Name: Sadik Suny
CSE 4095 Lab 6

User Flow Diagram:

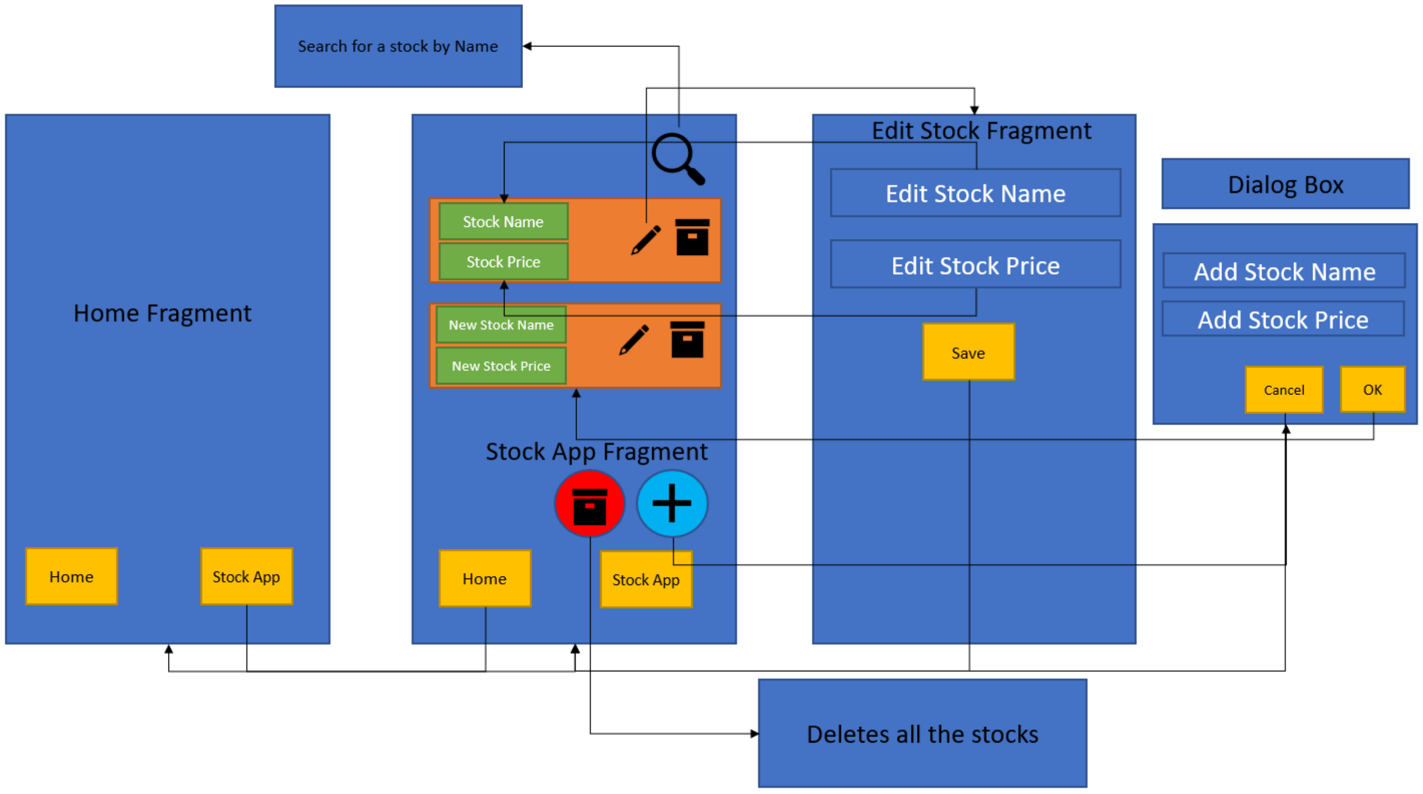


Figure: User Flow Diagram

Application Overview:

For my application, I am using MVVM or Model-View-ViewModel architecture. I am also using a Room SQlite database and Rxjava for database observer operations. I am using a bottom-tab front-end architecture with two fragments: Home and Stock App. Home fragment is used to launch the application.

Stock App fragment contains the recycler view with all the Stock Names, Stock Prices, Edit Buttons and Delete Buttons. Stock App fragment also contains an “Add Stock” button and a “Delete All Stocks” button. When the “Add Stock” button is clicked, a screen pops up where the user can add a stock name and stock price. After they are done adding, they can click the ok button to add a new stock to the recyclerview. Users can also click on the edit button which then opens up a fragment called “Edit Fragment” and there they can edit the stock by inputting the edited stock name, price and then they can save the edit by clicking the save button. Users can also search by stock names using the search bar on the Stock App fragment.

Application Architecture:

This application is a CRUD (Create, Retrieve, Update and Delete) application. I am using a room database for this application. I have a class called “Stock” which contains all the information about one stock. In the class, I have added the line @Entity (tablename=”stock”) to create a stock table so that each variable in the Stock class corresponds to a column in the database. Then I am using a StockDao interface to define all the necessary queries. The queries I am using for this app are: insert, delete, update, getStockByName, getAllStocks and deleteAll. Then I have a StockDB to instantiate a database for our database operation. Then I have a StocksViewModel class which contains a LiveData object containing a List<Stock> object. This viewmodel class is used for the ViewModel portion of the architecture and to save the data in the database.

In the StocksFragment, I am defining the recyclerview and the RxJava observer function. The observer function is used to store the data in the database using the queries I defined in StockDao. This fragment also contains the newStockDialog function which creates a new stock dialog. Moreover, the fragment contains the listeners for the “Add Stock” and “Delete Stock” buttons.

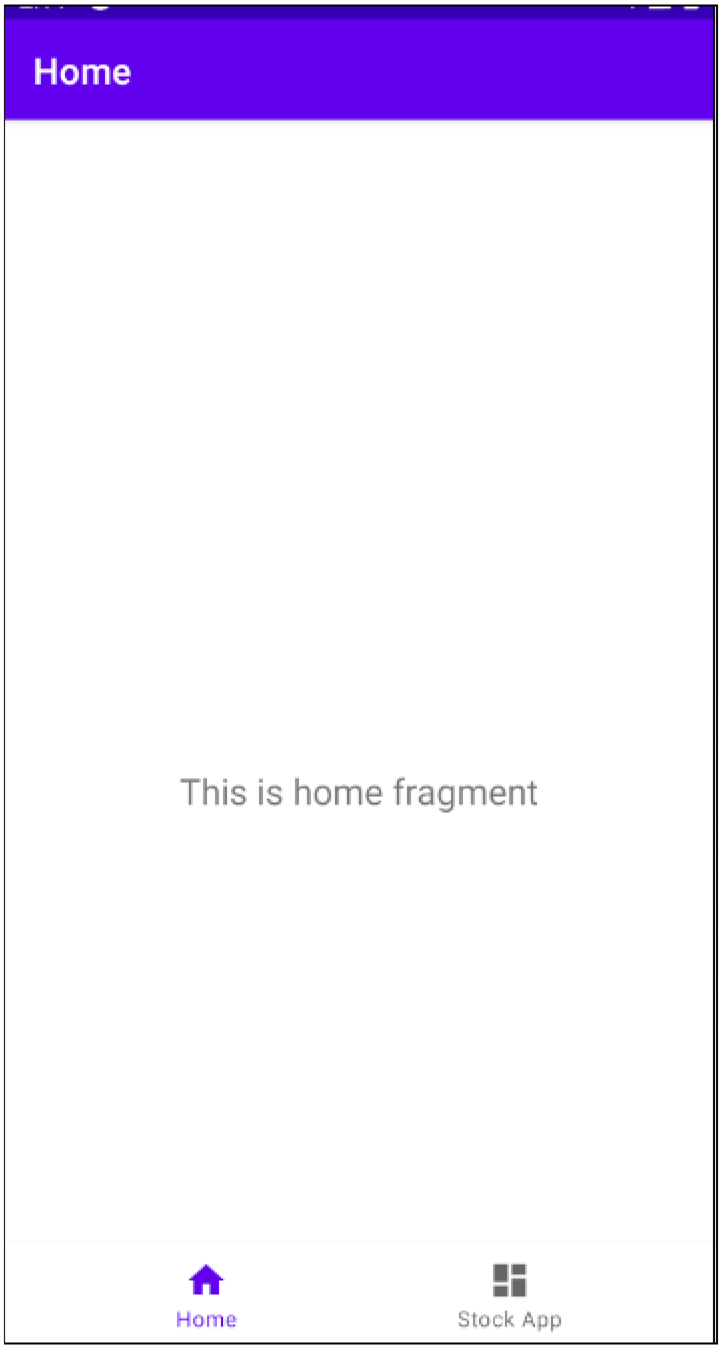
In order to keep the recyclerview consistent with the db, I am using a viewmodel observer function to update the stockList (Arraylist containing stocks) when something changes.

In my adapter class, I have onclick listeners for the edit and delete buttons for each cardviews. Then I have an interface called, onClickItemListener which the StocksFragment has to implement in order to use onDeleteClick to delete an individual stock from the fragment.

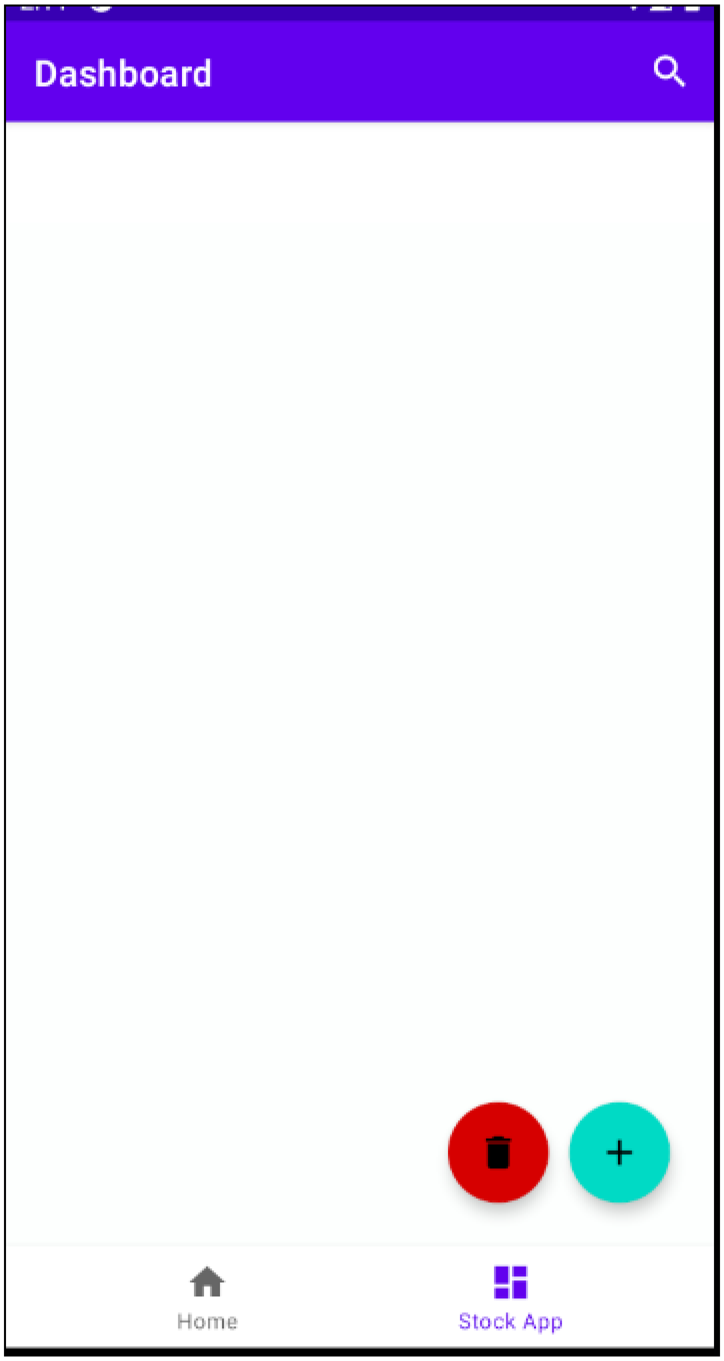
I am using a searchview to filter the recyclerview. I have a function which displays all the stocks after the searchview collapses. I also have a function which disables the soft input of a keyboard after the searchview collapses. I am using a Filter function in the adapter to create filters so that users can search for stocks by stock name.

Application Screenshots:

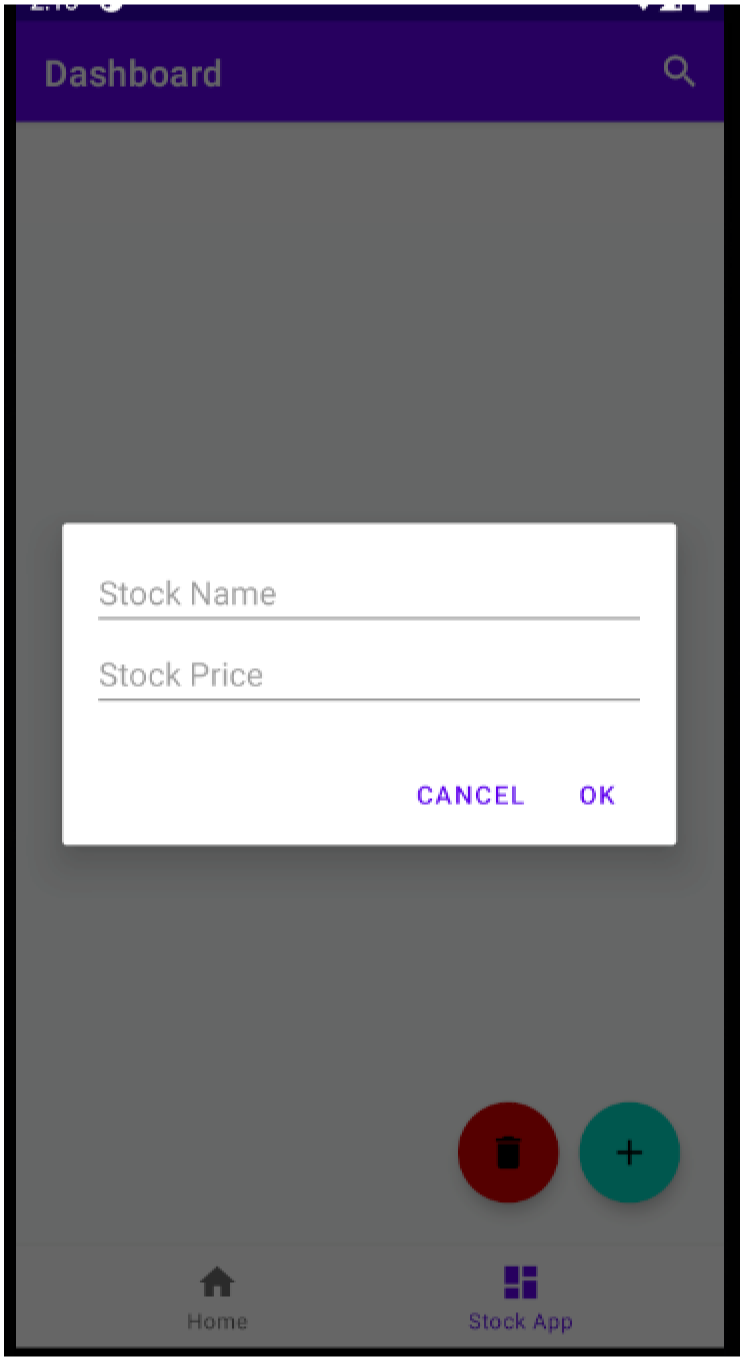
This is what the home fragment looks like:



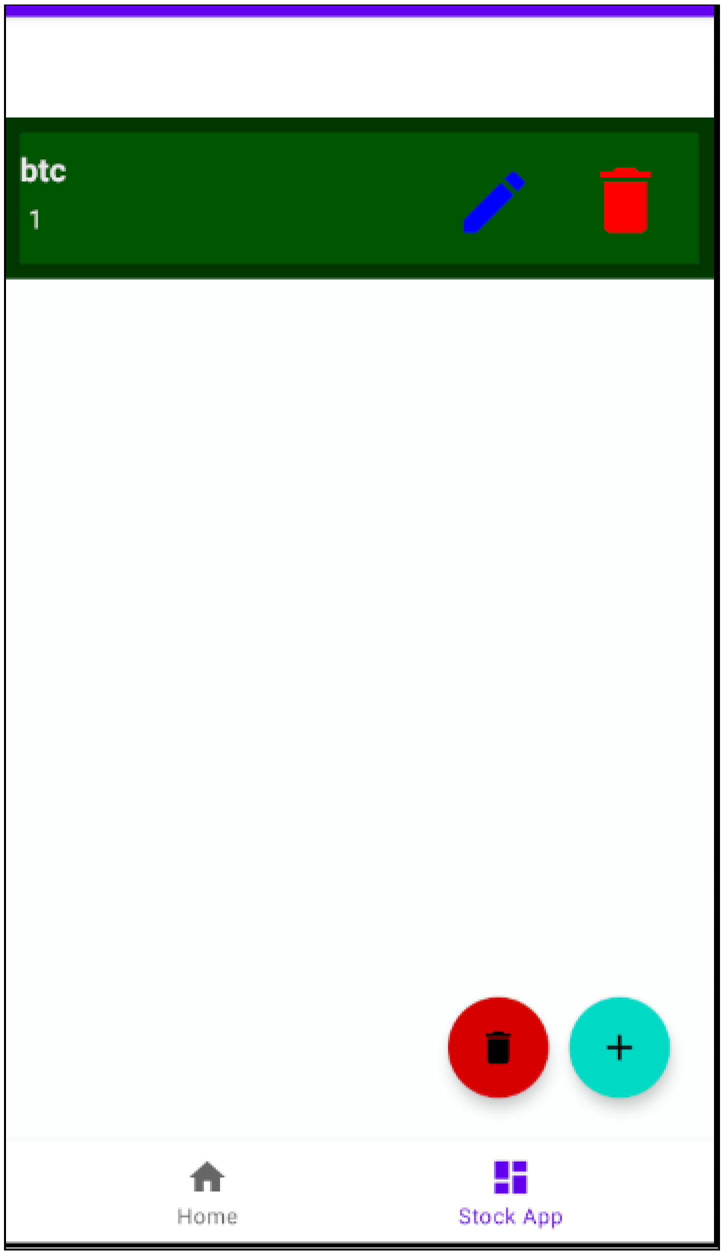
This is what the stock fragment looks like initially with no stocks:



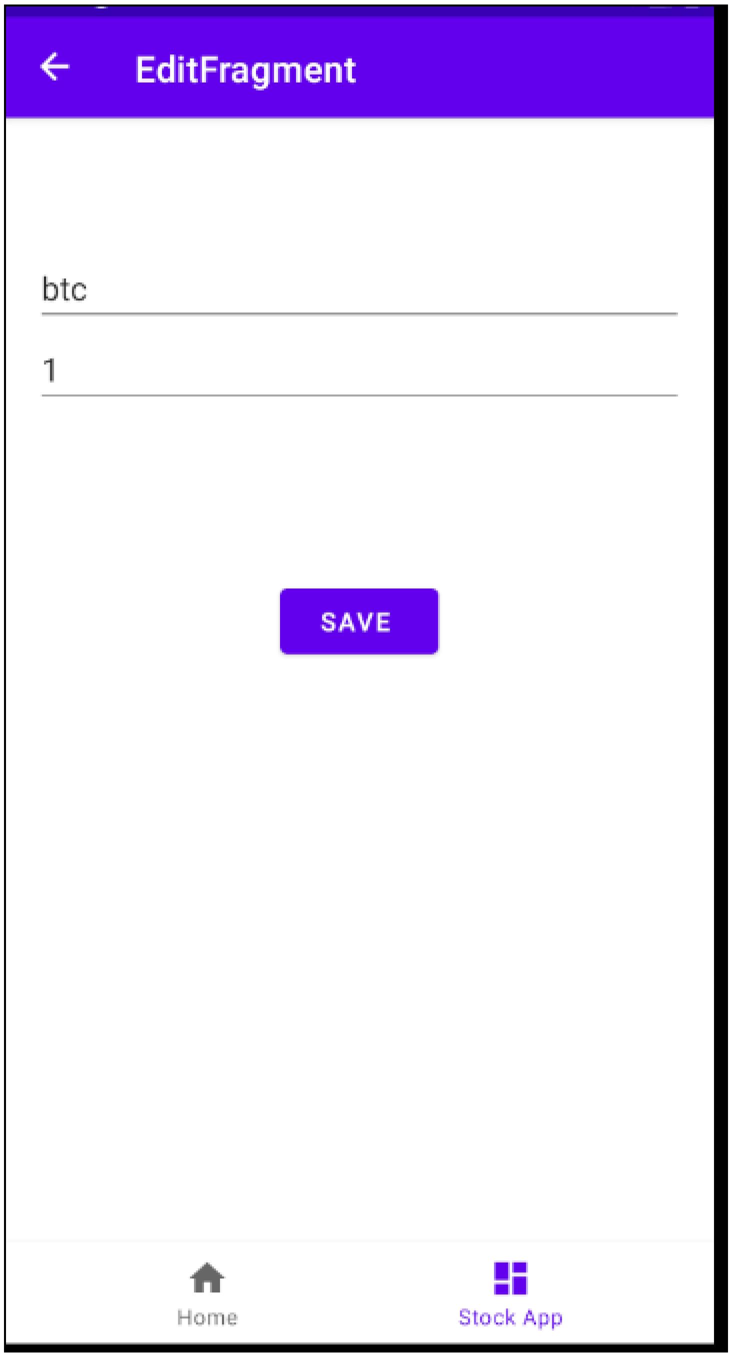
This is what the application looks like when I click the add button:



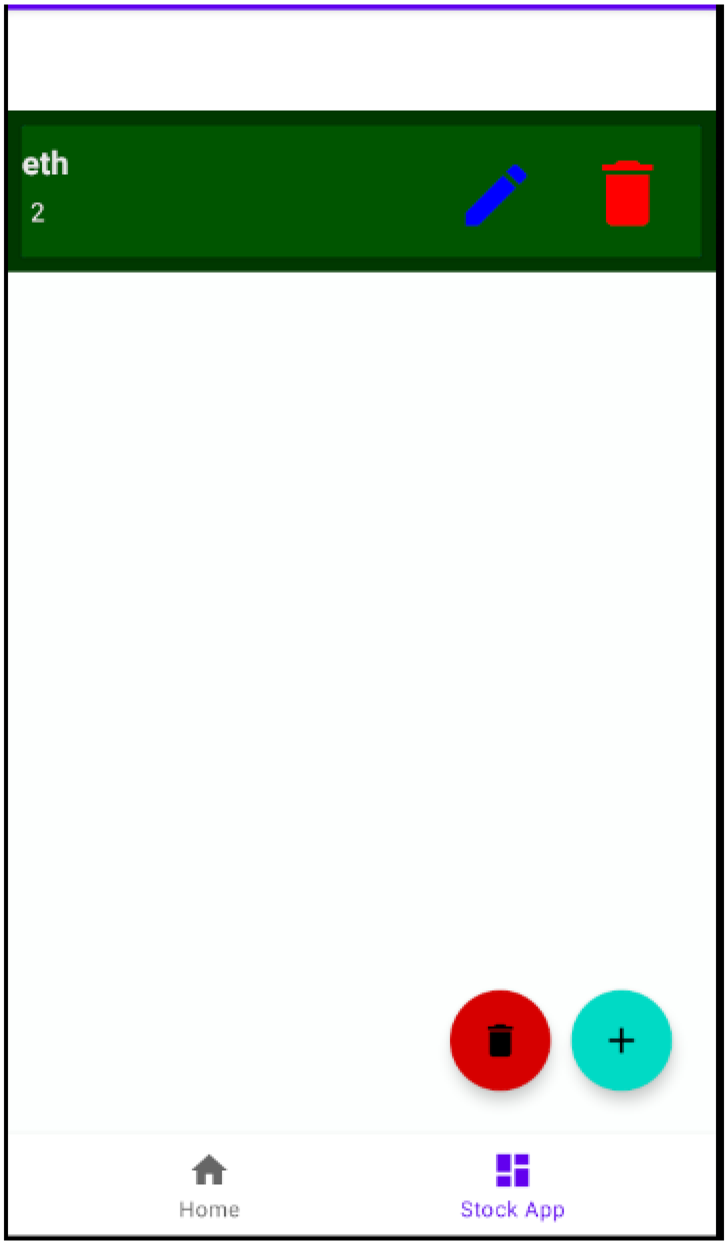
This is what it looks like after I add, stock: “btc” with price: “1”:



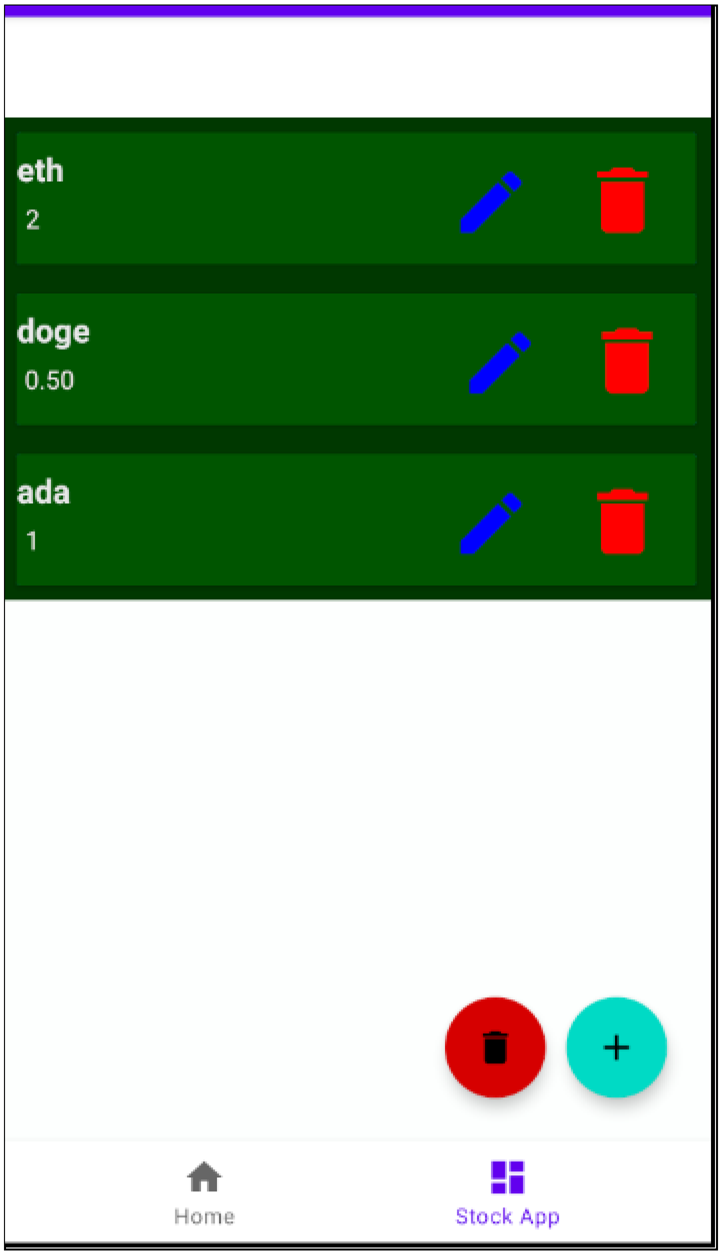
This is what the app looks like after I click the edit button (pencil icon in blue):



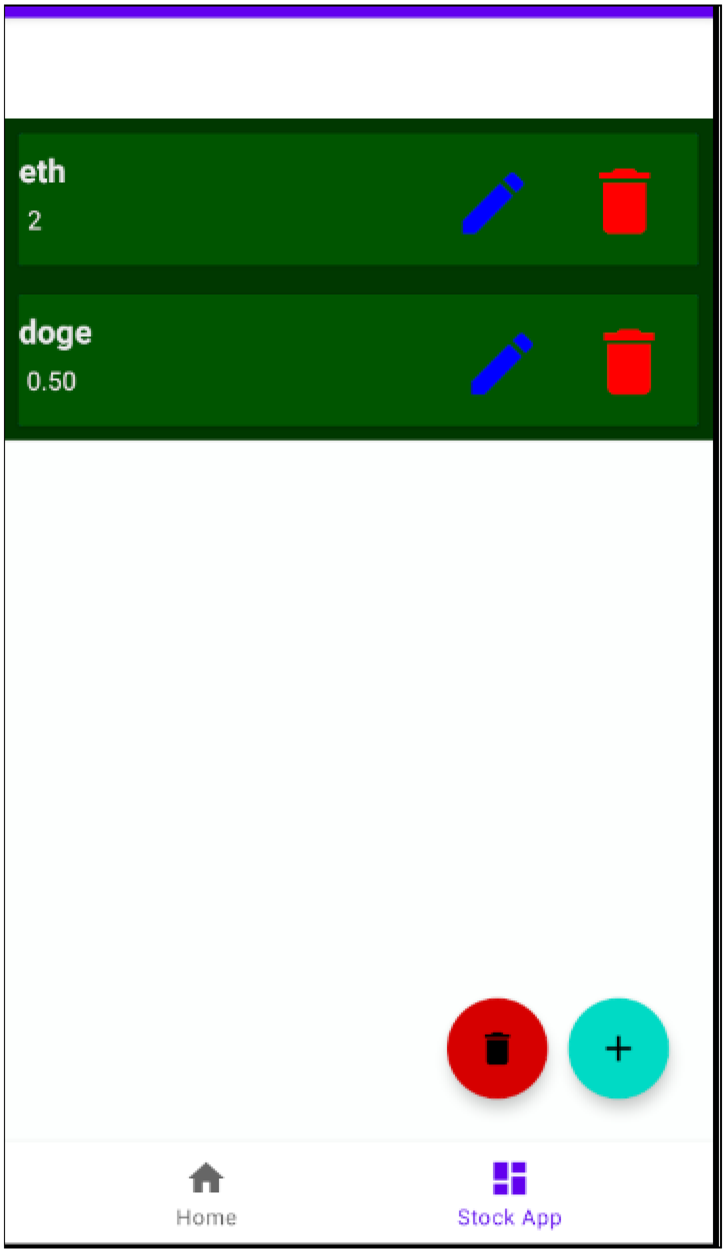
This is what the app looks like when I change the name to “eth” and price to “2”:



This is what the app looks like after adding more stocks:

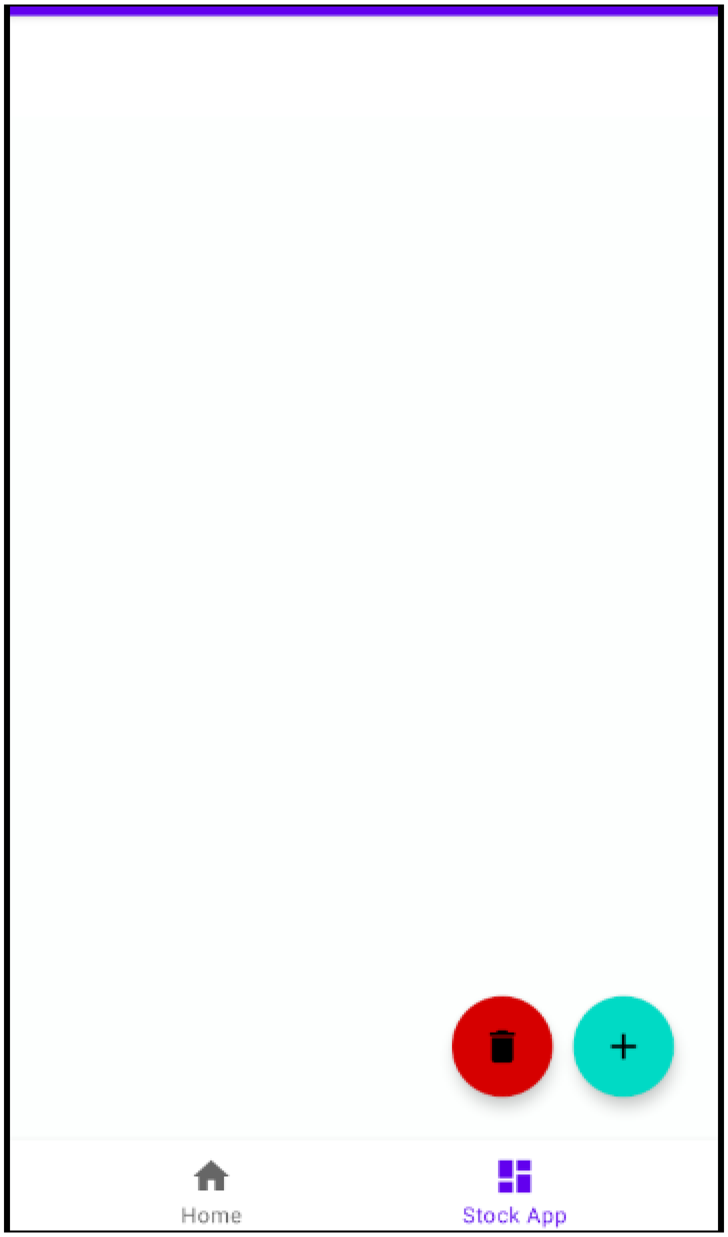


Now I am going to delete an individual stock (for example: “ada”) and this is what the app will look like:

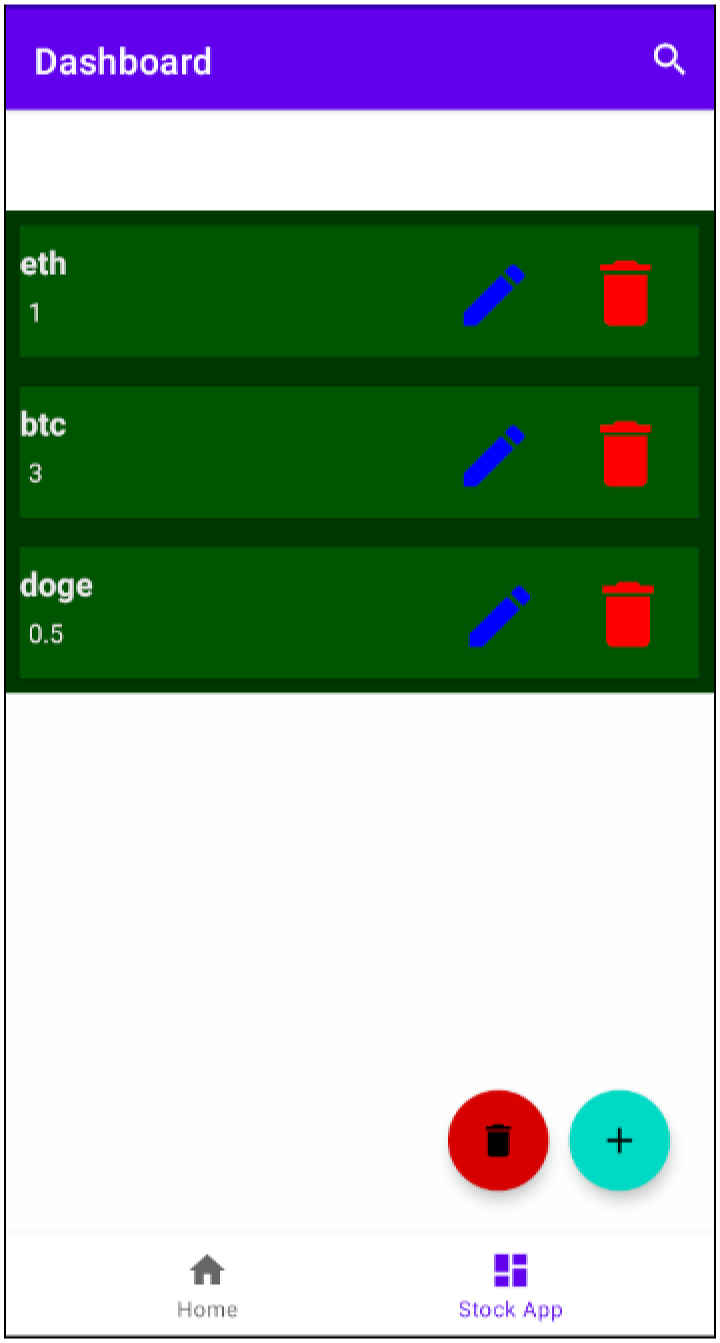


Now if I click the delete all button (button right next to the add button) the app will look like

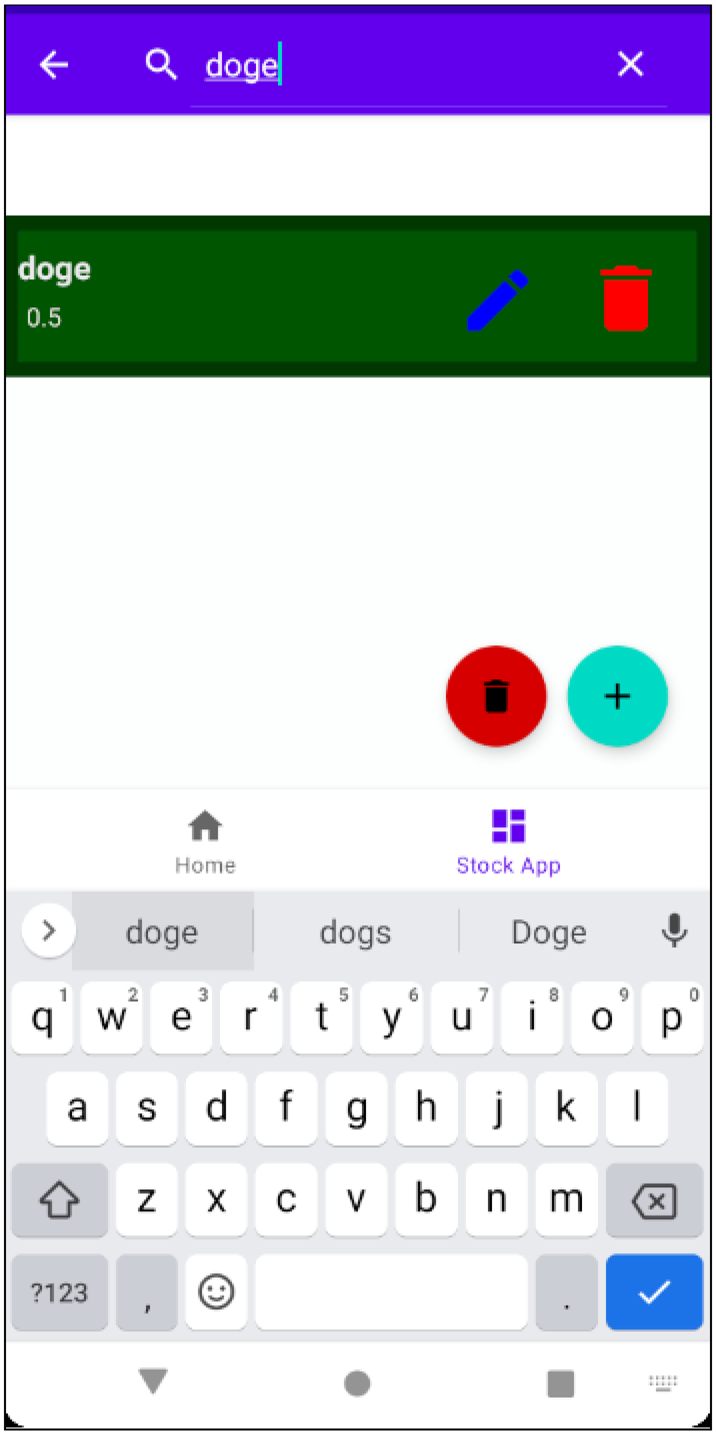
this:



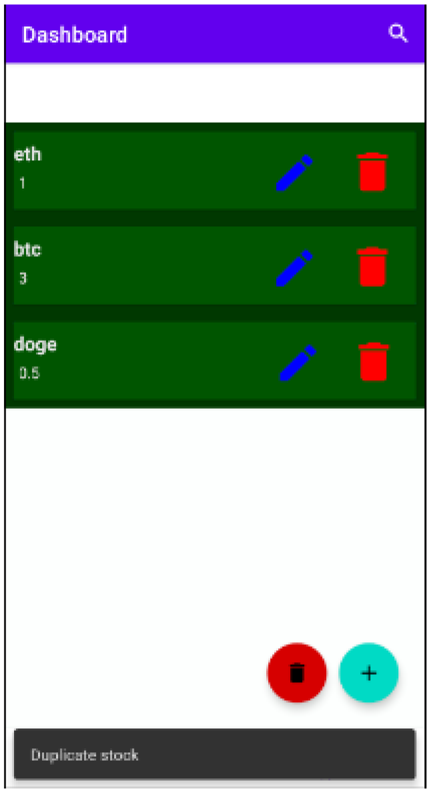
Now I am going to demonstrate the search functionality. I added more stocks and the app before search looks like this:



This is what happens when I type “doge” in the search bar (the app filters the recyclerview based on the search query) **(Note: you will have to go back to the Home Fragment and then go to Stock App again to see all the existing stocks)**:



If I try to add "doge" again then the app will give me an error:



Application Link:

https://github.com/sadiksuny/Lab6