

# Virop

Μια εφαρμογή για ασφαλής αγορές.

10.06.2020

Προηγμένα Θέματα Αλληλεπίδρασης

Καθηγητής: Ευκλείδης Κεραμόπουλος

---

Μακρυγιαννάκη Στεφανία

A M: 164703

[GitHub Repository](#) εργασίας



## Περίλιψη

Το Vinop είναι μια εφαρμογή για το κινητό, η οποία προσφέρει έναν ασφαλές και γρήγορο τρόπο πραγματοποίησης αγορών αναγκαίων προϊόντων.

## Περιεχόμενα

<b>Περίλιψη</b>	<b>1</b>
<b>Περιεχόμενα</b>	<b>1</b>
<b>Μέρος Α: δομή της εργασίας</b>	<b>3</b>
Ανοίγοντας η εφαρμογή	3
Χωρίς να έχει γίνει σύνδεση λογαριασμού	3
SharedPreferences	3
DrawerLayout	3
Home	4
Shop	4
Account	4
Πρώτο φύλλο ViewPager	4
Δεύτερο φύλλο ViewPager	5
Administrator	5
Πρώτο φύλλο ViewPager: User List	5
Δεύτερο φύλλο ViewPager: Products List	5
Τρίτο φύλλο ViewPager: Orders List	5
Cart	5
UiRefreshers	5
<b>Μέρος Β: κώδικας της εργασίας</b>	<b>6</b>
Java Επεξηγήσεις των φακέλων	6
AndroidManifest.xml	6
Φάκελος database	6
MyAppDatabase	6
MyDao	7
Φάκελος UI	7
account	7
UI	7
AccountFragment	8
AccountViewModel	8

UserDetails	8
UserOrderEditBottomSheetDialog	8
UserOrders	9
administrator	9
UI	9
AdministratorProfile	10
OrderListSheetDialog	10
OrdersList	10
ProductList	10
ProductsListSheetDialog	10
UserList	11
UIs for Home, Login and Sign up	11
login	11
signup	11
SignUpActivity	11
SignUpFragment	11
SignUpViewModel	12
Home	12
HomeFragment	12
HomeViewModel	12
shop	12
UI	12
CartBottomSheetDialog	13
CartFragment	13
CartMap	13
ShopFragment	13
ShopItems	13
ShopListAdapter	13
ShopViewModel	13
LoginActivity	13
MainActivity	13
MyApplication	13
ProfileMenu	13
SharedPreferencesConfig	13
UiRefresher	14
Πως χρησιμοποιείτε:	14
Override μέθοδοι: refreshUi() και onDestroy()	14
Κλήση μεθόδων για την ανανέωση των UI	14
User_profile	14

## Μέρος Α: δομή της εργασίας

### 1. Ανοίγοντας η εφαρμογή

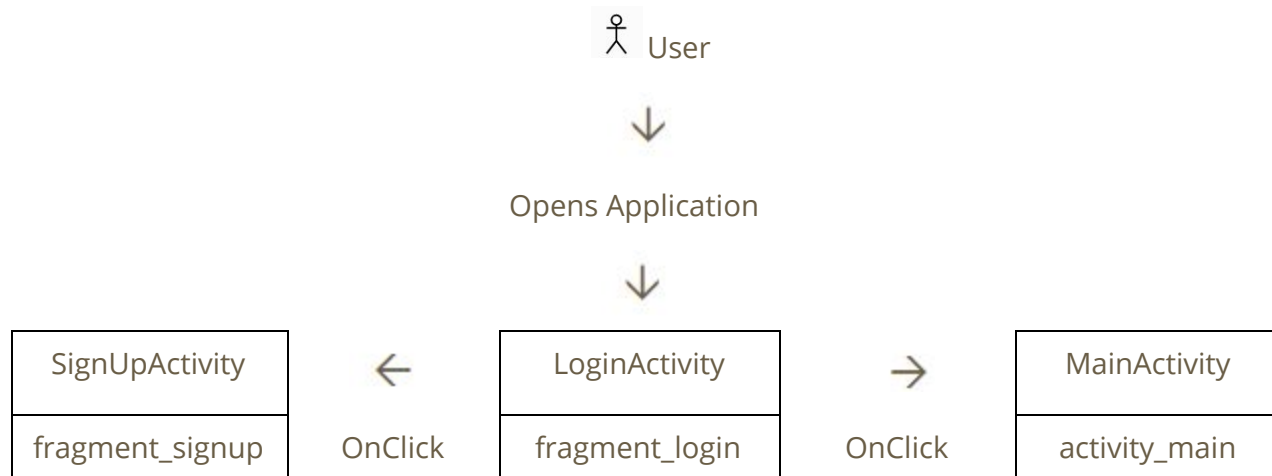
Χωρίς να έχει γίνει σύνδεση λογαριασμού

Ανοίγοντας το AndroidManifest.xml αρχείο, παρατηρούμε ότι η πρώτη (main) σελίδα που τρέχει η εφαρμογή, είναι η LoginActivity.

Από την LoginActivity μπορείτε να μεταφερθείτε στην SignUpActivity πατώντας το κουμπί Signur στο κάτω της σελίδας, και αντίστοιχα "Login" για την μεταφορά από την Signur στην Login.

Μόνο μετά τη σύνδεση μπορεί ο χρήστης να εισαχθεί στην κύρια εφαρμογή.

*Επεξήγηση πινάκων: η πρώτη σειρά είναι το Action, και η δεύτερη το Framgent του Action.*



### SharedPreferences

Αν ο χρήστης κάνει σύνδεση με λογαριασμό, και δεν κάνει αποσύνδεση μέσα από την εφαρμογή, τότε πρόκειται να μεταφέρεται με το άνοιγμα της εφαρμογής κατευθείαν στο MainActivity, χωρίς να ξαναπεράσει από το LoginActivity.

Πως λειτουργεί:

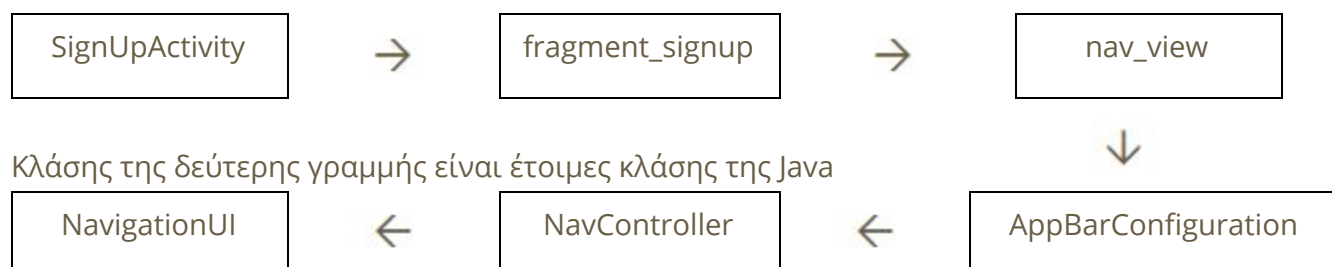
Υπάρχει η κλάση SharedPreferencesConfig, μέσω της οποίας η LoginActivity χρησιμοποιεί για να διαβάσει ή να αποθηκεύσει στο κινητό μία συγκεκριμένη μεταβλητή με την κατάσταση του χρήστη.

Αν η κατάσταση επιστρέψει ότι ο χρήστης έκλεισε την τελευταία φορά την εφαρμογή χωρίς να κάνει αποσύνδεση, τότε η LoginActivity δεν θα τρέχει τον κώδικα για να ανοίξει το

Fragment του, αλλά θα τρέξει την MainActivity.

## DrawerLayout

Χρησιμοποιήθηκε το έτοιμο DrawerLayout που προσφέρεται μέσα από το Android Studio, και προσαρμόστηκε αναλόγως με το πρόγραμμα.



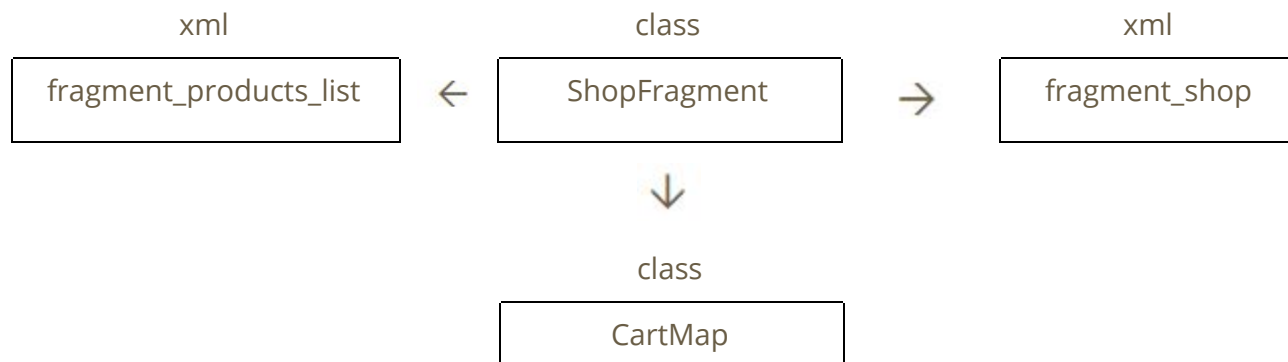
Για το AppBarConfiguration: Παίρνει παραμέτρους τα ξεχωριστά id για κάθε καρτέλα. Τα id βρίσκονται στα αρχεία: activity\_main\_drawer και mobile\_navigation

Στο Drawer περιέχονται 4εις καρτέλες, οι Home, Shop, Account και Administrator.

### Home



### Shop



### Account

Η AccountFragment περιέχει ένα ViewPager με δύο καρτέλες, και χρησιμοποιεί το fragment\_account για την δημιουργία του.

### Πρώτο φύλλο ViewPaper

Στο πρώτο φύλλο τοποθετείται το xml fragment\_user\_details με την κλάση UserDetails.

### Δεύτερο φύλλο ViewPaper

Στο δεύτερο φύλλο τοποθετείται το fragment\_user\_orders με την κλάση UserOrders. Η fragment\_user\_details χρησιμοποιεί επίσης τις κλάσεις UserOrderEditBottomSheetDialog με το xml του fragment\_order\_list\_sheet\_dialog, και DetailsMap.

### Administrator

Ανοίγει η κλάση AdministratorProfile, η οποία περιέχει περιέχει εξίσου ένα ViewPaper, το fragment\_administrator\_profile,, αλλά με 3α φύλλα.

#### Πρώτο φύλλο ViewPaper: User List

Χρησιμοποιείται η κλάση UserList, με το xml fragment\_users\_list.

#### Δεύτερο φύλλο ViewPaper: Products List

Χρησιμοποιείται η κλάση ProductsList, με το xml fragment\_products\_list. Για το άνοιγμα του sheet dialog για την προσθήκη ή την τροποποίηση προϊόντων, η ProductsList χρησιμοποιεί την κλάση ProductsListSheetDialog και το αντίστοιχο xml fragment\_oriducts\_list\_sheet\_dialog.

#### Τρίτο φύλλο ViewPaper: Orders List

Χρησιμοποιείται η κλάση OrderList, με το xml fragment\_orders\_list. Για το άνοιγμα του sheet dialog για τις πληροφορίες της παραγγελίας και την τροποποίηση της γίνεται το κάλεσμα της κλάσης ProductsListSheetDialog με το xml fragment\_products\_list\_sheet\_dialog.

### Cart

Το Cart είναι ένα Floating Action Button το οποίο λειτουργεί και μπορεί να το πατήσει ο χρήστης από τη στιγμή που θα εισέλθει στην κύρια εφαρμογή (δηλαδή κάνει login) μέχρι και να αποσυνδεθεί. Το design του βρίσκεται στο αρχείο app\_bar\_main.xml και με onClick συνάρτηση δημιουργεί αντικείμενο της κλάσης CartBottomSheetDialog και xml fragment\_cart.

Για να κρατήσει το πρόγραμμα το περιεχόμενο που έβαλε ο χρήστης στο καλάθι, αυτό δημιουργεί ένα Hashtable με την κλάση CartMap.

### UiRefreshher

Κλάση που περιέχει το Interface, το οποίο γίνεται implement στις κλάσεις που γίνονται αλλαγές κατά την εκτέλεση του ίδιου UI (όπως για παράδειγμα χρειάζεται ανανέωση των

λίστών όταν ο χρήστης αποθηκεύει ένα Edit ή ένα Delete της λίστας αυτής από το Sheet Dialog).

## Μέρος Β: κώδικας της εργασίας

### Java Επεξηγήσεις των φακέλων

Τα αρχεία της εφαρμογής έχουν χωριστεί και τοποθετηθεί το καθένα στον ίδιο φάκελο με τα αρχεία που τρέχουν για διεργασίες πάνω σε κοινό UI.

#### AndroidManifest.xml

Έγινε αλλαγή του ποιό θα είναι το main activity, ρόλος που δόθηκε στο LoginActivity. Αρχικό MainActivity τοποθετήθηκε και αυτό στο αρχείο, μαζί και με το SignUpActivity.

```
<activity
    android:name=".LoginActivity"
    android:label="Virop"
    android:theme="@style/AppTheme.NoActionBar">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

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

```
<activity
    android:name=".MainActivity"
    android:label="Virop"
    android:theme="@style/AppTheme.NoActionBar">
</activity>
<activity
    android:name=".ui.signup.SignUpActivity"
    android:label="Virop"
    android:theme="@style/AppTheme.NoActionBar">
</activity>
```

#### Φάκελος database

##### MyAppDatabase

Υλοποιεί την βάση του προγράμματος. Το αρχείο της βάσης είναι το simpleeshop.db, το οποίο ήταν και το αρχικό όνομα της εφαρμογής. Η αρχικοποίηση της γίνεται από το

Τα αρχεία OrderedItems, Orders, ProductImages, Products και User είναι οι κλάσεις που αντιστοιχούν στους πίνακες της βάσης. Η βάση γεμίζει μία φορά από το αρχείο simpleeshop.db, αν έχει γεμίσει μία φορά, ελέγχετε και δεν ξανά γεμίζει.

```
38 // Check if the database already exists.
39 // If it does not exist, create it from the simpleeshop.db file
40 // Comment the next line to reset database
41 if(!dbFile.exists())
42     builder.createFromAsset("simpleeshop.db");
```

Αρχείο MyAppDatabase.java

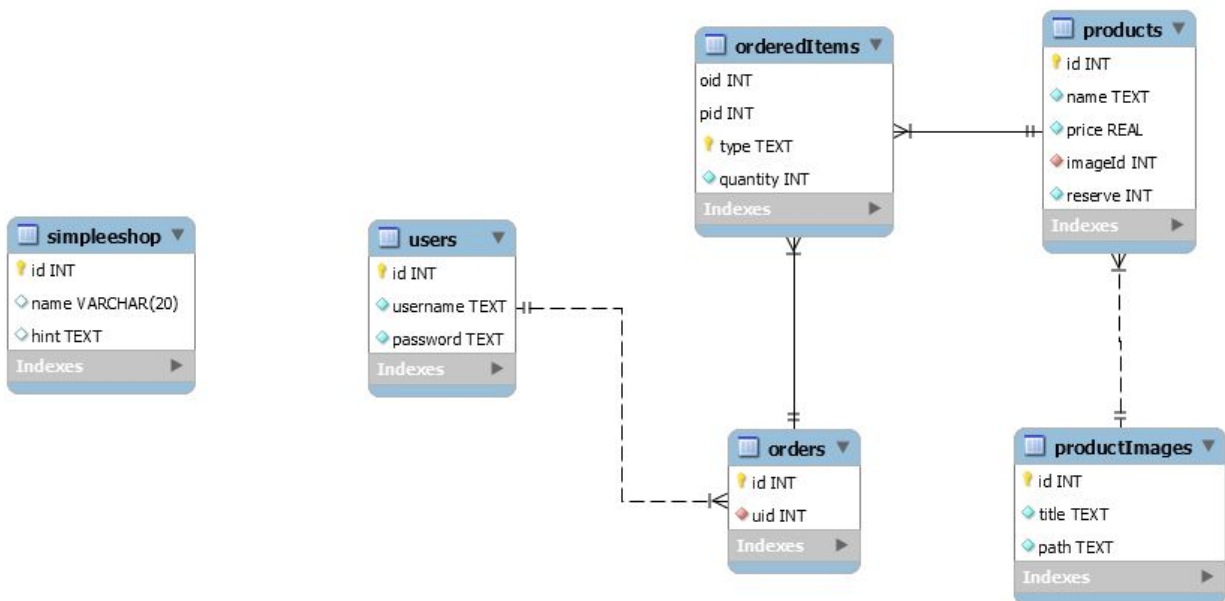
## MyDao

Περιέχει όλες τις κλήσεις της βάσης. Υπάρχουν σχόλια που οργανώνουν τον κώδικα.

Σε όποιο σημείο του προγράμματος υπάρχει διαγραφή κάποιας βάσης που επηρεάζει κάποιον άλλο πίνακα, διαχειρίζεται ανάλογα:

- Υπάρχει onDelete και onUpdate CASCADE
- Όταν αγοράζετε ένα προϊόν, η τιμή προμήθειας του reserve μειώνεται
- Όταν διαγράφεται μία παραγγελία, αυτή αποκαθιστά το reserve του προϊόντος ανάλογα.

Στο παρακάτω διάγραμμα φαίνεται το Database schema της βάσης:



## Φάκελος UI

Όλα τα αρχεία στους υποφακέλους account, administrator, home και shop λειτουργούν λίγο πολύ με τον ίδιο τρόπο· όπως και για τον signup και το login φάκελο.

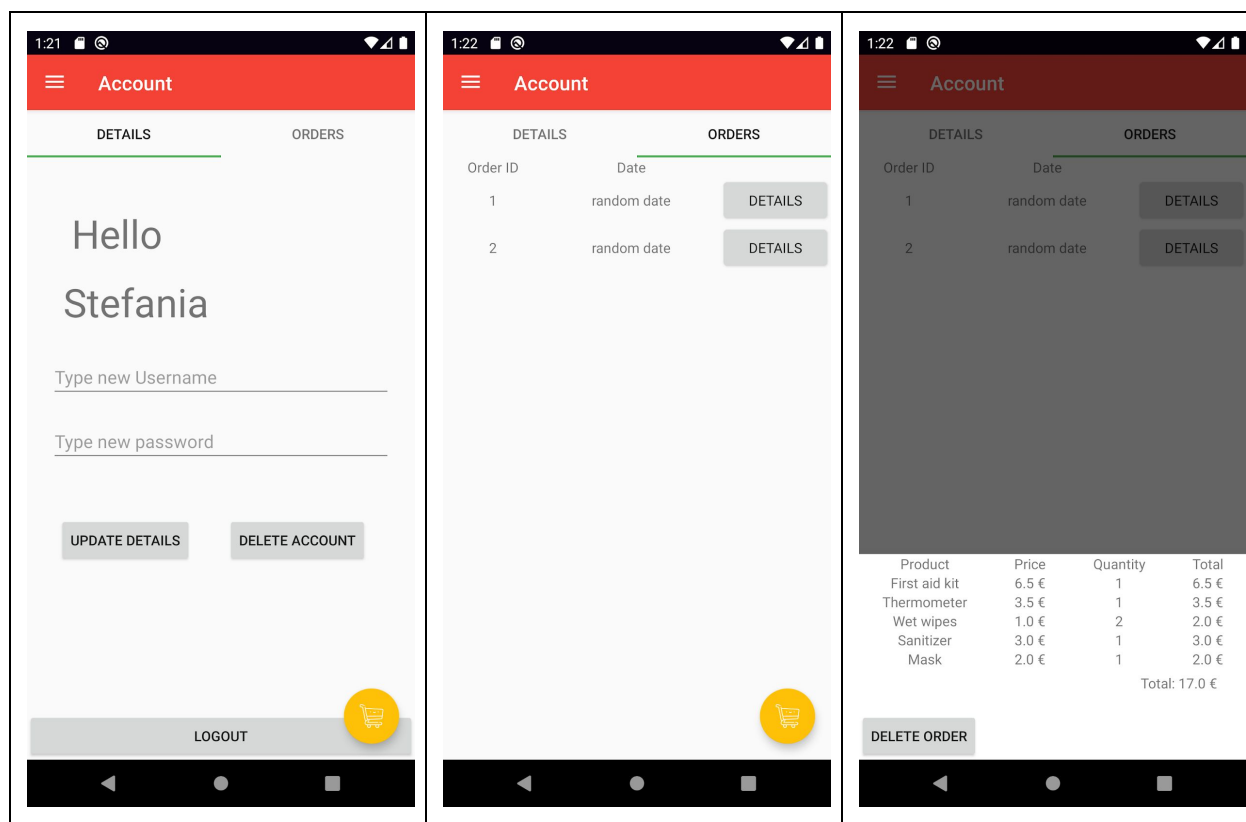
## account

### UI

#### Σχόλια

- Στο UI του χρήστη admin: δεν του εμφανίζονται τα πεδία για την αλλαγή Username και διαγραφή λογαριασμού.
- Το όνομα μετά το Hello είναι το Username του χρήστη
- Ο χρήστης μπορεί να αλλάξει είτε μόνο το Username, είτε μόνο το Password, είτε και τα δύο





### AccountFragment

Η κλάση που διαχειρίζεται και τρέχει τις υπόλοιπες κλάσεις στον φάκελο.

### AccountViewModel

Δεν έκανα αλλαγές.

### DetailsMap

Δημιουργεί και διαχειρίζεται το Hashtable για την αποθήκευση των προϊόντων του Cart.

### UserDetails

Η σελίδα που ανοίγει όταν ο χρήστης βρίσκεται στο πρώτο ViewPager. Παίρνει απλά το (αποθηκευμένο με SharedPreferences) id του χρήστη, το εμφανίζει και το χρησιμοποιεί για τις όποιες αλλαγές στον λογαριασμό θέλει να κάνει από τη σελίδα.

### UserOrderEditBottomSheetDialog

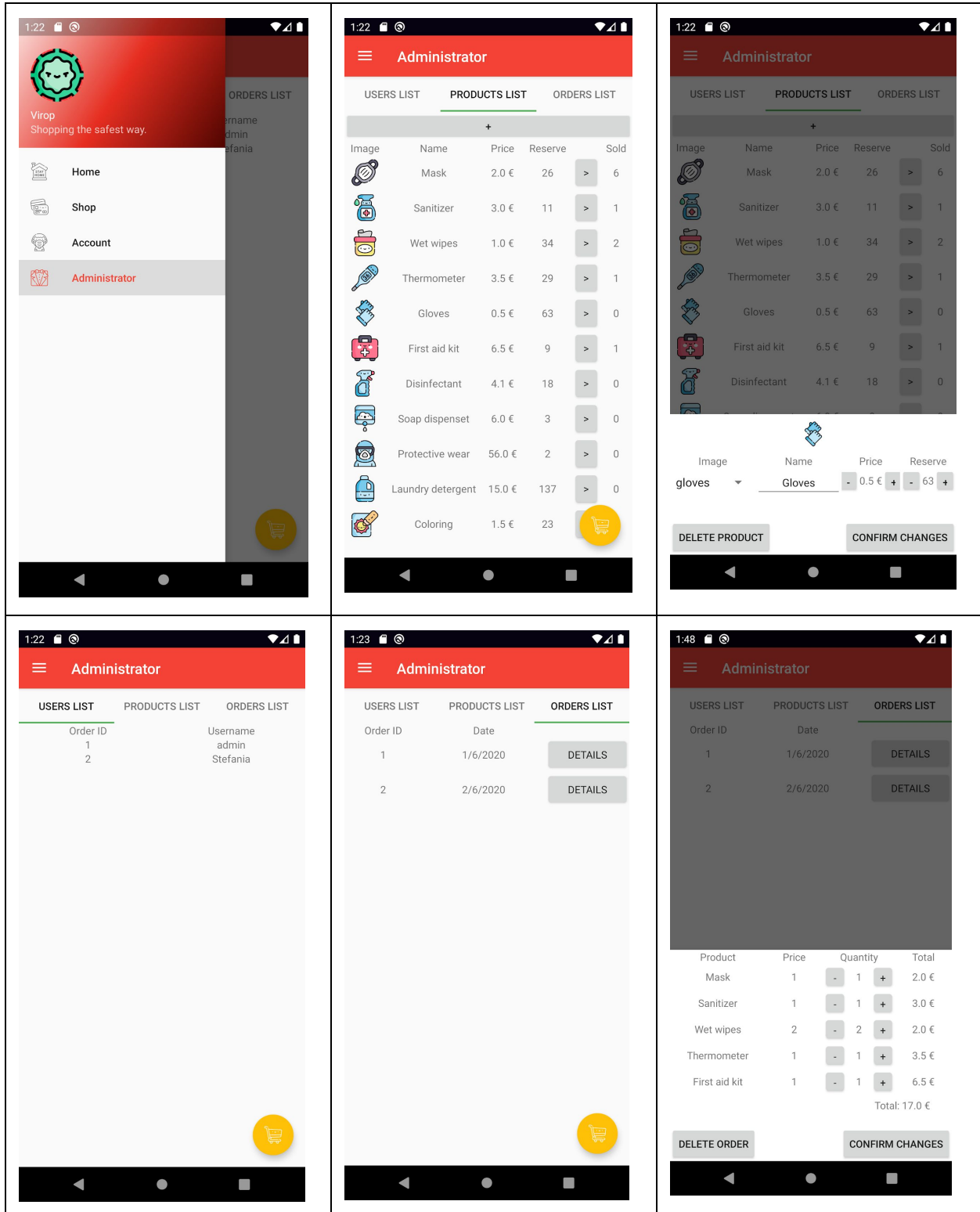
- Εμφανίζει ένα Bottom Sheet Dialog BottomSheetDialogFragment με τον πίνακα από το fragment\_user\_order\_dialog\_sheet,
- γεμίζει τον πίνακα με γραμμές δυναμικά παίρνοντας τιμές από την βάση

### UserOrders

Εμφανίζει τον πίνακα με τα Orders του συνδεδεμένου χρήστη.

# administrator

## UI



AdministratorProfile

Η κλάση που διαχειρίζεται και τρέχει το ViewPager, θέτοντας τα υπόλοιπα αρχεία στον φάκελο σε ποιά καρτέλα θα εμφανιστούν.

OrderListSheetDialog


Εμφανίζει τις πληροφορίες την παραγγελίας, και διαχειρίζεται τις ενέργειες των κουμπιών DELETE ORDER, CONFIRM ORDER και τα "+" και "-". (6ο screenshot στον πίνακα)

OrdersList

Εμφάνιση των παραγγελιών όλων των προϊόντων στην τρίτη καρτέλα του ViewPager. (5ο screenshot στον πίνακα)

ProductList

Εμφάνιση όλων των προϊόντων στην δεύτερη καρτέλα του ViewPager, μαζί με όλες τις πληροφορίες τους και τις συνολικές πωλήσεις του προϊόν στην τελευταία στήλη "sold". (2ο screenshot στον πίνακα, και screenshot ακριβώς από κάτω)

Image	Name	Price	Reserve		Sold
	Mask	2.0 €	26	>	6

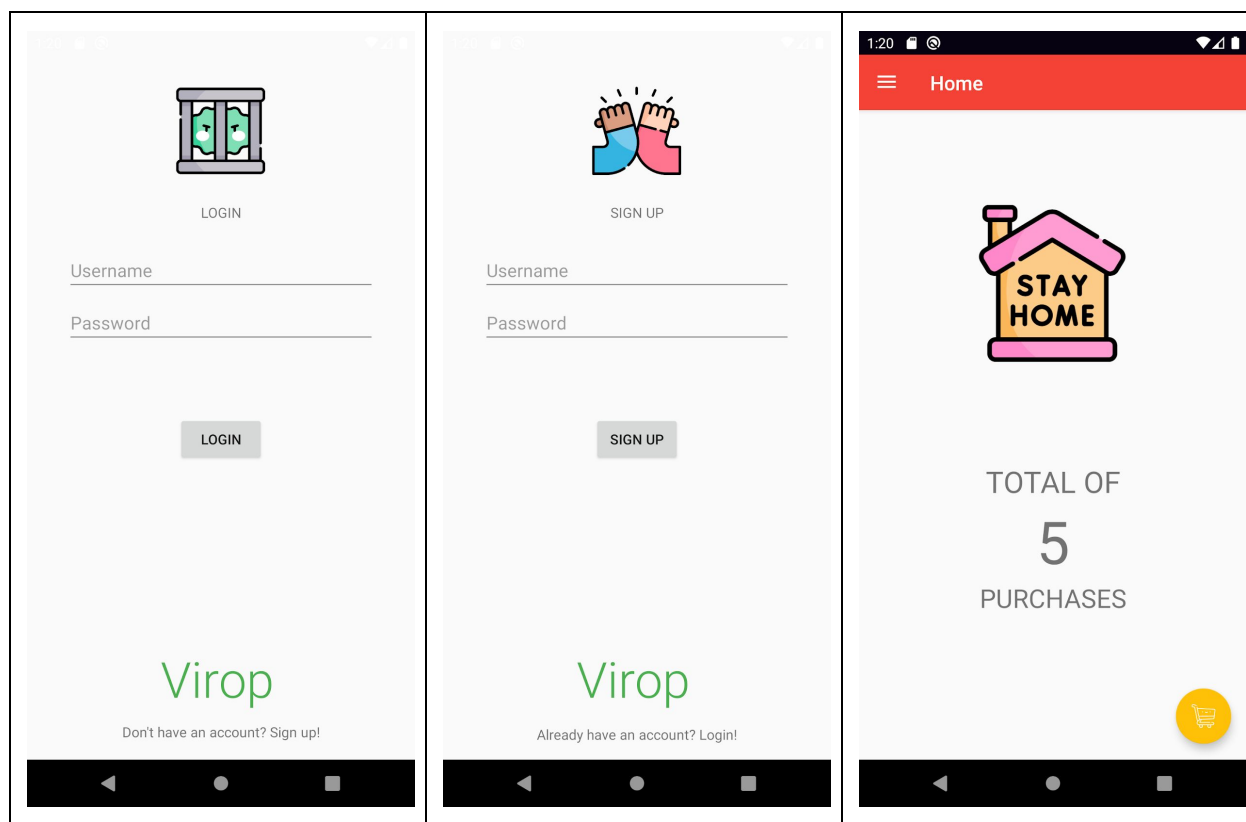
ProductsListSheetDialog

Εμφάνιση των πληροφοριών των προϊόντων, και διαχείριση των κουμπιών στο αντίστοιχο interface, μαζί με αυτά και τον κώδικα για τον Spinner. (3ο screenshot στον πίνακα)

UserList

Εμφάνιση των παραγγελιών όλων των χρηστών στην τρίτη καρτέλα του ViewPager. (4ο screenshot)

UIs for Home, Login and Sign up



## login

Δημιουργήθηκαν τελικά δικές μου κλάσεις και δεν αξιοποιήθηκαν αυτές στον φάκελο. (Η κλάση που κάνει το login είναι η LoginActivity που δεν βρίσκεται σε υποφάκελο).

## signup

### SignUpActivity

Η κλάση που διαχειρίζεται και τρέχει τις υπόλοιπες κλάσεις στον φάκελο.

### SignUpFragment

Απλά θέτει και επιστρέφει στο SignUpAction το fragment που είναι να χρησιμοποιηθεί.

### SignUpViewModel

Auto Generated από το Android Studio. Περιέχει μετατροπές των UI elements, αλλά μόνο τις auto generated μετατροπές, δεν έκανα αλλαγές σε αυτό το αρχείο.

## Home

### HomeFragment

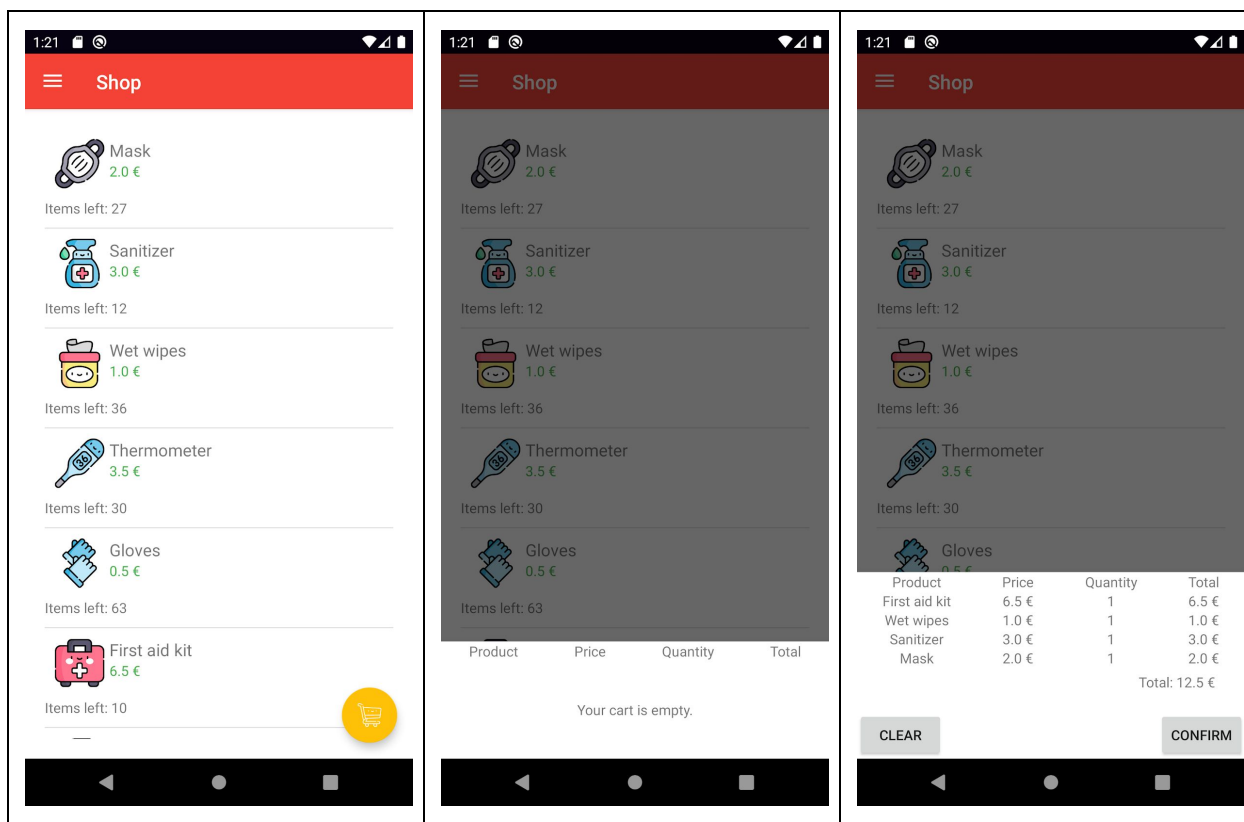
Η κλάση που διαχειρίζεται και τρέχει τις υπόλοιπες κλάσεις στον φάκελο.

## HomeViewModel

Auto Generated από το Android Studio. Περιέχει μετατροπές των UI elements, αλλά μόνο τις auto generated μετατροπές, δεν έκανα αλλαγές σε αυτό το αρχείο.

## shop

### UI



## CartBottomSheetDialog

Ανοίγει το Sheet Dialog για το καλάθι, και γεμίζει τον πίνακα με βάση το HashTable από το CartMap. Τρέχει με το πάτημα του κίτρινου κουμπιού στην κάτω δεξιά γωνία του UI.

## CartFragment

Δεν την χρησιμοποίησα.

## CartMap

Διαχειρίζεται το HashTable για το καλάθι αγορών.

### ShopFragment

Τρέχει όταν ανοίγει το Shop, και καλεί την ShopListAdapter για να εμφανίσει τα προϊόντα.

### ShopItems

Δεν την χρησιμοποιησα.

### ShopListAdapter

Εμφανίζει στη λίστα από το αρχείο fragment\_shop αντικείμενα με μορφοποίηση από το fragment\_shop\_item.

### ShopViewModel

Auto Generated από το Android Studio. Περιέχει μετατροπές των UI elements, αλλά μόνο τις auto generated μετατροπές, δεν έκανα αλλαγές σε αυτό το αρχείο.

## LoginActivity

Εισάγει τον χρήστη στην εφαρμογή και αποθηκεύει την κατάσταση και το user id με SharedPreferences.

## MainActivity

Ανοίγει το DrawerLayout, και θέτει το visibility του έξτρα μενού για τον admin σε visible ή invisible (setVisible(True) και setVisible(false)) ανάλογα να το id του συνδεδεμένου χρήστη είναι το 1, που ανήκει στον admin.

## MyApplication

Διαχειρίζεται την SharedPreferencesConfig.

## ProfileMenu

Δεν την χρησιμοποιησα.

## SharedPreferencesConfig

Περιέχει τις μεθόδους για να διαχειριστεί το status (logged in ή logged out, που είναι true και false ανάλογα) και του logged in user's user id.

## UiRefresher

Υπάρχει το interface refreshListener, το οποίο το κάνουν implement οι κλάσεις που χρειάζεται να ανανεώσουν το UI τους με καινούρια δεδομένα.

Πως χρησιμοποιείτε:

Override μέθοδοι: refreshUi() και onDestroy()

Οι κλάσεις που κάνουν το implement, κάνουν Override από το Interface refreshListener τις refreshUi() και onDestroy(). Η refreshUi() ξανα δημιουργεί το interface, ενώ η onDestroy() διαγράφει τον προηγούμενα φτιαγμένο refreshListener για την αποφυγή του memory leak και exceptions.

```

33 public class OrdersList extends Fragment implements UiRefresher.RefreshListener {

56     @Override
57     public void onDestroy() {
58         super.onDestroy();
59         UiRefresher.Instance().removeListener(this);
60     }

61
62     private void initializeOrdersTable(){
63         MyAppDatabase db = MyAppDatabase.Instance();
64
65         List<Orders> ordersList = db.myDao().getOrders();
66         for(Orders order : ordersList) {
67             addRow(order.getId(), date: order.getId() + "/6/2020");
68         }
69     }

```

*Αρχείο OrderList.java*

Κλήση μεθόδων για την ανανέωση των UI

```

159 // Refresh
160 UiRefresher.Instance().refreshUis();

```

*Αρχείο OrderListDialog.java*

User\_profile και shopItem

Δεν την χρησιμοποιησα.

Τέλος Εργασίας

Viirop