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

GitHub Username: benktesh

Benktesh Sharma

Description

SmartStock brings real time US Stock Exchange data. The market snapshots shows daily movement of major market indices covering New York Stock Exchange (NYSE), Nasdaq, and American Stock Exchange (AMEX). It keeps track of your watchlist and notifies you of any

thresholds and feeds sources of news from around the world related to the stock in your portfolio which can be used in trading decisions. Some salient feature of this app are listed below:

- Shows Major market indices real time.
- Provide Quotes, Charts, and News.
- Show the detailed stock information: open price, last price, change, change percent, volume, day low, day high, exchange time,etc.
- Ability to search and add the stocks to portfolio

The application will be developed using Java programming language on latest version of Android Studio as primary development environment. The development will make use of latest version of Gradle and dependent libraries. The following lists provide some high level technical design specification:

- App is will be written solely in the Java programming language
- App keeps all strings in a strings.xml file and enables RTL layout switching on all layouts
- In order to provide support for accessibility, the app will implement following best practices in the app:
    - Using ContentDescription
    - Using appropriate labels
    - Uses best practices to provide ease of navigation
    - Uses large touch targets
    - Uses adequate color contrast
- App provides a widget to provide relevant information for the last stock that user visited on the home screen
- The app performs a short short duration, on-demand requests to pull data from external web apis using an AsyncTask.

Table 1 shows some specific libraries, tools and their version if applicable that will potentially be used. Attempt will be made to use the latest version, however in some cases a non latest version may also be used as effort will not be put to update library tools to use the latest version if desired functionality is supported by non-latest stable release version.

Table 1

SN	Description of Tool/Library and version
1	Java programming language, Version 8 or higher
2	SQLite Database

3	Firebase
4	Gradle for building
5	AnyChart (https://github.com/AnyChart/AnyChart-Android/wiki/Getting-started) or Graphview (http://www.android-graphview.org) to render charts
6	Picasso. Used in rendering images
7	Espresso for unit testing

Intended User

The app is for people who like to stay current with stock market.

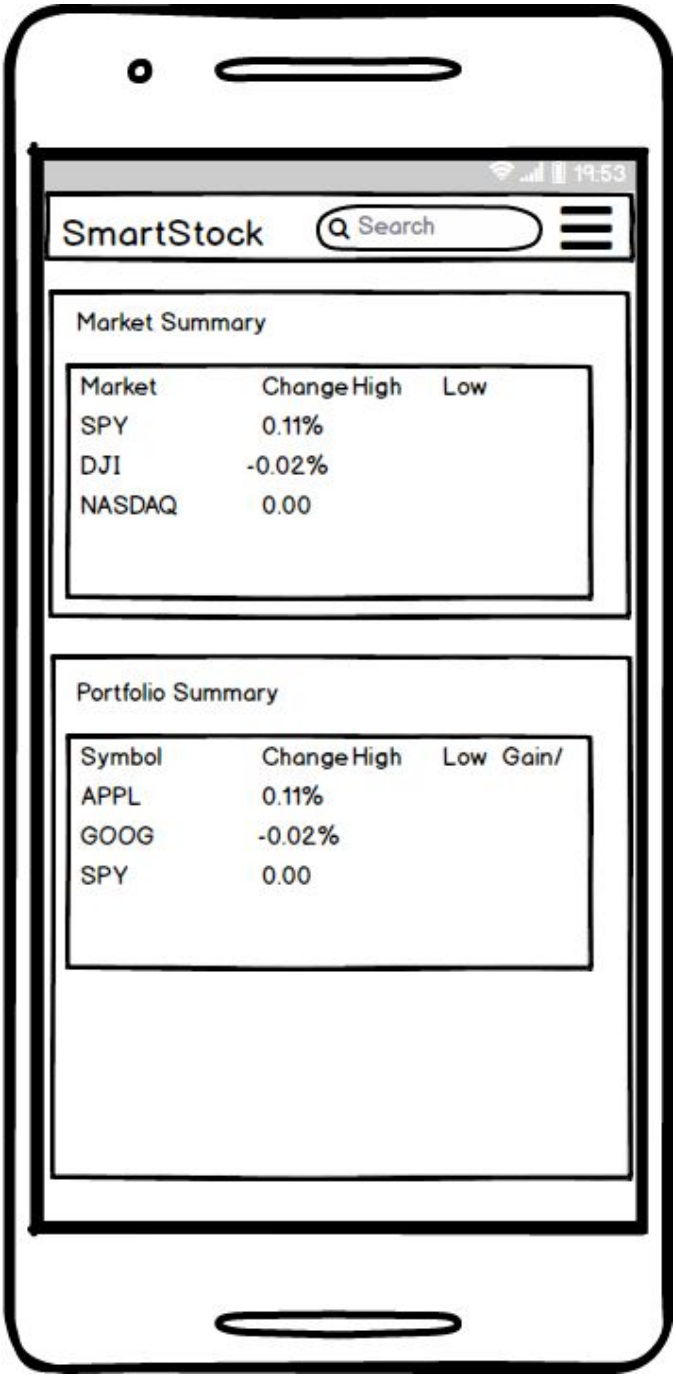
Features

List the main features of your app. For example:

- Save, edit, delete stock in the portfolio.
- Add thresholds and allow notification when stock crosses the thresholds.
- Ability to create custom chart for the stock.
- Ability to view the detailed information of last stock quotes in the widget

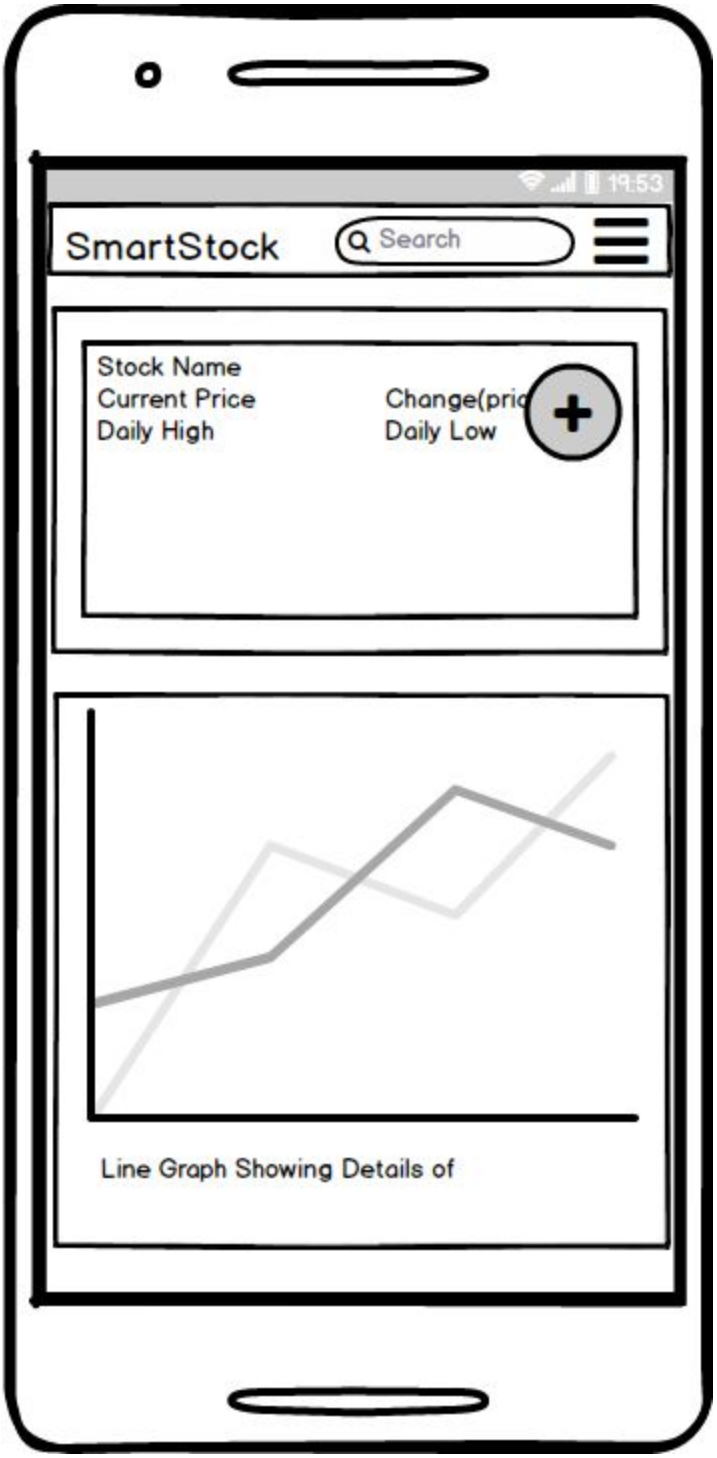
User Interface Mocks

Screen 1



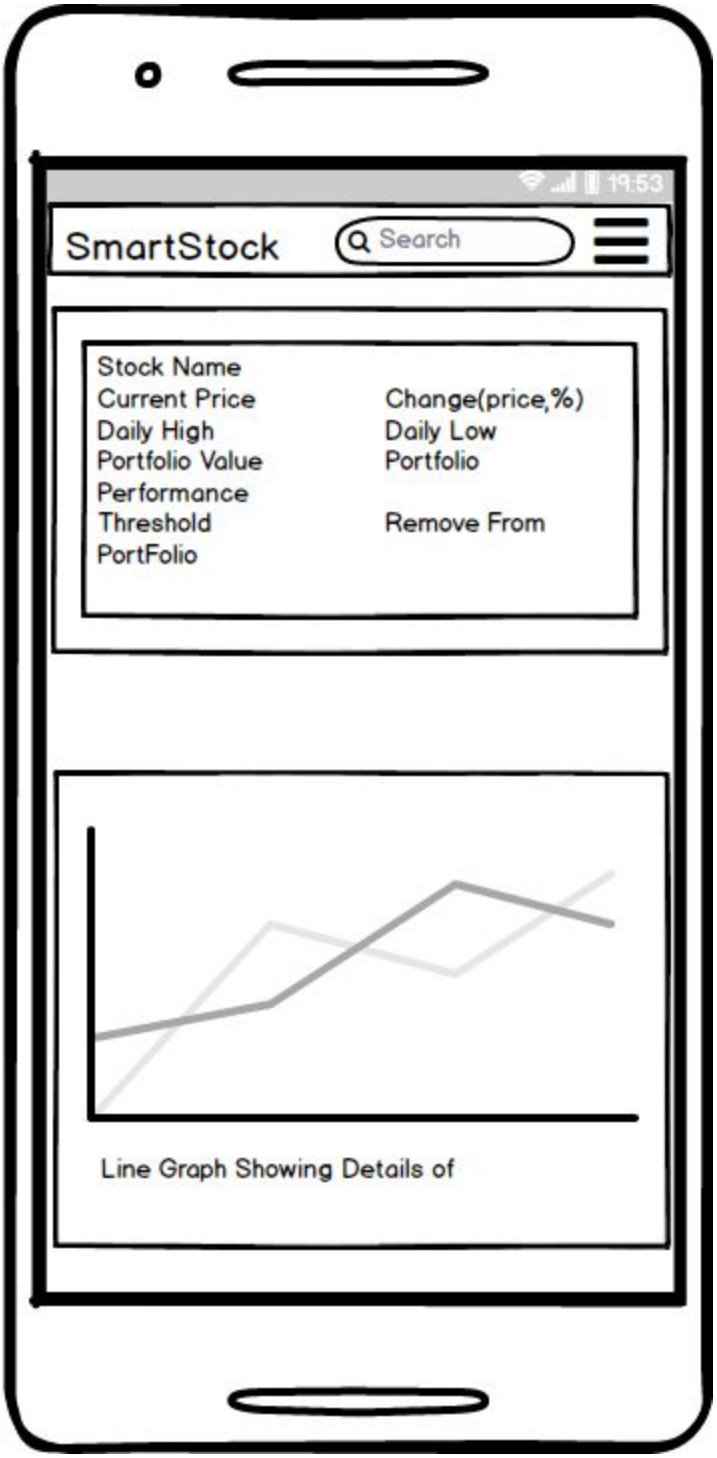
This is the main screen for SmartStock. In the main screen, summary of market as well as summary of portfolio will be shown. The fields shown in the mock-ups may change to make the UI better serve the user. Clicking any of the items, brings the detailed page.

Screen 2



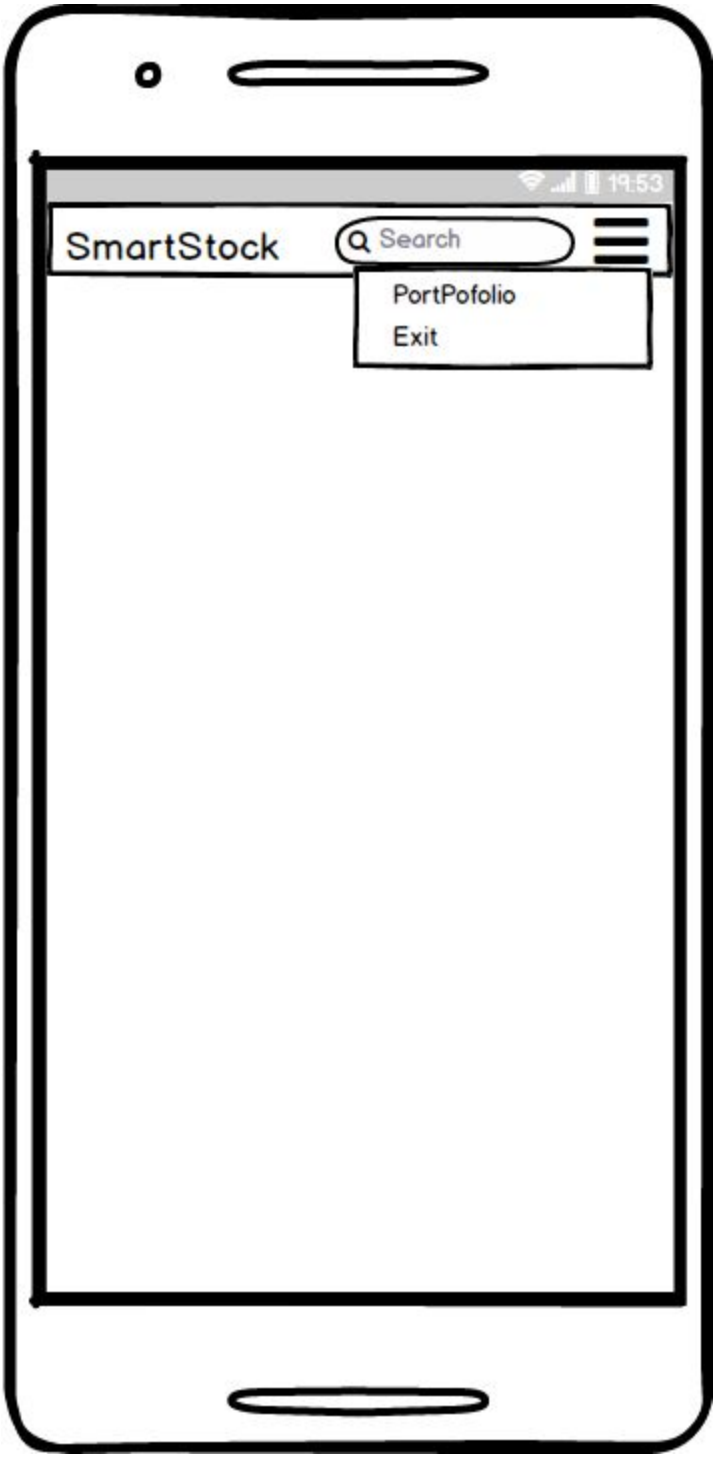
The stock details screen shows the details of the screen. If the stock is not in the user's portfolio the button '+' will show up. Pressing the button bring up the view where user can update the portfolio (this is shown later).

Screen 3



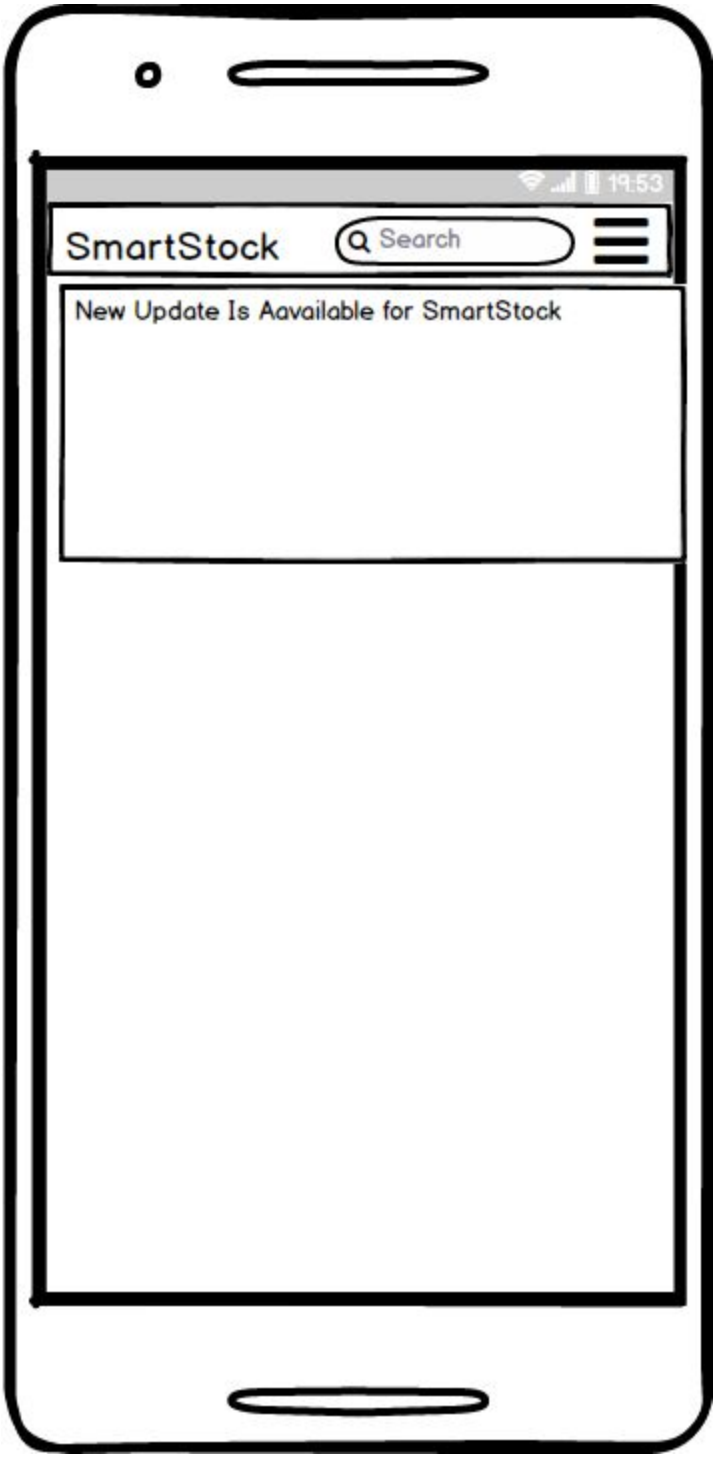
When a symbol is present in the user's portfolio, the add button will not appear, but portfolio values will indicate the value of the portfolio.

Screen 4



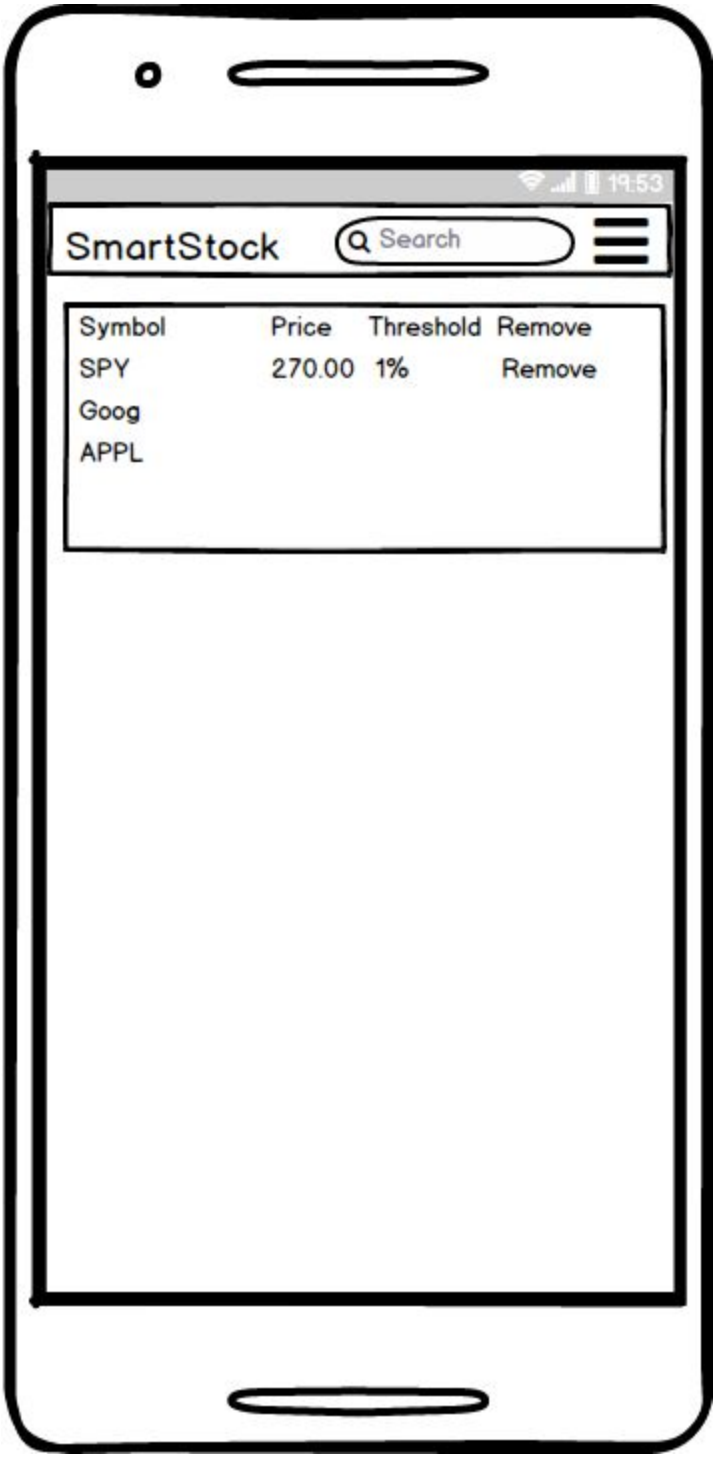
The menu bar contains navigation link to portfortpolio or exiting an app.

Screen 5



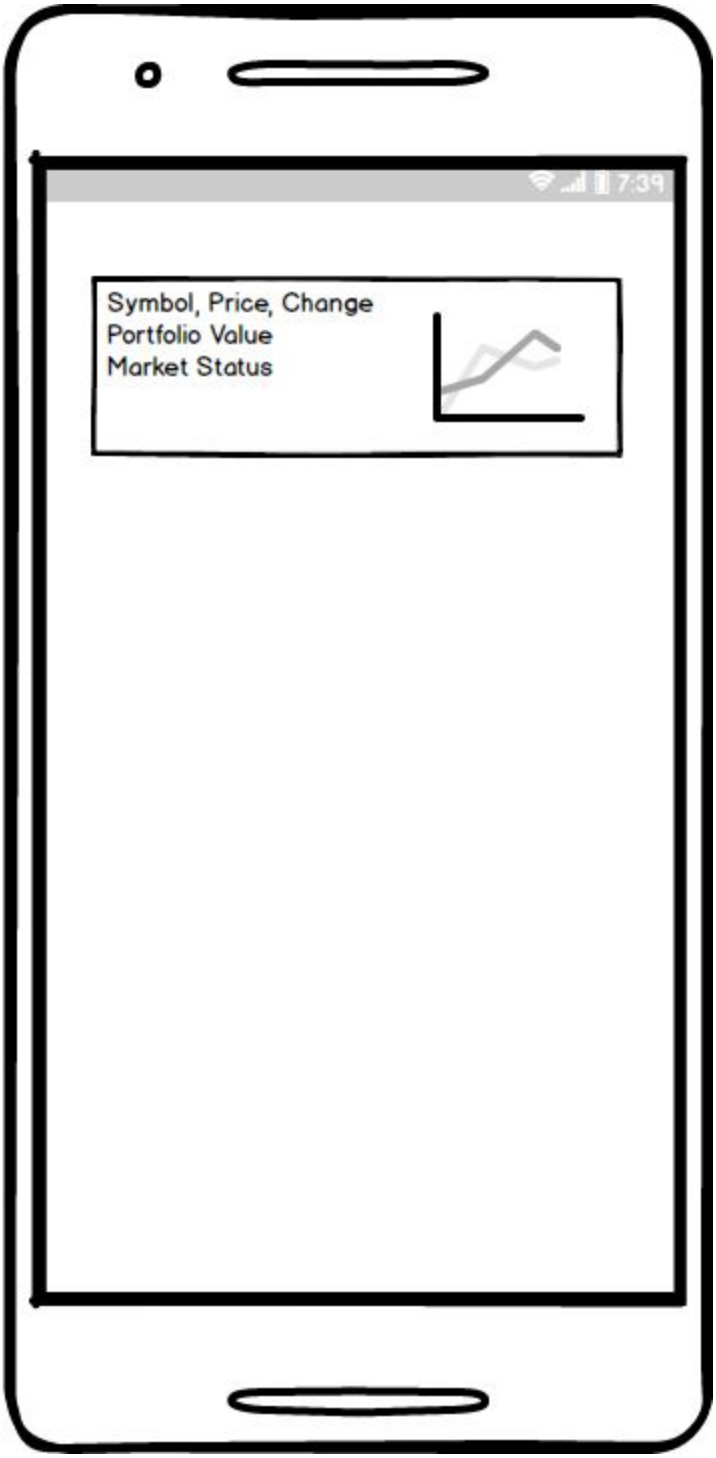
The Firebase server can send message to the user of the system. The message will appear in the main screen.

Screen 6



The portfolio view can be accessed either by menu or adding a new symbol from detailed page. In this page, threshold can be set and quantity can be specified.

Screen 7



The widget screen as shown above shows the brief summary of last visited stock. Clicking on the widget will open the detailed activity page.

Key Considerations

How will your app handle data persistence?

Application will use content provider to maintain data locally. Firebase will be used for push notification.

Describe any edge or corner cases in the UX.

When dependent data cannot be retrieved from the external API, an error message will be displayed.

If chart rendering is interrupted, then a blank placeholder image will be shown.

Any other fatal errors will make the app close.

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

Firebase will be used for push notification such as market news or some new features in the app.
Picasso will be used to render images. This is specially used in Detailed page.
AnyChart (https://github.com/AnyChart/AnyChart-Android/wiki/Getting-started) or Graphview (http://www.android-graphview.org) will be used to render graphs/charts. The plan is to explore both the libraries and evaluate based on performance, ease in use and dependencies and final appearance.

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

Firebase will be used for crash analytics and push notification.

Next Steps: Required Tasks

Task 1: Project Setup

Under this task, following subtasks will be implemented:

- Create repository
- Create default project
- Create firebase project

- Configure libraries
- Configure menu
- Configure Database/persistence

Task 2: Implement UI for Each Activity and Fragment

Under this task, following subtasks will be implemented:

- Build UI for MainActivity
- Build/refine database
- Add Firebase related logic
- Build UI for detailed activity
- Build UI for portfolio view and edit
- Add unit test as much as possible

Task 3: Implement App Widget

Following subtasks will be implemented:

- Implement app widget

Task 4: Testing App Accessibility and refinement

Following subtasks will be implemented:

- Test accessibility
- Any refinement
- Prepare for release (icon etc)

Task 5: Create variants and Integrate with Google Play Service

Following subtasks will be implemented:

- Create Build Variant
- Create Screen size variant for layout
- Implement Google Play Service

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**"