Stock market Trading simulator

Joshua Tomsett

Contents

[**Analysis** 2](#_Toc84775605)

[Problem Identification 2](#_Toc84775606)

[Introduction 2](#_Toc84775607)

[Stakeholders 2](#_Toc84775608)

[Why a Computational Solution? 2](#_Toc84775609)

[Research 4](#_Toc84775610)

[Existing solutions 4](#_Toc84775611)

[Research of features 7](#_Toc84775612)

[Features 8](#_Toc84775613)

[Essential Features 8](#_Toc84775614)

[Limitations 9](#_Toc84775615)

[Requirements 10](#_Toc84775616)

[Hardware 10](#_Toc84775617)

[Software 10](#_Toc84775618)

[Success Criteria 11](#_Toc84775619)

# **Analysis**

## Problem Identification

### Introduction

I plan to produce a program capable of allowing users to trade virtual stocks and shares in a risk-free environment. The stocks and shares being traded are real companies and use real prices. The prices and historical data, used to create the graphs, will be retrieved from the Yahoo finance API and I will use matplotlib to visualize the data. All user and stock data will be stored in a database which I will access using sqlite3.

### Stakeholders

The stakeholders interested in my project would be people wanting to learn how to trade on the stock market, or to improve at it. They could be of any age group, but most likely 16+. My project allows user to train in a risk-free environment, as they would be trading with fake currency, so their personal funds are not at risk.

Another stakeholder could be a GCSE or A level business teacher wanting to teach their students about the stock market. This would be ideal for them as it is free and simple to use, while still being accurate.

The criteria of my stakeholders would a program that allows them to buy and sell stocks and shares using real time prices. Because of this, I will use the Yahoo finance API to get the data and a user interface to allow the users to interact this the program.

### Why a Computational Solution?

My project suits a computational approach as attempting to practise trading on the stock market without a computer would be very tedious. A computational approach allows data to be easily accessed and viewed through graphs. It also allows for easy tracking of a user’s current positions and fault proof algorithms to buy and sell stocks, as no mathematical errors will take place by a computer whereas if done by people calculation errors are inevitable. Using a computational approach also allows for the use of a database. This means that all data can be stored easily and allows for multiple users to use the program.

There are other benefits to this solution regarding specific methods of programming. The method of **decomposition** is useful in this instance as I can plan components of my code I want to solve individually. In my case, my main steps are:

1. Retrieving data from the API and creating graphs with matplotlib
2. The algorithms for buying and selling shares
3. The database that stores the user’s information
4. Kivy GUI to displaying the user interface for users to interreact with

This will allow for me to focus on each step at a time when making my program and allows for easy testing of each component. The first version of my code will contain the first and second steps of this decomposition. The second version will integrate multiple users through a database as described in step 3. The third and final version will incorporate a Kivy GUI shown in step 4.

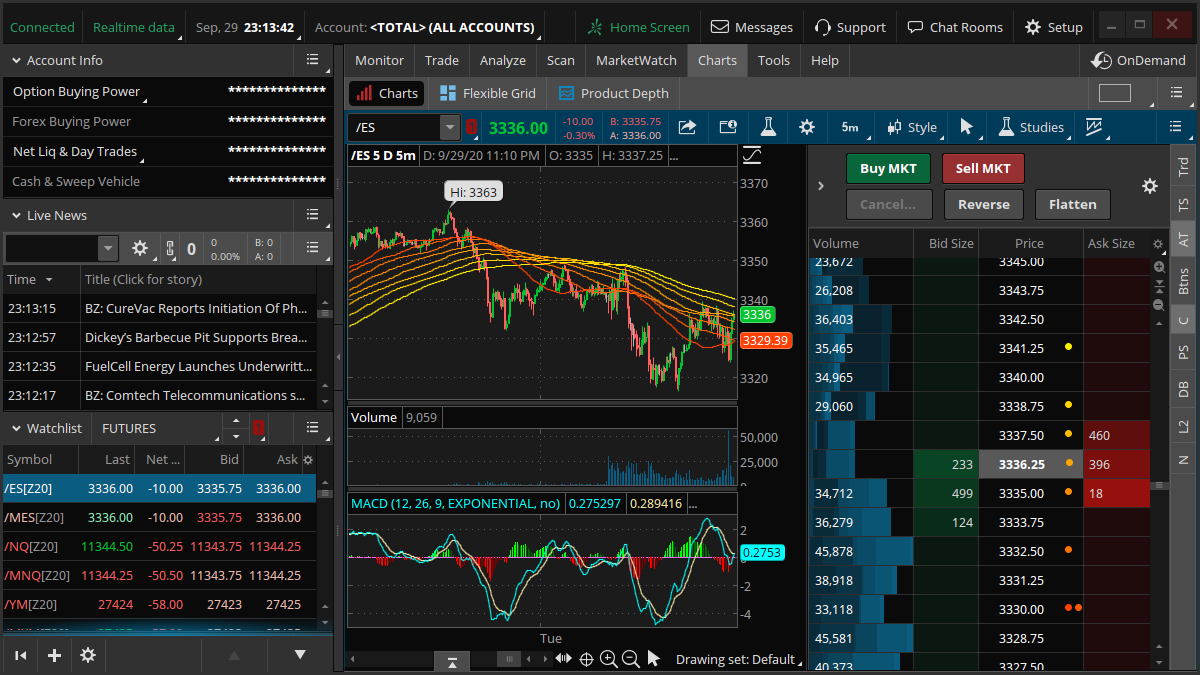
Another method I can make use of is **abstraction** as it allows me to reduce the workload for the project and focus on my main steps to produce a functional virtual stock trader. Some examples of my potential abstraction are as follows:

* Graphs are not interactable as matplotlib does not allow for it
* Graphs can only display data for a week, month, or year. This is because to generate a graph you need to retrieve the data and then plot it. So, adding more date options would take much longer
* Limited information about user’s current stock positions. Adding more information means the database will get larger much faster therefore taking up more space
* Not able to find a ticker without first adding it to the database. The database is stored on the user’s local device so having every ticker in it would take up a large amount of space, also the program displays all the tickers as buttons so having them all in the database would mean slower performance.

## Research

### Existing solutions

**Thinkorswim**

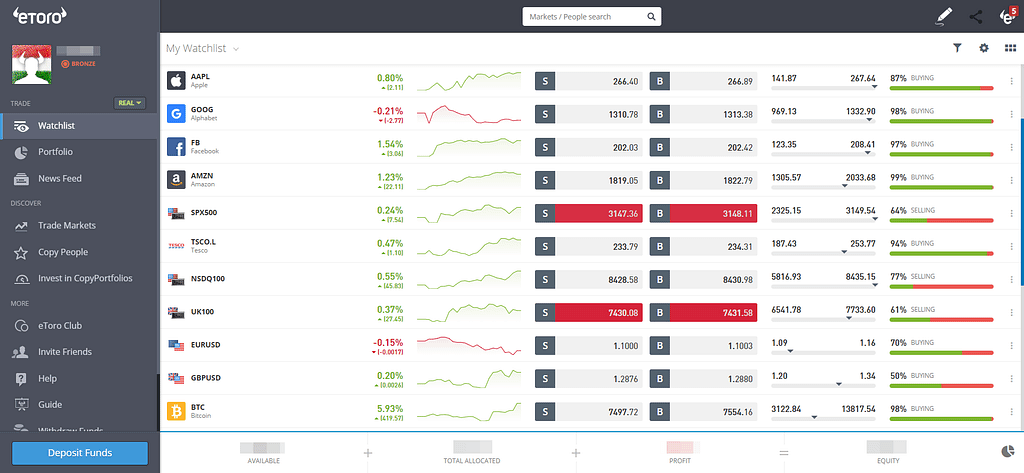
Thinkorswim is a professional-level trading platform made by TD Ameritrade for trading. It gives users an enormous number of tools as well as coming in web, desktop, and mobile form.

Source : <https://www.tdameritrade.com/tools-and-platforms.html>

|  |  |
| --- | --- |
| **Features** | **Essential Feature Y/N** |
| Graph to display data | **Yes**, as the user needs some way of seeing how the stock is doing. |
| Ability to have multiple graphs | **No**, as it will take time to create and display the one graph so adding multiple would cause the program to be very slow. |
| Buy and Sell buttons | **Yes**, as it is essential for the users to use the program. |
| Live news | **No**, as I have no way of getting the news into the program. |
| Analysis tools | **No**, as I am using matplotlib and they do not offer those tools. |
| Portfolio screen | **Yes**, as users must be able to see and manage the stocks they currently hold. |
| Multiple platforms | **No**, I will only be making a desktop app version as it will be made in python so converting to a web app would be very difficult and a mobile version would involve having to redesign the layout and extra optimization. |

**eToro**

EToro is a very popular trading platform with a modern design and a lot of useful tools. Such as the CopyTrader™ system which allows users to copy popular investors.



Source : <https://www.etoro.com/trading/platforms/>

|  |  |
| --- | --- |
| **Features** | **Essential Feature Y/N** |
| Graphs to display data | **Yes**, as the user needs some way of seeing how the stock is doing. |
| Buy and Sell buttons | **Yes**, as it is essential for the users to use the program. |
| News Feed | **No**, as I have no way of getting the news into the program. |
| Portfolio | **Yes**, as users must be able to see and manage the stocks they currently hold. |
| Social system | **No**, as the program will only run client side there is no network so no way to connect users to each other. |
| CopyTrader™ | **No**, as there would be no one else using the app so no one to copy. |
| Demo account | **Yes**, the purpose of my program is to act as a demo account by using virtual funds. |

**Trading 212**

Trading 212 is the UK’s largest trading platform with over 15 million users. They offer a modern trading app and allows users to have either a invest, ISA, or CFD account.

Source : <https://www.trading212.com/>

|  |  |
| --- | --- |
| **Features** | **Essential Feature Y/N** |
| Graphs to display data | **Yes**, as the user needs some way of seeing how the stock is doing. |
| Buy and Sell buttons | **Yes**, as it is essential for the users to use the program. |
| Portfolio | **Yes**, as users must be able to see and manage the stocks they currently hold. |
| Demo account | **Yes**, the purpose of my program is to act as a demo account by using virtual funds. |
| Different account types | **No**, as the user is using virtual funds different account types are not needed. |
| Auto invest | **No**, as the purpose of the program is for users to decide which stocks to buy so an auto invest system would defeat the purpose of my project. |
| Multiple platforms | **No**, I will only be making a desktop app version as it will be made in python so converting to a web app would be very difficult and a mobile version would involve having to redesign the layout and extra optimization. |

### Research of features

**Kivy / KivyMD**

Kivy is an open-source python library for applications that make use of user interfaces. It is not very popular, so I had to learn most of it from the [Kivy documentation](https://kivy.org/doc/stable/). Upon researching Kivy I stumbled across KivyMD, an extension to Kivy where the widgets are designed after [Googles Material Design](https://material.io/). I decided to primary use KivyMD as in general it just looks better. KivyMD is also not very popular, so I learnt almost all of it from the [KivyMD documentation](https://kivymd.readthedocs.io/en/latest/)



The graphs used in the program are generated using matplotlib, in their original form they cannot be displayed in kivy. While researching I found a kivy add on called [kivy garden](https://kivy.org/doc/stable/api-kivy.garden.html). This add on has a widget called FigureCanvasKivyAgg, which basically allows the graph to be drawn onto a canvas on the program.

**Hashing**

For the user’s security their passwords must be encrypted before being stored in the database. To do this I needed to find a way to hash the password, python has a built-in library for hashing called [hashlib](https://docs.python.org/3/library/hashlib.html) so I decided to use that as it allows for hashing in a single line of code.

**Yahoo finance API**

From the beginning of my project, I knew I would need some method to get data into the program, after looking at all the options I decided to use the [Yahoo Finance](https://uk.finance.yahoo.com/) API. It allows me to get current and historical data so users will know the current price as well as have the historical data plotted on a graph within the program. The Yahoo Finance API is also free to use. To access the API, I need to use [Pandas-datareader](https://pandas-datareader.readthedocs.io/en/latest/) and yfinance.

## Features

### Essential Features

|  |  |  |
| --- | --- | --- |
|  | Feature | Explanation |
| 1. | Graph to track the price of the share against time | The main purpose of the program is for users to analyse data and then decide on whether to buy the stock or not, because of this it is essential for users to be able to see the data in a more visual way. A graph works well as it allows users to quickly see highs and lows. |
| 2. | Buy and sell buttons | For users to buy and sell they obviously need some way of telling the computer they want to do so. The best way to do this is through buttons as it only needs a single mouse click. |
| 3. | Text input for the amount a user’s wants to buy or sell | Without this user’s would only be able to buy or sell a single share at a time, which in almost all cases would be very inconvenient. |
| 4. | Text with information about the currently selected stock. | When a user selects a stock to look at, they should be presented with some information about the stock, such as the current market price. This will be displayed next to the buy and sell buttons for users to see easily. |
| 5. | Scrollable list of buttons for each stock  There may need to be a limit at 3000 tickers | For the users to select a stock I decide to use buttons in a scrollable list, is must be scrollable as the number of stocks will grow over time. |
| 6. | Button to suggest a stock to be added to the list of stocks | It is essential for users to be able to add new stocks, otherwise you could not buy anything you have not bought before. |
| 7. | Search bar for the list of stocks | Once the list gets bigger it will become harder to find the stock you are looking for so a search bar will allow users to find any stock quickly. |
| 8. | Tabs at the top of the screen | Essential for users to switch between the different screens (main screen, portfolio screen, and settings screen). |
| 9. | Text to display the user’s current balance | The users will need to know what their current balance is so they can decide how much they want to spend in the future. |
| 10. | The portfolio | For the users to see what stocks they own there must be a portfolio section. I did this by having a scrollable list with custom widgets that display the stock ticker, current market price, the total units the user has, and the total value of all the units they own. |
| 11. | The settings | In the settings screen it is essential for users to have a way to sign out of their account. This could be because they want to login to a different account or just want to log out for security. |
| 12. | Login and sign-up screen | For users to be able to login to their account they will of course need a way to input their username and password, there must also be a button to sign up as otherwise no one would be able to create an account |

### Limitations

|  |  |  |
| --- | --- | --- |
|  | Feature | Explanation |
| 1. | Interactable graphs and analysis tools | I will be using matplotlib to create the graphs and then a canvas widget to draw the graph as an image, because of this I am unable to have the graph be interactable. I am also not able to integrate any analysis tools as they would take too long and be too difficult to make. |
| 2. | Multiple graphs | It would be possible to have multiple graphs, but due to the time is takes to retrieve the data and then plot said data onto a graph means having more than one will just take too long. It is also not needed for a simple simulation. |
| 3. | News feed | I have not found any method to retrieve this kind of data so I of course cannot include this feature. It is also not needed for functionality, so maybe not even wanted by stakeholders. |
| 4. | Can only see graphs for week, month, or year | As I said previously making a graph takes time and the program works by generating graphs for the week, month, and the year at the same time so adding more time options would mean the program taking longer to load after a ticker button is pressed. |
| 5. | Users must manually add stocks to the program | I could possibly be able to get every ticker into the database, but if a ticker expires or when a new ticker is created the database won’t be updated so over time it will become obsolete. Also, the database will take up much more space with every ticker in it, the program also displaying each ticker in the database as a button so the user can select one, so having every ticker would mean the program takes far too long to start up. |

## Requirements

### Hardware

**A computer**: A computer is required to run the program, almost any computer will suffice but it would be recommended to have plenty of spare storage and memory.

**peripherals**: Just a mouse, keyboard, and a monitor are required to interact with and see the program.

### Software

**Windows 10 operating system**: This is one of the standard operating systems. I have decided to use this over the others as I am most familiar with it, and it is the most available operating system to me. Python supports it.

**Python interpreter**: The program is written in Python 3.9, therefore a Python interpreter is required to read and run the code.

**Python Modules**: The program will have a lot of dependencies, all of which are listed below:

* Pandas-datareader – used for reading data from the yahoo finance API

<https://pandas.pydata.org/>

* Matplotlib – used to create graphs

<https://matplotlib.org/>

* Kivy – A user interface library

<https://kivy.org/#home>

* kivyMD – A collection of Material Design compliant widgets for use with Kivy

<https://kivymd.readthedocs.io/en/latest/>

* sqlite3 – Library for accessing databases

<https://docs.python.org/3/library/sqlite3.html>

* hashlib – Library consisting of many hashing algorithms

<https://docs.python.org/3/library/hashlib.html>

* yfinance – Library for interacting with the yahoo finance API

<https://pypi.org/project/yfinance/>

## Success Criteria

|  |  |  |
| --- | --- | --- |
|  | Criteria | Explanation |
| 1.1 | Login – username and password | Is the user asked to enter a username and password upon launching the program? |
| 1.2 | Login – database verification | Once the user has entered a username and password, are these compared to the database to see if they are valid inputs? |
| 1.3 | Login – completed login | Once a login has been successfully verified, is the user sent to the main menu?  Is the user’s data loaded into the program from the database? |
| 2.1 | Signup – username, password, and re-password | Is the user able to input a username, password, and Re-Password? |
| 2.2 | Signup – database verification | Once the user has entered a username, password, and re-password are these successfully validated? |
| 2.3 | Signup – completed login | If valid, are the entered username and password stored in the database?  Is the user sent back to the login screen? |
| 3.1 | Ticker buttons – displaying the buttons | Are the tickers stored in the database taken and displayed as buttons within the program? |
| 3.2 | Ticker buttons – search field | Is there a search field so that users can easily find the ticker they want? |
| 3.3 | Ticker buttons – suggest button | Is there a button that allows users to add tickers to the database?  Once successfully adding a ticker, is it displayed in the program? |
| 4.1 | Graph – the graph | When a ticker button is pressed, is a graph displayed plotting the stock’s price against time?  Is the default time range a month? |
| 4.2 | Graph – time buttons | Are there buttons for the user to change the time range of the graph? |
| 5.1 | Buying and selling - buttons | Is there a buy button and a sell button?  Is this buy button green and the sell button red? |
| 5.2 | Buying and selling – amount input | Is there a text input field for the amount? |
| 5.3 | Buying and selling – verification | Does the program know what the current ticker is?  For selling, does the user own the ticker and do they have enough to sell?  For buying, does the user have enough to afford the amount they have entered?  Is the user ask to approve the transaction? |
| 5.4 | Buying and selling - database | Does the user entry table get updated for the user’s new portfolio?  Does the users table get updated for the users’ new balance? |
| 6.1 | Portfolio - balance | Is the users balance retrieved from the database and displayed to the user? |
| 6.2 | Portfolio – stock positions | For each stock the user owns is the ticker, units owned, current price, and total value displayed? |