Informatics Large Practical

Patricia Milou s1616316

March 26, 2019

1 Description of the Algorithms & Data Structures Used

1.1 Downloading the Geo-JSON map

The process of downloading the Geo-JSON map was implemented as an asynchronous task, as this made the game faster. An asynchronous task allows the application to perform background operations and show the results on the UI thread without having to manipulate threads.

When an asynchronous task is executed, the task goes through 3 steps:

- onPreExecute: Step used to set up the task
- doInBackground: Step used to perform the actual task
- onPostExecute: once the doInBackground function finishes executing the task, this step delivers the result back to the main UI thread and stops the AsyncTask process.

For my implementation, I created a class called DownloadFileTask which extended the AsyncTask Interface. Please note that the PreExecute task was unnecessary since that step was developed in the Main Activity, by constructing the url of today's map. After that, the AsyncTask is called.

In the doInBackground function, I transform the url of type String into an InputStream. An InputStreamReader then reads that InputStream and by using a BufferedReader we retrieve the Geo-JSON map in type String. Then, the onPostExecute functions sends the text of the map , back in the Main Acitivty.

1.2 Parsing the Geo-JSON map

To parse the Geo-JSON map , I first turned the text of the map into a JSONObject object.By using a JSONObject, I can access all fields and store the values as I want.

For example, to get the exhange rate of the Shil currency, I used the following:

```
JSONObject json = new JSONObject(mapstr); \\where mapstr is the text of the Geo-JSON
JSONObject rates=json.getJSONObject("rates");
shil = rates.getString("SHIL");
```

From the Geo-JSON map, I extracted the exchange rates and all 50 coins. For each coin, I extracted, it's:

- ID
- Value
- Currency
- Marker-Symbol
- Marker-Color
- Coordinates

1.3 MapBox Map

The MapBox [2] map is located in the Main Activity. This is where most of the game will be played.Once the application starts, the map focuses on the location of the user. The top right button can be used to alternate the focus of the map between the user's location and the central campus.

All coins are placed as markers on the map.Note that the marker's icon depends on the Geo-JSON map, which includes the Marker-symbol and Marker-colour of each coin.

Ten different icons are loaded in the application, were their label is from 0-9. We choose which one to use, according to the Marker-Symbol. Then, we change the background of the marker to the appropriate colour, according to the Marker-Colour property.

1.4 Collecting a coin

The user can collect a coin when he is at most 25 metres away from it.To do this:

1. I keep track when the user's location changes. Whenever his location changes:
   (a) For all coins on the map, I calculate the distance between them and the user's location. For this I used the function "DistanceBetween" which is part of the public class Location.
   This function takes four parameters that indicate point's A latitude, point's A longitude, point's B latitude and point's B longitude and returns the distance between them in metres.
   (b) I find the coin closest to the user.
   (c) I then check if this distance is less than or equal to 25.
   (d) If yes, a HashMap object is created with the following fields and values:
   { Id, Currency, Value, Collection Date, Expiry Date }. This HashMap object is then stored in Fireabase under the Wallet Collection of that user.
   (e) The coin's marker is then removed from both the map and the marker's ArrayList which I am using. In this way, the user is not able to collect the same coin again.

1.5 Sending Spare Change

In the Wallet activity, if the user clicks on a coin, he has the option to send it to a friend. This is done with the following steps:

1. A Dialog first inflates where the user's friends are listed by their email address.
2. After the user selects one friend, the HashMap object stored in the wallet of the user is copied to the wallet of his friend with an extra field added called "Gift" which is set to true. Then the HashMap object which is located in the user's wallet is deleted.
3. If the coin was sent, a success toast message will appear.If not the user will be asked to try again through a different toast message.

1.6 Depositing a Coin

In the Wallet activity, if the user clicks on a coin, he has another option to deposit it into his bank account. This is done with the following steps:

1. I first extract the currency and the value of that coin, by accesing that HashMap object stored in Firebase.
2. I then create another HashMap object which represents the new coin with the following keys and fields:
   { Currency: Gold, Value: Previous Value * Exchange Rate , Deposited Date : Current Date }.
3. This new object is then stored under the Bank collection of that user.
4. Finally, the coin object stored in the user's wallet is deleted.

In the Bank Activity, I used the MPAndroidChart[4] library, to produce a chart showing the deposits made on a daily basis.

1.7 Removing Expired Coins

One of the bonus features that I have added is that coins expire after 2 days. Note that, the collection and expired date are stored in Firebase. Everytime, that the wallet activity is created, a function called removeExpiredCoins is called. In this function:

1. I retrieve all coins from the Firebase.
2. For each coin retrieved, I check whether the Expired Date is the same as the current date.
3. If yes, that coin is deleted from the Wallet Collection.
4. If not, the coin is added on the table located in the Wallet Activity.
5. A counter keeps track of how many coins were deleted.After all expired coins are removed, a toast message informs the user of that change.

1.8 Step Detector

As one of my bonus features, I decided to implement a pedometer which counts the steps of the user as he is playing the game. I found online a few tutorials on how to do it and then decided to go with the way that GadgetSaint [1] proposed on their website, so this is how I have developed it.

1. StepListener Interface: Listens to alerts about steps being detected.
2. SensorFilter class: Filters out the values that has a close approximation to steps.
3. StepDetector class: Accept updates from accelerometer sensor and deploys the filter to detect if a step has been covered by the user.

My Main Activity keeps track of the steps of the user and stores them in Firebase too. The application set goals to the user, and once the goal is reached a congratulation message appears and the goal inreases.

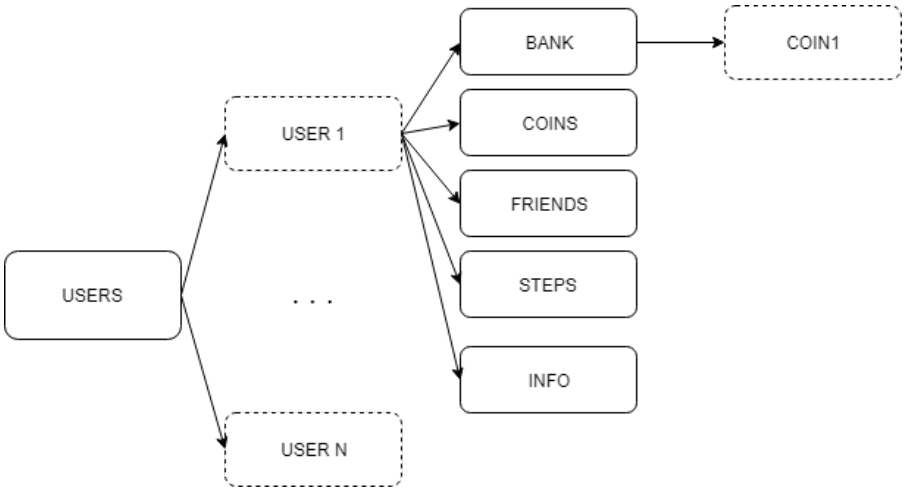
1.9 Navigation

Another bonus feature that I have included is navigation [3], which is part of the map service. Through the top left button, the user can navigate to the the coin which is closest to his location.

My algorithm is:

1. A dialog appears which asks the user to select a currency. The options are: Peny,Dolr,Shil,Quid,No preference
2. Find the closest coin,that satisfies the currency selection of the user.This is done in a very similar way such that collecting a coin works.
3. Ask the user if he would like to navigate to that point.
4. If yes, get the quickest route between the user's location and the location of the coin.
5. Show the directions to the user.

2 Database Service



Database Structure

Firebase [5] is a platform which allows you to build web and mobile applications without server side programming language.In Firebase, I almost store everything.

In the diagram above, the dashed ovals represent documents whereas the non-dashed ovals represent collections. In the User's collection , I store all users where their id is their email address. Each user has at most 5 collections (Bank,Coins,Friends,Steps,Wallet). In the Bank collection, I store all coins that have been deposited into the user's account. Each coin has a unique id and some more fields. Similarly, the coins store in the Coins collection are the ones collected by the user but not yet deposited. In the Friends collection I store the friends of the user and in the Info collection I store some information regarding the user for example their date of birth.

3 Parts of design which have not been realized in the implementation

3.1 Friends Location on Map

At design, I mentioned that the user's friends location will be marked on the map. This location could be real-time(if his/her friends were playing the game at the same time as the user), or it could be their previous location. During implementation, I realized that this bonus feature couldn't be developed since Mapbox doesn't offer something similar to a movable marker. The only way that I could develop that was to : (for 1 friend)

1. Add a markers on his/her friend location
2. Every time, that friend moves:
   (a) Remove their previous marker
   (b) Add another marker on his/her current location

This method is not the most efficient, since delay would have been created bye removing and adding markers. In addition to that, the user wouldn't be able to see a nice flow of the marker's movement. For these reasons, I decided not to proceed with the implementation of that bonus feature.

4 Additional features of your implementation which were not described in your design

For my application, I followed my design and project plan which I submitted for coursework 1 , therefore I haven't developed any additional features that haven't been described there.

5 Android Studio Project

- API level: 28
- Total Number of Xml Files developed: 16
- Total Number of Classes developed: 16
- Total Number of Espresso Tests developed: 8
    1. Closest Coin and Navigation(Executes a bonus feature test)
    2. Depositing a coin test(Note that the user must have at least 1 coin in order for the test to pass successfully)
    3. Friends Test (A test for the Friends Activity)
    4. Login Test (A test for the Login Activity)
    5. Profile Test (A test for the Profile and Edit Profile activities)
    6. Register User Test(A test for the Register Activity(uses a random email every time it runs)
    7. Settings Test (A test for the user preferences)
    8. Wallet test(A test for the Wallet activity, user needs to have at least one coin in order for the test to pass successfully).
    9. A Class to read Toast messages [6])

6 App in use

6.1 Login Page

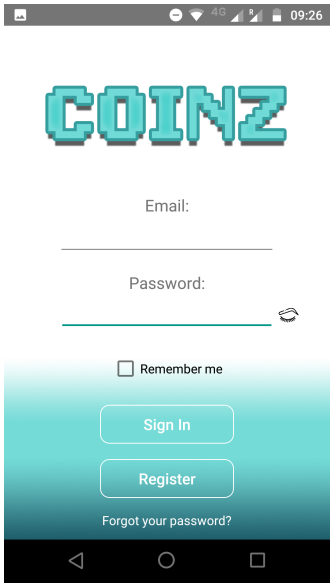


Figure 1: Log in Page

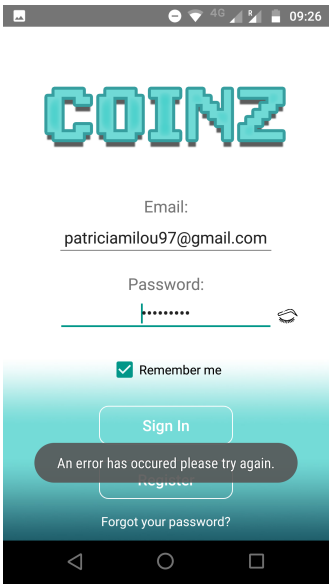


Figure 2: Exception in login

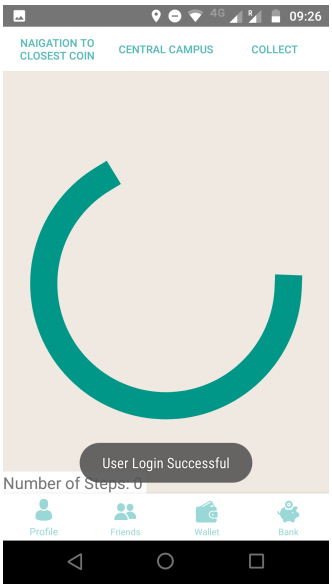


Figure 3: A successful login

In figure 1, the user must enter his/her credentials in order to be authenticated. If he/she enter wrong ones then they will see a toast message as shown in figure 2. In the case, they they are authenticated they will be notified with a "User Login Successful" toast message as shown in figure 3. They will also be redirected in Main activity.

If he/she had forgot their password, by clicking on the "Forgot your password?" a dialog will pop up. There the user must enter his email address.After that, then they will receive a password-reset email. The login page also includes a "Remember me" function and an icon which can make the password visible and then hidden as well. In the case that the user hasn't created an account yet, he can simply click on the "Register" button.

6.2 Registration and Profile

In the Register activity, figure 4, the user must enter an email address and a password. These will be the credentials that he/she will be asked to input in a later game to login. After a successful registration the user will be redirected in the Edit Profile activity, figure 5, where he will be asked to enter some information. With the back button, the user can skip this step. He can edit this information from his Profile activity later. By clicking on the "Update your profile" button he can access all his photos stored in his smart phone and choose his new profile picture. By clicking on the "Save" button the information is saved and he/she is redirected to the Show Profile activity, figure 6.Please note that for the user's profile I am using a circle image view [7].The reason behind it's because I wanted to create an application that looks modern.

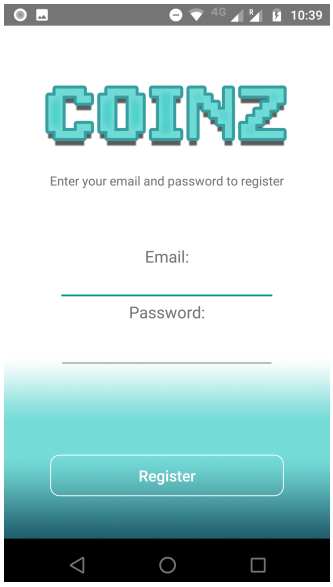


Figure 4: Register Activity

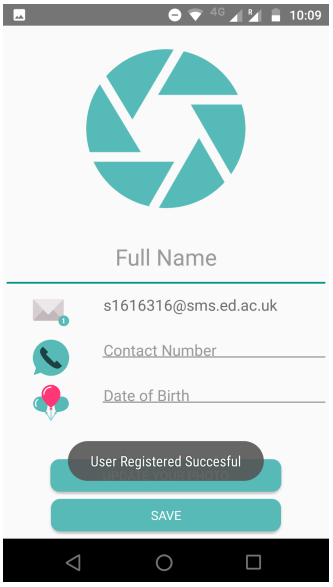


Figure 5: Edit Profile Activity

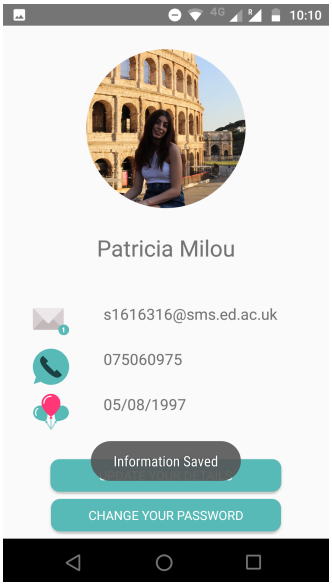


Figure 6: Show Profile Activity

6.3 Map View and Collection

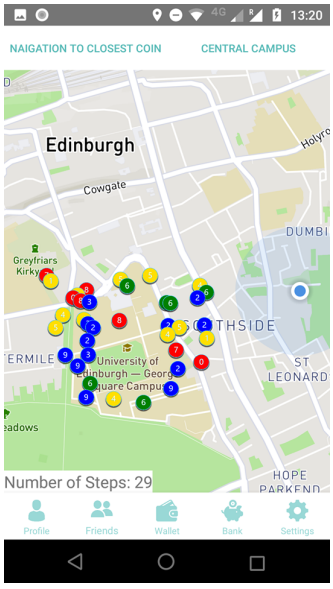


Figure 7: Main Activty

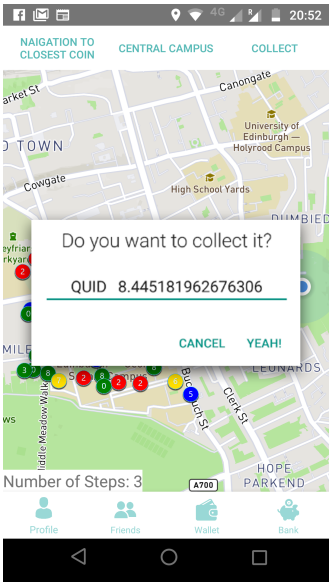


Figure 8: Close to the coin

In Main Activity, figure 7, the user can see the coins and his/her location on the map.In Main Activity, I included a bottom navigation bar [9],from which the user can move to different activities. If the user's location is within 25 metres from a coin then a dialog will pop up as shown in figure 8. The dialog will ask the user if he/she wants to collect it. By pressing on the "YEAH!" button, the coin will be saved in his/her wallet. If the user clicks on the "CANCEL"button, the dialog will close and the coin won't be collected.

6.4 Navigation

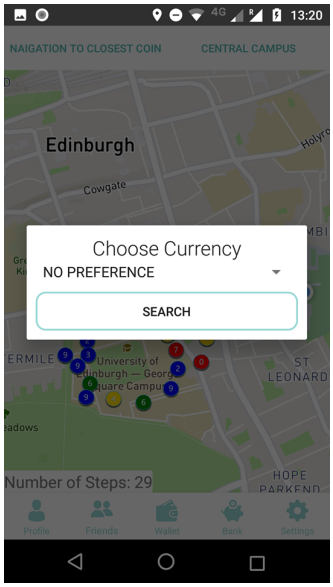


Figure 9: Choose Currency

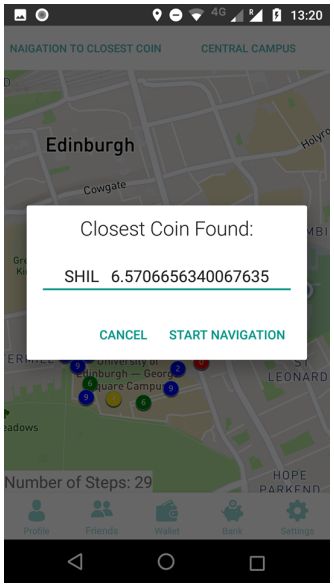


Figure 10: Navigation Dialog

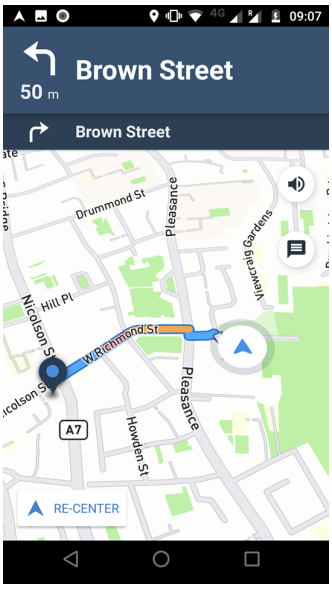


Figure 11: Navigation

By clicking on the top left button which is located in the Main activity a dialog comes up, like in figure 9.The user must then select the most preferable currency.By clicking on the "Search" button,another dialog appears,where the user can see which coin is closest to his location.This is shown in figure 10. By clicking on the "Start Navigation" button, the user can start a navigation towards that coin. This is shown in figure 11.

6.5 Friends

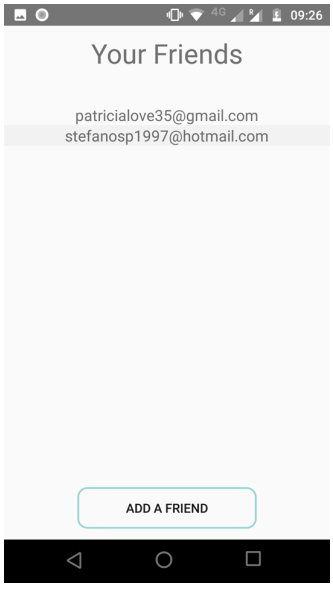


Figure 12: Friends Activity

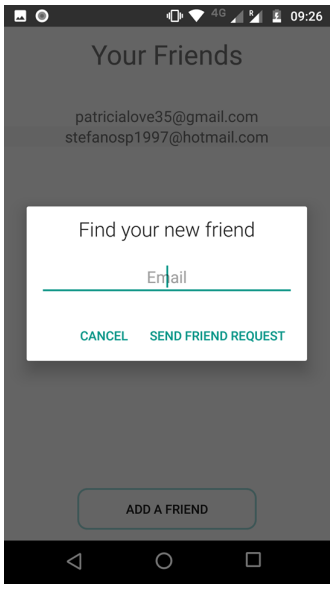


Figure 13: Add a new Friend

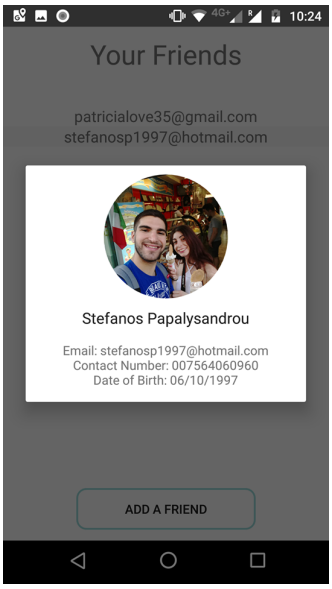


Figure 14: Friend's Profile

By clicking on the "Friends" tab in Main Activity, the user can access the Friends Activity, figure 12. In this activity the user can add a new friend by clicking on the "Add a Friend" button.This is shown in figure 13. The user can only add another user who is not currently in his friend list,but is already registered to Coinz. By clicking on a friend's email address, the user can see his friend's full profile, figure 14.

6.6 Wallet

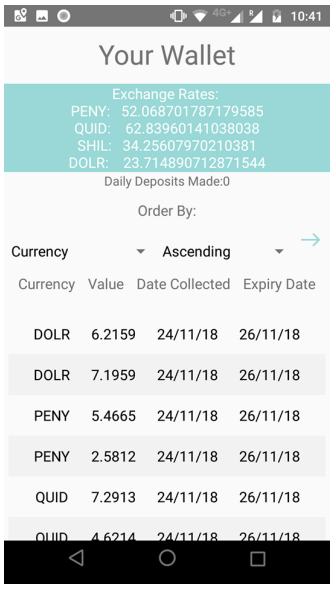


Figure 15: Wallet Activity

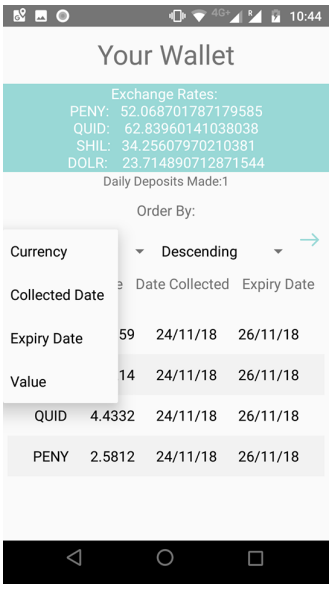


Figure 16: Changing the ordering

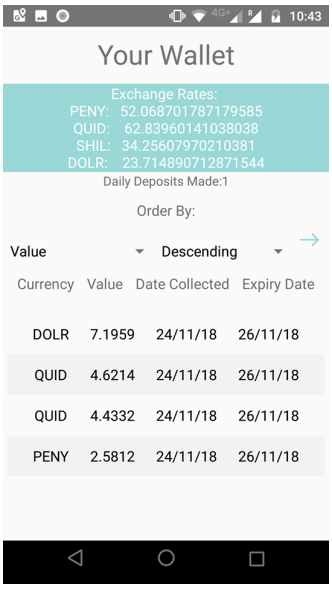


Figure 17: Another ordering

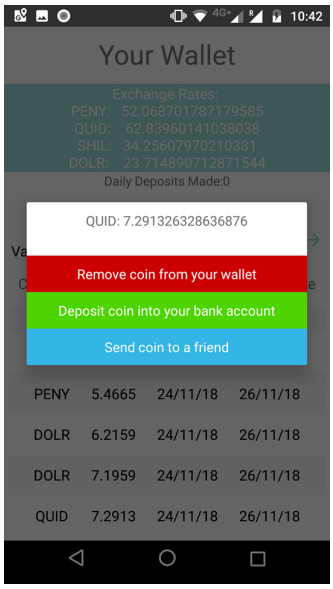


Figure 18: Coin Choices

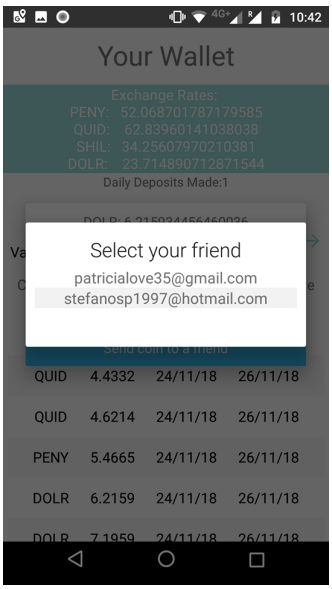


Figure 19: Sending Spare Change

By clicking on the "Wallet" tab in Main Activity, the user can access the Wallet Activity, figure 15.In this activity, the user can see the current exchange rates and the coins he has collected so far, and yet have not been deposited.The user can change the ordering of the coins table in any way he/she likes as shown in figures 16 and 17.

By clicking on a specific coin in the Wallet Activity, a dialog will pop up, as shown in figure 18.Through this dialog, the user can remove a coin from his wallet, deposit it into his bank account or send it to one of his/her friends as spare change. If the user selects to send it to a friend, then another dialog will pop up as shown in figure 19. The user must then select the desired friend. A toast message is used to confirm the user about the result of any action he carries out.

6.7 Bank

https://www.overleaf.com/project/5be9eae217ddf034f776a57a

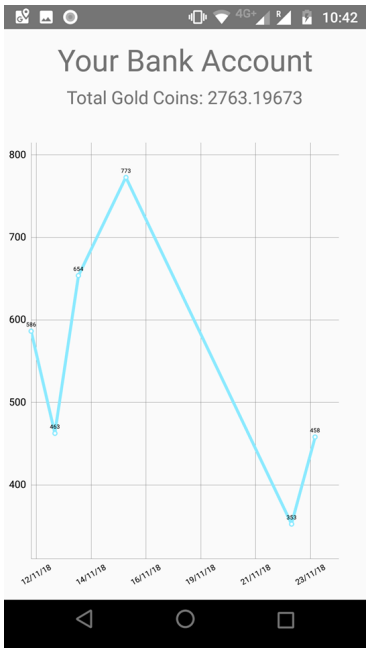


Figure 20: Bank Activity

By clicking on the Bank tab in Main activity, the Bank activity is launched. There, the user can see the total amount of deposits he has made. In addition to that, a graph [4] as shown in figure 20 , shows the amount of gold coins deposited on a per day basis.

6.8 Settings

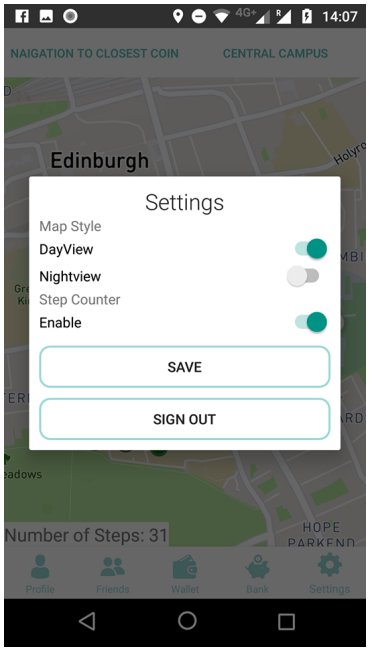


Figure 21: Settings Dialog

By clicking on the Settings tab located in the main activity, the user can access the setings dialog. Through that, he can customize the application in his own way.He can change the style of the map and enable/disable the step counter bonus feature. He can save the changes he/she has made by clicking on the "Save" button. By clicking on the "Sign out" button, the user is sign out and the Login activity restarts. The settings dialog is shown in figure 21.

References

[1] GadgetSaint, *Create a Simple Pedometer and Step Counter in Android*, http://www.gadgetsaint.com/android/create-pedometer-step-counter-android/.W-ykXOj7SUl

[2] MapBox, *Documentation*, https://www.mapbox.com/documentation/

[3] MapBox Navigation, *Documentation*, https://www.mapbox.com/android-docs/navigation/overview/

[4] MPAndroidChart, *Documentation*, https://github.com/PhilJay/MPAndroidChart/wiki

[5] Firebase, *Documentation*, https://firebase.google.com/docs/

[6] Toast Matcher, *How to check Toast window, on android test-kit Espresso*, http://baroqueworksdev.blogspot.com/2015/03/how-to-check-toast-window-on-android.html

[7] Circle Image View, *Used for the profiles of the users*, https://github.com/hdodenhof/CircleImageView

[8] Android ImageView that handles animated GIF images , *Used in the Splash Screen*, https://github.com/felipecsl/GifImageView

[9] BottomNavigationBar, *Used in the Main Activity*, https://github.com/aurelhubert/ahbottomnavigation