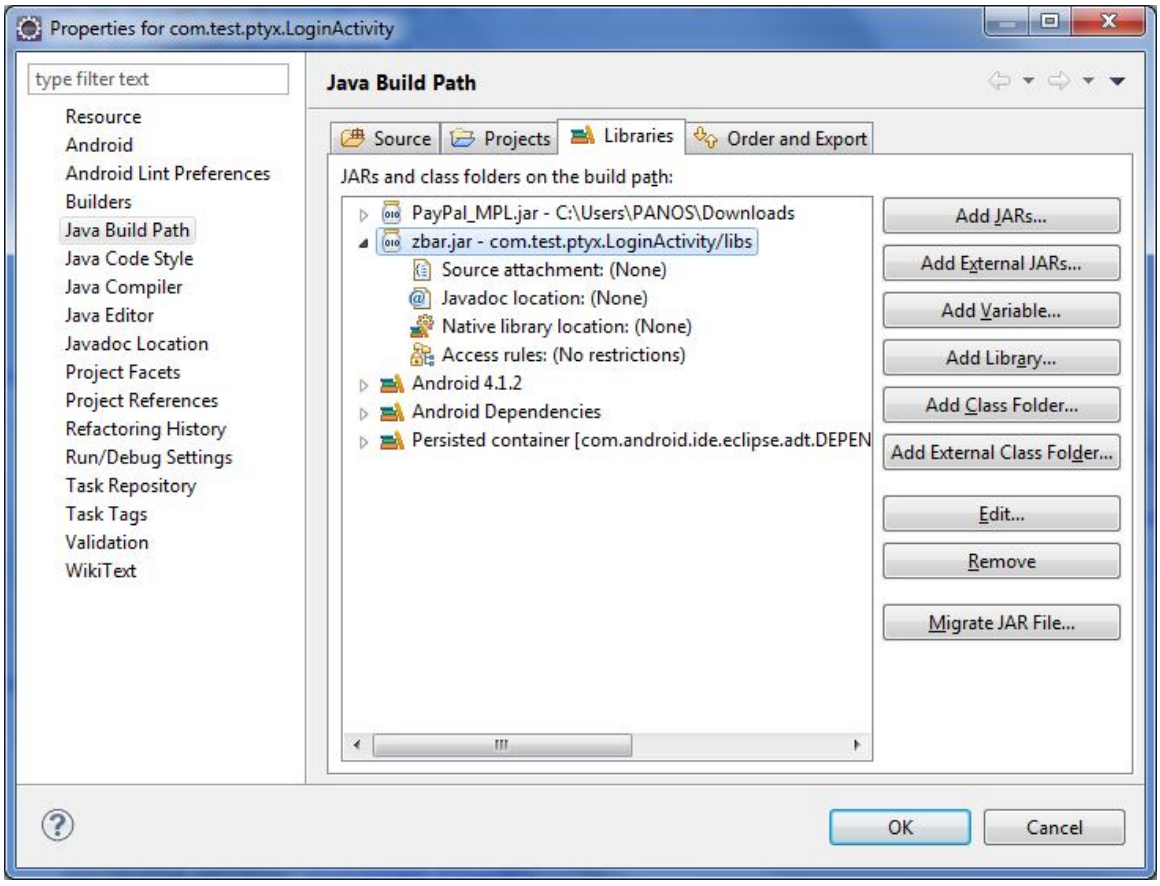
Εικόνα: ΒιβλιοθήκηZBar

Η βιβλιοθήκη ZBar είναι μια βιβλιοθήκη ανοιχτού κώδικα που χρησιμεύει στην ανάγνωση barcodes και υποστηρίζει διάφορες πηγές όπως ροές βίντεο, αρχεία εικόνων και αισθητήρες. Η ZBar υποστηρίζει πολλές συμβολογίες δηλαδή τύπους Barcode, όπως EAN-13/UPC-A, UPC-E, EAN-8, Code 128, Code 39 και QR Code. Μερικοί από τους λόγους που χρησιμοποιήσαμε την ZBar βιβλιοθήκη είναι:

1. Παρέχει υψηλή ταχύτητα ανάγνωσης.
2. Υποστηρίζει ανάγνωση από βίντεο-κάμερα
3. Ο κώδικας είναι εύκολος και λίγος αλλά παραμετροποιήσιμος
4. Δεν καταναλώνει πολλούς πόρους στο λειτουργικό σύστημα.

Η ZBar βιβλιοθήκη χρησιμοποιεί τεχνικές παρόμοιες με αυτές ενός barcode laser σαρωτή και έχει υλοποιηθεί έτσι ώστε τα barcodes να αποκωδικοποιούνται μέσω ενός αισθητήρος φωτός που σαρώνει τα λευκά και μαύρα σημεία ενός συμβόλου. Η ZBar βιβλιοθήκη αποκωδικοποιεί ανιχνεύοντας τα μήκη των γραμμών του εκάστοτε barcode και παράγει μια ροή από τελείως αποκωδικοποιημένα δεδομένα συμβόλων.

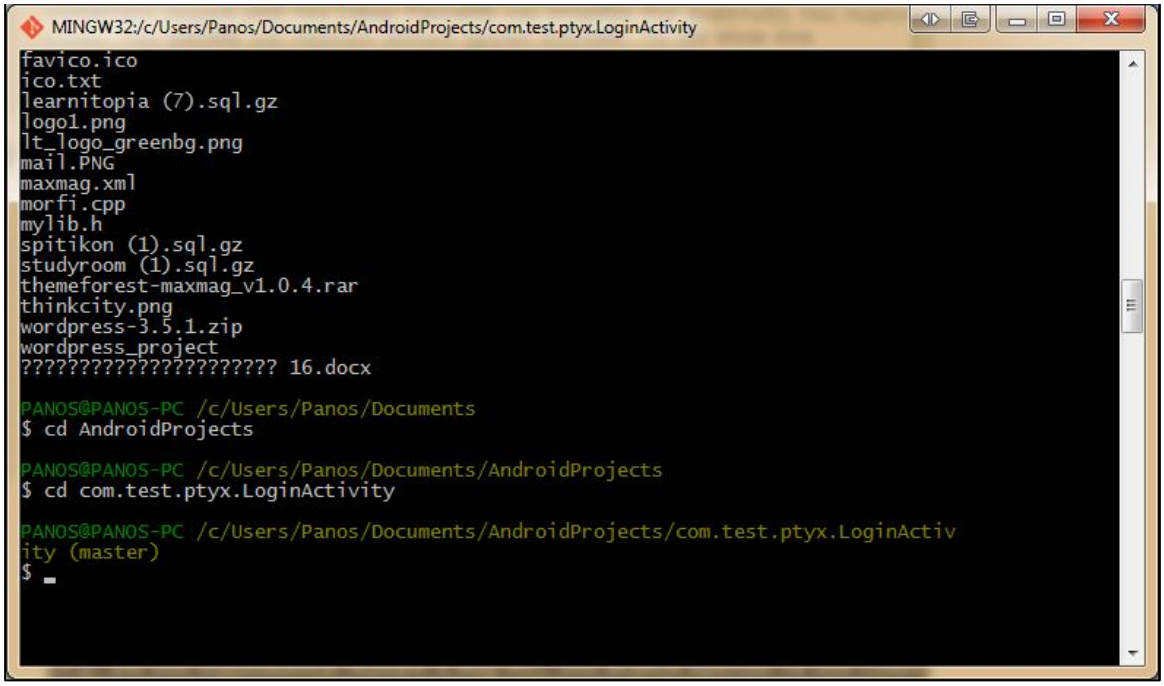
4.17. Εγκατάσταση βιβλιοθήκης ZBar



Εικόνα 1.62: Εγκατάσταση ZBar Βιβλιοθήκης

Για να εγκαταστήσουμε την βιβλιοθήκη ZBar, αφού κατεβάσουμε το αντίστοιχο jar αρχείο πατάμε δεξί κλικ στο όνομα του project της εφαρμογής μας, **Properties ->JavaBuildPath->Add external JARs** και επιλέγουμε από τον υπολογιστή μας το αρχείο της βιβλιοθήκης που έχουμε κατεβάσει.

4.18. Αρχικοποίηση Git φακέλου



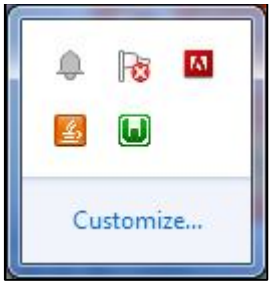
Εικόνα 1.63: Η κονσόλα του Git (GitBash)

Αφού βεβαιωθούμε ότι βρισκόμαστε στον φάκελο της εφαρμογής μας που έχει δημιουργηθεί από το Eclipse, πατώντας τα αντίστοιχα cd στην κονσόλα του git βρισκόμαστε στο φάκελο που θέλουμε και πληκτρολογούμε την εντολή git init για να εκκινήσουμε την αποθήκευση των σταδίων του πηγαίου κώδικα μας.

4.19. Εκκίνηση του Wamp Server

Αφότου έχουμε εγκαταστήσει την έκδοση του Wamp Server την ανοίγουμε κάνοντας κλικ στο αντίστοιχο εικονίδιο στην εγκατεστημένα προγράμματα του υπολογιστή μας.

Αν έχουμε κάνει τα σωστά βήματα για την εγκατάσταση του Wamp Server το εικονίδιο στην κάτω δεξιά μεριά της οθόνης μας θα πρέπει να έχει ανάψει πράσινο.



Εικόνα 1.64: Το εικονίδιο του WampServer

4.20. Εκκίνηση του phpMyAdmin

Το phpMyAdmin βρίσκεται στις επιλογές εγκατάστασης του WampServer και είναι πολύ χρήσιμο καθώς μας επιτρέπει να διαχειριστούμε την MySQL βάση δεδομένων μας. Για να μπούμε όμως στο μενού επιλογών και δημιουργίας της βάσης δεδομένων πρέπει να κάνουμε πρώτα είσοδος ως χρήστης (δηλαδή υπερχρήστης) για να έχουμε πλήρη έλεγχο της βάσης δεδομένων που θέλουμε να δημιουργήσουμε και στη συνέχεια να τροποποιήσουμε Εμείς χρησιμοποιήσαμε τον προεπιλεγμένο όνομα χρήστη και κωδικό. Δηλαδή όνομα χρήστη (username) : root και ο κωδικός χρήστη (password) κενός. **[Εικόνα 1.65]**

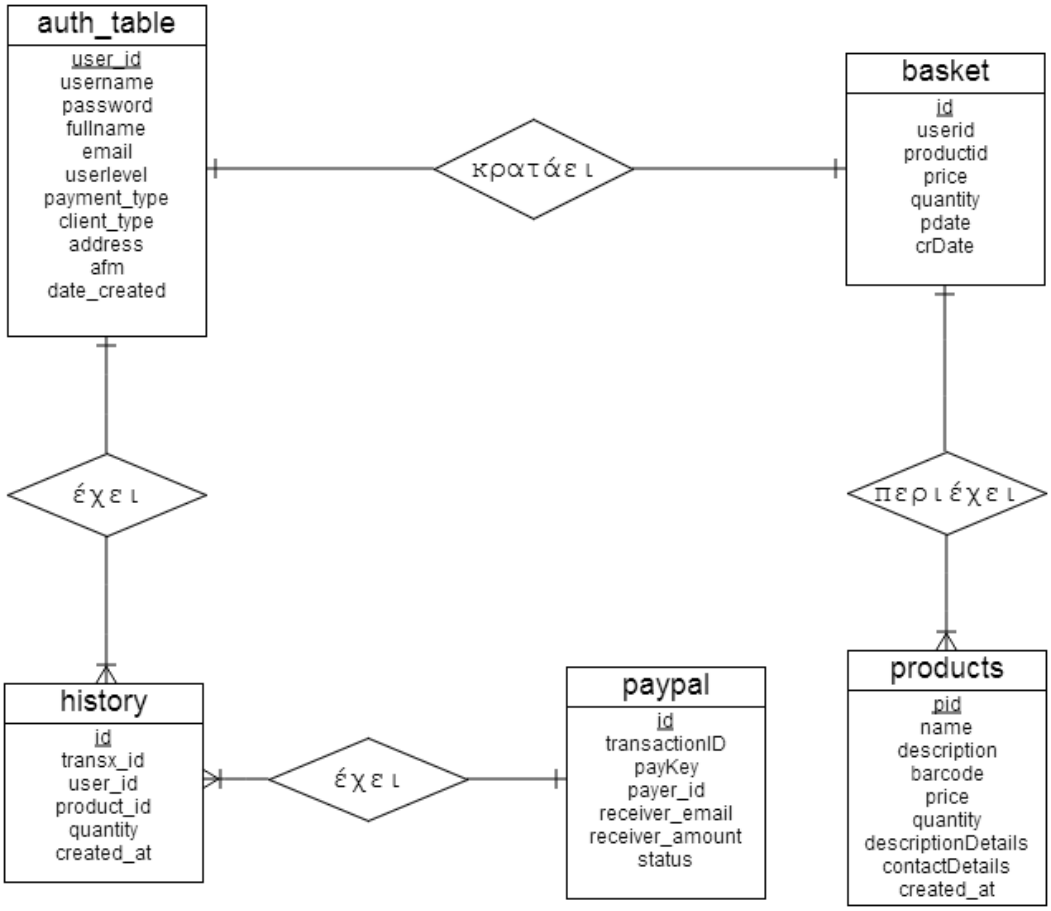
Εικόνα 1.65: Αρχική σελίδα διαχείρησης του phpMyAdmin

5. Δημιουργία της βάσης Δεδομένων

5.1.Μοντέλο και ανάλυση της βάσης δεδομένων

Αφού σκεφτήκαμε το μοντέλο και την υλοποίηση της βάσης δεδομένων καταλήξαμε ότι χρειαζόμαστε αρχικά έναν πίνακα για τους χρήστες ή/και διαχειριστές της εφαρμογής όπου θα αποθηκεύονται τα αναγνωριστικά στοιχεία κάθε χρήστη ή διαχειριστή καθώς και άλλα στοιχεία που χρησιμεύουν για τις συναλλαγές – αποδείξεις συναλλαγών του χρήστη με το κατάστημα.

Ακόμα, χρειαστήκαμε ένα πίνακα με όλες τις πληροφορίες των προϊόντων, έναν πίνακα που θα αποθηκεύονται κάθε φορά τα προϊόντα που βάζει ο χρήστης στο καλάθι αγορών του, έναν πίνακα που θα αποθηκεύει σε μορφή ιστορικού τις συναλλαγές του χρήστη με το κατάστημα και τέλος έναν πίνακα που θα αποθηκεύει τα στοιχεία τις συναλλαγής του Paypal με τον λογαριασμό του εκάστοτε πελάτη. Έτσι, πρώτα υλοποιήσαμε ένα μοντέλο οντοτήτων-συσχετίσεων της βάσης δεδομένων μας. **[Εικόνα 1.66]**



Εικόνα 1.66: Μοντέλο οντοτήτων-συσχετίσεων βάσης δεδομένων

Στην καρτέλα Databases (Βάσεις Δεδομένων) ξεκινάμε την δημιουργία της βάσης δεδομένων μας. Της δίνουμε ένα όνομα, "ptyx" στην δική μας περίπτωση (λόγω του ότι ανήκει στην πτυχιακή μας εργασία) και επιλέγουμε utf8_unicode_ci για την κωδικοποίηση των χαρακτήρων (character encoding) που θα έχει η βάση δεδομένων μας έτσι ώστε να μην παρουσιαστεί πρόβλημα κατά την εκχώρηση ελληνικών χαρακτήρων στην βάση δεδομένων

μας και τους πίνακες που θα έχει. Οταν έχουμε σιγουρευτεί για την επιλογή μας πατάμε το κουμπί Create για να δημιουργηθεί η βάση δεδομένων μας. **[Εικόνα 1.67]**

Εικόνα 1.67: Δημιουργία της βάσης δεδομένων μέσα από το phpMyAdmin

5.2.Δημιουργία πίνακα χρηστών (auth_table)

Ο πίνακας auth_table καταχωρεί και επεξεργάζεται τους χρήστες της εφαρμογής μας. Στην καρτέλα SQL που έχουμε δημιουργήσει τη βάση δεδομένων "ptyx" τρέχουμε τον κώδικα sqlγια να δημιουργήσουμε τον πίνακα auth_table. **[Εικόνα 1.68]**

```
CREATE TABLE IF NOT EXISTS `auth_table` (
  `user_id` mediumint(1) NOT NULL AUTO_INCREMENT,
  `username` varchar(55) COLLATE utf8_unicode_ci NOT NULL DEFAULT '',
  `password` varchar(88) COLLATE utf8_unicode_ci NOT NULL DEFAULT '',
  `fullname` varchar(200) COLLATE utf8_unicode_ci NOT NULL,
  `email` varchar(99) COLLATE utf8_unicode_ci NOT NULL,
  `userlevel` tinyint(4) DEFAULT '1',
  `payment_type` tinyint(4) NOT NULL DEFAULT '1',
  `client_type` tinyint(4) NOT NULL DEFAULT '1',
  `address` tinytext COLLATE utf8_unicode_ci NOT NULL,
  `afm` varchar(9) COLLATE utf8_unicode_ci NOT NULL,
  `date_created` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`user_id`),
  UNIQUE KEY `username` (`username`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
AUTO_INCREMENT=1 ;
```

Εικόνα 1.68: Ο SQL κώδικας για τη δημιουργία του πίνακα auth_table

- user_id : είναι το πρωτεύον κλειδί (primary key) δηλαδή όλες οι τιμές τις στήλης αυτής θα είναι μοναδικές. Επίσης είναι auto_incrementδηλαδή η τιμή του θα αυξάνεται αυτόματα όποτε γίνεται νέα εγγραφή στον πίνακα auth_table .
- username: Το λεγόμενο όνομα χρήστη που θα έχει κάθε χρήστης στην εφαρμογή μας. Εχουμε περιορίσει να έχει μέγιστο 55 χαρακτήρες (varchar(55)) και να είναι μοναδικό (unique_key).
- Password: Ο κωδικός του χρήστη, μέγιστος επιτρεπόμενος αριθμός χαρακτήρων 88 (varchar(88)).
- Email: Η διεύθυνση ηλεκτρονικού ταχυδρομείου του χρήστη (μέγιστος αριθμός χαρακτήρων 99 varchar(99))

- userlevel: Το επίπεδο προσβασιμότητας του χρήστη στην εφαρμογή μας. Προεπιλεγμένη τιμή που θα εισχωρείται στη βάση θα είναι το 1. Για superusers-admins το userlevel θα έχει τιμή 2.
- payment_type: Τύπος πληρωμής των χρηστών – πελατών στην εφαρμογή μας. Το 1 αντιπροσωπεύει τύπο πληρωμής: απόδειξη και το 2 τύπο πληρωμής: τιμολόγιο.
- client_type: Ο τύπος χρήστη-πελάτη στην εφαρμογή μας. Το 1 σημαίνει ότι ο χρήστης είναι ιδιώτης, ενώ το 2 σημαίνει ότι ο χρήστης είναι εταιρία.
- address: η διεύθυνση του πελάτη τύπου tinytext δηλαδή μέχρι 255 χαρακτήρες.
- afm: ο αριθμός φορολογικού μητρώου (αν ο τύπος χρήστη είναι εταιρία) με μέγιστο αριθμό χαρακτήρων εώς 9.
- date_created: Το αλφαριθμητικό που δείχνει πότε ο χρήστης δημιουργήθηκε στον πίνακα auth_table. Είναι τύπου timestamp δηλαδή της μορφής 1970-01-01 00:00:01 . Οι αριθμοί δείχνουν την ημερομηνία και την ώρα αντίστοιχα.

Name	Type	Collation	Attributes	Null	Default	Extra
user_id	mediumint(1)		UNSIGNED	No	None	AUTO_INCREMENT
username	varchar(55)	utf8_unicode_ci		No		
password	varchar(88)	utf8_unicode_ci		No		
fullname	varchar(200)	utf8_unicode_ci		No	None	
email	varchar(99)	utf8_unicode_ci		No	None	
userlevel	tinyint(4)			Yes	1	
payment_type	tinyint(4)			No	1	
client_type	tinyint(4)			No	1	
address	tinytext	utf8_unicode_ci		No	None	
afm	varchar(9)	utf8_unicode_ci		No	None	
date_created	timestamp			No	CURRENT_TIMESTAMP	

Εικόνα 1.69: Τα στοιχεία του πίνακα auth_table στη βάση δεδομένων

5.3.Δημιουργία πίνακα καλαθιού προϊόντων (basket)

Ο πίνακας basket χρησιμεύει για να αποθηκεύονται τα προϊόντα που έχει ο εκάστοτε χρήστης στο καλάθι του. Τα προϊόντα αυτά μένουν σε αυτόν τον πίνακα μέχρι ο χρήστης-πελάτης πραγματοποιήσει την πληρωμή. Αφότου η πληρωμή του πραγματοποιηθεί η τιμή paid γίνεται 1 (από την αρχική τιμή 0). Δηλαδή το εκάστοτε προϊόν θεωρείται πληρωμένο και δεν εμφανίζεται στην οθόνη καλαθιού του χρήστη-πελάτη.

- id: Ο αύξων και μοναδικός αριθμός για κάθε γραμμή του πίνακα basket.
- userid: το αριθμητικό του χρήστη που έχει εκχωρήσει ένα προϊόνστο καλάθι του.
- productid : το αριθμητικό του προϊόντοςπου έχει καταχωρήσει ο χρήστης στο καλάθι του
- price: η τιμή του ενός τεμαχίου του προϊόντος
- quantity: Η ποσότητα του κάθε προϊόντος που έχει καταχωρήσει ο χρήστης στο καλάθι του
- pdate: Το αλφαριθμητικό τύπου ημερομηνίας – ώρας που δείχνει την ημερομηνία – ώρα που πληρώθηκε το προϊόν.
- crDate : Το αλφαριθμητικό τύπου ημερομηνίας – ώρας που δείχνει την καταχώρηση κάθε γραμμής (δηλαδή προϊόντος στο καλάθι).
- paid: Από εδώ βλέπουμε αν το προϊόν έχει πληρωθεί ή όχι. Το 0 είναι η προεπιλεγμένη τιμή και δείχνει ότι το προϊόν δεν έχει πληρωθεί. Όταν το προϊόν πληρωθεί η τιμή κάθε γραμμής του εκάστοτε προϊόντος θα αλλάξει σε 1.

```
CREATE TABLE IF NOT EXISTS `basket` (
  `id` int(6) NOT NULL AUTO_INCREMENT,
  `userid` mediumint(1) unsigned NOT NULL,
  `productid` int(1) NOT NULL,
  `price` decimal(10,2) NOT NULL,
  `quantity` mediumint(4) NOT NULL DEFAULT '1',
  `pdate` datetime NOT NULL,
  `crDate` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `paid` tinyint(1) NOT NULL DEFAULT '0',
  PRIMARY KEY (`id`),
  KEY `id` (`id`),
  KEY `userid` (`userid`),
  KEY `productid` (`productid`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
AUTO_INCREMENT=1 ;
```

Εικόνα 1.70: Ο SQL κώδικας για τη δημιουργία του πίνακα basket

```
ALTER TABLE `basket`
  ADD CONSTRAINT `basket_ibfk_1` FOREIGN KEY (`productid`) REFERENCES
`products` (`pid`) ON DELETE CASCADE ON UPDATE CASCADE,
  ADD CONSTRAINT `basket_ibfk_3` FOREIGN KEY (`userid`) REFERENCES
`auth_table` (`user_id`) ON DELETE NO ACTION ON UPDATE NO ACTION;
```

Εικόνα 1.71: Οι περιορισμοί (constraints) του πίνακα history.

5.4.Δημιουργία πίνακα ιστορικού (history)

Ο πίνακας history (ιστορικό) μας χρησιμεύει για να αποθηκεύουμε το ιστορικό των παραγγελιών του χρήστη-πελάτη. Οι εκχωρήσεις σε αυτόν τον πίνακα αρχίζουν όταν τα προϊόντα που βρίσκονται στο καλάθι του χρήστη έχουν επιβεβαιωθεί ότι είναι πληρωμένα μέσω της συναλλαγής μέσω PayPal. Δηλαδή μόνο όταν το PayPal επιβεβαιώσει ότι τα προϊόντα που ήταν στο καλάθι του χρήστη έχουν πληρωθεί.

- id : Ο αύξων και μοναδικός αριθμός για κάθε γραμμή του πίνακα history.
- transx_id : Ο αναγνωριστικός αριθμός της συναλλαγής του χρήστη-πελάτη με το Paypal.
- user_id : το αριθμητικό του χρήστη που ανήκει η παραγγελία που έγινε.
- product_id: Ο εκάστοτε μοναδικός αριθμός τουπροϊόντοςπου περιείχε η παραγγελία.
- quantity : Η ποσότητα του εκάστοτε προϊόντος που περιείχε η παραγγελία.
- created_at : Το αλφαριθμητικό τύπου ημερομηνίας – ώρας που δείχνει πότε έγινε η καταχώρηση κάθε γραμμής.

```
CREATE TABLE IF NOT EXISTS `history` (
  `id` int(5) NOT NULL AUTO_INCREMENT,
  `transx_id` varchar(100) COLLATE utf8_unicode_ci NOT NULL,
  `user_id` mediumint(1) unsigned NOT NULL,
  `product_id` int(1) NOT NULL,
  `quantity` mediumint(4) NOT NULL,
  `created_at` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  KEY `user_id` (`user_id`),
  KEY `product_id` (`product_id`),
  KEY `transx_id` (`transx_id`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
AUTO_INCREMENT=1 ;
```

Εικόνα 1.72: Η δημιουργία του πίνακα history.

```
ALTER TABLE `history`
  ADD CONSTRAINT `history_ibfk_1` FOREIGN KEY (`user_id`) REFERENCES
`auth_table` (`user_id`) ON DELETE CASCADE ON UPDATE CASCADE,
  ADD CONSTRAINT `history_ibfk_2` FOREIGN KEY (`product_id`)
REFERENCES `products` (`pid`) ON DELETE CASCADE ON UPDATE CASCADE;
```

Εικόνα 1.73: Οι περιορισμοί (constraints) του πίνακα history.

5.5.Δημιουργία πίνακα προϊόντων (products)

Στον πίνακα products καταχωρούνται τα προϊόντα της εφαρμογής μας. Κάθε προϊόν περιλαμβάνει στήλες όπως όνομα, περιγραφή, τεμάχια που βρίσκονται σε διαθεσιμότητα και φυσικά το barcode του προϊόντος. Πιο αναλυτικά:

- pid : Ο αύξων και μοναδικός αριθμός για κάθε προϊόν.
- name : Η ονομασία του εκάστοτε προϊόντος.
- price : Η τιμή του εκάστοτε προϊόντος. Είναι τύπου decimal(10,2) δηλαδή δέχεται εώς και δύο ψηφία μετά την υποδιαστολή.
- barcode : Το barcode του εκάστοτε προϊόντος.
- description : Η περιγραφή του εκάστοτε προϊόντος.
- descriptionDetails : Επιπλέον στοιχεία του προϊόντος.
- contactDetails : Στοιχεία προέλευσης – εταιρίας που ενδέχεται να ανήκει το προϊόν.
- quantity : Διαθέσιμη ποσότητα για κάθε προϊόν.
- created_at : Το αλφαριθμητικό τύπου ημερομηνίας – ώρας που δείχνει πότε έγινε η καταχώρηση κάθε προϊόντος στη βάση δεδομένων μας.

```
CREATE TABLE IF NOT EXISTS `products` (
  `pid` int(1) NOT NULL AUTO_INCREMENT,
  `name` varchar(100) COLLATE utf8_unicode_ci NOT NULL,
  `price` decimal(10,2) NOT NULL,
  `barcode` varchar(55) COLLATE utf8_unicode_ci NOT NULL,
  `description` text COLLATE utf8_unicode_ci,
  `descriptionDetails` text COLLATE utf8_unicode_ci NOT NULL,
  `contactDetails` text COLLATE utf8_unicode_ci NOT NULL,
  `quantity` mediumint(1) NOT NULL,
  `created_at` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,|
  PRIMARY KEY (`pid`),
  KEY `barcode` (`barcode`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci AUTO_INCREMENT=1 ;
```

Εικόνα 1.74: Η δημιουργία του πίνακα products.

5.6.Δημιουργία πίνακα PayPal Συναλλαγών (paypal)

Στον πίνακα PayPal καταχωρούνται οι συναλλαγές που έχουν γίνει μέσω PayPal και αποθηκεύονται τα στοιχεία – μεταβλητές που «επιστρέφει» το PayPal όταν ολοκληρωθεί η διαδικασία μια πληρωμής του χρήστη. Πιο αναλυτικά:

- id : Ο αύξων και μοναδικός αριθμός για κάθε γραμμή του πίνακα PayPal.
- transactionID : το αλφαριθμητικό που μας δίνει το PayPal σε κάθε συναλλαγή που εκτελείται.
- payKey: Το κλειδί της πληρωμής από το οποίο αναγνωρίζεται η πληρωμή που έχει γίνει μέσω PayPal. Συνήθως αυτή η τιμή χρησιμοποιήται όταν θέλουμε να αντλήσουμε πληροφορίες της PayPal συναλλαγής μέσω της PayPal διεπαφής προγραμματισμού εφαρμογής (API).
- receiver_email : Ο λογαριασμός – email του αποδέκτη της συναλλαγής.
- receiver_amount: Το ποσό της συναλλαγής που καταχωρήθηκε στο λογαριασμού του αποδέκτη.
- status : Η κατάσταση της συναλλαγής που έγινε. Οταν ήταν επιτυχής η συναλλαγή το PayPal μας επιστρέφει την τιμή COMPLETED.

```
CREATE TABLE IF NOT EXISTS `paypal` (
  `id` int(1) NOT NULL AUTO_INCREMENT,
  `transactionID` varchar(100) COLLATE utf8_unicode_ci NOT NULL,
  `payKey` varchar(100) COLLATE utf8_unicode_ci NOT NULL,
  `payer_id` varchar(60) COLLATE utf8_unicode_ci NOT NULL,
  `receiver_email` varchar(110) COLLATE utf8_unicode_ci NOT NULL,
  `receiver_amount` decimal(10,2) NOT NULL,
  `status` varchar(55) COLLATE utf8_unicode_ci NOT NULL,
  PRIMARY KEY (`id`),
  KEY `payKey` (`payKey`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
AUTO_INCREMENT=1 ;
```

Εικόνα 1.75: Η δημιουργία του πίνακα paypal.

6. Κύριο Μέρος Πτυχιακής

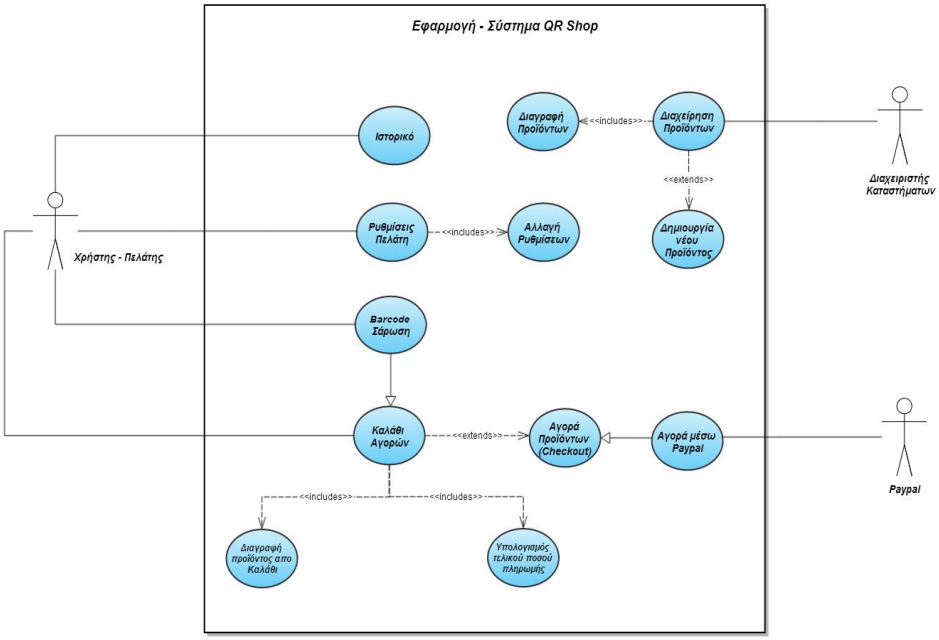
6.1. Ανάλυση Προβλήματος

Η εφαρμογή που υλοποιήθηκε θα περιέχει τρία βασικά μοντέλα, το μοντέλο του χρήστη-πελάτη, το μοντέλο του διαχειριστή και το μοντέλο του PayPal που θα παρεμβάλλεται για τις συναλλαγές του πελάτη που θέλει να αγοράσει τα προϊόντα που έχει στο καλάθι του **[Εικόνα 1.76]** .

Δηλαδή, η εφαρμογή θα χωρίζεται στο μενού του χρήστη-πελάτη και στο μενού του διαχειριστή(admin) ανάλογα με το όνομα χρήστη και τον κωδικό χρήστη που θα πληκτρολογήσει ο χρήστης. Στο μενού του χρήστη θα υπάρχουν επιλογές όπως Ιστορικό για την προβολή των προηγούμενων παραγγελιών που έχει κάνει ο χρήστης.

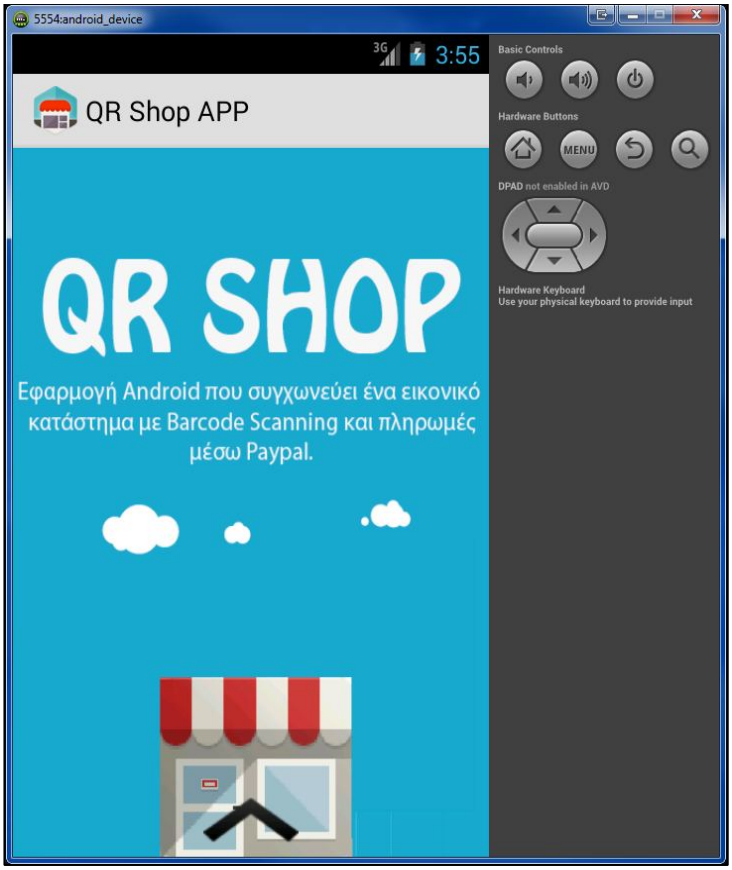
Με την επιλογή Σάρωση στο μενού του χρήστη-πελάτη θα ανοίγει η κάμερα του κινητού τηλεφώνου όπου θα μπορεί να σαρώσει το barcode ενός φυσικού προϊόντος και αν το barcode αυτό βρεθεί στην βάση δεδομένων θα προστεθεί στο καλάθι του όπου θα μείνει μέχρι ή να ολοκληρώσει ο χρήστης-πελάτης τη διαδικασία της πληρωμής.

Αφότου ο χρήστης-πελάτης θέλει να ολοκληρώσεις τις αγορές του, πατώντας ένα κουμπί θα μεταφέρεται στην όψη του PayPal όπου θα φαίνεται η παραγγελία που έχει στο καλάθι αγορών του.



Εικόνα 1.76: Use Case Διάγραμμα της εφαρμογής

6.2. Αρχική Οθόνη Εφαρμογής



Εικόνα 1.77: Αρχική Όψη Εφαρμογής

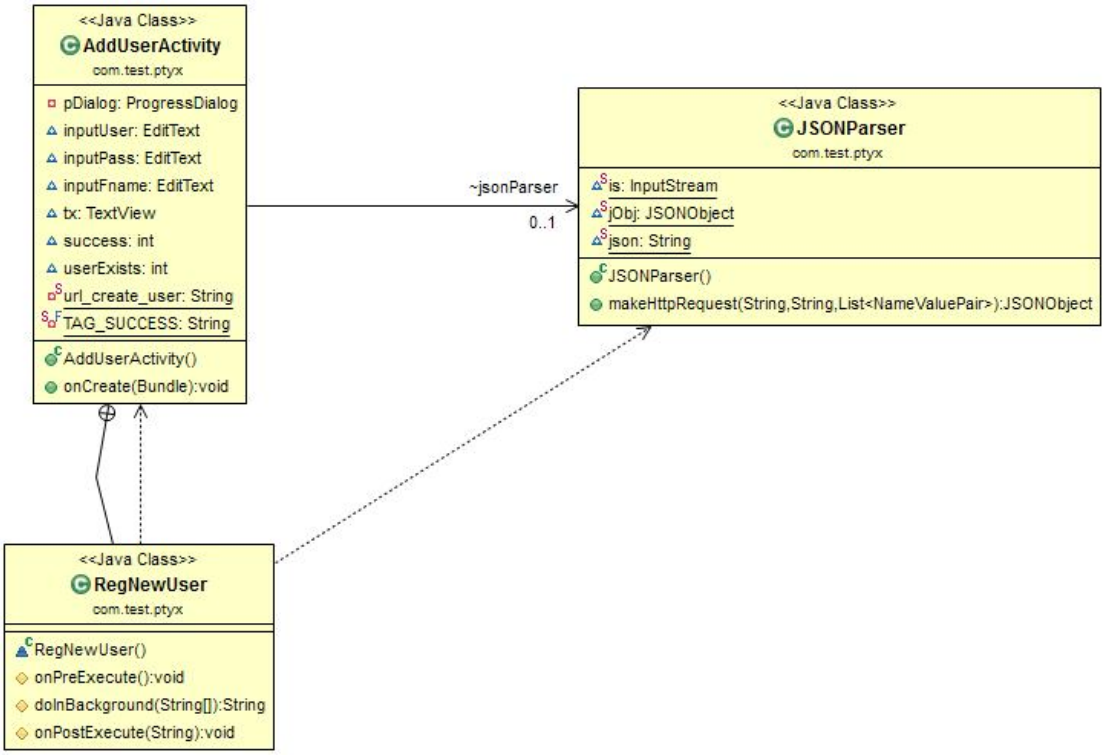
Στην όψη αυτή ο χρήστης βλέπει ένα background που σχεδιάσαμε για μα δώσουμε έμφαση στο περιεχόμενο της. Η αρχική όψη της εφαρμογής μας, εκτός από το background image **[εικόνα 1.77]**, έχει ένα Sliding Drawer το οποίο επιτρέπει στο χρήστη να σύρει μέσω μιας λαβής (στην περίπτωσή μας είναι ένα μαύρο βέλος που δείχνει προς τα πάνω) και μόλις τελειώσει το σύρσιμο φέρνει ένα νέο περιεχόμενο στην οθόνη. Τα περιεχόμενα που εμφανίζονται μόλις τελειώσει το σύρσιμο, είναι τα στοιχεία εισαγωγής του χρήστη στην εφαρμογή ή το κουμπί για εγγραφή νέου μέλους στην εφαρμογή μας.

6.3.Σύνδεση διεπαφής χρήστη με τη βάση δεδομένων

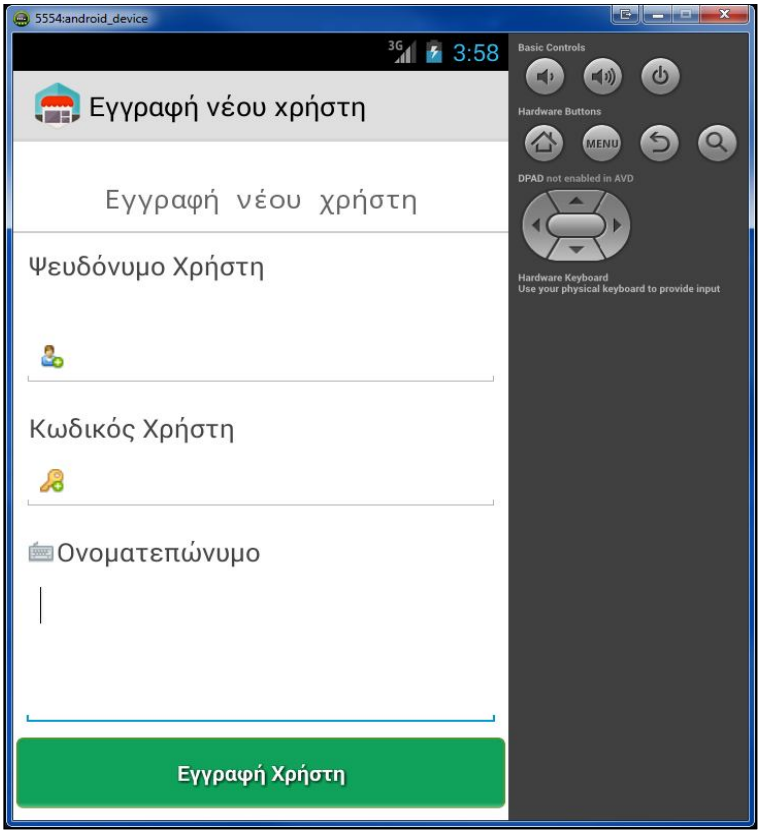
Στον περισσότερο πηγαίο κώδικα της εφαρμογής μας χρειάστηκε να υλοποιήσουμε ένα αντίστοιχο μοντέλο με αυτόν που φαίνεται στην εικόνα **[Εικόνα 1.78]** . Όπως βλέπουμε, κάθε Activity περιέχει την δήλωση των μεταβλητών που θα χρειαστούμε, ας δούμε όμως τα πιο σημαντικά στοιχεία πως δουλεύουν. Έχουμε λοιπόν το activity με όνομα AddUserActivity που είναι μια κλάση που εμφανίζεται στην εγγραφή νέου χρήστη-πελάτη στην εφαρμογή μας. Σε αυτή τη κλάση – δραστηριότητα έχουμε δημιουργήσει 3 textfields όπου ο χρήστης μπορεί να καταχωρήσει για να εγγραφεί στην εφαρμογή μας. Τα διαθέσιμα πεδία σε αυτήν την όψη είναι: το Ψευδόνυμο χρήστη (username), ο κωδικός χρήστη και το ονοματεπώνυμο.

Ας δούμε ένα παράδειγμα με τη δήλωση στην κλάση μας ενός EditText και τη σύνδεση του με το xml layout αρχείο. Με την εντολή EditText inputPass; δηλώνουμε μια

μεταβλητή τύπου EditText και όνομα inputPass που αντιστοιχεί στο πεδίο κειμένου για το κωδικό που θα έχει ο νέος χρήστης. Αργότερα μέσα στην onCreate() μέθοδο της κλάσης «λέμε» στην εφαρμογή που να βρει το EditText αυτό μέσα στο layout xml αρχείο που έχουμε μέσω της εντολής :
inputPass = (EditText) findViewById(R.id.inputpass);



Εικόνα 1.78: Η βασική αρχιτεκτονική για τη σύνδεση διεπαφής χρήστη με τη βάση δεδομένων.



Εικόνα 1.79: Η όθόνη δημιουργίας νέου χρήστη-πελάτη στην εφαρμογή

Πάμε τώρα στην κλάση RegNewUser όπως φαίνεται στην εικόνα.. **[Εικόνα 1.78]** . Πατώντας το κουμπί «Εγγραφή Χρήστη» όπως βλέπουμε στην εικόνα **[Εικόνα 1.79]** εκτελείται μέσω ενός onClickListener() η ασύχρονη διεργασία (AsyncTask) RegNewUser() (με τη δημιουργία ενός νέου αντικειμένου της κλάσης αυτής) με την εντολή:

new RegNewUser().execute();

Στην onPreExecute() μέθοδο της ασύγχρονης διεργασίας εμφανίζεται ένα progressDialog στην οθόνη του χρήστη όσο εκτελείται (στο παρασκήνιο) η διεργασία μέσω της doInBackground() μεθόδου. Σε αυτή τη μέθοδο καλείται η κλάση JSONParser **[Εικόνα 1.78]** που ως σκοπό έχει να δημιουργήσει τις σωστές παραμέτρους και να στείλει μέσω ενός HttpRequest το σωστό url που έχουμε ορίσει εμείς μέσω μια String μεταβλητής.

Παρακάτω βλέπουμε την λίστα-πίνακα που δημιουργούμε και τον τρόπο που ορίζουμε τις μεταβλητές username, pass και fullname. Στον πίνακα αυτόν μπαίνουν οι τιμές των πεδίων κειμένου (EditTexts) που πληκτρολόγησε ο χρήστης και αργότερα δημιουργείται ένα key value pair όπως φαίνεται παρακάτω:

```
List<NameValuePair> params = new ArrayList<NameValuePair>();
params.add(new BasicNameValuePair("username", name));
params.add(new BasicNameValuePair("pass", pass));
params.add(new BasicNameValuePair("fullname", fullname));
```

Το url με τις παραμέτρους στέλνεται μέσω του HttpRequest στα αρχεία του webserver μας αφού πρώτα μετατρέπεται στην κλάση JsonParser για να μπορεί να επεξεργαστεί από τον κώδικα του PHP αρχείου που στέλνεται όπως φαίνεται παρακάτω:

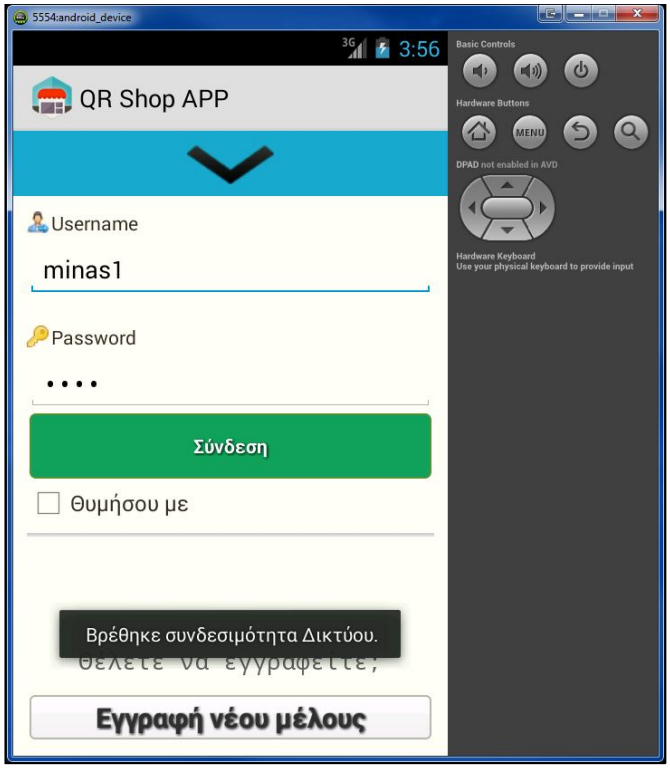
```
JSONObject json = jsonParser.makeHttpRequest(url_create_user,"POST", params);
```

Το PHP αρχείο registeruser.php **[Εικόνα 1.80]** δέχεται τις παραμέτρους που έχει στείλει η εφαρμογή μας και βλέπει στη βάση εάν υπάρχει το όνομα χρήστη ίδιο με εκείνο που έβαλε ο χρήστης στέλνοντας ένα query στη βάση δεδομένων. Αν δεν βρεθεί ένα ίδιο όνομα χρήστη συνεχίζει και εκχωρεί στη βάση δεδομένων με ένα query πάλι στη βάση όπως φαίνεται στη **γραμμή κώδικα 22 στην Εικόνα 1.80**. Αφού εκτελεστεί σωστά το query, επιστρέφεται το flag success = 1 κωδικοποιημένο σε JSON μορφή και έτσι καταλαβαίνουμε ότι ο νέος χρήστης δημιουργήθηκε επιτυχώς στην βάση δεδομένων.

```php
<?php

$response = array();

if (isset($_POST['username']) && isset($_POST['pass']) && isset($_POST['fullname'])) {

    $name = $_POST['username'];
    $pass = $_POST['pass'];
    $fullname = $_POST['fullname'];

    // include db connect class
    require_once __DIR__ . '/db_connect.php';

    // connecting to db
    $db = new DB_CONNECT();

    $usernameCheck = mysql_query("SELECT username FROM auth_table WHERE username = '".$name."'");
    $usernameExists= mysql_num_rows($usernameCheck)>0?1:0;

    // mysql inserting a new row
    $result = mysql_query("INSERT INTO auth_table(username, password, fullname) VALUES('$name', '$pass', '$fullname')");

    // check if row inserted or not
    if ($result) {
        // successfully inserted into database
        $response["success"] = 1;
        $response["userExists"] = $usernameExists;
        $response["message"] = "User successfully created.";

        // echoing JSON response
        echo json_encode($response);
    } else {
        // failed to insert row
        $response["success"] = 0;
        $response["userExists"] = $usernameExists;
        $response["message"] = "Oops! An error occurred.";

        // echoing JSON response
        echo json_encode($response);
    }
} else {
    // required field is missing
    $response["success"] = 0;
    $response["message"] = "Required field(s) for user register is(are) missing";

    // echoing JSON response
    echo json_encode($response);
}
?>
```

Εικόνα 1.80: Το αρχείο registeruser.php για την εγγραφή χρήστη

6.4.Είσοδος χρήστη



Εικόνα 1.81: Όψη σύνδεσης χρήστη

Ο χρήστης αφού πραγματοποιήσει την εγγραφή του μπορεί να εισέλθει στην εφαρμογή πατώντας το κουμπί «Σύνδεση». Με τον onclickListener του κουμπιού αυτού εκτελείται το AsyncTask που στέλνει τις τιμές (values) από τα πεδία (EditText) του «username» και «password». Αφού σταλθούν οι τιμές αυτών στο αρχείο PHP εκείνο τις παίρνει και εκτελεί ένα select query στην βάση για να ελέγξει αν τα πεδία είναι σωστά. Πρέπει και το username και το password να ισχύουν αλλιώς ο χρήστης δεν θα μπορέσει να εισέλθει στην εφαρμογή.

```php
<?php
// array for JSON response
$response = array();
// include db connect class
require_once __DIR__ . '/db_connect.php';
// connecting to db
$db = new DB_CONNECT();
if (isset($_POST["pid"])) {
    $pid = $_POST['pid'];
    $pass = $_POST['pass'];

    $result = mysql_query("SELECT * FROM auth_table WHERE username = '".$pid."' AND password = '".$pass."'");

    if (!empty($result)) {

        if (mysql_num_rows($result) > 0) {

            $result = mysql_fetch_array($result);

            $product = array();
            $product["userlevel"] = $result["userlevel"];
            $product["id"] = $result["user_id"];
            $product["username"] = $result["username"];
            $product["password"] = $result["password"];


            $response["success"] = 1;


            $response["product"] = array();

            array_push($response["product"], $product);


            echo json_encode($response);
        } else {

            $response["success"] = 0;
            $response["message"] = "No user found";

            echo json_encode($response);
        }
    } else {

        $response["success"] = 0;
        $response["message"] = "No user found";

        echo json_encode($response);
    }
} else {

    $response["success"] = 0;
    $response["message"] = "Required field(s) is missing";
    echo json_encode($response);
}
?>
```

Εικόνα 1.82: Αρχείο PHP για σύνδεση χρήστη ή διαχειριστή στο μενού της εφαρμογής

6.5.Ο χρήστης-πελάτης της εφαρμογής

Η υποενότητα αυτή αναλύει την διαδικασία που θα ακολουθήσει ένας πελάτης-χρήστης της εφαρμογής από την στιγμή που ξεκινάει να σαρώνει και να προσθέτει προϊόντα στο καλάθι του μέχρι και την ολοκλήρωση της συναλλαγής του με το κατάστημα καθώς και διάφορά άλλα στοιχεία ή επιλογές που έχει για να διευκολυνθεί όλη η διαδικασία.

6.5.1. Το κεντρικό μενού του χρήστη-πελάτη της εφαρμογής

Αφού γίνει η σύνδεση του χρήστη στην εφαρμογή εμφανίζεται το κεντρικό μενού που έχει 6 επιλογές όπως Ρυθμίσεις, Βοήθεια, Ιστορικό, Καλάθι Αγορών, Σάρωση, Αποσύνδεση όπως φαίνεται στην εικόνα **[Εικόνα 1.83]**. Κάθε επιλογή είναι ένα εικονίδιο-

κουμπί στο layout που έχουμε στον φάκελο /res του project μας. Το κάθε κουμπί είναι της μορφής:

```
<Button
    android:id="@+id/btn_help"
    style="@style/DashboardButton"
    android:drawableTop="@drawable/btn_help"
    android:text="Βοήθεια"
    android:textColor="#9D1309"
    android:typeface="monospace" />
```

Δηλαδή, με το android:drawableTop βάζουμε το εικονίδιο που βρίσκεται στον φάκελο drawable στο workspace. Το android:text είναι το κείμενο που θέλουμε να εμφανίζεται, το android:textColor είναι το χρώμα του κειμένου και το android:typeface δηλώνει τη γραμματοσειρά του κειμένου για κάθε επιλογή-κουμπί.

Το μενού έχει ένα header που είναι μια εικόνα που έχει δηλωθεί με την εξής μορφή:

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  style="@style/ActionBarCompat" >
  <ImageView
    android:layout_width="wrap_content"
    android:layout_height="fill_parent"
    android:layout_gravity="center_vertical|center"
    android:layout_weight="1"
    android:clickable="false"
    android:scaleType="center"
    android:src="@drawable/top_logo"/>
</LinearLayout>
```

Με το android:src ορίζουμε το φάκελο που βρίσκεται η εικόνα που φαίνεται στο πάνω μέρος της οθόνης του μενού της εφαρμογής. Κάθε φορά που ο χρήστης πατάει μια επιλογή-κουμπί στο μενού έχουμε ορίσει αντίστοιχους listeners που μεταφέρουν το χρήστη στην επόμενη οθόνη, δηλαδή στο επόμενο Activity.

Αν έχει συνδεθεί ο διαχειριστής της εφαρμογής δηλαδή κάποιος χρήστης που στον πίνακα της βάσης δεδομένων έχει userlevel διαφορετικό του 1 τότε πατώντας το ίδιο κουμπί έχουμε ορίσει να πηγαίνει σε άλλο Activity.

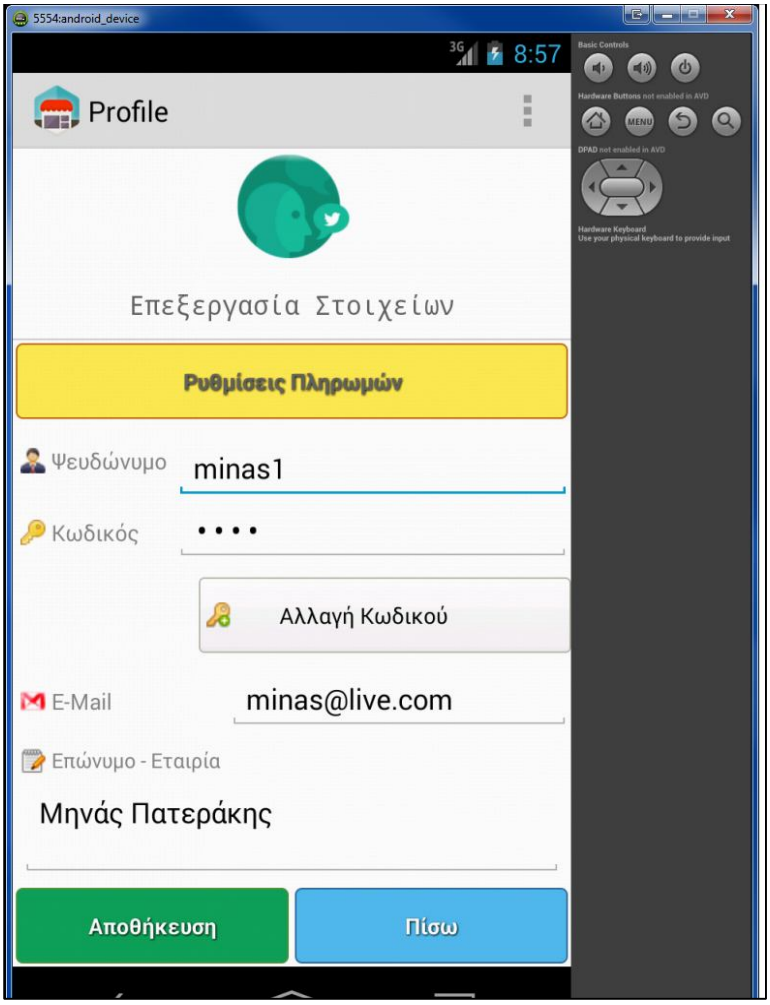
Ας δούμε σαν παράδειγμα τον κώδικα του κουμπιού που μεταφέρει έναν χρήστη στις προσωπικές του Ρυθμίσεις, αλλά έναν διαχειριστή σε μια άλλη (στην εμφάνιση όλων των προϊόντων του καταστήματος).

```
btnViewProducts.setOnClickListener(new View.OnClickListener() {
public void onClick(View view) {
// Αν είναι χρήστης πελάτης
if(theuserlevel.equals("1")){
Intent i = new Intent(getApplicationContext(), ClientProfileActivity.class);
i.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
i.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
startActivity(i);
}else{ // Αν είναι χρήστης διαχειριστής
Intent i = new Intent(getApplicationContext(), AdminAllProductsList.class);
i.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
i.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
startActivity(i);
}
}
});
```

Εικόνα 1.83: Το κεντρικό μενού της εφαρμογής αφότου συνδεθεί ένα χρήστης-πελάτης.

6.5.2. Ρυθμίσεις και εξεργασία στοιχείων χρήστη

Σε αυτή την όψη ο χρήστης μπορεί να αλλάξει τα στοιχεία του όπως το όνομα χρήστη, το email και το επώνυμό του. Όταν πατηθεί λοιπόν το κουμπί «Αποθήκευση» στέλνεται το HttpRequest με αντίστοιχες παραμέτρους που περιέχουν τις τιμές που έχει πληκτρολογήσει ο χρήστης στα αντίστοιχα EditTexts. Για λόγους ασφαλείας δεν φαίνεται ο κωδικός του χρήστη που περιέχεται στη βάση. Αυτό επιτυγχάνεται με τη δήλωση της παραμέτρου android:password="true" στο EditText του κωδικού.



Εικόνα 1.84: Η όψη των ρυθμίσεων του χρήστη

6.5.3. Ρυθμίσεις και εξεργασία επιλογών στις πληρωμές του χρήστη

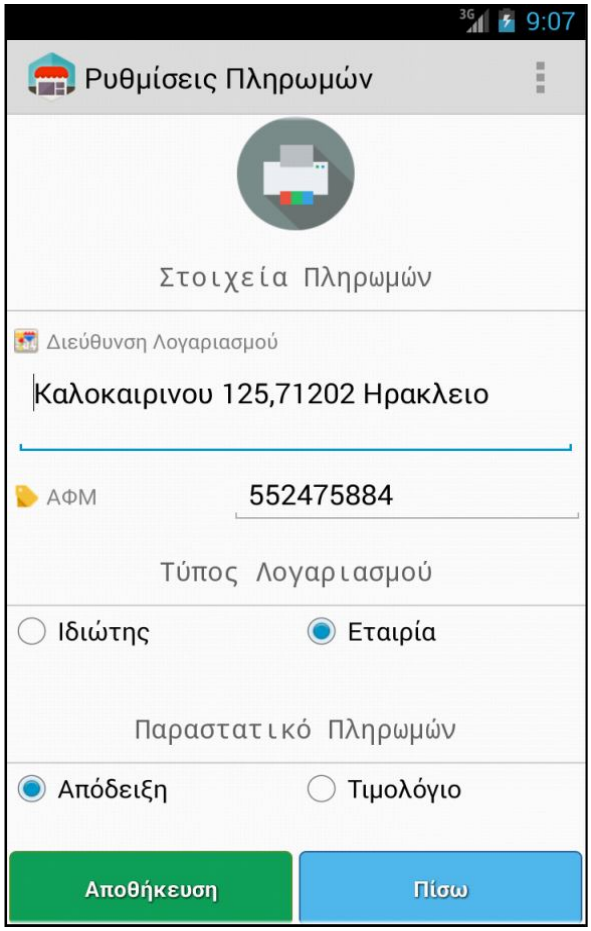
Σε αυτή την οθόνη ο χρήστης αλλάζει τα στοιχεία που αφορούν τις εικονικές πληρωμές που πραγματοποιεί, για να υπάρχει καλύτερος έλεγχος στις αποδείξεις του εικονικού καταστήματος. Για την εμφάνιση στην οθόνη του Τύπου Λογαριασμού και το Παραστατικό Πληρωμών χρησιμοποιήσαμε RadioGroups, το οποίο εμπεριέχει δύο RadioButtons, όπως φαίνεται παρακάτω:

```
<RadioGroup
   android:id="@+id/radioGroup0"
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:layout_above="@+id/fbplacename"
   android:layout_alignParentLeft="true"
   android:orientation="horizontal"
   android:paddingBottom="17dp" >

   <RadioButton
      android:id="@+id/radioClientType1"
      android:layout_width="0dp"
      android:layout_height="wrap_content"
      android:layout_weight="1"
      android:text="Ιδιώτης" />
```

```
    <RadioButton
        android:id="@+id/radioClientType2"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="Εταιρία" />

</RadioGroup>
```

Όπως βλέπουμε στην εικόνα παρακάτω **[Εικόνα 1.85]** ο χρήστης πρέπει να διαλέξει μια επιλογή από τη στήλη Τύπος Λογαριασμού και μία από τη στήλη Παραστατικό Πληρωμής. Στον κώδικα μας για να ελέγξουμε τί πάτησε ο χρήστης βλέπουμε πιο radio button είναι επιλεγμένο κάθε φορά και έχουμε ένα listener που αλλάζει μια τιμή τύπου int όπως σας δείχνουμε παρακάτω:

```
rg0.setOnCheckedChangeListener(new OnCheckedChangeListener()
        {

        public void onCheckedChanged(RadioGroup group, int checkedId)
          {
          // TODO Auto-generated method stub
          if(radioClientType1.isChecked())
            {
                                    //αλλάζει την τιμή int του τύπου Λογαριασμού σε 1
             clientTypeVal = 1;

            }
          else if(radioClientType2.isChecked())
            {                       //αλλάζει την τιμή int του τύπου Λογαριασμού σε 2
             clientTypeVal = 2;
            }
         }
});
```



Εικόνα 1.85: Η όψη των ρυθμίσεων πληρωμής ενός χρήστη της εφαρμογής μας

Όταν πατιέται το κουμπί «Αποθήκευση» στέλνεται το HttpRequest με τις παραμέτρους και σε ένα PHP αρχείο που το ονομάσαμε update_payment_client.php και φαίνεται στην εικόνα [**Εικόνα 1.86**] . Το αρχείο αυτό κάνει ένα update query στη βάση δεδομένων για να αλλάξει τα τιμές που έχουν τα EditTexts πεδία που καταχώρησε ο χρήστης. Όταν η τιμή success του json πίνακα μας που επιστράφηκε είναι 1 τότε σημαίνει ότι το update ερώτημα (query) έγινε επιτυχώς και έτσι η εφαρμογή επιστρέφει ένα Toast message όπως φαίνεται παρακάτω:

```
Toast.makeText(getApplicationContext(),
"Αποθήκευση στοιχείων: Επιτυχής.",Toast.LENGTH_LONG).show();
```

```php
<?php

$response = array();

// check for required fields
if (isset($_POST['pid'])) {

    $userid         = $_POST['pid'];
    $address        = $_POST['address'];
    $afm            = $_POST['afm'];
    $payment_type   = $_POST['payment_type'];
    $client_type    = $_POST['client_type'];

    // include db connect class
    require_once __DIR__ . '/db_connect.php';

    // connecting to db
    $db = new DB_CONNECT();

    $result = mysql_query("UPDATE auth_table SET address = '$address',
                                                 afm = '$afm',
                                                 payment_type = '$payment_type',
                                                 client_type = '$client_type' WHERE user_id = $userid")
                                                 or die(mysql_error());


    if ($result) {
        // successfull
        $response["success"] = 1;
        $response["message"] = "Payment profile successfully updated.";

        echo json_encode($response);
    } else {

    }
} else {
    $response["success"] = 0;
    $response["message"] = "Required field(s) is missing";


    echo json_encode($response);
}
?>
```

Εικόνα 1.86: Το PHP αρχείο αποθήκευσης των ρυθμίσεων πληρωμής στη βάση δεδομένων

6.5.4. Υλοποίηση Σάρωσης Barcodes

Για τη διαδικασία της σάρωσης των προϊόντων χρησιμοποιήσαμε τη βιβλιοθήκη ανοιχτού κώδικα zBar και το sdk της όπως αναφέραμε σε προηγούμενο κεφάλαιο.

Καταρχήν πρέπει να ορίσουμε τα permissions στο AndroidManifest.xml αρχείο του πρότζεκτ μας για να επιτρέπεται η πρόσβαση της κάμερας από την εφαρμογή μας :

<uses-permission android:name="android.permission.CAMERA" />

<uses-feature android:name="android.hardware.camera" />

<uses-feature android:name="android.hardware.camera.autofocus" />

Αργότερα, δημιουργούμε φυσικά το activity που θα ξεκινάει όταν ο χρήστης πατήσει το κουμπί “Σάρωση” στο κεντρικό μενού.

Μέσα στον κώδικα του Activity δηλώνουμε μια μεταβλητή τύπου Camera την mCamera. Η android κλάση Camera (Camera API) χρησιμεύει για ρυθμίσεις image capture, έναρξη/λήξη προεσκόπησης κάμερας, τράβηγμα φωτογραφιών (snapshots) και κωδικοποίηση frames για βίντεο.

Ακόμα, δηλώνουμε και την μεταβλητή scanner που είναι τύπου ImageScanner. Η κλάση ImageScanner περιέχεται στη βιβλιοθήκη zBar και χρησιμεύει για να διαβάσει barcodes από 2-D εικόνες. Μετά καλούμε μέσα από την μέθοδο getCameraInstance() ένα στιγμιότυπο της κάμερας και το βάζουμε στην μεταβλητή mCamera. Αν επιστρέψει τιμή null

ή exception error σημαίνει ότι δεν βρέθηκε κάμερα στο smartphone ή δεν είναι συμβατή με την εφαρμογή ή χρησιμοποιείται ήδη από μια άλλη εφαρμογή.

Μετά ορίζουμε το FrameLayout id από το layout xml αρχείο του Activity και βάζουμε μέσα την προεπισκόπηση της κάμερας.

Στην onPause() μέθοδο εκτελούμε την releaseCamera() όπως φαίνεται παρακάτω:

```
private void releaseCamera() {
    if (mCamera != null) {
        previewing = false;
        mCamera.setPreviewCallback(null);
        mCamera.release();
        mCamera = null;
    }
}
```

Η releaseCamera() απελευθερώνει τους πόρους του αντικειμένου της κάμερας που χρησιμοποιούμε για να είναι ελεύθεροι σε άλλες εφαρμογές. Όσο η κάμερα είναι διαθέσιμη προσθέτουμε μια ακόμα διεργασία στην ουρά μηνυμάτων του main UI Thread, την autoFocus() μέθοδο όπως φαίνεται παρακάτω:

```
private Runnable doAutoFocus = new Runnable() {
      public void run() {
        if (previewing)
          mCamera.autoFocus(autoFocusCB);
      }
    };
```

Με το autofocus ξεκινά αυτόματη εστίαση στην κάμερα και καταγράφει μια λειτουργία επανάκλησης (callback) να τρέχει όταν η κάμερα έχει εστιάσει. Η μέθοδος αυτή είναι έγκυρη μόνο όταν η προεπισκόπηση είναι ενεργή. Οι συσκευές που δεν υποστηρίζουν αυτόματη εστίαση θα λάβουν ένα "ψεύτικο" callback σε αυτή τη διασύνδεση (interface).

```
AutoFocusCallback autoFocusCB = new AutoFocusCallback() {
      public void onAutoFocus(boolean success, Camera camera) {
        autoFocusHandler.postDelayed(doAutoFocus, 1000);
      }
  };
```

Με την μέθοδο postDelayed που είδαμε στον παραπάνω κώδικα επιτρέπουμε στην μέθοδο doAutoFocus να μπει στην ουρά μηνυμάτων του UI Thread και να εκτελεστεί εκεί που βρίσκεται ο handler. Το 1000 φανερώνει τον αριθμό των milliseconds που θα τρέχει κάθε φορά περιοδικά η μέθοδος doAutoFocus.

Σε αυτό το Activity έχουμε δύο buttons και φυσικά τους αντίστοιχους onclickListeners. Με την διασύνδεση PreviewCallback έχουμε την μέθοδο onPreviewFrame που καλείται όταν εμφανίζονται τα frames της προεπισκόπησης της κάμερας.

Με την εντολή:

```
Image barcode = new Image(size.width, size.height, "Y800");
```

δημιουργούμε ένα αντικείμενο τύπου Image που παίρνουμε από μια κλάση της βιβλιοθήκης zBar την Image.java που χρησιμοποιείται για να αποθηκεύσει δεδομένα εικόνων μαζί με την αντίστοιχη μορφή τους (format) αλλά και size metadata. Εκτός από το μήκος και το ύψος δηλώνουμε και έναν κωδικό μορφής FourCC εικόνας για τα δεδομένα της εικόνας του barcode που θέλουμε να πάρουμε.

Με την εντολή barcode.setData(data); δίνουμε στη βιβλιοθήκη zBar το frame προεπισκόπησης της κάμερας για να πάρει τα δεδομένα του.

Με την εντολή int result = scanner.scanImage(barcode); η zBar βιβλιοθήκη προσπαθεί να βρει και να αποκωδικοποιήσει σύμβολα σχετικά με barcodes στα δεδομένα της εικόνας. Η scanImage μέθοδος επιστρέφει τον αριθμό των συμβόλων που βρέθηκαν γι'αυτό την δηλώσαμε ως integer.

Αν ο αριθμός των συμβόλων δεν είναι 0 (δηλαδή βρέθηκαν σύμβολα) τότε σταματάμε την προεπισκόπηση frames με την μέθοδο stopPreview(); στο αντικείμενο της κάμερας και καλούμε την SymbolSet κλάση της zBar που μας δίνει το αποτέλεσμα την αποκωδικοποιημένης εικόνας/συμβόλων με την εντολή: SymbolSet syms = scanner.getResults(); και μετά με μια επανάληψη (for) στην ακολουθία των εκάστοτε συμβόλων ειδοποιούμε με ένα toastMessage την οθόνη του χρήστη, τα εμφανίζουμε σε ένα TextView που έχουμε ορίσει και ταυτόχρονα αλλάζουμε την τιμή της boolean μεταβλητής barcodeScanned σε true με τον παρακάτω κώδικα:

```
for (Symbol sym : syms) {
scanText.setText("barcode result " + sym.getData());
barcodeScanned = true;
bcode=""+sym.getData();
Toast.makeText(getApplicationContext(), "Εγινε λήψη Barcode",
Toast.LENGTH_SHORT).show();
}
```

Εικόνα 1.87: Αποτέλεσμα σάρωσης ενός Barcode

Εάν σαρωθεί ένα προϊόν και δεν βρεθεί στη βάση δεδομένων μετά από το SELECT ερώτημα που εκτελείται στη βάση δεδομένων του καταστήματος, ο πελάτης μεταφέρεται σε μια νέα

όψη όπου ειδοποιείται ότι δεν βρέθηκε το προϊόν **[Εικόνα 1.88]** και έχει την επιλογή να ξαναδοκιμάσει να ξανασαρώσει το ίδιο ή κάποιο άλλο προϊόν.

Εικόνα 1.88: Όψη για την μη εύρεση του προϊόντος μετά από μια σάρωση ενός barcode

6.5.5. Το καλάθι αγορών του χρήστη-πελάτη

Τα προϊόντα που έχει στο καλάθι του ο κάθε πελάτης-χρήστης φορτώνονται μέσω μια ασύγχρονης διεργασίας (AsyncTask) που παίρνει τα δεδομένα από ένα αρχείο PHP που βρίσκεται στον webserver.

Η όψη του καλαθιού αγορών βασίζεται στην λογική της λίστας ανά αντικείμενο-προϊόν, δηλαδή στο ListView. Η προβολή στοιχείων σε λίστες είναι μια συχνή επιλογή που χρησιμοποιείται στις εφαρμογές. Ας ρίξουμε μια ματιά στην υλοποίηση για την προβολή των προϊόντων με τη χρήση λίστας.

Εικόνα 1.89: Εμφάνιση μιας λίστας σε ένα smartphone

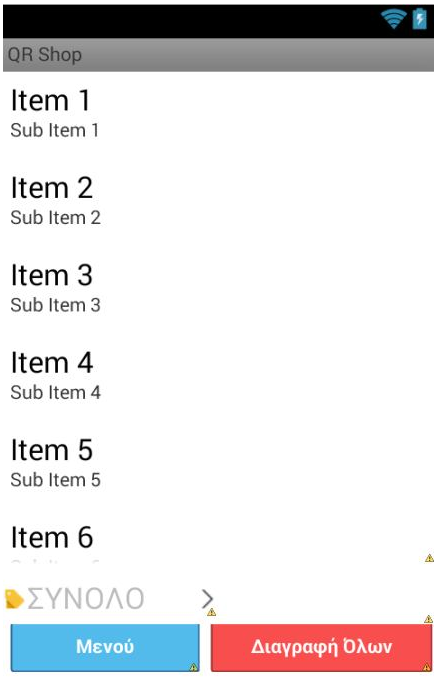
Με την ListView μπορούμε να δημιουργήσουμε μια λίστα που να κάνει εύκολα scroll. Για να βάλουμε όμως τα αντικείμενα (δηλαδή στην περίπτωση μας τα προϊόντα) ένα ένα σε στοίχηση, χρειαζόμαστε έναν Adapter που δράει ως ο δίαυλος μεταξύ του πίνακα με τα στοιχεία που πέρνουμε από την βάση δεδομένων (σε μορφή JSON) και της όψη της λίστας έτσι όπως θα εμφανιστεί στην οθόνη.

Ο Adapter μιας λίστας διαχειρίζεται το μοντέλο των δεδομένων που του δίνουμε και το προσαρμόζει σε μια σειρά (row) μέσα στην όψη της λίστας.

Η δημιουργία της λίστας μέσα στο layout xml αρχείο γίνεται ως εξής:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent" >

  <ListView
    android:id="@+id/lv_products"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:layout_above="@id/invoiceButton"
    android:layout_alignParentTop="true" />

</RelativeLayout>
```

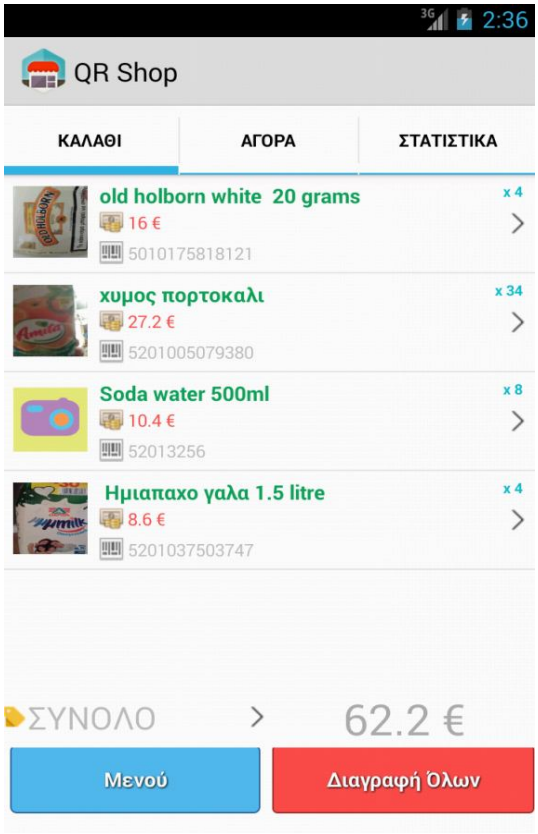
Και η όψη μας στην αρχική της μορφή διαμορφώνεται όπως φαίνεται εδώ: **[Εικόνα 1.90]**



Εικόνα 1.90: Εμφάνιση του ListView σε ένα layout αρχείο

Ο adapter είναι εκείνος που μετατρέπει τα δεδομένα στο view που θέλουμε εμείς. Πιο συγκεκριμένα χρησιμοποιούμε SimpleAdapter, με τον οποίο μπορούμε να μετατρέψουμε δεδομένα σε όψεις (views) που έχουμε ορίσει στο layout xml αρχείο μας, δηλαδή σε ένα id ενός ListView.

Κάθε φορά που εμφανίζει κάτι, ο adapter πρέπει όμως να εμφανίσει και την εικόνα για κάθε προϊόν. Αυτό γίνεται πάλι με μια ασύγχρονη διεργασία (AsyncTask) που κατεβάζει την κάθε φωτογραφία από το url που έχουμε εμείς ορίσει. Αργότερα ενώ εκτελείται η διεργασία αυτή, αποθηκεύει την κάθε φωτογραφία στο path της cache περιοχής της συσκευής. Η cache τεχνική χρησιμεύει για να μην χρειάζεται κάθε φορά να κατεβάζει τη φωτογραφία, κερδίζοντας έτσι ταχύτητα στην όψη της λίστας με το καλάθι των προϊόντων όπως τελικά εμφανίζεται **[εικόνα 1.91]** .



Εικόνα 1.91: Το καλάθι προϊόντων ενός χρήστη-πελάτη

Επίσης, αξίζει να αναφέρουμε ότι περιέχονται και κρυφά πεδία που μας βοηθούν στο να παίρνουμε το αναγνωριστικό (id) για κάθε προϊόν που παίρνουμε από την βάση δεδομένων για να μην εμφανίζονται στον πελάτη/διαχειριστή. Το κρυφό id δηλαδή χρησιμεύει όταν ο χρήστης πατήσει πάνω στο προϊόν και ενεργοποιηθεί ο clicklistener που μας πηγαίνει στο επόμενο activity.

6.5.6. Επεξεργασία προϊόντων καλαθιού αγορών

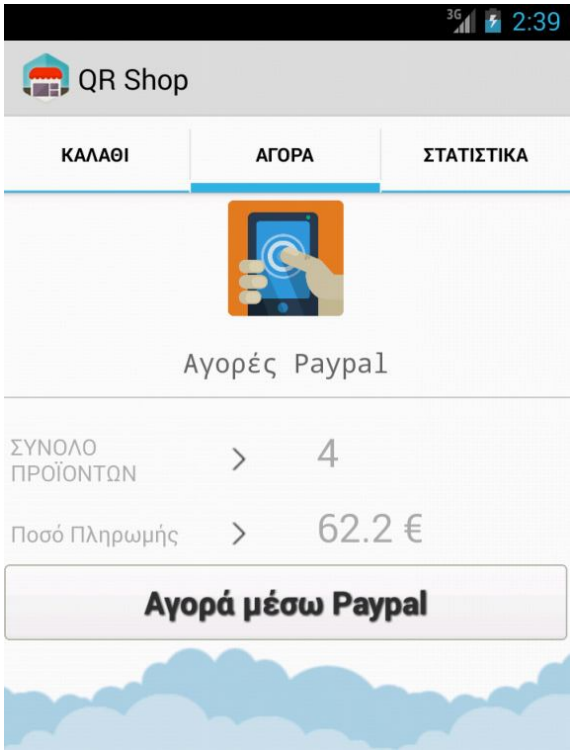
Πατώντας σε κάθε προϊόν ξεχωριστά, ο χρήστης-πελάτης μπορεί να δει διάφορα στοιχεία του εκάστοτε προϊόντος όπως τη ποσότητα που επέλεξε, την ονομασία και τη φωτογραφία του προϊόντος αλλά το σημαντικότερο σε αυτή την όψη είναι ότι μπορεί να αλλάξει την αρχική ποσότητα κάνοντας slide στο seekbar που υπάρχει στο κάτω μέρος της όψης αυτής όπως φαίνεται στην εικόνα **[Εικόνα 1.92]** .

Εικόνα 1.92: Η όψη για την επεξεργασία ενός προϊόντος που είναι στο καλάθι του πελάτη

Βέβαια, κάτι τέτοιο δεν θα ήταν εφικτό τόσο άμεσα σε ένα φυσικό σημείο πώλησης μέσα σε κάποιο κατάστημα που θα μπορούσε να χρησιμοποιηθεί αυτή η εφαρμογή, , αφού κάπως αλλιώς θα πρέπει να ελέγχεται η ποσότητα που έχει ο κάθε πελάτης στο καλάθι του. Εμείς όμως επιλέξαμε να προσθέσουμε έτσι αυτή τη λειτουργία για να καλύψουμε κάπως αυτό το κενό.

6.5.7. Διαδικασία ενσωμάτωσης εφαρμογής με Paypal

Όπως είδαμε στο κεφάλαιο 4 αφού εισάγουμε την βιβλιοθήκη για mobile συναλλαγές μέσω Paypal πρέπει να προσθέσουμε και τον κώδικα που θα συνδέεται με το Paypal στην αντίστοιχη Activity. Καταρχήν να αναφέρουμε ότι το Paypal μας επιτρέπει να έχουμε ένα δοκιμαστικό περιβάλλον (Sandbox Environment) που χρησιμεύει για προγραμματιστές έτσι ώστε να κάνουν δοκιμές στις εφαρμογές που δημιουργούν. Το Live Environment είναι το κανονικό περιβάλλον όπου όλα τα στοιχεία πρέπει να είναι τα αληθινά καθώς εκεί οι συναλλαγές είναι με πραγματικά χρήματα.



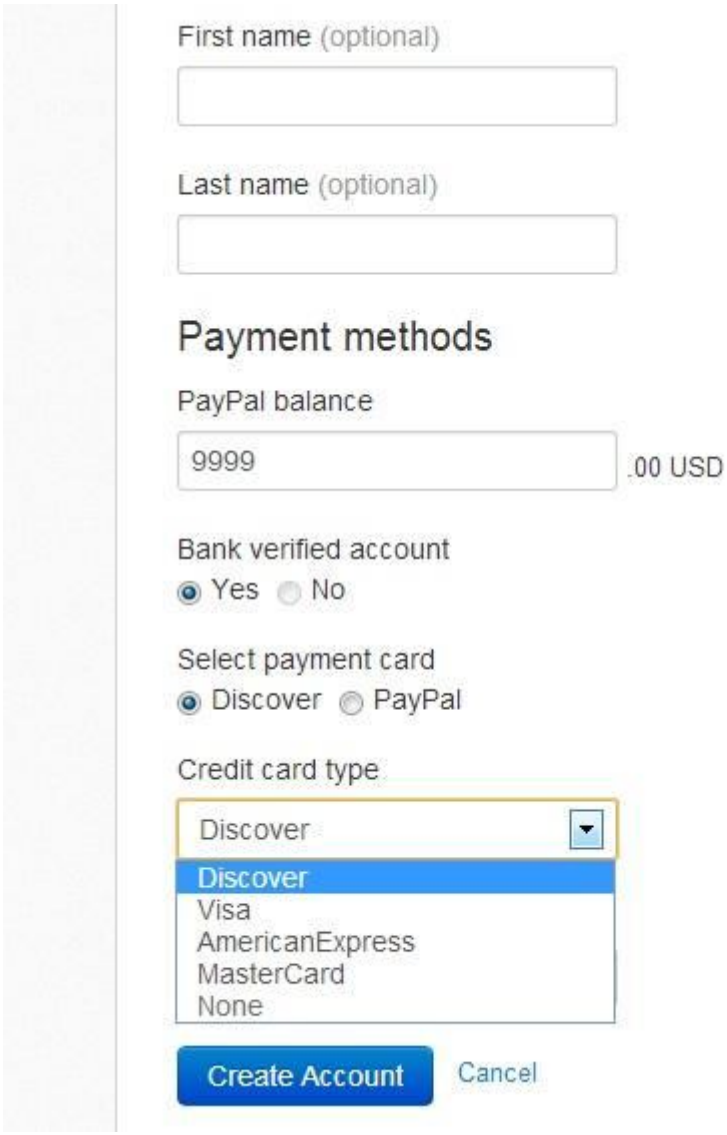
Εικόνα 1.93: Η όψη πριν την αγορά μέσω Paypal

Στο Sandbox περιβάλλον οι προγραμματιστές δημιουργούν εικονικούς λογαριασμούς με εικονικές πιστωτικές κάρτες που βάζουν εκείνοι οποιοδήποτε ποσό μέσα στις κάρτες αυτές για να κάνουν δοκιμές στην εφαρμογή τους εξομοιώνοντας έτσι την διαδικασία μιας πληρωμής.

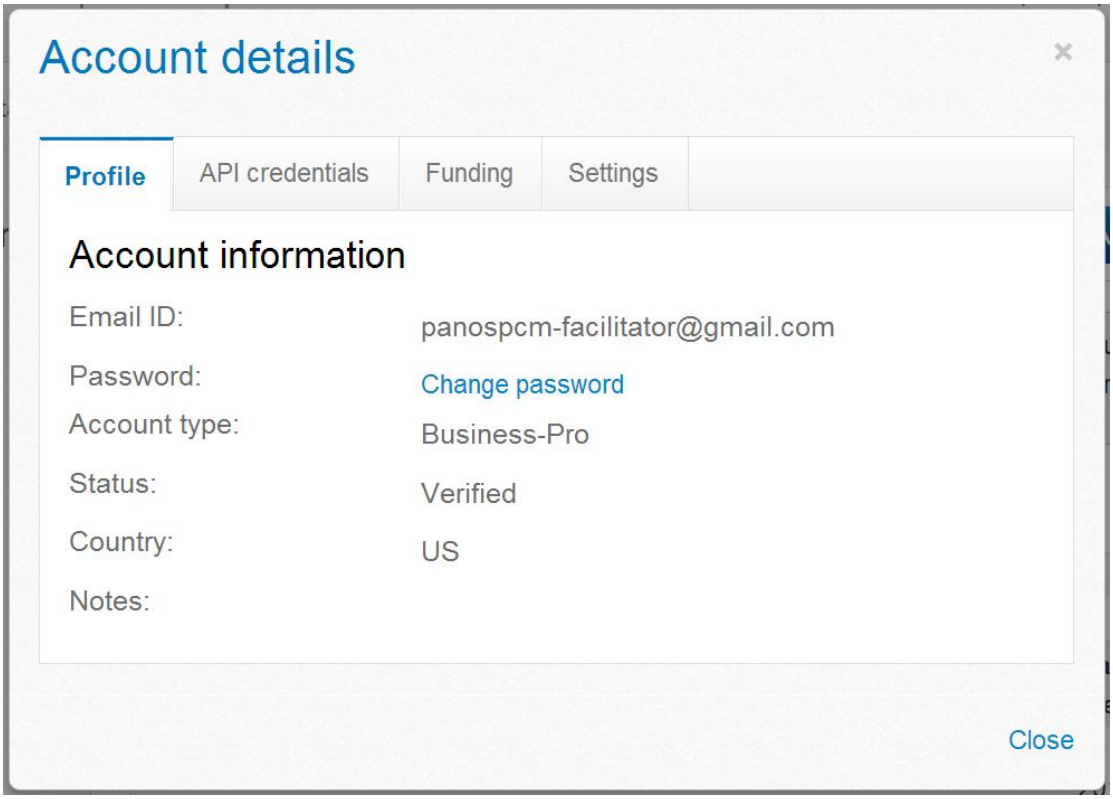
Πηγαίνουμε λοιπόν στη διεύθυνση https://developer.paypal.com/ και δημιουργούμε λογαριασμό για να μπούμε μέσα στο μενού. Όταν μπούμε στο μενού πηγαίνουμε στο link sandbox accounts και μετά πατάμε create new για να φτιάξουμε ένα (εικονικό) merchant λογαριασμό, δηλαδή το λογαριασμό του παραλήπτη της πληρωμής του καταστήματος που θα μας έρχονται τα χρήματα από τις συναλλαγές με τους πελάτες μας.

Έτσι, δημιουργήσαμε έναν τυχαίο λογαριασμό με email panospcm-facilitator@gmail.com ο οποίος είναι τύπου Business-Pro (αυτό χρειάζεται για να πούμε στο Paypal ότι αυτός ο λογαριασμός θα είναι ο merchant-πωλητής) **[Εικόνα 1.95]**.

Αργότερα, δημιουργούμε και έναν (εικονικό) λογαριασμό τύπου Personal ο οποίος θα είναι ο υποτιθέμενος-εικονικός πελάτης για τις πληρωμές που θα κάνουμε. Στον λογαριασμό του πελάτη βάζουμε μια εικονική πιστωτική κάρτα(Credit card type) με ένα εικονικό ποσό (Paypal Balance) που έχουμε μέσα στην κάρτα, για να μπορέσουμε έτσι να κάνουμε πληρωμές με το λογαριασμό αυτό όπως φαίνεται στην εικόνα **[Εικόνα 1.94]** .



Εικόνα 1.94: Φόρμα δημιουργίας εικονικού λογαριασμού στο PayPal Developers



Εικόνα 1.95: Πληροφορίες λογαριασμού στο PayPal Developers

Ας δούμε πως γίνεται η αρχικοποίηση του Paypal μέσα στον κώδικα μας. Αρχικά προσθέτουμε (αν δεν το έχουμε ήδη κάνει) δύο permissions στο AndroidManifest αρχείο του πρότζεκτ μας όπως φαίνεται παρακάτω.

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
```

Το πρώτο permission επιτρέπει στην εφαρμογή να ανοίγει θύρες δικτύου (network sockets) και το δεύτερο επιτρέπει στην εφαρμογή να διαβάζει στοιχεία από τη συσκευή.
Μέσα στο Activity του αρχείου PaymentPaypalGo.java που η όψη της φαίνεται στην εικόνα **[Εικόνα 1.93]** όταν πατηθεί το κουμπί "Αγορά μέσω PayPal" και ενεργοποιηθεί ο onclickListener τότε κάνουμε execute την ασύγχρονη διεργασία που αρχικοποιεί τις επιλογές που θέλουμε αφού πρώτα έχει φορτώσει μέσω μιας άλλης ασύγχρονης διεργασίας το καλάθι του χρήστη που θα πάει προς πληρωμή για να το εισάγει σε εκείνο του PayPal.

Πρώτα εισάγουμε μια μεταβλητή τύπου static final string με όνομα APP_ID που είναι το μοναδικό κλειδί της εφαρμογής που έχουμε πάρει από το paypal. Για το sandbox περιβάλλον δεν χρειάζεται να αλλάξουμε το κλειδί αυτό αλλά για το live θα έπρεπε να το αλλάξουμε για να δουλέψει μιας και κάθε κλειδί σχετίζεται με το e-mail του παραλήπτη των χρημάτων και το Paypal κάνει αυτήν την ταυτοποίηση πριν ξεκινήσει η διαδικασία της πληρωμής.

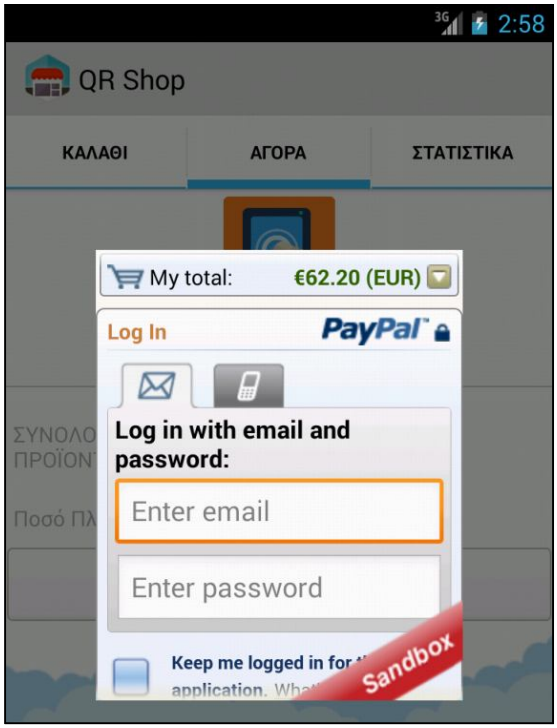
Παρακάτω βλέπουμε πως εισάγουμε τα προϊόντα που παίρνουμε από το HttpRequest στον webserver μας και τα προσθέτουμε μέσω μιας επανάληψης (for) κάθε μεταβλητή για κάθε προϊόν στο δικό της Arraylist. Στους πίνακες αυτούς προσθέτουμε για κάθε προϊόν τα στοιχεία του που θα στείλουμε στο PayPal δηλαδή το ArrayList paynameList περιέχει ένα ένα τα ονόματα των προϊόντων του καλαθιού μας,το paypriceList περιέχει την τιμή κάθε προϊόντος, το payquantityList περιέχει την ποσότητα για κάθε ένα προϊόν που έχουμε στο καλάθι, το paysubtotal περιέχει την τιμή του κάθε προϊόντων επί την ποσότητα του, δηλαδή το υποσύνολο κάθε προϊόντος, μεταξύ άλλων, όπως φαίνεται στον κώδικα παρακάτω:

```
private class PayPalInitializer extends AsyncTask < Void, Void, Boolean > {
```

```
private static final String APP_ID = "APP-80W284485P519543T";
private Context mContext;
private ProgressDialog mProgressDialog;
private ProgressDialog mProgressDialogCancelled;

public PayPalInitializer(Context context) {
   mContext = context;
}

@Override
protected void onPreExecute() {
   super.onPreExecute();
   mProgressDialog = new ProgressDialog(mContext);
   mProgressDialog.setMessage("Προσθήκη προϊόντων για αγορά...");
   mProgressDialog.show();
}

@Override
protected Boolean doInBackground(Void...params) {
   boolean success = false;
   int success2;
   try {
      JSONParser jsonParser = new JSONParser();
      List < NameValuePair > params1 = new ArrayList < NameValuePair > ();
      params1.add(new BasicNameValuePair("pid", theuserid));
      JSONObject json = jsonParser.makeHttpRequest(
         payment_cart_details, "GET", params1);
      success2 = json.getInt(TAG_SUCCESS);
      if (success2 == 1) {
         products = json.getJSONArray(TAG_PRODUCT);
         for (int i = 0; i < products.length(); i++) {
            JSONObject c = products.getJSONObject(i);
            paynameList.add(c.getString(TAG_NAME));
            paypriceList.add(c.getString(TAG_PRICE));
            payprodidList.add(c.getString(TAG_PRODUCTID));
            payquantityList.add(c.getInt(TAG_QUANTITY));
            paytotalList.add(c.getString(TAG_TOTALSUM));
            payprodnumList.add(c.getInt(TAG_PRODNUM));
            payeachprodtotal.add(c.getString(TAG_PRODTOTAL));
            paysubtotal.add(c.getString(TAG_SUBTOTAL));
            theuserid = c.getString(TAG_USERID);
         }
      } else {
      }
   } catch (JSONException e) {
      e.printStackTrace();
   }
```

```
        PayPal payPal = PayPal.getInstance();
        if (payPal == null) {
            payPal = PayPal.initWithAppID(mContext, APP_ID, PayPal.ENV_SANDBOX);
            payPal.setLanguage("en_US");
            payPal.setFeesPayer(PayPal.FEEPAYER_EACHRECEIVER);
            payPal.setShippingEnabled(false);
            payPal.setDynamicAmountCalculationEnabled(false);
            // --
            if (payPal.isLibraryInitialized()) {
                success = true;
            }
        }
        return success;
    }

    @Override
    protected void onPostExecute(Boolean result) {
        super.onPostExecute(result);
        mProgressDialog.hide();
        mProgressDialog = null;
        setupButton();

    }
};
```

Αφού εκτελεστεί σωστά ο κώδικας μας εμφανίζεται η οθόνη που ξεκινάει η συναλλαγή μας με το PayPal όπως φαίνεται στην εικόνα. **[εικόνα 1.96]**



Εικόνα 1.96: Αρχική όψη Paypal μέσα στην εφαρμογή

Για να δούμε αν τα προϊόντα που έχουμε στο καλάθι μας είναι ίδια με αυτά που δηλώθηκαν στο PayPal πατάμε το βελάκι στα δεξιά της όψης του PayPal και έτσι εμφανίζεται η λίστα με τα προϊόντα μας και τα στοιχεία που χρειαζόμαστε δηλαδή όνομα, τιμή, ποσότητα και υποσύνολο για το εκάστοτε προϊόν όπως φαίνεται στην εικόνα **[Εικόνα 1.97]** .

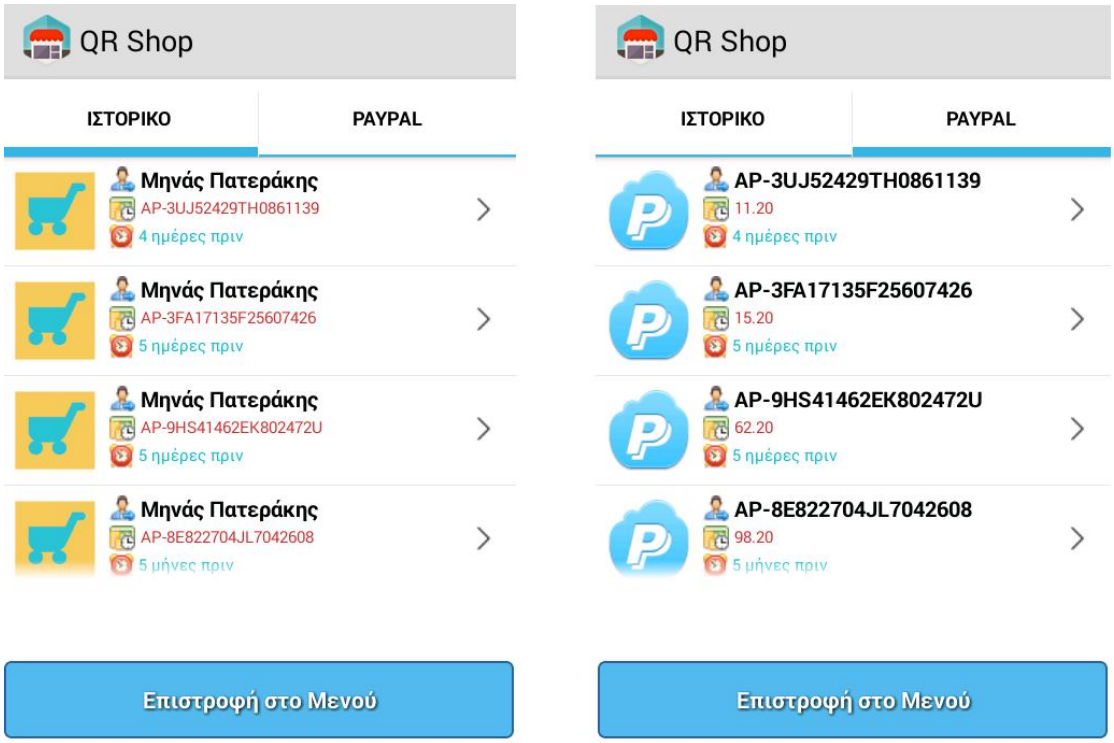
Εικόνα 1.97: Εμφάνιση προϊόντων καλαθιού μέσα στο Paypal

Αφού πατήσουμε το κουμπί της πληρωμής η διαδικασία συνεχίζεται. Το Paypal μας εμφανίζει μια οθόνη για πληκτρολόγηση-δημιουργία PIN (για ταχύτερη πληρωμή την επόμενη φορά) αλλά εμείς πατάμε skip **[Εικόνα 1.98]**.

Εικόνα 1.98: Το Paypal δείχνει ότι η πληρωμή πραγματοποιήθηκε

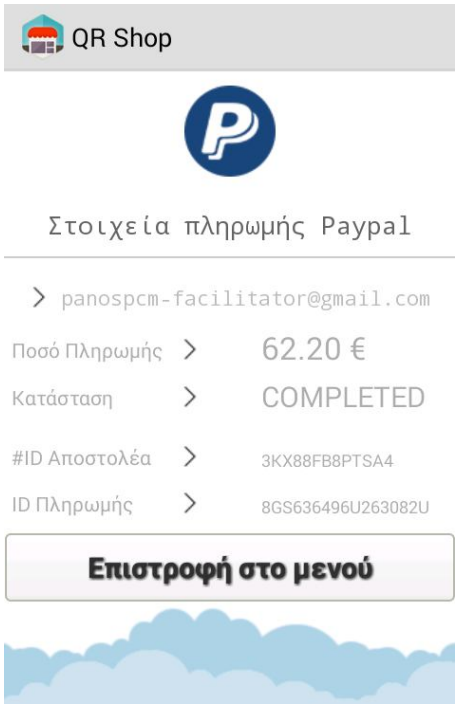
Αργότερα, το Paypal επιβεβαιώνει τα στοιχεία που δώσαμε για την συναλλαγή μια τελευταία φορά και αφού το κάνει πηγαίνει πίσω στην εφαρμογή μας και πιο συγκεκριμένα στην onActivityResult (που εκτελείται όταν γίνει exit μια Activity) και στέλνει την παράμετρο RESULT_OK αν δεν υπήρξαν σφάλματα ή οποιοδήποτε error στην διαδικασία αυτή. Έτσι εμείς στον κώδικα μας με μια switch , αν ολοκληρώθηκε επιτυχώς η διαδικασία μεταφέρουμε τον χρήστη-πελάτη σε ένα άλλο Activity για να μπορεί να δει την απόδειξη του μαζί με τα στοιχεία που επέστρεψε το Paypal.

Όταν ολοκληρωθεί η πληρωμή, στην βάση δεδομένων τα προϊόντα στο καλάθι του χρήστη στον πίνακα basket γίνονται paid = 1 με ένα update query. Αργότερα με ένα insert query για κάθε προϊόν μπαίνουν τα προϊόντα στον πίνακα history για να μπορεί αργότερα ο χρήστης-πελάτης να δει μια προηγούμενη παραγγελία που έχει κάνει. Επίσης κάνουμε και ένα insert query στον πίνακα paypal με το μοναδικό paykey που επέστρεψε το paypal στη συναλλαγή που έγινε αποθηκεύοντας έτσι τα στοιχεία για να υπάρχουν όταν ο χρήστης-πελάτης επιλέξει να τα δει.



Εικόνα 1.99: Οι καρτέλες ιστορικού πληρωμών και PayPal πληρωμών αντίστοιχα

Αφού ολοκληρωθεί η πληρωμή μπορούμε να πάμε πίσω στο αρχικό μενού και πατώντας στο ιστορικό βλέπουμε τις καρτέλες (tabs) για «ΙΣΤΟΡΙΚΟ» και «PAYPAL» όπου βρίσκονται οι (παλαιότερες) συναλλαγές που αφορούν συνολικά το κατάστημα και το Paypal αντίστοιχα. **[εικόνα 1.99]**



Εικόνα 1.100: Η οθόνη εμφάνισης τα στοιχεία της πληρωμής που έγινε μέσω PayPal

6.5.8. Απόδειξη πληρωμής

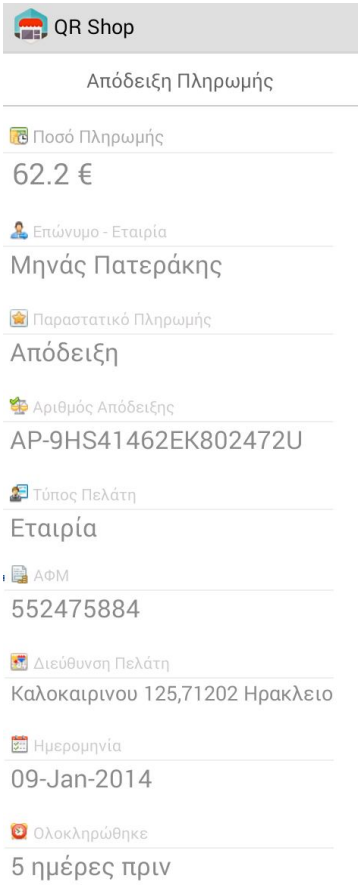
Ο χρήστης μπορεί να δει μέσω του ιστορικού του όλες τις αποδείξεις για τις πληρωμές του. Σαν παράμετρος στο PHP αρχείο στέλνεται ο αριθμός απόδειξης δηλαδή η στήλη transx_id από τον πίνακα history. Μετά εκτελείται (ως συνήθως) ένα MySQL query που συνδυάζει και συνδέει πίνακες της βάσης δεδομένων μεταξύ τους για να πάρει τα δεδομένα που εμφανίζονται. Παρακάτω **[Εικόνα 1.102]** βλέπουμε το PHP αρχείο που παίρνει τα δεδομένα από την βάση δεδομένων και τα εκτυπώνει σε μορφή JSON. Τα στοιχεία για ΑΦΜ, τύπος παραστατικού, τύπο πελάτη, διεύθυνση πελάτη, παίρνονται από τον πίνακα auth_table. Όπως είπαμε σε προηγούμενη ενότητα ο χρήστης μπορεί να αλλάξει τα στοιχεία αυτά από τις «Ρυθμίσεις Πληρωμών» και ότι διαλέξει θα φαίνεται απευθείας και στις αποδείξεις των πληρωμών του. **[Εικόνα 1.103]**
Επίσης χρησιμοποιήσαμε μια συνάρτηση στην PHP για να παίρνουμε το datetime timestamp της ημερομηνίας της πληρωμής, που είναι σε μορφή 2000-01-01 12:12:12 και να το μετατρέπει σε xx λεπτά-ώρες-μέρες-μηνες πριν κτλ.

```php
<?php

function nicetime($date)
{
    setlocale(LC_TIME, "el-GR");

    if(empty($date)) {
        return "No date";
    }

    $periods         = array("δευτερόλεπτα", "λεπτά", "ώρες", "ημέρες", "εβδομάδες", "μήνες", "χρόνια", "δεκαετίες");

    $lengths         = array("60","60","24","7","4.35","12","10");

    $now             = time(); // TODO change this for global settings.. DONE.
    $unix_date         = strtotime($date);

    // check validity of date
    if(empty($unix_date)) {
        return "some time go.";
    }

    // is it future date or past date
    if($now > $unix_date) {
        $difference     = $now - $unix_date;
        $tense         = "πριν";

    } else if ($now == $unix_date) {
        return "just now";
    } else {
        $difference     = $unix_date - $now;
        $tense         = "from now";
    }

    for($j = 0; $difference >= $lengths[$j] && $j < count($lengths)-1; $j++) {
        $difference /= $lengths[$j];
    }

    $difference = round($difference);

    if($difference != 1) {
        //  $periods[$j].= "s";
    }

    return "$difference $periods[$j] {$tense}";
}
?>
```

Εικόνα 1.101: Συνάρτηση για μετατροπή UNIX timestamp σε μορφοποιημένη μορφή ημερομηνίας

```php
<?php

$response = array();


require_once '../db_connect.php';

// σύνδεση με βάση δεδομένων
$db = new DB_CONNECT();
$sum = 0;

if (isset($_GET["transxid"])) {
    $transactionID = $_GET['transxid'];

    $result = mysql_query("SELECT   history.id,
                                    history.transx_id,
                                    history.user_id,
                                    history.product_id,
                                    history.quantity,
                                    history.created_at,
                                    history.afm,
                                    history.client_type,
                                    history.payment_type,
                                    products.pid,
                                    auth_table.fullname,
                                    auth_table.address,
                                    products.name,
                                    products.price,
                                    products.barcode,
                                    products.description
                                    FROM history
                                    INNER JOIN products ON products.pid = history.product_id
                                    INNER JOIN auth_table ON history.user_id = auth_table.user_id
                                    WHERE history.transx_id = '".$transactionID."'");
$prodsum = array();
$totalsum= array();
$sumArray = array();
$calcSum = array();
$i=0;
$calcSum = mysql_fetch_array($result);
    if (!empty($result)) {
        // check for empty result
        if (mysql_num_rows($result) > 0) {
            $counter = 0;

        for($counter = 0;$counter<mysql_num_rows($result);$counter++){

            $prodsum[$counter]              = $calcSum["quantity"] *  $calcSum["price"];


        }
        $totalPriceSum                      = array_sum($prodsum);

            mysql_data_seek($result, 0);
            $product = array();
             $response["product"] = array();

             while ($row = mysql_fetch_array($result)) {
             $product["pid"]          = $row["pid"];
             $product["productid"]    = $row["pid"];
             $product["quantity"]     = $row["quantity"];
             $product["name"]         = $row["name"];
             $product["userid"]       = $row["user_id"];
             $product["clientName"]   = $row["fullname"];
             $product["date_cr"]      = date("d-M-Y", strtotime($row["created_at"]));
             $product["dateText"]     = $db->nicetime($row["created_at"]);
             $product["price"]        = $row["price"];
             $product["transxid"]     = $row["transx_id"];
             $product["barcode"]      = $row["barcode"];

             $product["afm"]          = $row["afm"];
             $product["address"]      = $row["address"];
             $product["client_type"] = $row["client_type"]==1?"Ιδιώτης":"Εταιρία";
             $product["payment_type"]= $row["payment_type"]==1?"Απόδειξη":"Τιμολόγιο";
             $product["totalproduct"]= mysql_num_rows($result);
             $prodsum[$i]               = $row["quantity"] *  $row["price"];
             $totalsum[$i]              = $prodsum[$i];

          $product["totalsum"]        =array_sum($totalsum);
          $product["prodtotal"]       =array_sum($prodsum);
          $product["subtotal"]        = array_sum($prodsum);
          $product["totalPriceSum"]   = $totalPriceSum;
          $prodsum = array();
          $i++;
              $product["price"]       = $row["price"];
              $product["transxid"]    = $row["transx_id"];
              $response["success"] = 1;

              array_push($response["product"], $product);
              }

              echo json_encode($response);
          } else {

              $response["success"] = 0;
              $response["message"] = "No product found";


              echo json_encode($response);
          }
      } else {

          $response["success"] = 0;
          $response["message"] = "No product found";

          echo json_encode($response);
      }
  } else {

      $response["success"] = 0;
      $response["message"] = "Required field(s) is missing";

      echo json_encode($response);
  }
```

Εικόνα 1.102: Αρχείο PHP σύνδεσης με βάση δεδομένων για προβολή απόδειξης



Εικόνα 1.103: Εμφάνιση οθόνης Απόδειξης πληρωμής

6.6.Διαχειριστές της εφαρμογής

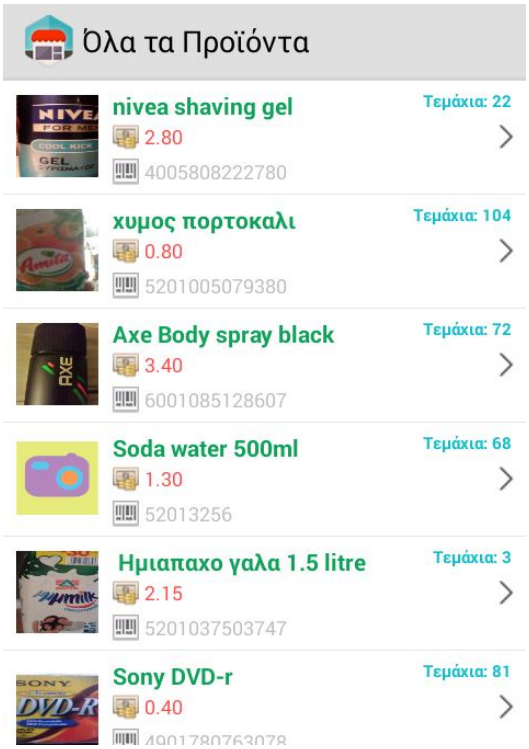
Διαχειριστής στην εφαρμογή μας είναι ή ο έμπορος ή κάποιοι υπάλληλοι που θα έχει το υποτιθέμενο κατάστημα. Σαν οντότητα στην εφαρμογή ο διαχειριστής έχει userlevel = 2 στην βάση δεδομένων και συγκεκριμένα στον πίνακα auth_table που ελέγχεται κατά την είσοδος (login) ενός χρήστη στην εφαρμογή.

6.6.1. Κεντρικό μενού Διαχειριστών

Το μενού του διαχειριστή διαφέρει μιας και ο διαχειριστής του καταστήματος δεν χρειάζεται να έχει καλάθι αγορών αφού το μόνο που θέλει να κάνει είναι να μπορεί εύκολα να προσθέσει και να διαχειρίζεται τα προϊόντα που έχει η αποθήκη του καταστήματος. Έτσι, όπως φαίνεται και στην εικόνα **[Εικόνα 1.104]** υπάρχουν οι επιλογές για προβολή όλων των προϊόντων (Τα Προϊόντα) , επιλογή προσθήκης προϊόντος (Προσθέστε Προϊόν) και επιλογή ανεβάσματος φωτογραφίας και αντιστοίχησης της σε ένα προϊόν (Photo->Server).

Εικόνα 1.104: Το μενού επιλογών του διαχειριστή του καταστήματος

6.6.2. Προβολή προϊόντων διαχειριστών



Εικόνα 1.105: Προβολή όλων των προϊόντων του καταστήματος

Ο τρόπος που ο διαχειριστής βλέπει τα προϊόντα του καταστήματος είναι ίδιος με αυτός που περιγράψαμε στην υποενότητα 6.5.5 (δηλαδή χρήση listView με adapter). Το μόνο που αλλάζει είναι το query που κάνουμε στη βάση δεδομένων. Εδώ, κάνουμε SELECT όλα τα προϊόντα που υπάρχουν στη βάση και δεν τα δείχνουμε ανάλογα με το id του χρήστη.

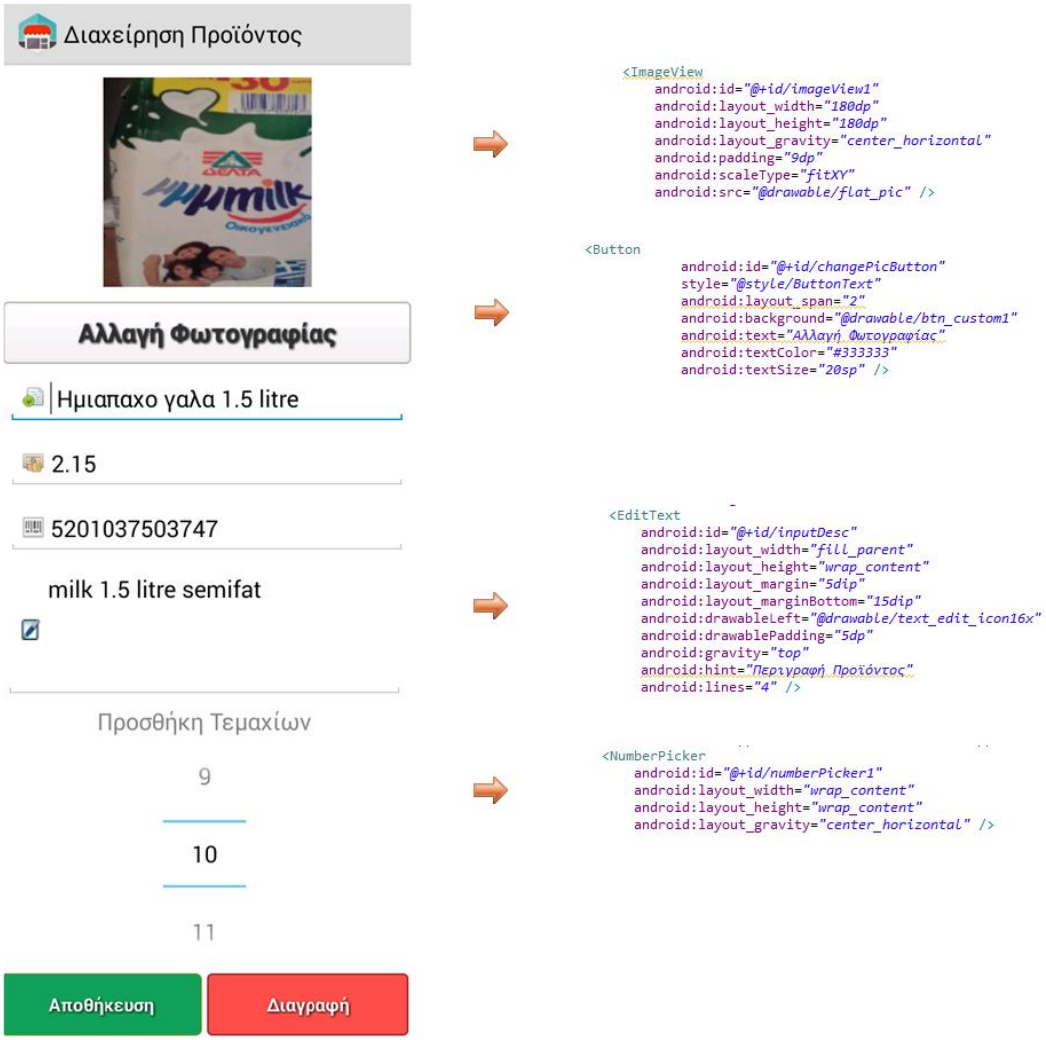
Στο layout αρχείο χρησιμοποιήσαμε RelativeLayout και μέσα του ένα imageView για να εμφανίσουμε την εικόνα, την οποία πέρνουμε από ένα url και την εμφανίζουμε ασύγχρονα. Ακόμα, βάλαμε TextView για ότι περιέχει κείμενο όπως το όνομα προϊόντος, την τιμή και τον αριθμό barcode και τέλος ένα ακόμα imageView για να εμφανίσουμε το εικονίδιο για το βελάκι. Έτσι, εμφανίσαμε την όψη που φαίνεται στην εικόνα **[Εικόνα 1.105]** .

6.6.3. Επεξεργασία προϊόντων καταστήματος

Στην επεξεργασία ενός προϊόντος ο διαχειριστής μπορεί να επεξεργαστεί τις ρυθμίσεις ενός προϊόντος. Δηλαδή, μπορεί να αλλάξει ή να προσθέσει μια φωτογραφία, να αλλάξει το όνομα του προϊόντος, την τιμή του, το barcode, την περιγραφή του και τέλος να προσθέσει τεμάχια για αυτό το προϊόν. Το layout αρχείο είναι τύπου Scrollview και τα πεδία που αλλάζει τις ρυθμίσεις του προϊόντος είναι τύπου EditText, ενώ η φωτογραφία είναι τύπου ImageView όπως φαίνεται στην εικόνα **[Εικόνα 1.106]** .

Η φωτογραφία φορτώνεται μέσα από ένα url που έχουμε ορίσει και μέσα από μια AsyncTask που μετατρέπει το αρχείο (που βρίσκεται στο url) σε αντικείμενο Bitmap μέσω της public κλάσης BitmapFactory και έτσι το εμφανίζει στο id του imageView που έχουμε ορίσει να εμφανίζεται η εικόνα **[εικόνα 1.109]**.

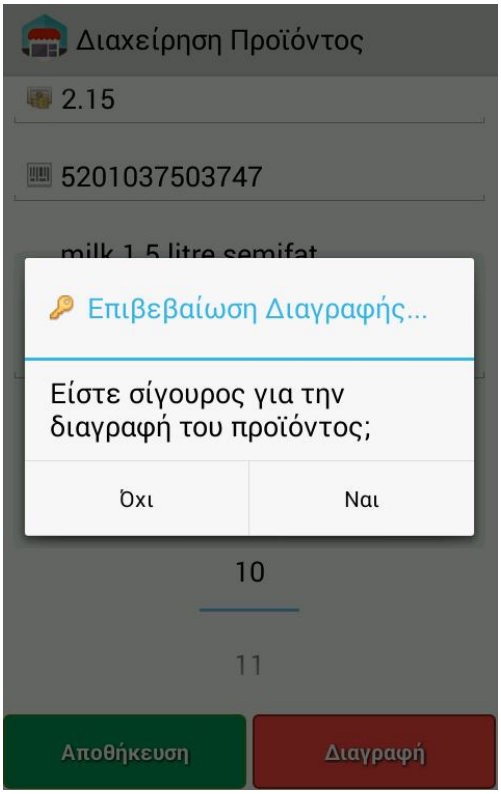
Όταν ο διαχειριστής πατήσει το button «Αποθήκευση» τότε στέλνονται οι τιμές στο PHP αρχείο που μέσω ενός Update query ενημερώνει την βάση δεδομένων και αλλάζει τις τιμές του προϊόντος.



Εικόνα 1.106: Αριστερά η όψη επεξεργασίας ενός προϊόντος και δεξιά αντίστοιχα tags του layout αρχείου

6.6.4. Διαγραφή προϊόντος

Αν ο πατήσει πατήσει το κουμπί «Διαγραφή» τότε εμφανίζεται ένα alertDialog **[Εικόνα 1.107]** για να επιβεβαιώσει τη διαγραφή του προϊόντος. Αν ο διαχειριστής πατήσει ναι τότε εκτελείται μια AsyncTask διεργασία που στέλνει το id του προϊόντος στο PHP αρχείο διαγραφής προϊόντος. Μετά το PHP αρχείο εκτελώντας ένα DELETE ερώτημα στη βάση δεδομένων διαγράφει το προϊόν από την βάση δεδομένων και έτσι δεν θα εμφανίζεται πια στη λίστα των διαθέσιμων προϊόντων.

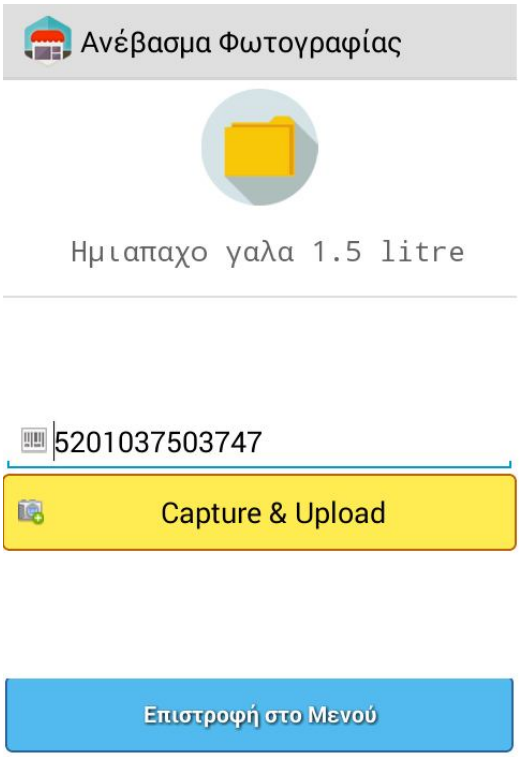


Εικόνα 1.107: AlertDialog για τη διαγραφή προϊόντος από την βάση δεδομένων

6.6.5. Αλλαγή φωτογραφίας προϊόντος

Πατώντας το κουμπί «Αλλαγή Φωτογραφίας» στην όψη της Επεξεργασίας ενός προϊόντος τότε στέλνουμε μαζί με το intent το barcode και το όνομα του προϊόντος και το εμφανίζουμε στην οθόνη σε EditText και TextView αντίστοιχα για να μπορεί να ξέρει ο διαχειριστής ποιο προϊόν να τραβήξει φωτογραφία. Πατώντας το button «Capture & Upload» ξεκινάμε ένα Intent με τη μορφή :

```
Intent camera Intent = new
Intent(android.provider.MediaStore.ACTION_IMAGE_CAPTURE);
```



Εικόνα 1.108: Όψη αλλαγής και upload φωτογραφίας για ένα προϊόν

Έτσι ανοίγει η κάμερα του κινητού μας και μπορούμε να τραβήξουμε μια φωτογραφία του προϊόντος. Όταν βγάλουμε τη φωτογραφία επιστρέφουμε πάλι πίσω στο activity και εκτελείται ο κώδικας που φαίνεται στην εικόνα **[εικόνα 1.109]** .

```
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (requestCode == CAMERA_REQUEST && resultCode == RESULT_OK) {
        Bitmap photo = (Bitmap) data.getExtras().get("data");

        imageView.setImageBitmap(photo);
        Bitmap bitmapOrg = BitmapFactory.decodeResource(getResources(),R.drawable.icon);
        ByteArrayOutputStream bao = new ByteArrayOutputStream();
        photo.compress(Bitmap.CompressFormat.JPEG, 90, bao);
        byte [] ba = bao.toByteArray();
        String ba1=Base64.encodeBytes(ba);
         ArrayList<NameValuePair> nameValuePairs = new ArrayList<NameValuePair>();
            nameValuePairs.add(new BasicNameValuePair("image",ba1));
            nameValuePairs.add(new BasicNameValuePair("bcode", bcText));


try{

            HttpClient httpclient = new DefaultHttpClient();
            HttpPost httppost = new HttpPost(ServerPath);
            httppost.setEntity((HttpEntity) new UrlEncodedFormEntity(nameValuePairs));
            HttpResponse response = httpclient.execute(httppost);
            Log.d("response",""+response);
            HttpEntity entity = response.getEntity();

            is = (InputStream) entity.getContent();


      }catch(Exception e){
            Log.e("log_tag", "Error in http connection "+e.toString());
      }

    }

}
```

Εικόνα 1.109: Κώδικας μετατροπής εικόνας και αποστολής στον webserver για επεξεργασία

```
//ΚΑΤΕΒΑΣΜΑ ΕΙΚΟΝΑΣ ΑΠΟ URL
private class DownloadImageTask extends AsyncTask<String, Void, Bitmap> {
      ImageView bmImage;

      public DownloadImageTask(ImageView bmImage) {
          this.bmImage = bmImage;
      }

      protected Bitmap doInBackground(String... urls) {
          String urldisplay = urls[0];
          Bitmap mIcon11 = null;
          try {
            InputStream in = new java.net.URL(urldisplay).openStream();
            mIcon11 = BitmapFactory.decodeStream(in);
          } catch (Exception e) {

              Log.e("Error", e.getMessage());
              e.printStackTrace();
          }

              return mIcon11;


      }

      protected void onPostExecute(Bitmap result) {

          bmImage.setImageBitmap(result);
      }
  }
```

Εικόνα 1.110: Κώδικας για κατέβασμα εικόνας από ένα url μέσω AsyncTask

6.6.6. Προσθήκη νέου προϊόντος

Οι διαχειριστές τις εφαρμογής έχουν την επιλογή να δημιουργήσουν νέα προϊόντα στο κατάστημα για να μπορέσουν αργότερα οι πελάτες του καταστήματος να πραγματοποιήσουν τις αγορές τους. Ο διαχειριστής προσθέτει τον κωδικό Barcode, το όνομα του προϊόντος, τιμή προϊόντος, μια σύντομη περιγραφή προϊόντων και την ποσότητα που έχει διαθέσιμη στην αποθήκη του καταστήματος. **[εικόνα 1.114]**

Για την επιλογή της ποσότητας δημιουργούμε ένα NumberPicker για να διευκολύνει το διαχειριστή έτσι ώστε να κάνει scroll επιλέγοντας την επιθυμητή ποσότητα. Αργότερα, μαζί με το NumberPicker παίρνουμε και τα αντίστοιχα values από τα EditTexts και τα στέλνουμε στο PHP αρχείο create_product.php **[εικόνα 1.111]** το οποίο παίρνει τα values και με ένα MySQL ερώτημα τύπου INSERT τα προσθέτει στη βάση δεδομένων.

```php
<?php
$response = array();
if (isset($_POST['name']) && isset($_POST['price']) && isset($_POST['description'])) {

    $name           = $_POST['name'];
    $price          = $_POST['price'];
    $description    = $_POST['description'];
    $bcode          = $_POST['barcode'];
    $quantity       = $_POST['quantity'];

    require_once __DIR__ . '/db_connect.php';

    // σύνδεση με τη βάση δεδομένων
    $db = new DB_CONNECT();

    $bcodeDetails = array();
    $bcodeDetails = explode("-",$db->BarcodeDetails($bcode));


        // mysql εντολή INSERT  στη βάση μας
    $result = mysql_query("INSERT INTO products(name, price, barcode, description, quantity, descriptionDetails, contactDetails)
                                        VALUES('$name', '$price','$bcode', '$description', '$quantity', '$bcodeDetails[0]', '$bcodeDetails[1]')");

    if ($result) {
        // προστέθηκε επιτυχώς στη βάση δεδομένων
        $response["success"] = 1;
        $response["message"] = "Προϊόν προστέθηκε επιτυχώς!";

        echo json_encode($response);
    } else {

        $response["success"] = 0;
        $response["message"] = "Oops! error.";


        echo json_encode($response);
    }
} else {

    $response["success"] = 0;
    $response["message"] = "Required field(s) is missing";

    echo json_encode($response);
}
?>
```

Εικόνα 1.111: PHP αρχείο για δημιουργία νέου προϊόντος στη βάση δεδομένων

6.6.7. Εξόρυξη επιπλέον δεδομένων στα προϊόντα μέσω barcodes

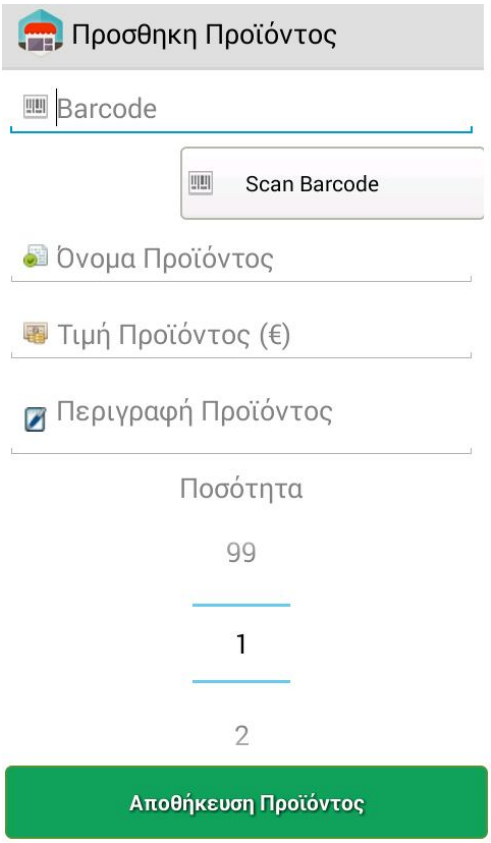
Στην διεύθυνση http://gepir.gs1.org/v32/xx/gtin.aspx υπάρχει μια υπηρεσία αντιστοίχησης των barcodes των προϊόντων με την εταιρία που είναι υπεύθυνη καθώς και στοιχεία επικοινωνίας για την εκάστοτε εταιρία των προϊόντων. Εμείς δημιουργήσαμε μια συνάρτηση σε PHP με τη χρήση CURL http request όπου στέλνει σαν παράμετρο το barcode στην φόρμα που υπάρχει στη διεύθυνση που αναφέραμε παραπάνω και παίρνουμε μέσω xml τα αποτελέσματα που μας δίνει.Έτσι στην δημιουργία ενός νέου προϊόντος στις επιλογές του διαχειριστή της εφαρμογής, εκτελούμε και αυτή τη συνάρτηση και παίρνουμε τα δεδομένα, αποθηκεύοντας τα αργότερα στη βάση δεδομένων.

Το κομμάτι αυτό της εφαρμογής δεν κρίνεται αναγκαίο αλλά σίγουρα διευκολύνει πρακτικά τον πελάτη του καταστήματος αν θέλει περισσότερες πληροφορίες για το προϊόν χωρίς να χρειάζεται να ψάχνει μόνος του αν π.χ θέλει να επικοινωνήσει με την εταιρία κάποιου προϊόντος που έχει αγοράσει. **[Εικόνα 1.112]**

Εικόνα 1.112: Επιπλέον στοιχεία για ένα προϊόν από μια εξωτερική βάση δεδομένων

```php
function BarcodeDetails($barcode){

        $agent = "Mozilla/5.0 (X11; U; Linux i686; en-US)
                AppleWebKit/532.4 (KHTML, like Gecko)
                Chrome/4.0.233.0 Safari/532.4";
    $referer = "http://www.google.com/";

    $url = 'http://gepir.gs1.org/v32/xx/gtin.aspx';
    $fields = array(
        'TabContainerGTIN:TabPanelGTIN:txtRequestGTIN'=>$barcode,
        '__EVENTARGUMENT'=>'',
        'LoginPanel_ScriptManager_HiddenField'=>
        ';;AjaxControlToolkit:en-US:c5c982cc-4942-4683-9b48-c2c58277700f:865923e8:411fea1c:e7c87f07;AjaxControlToolkit, Version=1.0.20229.20821, Culture=neutral,
        PublicKeyToken=28f01b0e84b6d53e:en-US:c5c982cc-4942-4683-9b48-c2c58277700f:865923e8:91bd373d:ad1f21ce:596d588c:8e72a662:411fea1c:acd642d2:77c58d20:14b56adc:269a19ae:d7349d0c',
        'TabContainerGTIN_ClientState' :'{"ActiveTabIndex":0,"TabState":[true]}',
        '__VIEWSTATE'=>'',
        'TabContainerGTIN:TabPanelGTIN:rblGTIN'=>'party',
        'TabContainerGTIN:TabPanelGTIN:btnSubmitGTIN'=>'Search',
                );

    foreach($fields as $key=>$value) { $fields_string .= $key.'='.$value.'&'; }
    rtrim($fields_string, '&');

    $ch = curl_init();

    curl_setopt($ch,CURLOPT_URL, $url);
    curl_setopt($ch,CURLOPT_POST, count($fields));
    curl_setopt($ch,CURLOPT_POSTFIELDS, $fields_string);

     curl_setopt($ch, CURLOPT_HTTPPROXYTUNNEL, 1);
      curl_setopt($ch, CURLOPT_USERAGENT, $agent);
      curl_setopt($ch, CURLOPT_REFERER, $referer);
      curl_setopt($ch, CURLOPT_FOLLOWLOCATION, 1);
      curl_setopt($ch, CURLOPT_MAXREDIRS, 2);


    curl_setopt ($ch, CURLOPT_RETURNTRANSFER, '1');
    curl_setopt($ch, CURLOPT_HEADER, 0);
    curl_setopt($ch, CURLOPT_ENCODING, "");

    $result = curl_exec($ch);

    include('simple_html_dom.php');


    $domd = new DOMDocument();
    libxml_use_internal_errors(true);
    $domd->loadHTML($result);
    libxml_use_internal_errors(false);
    $div = $domd->getElementById("addressscroll");          //Checking Address Details

    if ($div) {
      $dom2 = new DOMDocument();
      $dom2->appendChild($dom2->importNode($div, true));
      $AddressDetails = trim(strip_tags(preg_replace('<br>', '',$dom2->saveHTML())));
    } else {
      $AddressDetails =  "Δεν βρέθηκε.";
    }

    $div = $domd->getElementById("Contact");                //Checking Contact Details

    if ($div) {
      $dom2 = new DOMDocument();
      $dom2->appendChild($dom2->importNode($div, true));
      $ContactDetails = trim(strip_tags(preg_replace('<br>', '',$dom2->saveHTML())));
    } else {
      $ContactDetails =  "Δεν βρέθηκε.";
    }

return $AddressDetails.'-'.$ContactDetails;
}
```

Εικόνα 1.113: Συνάρτηση PHP εξόρυξης δεδομένων barcodes



Εικόνα 1.114: Όψη για προσθήκη νέου προϊόντος

7. Συμπεράσματα

Στην πτυχιακή αυτή εργασία προσπαθήσαμε να αναπτύξουμε χρησιμοποιώντας την πλατφόρμα Android μία εφαρμογή που θα μπορεί να χρησιμοποιηθεί σε ένα κατάστημα χωρίς οι πελάτες να χρειάζεται να πληρώσουν με την «φυσική» πιστωτική τους κάρτα στο σημείο πώλησης κατά την ολοκλήρωση της πληρωμής τους.

Αρχικά, εκτιμήσαμε τί λογισμικό θα χρειαστούμε για να επιτύχουμε τον σκοπό της εφαρμογής. Διαλέξαμε προγραμμάτα και βιβλιοθήκες ανοιχτού κώδικα που είναι συνιστώμενα για τη δημιουργία μιας εφαρμογής.

Προσπαθήσαμε να προσαρμόσουμε τη βάση δεδομένων MySQL που χρησιμοποιήσαμε στις ανάγκες μας χωρίς να προσθέσουμε περιττά στοιχεία και δώσαμε έστω και μια μικρή έμφαση στην ασφάλεια βάζοντας constraints και ξένα κλειδιά όπου πιστεύαμε ότι έπρεπε. Επίσης, να αναφέρουμε ότι χρησιμοποιήσαμε webserver με ένα δικό μας domain για να αποθηκεύσουμε και να πέρνουμε πληροφορίες σχετικές με τη βάση δεδομένων από τα PHP αρχεία. Πιστεύουμε ότι έτσι δοκιμάσαμε την εφαρμογή σε πιο πραγματικές συνθήκες απ'ότι απλά να είχαμε τη βάση δεδομένων τοπικά με χρήση SQLite μέσα στο smartphone.

Φυσικά, ψάξαμε για ανοιχτού κώδικα βιβλιοθήκες για την σάρωση των barcodes όπως τη βιβλιοθήκη zBar μιας και η υλοποίηση μιας δική μας βιβλιοθήκης θα ήταν αρκετά χρονοβόρα και δεν συμβαδίζει με τον σκοπό της εργασίας αυτής.

Αργότερα, έπρεπε να εντοπίσουμε και μια Android βιβλιοθήκη για την διαδικασία των πληρωμών. Το PayPal φάνηκε εξαρχής η πρώτη μας επιλογή αφού για πολλούς καταναλωτές είναι ο ασφαλέστερος τρόπος για online (και in-store) πληρωμές. Ακόμα, το PayPal βοηθάει πολύ τους developers έχοντας αναλυτικές οδηγίες στις διάφορες ιστοσελίδες του και φόρουμς που μπορούν οι developers να λύσουν απορίες μεταξύ τους. Ένας άλλος λόγος χρησιμοποίησης του PayPal είναι ότι εξελίσσεται συνεχώς και αναβαθμίζει τις υπηρεσίες του σε πολλούς τομείς βοηθώντας μας έτσι σε περίπτωση που κάποια στιγμή θέλουμε να επεκτείνουμε αυτή την εφαρμογή.

7.1. Μελλοντική Εργασία και Επεκτάσεις

Η εφαρμογή που δημιουργήσαμε θα μπορούσε να έχει πολλές επεκτάσεις. Καταρχήν, θα μπορούσαμε να την προσαρμόσουμε σε κάποιο συγκεκριμένο κατάστημα, είτε αυτό είναι κάποιο εστιατόριο, είτε μια καφετέρια όπου οι χρήστες-πελάτες θα μπορούν να πληρώνουν απ'ευθείας από το κινητό τους τηλέφωνο. Βέβαια, έτσι θα χρειαστεί να κάνουμε και αρκετές μετατροπές ανάλογα τις προτιμήσεις του εκάστοτε εμπόρου.

Ακόμα, μια ενδιαφέρουσα λειτουργία θα ήταν η προσθήκη, επεξεργασία και χρήση προσφορών και ψηφιακών κουπονιών για τους πελάτες που χρησιμοποιούν την εφαρμογή.

Επίσης, αν η εφαρμογή χρησιμοποιηθεί σε κάποια αλυσίδα καταστημάτων θα μπορούσαμε να προσθέσουμε λειτουργίες εύρεσης κοντινών καταστημάτων, όπου οι χρήστες θα μπορούν να βλέπουν τις προσφορές για κάθε κατάστημα και να επωφεληθούν απ'αυτές.

Θα μπορούσε δηλαδή, η εφαρμογή μας να έχει τη δυνατότητα λειτουργίας μιας μικρής κοινότητας όπου σε αυτό το δίκτυο του καταστήματος να μπορούν να έχουν περισσότερες επιλογές οι χρήστες, όπως να βαθμολογήσουν και να ασκήσουν κριτική σε διάφορα προϊόντα ή ακόμα και να δουν δημοφιλή-προτεινόμενα προϊόντα που αγοράστηκαν περισσότερο από τους πελάτες του καταστήματος.

Για την ανάρτηση της εφαρμογής στο Google Play Store, θα μπορούσαμε να κάνουμε κάποιες διαβαθμίσεις στις εφαρμογή μας, κάνοντας την διαθέσιμη για ότι κατάστημα έχει ο

έμπορος αλλάζοντας έτσι τις ρυθμίσεις ανάλογα με τις απαιτήσεις του διαχειριστή. Δηλαδή, αν την χρησιμοποιήσει ένας ιδιοκτήτης μιας καφετέριας δεν θα θέλει να περιέχονται barcodes αφού δεν χρειάζεται, σε αντίθεση με ένα κατάστημα ρούχων ή σούπερμαρκετ που η εφαρμογή μας όπως είναι τώρα είναι πιο προσαρμοσμένη για να λειτουργεί κάτω από αυτές τις συνθήκες.

Βιβλιογραφία

[1] Hypertext Transfer Protocol -- HTTP/1.1. "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Available: http://www.ietf.org/rfc/rfc2616.txt

[2] Android Tutorials. Available: http://www.androidhive.info/

[3] Android SDK and documentation for app developers and designers. Available: http://developer.android.com/

[4] Android version history http://en.wikipedia.org/wiki/Android_version_history

[5] PayPal Developer Portal. // REST APIs. // Native SDKs. // Built for developers Available: https://developer.paypal.com/

[6] Eclipse - The Eclipse Foundation open source community website. Available: http://www.eclipse.org/

[7] Vibes report from August 2013. Mobile Wallet Consumer Report. Available : http://client.vibes.com/references/MobileWallet_ConsumerReport.pdf

[8] Study in the future of mobile wallets and digital coupons. Available: http://venturebeat.com/2013/08/22/the-future-of-mobile-wallets-is-digital-coupons-study/

[9] JavaScript Object Notation. Available: http://en.wikipedia.org/wiki/JSON

[10] Android ADT Plugin http://developer.android.com/tools/sdk/eclipse-adt.html

[11] Mobile payment(s). Available: http://en.wikipedia.org/wiki/Mobile_payment

[12] Support in the area of Eclipse, Java and Android development and well as the Git version control system. Available: http://www.vogella.com/

[13] Eclipse IDE Tutorial Available: http://www.vogella.com/tutorials/Eclipse/article.html

[14] Android Persistence with preferences and files - Tutorial http://www.vogella.com/tutorials/AndroidFileBasedPersistence/article.html

[15] Mobile Payment Libraries Getting Started Guide – Paypal Android Library. https://developer.paypal.com/webapps/developer/docs/classic/mobile/gs_MPL/

[16] Android ListView. http://www.vogella.com/tutorials/AndroidListView/article.html

[17] Android Intents Tutorial. Available: http://www.vogella.com/tutorials/AndroidIntent/article.html

[18] Global mobile statistics 2012 Section F: Mobile payment, NFC, m-commerce, m-ticketing and m-coupons. Available: http://mobithinking.com/mobile-marketing-tools/latest-mobile-stats/f

[19] Android Background Processing with Handlers and AsyncTask and Loaders – Tutorial. Available: http://www.vogella.com/tutorials/AndroidBackgroundProcessing/article.html

ΠΕΡΙΛΗΨΗ Πτυχιακής Εργασίας σε στυλ δημοσίευσης

Τμήμα Μηχανικών Πληροφορικής

Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Τίτλος Πτυχιακής Εργασίας

Υλοποίηση εφαρμογής για αγορά προϊόντων από κατάστημα με τη χρήση κινητού τηλεφώνου τύπου Android

Φοιτητές:

Βογιατζάκης Παναγιώτης ΑΜ:1842

Πατεράκης Μηνάς ΑΜ:2010

Εισηγητής:

Παναγιωτάκης Σπύρος

Εισαγωγή

Το αντικείμενο της πτυχιακής αυτής εργασίας είναι η υλοποίηση μιας εφαρμογής βασισμένης στο μοντέλο του κινητού πορτοφολιού, δηλαδή να μπορούν οι χρήστες να αγοράζουν προϊόντα μέσω σάρωσης barcodes με το κινητό τους τηλέφωνο σαν να βρίσκονται σε ένα κατάστημα. Αργότερα θα μπορούν να προσθέτουν στο καλάθι τα προϊόντα που σαρώνουν τα οποία με τη σειρά τους θα καταχωρούνται σε μια βάση δεδομένων που θα βρίσκεται σε online περιοχή-server. Τέλος, αφού ο χρήστης θέλει να πληρώσει θα του παρέχεται η επιλογή να πραγματοποιήση την πληρωμή με τη χρήση του PayPal. Η εφαρμογή θα βασίζεται στο Android λειτουργικό σύστημα.

Περίληψη

Σε αυτή την πτυχιακή εργασία περιγράφεται αρχικά η πλατφόρμα Android. Δίνονται πληροφορίες για την ιστορία της, την εξέλιξή της, τα χαρακτηριστικά της, τις λειτουργίες της και τις δυνατότητες της.

Αργότερα, θα περιγράψουμε την τεχνολογία του κινητού πορτοφολιού (digital wallet) πάνω στην οποία βασίζεται και η εφαρμογή μας. Θα μάθουμε πως ξεκίνησαν, σε ποιο σημείο βρίσκονται τώρα και το σημαντικότερο, το μέλλον των κινητών πορτοφολιών καθώς και τους σημαντικότερους παρόχους αυτής της σύγχρονης υπηρεσίας.

Ανοίγοντας την εφαρμογή βλέπουμε το λογότυπό της και σέρνοντας το βελάκι προς τα επάνω εμφανίζονται τα πεδία του ονόματος (username) και του κωδικού (password). Ανάλογα με το ποιος τα συμπληρώνει εμφανίζει και το κατάλληλο μενού στη συνέχεια.

Αν τα πεδία τα συμπληρώσει ο διαχειριστής (admin) της εφαρμογής, με το κατάλληλο όνομα και κωδικό, θα εμφανιστεί ένα μενού από το οποίο θα μπορεί να προσθέσει ή να αφαιρέσει ένα προϊόν, να αλλάξει το όνομα, τον κωδικό και την περιγραφή των προϊόντων και γενικότερα να διαχειριστεί όπως θέλει τα προϊόντα. Επίσης υπάρχει και ένα κουμπί που θα μπορεί να αποσυνδεθεί από το λογαριασμό του (logout).

Αν τα πεδία τα συμπληρώσει ένας χρήστης (user) της εφαρμογής θα εμφανιστεί ένα μενού από το οποίο θα μπορεί να πληροφορηθεί για τη λειτουργία της εφαρμογής και να κατατοπιστεί βήμα-βήμα τι μπορεί να κάνει με την εφαρμογή που έχει στα χέρια του, να αλλάξει κάποια χαρακτηριστικά του προφίλ του, να δει το ιστορικό των αγορών του, να σκανάρει το κωδικό από το προϊόν που επιθυμεί και να το προσθέσει στο καλάθι, να μπει στο καλάθι με τα προϊόντα που ήδη έχει προσθέσει από όπου θα μπορεί να προχωρήσει στην αγορά τους με κάποιο ηλεκτρονικό λογαριασμό και τέλος και εδώ υπάρχει κουμπί που αποσυνδέεται (logout).

Δηλαδή, η εφαρμογή θα χωρίζεται στο μενού του χρήστη-πελάτη και στο μενού του διαχειριστή(admin) ανάλογα με το όνομα χρήστη και τον κωδικό χρήστη που θα πληκτρολογίσει ο χρήστης. Στο μενού του χρήστη θα υπάρχουν επιλογές όπως Ιστορικό για την προβολή των προηγούμενων παραγγελιών που έχει κάνει ο χρήστης. Με την επιλογή Σαρωση στο μενού του χρήστη-πελάτη θα ανοίγει η κάμερα του κινητού τηλεφώνου όπου θα μπορεί να σαρώσει το barcode ενός φυσικού προϊόντος και αν το barcode αυτό βρεθεί στην βάση δεδομένων θα προστεθεί στο καλάθι του όπου θα μείνει μέχρι ή να το διαγράψει ή όταν θα ολοκληρώσει τη διαδικασία της πληρωμής.

Αφότου ο χρήστης-πελάτης θέλει να ολοκληρώσεις τις αγορές του, πατώντας ένα κουμπί θα μεταφέρεται στην όψη του PayPal όπου θα φαίνεται η παραγγελία που έχει στο καλάθι αγορών του.

Mobile Wallets

Το mobile wallet ή αλλιώς κινητό πορτοφόλι είναι ένα εικονικό πορτοφόλι που αποθηκεύει διάφορα στοιχεία πληρωμών που μας επιτρέπουν να κάνουμε online αγορές ή αγορές (μέσα) σε ένα κατάστημα χωρίς να χρησιμοποιήσουμε την «φυσική» πιστωτική μας κάρτα.

Παρότι διαφορετικά mobile wallets έχουν βγει στην αγορά τα τελευταία χρόνια, πολλά απ'αυτά βρίσκονται στο αρχικό στάδιο της λειτουργίας τους. Αλλά αυτό δεν σημαίνει ότι δεν υπάρχουν ήδη αρκετές υπηρεσίες που μπορούν οι χρήστες τους να επωφεληθούν. Αρχικά, ενώ η διείσδυση των mobile wallets ήταν επικεντρωμένη στη διανομή κουπονιών-προσφορών και εκπτωτικών καρτών, τώρα τελευταία τα mobile wallet παρουσιάζουν ποικίλες δυνατότητες εκτός των προαναφερθέντων όπως δυνατότητες για διαχείριση οικονομικών υπηρεσιών, επιλογές πληρωμών και mobile banking.

Ας αναφέρουμε μερικά βασικά χαρακτηριστικά ενός mobile wallet:

- Διαχείριση από τον καταναλωτή ενός διευρυμένου χαρτοφυλακίου των κινητών υπηρεσιών πληρωμών από διαφορετικούς παρόχους
- Διευκόλυνση των πληρωμών (επιλογή και έλεγχος ταυτότητας) για τα αγαθά, τις υπηρεσίες ή τις πληρωμές πρόσωπο με πρόσωπο.
- Αποθήκευση των εισιτηρίων, κάρτες επιβίβασης που μπορεί να παρουσιαστούν σε ένα σημείο ελέγχου.
- Προσφορά ενιαίου χώρου αποθήκευσης για προγράμματα πιστότητας και κουπόνια.
- Αποθήκευση διαπιστευτηρίων για την εύκολη και βολική αναγνώριση και ταυτοποίηση σε κάποιο σημείο πληρωμής.
- Αποθήκευση προσωπικών πληροφοριών, όπως τη διεύθυνση παράδοσης για να διευκολυνθούν οι online αγορές ενός καταναλωτή.

Το PayPal

Το PayPal επιτρέπει σε κάθε επιχείρηση ή άτομο με μια διεύθυνση ηλεκτρονικού ταχυδρομείου την ασφαλή , εύκολη και αποτελεσματική (από άποψη κόστους) λήψη και αποστολή πληρωμών. Το δίκτυό της PayPal βασίζεται στην υπάρχουσα οικονομική υποδομή των τραπεζικών λογαριασμών και των πιστωτικών καρτών με σκοπό να δημιουργήσει μια παγκόσμια λύση πληρωμών σε πραγματικό χρόνο . Το PayPal χρησιμεύει ως μια ηλεκτρονική εναλλακτική λύση στις παραδοσιακές μεθόδους, όπως οι επιταγές και οι εντολές πληρωμών.

Ένας λογαριασμός PayPal (που συνδέεται πάντα με μια διεύθυνση e-mail) μπορεί να χρηματοδοτηθεί με ηλεκτρονική πίστωση από ένα τραπεζικό λογαριασμό ή από μια πιστωτική κάρτα. Το PayPal είναι ένα παράδειγμα μιας πληρωμής σε υπηρεσίες διαμεσολαβήσεως, που διευκολύνει τον κόσμο στις καθημερινές διαδικτυακές συναλλαγές του.

Περιβάλλον Σχεδίασης – Λογισμικό

Χρησιμοποιήσαμε λειτουργικό σύστημα Windows 7 Professional Edition 64 bit και οι συσκευές που χρησιμοποιήσαμε για την υλοποίηση αλλά και τις δοκιμές της εφαρμογής μας ήταν:

- Ηλεκτρονικός Υπολογιστής με χρήση Android εξομοιωτή
- Δύο Android smartphone συσκεύες για καλύτερο έλεγχο συμβατότητας.

Το λογισμικό που χρησιμοποιήσαμε:

- MySQL

- PhpMyAdmin
- Wamp Server x64 version 2.2
- Eclipse 3.7.2
- GitBash (για τη συνεργασία στον κώδικα μας)
- Aptana Studio 3, build: 3.2.2.201208201020 (γιατον PHP κώδικα)

Οι γλώσσες προγραμματισμού που χρειαστήκαμε για την υλοποίηση της εφαρμογής ήταν:

- Java (Android)
- PHP
- SQL

Βιβλιοθήκη ZBar

Η βιβλιοθήκη ZBar είναι μια βιβλιοθήκη ανοιχτού κώδικα που χρησιμεύει στην ανάγνωση barcodes και υποστηρίζει διάφορες πηγές όπως ροές βίντεο, αρχεία εικόνων και αισθητήρες. Η ZBar υποστηρίζει πολλές συμβολογίες (τύπος Barcode) όπως EAN-13/UPC-A, UPC-E, EAN-8, Code 128, Code 39 και QR Code.

Η βιβλιοθήκη Mobile PayPal Library

Η βιβλιοθήκη Mobile PayPal Library (MPL) παρέχει μια ασφαλή και εκτενής λειτουργία στη δυνατότητα των Android (αλλά και iOS) εφαρμογών να προσθέσουν το μοντέλο πληρωμών που διαθέτει το PayPal. Με αυτή τη βιβλιοθήκη οι πελάτες-χρήστες της εφαρμογής ολοκληρώνουν τις συναλλαγές τους σε μία εφαρμογή με το πάτημα ενός κουμπιού, πολύ εύκολα χωρίς να χρειαστεί να μεταφερθούν σε άλλο παράθυρο κάνοντας έτσι την ροή των πληρωμών πολύ προσιτή και ενιαία.

Δημιουργία της βάσης δεδομένων

Στην καρτέλα Databases (Βάσεις Δεδομένων) ξεκινάμε την δημιουργία της βάσης δεδομένων μας. Της δίνουμε ένα όνομα, “ptyx” στην δική μας περίπτωση (λόγω του ότι ανήκει στην πτυχιακή μας εργασία) και επιλέγουμε utf8_unicode_ci για την κωδικοποίηση των χαρακτήρων (character encoding) που θα έχει η βάση δεδομένων μας έτσι ώστε να μην παρουσιαστεί πρόβλημα κατά την εκχώρηση ελληνικών χαρακτήρων στην βάση δεδομένων μας και τους πίνακες που θα έχει. Οταν έχουμε σιγουρευτεί για την επιλογή μας πατάμε το κουμπί Create για να δημιουργηθεί η βάση δεδομένων μας. Ας προσδιορίσουμε τους πίνακες που χρησιμοποιήθηκαν στη δημιουργία της βάσης δεδομένων:

- **auth_table** : Ο πίνακας auth_table καταχωρεί και επεξεργάζεται τους χρήστες της εφαρμογής μας.
- **basket** : Ο πίνακας basket χρησιμεύει για να αποθηκεύονται τα προϊόντα που έχει ο εκάστοτε χρήστης στο καλάθι του. Τα προϊόντα αυτά μένουν σε αυτόν τον πίνακα μέχρι ο χρήστης-πελάτης πραγματοποιήσει την πληρωμή. Αφότου η πληρωμή του πραγματοποιηθεί η τιμή paid γίνεται 1 (από την αρχική τιμή 0). Δηλαδή το εκάστοτε προϊόν θεωρείται πληρωμένο και δεν εμφανίζεται στην οθόνη καλαθιού του χρήστη-πελάτη.
- **history** : Ο πίνακας history (ιστορικό) μας χρησιμεύει για να αποθηκεύουμε το ιστορικό των παραγγελιών του χρήστη-πελάτη. Οι εκχωρήσεις σε αυτόν τον πίνακα αρχίζουν όταν τα προϊόντα που βρίσκονται στο καλάθι του χρήστη έχουν επιβεβαιωθεί ότι είναι πληρωμένα μέσω της συναλλαγής μέσω PayPal. Δηλαδή μόνο όταν το PayPal επιβεβαιώσει ότι τα προϊόντα που ήταν στο καλάθι του χρήστη έχουν πληρωθεί.
- **products** : Στον πίνακα products καταχωρούνται τα προϊόντα της εφαρμογής μας. Κάθεπροϊόνπεριλαμβάνει στήλες όπως όνομα, περιγραφή, τεμάχια που βρίσκονται σε διαθεσιμότητα και φυσικά το barcode του προϊόντος.

- **paypal** : Στον πίνακα PayPal καταχωρούνται οι συναλλαγές που έχουν γίνει μέσω PayPal και αποθηκεύονται τα στοιχεία – μεταβλητές που «επιστρέφει» το PayPal όταν ολοκληρωθεί η διαδικασία μια πληρωμής του χρήστη.

Αρχική όψη εφαρμογής

Η αρχική όψη της εφαρμογής μας έχει ένα Sliding Drawer το οποίο επιτρέπει στο χρήστη να σύρει μέσω μιας λαβής (στην περίπτωσή μας είναι ένα μαύρο βέλος που δείχνει προς τα πάνω) και μόλις τελειώσει το σύρσιμο φέρνει ένα νέο περιεχόμενο στην οθόνη.

Εικόνα: Αρχική όψη εφαρμογής

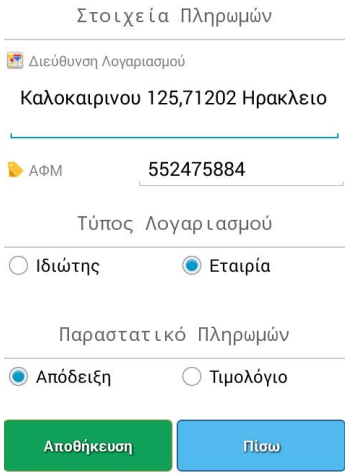
Κεντρικό μενού

Αφού γίνει η σύνδεση του χρήστη στην εφαρμογή εμφανίζεται το κεντρικό μενού που έχει 6 επιλογές όπως Ρυθμίσεις, Βοήθεια, Ιστορικό, Καλάθι Αγορών, Σάρωση, Αποσύνδεση. Κάθε επιλογή είναι ένα εικονίδιο-κουμπί στο layout που έχουμε στον φάκελο /res του project μας. Κάθε φορά που ο χρήστης πατάει μια επιλογή-κουμπί στο μενού έχουμε ορίσει αντίστοιχους listeners που μεταφέρουν το χρήστη στην επόμενη οθόνη, δηλαδή στο επόμενο Activity. Αν έχει συνδεθεί ο διαχειριστής της εφαρμογής δηλαδή κάποιος χρήστης που στον πίνακα της βάσης δεδομένων έχει userlevel διαφορετικό του 1 τότε πατώντας το ίδιο κουμπί έχουμε ορίσει να πηγαίνει σε άλλο Activity.

Εικόνα: Κεντρικό μενού εφαρμογής

Ρυθμίσεις Πληρωμών

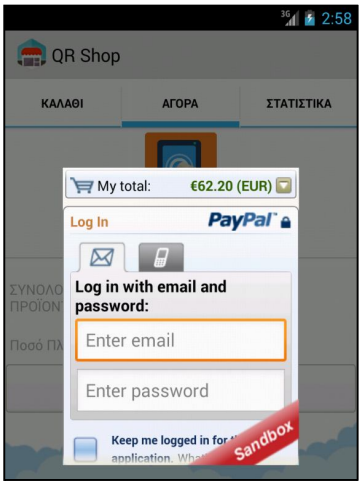
Σε αυτή την οθόνη ο χρήστης αλλάζει τα στοιχεία που αφορούν τις εικονικές πληρωμές που πραγματοποιεί, για να υπάρχει καλύτερος έλεγχος στις αποδείξεις του εικονικού καταστήματος. Για την εμφάνιση στην οθόνη του Τύπου Λογαριασμού και το Παραστατικό Πληρωμών χρησιμοποιήσαμε RadioGroups, το οποίο εμπεριέχει δύο RadioButtons



Εικόνα: Ρυθμίσεις πληρωμών χρήστη

Ενσωμάτωση PayPal στην εφαρμογή

Αφού εισάγουμε την βιβλιοθήκη για mobile συναλλαγές μέσω PayPal πρέπει να προσθέσουμε και τον κώδικα που θα συνδέεται με το PayPal στην αντίστοιχη Activity. Καταρχήν να αναφέρουμε ότι το PayPal μας επιτρέπει να έχουμε ένα δοκιμαστικό περιβάλλον (Sandbox Environment) που χρησιμεύει για προγραμματιστές έτσι ώστε να κάνουν δοκιμές στις εφαρμογές που δημιουργούν. Το Live Environment είναι το κανονικό περιβάλλον όπου όλα τα στοιχεία πρέπει να είναι τα αληθινά.
Στο Sandbox περιβάλλον οι προγραμματιστές δημιουργούν εικονικούς λογαριασμούς με εικονικές πιστωτικές κάρτες που βάζουν εκείνοι οποιοδήποτε ποσό μέσα στις κάρτες αυτές για να κάνουν δοκιμές στην εφαρμογή τους εξομοιώνοντας έτσι την διαδικασία μια πληρωμής.
Πηγαίνουμε λοιπόν στη διεύθυνση https://developer.paypal.com/ και δημιουργούμε λογαριασμό για να μπούμε μέσα στο μενού. Όταν μπούμε στο μενού πηγαίνουμε στο link sandbox accounts και μετά πατάμε create new για να φτιάξουμε ένα (εικονικό) merchant λογαριασμό, δηλαδή το λογαριασμό του παραλήπτη της πληρωμής του καταστήματος που θα μας έρχονται τα χρήματα από τις συναλλαγές με τους πελάτες μας. Έτσι, δημιουργήσαμε το λογαριασμό για τον έμπορο-πωλητή ο οποίος είναι τύπου Business-Pro (αυτό χρειάζεται για να πούμε στο PayPal ότι αυτός ο λογαριασμός θα είναι merchant).
Αργότερα δημιουργούμε και έναν (εικονικό) λογαριασμό τύπου Personal ο οποίος θα είναι ο υποτιθέμενος-εικονικός πελάτης για τις πληρωμές που θα κάνουμε. Στον λογαριασμό του πελάτη βάζουμε μια εικονική πιστωτική κάρτα(Credit card type) με ένα εικονικό ποσό (PayPal Balance) που έχουμε μέσα στην κάρτα, για να μπορέσουμε έτσι να κάνουμε πληρωμές με το λογαριασμό αυτό.



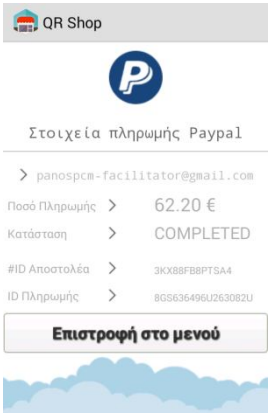
Εικόνα: Αρχική όψη PayPal μέσα στην εφαρμογή

Για να ελέγξουμε αν τα προϊόντα που έχουμε στο καλάθι μας είναι ίδια με αυτά που δηλώθηκαν στο PayPal πατάμε το βελάκι στα δεξιά της όψης του PayPal και έτσι εμφανίζεται η λίστα με τα προϊόντα μας και τα στοιχεία που χρειαζόμαστε δηλαδή όνομα, τιμή, ποσότητα και υποσύνολο για το εκάστοτε προϊόν.

Αργότερα, το Paypal επιβεβαιώνει τα στοιχεία που δώσαμε για την συναλλαγή μια τελευταία φορά και αφού το κάνει πηγαίνει πίσω στην εφαρμογή μας και πιο συγκεκριμένα στην onActivityResult (που εκτελείται όταν γίνει exit μια Activity) και στέλνει την παράμετρο RESULT_OK αν δεν υπήρξαν σφάλματα ή οποιοδήποτε error στην διαδικασία αυτή.

Όταν ολοκληρωθεί η πληρωμή, στην βάση δεδομένων τα προϊόντα στο καλάθι του χρήστη στον πίνακα basket γίνονται paid = 1 με ένα update query. Αργότερα με ένα insert query για κάθε προϊόν μπαίνουν τα προϊόντα στον πίνακα history για να μπορεί αργότερα ο χρήστης-πελάτης να δει μια προηγούμενη παραγγελία που έχει κάνει. Επίσης

κάνουμε και ένα insert query στον πίνακα paypal με το μοναδικό paykey που επέστρεψε το paypal στη συναλλαγή που έγινε αποθηκεύοντας έτσι τα στοιχεία για να υπάρχουν όταν ο χρήστης-πελάτης επιλέξει να τα δει.



Εικόνα: Η οθόνη εμφάνισης τα στοιχεία της πληρωμής που έγινε μέσω PayPal

Ο διαχειριστής του καταστήματος της εφαρμογής

Διαχειριστής στην εφαρμογή μας είναι ή ο έμπορος ή κάποιοι υπάλληλοι που θα έχει το υποτιθέμενο κατάστημα. Σαν οντότητα στην εφαρμογή ο διαχειριστής έχει userlevel = 2 στην βάση δεδομένων και συγκεκριμένα στον πίνακα auth_table που ελέγχεται κατά την είσοδος (login) ενός χρήστη στην εφαρμογή.

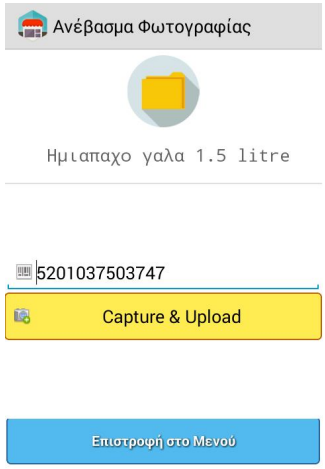
Το μενού του διαχειριστή διαφέρει μιας και ο διαχειριστής του καταστήματος δεν χρειάζεται να έχει καλάθι αγορών αφού το μόνο που θέλει να κάνει είναι να μπορεί εύκολα να προσθέσει και να διαχειρίζεται τα προϊόντα που έχει η αποθήκη του καταστήματος. Στο μενού του διαχειριστή υπάρχουν οι επιλογές για προβολή όλων των προϊόντων (Τα Προϊόντα) , επιλογή προσθήκης προϊόντος (Προσθέστε Προϊόν) και επιλογή ανεβάσματος φωτογραφίας και αντιστοίχησης της σε ένα προϊόν (Photo->Server).

Στην επεξεργασία ενός προϊόντος ο διαχειριστής μπορεί να επεξεργαστεί τις ρυθμίσεις ενός προϊόντος. Δηλαδή, μπορεί να αλλάξει ή να προσθέσει μια φωτογραφία, να αλλάξει το όνομα του προϊόντος, την τιμή του, το barcode, την περιγραφή του και τέλος να προσθέσει τεμάχια για αυτό το προϊόν.

Πατώντας το κουμπί «Αλλαγή Φωτογραφίας» στην όψη της Επεξεργασίας ενός προϊόντος τότε στέλνουμε μαζί με το intent το barcode και το όνομα του προϊόντος και το εμφανίζουμε στην οθόνη σε EditText και TextView αντίστοιχα για να μπορεί να ξέρει ο διαχειριστής ποιο προϊόν να τραβήξει φωτογραφία. Πατώντας το button «Capture & Upload» ξεκινάμε ένα Intent με τη μορφή :

```
Intent camera Intent = new
Intent(android.provider.MediaStore.ACTI
ON_IMAGE_CAPTURE);
```

Έτσι ανοίγει η κάμερα του κινητού μας και μπορούμε να τραβήξουμε μια φωτογραφία του προϊόντος.



Όψη αλλαγής και upload φωτογραφίας για ένα προϊόν

Συμπεράσματα

Στην πτυχιακή αυτή εργασία προσπαθήσαμε να αναπτύξουμε χρησιμοποιώντας την πλατφόρμα Android μία εφαρμογή που θα μπορεί να χρησιμοποιηθεί σε ένα κατάστημα χωρίς οι πελάτες να χρειάζεται να πληρώσουν με την «φυσική» πιστωτική τους κάρτα στο σημείο πώλησης κατά την ολοκλήρωση της πληρωμής τους.

Αρχικά, εκτιμήσαμε τί λογισμικό θα χρειαστούμε για να επιτύχουμε τον σκοπό της εφαρμογής. Διαλέξαμε προγραμμάτα που είναι δωρεάν και συνιστώμενα για τη δημιουργία μιας εφαρμογής.

Προσπαθήσαμε να προσαρμόσουμε τη βάση δεδομένων MySQL που χρησιμοποιήσαμε στις ανάγκες μας χωρίς να προσθέσουμε περιττά στοιχεία και δώσαμε έστω και μια μικρή έμφαση στην ασφάλεια βάζοντας constraints και ξένα κλειδιά όπου πιστεύαμε ότι έπρεπε. Επίσης, να αναφέρουμε ότι χρησιμοποιήσαμε webserver με ένα δικό μας domain για να αποθηκεύσουμε και να πέρνουμε πληροφορίες σχετικές με τη βάση δεδομένων από τα PHP αρχεία.

Φυσικά, ψάξαμε για ανοιχτού κώδικα βιβλιοθήκες για την σάρωση των barcodes όπως τη βιβλιοθήκη zBar μιας και η υλοποίηση μιας δική μας βιβλιοθήκης θα ήταν αρκετά χρονοβόρα και δεν συμβαδίζει με τον σκοπό της εργασίας αυτής.

Αργότερα, έπρεπε να εντοπίσουμε και μια Android βιβλιοθήκη για την διαδικασία των πληρωμών. Το PayPal φάνηκε εξαρχής η πρώτη μας επιλογή αφού για πολλούς καταναλωτές είναι ο ασφαλέστερος τρόπος για online (και in-store) πληρωμές. Ακόμα, το PayPal βοηθάει πολύ τους developers έχοντας αναλυτικές οδηγίες στις διάφορες ιστοσελίδες του και φόρουμς που μπορούν οι developers να λύσουν απορίες μεταξύ τους. Ένας άλλος λόγος χρησιμοποίησης του PayPal είναι ότι εξελίσσεται συνεχώς και αναβαθμίζει τις υπηρεσίες του σε πολλούς τομείς βοηθώντας μας έτσι σε περίπτωση που κάποια στιγμή θέλουμε να επεκτείνουμε αυτή την εφαρμογή.

Μελλοντική Εργασία και Επεκτάσεις

Η εφαρμογή που δημιουργήσαμε θα μπορούσε να έχει πολλές επεκτάσεις. Καταρχήν, θα μπορούσαμε να την προσαρμόσουμε σε κάποιο συγκεκριμένο κατάστημα, είτε αυτό είναι κάποιο εστιατόριο, είτε μια καφετέρια όπου οι χρήστες-πελάτες θα μπορούν να πληρώνουν απ'ευθείας από το κινητό τους τηλέφωνο. Βέβαια, έτσι θα χρειαστεί να κάνουμε και αρκετές μετατροπές ανάλογα τις προτιμήσεις του εκάστοτε εμπόρου.

Ακόμα, μια ενδιαφέρουσα λειτουργία θα ήταν η προσθήκη, επεξεργασία και χρήση προσφορών και ψηφιακών κουπονιών για τους πελάτες που χρησιμοποιούν την εφαρμογή.

Επίσης, αν η εφαρμογή χρησιμοποιηθεί σε κάποια αλυσίδα καταστημάτων θα μπορούσαμε να προσθέσουμε λειτουργίες εύρεσης κοντινών καταστημάτων, όπου οι χρήστες θα μπορούν να βλέπουν τις προσφορές για κάθε κατάστημα και να επωφεληθούν απ'αυτές.

Θα μπορούσε δηλαδή, η εφαρμογή μας να έχει τη δυνατότητα λειτουργίας μιας μικρής κοινότητας όπου σε αυτό το δίκτυο του καταστήματος να μπορούν να έχουν περισσότερες επιλογές οι χρήστες, όπως να βαθμολογήσουν και να ασκήσουν κριτική σε διάφορα προϊόντα ή ακόμα και να δουν δημοφιλή-προτεινόμενα προϊόντα που αγοράστηκαν περισσότερο από τους πελάτες του καταστήματος.

Για την ανάρτηση της εφαρμογής στο Google Play Store, θα μπορούσαμε να κάνουμε κάποιες διαβαθμίσεις στις εφαρμογή μας, κάνοντας την διαθέσιμη για ότι κατάστημα έχει ο έμπορος αλλάζοντας έτσι τις ρυθμίσεις ανάλογα με τις απαιτήσεις του διαχειριστή. Δηλαδή, αν την χρησιμοποιήσει ένας ιδιοκτήτης μιας καφετέριας δεν θα θέλει να περιέχονται barcodes αφού δεν χρειάζεται, σε αντίθεση με ένα κατάστημα ρούχων ή σούπερμαρκετ που η εφαρμογή μας όπως είναι τώρα είναι πιο προσαρμοσμένη για

να λειτουργεί κάτω από αυτές τις συνθήκες.