Custom project final report

COS30017 Software Development for Mobile Devices 2023

Name: Trung Kien Nguyen

Student ID: 104053642

Table of Contents

[Overview of project 1](#_Toc149809827)

[Level 1: Design evidence 1](#_Toc149809828)

[Level 2: App evidence 5](#_Toc149809829)

[Level 3: Extended research evidence 5](#_Toc149809830)

[References 5](#_Toc149809831)

[Appendix 1 - Weekly reports 5](#_Toc149809832)

[Week 7 5](#_Toc149809833)

[Week 8 7](#_Toc149809834)

[Week 9 9](#_Toc149809835)

[Week 10 11](#_Toc149809836)

[Week 11 16](#_Toc149809837)

# Overview of project

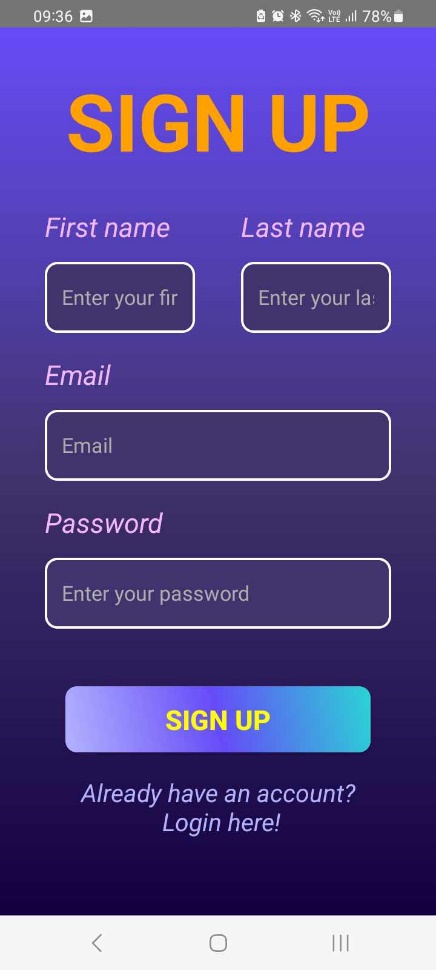
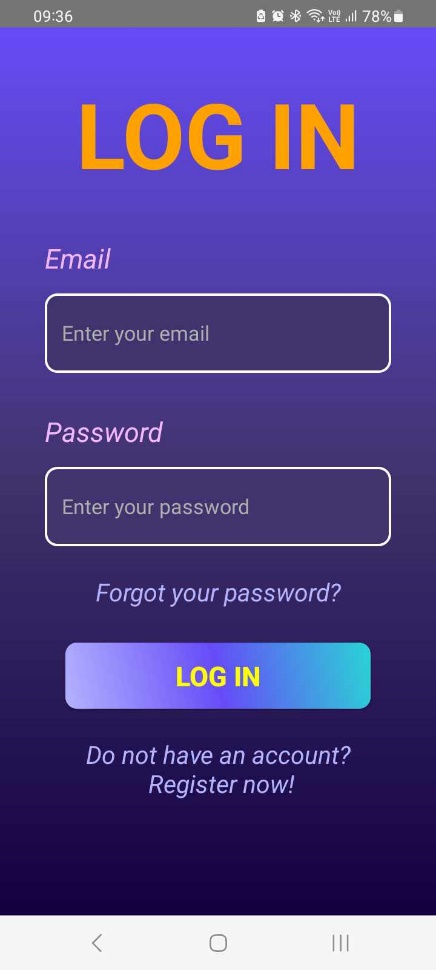
My project is a cryptocurrency app using Android Development with Kotlin. Like many typical apps in the market, such as CoinGecko, Coinbase, Kraken, etc., my app has some basic functionalities as follows:

* User authentication: The app allows users to create accounts, log in with them, or reset their password in case of forgetting.
* Wallet managing: It also creates and manages cryptocurrency wallets for different cryptocurrencies, e.g. Bitcoin, Etherium, ….
* Viewing transaction history
* Buy and sell cryptocurrencies as will.

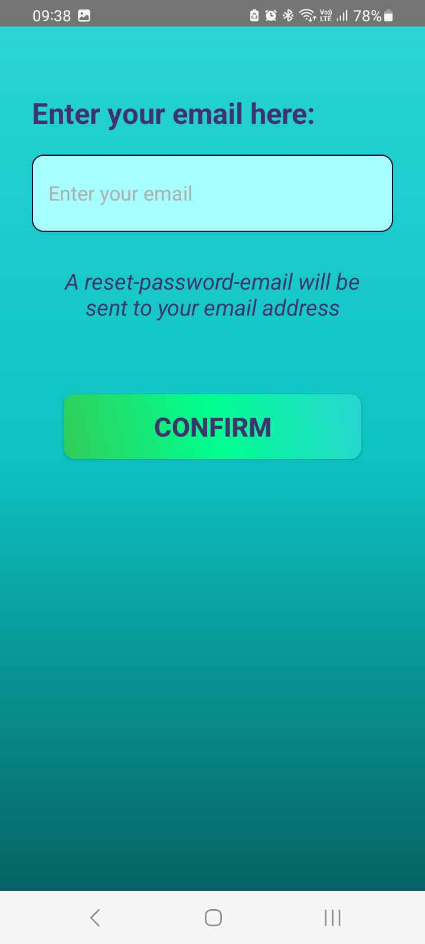
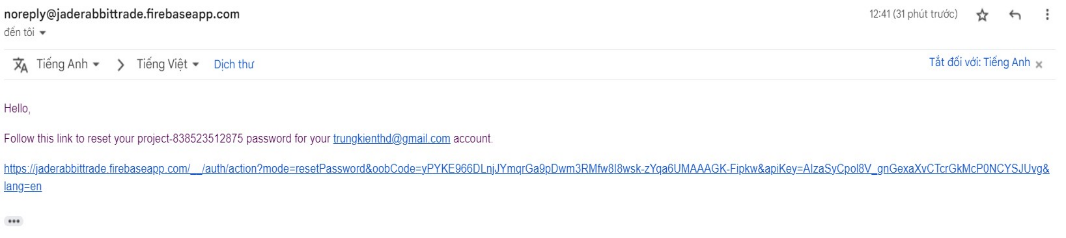
# Level 1: Design evidence

This following section includes some sketches and basic functionality of the main activities for my app:

* Login Activity and Signup Activity:

These activities are responsible for signing in and signing up purposes, using the similar layout and theme as the login activity. In each activity, there is a text at the bottom of the layout, linking to each other by clicking on it.

* Forgot Password Activity:

In the Login Activity, when clicking on the text “Forgot your Password?”, it will lead this activity. This includes an EditText view for the user to enter his/her email. When clicking on the Confirm button, an resetting email is sent to then enterred email address. E.g:

* Home Activity:

This is the base activity in the app, containing the list of purchasable cryptocurrencies in a Recycler View. This has a searching (in the menus) and sorting function (clicking on the buttons of Hot/Price/24h Change)

* Profile Activity:

This contains some information of the User, including his/her full name, avatar, and most importantly the history of transaction as a Recycler View.

* Wallet Activity:

This has a Recycler View, designed for the user to view all of their cryptocurrencies in his/her wallet. User can also click on each item to go to Trading Activity to sell his/her corresponding cryptocurrency

* Settings Activity/Fragment:

This activity is created for app settings, such as Profiles editting, Notifications allowing, Currency display, Information about the fictional organization, some FAQs, and a Logout buttons.

# Level 2: App evidence

A database of “transactions” was created using Firestore functionality (provided by Google Firebase), storing the data of users’ history transaction.

The data of cryptocurrency assets (coin’s ID, current price (in dollars), recent changes in percentage, …) for the corresponding RecyclerView in the Home Activity will be set up using the same functionality.

# Level 3: Extended research evidence

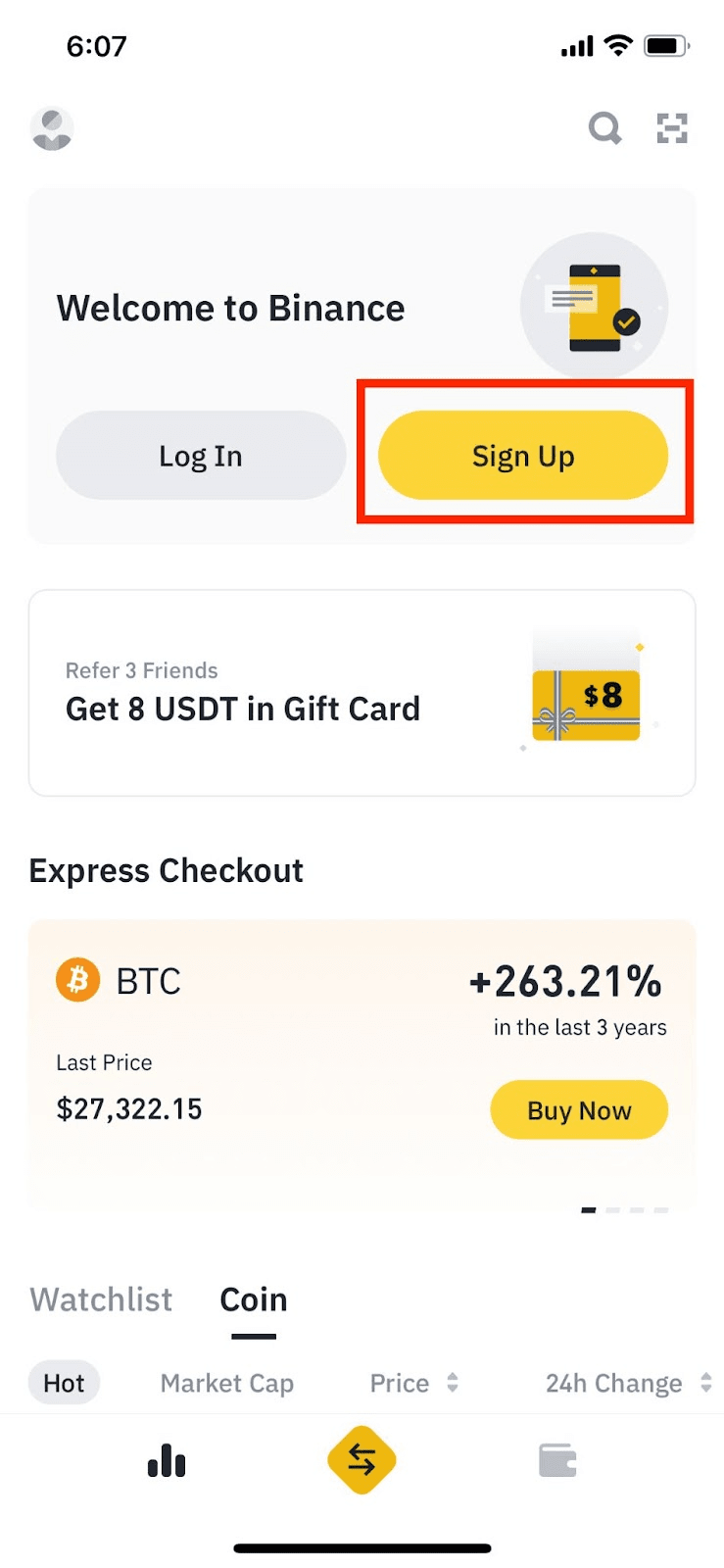
# References

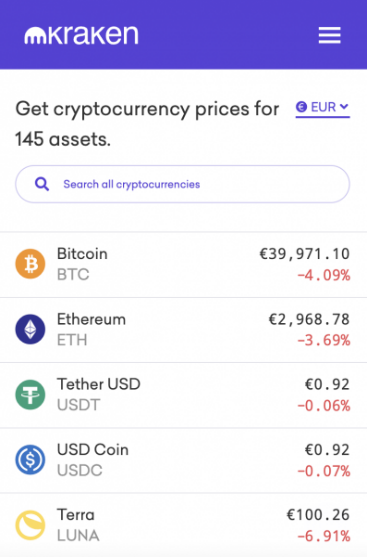
# Appendix 1 - Weekly reports

## Week 7

*Note: I have completed Core 1, submitted Core 2, but not completed the discussion tasks. However, I decided to continue with the custom project as the resubmission for those discussions would not be available until 6 Oct 2023 (end of week 9)*

In this step, I focused mainly in searching and referring to the cryptocurrency applications on the market, especially those available on the Android platform:

* Coinbase: In my opinion, this app has an user-friendly interface, ideal for beginners. It offers a secure and regulated platform for buying, selling, and managing cryptocurrencies.
* Binance: This offers not only a wide range of cryptocurrencies for trading, but also additional features like advanced charting tools. It's one of the most popular app among experienced traders.



* Kraken: I firmly believe that this app is well-known for its strong security features, providing access to a variety of cryptocurrencies and trading pairs.

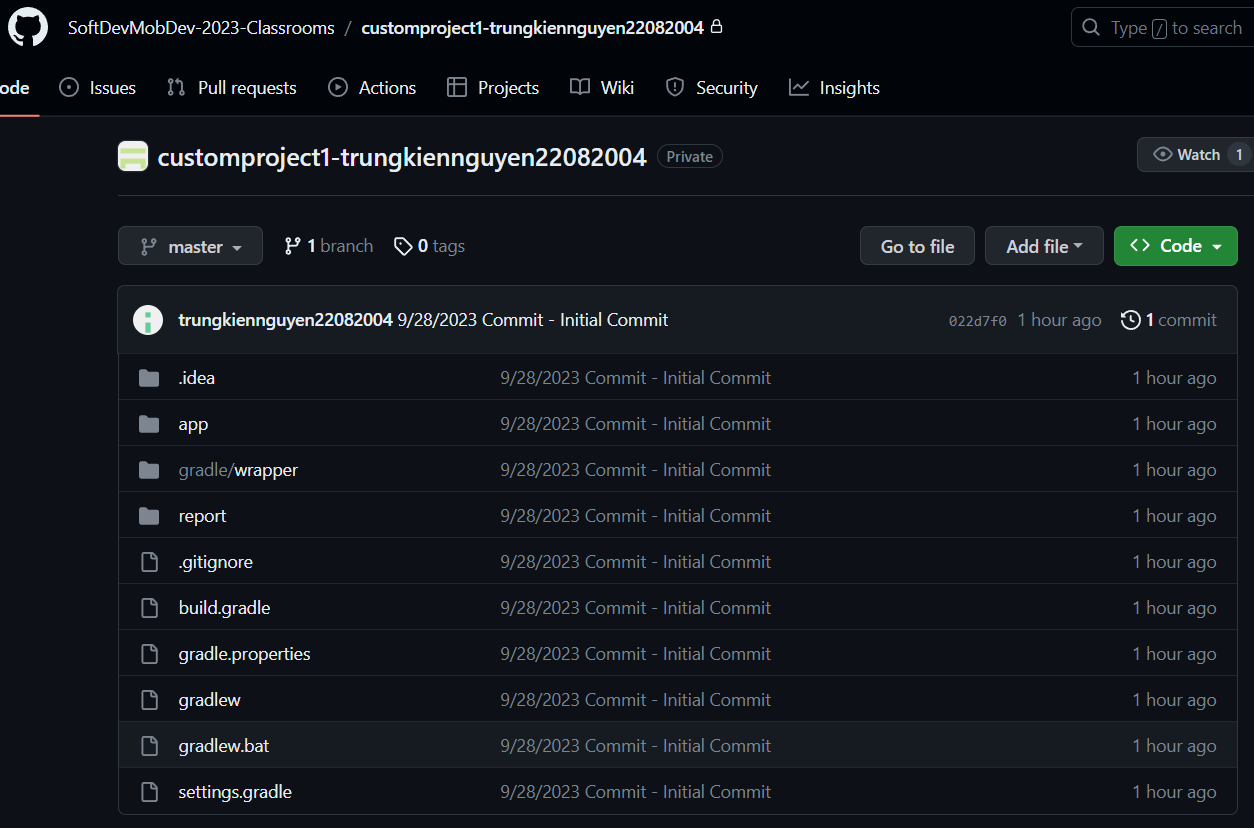
In the following week (Week 8), I think I will try to complete the Authentication feature of the app (Login/Signup). I prefer to use the Google Firebase’s authentication functionality.

## Week 8

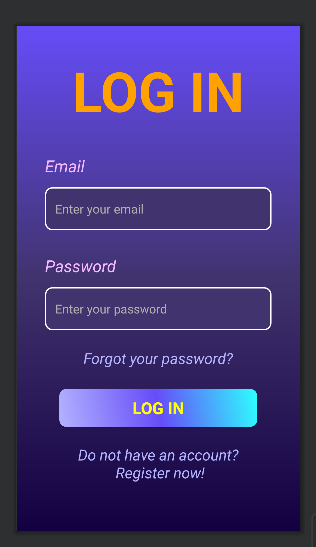
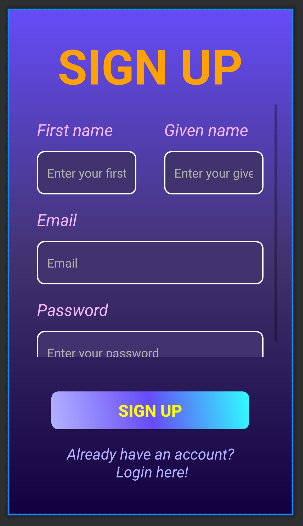
*Note: I have completed Core 1, submitted Core 2, but not completed the discussion tasks. However, I decided to continue with the custom project as the resubmission for those discussions would not be available until 6 Oct 2023 (end of week 9)*

To begin with, I have set up the repository for the Custom Project via the Github Classroom link on Canvas:

<https://github.com/SoftDevMobDev-2023-Classrooms/customproject1-trungkiennguyen22082004>



For the Authentication task, I have implemented two activities, including LoginActivity and SignupActivity.

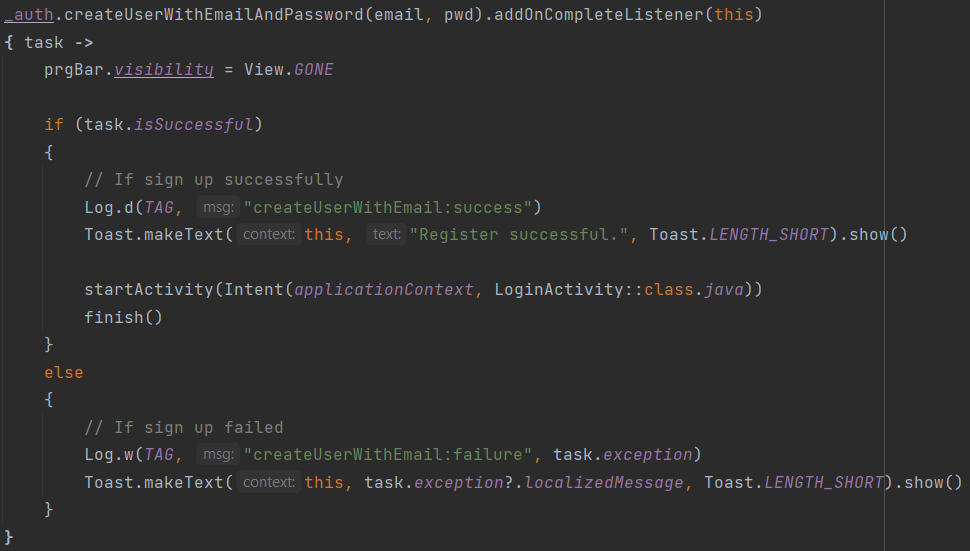
*LoginActivity’s layout SignupActivity’s layout*

I have use the Google Firebase’s functionality of Authentication using Email/Password:

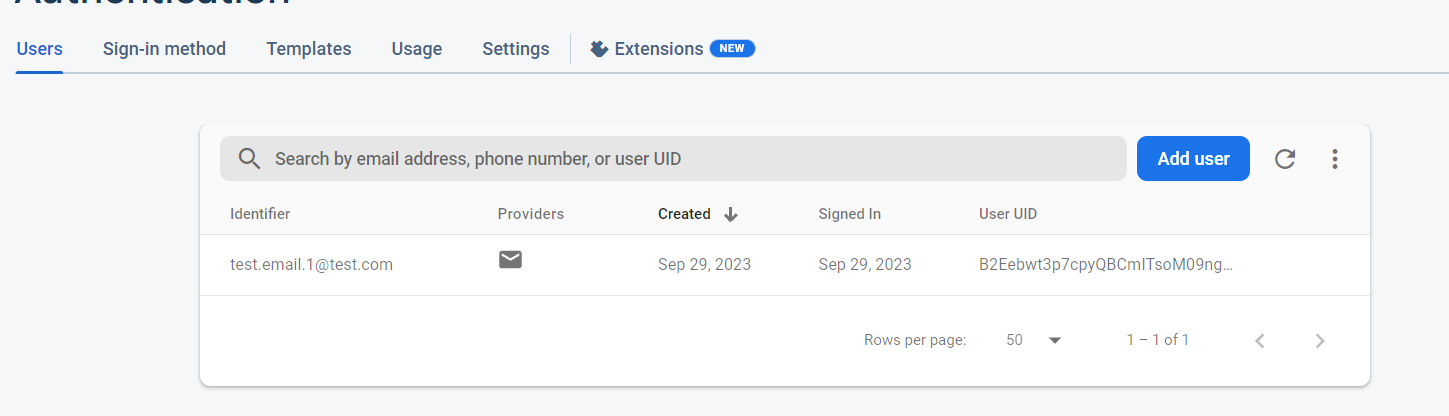
* Login: Using the method “signInWithEmailAndPassword()” of FirebaseAuth



* Signup: Using the method “createUserWithEmailAndPassword()” of FirebaseAuth



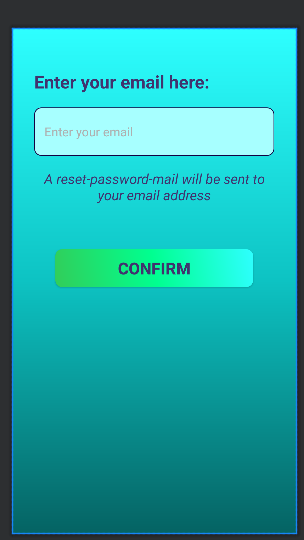
I have use “Signup” to create some new users for testing:



## Week 9

*Note: I have completed Core 1, completed Core 2, I have just submitted the Redo for discussion tasks since they had been available in the Friday morning.*

*Firstly, I continued my work on the Authentication functionality of my custom app, adding the forgot-password and reset-password feature:*

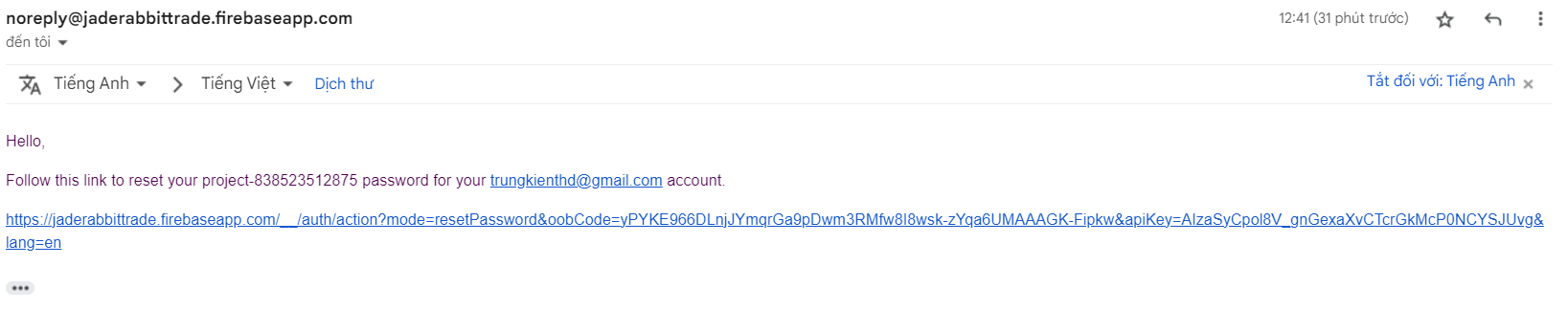
**

*ForgotPasswordActivity’s layout*

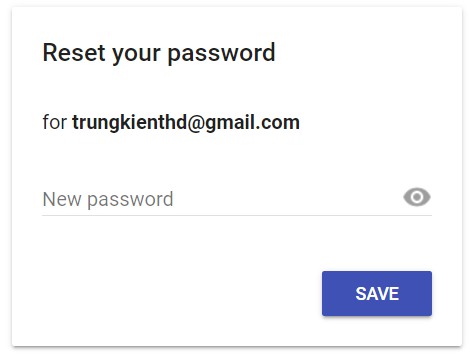
I have used the “sendPasswordResetEmal()” method to sent a resetting-password email to the entered email address:



I have tested that functionality, the resetting-password email will look like:

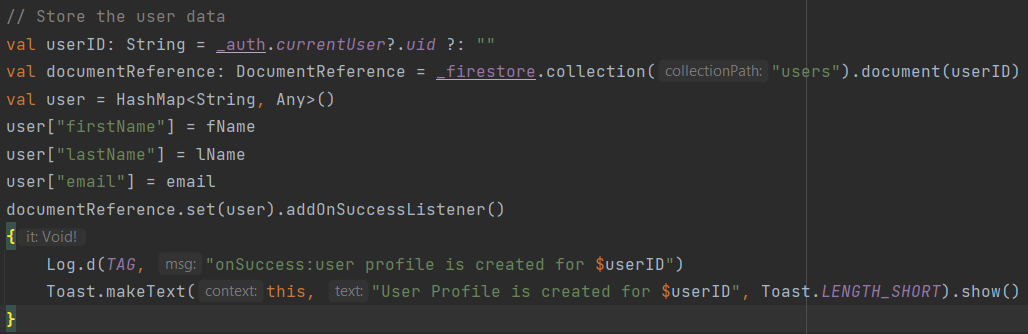


And this is the resetting-password interface:

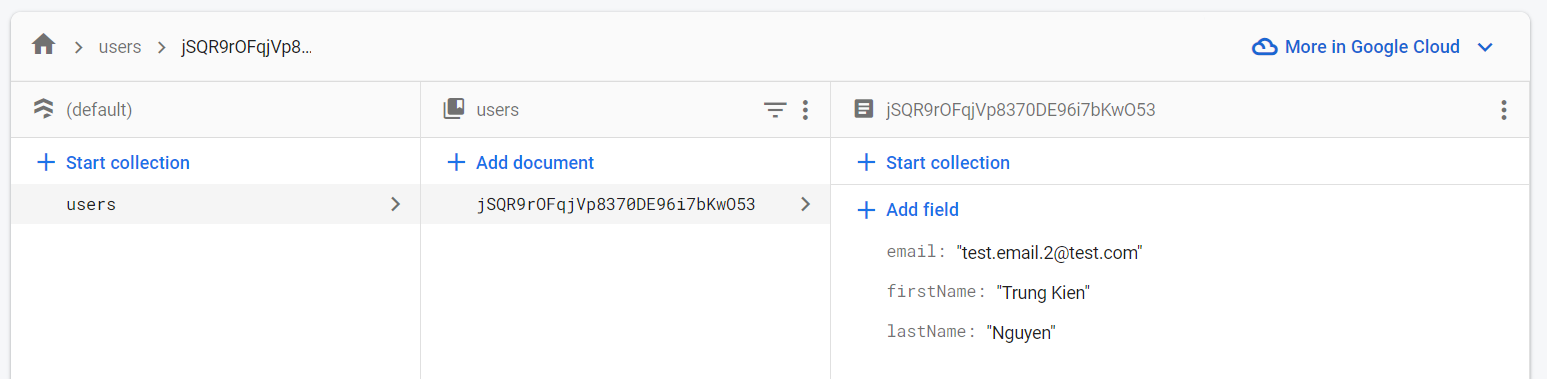


Next, I stored the User data (First Name, Last Name, …) in Firebase’s feature of Cloud Firestore

After create a Cloud Firestore Database, I have added the “Firestore.DocumentReference.set(user)” method to store the First name, last name and email of the user to the Firebase Cloud Firestore’s database during the signing up process:



I have created a new account for testing this function, and this is the output:



In the following weeks, I will continue to make the Market activity (for displaying digital assets of cryptocurrency), Transaction activity (for displaying history transaction of the user), and probably the User Activity (for viewing and modifying the user’s details)

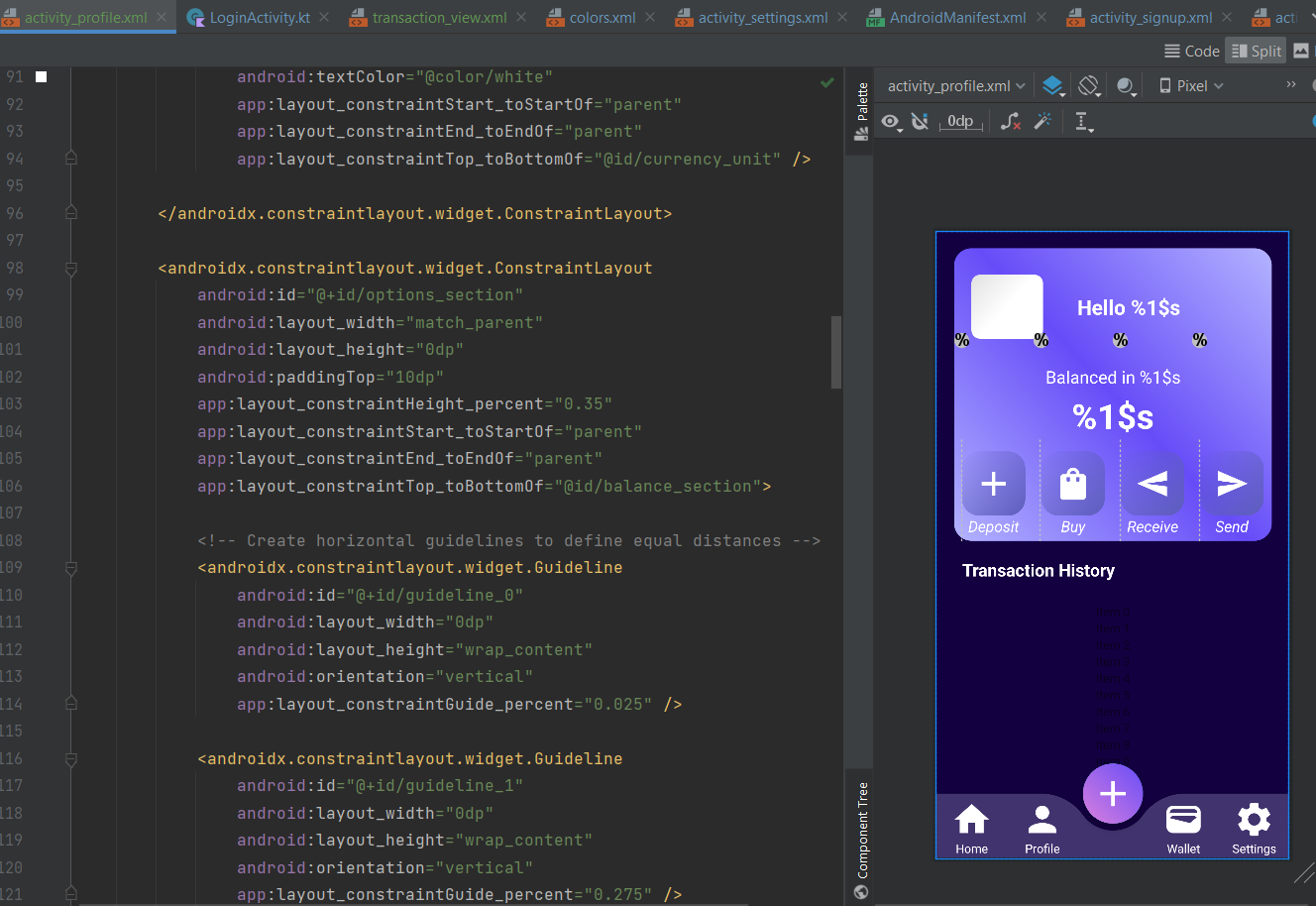
## Week 10

*[NOTE: as of week 10, if you have not completed Core 1, not completed Core, not submitted Core 3 nor an extension task, you will also need to justify why you should be encouraged to continue with a custom project. There is no point focusing on this task when the basics are not complete and your progress report will be marked as incomplete.]*

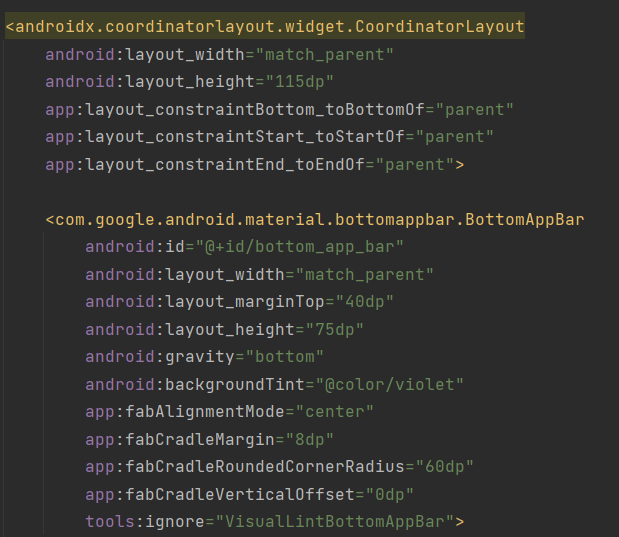
Note: I have added the design for the main activities for my app in the “Level 1: Design evidence” part

* Profile Activity:

The layout for this activity was quite easily to created for me, using a number of ConstraintLayouts and ConstraintLayout.widget.GuideLines:



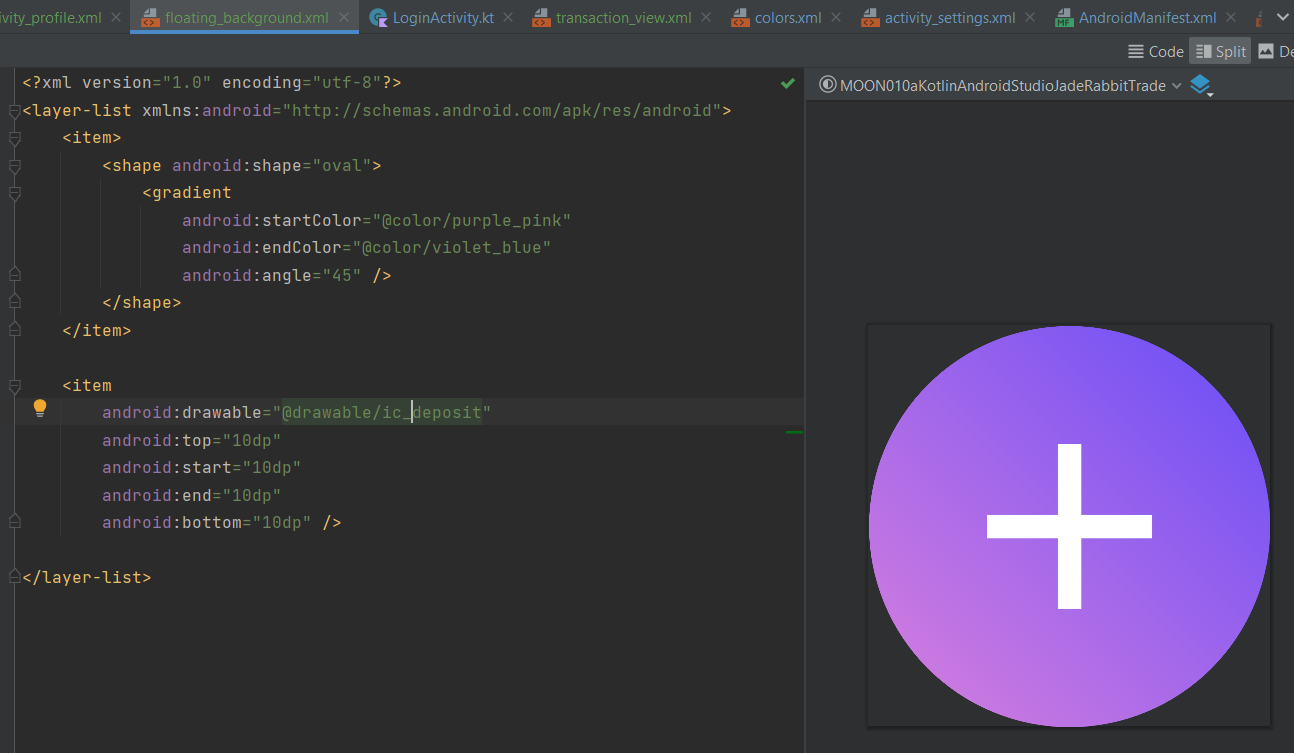
For the Navigation tab, currently I left it inside the Profile Activity instead of creating it in a separate layout file. I have used the BottomAppBar inside a CoordinatorLayout for the base of the navbar:



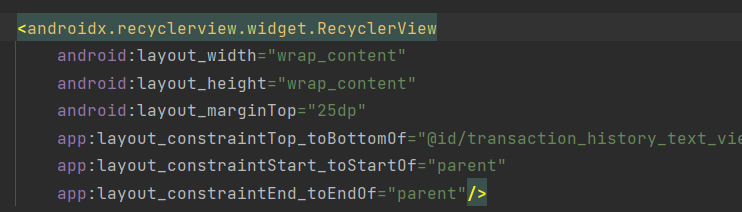
And the ConstraintLayouts, each contains a ImageButton and a TextView, for the functionality buttons:



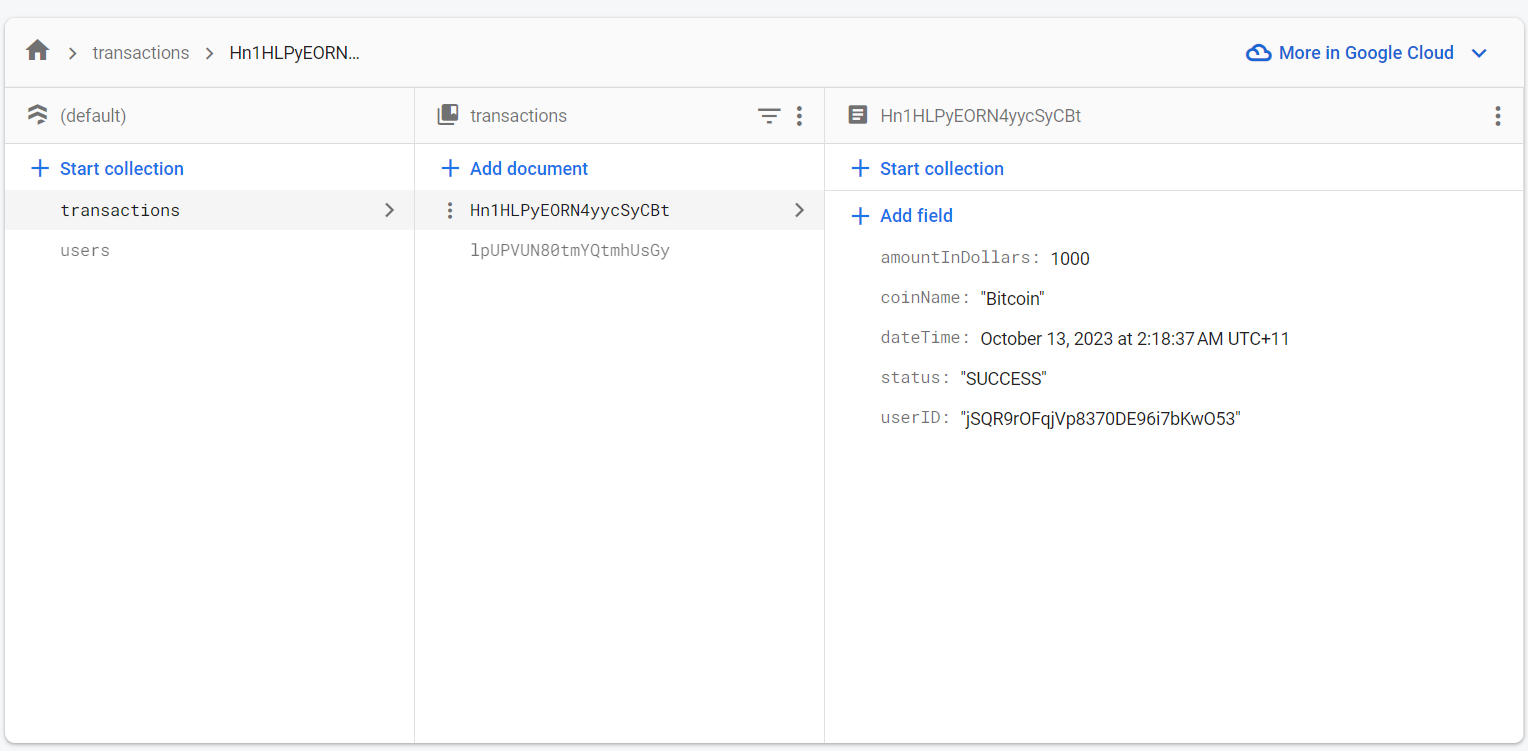
Following that, a float button of the navbar was created using FloatingActionButton, with the drawable of “floating\_background” as the foreground, which are shown as below:



The next part is the RecyclerView consisting the transaction history:



For the database, I have again used the Firestore functionality, creating a database named “transaction”:

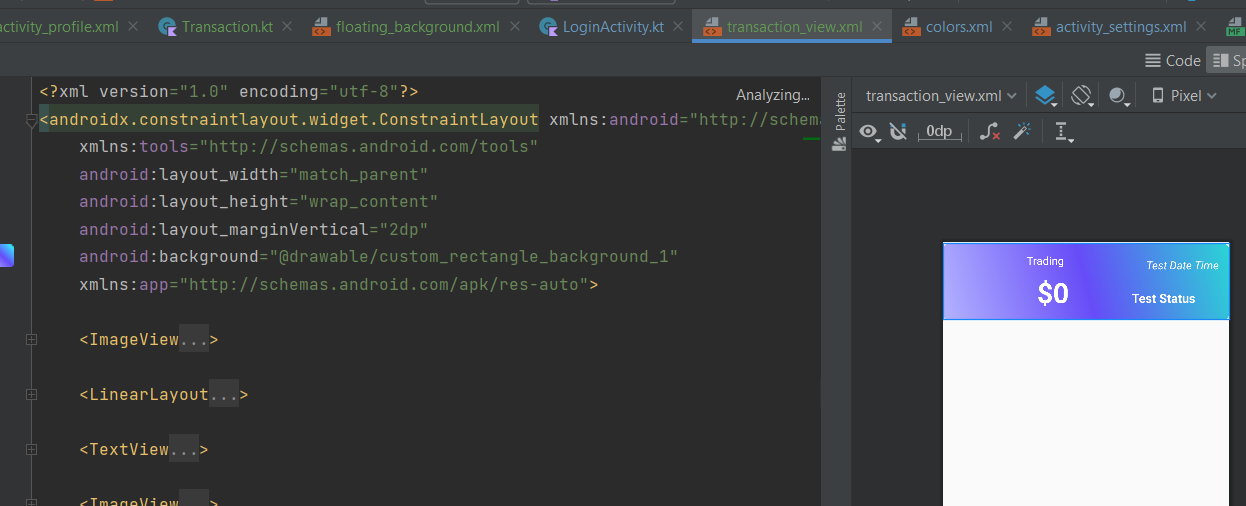


*The records were entered manually by me instead of implementing in the code, as the Buy/sell functionality of the HomeActivity had not been done yet*

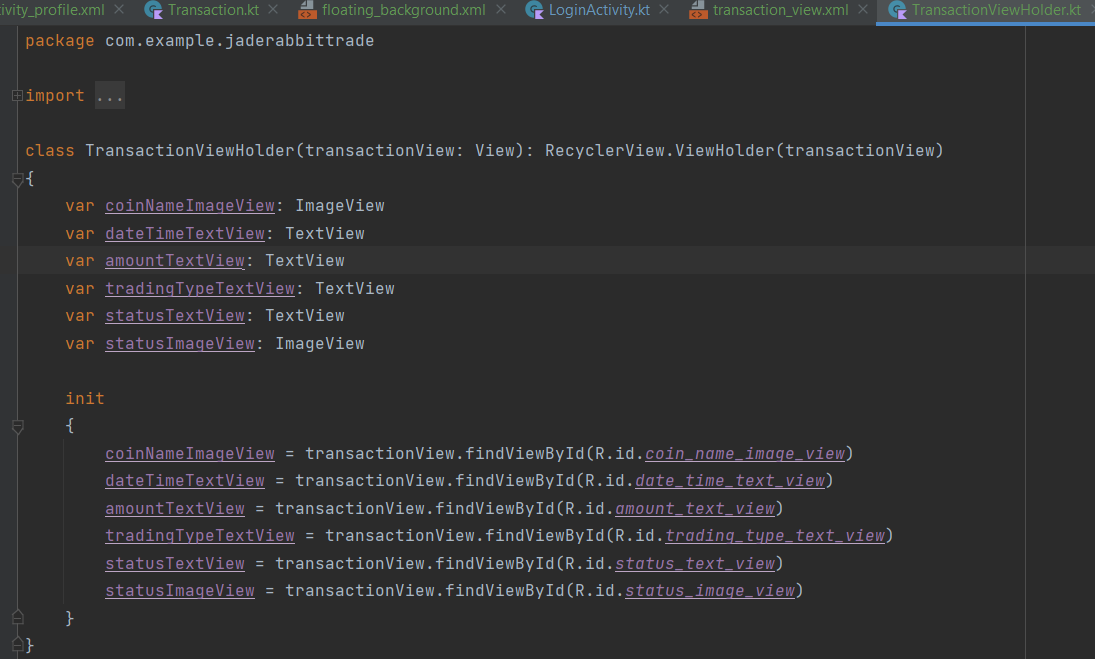
And the corresponding Kotlin class, storing the information of a transaction, named Transaction:



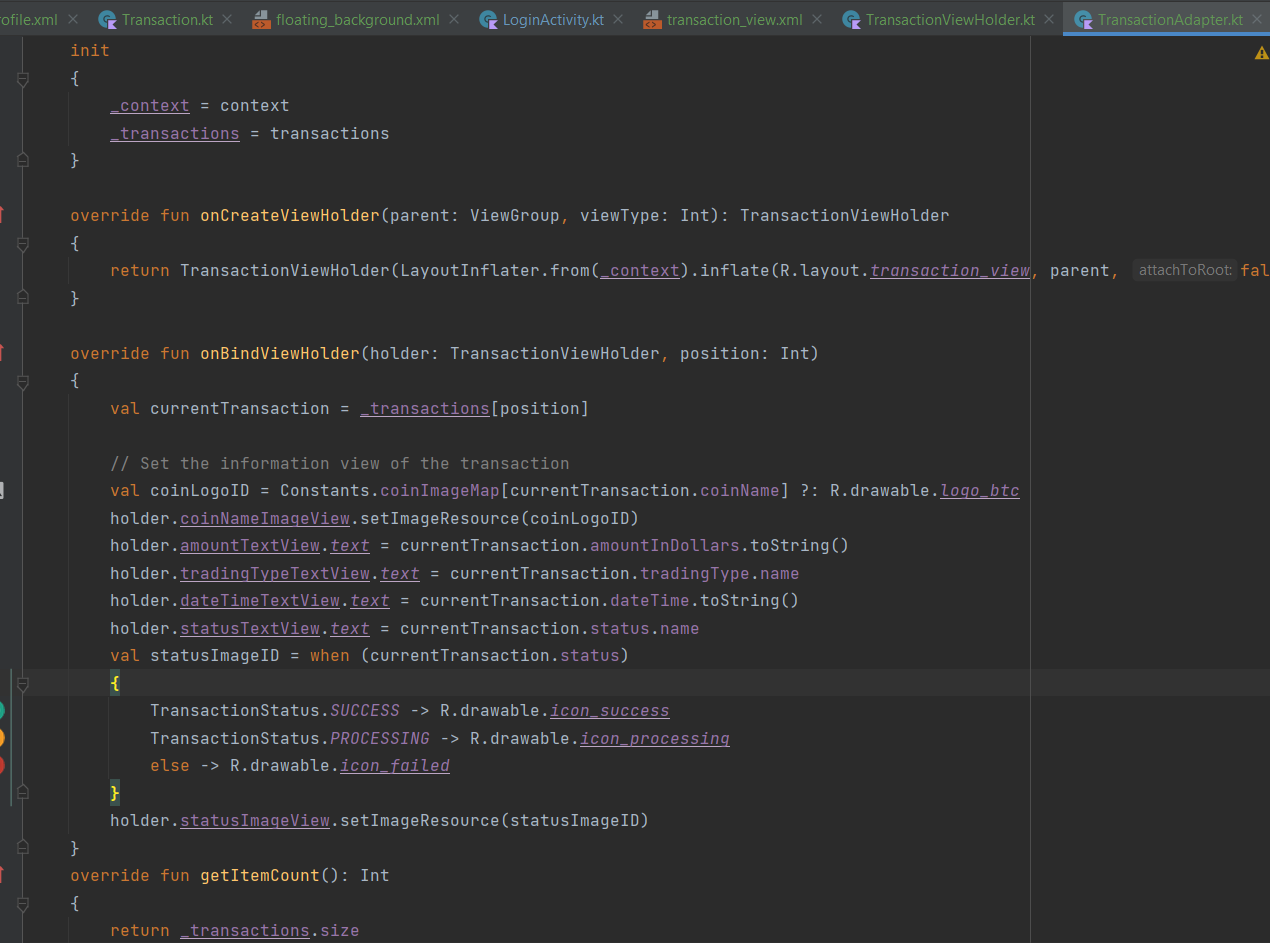
The layout for each element in the RecyclerView, which is “transaction\_view.xml”:



The ViewHolder class for initializing the views/widgets in that layout (TransactionViewHolder):



And the adapter for the RecyclerView, using those TransactionViewHolders:



I’m in progress of creating Home and Settings activities with the floating navbar, which is expected to be done by this Sunday (10/15/2023)

**My questions for this week:**

* For the task “Architectural design, with the diagrams as an appendix”, do I have to include them here on the weekly reports, or just add in the final one? If they need to be added here, can I add them in the next (final) weekly report?
* Can you please give some further instructions for this requirement, or may be an example? I have read that “UML might not be suitable for your app”, but is it satisfactory to use them?

## Week 11

*[NOTE: as of week 11, if you have not submitted/completed all other Core/Extension tasks, you will also need to justify why you should be encouraged to continue with a custom project. There is no point focusing on this task when the basics are not complete and your progress report will be marked as incomplete.]*

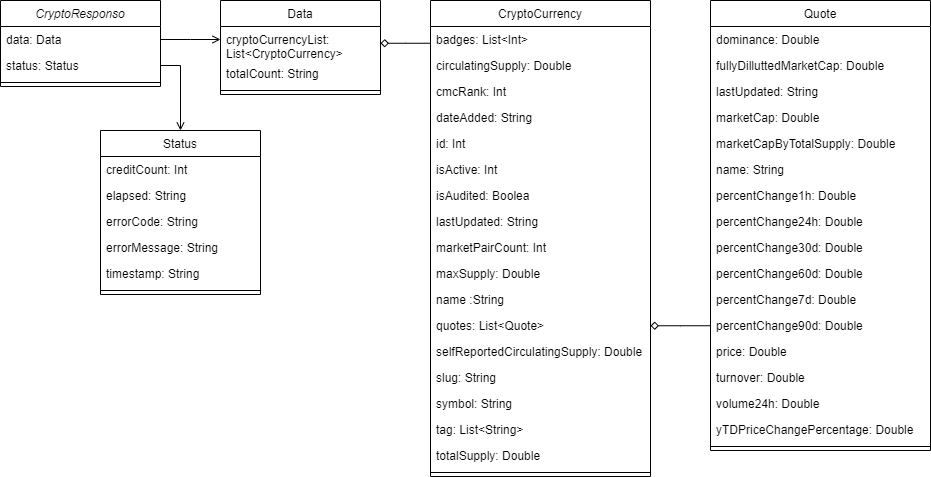
My work in this week is mainly focus on the Home Activity, including retrieving the crypto assets from the free API of Coin Market Cap using Retrofit.

The link of the API:

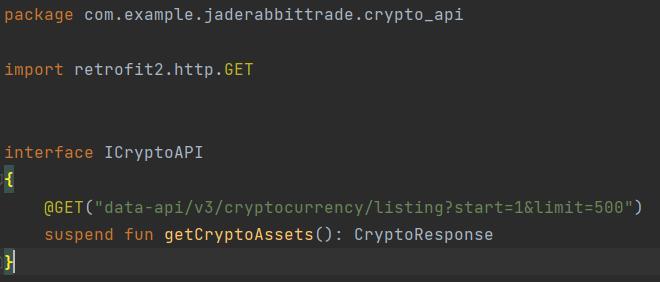
[api.coinmarketcap.com/data-api/v3/cryptocurrency/listing?start=1&limit=500](https://api.coinmarketcap.com/data-api/v3/cryptocurrency/listing?start=1&limit=500)



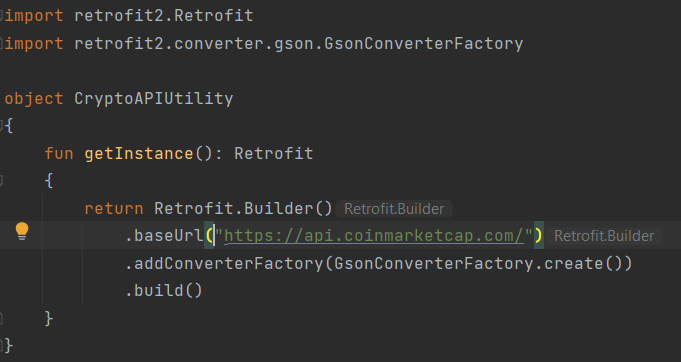
I have also create some classes to get the Crypto Assets Response from the API, which are shown in the following UML Diagram:



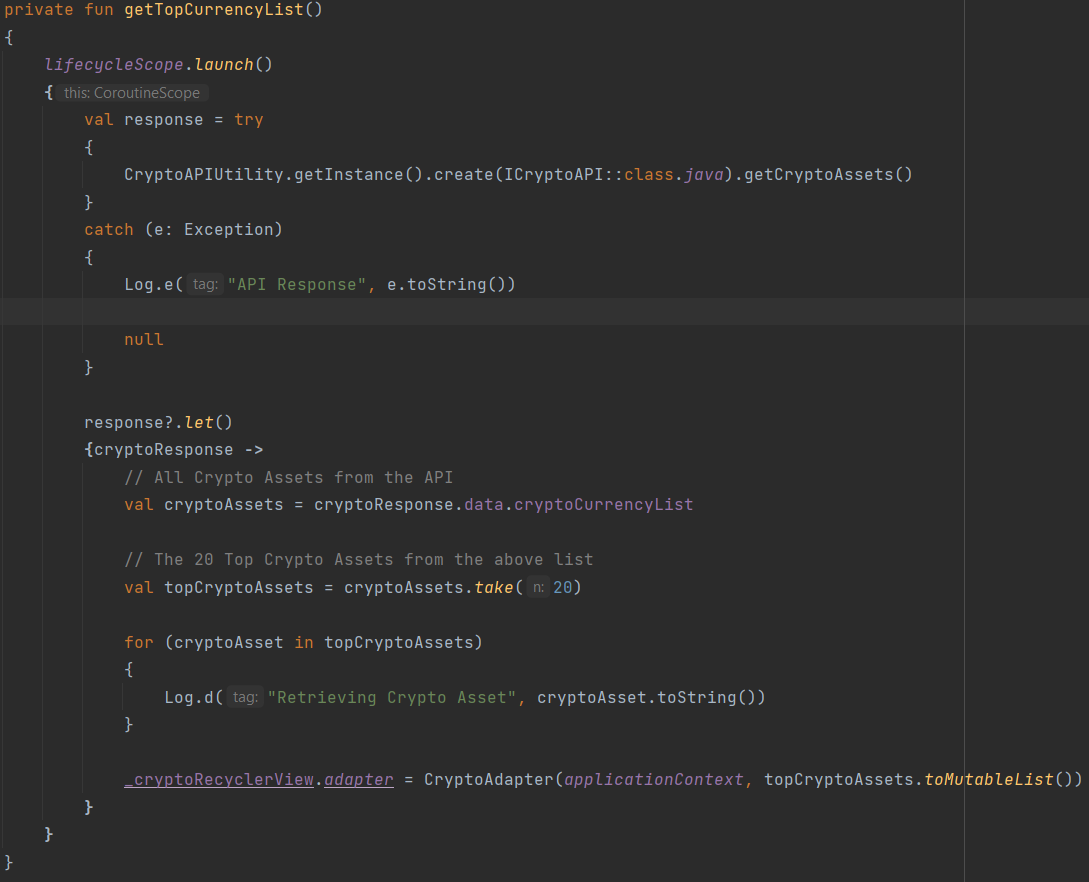
To retrieve the data from the API, firstly, I created the Kotlin interface of “ICryptoAPI” with the method “getCryptoAssets()”, defining the endpoint of the api (“data-api/v3/cryptocurrency/listing?start=1&limit=500%22”):



Then, I created a Kotlin object of CryptoAPIUtility. Its instance return a Retrofit object, with the base URL of the API:

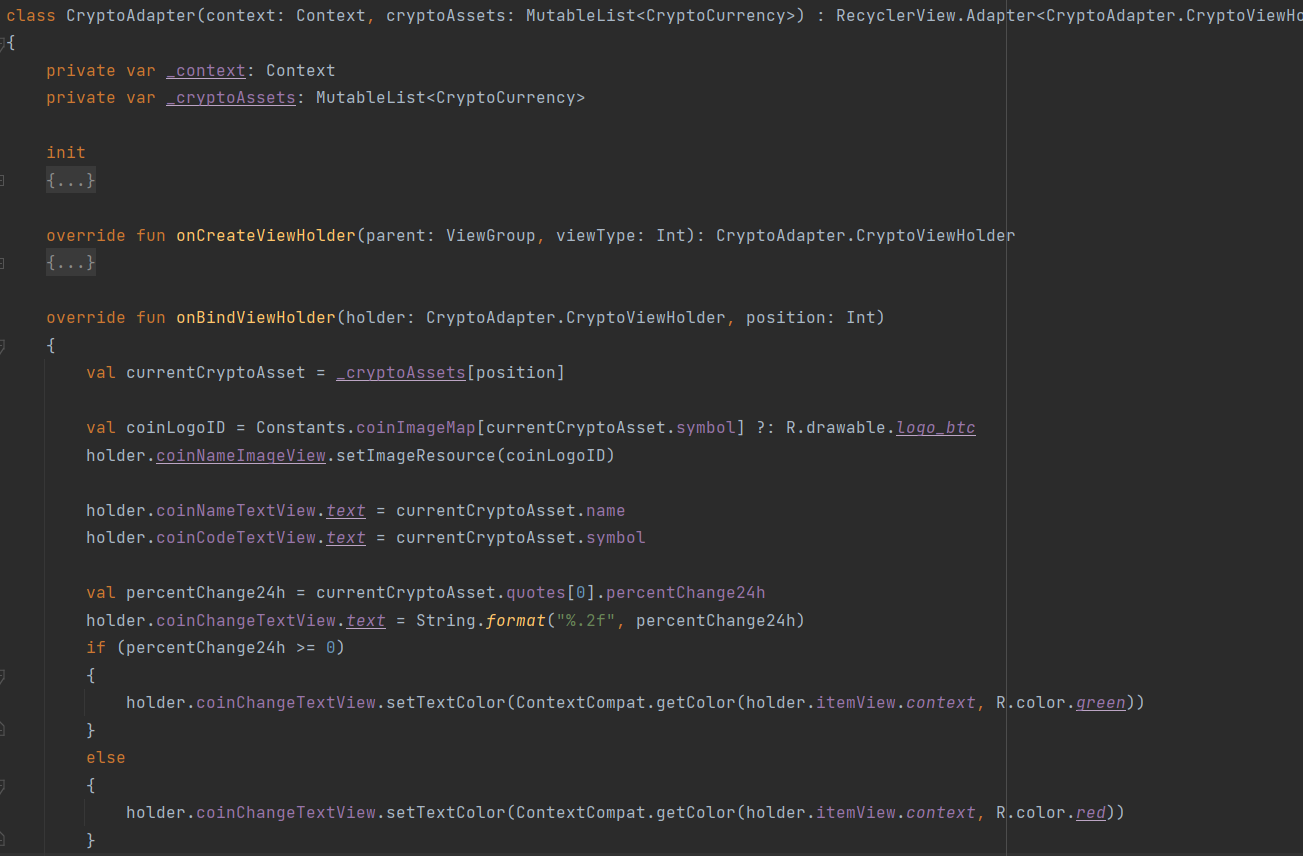


Finally, in the Home Activity, I created a method “getTopCurrencyList()” to get the top crypto assets in the API:



All the code is put in a Coroutine:

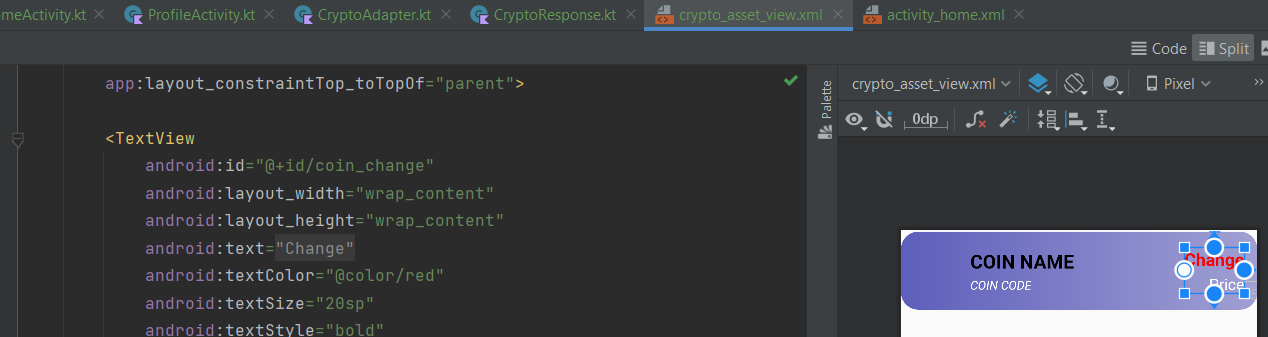
* Inside the coroutine, it tries to make a network request to a crypto API using Retrofit
* If response is not null, proceed with processing the data
* Inside the “let” block, it does the following:
  + Retrieve the list of cryptocurrency assets from the API response.
  + Select the top 20 cryptocurrencies from the list (This number can be changed easily to get more or less crypto assets in the API, for this app, I will use 20 top crypto assets from the API for easily visualizaing)
  + Finally, it sets up a RecyclerView to display that top 20 cryptocurrencies using a custom CryptoAdapter (shown as the below illustations). The adapter property of “\_cryptoRecyclerView” is set to a new instance of the CryptoAdapter, passing the applicationContext and topCryptoAssets as data to be displayed in the RecyclerView.





*CryptoAdapter’s class*

A custom View displaying each single Crypto Asset is also created:



And, this is the output:

The Green/Red Text displays the nearest 24 price change in percentage, while the below text displays the price in USD, which in the following week may swap to other currencies such as EUR (Euro) or AUD (Australian Dollars)

The search/filter ffunctionalities for this RecyclerView, as well as the images (icons) of the Crypto Assets have not been implemented yet, due to lack of time. I may finish them before the start of the week 12.