GitHub Username: **alex01001**

Zone Trading Alerts

Description

- I'll use Java as the programming language.
- Android Studio 3.1.3
- Build #AI-173.4819257, built on June 4, 2018
- JRE: 1.8.0_152-release-1024-b02 amd64
- JVM: OpenJDK 64-Bit Server VM by JetBrains s.r.o
- Windows 10 10.0
- Gradle version 4.4
- TradingView Charting Library v. 1.14
- ButterKnife v. 8.8.1

Project background:
Some time ago I've created a system which monitors the stock market and sends alerts when it identifies a moment when stock price is about to start growing (you can read more at http://stocksbuyalerts.com/). The system is based on the theory of so-called supply and demand zones. My software recognizes demand zones, assesses the probability of the price growth, and notifies the subscribers. The target audience for these alerts is day traders (people who actively buy and sell stocks within a day), so it is very important that subscribers receive alerts quickly. Currently, the system sends e-mail alerts only. Sometimes, e-mail delivery is delayed by e-mail servers and my subscribers receive alerts too late. That's why I've decided to build an app!

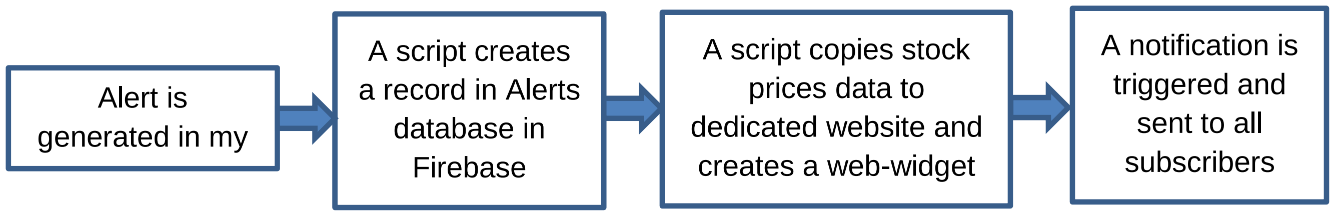
Intended User

The target audience for these alerts is day traders (people who actively buy and sell stocks within a day).

Features

- Receive the instant alerts (notification) to all subscribed users. Alerts will be generated in Firebase.
- Display the chart and key parameters for the stock for which alter is sent.
- Show historical charts for previous days (one month back).
- Show price for a selected stock in a form of a **widget.**
- Enable a user to share successful trades in his/her social network (optional[1])
- View news/tweets about a selected company (optional)

Process Flow:
The sequences of events when alert is generated:


Alert is generated in my
A script creates a record in Alerts database in Firebase
A script copies stock prices data to dedicated website and creates a web-widget
A notification is triggered and sent to all subscribers

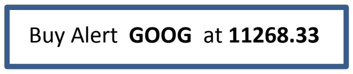
When a user clicks on the alert, the app should make a query to the Firebase's database and copy all data related to the Alert (and save it in the app's local database). This

[1] I'll take care of it if I have time

information is static and won't be changed. A user will see the charts as web-widgets through WebView.

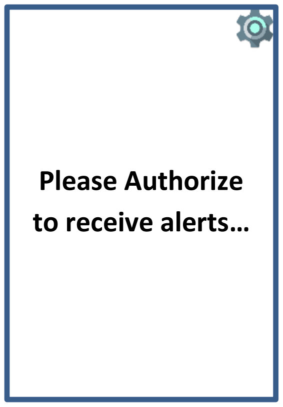
User Interface Mocks

1. Notification on home Screen

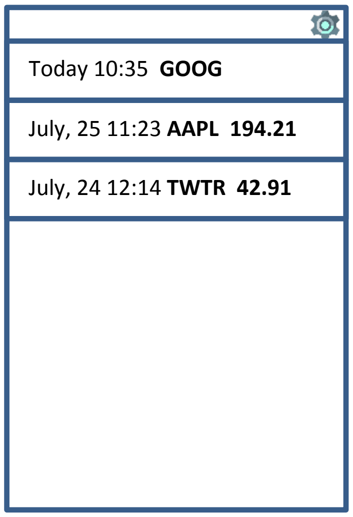


Buy Alert **GOOG** at **11268.33**

2. Main Activity
   2a. User is not authorized. 2b. User is authorized.



Please Authorize to receive alerts…

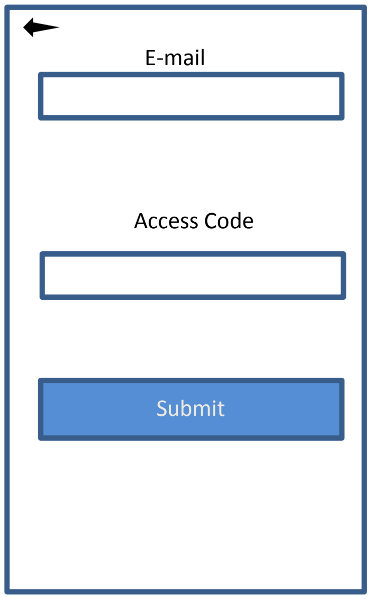


Today 10:35 **GOOG**

July, 25 11:23 **AAPL 194.21**

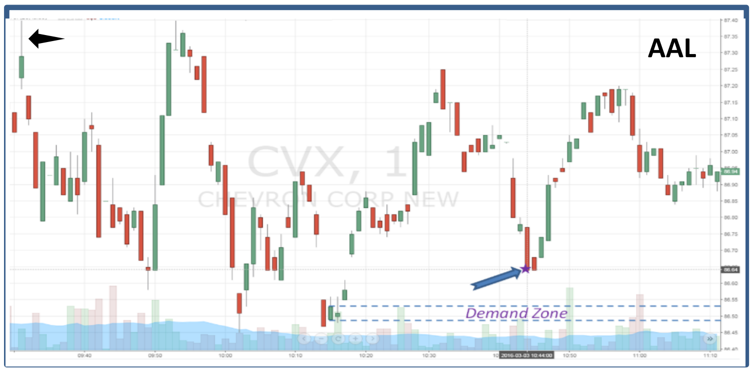
July, 24 12:14 **TWTR 42.91**

3. User's authorization:

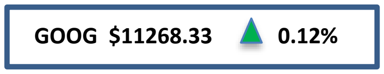


4. Stock chart view (always in landscape).

Web-widget displayed through WebView.



5. Widget


GOOG $11268.33 0.12%

Key Considerations

How will your app handle data persistence?

I plan to use a firebase database.

Database structure:
The database will include one table: Alerts.
Table structure:
Alerts:

- Symbol – String
- AlertTime – Timestamp
- ZoneTime – Timestamp
- ZoneTop - decimal (8,4)
- ZoneBottom - decimal (8,4)
- AlertPrice - decimal (8,4)

Describe any edge or corner cases in the UX.

The user will navigate back by clicking the “back button” on the toolbar.
Several controls will need to be implemented:

- The app should inform the user if the device is offline and updates are not possible
- The app should check if a user entered a valid stock symbol for widget updates.

I'll utilize darker color theme for the app.
The app will support **accessibility** by using SP for text sizes.
I'll try to implement RTL, but not sure my charting library will support it.

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

I'll use ButterKnife for view binding.

I plan to use TradingView library to plot charts (https://www.tradingview.com/HTML5-stock-forex-bitcoin-charting-library/).

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

I'll use Firebase for authentication, notifications, and database access.

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

- Create the project in Android Studio.
- Configure the stock charting library and experiment with it.
- Configure firebase account.

Task 2: Set up interaction between my system and Firebase.

- Build a script to copy data from my database to the Firebase database
- Build a notification-triggering script.

I'll use Python for this task.

Task 3: Apply ButterKnife library.

- Add dependency in *build.gradle*.
- Apply plugin in *build.gradle*.
- Use ButterKnife to bind views.

Task 4: Implement UI for Each Activity and Fragment

- Implement Main Activity. I'll use **recycler view** to display alerts.
- Implement User Authorization Activity.
- Implement and test the notification process.
- Implement the charting library (including reading data from the Firebase)

I'll use **AsyncTasks** and/or **Loaders** for price updates.

Task 5: Implement widget

Widget will show current price for a selected stock. The prices will be taken Google Finance web service.

- Design the layout
- Create a class which will pull data from Google Finance and update it in the layout

Task 6: Test

- Test the notifications delivery
- Test charting
- Check data accuracy

Task 7 (optional): Implement Chart Sharing

Implement functionality to share stock charts on users' social networks.