

How to Use this Template

1. Create a new document, and copy and paste the text from this template into your new document [Select All → Copy → Paste into new document]
 2. Name your document file: “**Capstone_Stage1**”
 3. Replace the text in green
-

Description

Intended User

Features

User Interface Mocks

Screen 1

Screen 2

Screen 3

Screen 4

Key Considerations

How will your app handle data persistence?

Describe any edge or corner cases in the UX.

Describe any libraries you'll be using and share your reasoning for including them.

Describe how you will implement Google Play Services or other external services.

Next Steps: Required Tasks

Task 1: Project Setup

Task 2: Implement UI for Each Activity and Fragment

Task 3: Build Main Market View

Task 4: Build Watchlist/Portfolio

Task 5: Build Settings/Add Item Activities

Task 6: Build Details Activity

GitHub Username: shuaybk (<https://github.com/shuaybk>)

Crypfolio

Description

This is a cryptocurrency price tracking app. It displays the current and historical prices of various cryptocurrencies. Additional features include the ability to set a personal crypto watchlist and to create and track your own personal portfolio of cryptocurrencies.

Intended User

Anyone that is interested in tracking crypto prices.

Features

The main page will be an overall market summary of the current top cryptocurrencies.

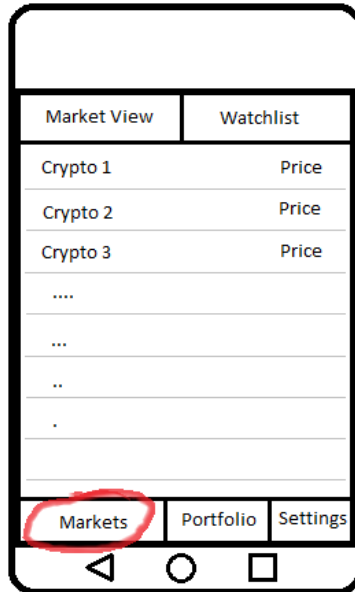
Additional features include:

- A personal watchlist of user selected cryptocurrencies
- A portfolio feature that allows a user to enter their crypto holdings for tracking the portfolio value
- A detail page for any selected cryptocurrency that will display additional details about the currency and provide a chart of the historical price action

User Interface Mocks

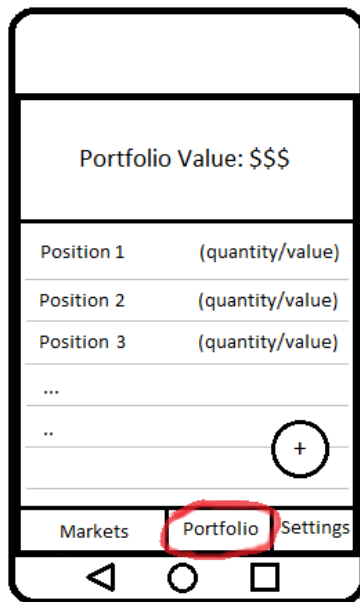
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Google Drawings, www.ninjamock.com, Paper by 53, Photoshop or Balsamiq.

Screen 1



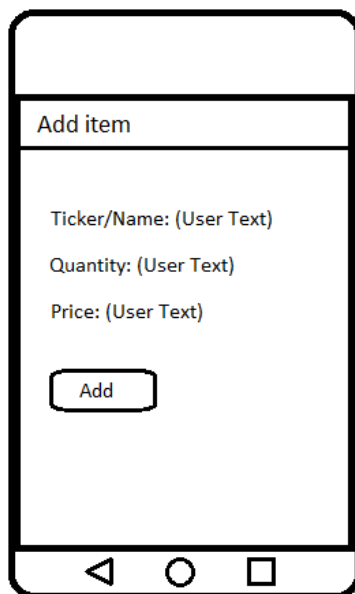
This is the main screen, defaults to Markets tab which allows the user to select the Market View or their Watchlist from the top tabs. The Market View and the Watchlist look similar.

Screen 2



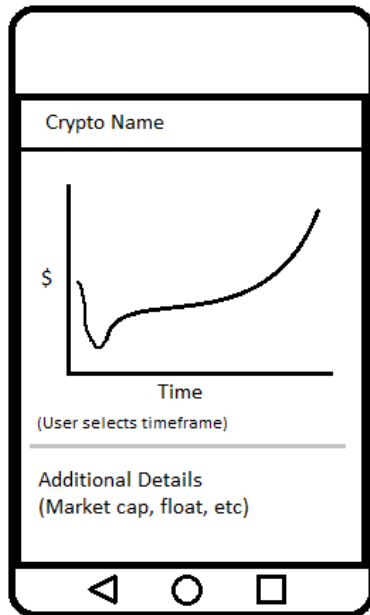
The portfolio page will show the user each of their positions and the current value. There will also be a FAB for adding new entries.

Screen 3



This is the add screen if the user is adding an item to the Watchlist/Portfolio. If it's the Watchlist, it will only ask for the Ticker/Name. If it's the Portfolio, it will ask for the Quantity/Price as well.

Screen 4



The details screen displays details for any cryptocurrency that is selected. It displays a chart (user can select the timeframe for historical data). It also displays additional details such as market cap, coin supply, etc.

Key Considerations

How will your app handle data persistence?

Data about the user's watchlist/portfolio will be stored locally on the device in a database using Room. The settings data will be saved locally using Shared Preferences.

Describe any edge or corner cases in the UX.

If the back button is pressed from any bottom tab or the details/settings screen, the user will be brought back to the Markets tab. If the user is already on the Markets tab, it will take the user to the device home screen.

To delete an item from the Portfolio/Watchlist, the user can do a left swipe or press and hold on the item (which will provide a delete option). A floating action button will be available on the Portfolio/Watchlist to add items as well.

Describe any libraries you'll be using and share your reasoning for including them.

- Picasso will be used for images due to its ease of use and my personal familiarity with it.
- Coingecko is the API for providing the cryptocurrency data (JSON format). It is a free service and provides all the required info.
- Volley library will be used for making the network requests to the API. This is easy to use and I'm familiar with it.
- Possibly use AnyChart library to create the graphs, however will need to test this. Alternatively will look up other libraries or use Android's built-in GraphView to generate the charts.
- Gson library will be used to parse JSON data. The library is well established/supported, and easy to use.

Describe how you will implement Google Play Services or other external services.

- Will use Google Play Services for ads in a free version. Paid version will remove ad related dependencies.
- The API is from Coingecko.com and provides the cryptocurrency data in JSON format.

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and break them down into tangible technical tasks that you can complete one at a time until you have a finished app.

Task 1: Project Setup

Add the required dependencies for the libraries above.

Task 2: Implement UI for Each Activity and Fragment

Build the basic UI components:

- Create the tabs and fragments (just the basic views, no real functionality) for each of the required screens. Include FABs and build functionality to switch between views properly. There will only be 4 activities:
 - The main view
 - The Settings

- The Add screen
 - The details screen
- There will be separate fragments on the main view for each of the below:
 - The main market view
 - The watchlist view
 - The portfolio view

Task 3: Build Main Market View

Build the functionality required for displaying the main market information

- Create helper classes for making network calls to the API for main crypto market information and storing crypto information
- Make the network calls and retrieve the data in JSON format
- Create a RecyclerView and display the data in a list format

Task 4: Build Watchlist/Portfolio

- Create two databases using Room that can store details about the users watchlist and portfolio (will store price info as well with LiveData). A separate database for watchlist and portfolio.
- Build Watchlist
 - Pull the watchlist items from its Room database and display saved prices
 - On start, refresh the current prices for the watchlist items (use LiveData to automatically update the prices in the view)
- Build Portfolio
 - Pull the portfolio items from its Room database and display the saved position values/prices
 - On start, refresh the current prices for the portfolio items and recalculate the position values (use LiveData to automatically update the prices in the view)

Task 5: Build Settings/Add Item Activities

Build the Settings activity:

- Save settings info to SharedPreferences
- Make setting option for preferred fiat currency (USD, CAD, etc)
- Add any other required settings

Build the Add Item activity:

- If it's for the watchlist, just get the ticker/name of the crypto and add it to the watchlist Room database.
- If it's for the portfolio, get the ticker/name, quantity, and purchase price and add it to the portfolio Room database.

Task 6: Build Details Activity

- Determine the crypto that needs details to be displayed
- Make network calls to retrieve this cryptos details (historical data, market cap, supply, etc) and parse the JSON. Create classes to store the data in temporarily.
- Create the chart for the historical data (and a selector for the user to pick the timeframe of the chart)
- Display the required additional information about the crypto

Add as many tasks as you need to complete your app.

Submission Instructions

- After you've completed all the sections, download this document as a PDF [File → Download as PDF]
 - Make sure the PDF is named "**Capstone_Stage1.pdf**"
- Submit the PDF as a zip or in a GitHub project repo using the project submission portal

If using GitHub:

- Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
- Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"