ΕΛΛΗΝΙΚΟ ΜΕΣΟΓΕΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ

Τμήμα Μηχανικών Πληροφορικής

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Τίτλος: Εφαρμογή Android για την πώληση αγροτικών προϊόντων

(Android app for selling agriculture products)

Σπουδαστές

ΖΩΤΟΥ ΑΝΤΩΝΙΑ (ΑΜ: 3458)

Επιβλέπων εκπαιδευτικός: Παπαδάκης Νικόλαος

Ημερομηνία παρουσίασης: Ιούνιος, 2019

Πίνακας Περιεχομένων

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον κ. Παπαδάκη Νικόλαο και τον Παπαθεοδώρου Νικόλαο Τσαμπίκο, για την καθοδήγηση που μου προσέφεραν σε όλη την έρευνα και την εφαρμογή της διατριβής μου.

Abstract

The application that was implemented provides services for the purchase of agricultural products. The user will have the possibility after been registered to be able to purchase a product, to search for a product or to see sorted some products according to their category. The purchase

will be made through the selection of a product and then the user's shipping and communication information should be given. In addition, the user will be able to change his/her account details through the "Settings" option. Then after the user "makes" an order the administrator will have to give approval that the order was accepted and thus send the package to the user at the address that he has given.

Σύνοψη

Η εφαρμογή που υλοποιήθηκε παρέχει υπηρεσίες αγοράς αγροτικών προϊόντων. Ο χρήστης θα έχει την δυνατότητα αφού έχει εγγραφεί να μπορεί να αγοράσει κάποιο προϊόν, να αναζητήσει ένα προϊόν η να δει ταξινομημένα κάποια προϊόντα ανάλογα με την κατηγορία τους. Η αγορά θα

γίνετε μέσω της επιλογής ενός προϊόντος και στην συνέχεια θα πρέπει ο δοθούν τα στοιχεία αποστολής και επικοινωνίας του χρήστη. Επιπλέον ο χρήστης θα μπορεί να αλλάξει τα στοιχεία του λογαριασμού του, μέσω της επιλογής «ρυθμίσεις». Στην συνέχεια αφού ο χρήστης «κάνει» μια παραγγελία ο διαχειριστής θα πρέπει να δώσει έγκριση ότι η παραγγελία έγινε αποδεκτή και έτσι να σταλεί το πακέτο στον χρήστη στην διεύθυνσή που αυτός έχει δώσει.

1. Εισαγωγή

1.1 Περίληψη

Η εφαρμογή που ανέπτυξα παρέχει υπηρεσίες αγοράς αγροτικών προϊόντων σε οποιοδήποτε χρήστη που επιθυμεί να αγοράσει ένα αγροτικό προϊόν όπως για παράδειγμα ένα φτυάρι μέσω του κινητού του τηλεφώνου το οποίο προϋποθέτει να έχει λογισμικό Android. Θα υπάρχει η δυνατότητα δημιουργίας προσωπικού λογαριασμού και η αλλαγή τον στοιχείων του. Στην συνέχεια στον χρήστη θα παρουσιάζεται μια λίστα με τα προϊόντα της εφαρμογή. Επιπλέον ο χρήστης έχει την δυνατότητα να αναζήτησή και να «εμφανίσει» ανά κατηγορία τα προϊόντα που επιθυμεί. Τέλος επιλέγει τα προϊόντα που θέλει να αγοράσει, συμπληρώνει την φόρμα με τα στοιχεία της διεύθυνσης αποστολής και ολοκληρώνει την παραγγελία η οποία θα πρέπει να επιβεβαιωθεί από έναν διαχειριστή της εφαρμογής για την αποστολή της.

1.2 Λόγοι πραγματοποίησης και στόχοι της Εργασίας

Αρχικά, ο λόγος που επίλεξα το συγκεκριμένο θέμα είναι ο ανθρωπος καθημερινα έχει πολλες υποχρεωσεις και ο χρόνος που του μένει είναι περιορισμένος. Ο χρόνος που ένα άτομο διαθέτει για να πάει σε ένα μαγαζι θεωρείται νεκρος και θα πρεπεπι να μην είναι υπερβολικος διοτι υπαρχουν καθυστερησεις στο υπολοιπο προγραμμα του καθως και τυχον μη ολοκληρωσει μια υποχρεωσης του. Αυτό οδηγεί σε στρες στην καθημερινή του ζωή ως συνέπεια της σύγκρουσης με τον εαυτό του ή με τους συνανθρώπους του. Η εφαρμογή που υλοποίησα έχει ως στόχο την διευκόλυνση του χρήστη με σκοπό να έχουν την δυνατότητα να κάνουν εύκολα και γρήγορα τις αγορές τους μέσω του κινητού τους τηλεφώνου.

ΠΙΝΑΚΑΣ ΣΗΜΑΝΤΙΚΟΤΗΤΑΣ - ΕΠΕΙΓΟΝΤΟΣ		
	ΣΗΜΑΝΤΙΚΟ	ΜΗ ΣΗΜΑΝΤΙΚΟ
ΕΠΕΙΓΟΝ	Σοβαρά απρόβλεπτα γεγονότα	Πιεστικά θέματα της στιγμής
	Κρίσεις	Καθημερινές ενέργειες
	Έργα με προθεσμίες	
ΜΗ ΕΠΕΙΓΟΝ	Προγραμματισμός	Ενασχόληση με λεπτομέρειες
	Προετοιμασία	Ενασχόληση με ευχάριστες δραστηριότητες – Αποφυγή δυσάρεστων
	Επαναπροσδιορισμός πορείας	Ονειροπόληση
	Ανάπτυξη δεξιοτήτων	
	Αναψυχή	

[ΕΙΚΟΝΑ 1 | ΠΙΝΑΚΑΣ ΣΗΜΑΝΤΙΚΟΤΗΤΑΣ - ΕΠΕΙΓΟΝΤΟΣ]

1.3 Δομή Εργασίας

Στη συνέχεια, το κεφάλαιο 2 θα αναλύσει τη μεθοδολογία για την εφαρμογή μας, δηλαδή τον τρόπο με τον οποίο αναλύθηκε και αναπτύχθηκε. Στη συνέχεια, υπάρχει το κεφάλαιο 3, όπου θα

αναπτύξω το σχέδιο δράσης που ακολούθησα και τους σημαντικούς στόχους για την ολοκλήρωση της εφαρμογής. Στη συνέχεια, το κεφάλαιο 4, όπου αναλύω το κύριο μέρος της εφαρμογής μου, αναλύοντας το πρόβλημά, τις απαιτήσεις του συστήματος, τον προγραμματισμό υλοποίησης και τον τρόπο με τον οποίο η εφαρμογή υλοποιήθηκε τελικά. Τέλος, ακολουθεί το κεφάλαιο 5, το οποίο θα αναλύσει τα αποτελέσματα και τα συμπεράσματα των εργασιών, καθώς και τον τρόπο που θα μπορούσε να αναπτυχθεί η εφαρμογή στο άμεσο μέλλον με τρόπους βελτίωσης και με προεκτάσεις που θα μπορούσαν να γίνουν.

2. Μεθοδολογία Υλοποίησης

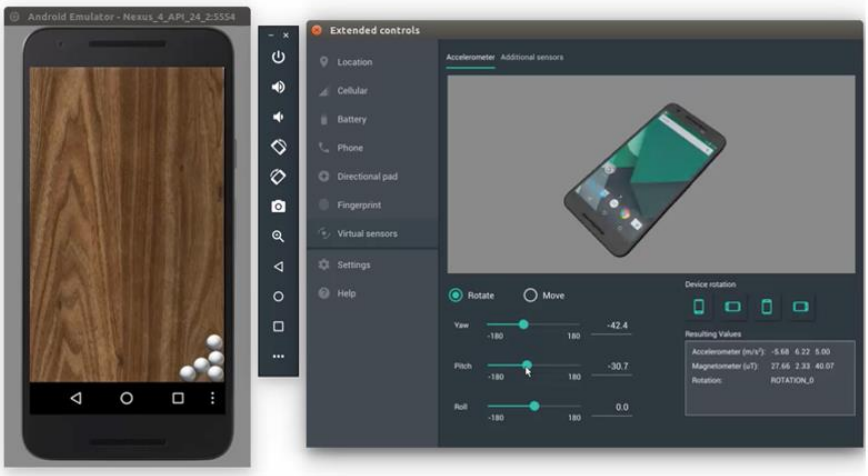
2.1 Μέθοδος Ανάλυσης & Ανάπτυξης

Χρησιμοποιήθηκε το εργαλείο Android Studio για την ανάπτυξη της εφαρμογής. Το Android Studio είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) για το λειτουργικό σύστημα Android της Google, το οποίο βασίζεται στο λογισμικό IntelliJ JetBrains και έχει σχεδιαστεί ειδικά για την ανάπτυξη του Android.

Είναι διαθέσιμο για λήψη σε λειτουργικά συστήματα με Windows, macOS και Linux. Είναι μια αντικατάσταση της έκλειψη Android εργαλεία ανάπτυξης (ADT) ως το πρωτεύον IDE για την ανάπτυξη εφαρμογών Android.

Το Android Studio ανακοινώθηκε στις 16 Μαΐου, 2013 στη διάσκεψη του Google I/O.

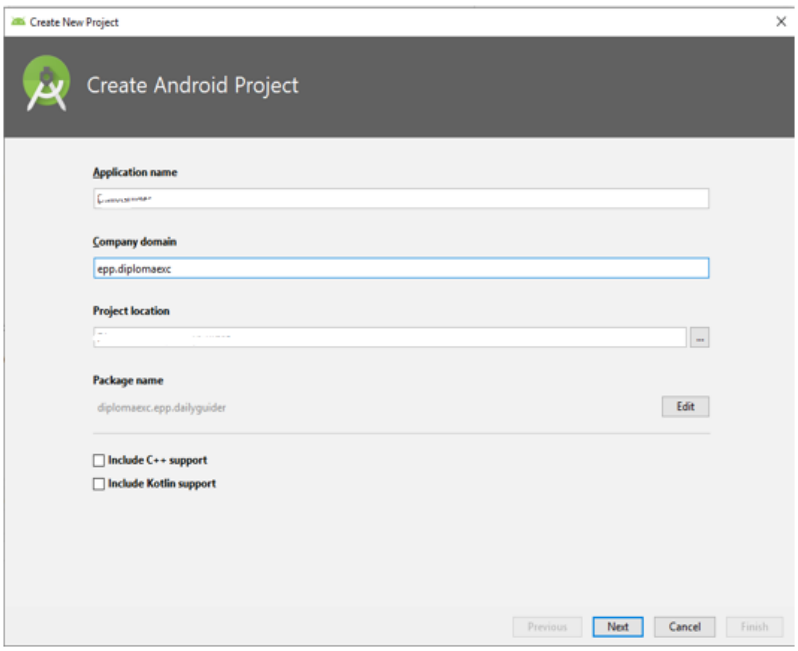
[ΕΙΚΟΝΑ 2 | ΛΟΓΟΤΥΠΟ ΠΡΟΓΡΑΜΜΑΤΟΣ ANDROID STUDIO]



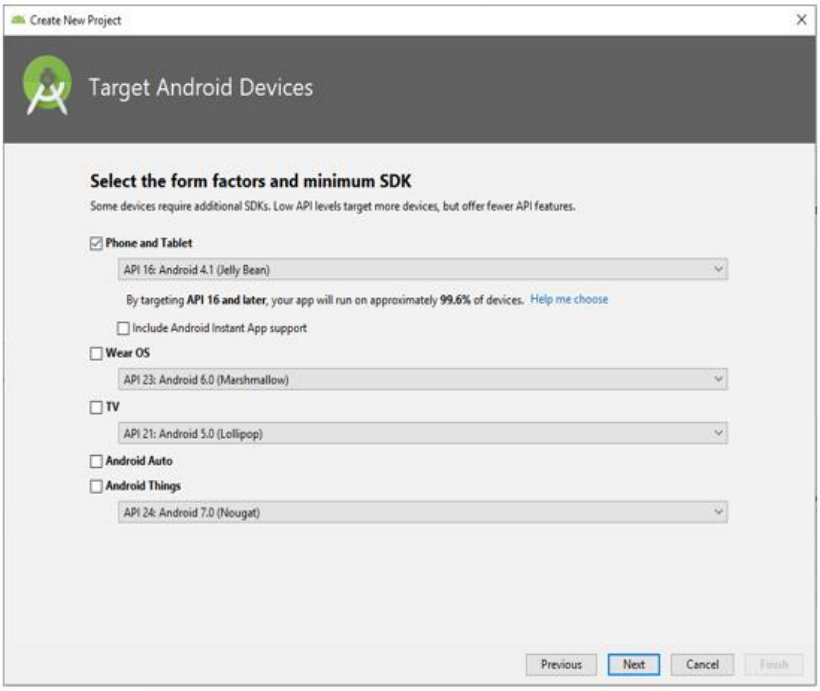
[ΕΙΚΟΝΑ 3 | ANDROID EMULATOR]

Το Android Studio υποστηρίζει τις γλώσσες προγραμματισμού Java και C++.

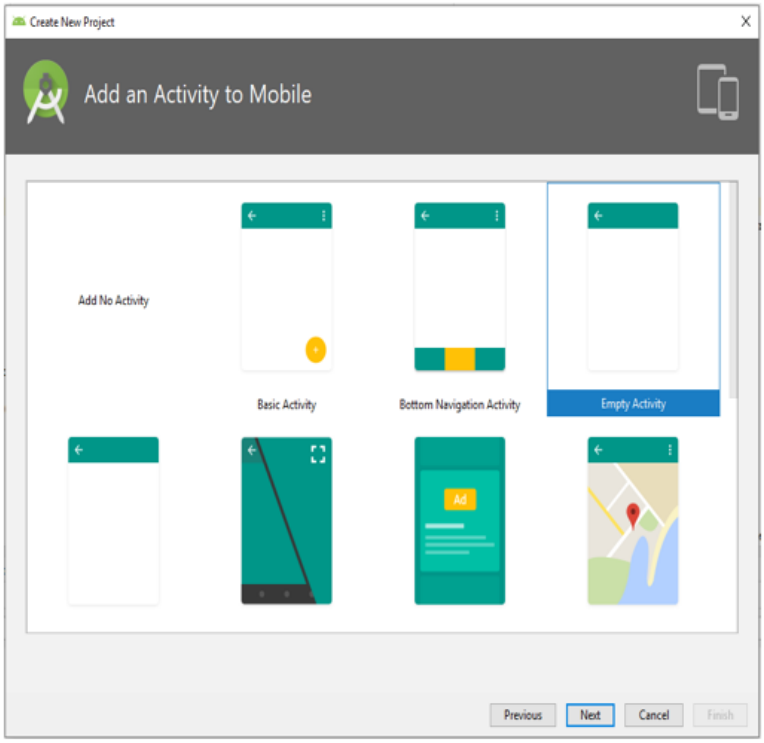
Αρχικά, μετά την εγκατάσταση του εργαλείου Android Studio δημιουργήσαμε μια νέα εφαρμογή Android, η οποία θα μας φέρει στο παράθυρο που φαίνεται παρακάτω και θα πρέπει να επιλέξουμε τις επιλογές, που εμφανίζονται στην εικόνα 4, 5, 6, 7.



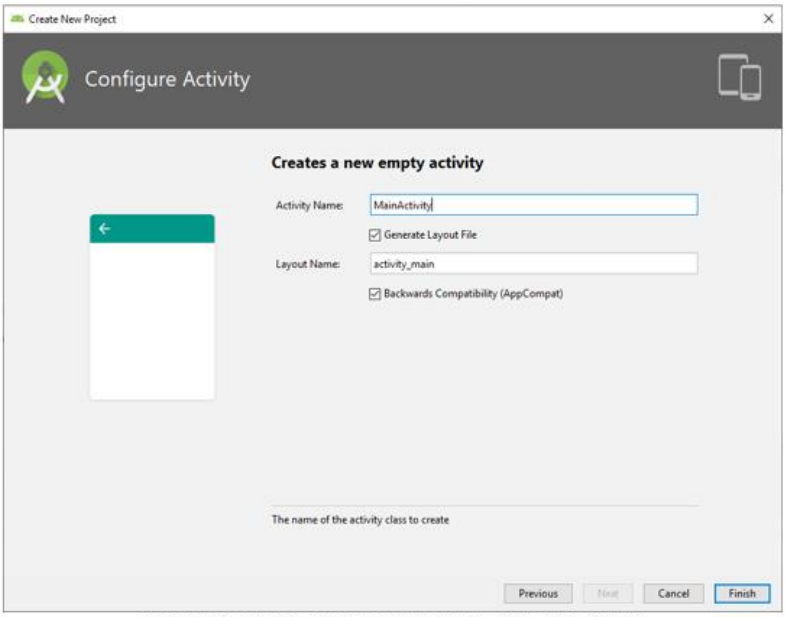
[ΕΙΚΟΝΑ 4 | ΔΕΙΓΜΑ ΔΗΜΙΟΥΡΓΙΑΣ ΕΝΩΣ ΝΕΟΥ ΠΡΟΤΖΕΚΤ (1)]



[ΕΙΚΟΝΑ 5 | ΔΕΙΓΜΑ ΔΗΜΙΟΥΡΓΙΑΣ ΕΝΩΣ ΝΕΟΥ ΠΡΟΤΖΕΚΤ (2)]

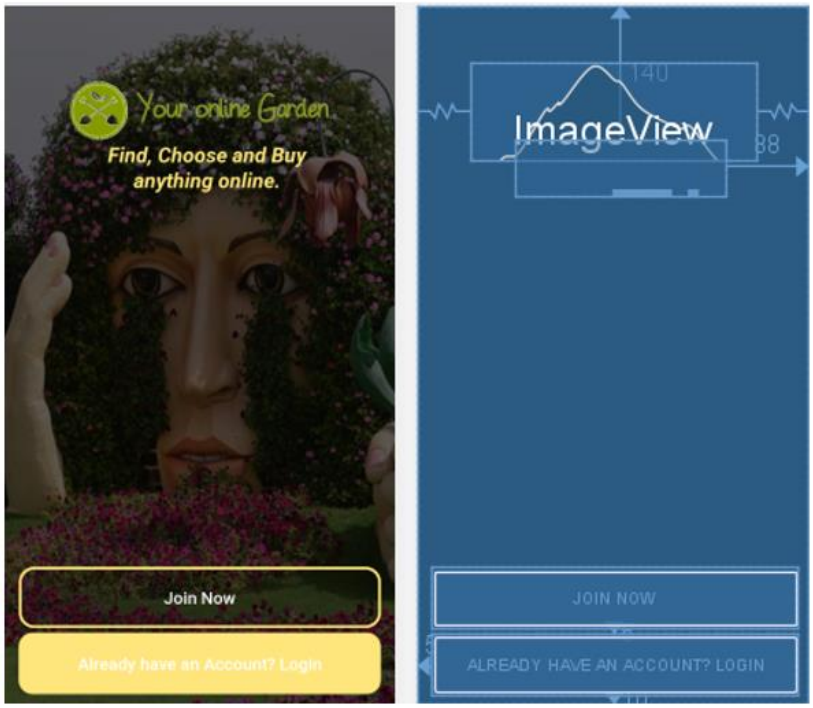


[ΕΙΚΟΝΑ 6 | ΔΕΙΓΜΑ ΔΗΜΙΟΥΡΓΙΑΣ ΕΝΩΣ ΝΕΟΥ ΠΡΟΤΖΕΚΤ (3)]



[ΕΙΚΟΝΑ 7 | ΔΕΙΓΜΑ ΔΗΜΙΟΥΡΓΙΑΣ ΕΝΩΣ ΝΕΟΥ ΠΡΟΤΖΕΚΤ (4)]

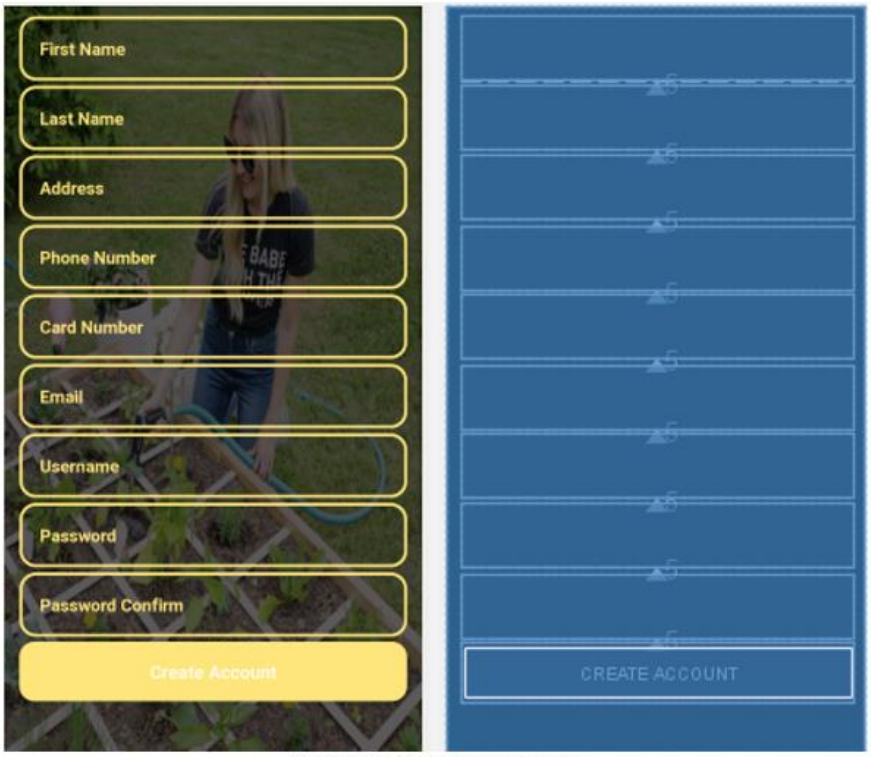
Στη συνέχεια, θα γράψουμε τον κώδικα XML πρώτα επιλέγοντας το όνομα της εφαρμογής μας, τότε αρχίζουμε να γράφουμε στον κώδικα XML με βάση την εμφάνιση της εφαρμογής μας που έχουμε κατά νου και κάνοντας την αναπαράσταση στον εξομοιωτή του Android Studio, όπως φαίνεται στην εικόνα 8.



[ΕΙΚΟΝΑ 8 | ΤΟ ΚΕΝΤΡΙΚΟ ΜΕΝΟΥ ΤΗΣ ΕΦΑΡΜΟΓΗΣ]

Πλέον, είμαστε έτοιμοι για τη συγγραφή του κώδικα μας σε γλώσσα Java.

Ταυτόχρονα, θα χρησιμοποιήσουμε την βάση δεδομένων Firebase. Firebase είναι μια βάση δεδομένων πραγματικού χρόνου που επιτρέπει την αποθήκευση δέντρο των λιστών των αντικειμένων. Επιπλέον επιτρέπει τον συγχρονισμό δεδομένων μεταξύ διαφορετικών συσκευών. Είναι μια βάση δεδομένων NoSQL JSON. Ο στόχος είναι να αποθηκευτεί το όνομα και το επώνυμο του χρήστη, η διεύθυνση, ο αριθμός τηλεφώνου, την πίστωση και έναν μοναδικό κωδικό που ο χρήστης πληκτρολογεί και για επιβεβαίωση δακτυλογραφείται ξανά για να επαληθεύσει ότι είναι ίδια με τη αρχική πληκτρολόγηση. Επίσης, ο χρήστης θα έχει ένα όνομα χρήστη και έναν κωδικό πρόσβασης, όπως φαίνεται στην εικόνα 9. Επιπλέον στην βάση αποθηκεύονται όλες οι παραγγελίες, τα προϊόντα αλλά και τα προϊόντα που ο κάθε χρήστης έχει «κρατήσει» στο καλάθι του.



[ΕΙΚΟΝΑ 9 | ΤΟ ΜΕΝΟΥ ΕΓΓΡΑΦΗΣ]

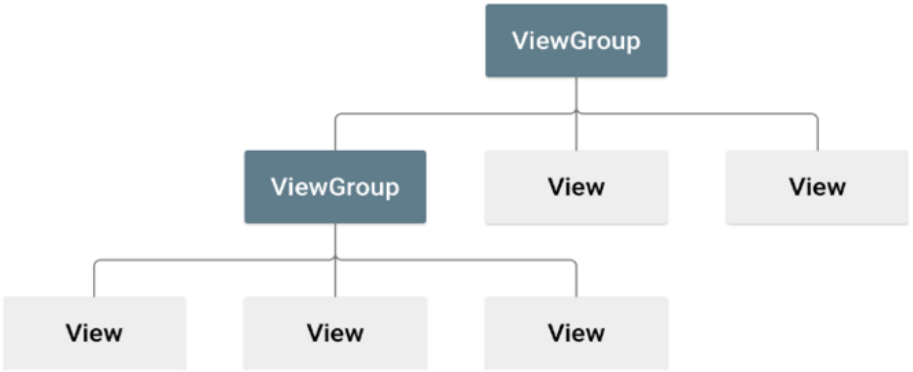
3. Σχέδιο Δράσης για την εκπόνηση της Διπλωματικής Εργασίας

3.1. Android Studio Project – Your online Garden

3.1.1 XML

Το XML αντιπροσωπεύει την επεκτάσιμη γλώσσα σήμανσης, τη γλώσσα σήμανσης εφελκυσμού. Το XML είναι μια αυτοπεριγραφική γλώσσα σήμανσης παρόμοια με την HTML και έχει σχεδιαστεί για την αποθήκευση και τη μεταφορά δεδομένων.

Ένα σχήμα ορίζει τη δομή για ένα περιβάλλον εργασίας χρήστη στην εφαρμογή. Όλα τα στοιχεία της διάταξης κατασκευάζονται με τη χρήση μιας ιεραρχίας αντικειμένου προβολή και εμφάνιση ομάδας. Μια προβολή συνήθως σχεδιάζει κάτι με το οποίο μπορεί να δει και να αλληλοεπιδράσει ο χρήστης. Ενώ μια εμφάνιση ομάδας είναι ένα αόρατο κοντέινερ που καθορίζει τη δομή διάταξης για την προβολή άλλων αντικειμένων στο ViewGroup.



[EIKONA 10 | VIEWGROUP]

Τα αντικείμενα View συνήθως ονομάζονται "στοιχεία γραφικών" και μπορεί να είναι μία από τις διάφορες υποκατηγορίες, όπως κουμπί ή προβολή κειμένου. Τα αντικείμενα προβολής ομάδας ονομάζονται συνήθως "διατάξεις" και μπορεί να είναι ένας από τους πολλούς τύπους που παρέχουν μια διαφορετική δομή διάταξης, όπως η γραμμική διάταξη ή ο περιορισμός διάταξης.

Μια διάταξη μπορεί να δηλωθεί με δύο τρόπους:

- Καταχώρηση στοιχείων διεπαφής χρήστη (UI) σε XML. Το Android παρέχει ένα απλό λεξιλόγιο XML που αντιστοιχεί στην προβολή κλάσεων και δευτερεύουσες τάξεις, όπως αυτές για widgets και διατάξεις. Επίσης, μπορεί να χρησιμοποιηθεί επεξεργαστής διάταξη του Android Studio για να δημιουργήσετε το σχήμα XML χρησιμοποιώντας μια διασύνδεση μεταφοράς και απόθεσης.

• Λεπτομέρειες διάταξης εγγράφου κατά το χρόνο εκτέλεσης. Η εφαρμογή μπορεί να δημιουργήσει αντικείμενα προγραμματισμού προβολή και προβολής ομάδας.

Η δήλωση του περιβάλλοντος εργασίας χρήστη (UI) σε XML επιτρέπει να διακρίνει την παρουσίαση της εφαρμογής από τον κώδικα που ελέγχει τη συμπεριφορά της. Η χρήση αρχείων XML καθιστά επίσης ευκολότερη την παροχή διαφορετικών προφίλ για διαφορετικά μεγέθη και προσανατολισμούς.

Πρώτα απ ' όλα, γράφουμε τον κώδικα XML πρώτα, επιλέγοντας το όνομα της εφαρμογής ως "το διαδικτυακό σας κήπο" τότε αρχίζω να γράφω στον κώδικα XML με βάση την εμφάνιση της εφαρμογής που έχουμε κατά νου, αποθηκευμένη σε ένα αρχείο (. XML).

Για παράδειγμα, η δομή ενός κώδικα XML είναι η εξής:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="wrap_content"
    android:layout_height="match_parent"
    tools:context=".RegisterActivity">

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="127dp"
        android:layout_height="129dp"
        android:layout_alignParentStart="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"
        android:layout_marginStart="27dp"
        android:layout_marginLeft="27dp"
        android:layout_marginTop="0dp"
        app:srcCompat="@drawable/logo" />

    <TextView
        android:id="@+id/textView3"
        android:layout_width="246dp"
        android:layout_height="wrap_content"
        android:layout_below="@+id/imageView"
        android:layout_alignStart="@+id/imageView"
        android:layout_alignEnd="@+id/textView5"
        android:layout_alignRight="@+id/textView5"
        android:layout_marginStart="0dp"
        android:layout_marginTop="20dp"
        android:layout_marginEnd="79dp"
        android:layout_marginRight="79dp"
        android:text="Already a member?"
        android:textColor="@color/colorPrimary"
        android:textSize="24sp" />

    <EditText
        android:id="@+id/txtlogusername"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/textView3"
        android:layout_alignStart="@+id/imageView"
        android:layout_alignEnd="@+id/textView3"
        android:layout_alignRight="@+id/textView3"
        android:layout_marginStart="0dp"
        android:layout_marginTop="0dp"
        android:layout_marginEnd="31dp"
        android:layout_marginRight="31dp"
        android:ems="10"
        android:hint="Username"
        android:inputType="textPersonName"
        android:textColor="@color/colorAccent" />
</RelativeLayout>
```

[ΕΙΚΟΝΑ 11 | ΠΑΡΑΔΕΙΓΜΑ ΚΩΔΙΚΑ XML]

Ταυτόχρονα, κάνουμε την αναπαράσταση στον εξομοιωτή του Android Studio, όπως φαίνεται στην εικόνα 12.

[ΕΙΚΟΝΑ 12 | Εξομοιωτής]

3.1.2 Java

Η γλώσσα προγραμματισμού Java είναι μια γλώσσα προγραμματισμού του τυπου reverse engineering, βασισμένη σε κλάσεις και αντικείμενα. Έχει πολλά πλεονεκτήματα, καθώς έχει σχεδιαστεί και ειδικά στο Android Studio όταν κάποιος αναπτύσσει μια εφαρμογή. Όταν ανοίγεται ένα νέο παράθυρο με το όνομα "Δημιουργία νέας κλάσης", μπορείτε εύκολα να δημιουργήσετε τις ακόλουθες νέες τάξεις και τύπους:

- Κλάσεις Java
- Τάξεις καταμέτρησης και Σίνγκλετον
- Τύποι διασύνδεσης και σχολιασμού

Αφού συμπληρώσετε τα παράθυρα διαλόγου "Δημιουργία νέας κλάσης" και κάνετε κλικ στο κουμπί OK, το Android Studio δημιουργεί ένα αρχείο (. Java) που περιέχει έναν κωδικό σκελετού, ο οποίος περιλαμβάνει μια δήλωση πακέτου, όλες τις απαραίτητες εισροές, μια κεφαλίδα και δήλωση κλάσης ή τύπου. Στη συνέχεια, μπορεί να προστεθεί ένας κωδικός σε αυτό το αρχείο.

Τα πρότυπα αρχείων καθορίζουν πώς το Android Studio δημιουργεί σκελετό κώδικα. Τα πρότυπα αρχείων που παρέχονται στο Android Studio μπορούν να χρησιμοποιηθούν όπως είναι ή προσαρμοσμένα για να ταιριάζει με τη διαδικασία ανάπτυξης της εφαρμογής.

Παρακάτω, θα παρουσιάσουμε βασικά μέρη της Java για να το κατανοήσουμε και να το αναπτύξουμε σωστά.

- Πρώτον, θα αναπτύξουμε τις βασικές έννοιες της Java. Οι τάξεις και ο τρόπος δήλωσης των μεταβλητών και μεθόδων κλάσης.

```
1   package javatutorial;
2
3   public class Car{
4
5       String name;
6       int gear;
7       double speed;
8
9       void Accelerate(){
10      }
11
12      void SlowDown(){
13      }
14
15  }
```

[ΕΙΚΟΝΑ 13 | JAVATUTORIAL ΚΛΑΣΕΙΣ]

- Constructor είναι μια μέθοδος που εφαρμοζεται για την προετοιμασία του αντικειμένου και για την προετοιμασία άλλων παραμέτρων μέσα στο αντικείμενο. Γενικά, ένα αντικείμενο μπορεί να έχει περισσότερες από μία κατασκευές, με διαφορετική προσαρμογή.

```
1   package javatutorial;
2
3   public class Car{
4
5       string name;
6       int gear;
7       double speed;
8
9       public void car (String Cname){
10          name=Cname;
11      }
12
13  }
```

[ΕΙΚΟΝΑ 14 | JAVATUTORIAL CONSTRUCTOR]

- Το Inheritance είναι όταν μια κλάση έχει ήδη χαρακτηριστικά μιας άλλης κλάσης. Επίσης, η νέα κλάση μπορεί να διαχειριστεί μεταβλητές κλάσης χωρίς να χρειάζεται να δηλωθεί αν έχει μεταβιβαστεί από την άλλη κλάση.

```
1   package javatutorial;
2
3   public class Opel extends Car{
4
5       public void Opel(){
6           this.name="Astra";
7       }
8
9   }
```

[ΕΙΚΟΝΑ 15 | JAVATUTORIAL ΚΛΗΡΟΝΟΜΙΚΟΤΗΤΑ]

- Απόκρυψη δεδομένων δηλαδή με τη χρήση της λέξης Private δηλώνουμε μια μεταβλητή ως ιδιωτική που επιτρέπει στο αντικείμενο να προστατεύσει τις πληροφορίες από άλλες μεθόδους εκτός από το ίδιο το αντικείμενο.

```
1   package javatutorial;
2
3   public class Car{
4
5       private String name;
6       private int gear;
7       private double speed;
8
9       public void car(String Cname){
10          name=Cname;
11      }
12
13  }
```

[ΕΙΚΟΝΑ 16 | ΑΠΟΚΡΥΨΗ ΠΛΗΡΟΦΟΡΙΑΣ]

3.1.3 Firebase Database

Η Firebase Database είναι ένα platform ανάπτυξης Mobile και Web App που παρέχει στους προγραμματιστές μια πληθώρα εργαλείων και υπηρεσιών για να τους βοηθήσει να αναπτύξουν εφαρμογές υψηλής ποιότητας, να καλλιεργήσουν τη βάση χρήστη τους και να κερδίσουν περισσότερο κέρδος.

Η βάση δεδομένων του firebase σε πραγματικό χρόνο είναι μια βάση δεδομένων NoSQL που φιλοξενείται από το cloud και επιτρέπει να αποθηκεύοντε και να συγχρονίζοντε μεταξύ των χρηστών σας σε πραγματικό χρόνο. Η βάση δεδομένων σε πραγματικό χρόνο είναι πραγματικά μόνο ένα μεγάλο αντικείμενο JSON που οι προγραμματιστές μπορούν να διαχειριστούν σε πραγματικό χρόνο.

[ΕΙΚΟΝΑ 17 | Realtime Database -> Tree of Values]

Με ένα μόνο API, η βάση δεδομένων firebase παρέχει στην εφαρμογή σας τόσο την τρέχουσα τιμή των δεδομένων όσο και τυχόν ενημερώσεις για τα δεδομένα αυτά. Ο συγχρονισμός σε πραγματικό χρόνο διευκολύνει τους χρήστες σας να έχουν πρόσβαση στα δεδομένα τους από κάθε τύπου συσκευή, είτε πρόκειται για Web είτε για κινητές συσκευές. Η βάση δεδομένων σε πραγματικό χρόνο βοηθά επίσης τους χρήστες σας να συνεργαστούν. Ένα άλλο εκπληκτικό όφελος της βάσης δεδομένων σε πραγματικό χρόνο είναι ότι πλοία με κινητά και Web SDK, επιτρέποντάς σας να χτίσετε τις εφαρμογές σας χωρίς την ανάγκη για διακομιστές. Όταν οι χρήστες σας λειτουργούν χωρίς σύνδεση, τα SDK της βάσης δεδομένων σε πραγματικό χρόνο χρησιμοποιούν την τοπική μνήμη cache στη συσκευή για την εξυπηρέτηση και την αποθήκευση των αλλαγών. Όταν η συσκευή έρθει σε σύνδεση, τα τοπικά δεδομένα συγχρονίζονται αυτόματα. Η βάση δεδομένων σε πραγματικό χρόνο μπορεί επίσης να ενσωματωθεί με τον έλεγχο ταυτότητας firebase για να παρέχει μια απλή και διαισθητική διαδικασία ελέγχου ταυτότητας.

Το firebase έχει απλοποιήσει τη διαδικασία ανάκτησης συγκεκριμένων δεδομένων από τη βάση δεδομένων μέσω ερωτημάτων. Τα ερωτήματα δημιουργούνται με την αλυσιδωτή σύνδεση μιας ή περισσότερων μεθόδων φιλτραρίσματος.

Το firebase έχει τέσσερεις συναρτήσεις ταξινόμησης.

- orderByKey()
- orderByChild('child')
- orderByValue()
- orderByPriority()

Σημειώστε ότι θα λαμβάνετε δεδομένα μόνο από ένα ερώτημα, εάν έχετε χρησιμοποιήσει τη μέθοδο on () ή μία (). Μπορείτε επίσης να χρησιμοποιήσετε αυτές τις σύνθετες συναρτήσεις ερωτημάτων για να περιορίσετε περαιτέρω τα δεδομένα:

- startAt('value')
- endAt('value')
- equalTo('child_key')
- limitToFirst(10)
- limitToLast(10)

Στο firebase, η υποβολή ερωτημάτων περιλαμβάνει επίσης βήματα. Πρώτα, μπορείτε να δημιουργήσετε μια αναφορά στο γονικό κλειδί και, στη συνέχεια, μπορείτε να χρησιμοποιήσετε μια λειτουργία παραγγελίας. Προαιρετικά, μπορείτε επίσης να προσαρτήσετε μια συνάρτηση ερωτήματος για έναν πιο προηγμένο περιορισμό.

```
const db = firebase.database();

const firebaseRef = db.child(`child`);

firebaseRef.orderByChild("user").equalTo("GeekyAnts").on("child_added",
    function(snapshot) {
        console.log(snapshot.key);
    }
);
```

[ΕΙΚΟΝΑ 18 | FIREBASE GET DATA]

Το firebase Storage είναι μια αυτόνομη λύση για την αποστολή περιεχομένου που παράγεται από χρήστες, όπως εικόνες και βίντεο από μια συσκευή iOS και Android, καθώς και από το Web. Ο αποθηκευτικός χώρος firebase έχει σχεδιαστεί ειδικά για την κλιμάκωση των εφαρμογών σας, την παροχή ασφάλειας και την εξασφάλιση της ανθεκτικότητας του δικτύου. Ο χώρος αποθήκευσης firebase χρησιμοποιεί ένα απλό φάκελο/σύστημα αρχείων για τη διάρθρωση των δεδομένων του.

```
var storageRef = firebase.storage.ref("folderName/file.jpg");
var fileUpload = document.getElementById("fileUpload");
fileUpload.on('change',
    function(evt) {
        var firstFile = evt.target.file[0]; // get the first file uploaded
        var uploadTask = storageRef.put(firstFile);
    }
);
```

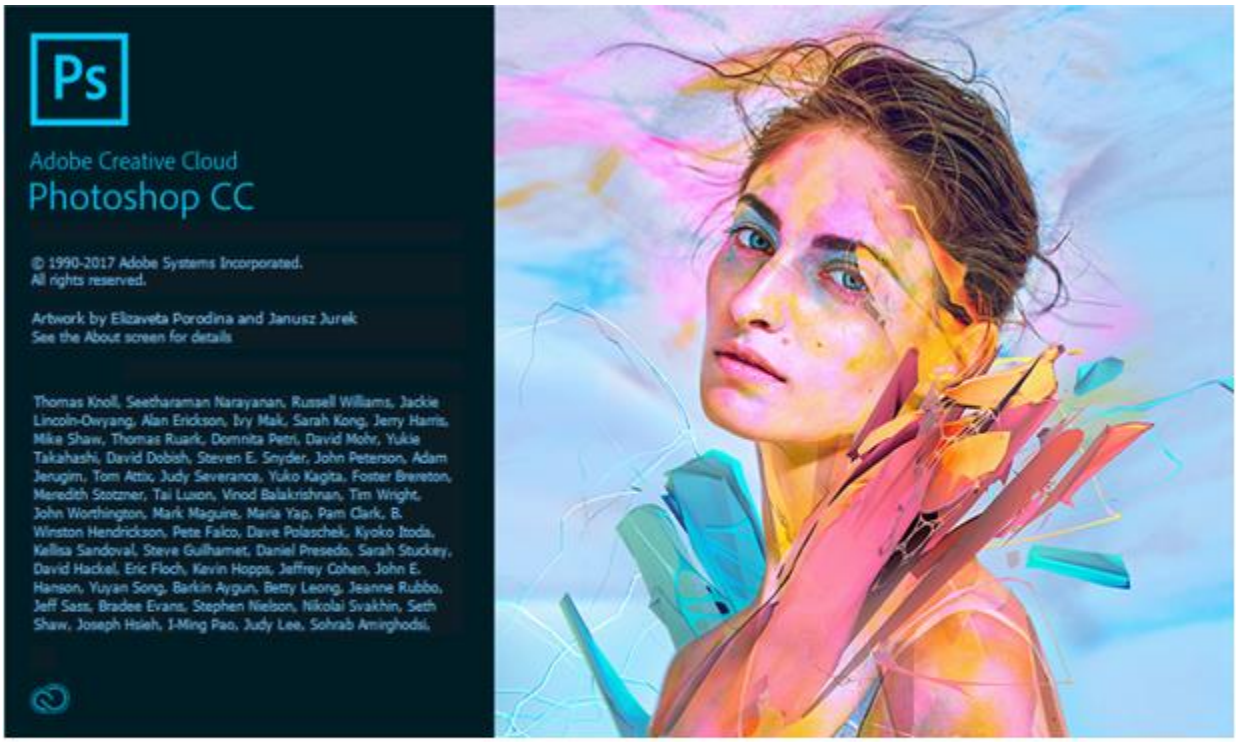
[ΕΙΚΟΝΑ 19 | FIREBASE STORAGE]

3.1.4 Adobe Photoshop

Ένα λογισμικό επεξεργασίας εικόνας που αναπτύχθηκε και κατασκευάστηκε από την Adobe Systems Inc. Photoshop θεωρείται ένας από τους ηγέτες στο λογισμικό επεξεργασίας φωτογραφιών.

Το λογισμικό επιτρέπει στους χρήστες να χειρίζονται, να περικόψετε, να αλλάξετε το μέγεθος και να διορθώσετε το χρώμα σε ψηφιακές φωτογραφίες. Το λογισμικό είναι ιδιαίτερα δημοφιλές μεταξύ των επαγγελματιών φωτογράφων και γραφίστες.

Είναι το κυρίαρχο λογισμικό για την επεξεργασία και το χειρισμό φωτογραφιών στην αγορά. Οι χρήσεις του κυμαίνονται από την πλήρη εμφάνιση μεγάλων παρτίδων φωτογραφιών μέχρι τη δημιουργία περίπλοκων ψηφιακών σχεδίων και σχεδίων που μιμούνται αυτά που γίνονται με το χέρι.



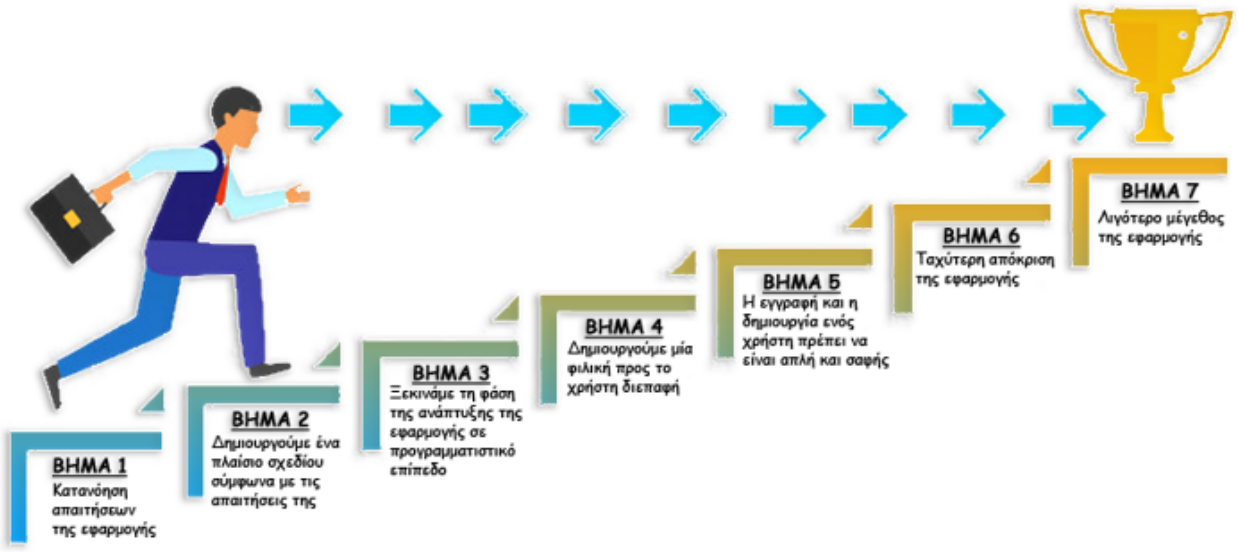
[ΕΙΚΟΝΑ 20 | ADOBE PHOTOSHOP LOGO]

Στην εφαρμογή μας χρησιμοποιήσαμε το Adobe Photoshop για την δημιουργία και επεξεργασία των εικονιδίων.

3.2 Στόχοι ολοκλήρωσης της Πτυχιακής Εργασίας

Για να επιτύχουμε τους στόχους για την ολοκλήρωση της εφαρμογής μας πρέπει να ακολουθήσουμε τα εξής βήματα:

1. Κατανόηση των απαιτήσεων της εφαρμογής μας
2. Δημιουργούμε ένα πλαίσιο σχεδίου σύμφωνα με τις απαιτήσεις της
3. Ξεκινάμε τη φάση της ανάπτυξης της εφαρμογής σε προγραμματιστικό επίπεδο
4. Δημιουργούμε μία φιλική προς τον χρήστη διεπαφή
5. Η εγγραφή και η δημιουργία ενός χρήστη πρέπει να είναι απλή και σαφής
6. Ταχύτερη απόκριση της εφαρμογής
7. Λιγότερο μέγεθος της εφαρμογής



[ΕΙΚΟΝΑ 22 | ΣΤΟΧΟΙ ΟΛΟΚΛΗΡΩΣΗΣ ΕΦΑΡΜΟΓΗΣ]

4. Κύριο μέρος Πτυχιακής Εργασίας

4.1. Ανάλυση Προβλήματος

Προκειμένου να ξεκινήσουμε ένα Activity στο Android Studio, θα πρέπει πρώτα να φτιάξουμε τη διεπαφή μας μέσω του Visual layout Editor που παρέχεται από το Android Studio. Στη συνέχεια, θα αναπτύξουμε τον κώδικα Java σε συνδυασμό με τη βάση Firebase Database για την αποθήκευση δεδομενων αλλα και την ανάκτηση αυτωμ. Ταυτόχρονα, για την εύρεση τοποθεσιών χρησιμοποιήσαμε κάποια από τα APIs της Google και πιο συγκεκριμένα το Directions API, το Places API. Επιπλέον, χρησιμοποιήθηκε το Map SDK for Android. Παράλληλα, για την διεπαφή μας δημιουργήθηκαν κάποια εικονίδια για την εφαρμογή μας από εμάς μέσω του προγράμματος και σχεδιασμού του Photoshop. Παρακάτω θα εμβαθύνουμε σε κάθε ένα από αυτά αναλύοντας τη διαδικασία που ακολουθήσαμε για την ανάπτυξη της εφαρμογής μας Daily Guider.

Πρώτα από όλα, αναλύσα το πρόβλημά σε μικρότερα προβλήματα, προκειμένου να καταστήσω εύκολη και ταχύτερη την εφαρμογή. Το πρώτο μου πρόβλημα ήταν να κατανοήσω τις ανάγκες και τα βοηθητικά προγράμματα της αίτησής μας στον χρήστη. Στη συνέχεια μελέτησα το σχεδιασμό της διεπαφής μου, με σκοπο να είναι όσο το πιο κατανοητή και φιλική προς το χρήστη. Ένα κατανοητό και εύκολο στη χρήση interface είναι απαραίτητο για την εφαρμογή, επειδή είναι μια εφαρμογή χωρίς όριο ηλικίας. Μετά από σκέψη για τη μορφολογία των "Windows" και τη διεπαφή μας γενικά, το επόμενο βήμα ήταν να σκεφτώ πώς θα είναι το "γραφικό" μέρος της εφαρμογής, με αυτό να συνεπάγεται όλα τα εικονίδια που θα χρησιμοποιηθούν σε αυτό. Μετά την επιτυχή εφαρμογή των προαναφερθέντων προβλημάτων, έπρεπε να ασχοληθώ με το πρόβλημα αφού το σχεδιάσουμε και στη συνέχεια το εφαρμόσουμε.

4.1.2 Απαιτήσεις Συστήματος

Για να ολοκρηρώσω την πτυχιακή χρειάστηκα έναν εξοπλισμό που να εκπλήρωνε την απαίτηση όλων των απαραίτητων χρησιμοποιούμενων προγραμμάτων, των οποίων η βοήθεια ήταν απαραίτητη για την πλήρη και ορθή εφαρμογή της αίτησής μας. Τα προγράμματα που χρησιμοποιήσαμε ήταν Android Studio, firebase και Adobe Photoshop. Το πιο απαιτητικό από τα παραπάνω προγράμματα ήταν το Android Studio, που σημαίνει ότι αν εκπληρώσουμε τις απαιτήσεις του μεταγλωττιστή μας, θα έχουμε εκπληρώσει τις απαιτήσεις άλλων προγραμμάτων. Οι απαιτήσεις του Android Studio ήταν ένα λογισμικό νέας γενιάς της Microsoft (π.χ. Windows 7/8/10), 8GB RAM με ένα επιπλέον άλλο 1GB για εξομοιωτή του Android, 4GB ελεύθερου

χώρου στο σκληρό δίσκο (500MB για το IDE και 1,5 Gigabyte μνήμη ραμ για το Android SDK και το είδωλο του συστήματος εξομοίωσης) και Τέλος, ένα ψήφισμα μεγαλύτερο ίσο με 1280 x 800 εικονοστοιχεία. Στη συνέχεια, χρειαζόμασταν ένα smartphone για να δοκιμάσει την τελική έκδοση της εφαρμογής μας. Τα απαραίτητα χαρακτηριστικά του κινητού ήταν μια έκδοση λογισμικού Android ζελέ φασολιών το μεγαλύτερο, 2GB RAM, GPS και υποστήριξη των υπηρεσιών της Google.

4.2 Σχεδιασμός Προβλήματος

Πρώτα από όλα, σχεδιάστηκαν και επεξεργάστηκαν τα εικονίδια και οι εικόνες που χρησιμοποιήθηκαν στην εφαρμογή μας με το Adobe Photoshop τα οποία χωρίζονται στα παρακάτω :

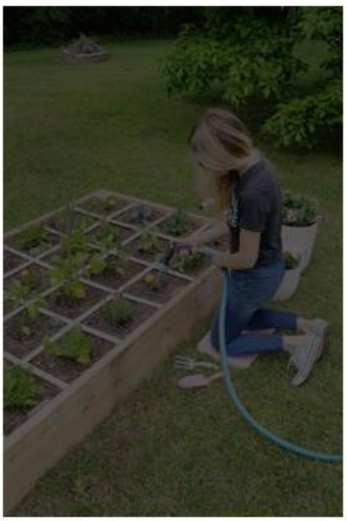
- Το λογότυπο της εφαρμογής

[ΕΙΚΟΝΑ 23 | ΛΟΓΟΤΥΠΟ ΕΦΑΡΜΟΓΗΣ]

- Η αρχική εικόνα της εφαρμογής

[ΕΙΚΟΝΑ 24 | ΑΡΧΙΚΗ ΕΙΚΟΝΑ ΕΦΑΡΜΟΓΗΣ]

- Η εικόνα της εισόδου του χρήστη



[ΕΙΚΟΝΑ 25 | ΕΙΚΟΝΑ ΕΙΣΑΓΩΓΗΣ ΧΡΗΣΤΗ]

- Η εικόνα της εγγραφής του χρήστη

[ΕΙΚΟΝΑ 26 | ΕΙΚΟΝΑ ΕΓΓΡΑΦΗΣ ΧΡΗΣΤΗ]

- Τα εικονίδια των κατηγοριών των προϊόντων

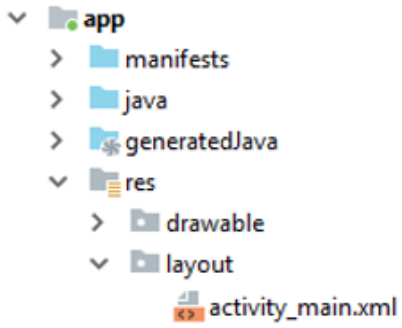
[ΕΙΚΟΝΑ 27 | ΕΙΚΟΝΙΔΙΑ ΚΑΤΗΓΟΡΙΩΝ ΠΡΟΙΟΝΤΩΝ]

- Τα εικονίδια του μενού

[ΕΙΚΟΝΑ 28 | ΑΡΧΕΙΑ ΠΡΟΤΖΕΚΤ]

Στη συνέχεια αντιμετώπισα το πρόβλημα του σχεδιασμού της διεπαφής της εφαρμογής και ακολουθούσα τα παρακάτω βήματα:

1) Μετά τη δημιουργία του έργου μας και ειδικά της πρώτης μας κλάση με το όνομα activity_main όπως είδαμε προηγουμένως σε εικόνες 4,5,6,7 πήγα στο φάκελο res/layout όπως φαίνεται παρακάτω στις εικόνες 29 άνοιξα το αρχείο xml μέσω του Visual Layout Editor του Android Studio με σκοπό να σχεδιάσω το γραφικό περιβάλλον.



[ΕΙΚΟΝΑ 29 | ΑΡΧΕΙΑ ΠΡΟΤΖΕΚΤ]

[EIKONA 30 | ACTIVITY MAIN]

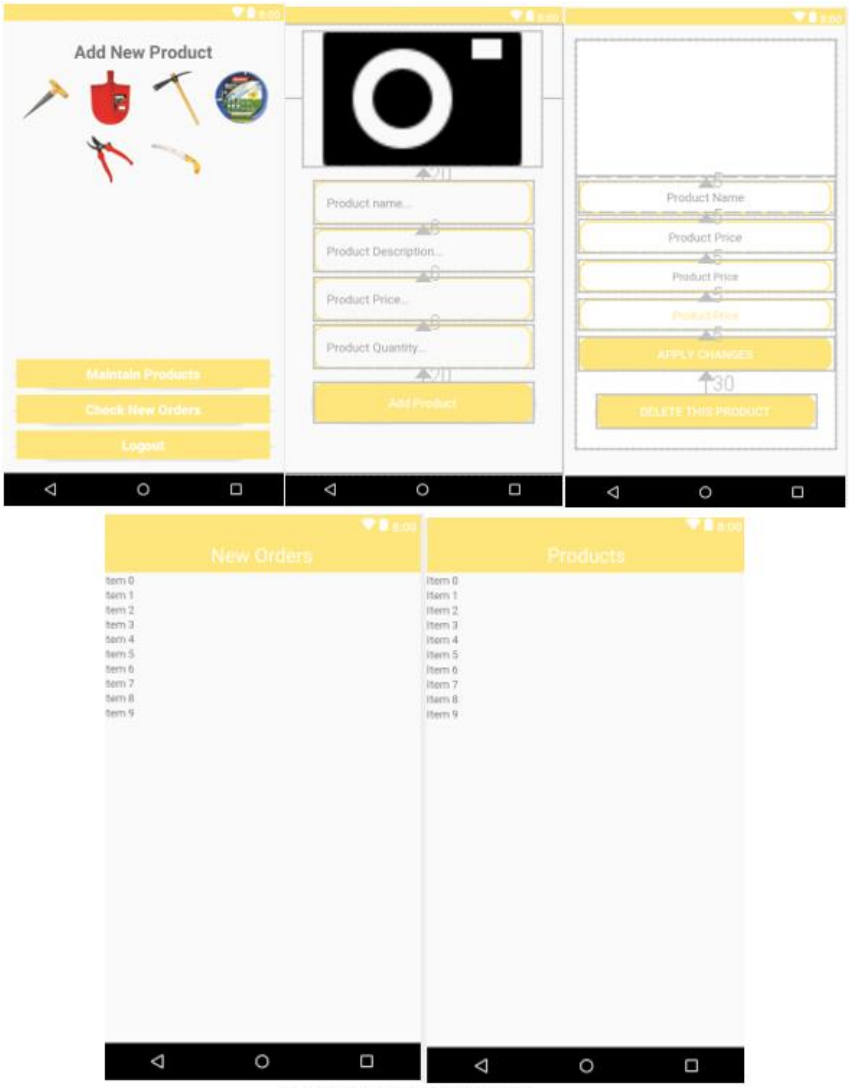
2) Αφού τελειώσα με την activity_main, το επόμενο βήμα ήταν να δημιουργήσω μια νέα κλάση την activity_register. Η κλάση αυτή την οποία σχεδίασα στο Visual layout Editor όπως έκανα στο Βήμα 1, με σκοπό ένας χρήστης να εγγράφεται.

[EIKONA 31 | ACTIVITY REGISTER]

3) Μετά δημιούργησα την κλάση activity_login την οποία σχεδίασα με στο Visual Layout Editor με σκοπό ο χρήστης να εισάγει το username και το password του για να συνδεθεί είτε σαν χρήστης είτε σαν διαχειριστής.

[ΕΙΚΟΝΑ 32 | ACTIVITY LOGIN]

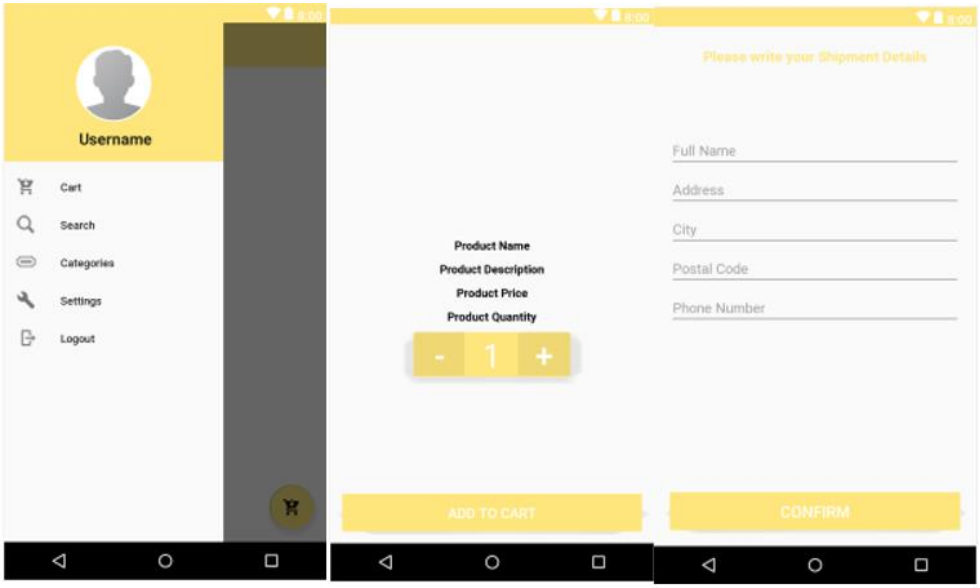
4) Μετά δημιούργησα και σχεδίασα με το Visual Layout Editor τις κλάσεις, που θα είναι αποκλειστικά για τον διαχειριστή της εφαρμογής :
   1) activity_admin_category με σκοπό να οδηγείτε εκεί ένας διαχειριστής μόλις κάνει είσοδο ο οποίος θα έχει τις δυνατότητες να εισάγει ένα νέο προϊόν σε μια κατηγορία.
   2) activity_admin_add_new_product με σκοπό εισάγει ένα προϊόν σε μια κατηγορία.
   3) activity_admin_maintain_products με σκοπό να επεξεργάζεται τα στοιχεία ενός προϊόντος.
   4) activity_admin_new_orders με σκοπό να «επιβεβαιώνει» τις παραγγελίες των χρηστών.
   5) activity_admin_user_products με σκοπό να μπορεί να βλέπει τα προϊόντα μιας παραγγελίας ενός χρήστη.



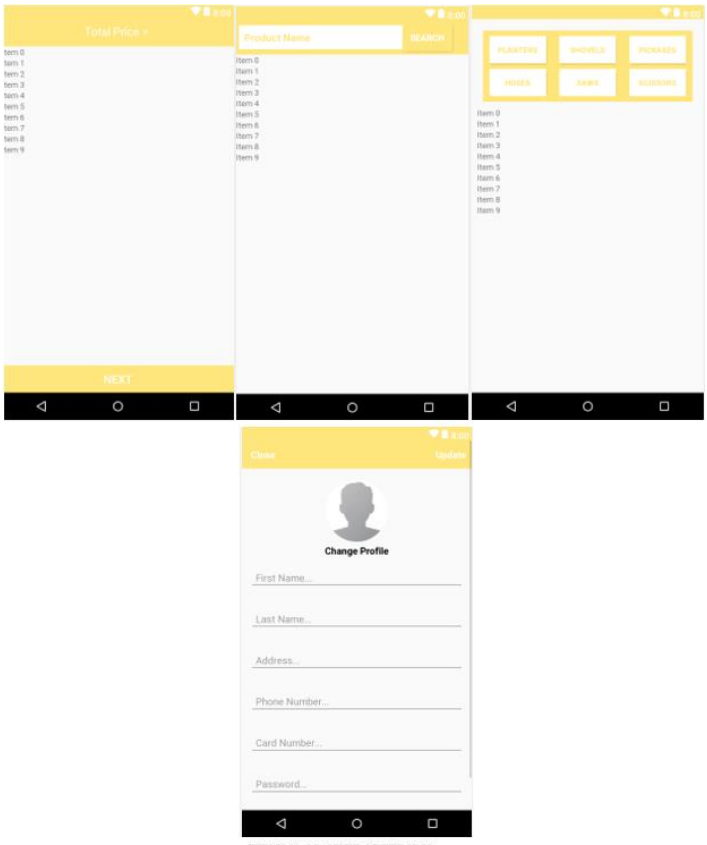
[ΕΙΚΟΝΑ 33 | ADMIN ACTIVITIES]

5) Επίσης δημιούργησα και σχεδίασα με το Visual Layout Editor τις κλάσεις, που θα είναι αποκλειστικά για τον χρήστης της εφαρμογής :
   1) activity_home με σκοπό να εμφανίζει όλα τα προϊόντα που θα έχει το μαγαζί αλλά και να έχει ένα nav menu το οποίο θα έχει τις «δικές του» κλάσεις.
      - activity_cart με σκοπό να εμφανίζονται τα προϊόντα που θα θέλει να αγοράσει ο χρήστης.
      - activity_search_products με σκοπό να μπορεί να αναζητήσει ένας χρήστης ένα προϊόν.
      - activity_categories με σκοπό να βλέπει ο χρήστης τα προϊόντα ανά κατηγορίες.
      - activity settings με σκοπό να μπορεί να επεξεργαστεί ο χρήστης το προφίλ του.
   2) activity_product_details με σκοπό να εμφανίζονται οι πληροφορίες του προϊόντος (μόλις «πατηθεί»)

3) activity_confirm_final_order με σκοπό να εισάγει τα στοιχεία παράδοσης αλλά και να την «στέλνει» για επιβεβαίωση στον διαχειριστή.



[EIKONA 34 | USER ACTIVITIES]



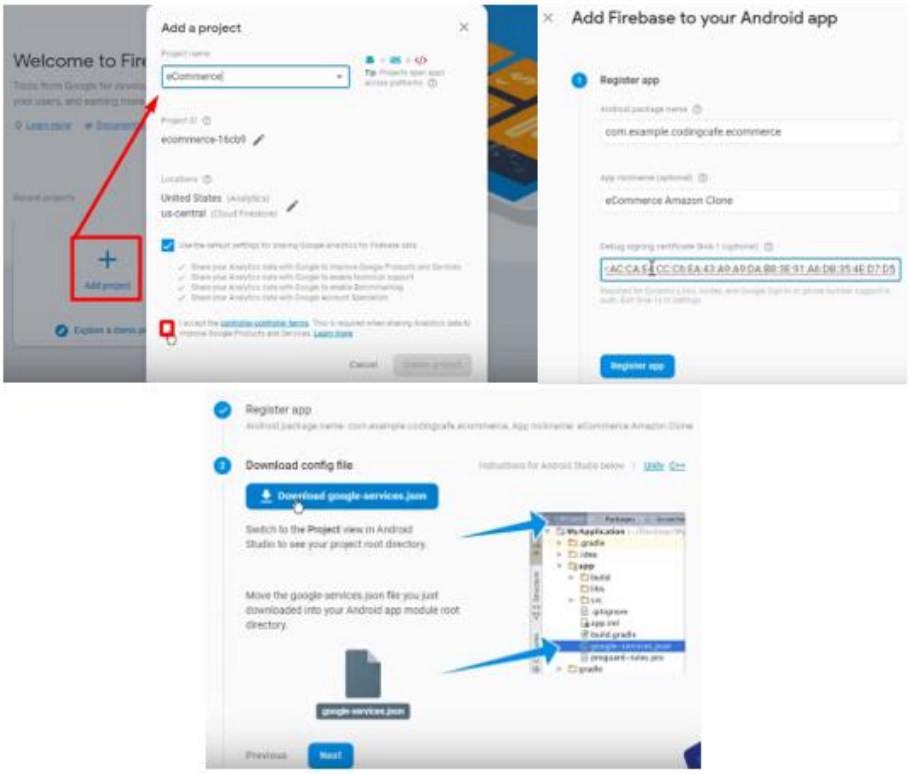
[EIKONA 35 | USER SETTINGS]

6) Τέλος δημιούργησα και σχεδίασα κάποια layouts όπως το orders_layout, cart_items και το product_items_layout με σκοπό να εμφανίζονται τα προϊόντα των παραγγελιών, του «καλαθιού» και όλων τον προϊόντων αντίστοιχα.

*Όλα τα παραπάνω βήματα σχεδιάστηκαν στο Visual layout Editor.
Ο Visual layout Editor του Android Studio μας δίνει τη δυνατότητα να δημιουργούμε layouts σχεδιάζοντας με drag and drop components στην οθόνη, αντί να γράφουμε με το χέρι το layout XML.

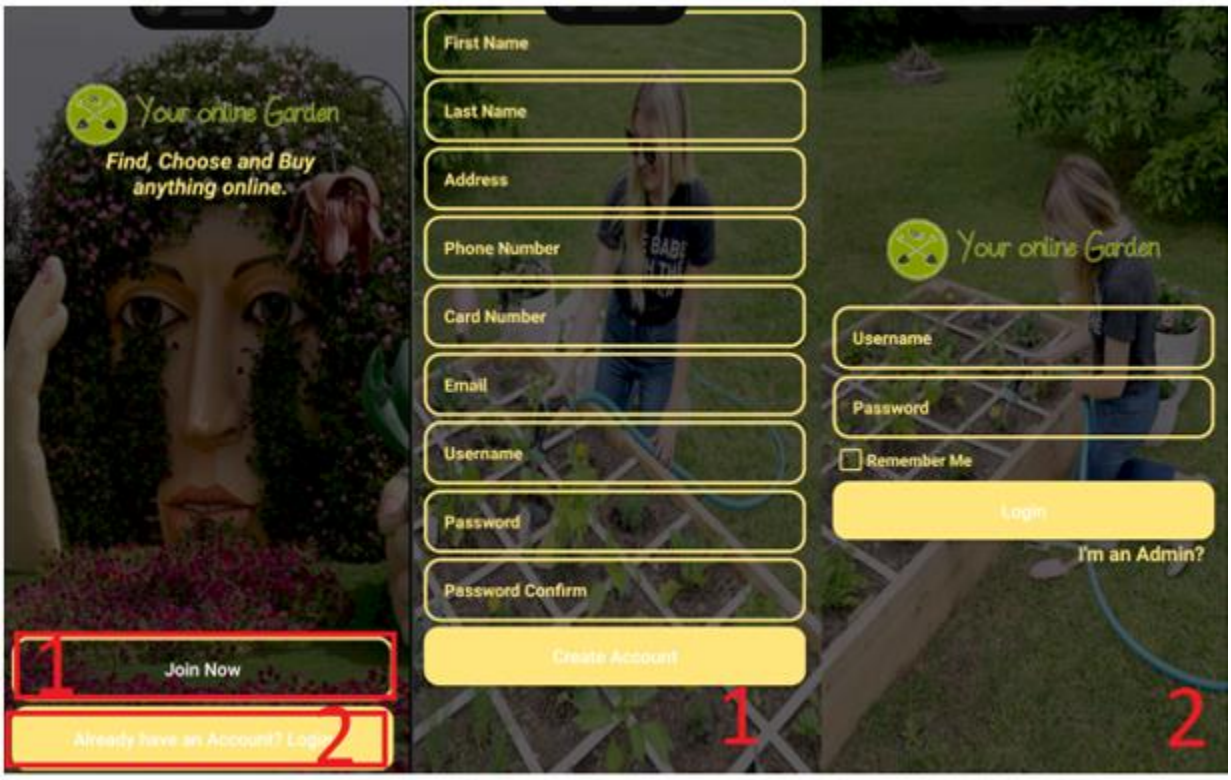
4.3 Υλοποίηση

Στην συνέχεια μετά το πέρας του σχεδιασμού της διεπαφής της εφαρμογής, όπως αναγράφεται παραπάνω ξεκίνησα την φάση ανάπτυξης της εφαρμογής σε προγραμματιστικό επίπεδο. Πρώτα δημιουργώ και συνδέω το Firebase Database στην εφαρμογή μου με τον τρόπο που φαίνεται παρακάτω και στο παράρτημα 1.



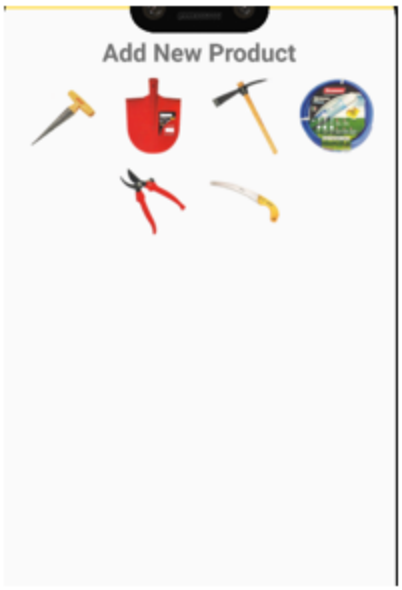
[ΕΙΚΟΝΑ 36 | FIREBASE DATABASE]

Στην συνέχεια, υλοποίησα τις κλάσεις RegisterActivity και LoginActivity με τις οποίες ο χρήστης θα μπορεί να εγγραφτεί και να κάνει είσοδο στην εφαρμογή όπως φαίνεται στο παράρτημα 2 αλλά και στην εικόνα 37 αντίστοιχα. Επιπλέον πρόσθεσα την δυνατότητα με την οποία η εφαρμογή θα μπορεί να «θυμάται» το username και το password του χρήστη με την χρήση της βιβλιοθήκης PaperDB.



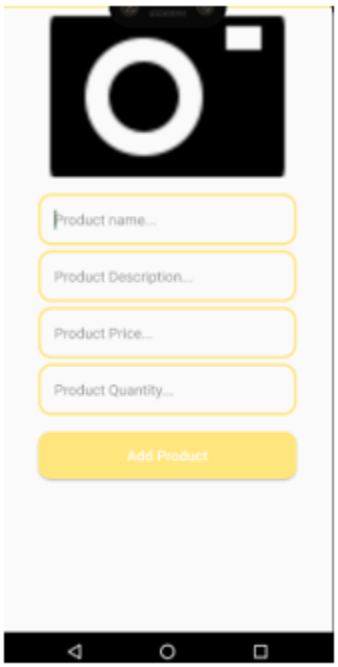
[ΕΙΚΟΝΑ 37| MAIN, REGISTER AND LOGIN ACTIVITY]

Έπειτα προγραμμάτισα μια από τις επιλογές του διαχειριστή. Ο διαχειριστής θα μπορεί να εισάγει νέα προϊόντα στην κατηγορία με το συγκεκριμένο όνομα (το οποίο «κλικάρει»). Για να γίνει αυτό έπρεπε να περάσω μια τιμή από ένα activity σε ένα άλλο. Έστειλα λοιπόν το όνομα της κατηγορίας που επιλέγει ο διαχειριστής στην κλάση AddCategoryActivity όπως φαίνεται στο παράρτημα 3 και εικόνα 38 αντίστοιχα.



[ΕΙΚΟΝΑ 38| ADMIN CATEGORY ACTIVITY]

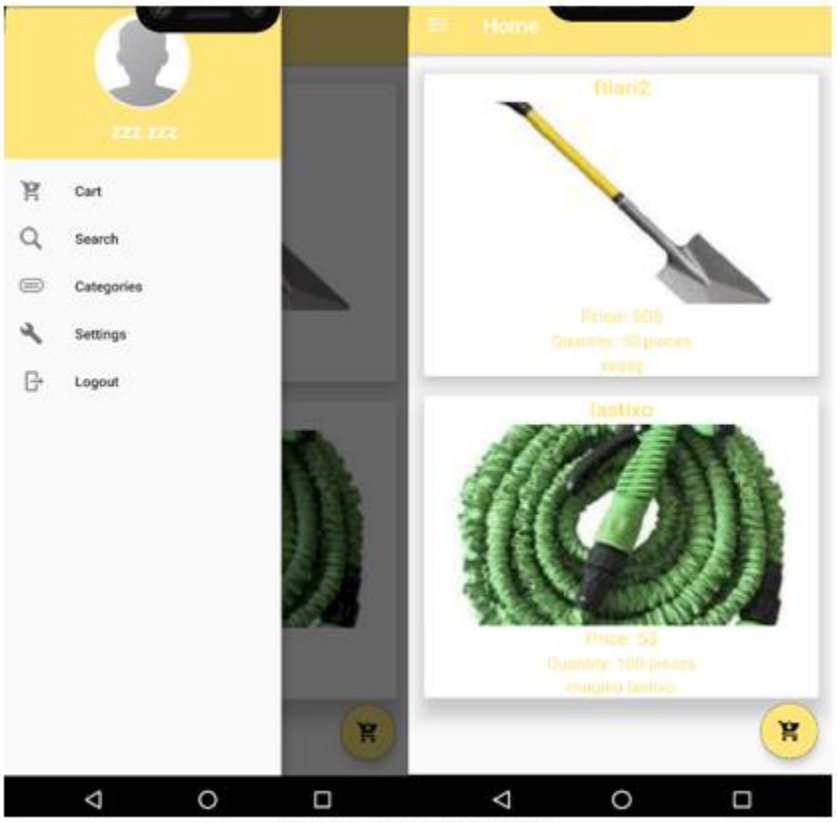
Επόμενο βήμα ήταν η να προγραμματίσουμε την AddNewProductActivity (στην οποία θα δίνονται τα στοιχεία του προϊόντος που είναι προς εισαγωγή) με την βοήθεια της τιμής που περάσαμε, του Firebase Database αλλά και του Firebase Storage όπως φαίνεται στο παράρτημα 4 και εικόνα 39 αντίστοιχα. Ακόμη θα υλοποιήσουμε την μέθοδο getDownloadUrl().



[ΕΙΚΟΝΑ 39| ADD NEW PRODUCT ACTIVITY]

Το αμέσως επόμενο βήμα ήταν να προσθέσω ένα Android Navigation Drawer και να το προγραμματίσω ώστε να εμφανίζει ως header το όνομα του χρήστη. Μετά προγραμμάτισα το

HomeActivity ώστε να εμφανίζονται όλα τα προϊόντα στον χρήστη όπως φαίνεται στην εικόνα 40 αλλά και παράρτημα 5 αντίστοιχα, χρησιμοποιώντας το Firebase Recycler Adapter και το Firebase RecyclerView τα οποία θα κάνουν retrieve τα data από την βάση και θα εμφανίζει τα προϊόντα μαζί με τα στοιχεία τους.



[ΕΙΚΟΝΑ 40| HOME ACTIVITY]

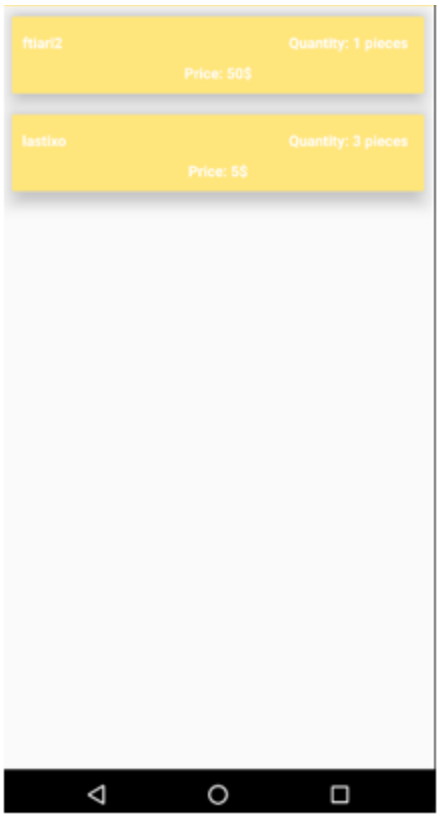
Στην συνέχεια υλοποίησα την κλάση SettingsActivity δηλαδή την επιλογή του μενού με το όνομα «Settings», η οποία θα πρέπει να κάνει ανάκτηση δεδομένων από την βάση αλλά και θα επιτρέπει στον χρήστη να ανεβάζει μια φωτογραφία και να την αποθηκεύει στην βάση με την βοήθεια του Firebase Storage. Αυτό φαίνεται στην εικόνα 41 και παράρτημα 6.

[EIKONA 41| SETTINGS ACTIVITY]

Επόμενο βήμα ήταν να προγραμματίσω την κλάση ProductDetailsActivity και το κουμπί της επιλογής του προϊόντος το οποίο όταν πατείτε θα ατακτεί της πληροφορίες του συγκεκριμένου προϊόντος από την βάση όπως φαίνεται στην εικόνα 42 και παράρτημα 7 αντίστοιχα. Επιπλέον ο χρήστης θα έχει την δυνατότητα αν θέλει να το βάζει στο καλάθι αγορών για να το αγοράσει.

[EIKONA 42| PRODUCT DETAILS ACTIVITY]

Επόμενο βήμα είναι να ατακτεί και να εμφανίζει την λίστα από τα προϊόντα που έχει βάλει ο χρήστης στο καλάθι του μέσω της κλάσης CartActivity (τα οποία έχουν κρατηθεί αποθηκευμένα στην βάση. Επίσης το προγραμμάτισα έτσι ώστε ο χρήστης να μπορεί να επεξεργαστεί η να διαγράψει προϊόντα από το καλάθι του με την βοήθεια της Real Time βάσης Firebase). Μία άλλη δυνατότητα που έδωσα είναι να υπολογίζεται το συνολικό ποσό/τιμή των προϊόντων που υπάρχουν στο καλάθι (η οποία θα εμφανίζεται αφού πατηθεί το κουμπί της «συνέχειας») όπως φαίνεται στην εικόνα 43 και το παράρτημα 7 αντίστοιχα.

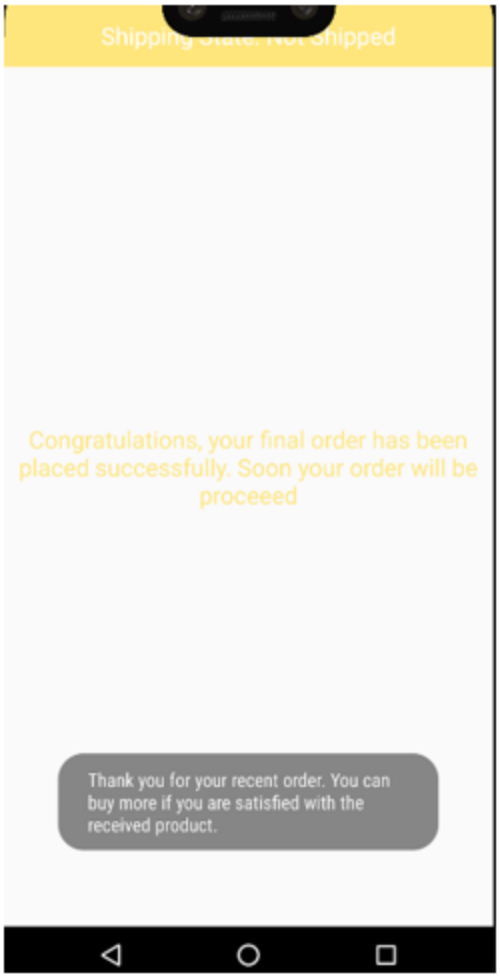


[ΕΙΚΟΝΑ 43| CART ACTIVITY]

Στην συνέχεια προγραμμάτισα την κλάση ConfirmFinalOrderActivity στην οποία ο χρήστης θα έχει την δυνατότητα να εγκρίνει την τελική του παραγγελία για να την στείλει για επιβεβαίωση στον διαχειριστή. Για να γίνει αυτό όμως προηγείται η συμπλήρωση και η αποστολή στον διαχειριστή, των στοιχείων της διεύθυνσης αποστολής του χρήστη όπως φαίνεται στην εικόνα 44 και παράρτημα 8.

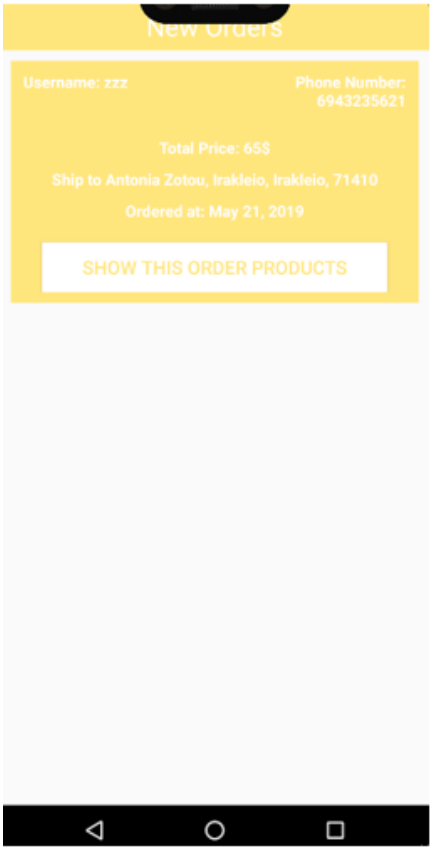
[ΕΙΚΟΝΑ 44| CONFIRM FINAL ORDER ACTIVITY]

Το επόμενο βήμα είναι να γυρίσουμε κάποια βήματα πίσω και να πάμε πάλι στο CartActivity ώστε να προσθέσουμε ελέγχους για να ελέγχετε το αν ο διαχειριστής έχει εγκρίνει μια παραγγελία και αντίστοιχα να εμφανίζεται το ανάλογο μήνυμα όπως φαίνεται στην εικόνα 45 και το παράρτημα 9.



[EIKONA 45| CART ACTIVITY VALIIDATION]

Συνεχίζω γυρίζοντας πίσω στον τομέα του διαχειριστή με σκοπό να προγραμματίσω κάποιες επιπλέον δυνατότητες. Η πρώτη που άρχισα να προγραμματίζω είναι η AdminNewOrdersActivity η οποία θα έχει την δυνατότητα να ελέγχει και θα επιβεβαιώνει, τις παραγγελίες των χρηστών. Επιπλέον θα διαγράφει από την βάση μας, τις παραγγελίες που έχουν επιβεβαιωθεί όπως φαίνεται στην εικόνα 46 και το παράρτημα 10.

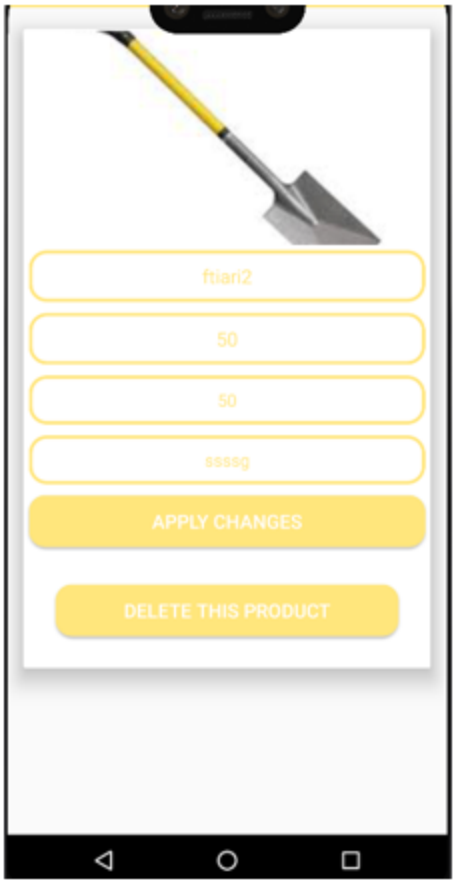


[ΕΙΚΟΝΑ 46| ADMIN NEW ORDERS ACTIVITY]

Συνεχίζοντας, προγραμμάτισα την λειτουργία του μενού με όνομα “Search”, SearchProductActivity η οποία δίνει την δυνατότητα στον χρήστη να αναζητήσει ένα προϊόν με την βοήθεια του Firebase Recycler View, Recycler Adapter αλλά και της Firebase Database όπως φαίνεται στην εικόνα 47 και το παράρτημα 11.

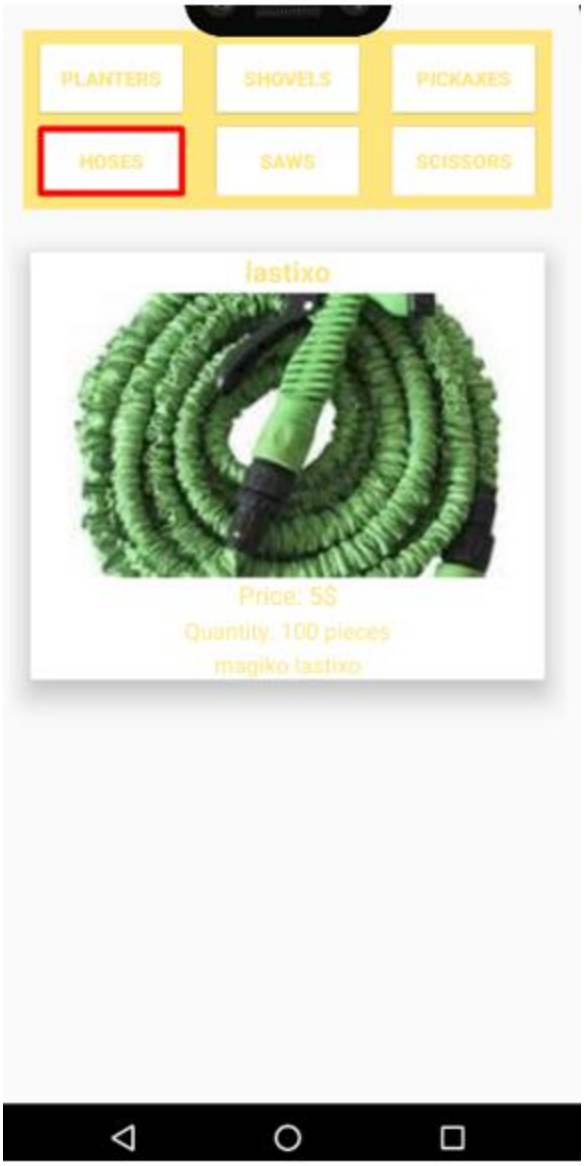
[EIKONA 47| SEARCH PRODUCT ACTIVITY]

Ως επόμενο βήμα, υλοποίησα την κλάση AdminMaintainProductsActivity του διαχειριστή η οποία του δίνει την δυνατότητα να επεξεργαστεί τις πληροφορίες ενός προϊόντος με την βοήθεια της βάσης δεδομένων η οποία θα ανακτεί τα στοιχεία του προϊόντος δίνοντας την δυνατότητα σε εμάς να την επεξεργαστούμε. Ακόμη θα μπορεί να διαγράψει ένα προϊόν όπως φαίνεται στην εικόνα 48 και παράρτημα 12.



[ΕΙΚΟΝΑ 48| ADMIN MAINTAIN PRODUCTS ACTIVITY]

Τέλος, προγραμμάτισα την λειτουργία του μενού, CategoriesActivity η οποία δίνει την ευελιξία στον χρήστη να βλέπει τα προϊόντα με βάση το κουμπί της κατηγορίας το οποίο θα επιλέξει. Αυτό επιτεύχθηκε με την βοήθεια του Firebase Recycler View, Recycler Adapter αλλά και της Firebase Database όπως φαίνεται στην εικόνα 49 και το παράρτημα 13.



[EIKONA 49| CATEGORIES ACTIVITY]

5. Αποτελέσματα

5.1 Συμπεράσματα

Οι στόχοι υλοποίησης της εφαρμογής έχουν επιτευχθεί. Στην εφαρμογή μας αποθηκεύουμε σημαντικές πληροφορίες που θα αναπτυχθούν παρακάτω, καθώς και να μπορεί εκπληρώσει πολύ σημαντικές λειτουργίες. Ο λόγος που ανέπτυξα την εφαρμογή στο Android Studio είχε ως στόχο να μπορεί να λειτουργήσει σε συσκευές Android, ο κύριος λόγος είναι ότι το ποσοστό της χρήσης αυτού του λειτουργικού συστήματος σε σύγκριση με άλλα λειτουργικά συστήματα στην αγορά. Ως εκ τούτου, είναι μια βολική εφαρμογή που καλύπτει τις ανάγκες των αγοραστών έτσι ώστε η καθημερινή ζωή τους γίνεται απλούστερη και ευκολότερη χωρίς στρες και ταλαιπωρία.

Ειδικότερα, έπρεπε να αποθηκευτούν οι ακόλουθες πληροφορίες για :

- τον κάθε χρήστη του συστήματος δηλαδή, πλήρες όνομα, διεύθυνση, τηλέφωνο, αριθμό πιστωτικής, username και ένα password.
- τις πληροφορίες κάθε προϊόντος δηλαδή όνομα, τιμή, ποσότητα, περιγραφή και τη κατηγορία στην οποία ανήκει.
- τα προϊόντα που έχει κάθε χρήστης στο «καλάθι αγορών του».
- τις παραγγελίες που πρέπει να «επιβεβαιώσει» ο διαχειριστής.

Το σύστημα υποστηρίζει τις λειτουργίες που είναι:

- η εγγραφή ενός χρήστη. Ο χρήστης δίνοντας τα απαραίτητα στοιχεία και δίνοντας ένα δικό του username και password, το σύστημα θα ελέγχει αν είναι unique το username και αν δεν είναι τότε η εγγραφή του χρήστη θα γίνεται επιτυχώς.
- η είσοδος ως διαχειριστής. Ο διαχειριστής μπορει να εισάγει και να επεξεργαστεί ένα προϊόν αλλά και να «επιβεβαιώσει» τις παραγγελίες των χρηστών.
- η είσοδος ως χρήστης. Ο χρήστης έχει το προνόμιο να βλέπει όλα τα προϊόντα. Να ανοίγει ένα μενού στο οποίο θα έχει κάποιες επιλογές οι οποίες θα μπορούν να τον οδηγήσουν είτε στο καλάθι αγορών, είτε σε ένα instance που θα μπορεί να αναζήτηση ένα προϊόν, είτε σε ένα instance που θα μπορεί να δει τα προϊόντα ανά κατηγορία, είτε στις ρυθμίσεις όπου θα μπορεί να αλλάξει κάποιες πληροφορίες του προφίλ του. Ο χρήστης πατώντας ένα προϊόν θα μπορεί να επιλέξει την ποσότητα που θέλει να εισάγει στο καλάθι αγορών, έπειτα θα πηγαίνει στο καλάθι αγορών του και θα συνεχίζει την παραγγελία εισάγοντας την διεύθυνση αποστολής και τέλος με το κουμπί ολοκλήρωσης παραγγελίας θα στέλνετε η παραγγελία του χρήστη στον διαχειριστή για επιβεβαίωση.

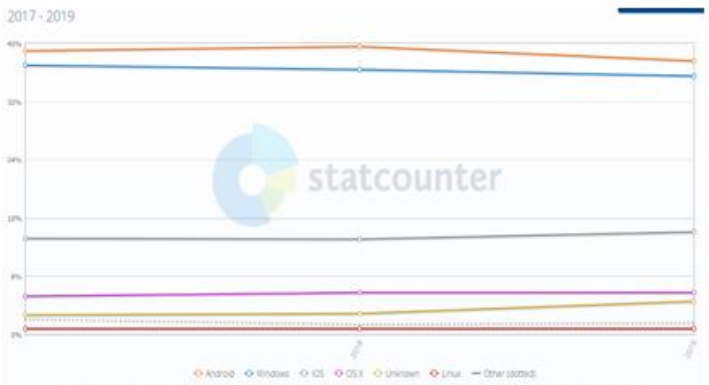
5.2 Μελλοντική Εργασία και Επεκτάσεις

Για κάθε εφαρμογή, υπάρχουν τρόποι βελτίωσης, οι δική μου θα αναλυθούν παρακάτω:

- Δυνατότητα «Ξέχασα τον κωδικό μου».
  Δηλαδή, να υπάρχει η δυνατότητα να μπορεί ο χρήστης δίνοντας κάποια στοιχεία να μπορεί να ανακτήσει τον κωδικό πρόσβασης του, ο οποίος θα στέλνετε είτε στο κινητό είτε στο email του.

- Δυνατότητα πληρωμής και αποστολής στοιχείων.
  Δηλαδή ο χρήστης, να έχει την δυνατότητα να πληρώσει μια παραγγελία με διάφορους μεθόδους όπως για παράδειγμα πιστωτική κάρτα, PayPal, e-wallet. Επιπλέον όταν γίνεται η παραγγελία να αποστέλλετε στον χρήστη email η και μήνυμα στο κινητό του τηλέφωνο με τις πληροφορίες της παραγγελίας.

- Δυνατότητα εγκατάστασης σε άλλα λογισμικά.
  Δηλαδή, θα μπορεί η εφαρμογή να εγκατασταθεί και σε συσκευές με άλλο λογισμικό εκτός του Android όπως για παράδειγμα iOS.



[ΕΙΚΟΝΑ 50 | ΓΡΑΦΗΜΑ ΧΡΟΝΟΛΟΓΙΩΝ ΛΕΙΤΟΥΡΓΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ 2017 - 2019]

Όπως μπορούμε να δούμε (στην εικόνα 49) από το διαγραμμα ημερομηνιας 2017-2019, που απεικονίζει τα αποτελέσματα της έρευνας και δείχνει τα στατιστικά στοιχεία σε ποσοστά ανάλογα με τη χρήση των συσκευών λογισμικού, θα πρέπει να τονιστεί ότι το ποσοστό των χρηστών με Android που είναι ενεργοί περισσότεροι από τα υπόλοιπα. Ωστόσο, έχουμε καλά ποσοστά και στα άλλα λογισμικά, έτσι θα ήταν ευεργετικό να επεκταθεί η εφαρμογή μας σε άλλες συσκευές εκτός από το Android, έτσι ώστε και άλλοι χρήστες με άλλο λογισμικό να μπορούν να το χρησιμοποιήσουν.

Πίνακες

Πίνακας εικόνων

Βιβλιογραφία

Android Studio, “Android Studio provides the fastest tools for building apps on every type of Android device”, (https://developer.android.com/studio/), (14/11/2018)

hdodenhof, “A circular ImageView for Android”, (https://github.com/hdodenhof/CircleImageView), (06/01/2019)

ThisIsAreku and pilgr, “Paper is a fast NoSQL-like storage for Java/Kotlin objects on Android with automatic schema migration support.”, (https://github.com/pilgr/Paper), (27/03/2019)

ArthurHub, “Image Cropping Library for Android, optimized for Camera / Gallery”, (https://github.com/ArthurHub/Android-Image-Cropper), (08/02/2019)

Ashik Vetrivelu, “A simple Android library to implement a number counter with increment and decrement buttons”, (https://android-arsenal.com/details/1/4136), (10/08/2016)

GeekyAnts, “Introduction to Firebase”, (https://hackernoon.com/introduction-to-firebase-218a23186cd7), (28/12/2017)

Lawrence A. Cunningham (2005). "Language, Deals and Standards: The Future of XML Contracts". Washington University Law Review. SSRN 900616

Kelly, Sean (February 6, 2006). "Making Mistakes with XML". Developer.com. Retrieved 26 October 2010.

St. Laurent, Simon (February 12, 2003). "Five years later, XML." O'Reilly XML Blog. O'Reilly Media. Retrieved 26 October 2010.

"W3C XML is Ten!". World Wide Web Consortium. 12 February 2008. Retrieved 26 October 2010.

Gosling, James; Joy, Bill; Steele, Guy L., Jr.; Bracha, Gilad (2005). The Java Language Specification (3rd ed.). Addison-Wesley. ISBN 0-321-24678-0.

Lindholm, Tim; Yellin, Frank (1999). The Java Virtual Machine Specification (2nd ed.). Addison-Wesley. ISBN 0-201-43294-3.

Παραρτήματα της Πτυχιακής Εργασίας

Παράρτημα 1: Firebase Database

MainActivity.class

```
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

Build.gradle(app)

```
dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation 'com.android.support:appcompat-v7:28.0.0'
    implementation 'com.android.support.constraint:constraint-l
    implementation 'com.google.firebase:firebase-core:16.0.1'
    testImplementation 'junit:junit:4.12'
    androidTestImplementation 'com.android.support.test:runner:
    androidTestImplementation 'com.android.support.test.espress
}
```

Παράρτημα 2: Main & Register & Login Activities

MainActivity

```
loginButton.setOnClickListener((v) → {
        Intent intent = new Intent( packageContext: MainActivity.this, LoginActivity.class);
        startActivity(intent);
});

joinNowButton.setOnClickListener((v) → {
        Intent intent = new Intent( packageContext: MainActivity.this, RegisterActivity.class);
        startActivity(intent);
});
```

RegisterActivity

```
final DatabaseReference RootRef;
RootRef = FirebaseDatabase.getInstance().getReference();

RootRef.addListenerForSingleValueEvent(new ValueEventListener()
{
    @Override
    public void onDataChange(@NonNull DataSnapshot dataSnapshot)
    {
        if( (!(dataSnapshot.child("Users").child("username").exists())) )
        {
            HashMap<String, Object> userdataMap = new HashMap<>();
            userdataMap.put("firstname", firstname);
            userdataMap.put("lastname", lastname);
            userdataMap.put("address", address);
            userdataMap.put("phone", phone);
            userdataMap.put("cardnumber", cardnumber);
            userdataMap.put("email", email);
            userdataMap.put("username", username);
            userdataMap.put("password", password);
            userdataMap.put("image", "no image");

            RootRef.child("Users").child(username).updateChildren(userdataMap).addOnCompleteListener((task) → {
                    if (task.isSuccessful())
                    {
                        Toast.makeText( context: RegisterActivity.this, text: "Congratulations, your account has been created.",Toast.LENGTH_LONG).show
                        loadingBar.dismiss();

                        Intent intent = new Intent( packageContext: RegisterActivity.this, LoginActivity.class);
                        startActivity(intent);
                    }
                    else
                    {
                        loadingBar.dismiss();
                        Toast.makeText( context: RegisterActivity.this, text: "Network Error: Please try again.",Toast.LENGTH_LONG).show();
                    }
                });
        }
```

```
    else
    {
        Toast.makeText( context: RegisterActivity.this,  text: "This" +username+ "already exists.", Toast.LENGTH_LONG);
        loadingBar.dismiss();
        Toast.makeText( context: RegisterActivity.this, text: "Pease try again using another phone number or username",Toast.LENGTH_LONG);

        Intent intent = new Intent( packageContext: RegisterActivity.this, MainActivity.class);
        startActivity(intent);
    }
}
```

LoginActivity

```
final DatabaseReference RootRef;
RootRef = FirebaseDatabase.getInstance().getReference();

RootRef.addListenerForSingleValueEvent(new ValueEventListener() {
    @Override
    public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
        if ( dataSnapshot.child(parentDbName).child(username).exists() )
        {
            Users usersData = dataSnapshot.child(parentDbName).child(username).getValue(Users.class);

            if ( usersData.getUsername().equals(username) )
            {
                if ( usersData.getPassword().equals(password) )
                {
                    if (parentDbName.equals("Admins"))
                    {
                        Toast.makeText( context: LoginActivity.this, text: "Welcome Admin you are, logged in Successfully...",Toast.LENGTH_LONG).show(
                        loadingBar.dismiss();

                        Intent intent = new Intent( packageContext: LoginActivity.this, AdminCategoryActivity.class);
                        startActivity(intent);
                    }
                    else if (parentDbName.equals("Users"))
                    {
                        Toast.makeText( context: LoginActivity.this, text: "Logged in Successfully...",Toast.LENGTH_LONG).show();
                        loadingBar.dismiss();

                        Intent intent = new Intent( packageContext: LoginActivity.this, HomeActivity.class);
                        Prevalent.currentOnlineUser = usersData;
                        startActivity(intent);
                    }
                }
                else
                {
                    Toast.makeText( context: LoginActivity.this, text: "Password is incorrect...",Toast.LENGTH_LONG).show();
                    loadingBar.dismiss();
                }
            }
```

```
        }
        else
        {
            Toast.makeText( context: LoginActivity.this, text: "Account with username "+ username + " doesn't exists.",Toast.LENGTH_LONG).show();
            loadingBar.dismiss();
        }
    }
```

Παράρτημα 3: AdminCatergoryActivity

AdminCategoryActivity

```
fiteutiria.setOnClickListener((v) → {
        Intent intent = new Intent( packageContext: AdminCategoryActivity.this,AdminAddNewProductActivity.class);
        intent.putExtra( name: "category",  value: "fiteutiria");
        startActivity(intent);
});

ftiaria.setOnClickListener((v) → {
        Intent intent = new Intent( packageContext: AdminCategoryActivity.this,AdminAddNewProductActivity.class);
        intent.putExtra( name: "category",  value: "ftiaria");
        startActivity(intent);
});
```

Παράρτημα 4: AddNewProductActivity

AddNewProductActivity

```
private void SaveProductInfoToDatabase()
{
    HashMap<String, Object> productMap = new HashMap<>();
    productMap.put("pid", productRandomKey);
    productMap.put("date", saveCurrentDate);
    productMap.put("time", saveCurrentTime);
    productMap.put("description", Description);
    productMap.put("image", downloadImageUrl);
    productMap.put("category", CategoryName);
    productMap.put("price", Price);
    productMap.put("pname", ProductName);
    productMap.put("quantity", Quantity);

    ProductsRef.child(productRandomKey).updateChildren(productMap).addOnCompleteListener((task) → {
            if (task.isSuccessful())
            {
                Intent intent = new Intent( packageContext: AdminAddNewProductActivity.this, AdminCategoryActivity.class);
                startActivity(intent);

                loadingBar.dismiss();
                Toast.makeText( context: AdminAddNewProductActivity.this,  text: "Product is added Successfully...", Toast.LENGTH_SHORT).show();
            }
            else
            {
                loadingBar.dismiss();
                String message = task.getException().toString();
                Toast.makeText( context: AdminAddNewProductActivity.this,  text: "Error: " + message, Toast.LENGTH_SHORT).show();

            }
    });
}
```

Παράρτημα 5: HomeActivity

HomeActivity

Show all the products from the Database

```
@Override
protected void onStart() {
    super.onStart();

    FirebaseRecyclerOptions<Products> options =
            new FirebaseRecyclerOptions.Builder<Products>()
            .setQuery(ProductsRef, Products.class)
            .build();

    FirebaseRecyclerAdapter<Products, ProductViewHolder> adapter =
            new FirebaseRecyclerAdapter<~>(options)
            {
                @Override
                protected void onBindViewHolder(@NonNull ProductViewHolder holder, int position, @NonNull final Products model)
                {
                    holder.txtProductName.setText(model.getPname());
                    holder.txtProductDescription.setText(model.getDescription());
                    holder.txtProductPrice.setText("Price: " + model.getPrice() + "$");
                    holder.txtProductQuantity.setText("Quantity: " + model.getQuantity() + " pieces");
                    Picasso.get().load(model.getImage()).into(holder.imageView);

                    holder.itemView.setOnClickListener((v) -> {
                            if ( type.equals("Admin"))
                            {
                                Intent intent = new Intent( packageContext: HomeActivity.this, AdminMaintainProductsActivity.class);
                                intent.putExtra( name: "pid", model.getPid());
                                startActivity(intent);
                            }
                            else
                            {
                                Intent intent = new Intent( packageContext: HomeActivity.this, ProductDetailsActivity.class);
                                intent.putExtra( name: "pid", model.getPid());
                                startActivity(intent);
                            }
                        });
                    }

                @NonNull
                @Override
                public ProductViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int i)
                {
                    View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.product_items_layout, parent,  attachToRoot: false);
                    ProductViewHolder holder = new ProductViewHolder(view);
                    return holder;
                }
            };
    recyclerView.setAdapter(adapter);
    adapter.startListening();
}
```

Παράρτημα 6: SettingsActivity

SettingsActivity

Display the user info

```
private void userInfoDisplay(final CircleImageView profileImageView, final EditText firstNameEditText, final EditText lastNameEditText, final EditText addre
{
    DatabaseReference UsersRef = FirebaseDatabase.getInstance().getReference().child("Users").child(Prevalent.currentOnlineUser.getUsername());

    UsersRef.addValueEventListener(new ValueEventListener()
    {
        @Override
        public void onDataChange(@NonNull DataSnapshot dataSnapshot)
        {
            if ( dataSnapshot.exists() )
            {
                if ( dataSnapshot.child("image").exists() )
                {
                    String image = dataSnapshot.child("image").getValue().toString();
                    String firstname = dataSnapshot.child("firstname").getValue().toString();
                    String lastname = dataSnapshot.child("lastname").getValue().toString();
                    String address = dataSnapshot.child("address").getValue().toString();
                    String phone = dataSnapshot.child("phone").getValue().toString();
                    String cardnumber = dataSnapshot.child("cardnumber").getValue().toString();
                    String password = dataSnapshot.child("password").getValue().toString();

                    Picasso.get().load(image).into(profileImageView);
                    firstNameEditText.setText(firstname);
                    lastNameEditText.setText(lastname);
                    addressEditText.setText(address);
                    phoneEditText.setText(phone);
                    cardnumberEditText.setText(cardnumber);
                    passwordEditText.setText(password);
                    passwordconfEditText.setText(password);

                }
            }
        }
```

Update user info

```
    DatabaseReference ref = FirebaseDatabase.getInstance().getReference().child("Users");

    HashMap<String, Object> userMap = new HashMap<>();
    userMap.put("firstname", firstNameEditText.getText().toString());
    userMap.put("lastname", lastNameEditText.getText().toString());
    userMap.put("address", addressEditText.getText().toString());
    userMap.put("phone", firstNameEditText.getText().toString());
    userMap.put("cardnumber", cardnumberEditText.getText().toString());
    userMap.put("password", passwordEditText.getText().toString());
    ref.child(Prevalent.currentOnlineUser.getUsername()).updateChildren(userMap);

    startActivity(new Intent( packageContext: SettingsActivity.this, HomeActivity.class));
    Toast.makeText( context: SettingsActivity.this,  text: "Profile info update successfully", Toast.LENGTH_SHORT).show();
    finish();
}
```

Παράρτημα 7: CartActivity

CartActivity

```
@Override
protected void onStart()
{
    super.onStart();

    CheckOrderState();

    final DatabaseReference cartListRef = FirebaseDatabase.getInstance().getReference().child("Cart List");

    FirebaseRecyclerOptions<Cart> options =
            new FirebaseRecyclerOptions.Builder<Cart>()
            .setQuery(cartListRef.child("User View")
                    .child(Prevalent.currentOnlineUser.getUsername()).child("Products"), Cart.class)
                    .build();

    FirebaseRecyclerAdapter<Cart, CartViewHolder> adapter
            = new FirebaseRecyclerAdapter<~>(options)
    {
        @Override
        protected void onBindViewHolder(@NonNull CartViewHolder holder, int position, @NonNull final Cart model)
        {
            holder.txtProductQuantity.setText("Quantity: " + model.getQuantity() + " pieces");
            holder.txtProductPrice.setText("Price: " + model.getPrice() + "$");
            holder.txtProductName.setText(model.getPname());

            int oneTypeProductTPrice = (Integer.valueOf(model.getPrice())) * (Integer.valueOf(model.getQuantity()));
            overTotalPrice = overTotalPrice + oneTypeProductTPrice;
```

```
  holder.itemView.setOnClickListener((v) → {
            CharSequence options[] = new CharSequence[]
                    {
                            "Edit",
                            "Remove"
                    };
            AlertDialog.Builder builder = new AlertDialog.Builder( context: CartActivity.this);
            builder.setTitle("Cart Options");

            builder.setItems(options, (dialog, i) → {
                    if (i == 0)
                    {
                        Intent intent = new Intent( packageContext: CartActivity.this, ProductDetailsActivity.class);
                        intent.putExtra( name: "pid", model.getPid());
                        startActivity(intent);
                    }
                    if (i == 1)
                    {
                        cartListRef.child("User View")
                                .child(Prevalent.currentOnlineUser.getUsername())
                                .child("Products")
                                .child(model.getPid())
                                .removeValue()
                                .addOnCompleteListener((task) → {
                                        if (task.isSuccessful())
                                        {
                                            Toast.makeText( context: CartActivity.this,  text: "Item removed successfully...", Toast.LENGTH_SHORT).show();
                                        }
                                });
                    }
            });
            builder.show();
    });
  }
```

Παράρτημα 8: ConfirmFinalOrderActivity

ConfirmFinalOrderActivity

```
private void ConfirmOrder()
{
    final String saveCurrentDate, saveCurrentTime;

    Calendar calForDate = Calendar.getInstance();
    SimpleDateFormat currentDate = new SimpleDateFormat( pattern: "MMM dd, yyyy");
    saveCurrentDate = currentDate.format(calForDate.getTime());

    SimpleDateFormat currentTime = new SimpleDateFormat( pattern: "HH:mm:ss a");
    saveCurrentTime = currentDate.format(calForDate.getTime());

    final DatabaseReference ordersRef = FirebaseDatabase.getInstance().getReference()
            .child("Orders")
            .child(Prevalent.currentOnlineUser.getUsername());

    HashMap<String, Object> ordersMap = new HashMap<>();

    ordersMap.put("totalAmount", totalAmount);
    ordersMap.put("name" , nameEditText.getText().toString());
    ordersMap.put("phone", phoneEditText.getText().toString());
    ordersMap.put("address", addressEditText.getText().toString());
    ordersMap.put("city", cityEditText.getText().toString());
    ordersMap.put("postalcode", postalCodeEditText.getText().toString());
    ordersMap.put("date", saveCurrentDate);
    ordersMap.put("time", saveCurrentTime);
    ordersMap.put("state", "not shipped");
    ordersMap.put("username", Prevalent.currentOnlineUser.getUsername());

    ordersRef.updateChildren(ordersMap).addOnCompleteListener((task) → {
```

```
        if (task.isSuccessful())
        {
            FirebaseDatabase.getInstance().getReference()
                    .child("Cart List")
                    .child("User View")
                    .child(Prevalent.currentOnlineUser.getUsername())
                    .removeValue()
                    .addOnCompleteListener((task) → {
                            if (task.isSuccessful())
                            {
                                Toast.makeText( context: ConfirmFinalOrderActivity.this,  text: "Your final order has been placed successfully...", Toast.LEN

                                Intent intent = new Intent( packageContext: ConfirmFinalOrderActivity.this, HomeActivity.class);
                                intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TASK | Intent.FLAG_ACTIVITY_CLEAR_TASK);
                                startActivity(intent);
                            }
                    });
        }
    });
}
```

Παράρτημα 9: Update CartActivity

CartActivity

Check if order has been delivered

```
private void CheckOrderState()
{
    DatabaseReference ordersRef;
    ordersRef = FirebaseDatabase.getInstance().getReference().child("Orders").child(Prevalent.currentOnlineUser.getUsername());

    ordersRef.addValueEventListener(new ValueEventListener()
    {
        @Override
        public void onDataChange(@NonNull DataSnapshot dataSnapshot)
        {
            if (dataSnapshot.exists())
            {
                String shippingState = dataSnapshot.child("state").getValue().toString();
                String fullName = dataSnapshot.child("name").getValue().toString();

                if ( shippingState.equals("shipped") )
                {
                    txtTotalAmount.setText("Dear " + fullName + "\n order is shipped successfully.");
                    recyclerView.setVisibility(View.GONE);

                    txtMsg1.setVisibility(View.VISIBLE);
                    txtMsg1.setText("Congratulations, your final order has been placed successfully. Soon you will receive your order.");
                    NextProcessBtn.setVisibility(View.GONE);

                    Toast.makeText( context: CartActivity.this, text: "Thank you for your recent order. You can buy more if you are satisfied with the received p
                }
                else if ( shippingState.equals("not shipped") )
                {
                    txtTotalAmount.setText("Shipping State: Not Shipped");
                    recyclerView.setVisibility(View.GONE);

                    txtMsg1.setVisibility(View.VISIBLE);
                    NextProcessBtn.setVisibility(View.GONE);

                    Toast.makeText( context: CartActivity.this, text: "Thank you for your recent order. You can buy more if you are satisfied with the received p
                }
            }
```

Παράρτημα 10: AdminNewOrdersActivity

AdminNewOrdersActivity

```
@Override
protected void onStart()
{
    super.onStart();

    FirebaseRecyclerOptions<AdminOrders> options =
            new FirebaseRecyclerOptions.Builder<AdminOrders>()
            .setQuery(ordersRef, AdminOrders.class)
            .build();

    FirebaseRecyclerAdapter<AdminOrders, AdminOrdersViewHolder> adapter =
            new FirebaseRecyclerAdapter<~>(options)
            {
                @Override
                protected void onBindViewHolder(@NonNull AdminOrdersViewHolder holder, final int position, @NonNull final AdminOrders model)
                {
                    holder.userUsername.setText("Username: " + model.getUsername());
                    holder.userPhoneNumber.setText("Phone Number: " + model.getPhone());
                    holder.userTotalPrice.setText("Total Price: "  + model.getTotalAmount() + "$");
                    holder.userAddressCityPc.setText("Ship to " + model.getName() + ", " + model.getAddress() + ", " + model.getCity() + ", " + model.getPos
                    holder.userDateTime.setText("Ordered at: " + model.getDate());

                    holder.showOrdersBtn.setOnClickListener((v) → {
                            String uUsername = getRef(position).getKey();

                            Intent intent = new Intent( packageContext: AdminNewOrdersActivity.this, AdminUserProductsActivity.class);
                            intent.putExtra( name: "userUsername", uUsername);
                            startActivity(intent);
                    });

                    holder.itemView.setOnClickListener((v) → {
                            CharSequence options[] = new CharSequence[]
                                    {
                                            "Yes",
                                            "No"
                                    };

                            AlertDialog.Builder builder = new AlertDialog.Builder( context: AdminNewOrdersActivity.this);
                            builder.setTitle("Have you shipped this order?");

                            builder.setItems(options, (dialogInterface, i) → {
                                    if ( i == 0)
                                    {
                                        String uUsername = getRef(position).getKey();
                                        RemoveOrder(uUsername);
                                    }
                                    else
                                    {
                                        finish();
                                    }
                            });
                            builder.show();
                    });
                }

                @NonNull
                @Override
                public AdminOrdersViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType)
                {
                    View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.orders_layout, parent, attachToRoot: false)
                    return new AdminOrdersViewHolder(view);
                }
            };
    ordersList.setAdapter(adapter);
    adapter.startListening();
}
```

Παράρτημα 11: SearchProductActivity

SearchProductActivity

```
@Override
protected void onStart()
{
    super.onStart();

    DatabaseReference reference = FirebaseDatabase.getInstance().getReference().child("Products");

    FirebaseRecyclerOptions<Products> options =
            new FirebaseRecyclerOptions.Builder<Products>()
            .setQuery(reference.orderByChild("pname").startAt(searchInput), Products.class)
            .build();

    FirebaseRecyclerAdapter<Products, ProductViewHolder> adapter =
            new FirebaseRecyclerAdapter<~>(options)
            {
                @Override
                protected void onBindViewHolder(@NonNull ProductViewHolder holder, int position, @NonNull final Products model)
                {
                    holder.txtProductName.setText(model.getPname());
                    holder.txtProductDescription.setText(model.getDescription());
                    holder.txtProductPrice.setText("Price: " + model.getPrice() + "$");
                    holder.txtProductQuantity.setText("Quantity: " + model.getQuantity() + " pieces");
                    Picasso.get().load(model.getImage()).into(holder.imageView);

                    holder.itemView.setOnClickListener((v) → {
                            Intent intent = new Intent( packageContext: SearchProductsActivity.this, ProductDetailsActivity.class);
                            intent.putExtra( name: "pid", model.getPid());
                            startActivity(intent);
                        });
                }

                @NonNull
                @Override
                public ProductViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType)
                {
                    View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.product_items_layout, parent, attachToRoot: false);
                    ProductViewHolder holder = new ProductViewHolder(view);
                    return holder;
                }
            };
    searchList.setAdapter(adapter);
    adapter.startListening();
}
```

Παράρτημα 12: AdminMaintainProductsActivity

AdminMaintainProducts

Display products info

```
private void displaySpecificProductInfo()
{
    productsRef.addValueEventListener(new ValueEventListener()
    {
        @Override
        public void onDataChange(@NonNull DataSnapshot dataSnapshot)
        {
            if (dataSnapshot.exists())
            {
                String prName = dataSnapshot.child("pname").getValue().toString();
                String prPrice = dataSnapshot.child("price").getValue().toString();
                String prDescription = dataSnapshot.child("description").getValue().toString();
                String prQuantity = dataSnapshot.child("quantity").getValue().toString();
                String prImage = dataSnapshot.child("image").getValue().toString();

                name.setText(prName);
                price.setText(prPrice);
                description.setText(prDescription);
                quantity.setText(prQuantity);
                Picasso.get().load(prImage).into(imageView);
            }
        }
```

Update Product Info

```
    HashMap<String, Object> productMap = new HashMap<>();
    productMap.put("pid", productID);
    productMap.put("description", pr2Description);
    productMap.put("price", pr2Price);
    productMap.put("pname", pr2Name);
    productMap.put("quantity", pr2Quantity);

    productsRef.updateChildren(productMap).addOnCompleteListener((task) → {
            if (task.isSuccessful())
            {
                Toast.makeText( context: AdminMaintainProductsActivity.this,  text: "Changes applied successfully.", Toast.LENGTH_SHORT).show();

                Intent intent = new Intent( packageContext: AdminMaintainProductsActivity.this, AdminCategoryActivity.class);
                startActivity(intent);
                finish();
            }
    });
}
```

Παράρτημα 13: CategoriesActivity

CategoriesActivity

```
private void displayCategoryProducts(final String category)
{
    DatabaseReference reference = FirebaseDatabase.getInstance().getReference().child("Products");

    FirebaseRecyclerOptions<Products> options =
            new FirebaseRecyclerOptions.Builder<Products>()
                    .setQuery(reference.orderByChild("category").equalTo(category), Products.class)
                    .build();

    FirebaseRecyclerAdapter<Products, ProductViewHolder> adapter =
            new FirebaseRecyclerAdapter<~>(options)
            {
                @Override
                protected void onBindViewHolder(@NonNull ProductViewHolder holder, int position, @NonNull final Products model)
                {
                        holder.txtProductName.setText(model.getPname());
                        holder.txtProductDescription.setText(model.getDescription());
                        holder.txtProductPrice.setText("Price: " + model.getPrice() + "$");
                        holder.txtProductQuantity.setText("Quantity: " + model.getQuantity() + " pieces");
                        Picasso.get().load(model.getImage()).into(holder.imageView);

                        holder.itemView.setOnClickListener((v) → {
                                Intent intent = new Intent( packageContext: CategoriesActivity.this, ProductDetailsActivity.class);
                                intent.putExtra( name: "pid", model.getPid());
                                startActivity(intent);
                        });

                }

                @NonNull
                @Override
                public ProductViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType)
                {
                    View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.product_items_layout, parent,  attachToRoot: false);
                    ProductViewHolder holder = new ProductViewHolder(view);
                    return holder;
                }
            };
    categoriesList.setAdapter(adapter);
    adapter.startListening();
}
```