Software Requirement Specification (SRS)

Virtual Stock Market Application

Team 9: Christopher McCracken, Darren Gabrido, Louis Brian Santos, Tai Martinez

Table of Contents

1. <u>Introduction</u>:

The virtual stock market application is a virtual investment portfolio which simulates an environment where users can practice and learn how to participate in the U.S. Stock Market. This application is not connected to any real money trading nor does it allow users to trade actual money but rather allows the user to mimic investing without any actual stake. The purpose of this application is to help the user learn the dynamics of the U.S. Stock Market and familiarize themselves with how trading is completed.

2. <u>Current system</u>

There is a current system in the form of an online multiplayer game located on https://www.marketwatch.com/game/[1] that simulates trading and selling in the U.S. stock market without the usage of actual money. Upon arrival to the website, the user must login and join a game with other users. Then the user can view the "Overview", "Portfolio", "Rankings", "Discuss" and "Settings" pages. The Overview page is where the user views their overall profile with money statistics, where they can search for certain companies, and the game statistics. When a user searches for a certain company, before trading the user can view the stock chart for that company as well as viewing an order summary with the price, shares, and time as well as whether it is a buy or sell of the stock. The Portfolio page is where users are able to view their transactions and performance throughout the game. The Rankings page is simply where the user views the game rankings compared to other users. The Discuss page is an active chat where users can open discussions and carry on conversations. The Settings page is where the user can adjust the difficulty and experience of the game.

3. <u>Proposed system</u>

3.1 To-Be Solution:

The solution is a functioning virtual stock market desktop application that uses the U.S. stock market API to implement real time online stock data to track gains and losses per stock of companies. This application will act as an investment portfolio without actual money tied to it to help users learn the dynamics of the U.S. Stock Market. The application will allow users to create an account that they sign into so they can continually trade stock from a personal portfolio. Users can create multiple portfolios on a single account to help understand what certain trading factors will affect their money in certain ways. In the portfolio, users are able to buy and/or sell stock from different companies after viewing the statistics from that company. They will then view a confirmation page for that bought or sold stock. Users can then see their net gain or loss per stock and the net gain or loss per portfolio to see how their money is changing after they participate in stock market trading.

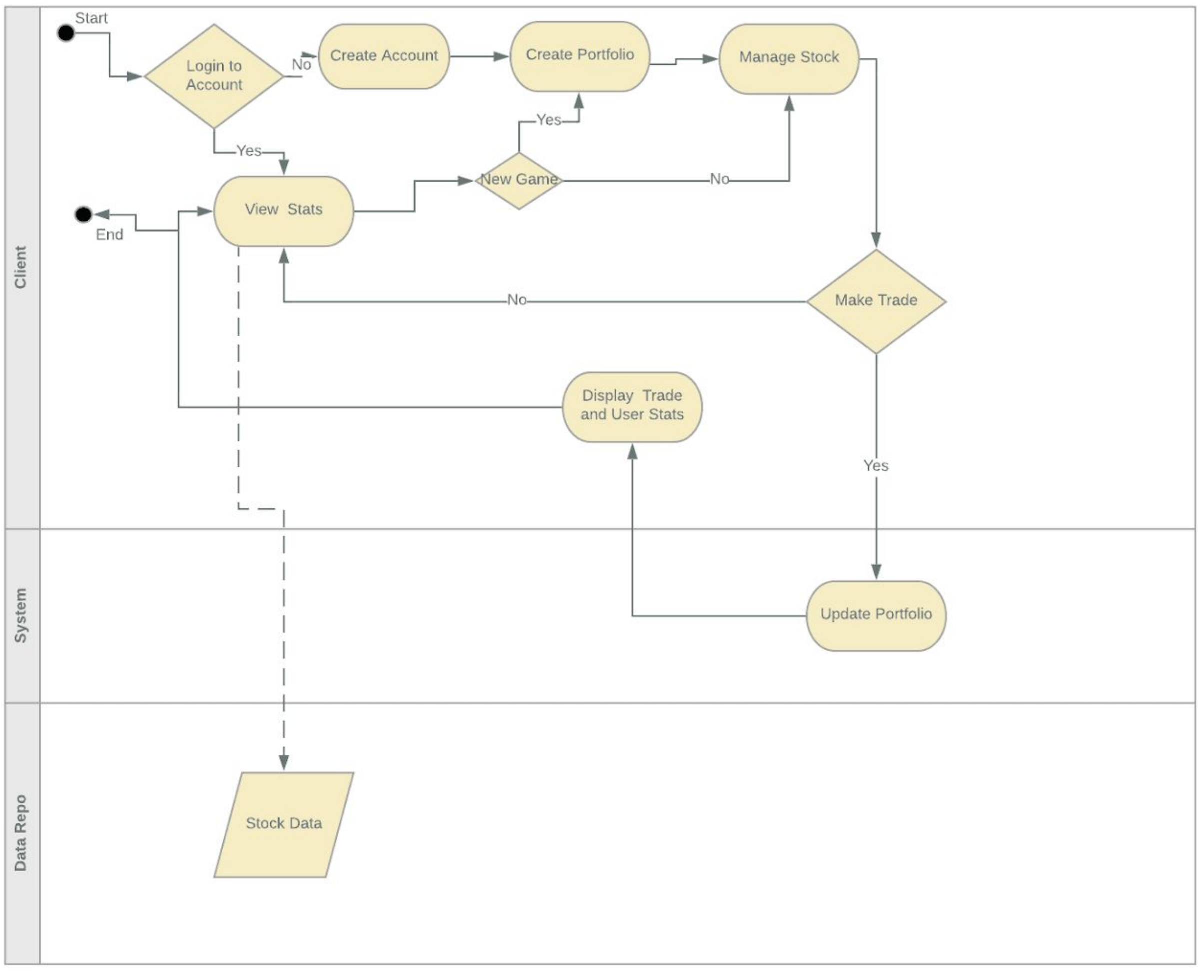


Diagram 1: To-Be Solution Diagram

3.2 Requirements

3.2.1 Functional:

a. Use case: User login

Actors: User, Stock Market platform

Entry Condition: System is functional and the user has an existing account

Flow of Events:

1. User enters credentials
2. System validates the user’s username and password
3. System logs user into the platform

Exit Condition: System displays home page for the user

Exceptions:

1. User enters incorrect credentials
2. System is unable to validate the credentials properly
3. System loses connection

b. Use case: Buy stocks

Actors: User, Stock Market platform

Entry Condition: User already has an account set up and wants to purchase a new stock

Flow of Events:

1. User searches for stocks
2. User selects a company’s stock for purchase
3. System displays the amount of money available
4. User enters the amount of shares to purchase
5. User reviews the transaction
6. User confirms the transaction and submits it
7. System receives responses from user input
8. System performs transaction
9. System displays the updated portfolio

Exit Condition: Portfolio is updated

Exceptions:

1. User enters an amount larger than the available funds
2. User's portfolio does not reflect changes
3. System loses connection

c. Use case: Search for a stock

Actors: User, Stock Market platform

Entry condition: User already has an account set up and wants to look up a stock

Flow of Events:

1. User enters the company's name or symbol
2. System receives the input and retrieves the data
3. System displays the company's stock quote

Exit Condition: Stock quote is displayed

Exceptions:

1. User enters an invalid input
2. User searches for a company's stock that is not within the system's database
3. System loses connection

d. Use case: Sell Stocks

Actors: User, Stock Market Platform

Entry Condition: User already has an account set up and wants to sell stocks

Flow of Events:

1. User searches for stocks to sell
2. User selects a company's stocks to sell
3. System displays the amount of shares available to sell
4. User enters the amount of shares to sell at listed price
5. User reviews the transaction
6. User confirms the transaction and submits it
7. System receives responses from user input
8. System performs transaction

9. System displays the updated portfolio

Exit Condition: Portfolio is updated

Exceptions:

1. User enters an amount larger than than the available shares to sell
2. User's portfolio does not reflect changes
3. System loses connection

e. Use case: View Portfolio

Actors: User, Stock Market Platform

Entry Condition: User already has an account and wants to view portfolio

Flow of Events:

1. User selects to view portfolio
2. System displays the user's portfolio

Exit Condition: Portfolio is displayed

Exceptions:

1. User's portfolio does not show up
2. User's portfolio does not reflect transactions
3. System loses connection

f. **Use case: User Creates Account**

Actors: User, Stock Market platform

Entry Condition: System is functional and the user is able to navigate the UI

Flow of Events:

1. User opts to sign up for the platform
2. System takes them to the sign up page
3. User enters necessary credentials (name, email, username, password, etc.)
4. System stores credentials
5. System sends confirmation email

Exit Condition: User is able to login

Exceptions:

1. Username is already taken
2. System fails to store credentials
3. System loses connection

g. **Use case: View Company Stats**

Actors: User, Stock Market Platform

Entry Condition: User has an account and is signed in

Flow of Events:

1. User searches for company
2. System receives input and retrieves data
3. System displays company page

Exit Condition: Company's stats are displayed to user

Exceptions:

1. System does not display the correct company based on user's search
2. System does not display the correct stats of the company

h. **Use Case: View Buy Confirmation**

Actors: User, Stock Market Platform

Entry Condition: User has already selected the amount of shares to buy and wants to confirm transaction.

Flow of Events:

1. User selects confirmation
2. System receives input
3. System displays the buy confirmation details

Exit Condition: User confirms buy

Exceptions:

1. System does not display the correct information to user
2. System loses connection

i. **Use Case: View Sell Confirmation**

Actors: User, Stock Market System

Entry Condition: User has already selected the amount of shares to sell and wants to confirm transaction.

Flow of Events:

1. User selects confirmation
2. System receives input
3. System displays the sell confirmation details

Exit Condition: User confirms sell

Exceptions:

1. System does not display the correct information to user
2. System loses connection

j. **Use Case: View Transaction History**

Actors: User, Stock Market Platform

Entry Condition: User already has an account and wants to view transaction history

Flow of Events:

1. User selects to view transaction history
2. System displays transaction history

Exit Condition: Transaction history is displayed

Exceptions:

1. Transaction history does not reflect recent transactions
2. Transactions are not accurate

k. **Use Case: Cancel Transaction**

Actors: User, Stock Market Platform

Entry Condition: User opts to buy/sell stock but decides not to complete the transaction

Flow of Events:

1. User decides to buy/sell a certain amount of shares
2. System shows the user a review of the transaction
3. User selects to cancel transaction

Exit Condition:

Exceptions:

1. System does not respond to user's request

l. **Use Case: View Main Menu**

Actors: User, Stock Market System

Entry Condition: User already has an account set up and wants to view the main menu

Flow of Events:

1. User selects to view main menu
2. System displays the main menu to the user

Exit Condition: Main menu is displayed

Exceptions:

1. System does not respond to user's request

m. **Use Case: Incorrect Login**

Actors:

Entry Condition: User enters invalid credentials on login screen

Flow of Events:

1. User is notified that credentials entered are invalid
2. User is able to try again

Exit Condition: User enters valid credentials and is granted access

Exceptions:

1. User continuously enters invalid credentials and after 5 attempts in a short period is locked out for a period of time

n. **Use Case: User Deletes Portfolio**

Actors: User, Stock Market Platform

Entry Condition: User already has an account with portfolios and wants to delete a portfolio

Flow of Events:

1. User selects certain portfolio
2. System displays the user's portfolio
3. User selects delete button

Exit Condition: System shows delete confirmation page

Exceptions:

1. User's portfolio does not show up
2. User's portfolio does not have delete option
3. User's portfolio is not removed from account
4. System loses connection

3.2.2 Non-functional requirements:

a. Reliability: Prices listed are up to date, data will not be lost randomly
. b. Performance: Bugs in the code and in the application will be fully fixed upon detection
c. Usability: Able to buy multiple stocks at once
d. Portability: Usable on Windows 10
e. Scalability: User can create and maintain multiple portfolios
f. Availability: Download available through Github[2]

3.2.3 Constraints

a. Time: There is a limited amount of time for project, limiting capabilities
b. Funding: Limited funding for development team
c. Language limitations: Lower level languages typically take longer to add features

3.3 System models

3.3.1 Use Case Diagrams

Diagram 1: User Login

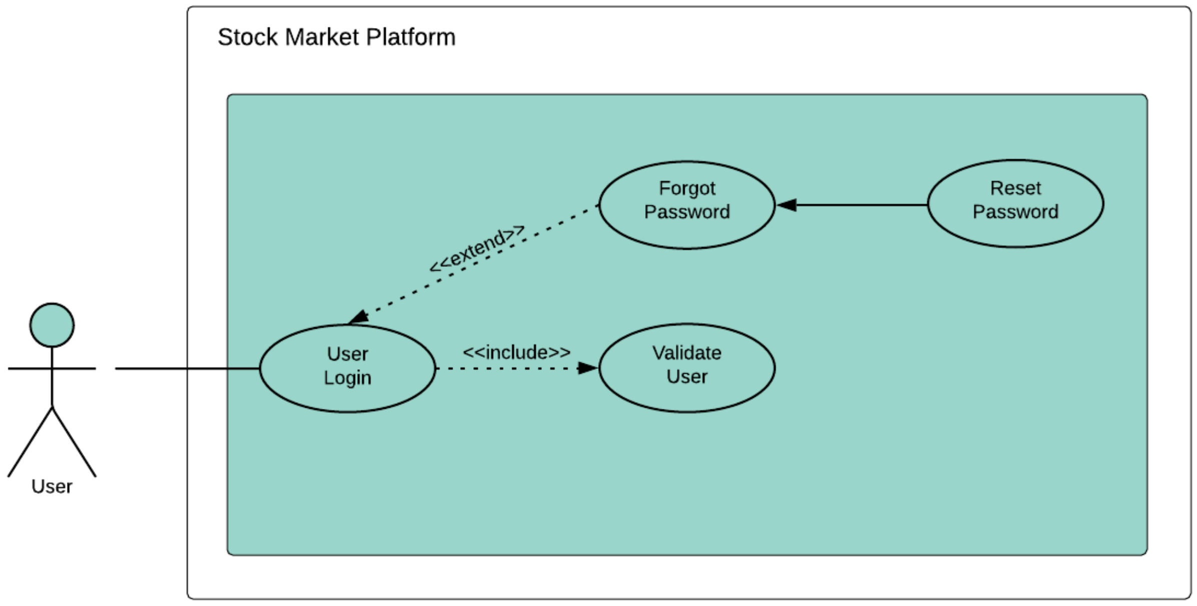


Diagram 2: Search for a Stock

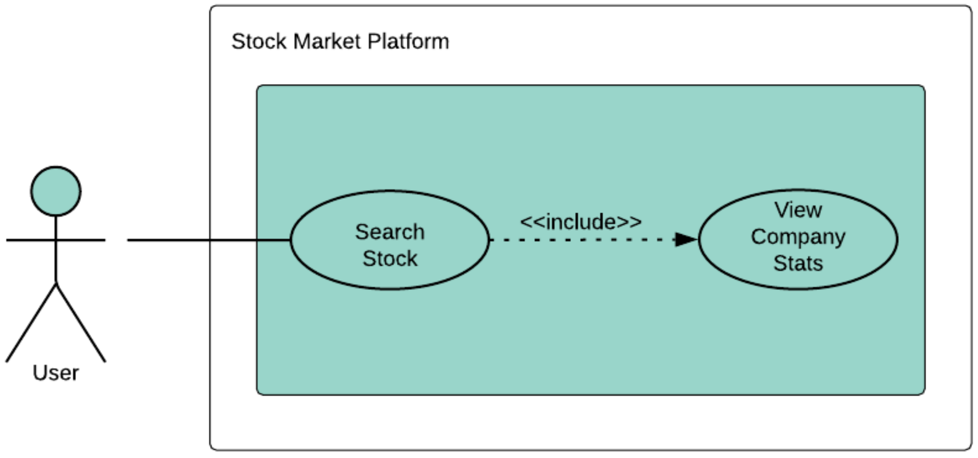


Diagram 3: Buy a Stock

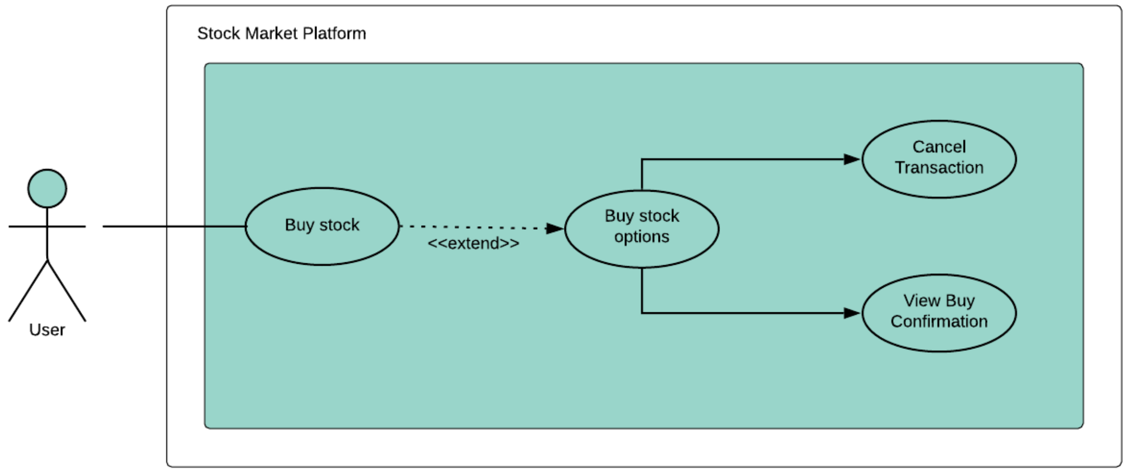
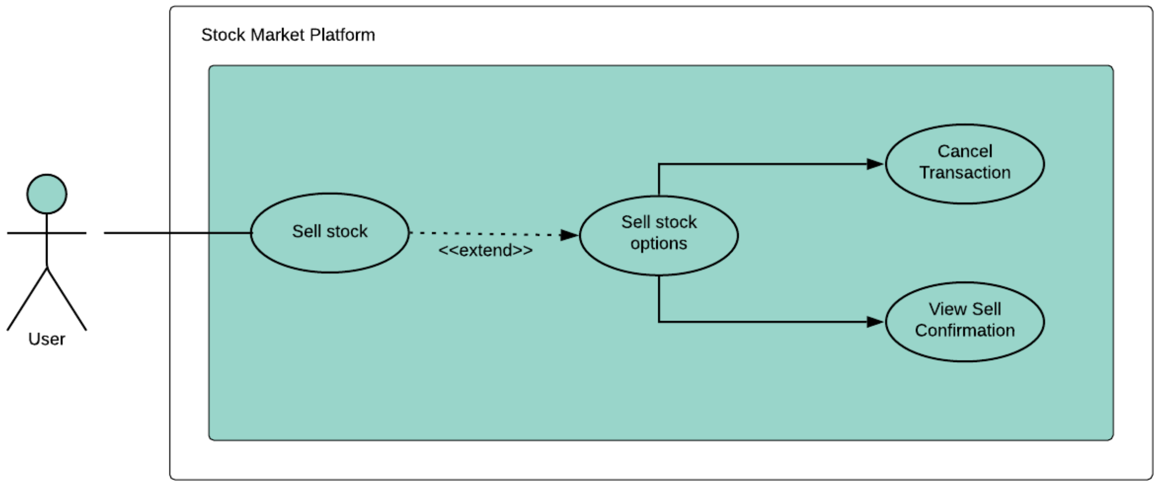


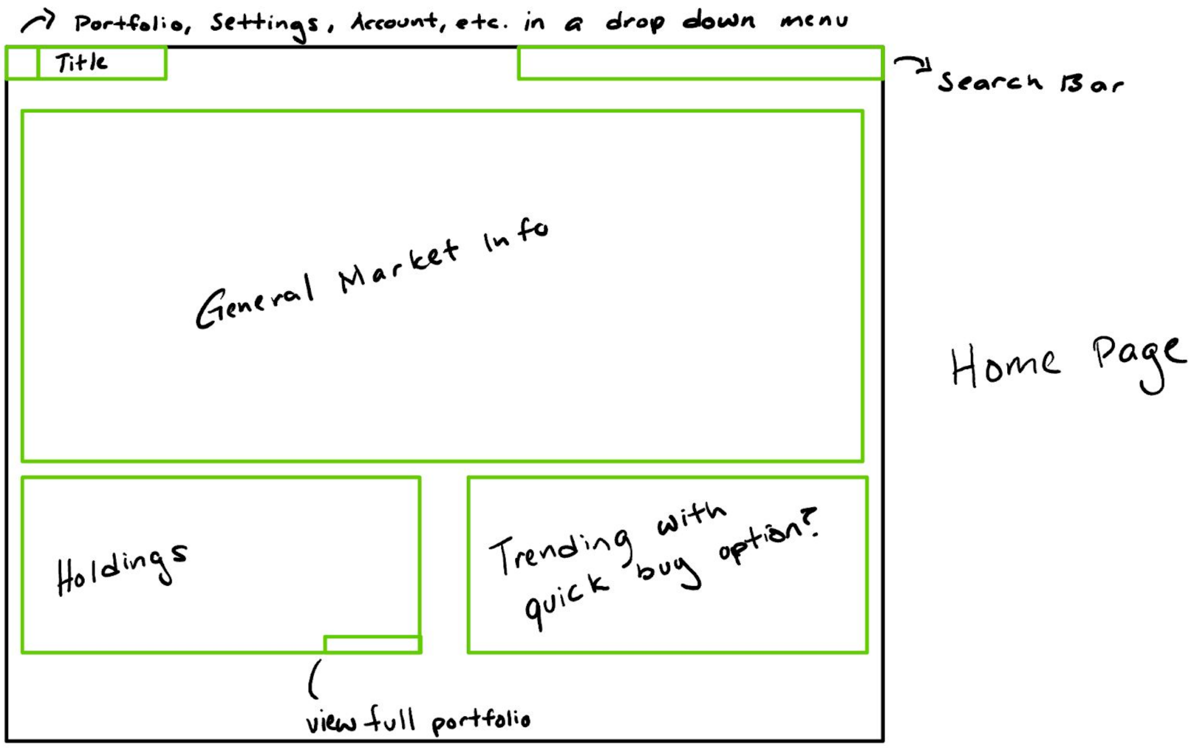
Diagram 4: Sell a Stock

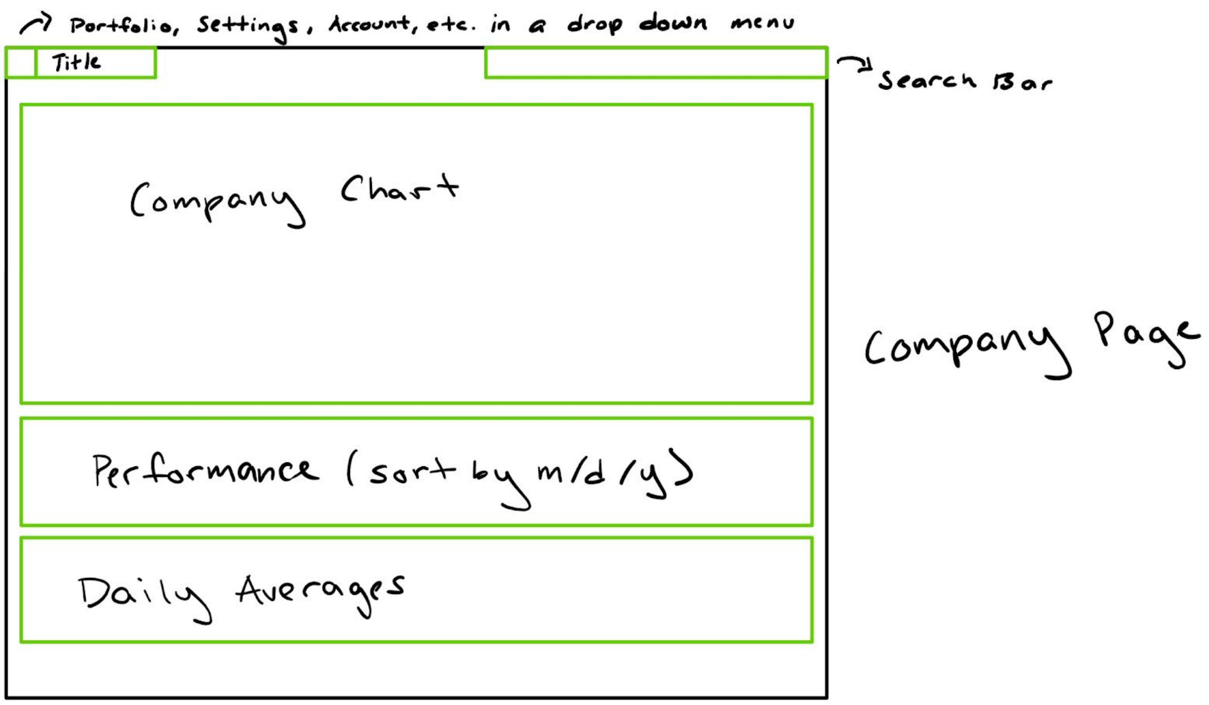


3.3.2 Data Dictionary

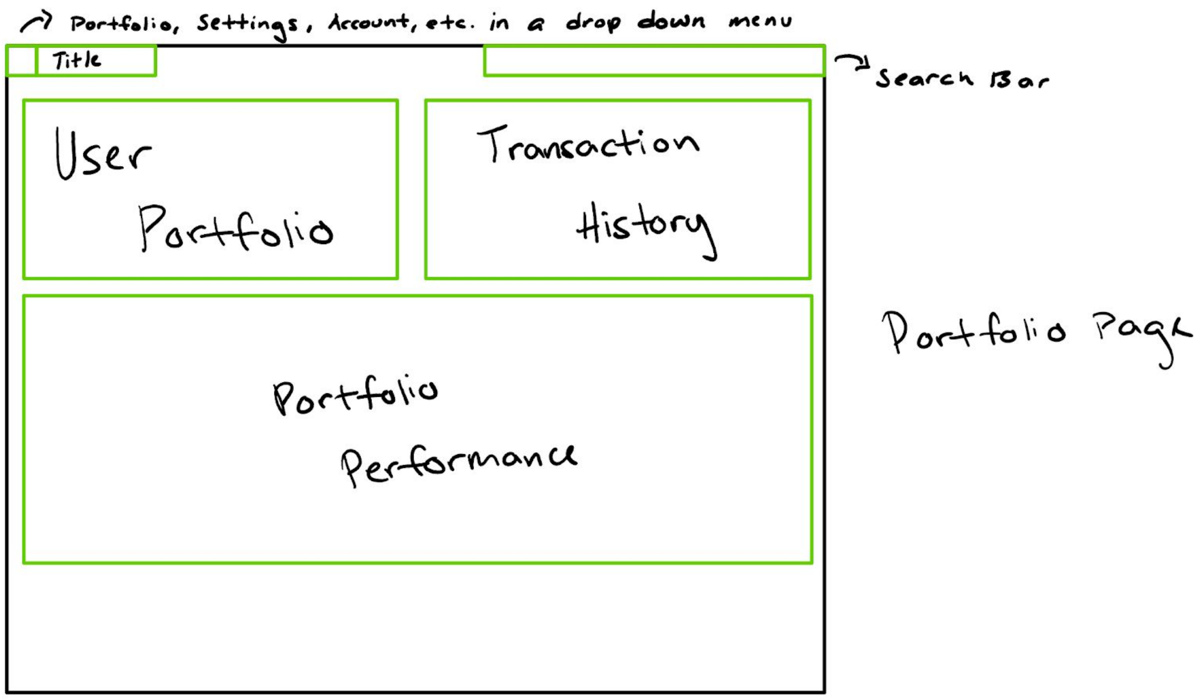
Field Name	**Data Type**	**Data Format**	**Field Size**	**Description**	**Example**
UserID	Integer	###-###	6	Identification number unique to the user	123-456
Last Name	String		30	The last name of the user	Doe
First Name	String		20	The first name of the user	John
Email	String	text@text.text	75	The email of the user when they sign up for the account	jdoe@yahoo.com

3.3.3 User interface Prototypes


Portfolio, Settings, Account, etc. in a drop down menu
Title
Search Bar
General Market Info
Home Page
Holdings
Trending with quick buy option?
view full portfolio



Portfolio, Settings, Account, etc. in a drop down menu
Title
Search Bar
Company Chart
Performance (sort by m/d/y)
Daily Averages
Company Page



Portfolio, Settings, Account, etc. in a drop down menu
Title
Search Bar
User Portfolio
Transaction History
Portfolio Performance
Portfolio Page

3.3.4. Formulas for Calculations

% Net Gain/Net Loss per Stock = (Avg. Current Price * Amount of Stock / Avg Amount Spent per Stock * Amount of Stock) * 100

% Net Gain/Net Loss per Portfolio = (Total Value of All Stocks Owned / Amount Spent Total) * 100

4. Glossary / References

4.1. Define any non-standard acronyms

a. API: Application Programming Interface
b. U.S. : United States of America

4.2. List any references

1. "Free stock market game," *MarketWatch*. [Online]. Available: https://www.marketwatch.com/game.
2. "GitHub," *GitHub*. . "https://github.com/"