PES UNIVERSITY

FINAL SEMESTER ASSESSMENT (FSA)
B.TECH. (CSE)
VI SEMESTER

UE18CS355 – OBJECT ORIENTED ANALYSIS AND DESIGN WITH SOFTWARE ENGINEERING LABORATORY

PROJECT REPORT

ON

STOCK MARKET TRADING

SUBMITTED BY

NAME	SRN
1) Aditi Ahuja	PES1201800165
2) N Sanketh Reddy	PES1201800389
3) Bhavana Madhuri V	PES1201801231

JANUARY – MAY 2021

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
RR CAMPUS,
BENGALURU – 560100, KARNATAKA, INDIA

TABLE OF CONTENTS

Abstract
This project is a software platform designed to simulate the workings of a stock trading platform. The planning has been done in accordance with software engineering principles with regard to system architecture, class organization and use case planning. The project has been comprehensively tested using test cases for each feature of the use cases.

The use cases simulated include buying and selling of stocks, viewing transactions history and summaries and administrative handling of the platform. Buying and selling of stocks simulates actual stock trading with the fluctuating prices and the user's account balance changing accordingly. The user can view their entire transactions history as well as transactions grouped by company and summary based on amount and number of shares.(add about admin use cases)

The use cases were developed and tested individually using unit tests. The use cases were then integrated into a unified platform with combined functionality for each of the use cases. The integration was tested using integration test cases.

CHAPTER 1 :

Software Requirement Specification

1. Introduction

1.1 Purpose

This web app aims at automating the tasks that are supposed to be done as a stock broker. On behalf of traders and investors, stock brokers facilitate or simplify the exchange or buying and selling of the stocks. The main tasks of a trading platform are to provide real time values of each company's stocks,to provide graphical statistics on the stock prices over different time frames, to suggest the customers about certain decisions about buying and selling of the stocks, to assist them in the complete transaction process, to provide real time stock related news and alerts to the customers as notifications. The details of the functionalities are provided in the subsequent sections.

1.2 Intended Audience

Intended audience for this web app includes the trading and stock market enthusiasts. The people who are indulged in stock exchanges are our primary targets. We can attract them to use our trading platform by providing relatively less brokerages. Secondly, as our functionalities include few aspects like providing news alerts(some of them are to be provided only to the subscribers) and analysis of companies, any stock enthusiast can use our software so as to get better knowledge.

1.3 Product Scope

This web app facilitates the businesses and the companies to be traded by the public. This will in turn create additional capital which can be used for company's expansion by selling their company's shares in a market that is public. From a trader's perspective, it creates a huge opportunity to earn money out of trading by making smarter decisions about buying/selling the shares. As a stock broker we are expected to facilitate stocks related transactions to the customers. The objective of this software is to provide the investors with real time data related to the companies so as to assist them in making proper decisions, and also to provide them with news alerts. As there are many stock brokers already present in the

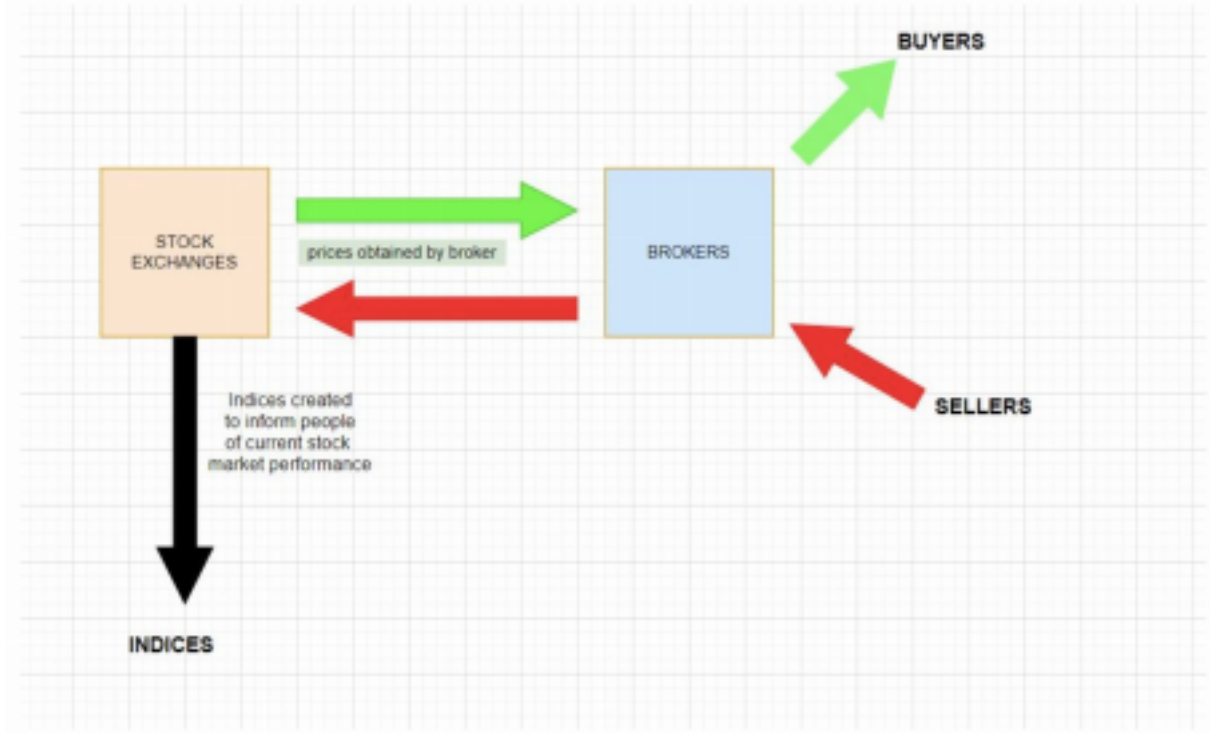
market, it is a huge challenge to beat this competition. So our strategy is to provide less brokerages to the investors and provide them with a great user interface and a smooth transaction environment. Also, we should provide legitimate data and graphs so as to assist our customers in technical analysis about the companies of their interest.

2. Overall Description

2.1 Product Perspective

This software is basically a trading platform assisting the people for buying/selling stocks over the internet with additional features as mentioned in the next section. This is not a self-contained product but there are a series of such products which are already performing well in the market. Zerodha is one such company which is performing really well in the Indian market.

This product is centered around two biggest Indian stock exchanges. These are Bombay Stock Exchange(BSE) and National Stock Exchange(NSE). We are focused only on these exchanges and do not consider any other stock exchanges of any region. Bombay Stock Exchange(BSE) uses **Sensex** index and National Stock Exchange(NSE) uses **Nifty** index. The prices of the stocks of any company at any instance of time that we display will be based on these indices only. This is because, majority or almost all the equity share transactions happen under these exchanges and under any other exchanges.



The above diagram(drawn using draw.io) depicts the outer view on how the entire stock market works. Here we fall under brokers section which is the main interface for the customers to deal with the buying and selling of stocks

2.2 Product Functions

List of functionalities that are expected to be built in our product:

- Easy client registration process
- Obtain real time prices and other data of companies
- Provide order execution tools (stock buy/sell)
- Provide news feed
- Provide scanning tools (top 10 gainers/losers etc)

- Provide charts for technical analysis
- Provide customer support (via email/phone)
- Trading portfolio

2.3 User Classes and Characteristics

As anticipated, there can be three user classes who will use our software

2.3.1 Unsubscribed users:

This is basically for educational purposes for the stock market enthusiasts who don't carry financial transactions on our platform and have not subscribed to our premium features. These users are provided with things like:

- Access to real time stock prices
- Access to data used for fundamental analysis
- Access to price charts/ graphs of companies
- Create their own watch list
- Access to scanning tools

This class of users are not provided with other features that a subscribed user can access. This is due to the fact that we obviously cannot earn brokerage fees from this class of users. Hence this class of users are provided with basic information which even other websites/apps provide. We don’t focus on this class of users much and hence we deny many features like support, news etc.

2.3.2 Subscribed users:

This class of users do not really carry financial transactions on our platform (might or might not be involved in trading on some other platform) but they do want to be accessible to our premium features. To be able to become a member of this class, we charge a certain amount in the form of monthly or annual subscription plan. Following are the features that this class of users get in addition to the features mentioned for unsubscribed users:

- Access to news feed (game changing stock news!)
- Analyst ratings and reports
- Advance charting features like fibonacci plotting, line break charting etc

2.3.3 Investors :

This class of users is our primary focus. These users actually use our platform for the trading purpose as well. We charge a certain percentage of brokerage charges for the transactions carried out through our software. These users are provided with all the features that we design for the product. The following list includes some of the premium features that this class of users can access in addition to those mentioned in the above sections, some of them make sense only for this class of users and irrelevant to the other classes of users:

- Access to order execution tools
- Trading portfolio
- Suggestions related to buy/sell stocks
- Customer support via call and email

2.4 Operating Environment

This software is hosted on a server, it runs as a web service. Any device that has an internet access can avail the services that are provided by the software over the internet on their web browser. Operating systems supported are Windows, Linux, Mac and Solaris. We need a cloud service on the server side to host our webapp. Also we need cloud service to fetch streaming real time data.

2.5 Design Implementation Constraints

- Fetching streaming real time data would be little challenging as there should not be

any delay.
- We need to use only secured protocol for communication so as to ensure security and data integrity
- Using cloud platform efficiently
- Carrying out encryption of certain important data
- Tools to use are to be decided

2.6 Assumptions and Dependencies:
- Users are expected to have basic knowledge of stocks
- Users should be aware of terms and options related to trading such as short selling, etc
- Users are expected to have stable internet connection to carry out any transactions
- Users are expected to have computing device that can access our service
- Users are expected to have knowledge on brokerage charges so as to carry the business smoothly.

3. External Interface Requirements

3.1. User Interfaces :

The user interface of the software will work with any browser such as Internet Explorer, Mozilla or Netscape Navigator through which the user can access the system.The user interface will be used using any tool or software package such as Java Applet, MS Front Page, EJB etc. Input can be anything in the database.At the top, trading looks like a no-brainer from a user's point of view: what you need is low buying, high sales, and a certain passion in trading to make your billions:

- Find an item to buy or sell (this item is usually called a tool)
- Decide when to buy or sell
- Make a purchase or sell
- Monitor the effect of transactions on your account

It seems simple enough. An intuitive user interface that will make this process an easy trader.That might look better like this.

3.2. Software Interfaces :

We often associate shopping with goods. But in the market the customer also buys services. Let's learn more about these services and their types and types of services. Trading software helps to trade and analyze financial products, such as stocks, options, futures, or money. Most of the time, trading firms provide their clients with trading software to set up trading and manage their accounts. The software can be downloaded and opened from a desktop or mobile device, or it can be downloaded from the web, where the merchant accesses the software through a web site. Merchants can also develop software provided by brokerages.

Some vendors also provide libraries in various languages to make communication with their API easier. For example, a trader may provide a Python library that provides a set of tasks, or methods, for trading rather than listing your own activities to do so. This can help speed up the development of trading systems and / or make them less expensive to develop

To be able to trade security on a particular stock exchange, security must be listed there. Normally, there is at least a medium-sized record-keeping area, but trading is increasingly dependent on physical contact, as today's markets use electronic communication networks, which offer the benefits of increased speed and reduced transaction costs. Exchange trading is restricted to buyers who are members of the exchange. In recent years, various other trading

platforms, such as electronic communication networks, alternative trading systems and "black pools" have removed much of the trade away from traditional stock exchanges.

- The system will be redirected as software that will allow users to complete secure transactions. This will usually be the third most widely used software application for online transactions.
- The stock market is a platform where investors come to trade financial instruments such as stocks, bonds, and other securities. The stock exchange acts as a promoter of these transactions and enables the purchase and sale of shares.
- Ensuring a fair and equitable market for investors to grow.
- Ensure the implementation of the guidelines and guidelines issued by SEBI

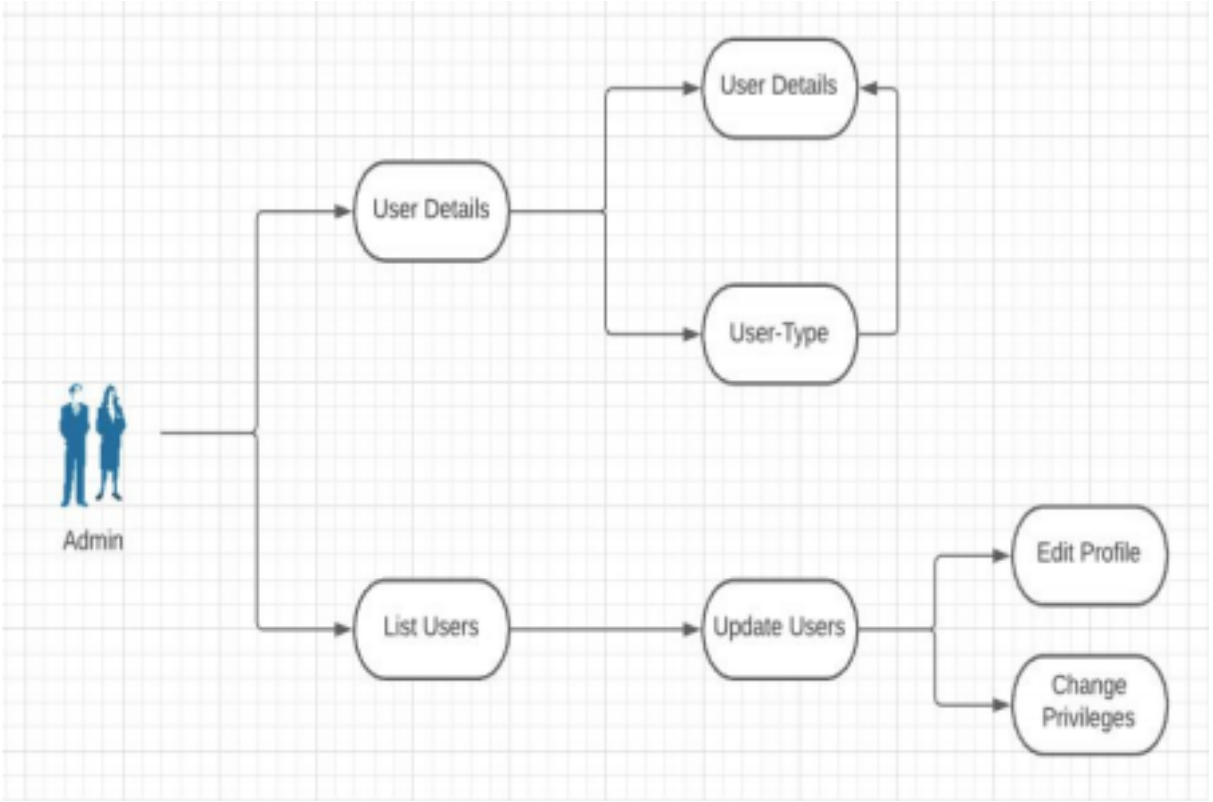
3.3. Communications Interfaces :

Communication on electronic trading platforms is based on a list of well-defined agreements. Although the FIX protocol has grown significantly in the market, certain exchange agreements (also called "Indigenous Methods") have found strong support with people using low latency trading.

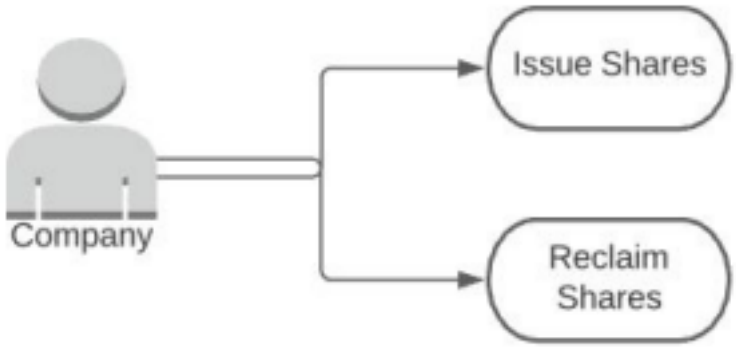
4. Analysis Models

Use Case Diagrams

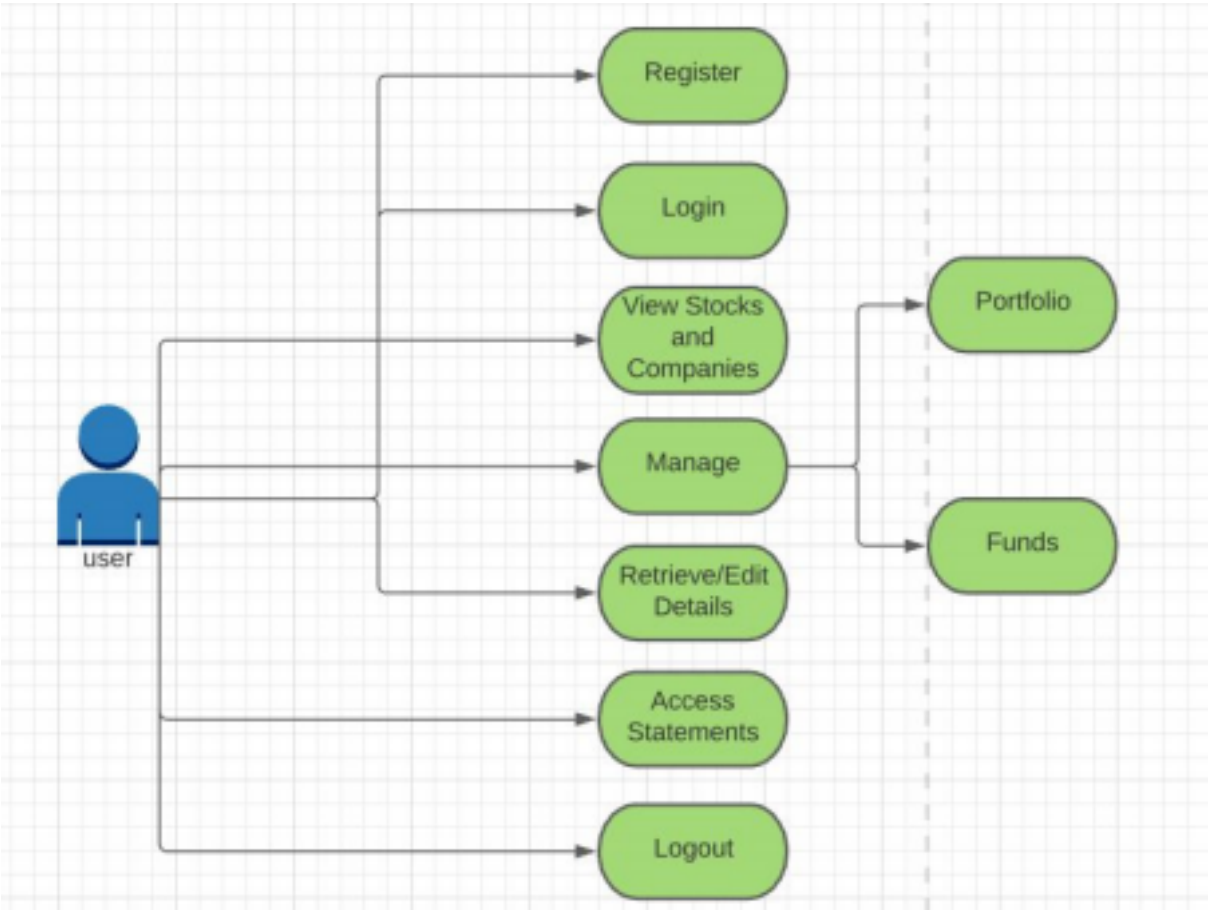
- For the Admin



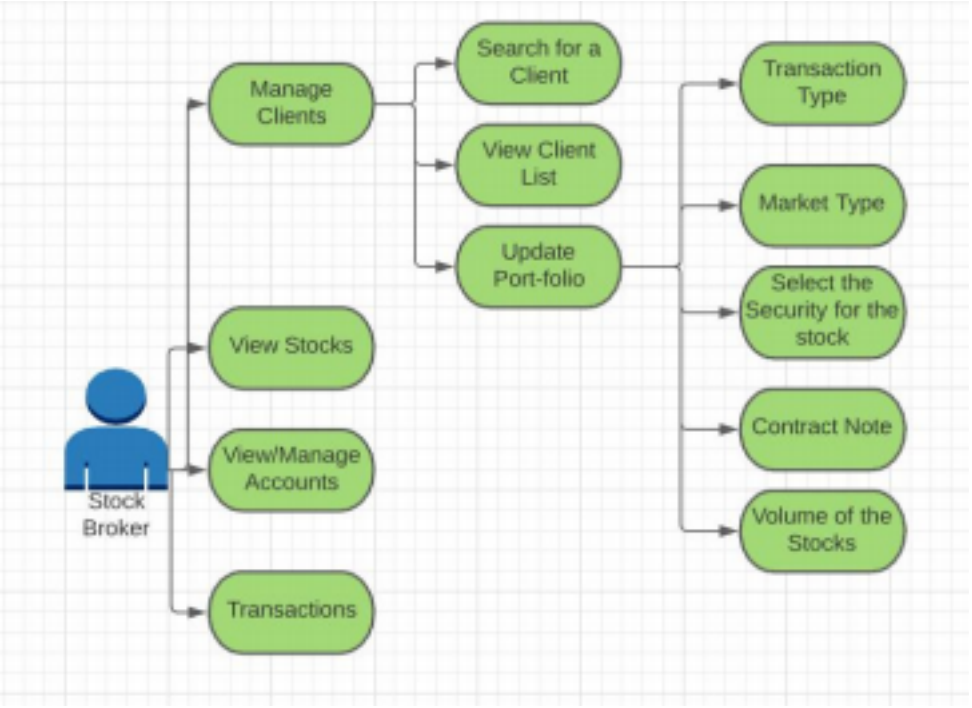
- For the Company



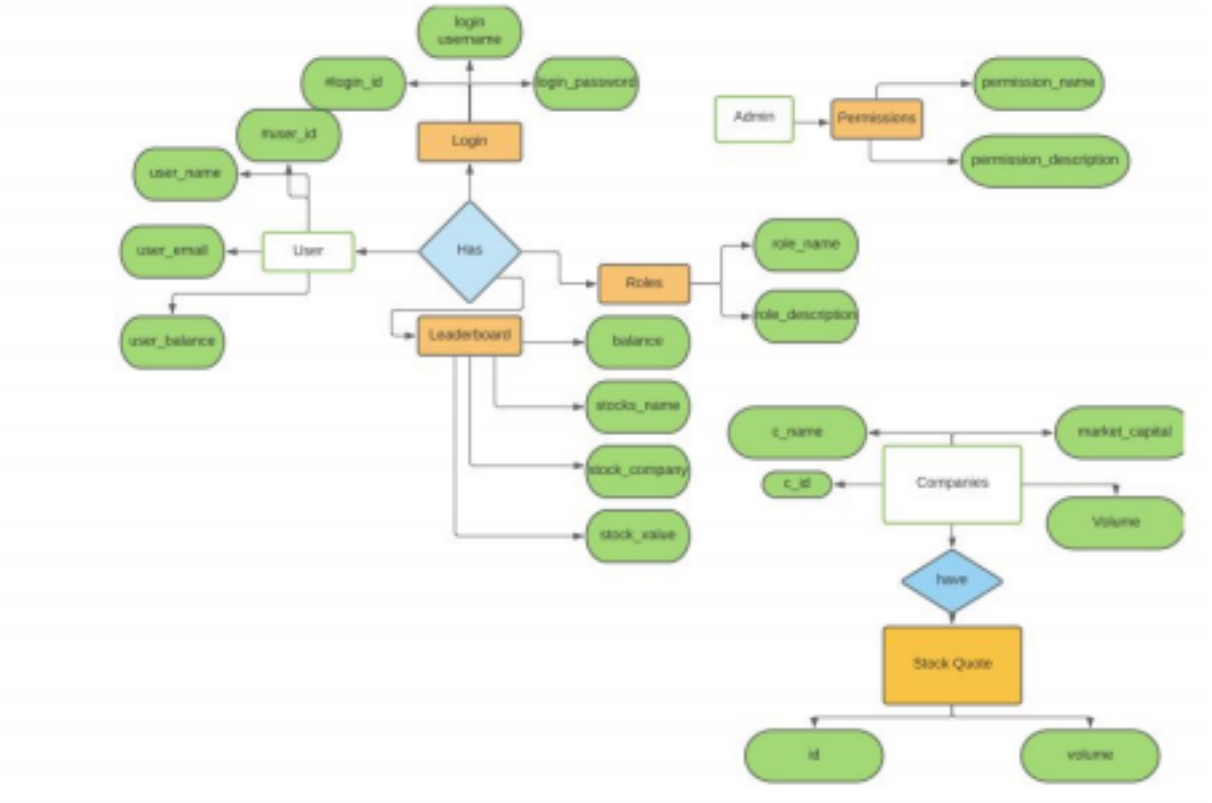
- For the User



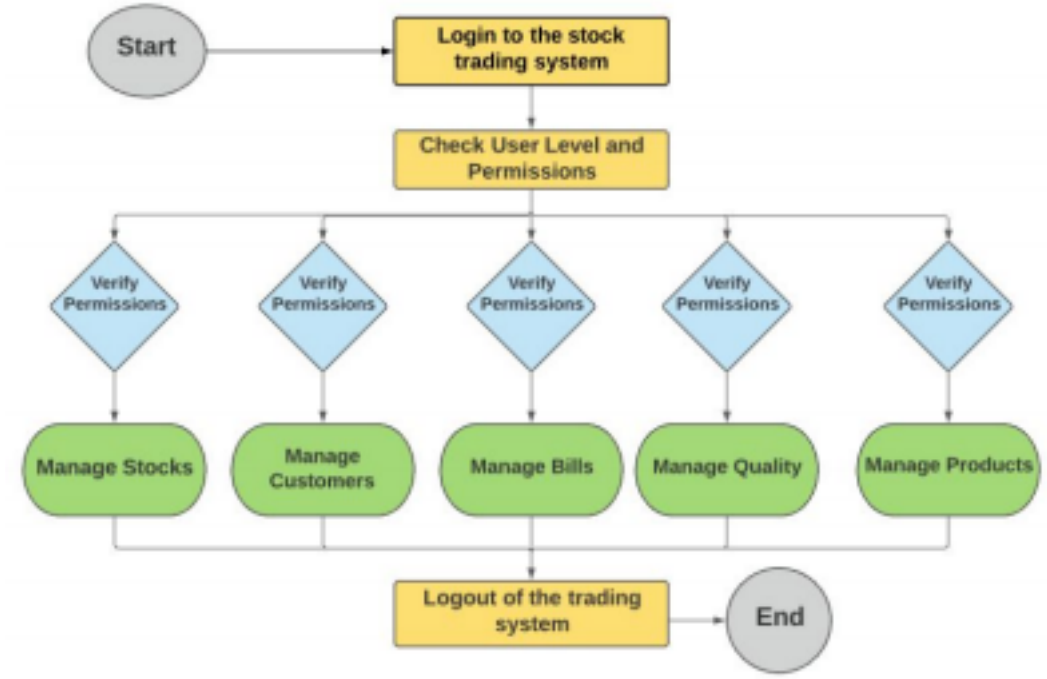
- For the Stock - Broker



- Entity - Relationship Diagram



- Activity Diagram :



5. System Features

This section explains the primary functionalities of the web app.

5.1 Client Registration

5.1.1 Overview and Priority

This is a medium priority feature as the frequency of registration requests are less when compared to transactions and other features. New customers register to our web app in this phase so as to access our services.

5.1.2 Stimulus/Response Sequences:

On click of Register option user is to be taken to a page where the user is requested to provide following details:

- First name, Last name
- Permanent Address
- Bank Details (Acc no, Branch, IFSC)
- PAN card number

These details are to be collected so as to generate **Demat Account Number**

5.1.3 Functional Requirements

- Full name should include only alphabets and spaces
- 4 lines are to be provided for address(each line should allow at most 25 characters)
- Relevant fields for the Acc no, Branch and IFSC code (LLLL######) where L represents letter and # represent numbers are to be provided. If the data provided is out of format, it should be marked as invalid input
- Field for PAN card number should be of the form LLLLL####L. If the data provided is out of format, it should be marked as invalid input

5.2 Obtain real time prices and other data of companies 5.2.1 Overview and Priority
This functionality is of relatively high priority as there shouldn't be any delay in fetching the stock data. Users are to be provided with real time data such as current prices of the companies which is crucial for a customer to make decisions. The value of the shares can be highly volatile, it keeps changing every second. Hence we should access this real time data rapidly.

5.2.2 Stimulus/Response Sequences
This is a feature which is to be invoked automatically as we need real time stock data at any instance

5.2.3 Functional Requirements

- The values of the companies are to be brought to the database, we need to store current day's high and low values of the stocks which can later be used for analysis purpose
- We should store the opening and closing price of the stocks on each day which can be used to construct graphs and get useful insights out of it.
- Aspects like dividend are to be kept track and updated so as to make certain buy/sell decisions
- The data considered for constructing graphs is to be periodically updated so as to provide graphs based on latest data.
-

5.3 Provide Order Execution
5.3.1 Overview and Priority
This phase is of high priority as it is the main module of the entire web app that carries out transactions. Clients are to be properly informed about the price breakup that involves the amount to be paid for stocks that are to be bought plus the brokerage charges.

5.3.2 Stimulus/Response Sequences
This functionality is to be invoked by the operator. Here input will be the demat account number of the user who's carrying out the transaction and the company whose shares he is going to buy. Based on the current price of the stock and our brokerage rates we should display the price breakup. On confirmation from the user side, we should carry out this transaction and should display the buying info on the user's portfolio

5.3.3 Functional Requirements

- We have access to client's Demat Account's number which is required to carry out the transactions
- We should access real time stock value of the selected company at that instance
- Price of shares can be obtained by multiplying number of shares and value per share
- Brokerage can be calculated by multiplying brokerage % and total price of shares obtained in the previous step
- Confirmation from user side is expected to carry out the transaction

5.4 Provide News Feed :

5.4.1 Overview and Priority This phase is of medium priority as we can be flexible in terms of time to deliver the news alerts and updates related to the user. This feature turns out to be very useful for the users to make important buy/sell decisions. This feature doesn't really need any input from the user

5.4.2 Stimulus/Response Sequences This is a feature which is to be invoked automatically as and when we get the news updates from other sources. On getting these updates from the other

sources (TBD) A notification is to be sent to the users about this update

5.4.3 Functional Requirements

- The news got from the sources are to be brought into database, we need to store the details like company which the news is all about
- We should access the users' watchlist and portfolio so as to send them alerts if the news is related to one of those companies, if not a normal in-app notification is to be sent.

5.5 Provide Scanning Tools

5.5.1 Overview and Priority

This functionality is of medium priority. This phase is basically responsible for finding the companies that are top gainers/losers of the day based on opening and closing prices. This helps the user to recognise highly volatile stocks of the day. Based on this data user might wish to choose those companies for intraday trading

5.5.2 Stimulus/Response Sequences

This feature is to be invoked automatically. Based on streaming real time data, we can sort the the companies based on their opening and current price and decide about top gainers/losers and display them

5.5.3 Functional Requirements

- Real time stock prices of each company are to be fetched.
- Opening prices of each stock are to be fetched.
- % gain/loss are to be calculated based on opening price and current price of each companies
- Top 10 gainers and losers are to be calculated and displayed

5.6 Provide Customer Support

5.6.1 Overview and Priority

This phase is of medium priority as a certain amount of delay can be acceptable. This is basically to help our customers to solve certain issues or to provide suggestions which are related to usage of the platform, suggestions related to the transactions(suggest potential buyers/sellers) etc .This feature is a very useful feature as customers can directly contact our support crew through phone or email.

5.6.2 Stimulus/Response Sequences

This phase is initiated by the customer. If a customer wants to contact our support team he can choose one of the methods: email/phone. A proper and legitimate response is to be provided by our support team within minimal amount of time

5.6.3 Functional Requirements

- User has to be provided with an option to contact our support team
- Each of our support experts is to be assigned to handle queries from certain group of customers
- We should periodically check for emails so as to respond quickly
- A good number of people are to be allotted as the support crew
- Faster response is to be provided .

5.7 Provide Trading Portfolio

5.7.1 Overview and Priority

This feature is of relatively high priority as the users are interested to check their share prices hence they will frequently keep looking at their portfolio. Portfolio basically contains details about the shares that are currently held by the users. This provides info like profit/loss earned by particular company,etc

5.7.2 Stimulus/Response Sequences

The portfolio page is to be shown to the user once he/she selects the portfolio option. But the program that does the computation task to calculate details that are to be displayed on this section will be invoked and up and running automatically.

5.7.3 Functional Requirements

- Access the data related to transactions done by particular user
- Data of the shares that held by the users is of primary importance
- Based on buying price and the current price, the gain/loss both in % and price is to be calculated
- The gained amount is then multiplied with the number of shares of that company to get the total loss/gain
- This calculated is to be displayed in the portfolio page

6. Other Nonfunctional Requirements 6.1 Performance Requirements

The database stores a large amount of data and it also helps in quick addition and retrieval of data from it. So, the system furnishes all the required information at the request of the user to view the details.A vast amount of data is stored in the database and it also helps to easily upload and retrieve data from it. So, at the request of the user, the device supplies all the necessary information to access the data.

6.2 Safety Requirements

The possibility of transaction handling is needed in this. The completed transaction must be represented in the databases. While a stock is obtained or sold, there should be no data loss. Also the database needs to be used in such a manner such that it has to eliminate data loss or loss of data integrity.

6.3 Security Requirements

Stock market is a software in which sensitive details of clients are stored. Thus the data should be shielded from interruption and unapproved use of the information of the customers. For this the information bases are assembled with more safety that
there is no trade off as far as privacy is concerned and can only be altered by the administrator who is allowed to do as such.

6.4 Software Quality Attributes

For both the customers and the administrators and also for the developers, the additional quality features are significant. Some of our system's attributes are as follows:

Adaptability:

The system should be adaptable as it must be ready to deal with various sorts of clients and various kinds of admins.

Accessibility:

The framework should be accessible at a reasonable rate. Likewise should be furnished with appropriate license for a certain period.

Correctness:

The system must be precise and less prone to error. The scheme must ensure that each and every module is correct during the design stage itself.

Flexibility:

The framework must be versatile in the sense that different types of users and different types of administrators need to be able to manage it. It can also be run in various environments as well.

Maintainability:

The machine should have self-maintenance capabilities. All must be streamlined from time to time for successful results. This function checks the maintenance and the capacity of administrators to manage the system.

Portability:

As a whole, the device built should be very limited in scale. With the assistance of a hand-held computer, it is conveniently portable.

Reliability:

The framework needs to be accurate. It should be consistent and good outcomes over a period of time should be given.

Reusability:

The system should have the requirement to use it more times than before. If the entire device has a problem, it must be reinstalled and it should be installed and put to use again.

Robustness:
The framework ought to be ethical. For substantial data, it should be able to produce performance, although some of the features have struggled.

Testability:
For all kinds of users, the framework should be proven, predictable, verifiable and absolute.

Usability:
The quality, effectiveness, and satisfaction with which users can accomplish tasks in a specific product setting. High usability means that a system is simple to learn and remember, successful, visually appealing and enjoyable to use and easy to recover from mistakes.

6.5 Business Rules
The rules to be followed are:

- No pointless clashes and disturbances are encouraged
- No bribing must be involved.
- If the client wants an inquiry, he should surely have an ID and a password.

7 . Other Requirements :
The stock market is rife with a variety of industries, from agriculture to esports. Financial assets and tools in a given market or company may fluctuate at very high prices and at unusual times. If you are just beginning to figure out what these changes are and when they can be, then there is an alpha to do.
"Internationalization" of security markets refers to the growing problem of suppliers, investors, retailers, retailers and cross-border markets seeking transactions. This benefits all stakeholders, including the affected nations, if they are accountable.
Governments should recognize that security is widely traded in foreign markets and that investors seek greater investment opportunities in foreign markets, and that this practice is driven by economic power, promotes competition, and increases the depth and volume of money in existing markets. Thus, governments need to see what steps can be taken to increase the efficiency of growing global trading markets while providing market integrity and investor protection.
The source code created for this program will be stored in the configuration management tool.

Legal Requirements
Global financial markets are governed by laws and regulations aimed at protecting the investor and the consumer.
Under the SEBI Act, 1992, SEBI is empowered to conduct stock audits. SEBI has been monitoring stock exchange once a year from 1995-96. During this review, a review of market performance, organizational structure and exchange management management was conducted to ensure that:

- Exchanges provide a fair, equitable and growing market for investors

- Organization, systems and exchange practices comply with the Securities Contracts Act (SC (R) Act), 1956 and the laws enacted under them.
- The exchange operated indicators, guidelines and instructions issued by SEBI from time to time.
- The exchange is subject to the conditions, if any, imposed on it during the renewal / granting of recognition under section 4 of the SC (R) Act, 1956.

Storing Financial Stocks :

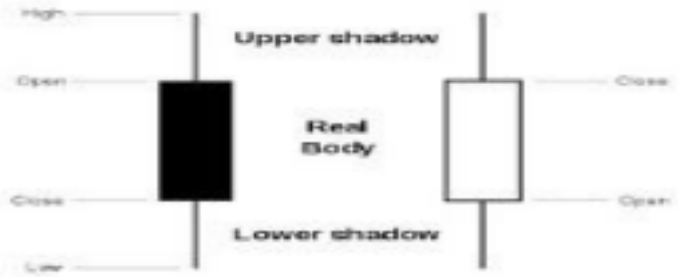
4 million lines could cover the last 20 years of price bars made of common goods other than extended trading hours - such as futures contracts or regular cash shares -. In the event of a tick analysis it will be bigger than it is, but the details of the ticks are really expensive to find, manage and make money; and unless you reverse scalping techniques or work in the HFT industry their profits could be in doubt.

Although 4 million rows sound unattractive, we need to understand that it is 4 million rows per item. So analyzing all the assets from Russell 2000 could mean 8 billion lines (now large). Enter other European stock markets, your home country, small and medium-sized caps, commodities and forex wherever you go: you just came across a huge database of details.

- SQL relational databases.
- Serialized storage of large arrays.
- Key/Value databases (such as Oracle Berkeley DB).
- CSV files.

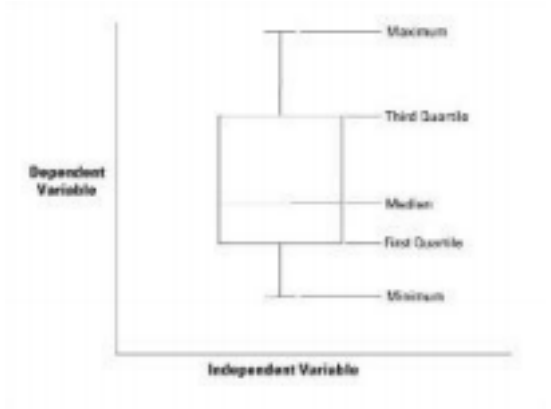
Why do we use them :

Almost every data you come across in the financial markets will be price bars or candles. There is a reason for this. The high / low / open / close structure of the financial price bar is similar to the old beard and box areas used in the calculation but are easy to find.



I used the word parallel because all the important details given to the beard and the box (quartiles and medians) are not found on the financial lampstand. Quartiles and mediators are not contradictory, providing descriptive details about the variables being analyzed. Finding prices for building high, low candlesticks, opening and closing for a while is very easy. You just need a list of all the tasks at a time. Trading is now done electronically and happens in one way, but remember that this was not always the case. Electronic trading is very modern (it began to gain popularity in the mid-80s).

Electronic trading is very modern (it began to gain popularity in the mid80s). Prior to that, trading pits were common and goods were sold in pits from time to time. The concept of a continuous trading market as we know it was not working at the time.



One might wonder why instruments and commodities can be used in financial markets (I think their use can lead to an understanding of effective and innovative pricing methods), and the reason for that is simple. Candles are thought to have been first used during the 18th century by a Japanese rice merchant named Munehisa Homma. If you are wondering if there were future contracts in the 18th century you would like to know that although the original contract was Chicago Board of Trade there were future markets in Europe and Japan in the 17th century.

Performance issues without optimization :

Using this method, there are performance issues:

When you do a lot of data load (using Python and ORM to make sure timestamps are properly managed in SQLite) it takes 14 minutes on 1 VPS server. Receiving all given session data takes 5 seconds on the same VPS server. The same effect on finding the opening or closing price.

14 minutes sounds like too much but bulk loads are rare and may be acceptable. Or finding a more optimized data upload method is not an important point. Just grab a cup of coffee or loads of system packages at the initial data distribution, which will only happen once.

Conversely, 5 seconds may not sound great but it is. Keep in mind that we are dealing here with the very rich simulation situations of Monte Carlo. That means thousands of cycles. So 5 seconds per thousand most of the time. We need to do well here.

As mentioned earlier, the real problem in these areas is recovery time. In serialized arrays, it can be as low as microseconds because the strategy can be defined to allocate space in the memory of all data.

If you think every month has 31 days to get a given minute it's just a matter of looking at a very simple memory. There is no such thing as an index or search activity.

Summary :

Keeping financial value data in a relationship database is not a good idea for performance. Financial market data is a series of data times, and its frequent use is a long data option using basic search and retrieval queries.

Apart from this, having data on a relationship website makes it easier to work with. So you can use it.

The three simple application tips / patterns mentioned here will lead to better results:

- Create a timestamp indicator.
- Divide data sets into different details (in SQLite, that's quite simple).
- Retrieve data subsets from the relationships database and use the latest memory / memory list.

Appendix A: Glossary

Stock Market Basic Terminologies :

1. **NSE :** National Stock Exchange.

2. **BSE** : Bombay Stock Exchange.

3. **Demat Account** : Decentralized Account.

4. **Sensex** : The figure displays the relative prices of the Mumbai (Bombay) Stock Exchange shares.

5. **Agent** : He works on behalf of the investor in the purchase or purchase of stock, a brokerage firm is said to be an agent.

6. **PAN :** Permanent Account Number issued by the government of India.

7. **PE Ratio** : The price-earnings ratio (P/E ratio) compares the share price of a company to its per-share earnings. A high P/E ratio could mean that the stock of a company is overvalued.

8. **EPS Ratio** : Earning per share (EPS) proportion quantifies the number of dollars of net gain that have been procured by each portion of basic stock during a specific time span.

9. **Portfolio** : Holding of any individual or foundation. A portfolio includes different kinds of securities of various organizations working in various areas.
10. **Yield** : In terms of percentage, it is the indicator of investment return. The stock yield is determined by dividing the current share price by the annual dividend paid for that share by the company.

Appendix B: Field Layouts

An Excel sheet containing field layouts and properties/attributes and report requirements.

Sample sheet with information required to register the customer

Field	Length	Data Type	Description	Is Mandatory
Account Number	16	Numeric	The account number of the client	Y
IFSC Code	11	Alphanumeric		Y
Card Amount	20	Numeric		Y
Mandate Start Date	8	Date	Date of Mandate Registration	N
Mandate End Date	8	Date	Date of Mandate Expiry	N
Status	25	Alphanumeric	Status of Registration	Y
Customer Name	60	String		Y
Reject Reason Code	4	String	Reject Reason code in case mandate is rejected	N

Sample Report Requirements: Include the fields to be included in the report

Registration Report	Transaction Report
Bank Account Number	Transaction Reference Number
IFSC Code	Bank Account Number
Bank Name	IFSC Code
Account Status	Bank Name
Account Type	Customer Name
Customer Name	Card Number

Card Number	Debit Transaction Amount
SI Start Date	Transaction Date
Status	Status
Remarks	Debit Attempt Number
	Remarks

Appendix C: Requirement Traceability Matrix

Sl. No	Requireme--nt ID	Brief Description of Requirement	Architect--ure Reference	Design Reference	Code File Referenc e	Test Case ID	System Test Case ID
1	Req01	Login for the applications	User analysis model	Shows the user that he is successfully logged in by redirecting the user to home page with his session information	app.py	UT-07	
				Prints “Incorrect username/pass word!”		UT-08	
				Prints “Incorrect username/pass word!”		UT-09	
2	Req02	Register new user with proper credentials	User analysis model	Prints “You have successfully registered”	app.py	UT-01	
		Register new user with an existing username		Prints “Account already exists in this username!”		UT-02	
		Register new user with unaccepted username		Prints “Username must contain only characters and numbers!”		UT-03	
		Register new user with unaccepted phone number		Prints “Phone Number format is wrong”		UT-04	
		Register new user with unaccepted phone number		Prints “You are not old enough to Register!”		UT-05	
		Trying to register with incomplete information		Pops up an alert box next to the empty textbox with ”Please fill this field”		UT-06	
3	Req03	Reload Home/Buy/Sell/Profile page	User analysis	Reloads the same page without	app.py	IT-01	

		Reload History/Company-wise transactions pages	model	terminating the session		IT-04	
4	Req04	Navigation between Home/Buy/Sell/Profile pages	User analysis model	Loads the contents of the required page	app.py	IT-02	
		Navigation between the Transaction History and Company-wise transactions pages.		Navigates to another page without session termination.		IT-03	
5	Req05	The stock prices should not be constant	Company analysis model	A new thread is spawned in the application for fluctuating the stock prices to simulate the real stock price fluctuation	app.py, buy.html	UT-09	
		The updated stock prices must be displayed in the webpage		There is hot reload function implemented in the Buy and Sell page so that he stock price is updated real time		UT-10	
6	Req06	Must display all the available stocks	User analysis model	If a user buys a stock it is sent to the View/Sell page and removed from the table in the Buy page	app.py, buy.html	UT-11	
		When a user buys a stock from the company and if the company has more than 1 stock		If Tesla has 7 stocks and if the user clicks on “Buy” one time then the available company stocks reduces to 6		UT-13	
		When a user buys a stock from the company which has only 1 stock remaining		If Google has only 1 stock remaining and if the user clicks on “Buy” then the Google row is removed from the /buy webpage		UT-14	
		Only authorized users should be able to perform buy transactions.	Company analysis model	Authorised users are shown a message Correct password and redirected when they enter the correct credit card password.	app.py,verify.html	UT-26	
		Each buy transaction must deduct from the user’s account balance	User analysis model	The user balance is updated in the database.	app.py		
7	Req07	Must display all the bought stocks	Company analysis	If the user sells a stock it is removed from the buy sell	app.py,sell.html	UT-12	

			model	page and again appended to the table in the Buy page so that other users can buy			
		Each sell transaction must add to the user's account balance	User analysis model	The user balance is updated in the database.	app.py	UT-20	
8	Req08	The summary of company-wise transactions based on amount spent and earned should be displayed.	User analysis model	Buy and sell amounts are displayed in the summary details.	app.py, view.html	UT-21	
		Summarise number of transactions per company		Number of buy and sell transactions are displayed in the summary details.	app.py, transactions.html	UT-22	

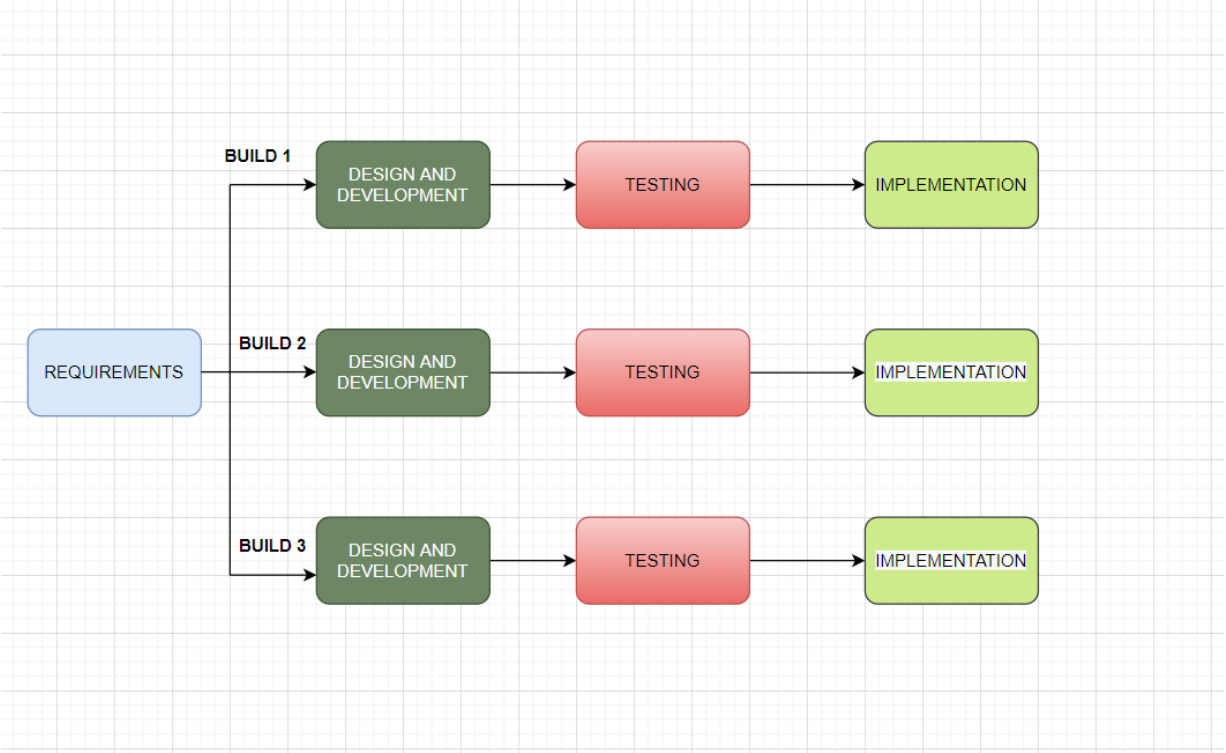
CHAPTER 2 :

Project Plan Document

1. Lifecycle of the project

Iterative and Incremental Approach

This model suggests us to start from a simple implementation of a smaller set of functionalities of the project and keep improvising these features iteratively until we achieve a final stage where the complete system is implemented and can be put under production. In this model we don't have to decide about a complete specification of the requirements in prior, but we need to define major requirements . We can start development by specifying and implementing certain parts of the project and keep modifying the features as and when required and implementation goes accordingly. At the end of each of such iterations, we produce a new version of the software. At each iteration, we achieve design modifications on the previous iteration and add some additional functionalities.In this model basically, the system goes through repeated cycles(iterative) and at a time we develop in smaller portions(incremental)



Above image(drawn using draw.io) shows the diagrammatic representation of the model that is used to build our software.

Reasons for using this approach:

- In the life cycle, some important working functionality can be quickly and early developed
- We obtain early and periodic results
- We can parallely develop features
- We can measure the development progress
- Testing becomes easy during smaller iterations
- Better risk analysis
- We can change certain requirements in the later stages as well
- Less initial operating time
- Software can be developed early and which helps in customer evaluation and review

2. Tools

Planning

SpiraTeam
This tool is used to manage our entire project lifecycle. This tool assists in specifying requirements, specifying test cases and planning tasks and coding them up. This tool is powerful, straightforward and fast. It helps in creating user stories, assigning them to sprints, setting deadlines, and getting them to work . SpiraTeam's flexible planning boards as well as powerful dashboards help to assign and track the above mentioned things.

Design

Reactjs
React is an efficient and flexible javascript library that can be used to easily and impressively build user interfaces for the software. It is maintained by facebook

Version Control

Git
It is an open source version control system designed in order to handle projects efficiently. As we are using iterative+incremental approach it is easier to maintain branches for different builds because of git's branching capabilities

Development

Visual Studio IDE
This includes a code editor that supports the code completion component(IntelliSense) and also helps in the task of code refactoring. Live coding assistance is provided regardless of the coding language that has been used to code. Team productivity can be increased with the assistance of the collaborative capabilities of this platform.

Github
It offers SCM(source code management) and distributed version control functionalities of git plus its own features. It's free of cost and provides additional functionalities like bug tracking, task management, feature requests and access control.

Bug tracking

SpiraTeam
Helps in managing tasks and bugs quickly and easily, we can sync them with our team. We can customise incident fields including statuses, priorities, defect types and severities. This platform is capable of reporting about the bugs via email.

Testing

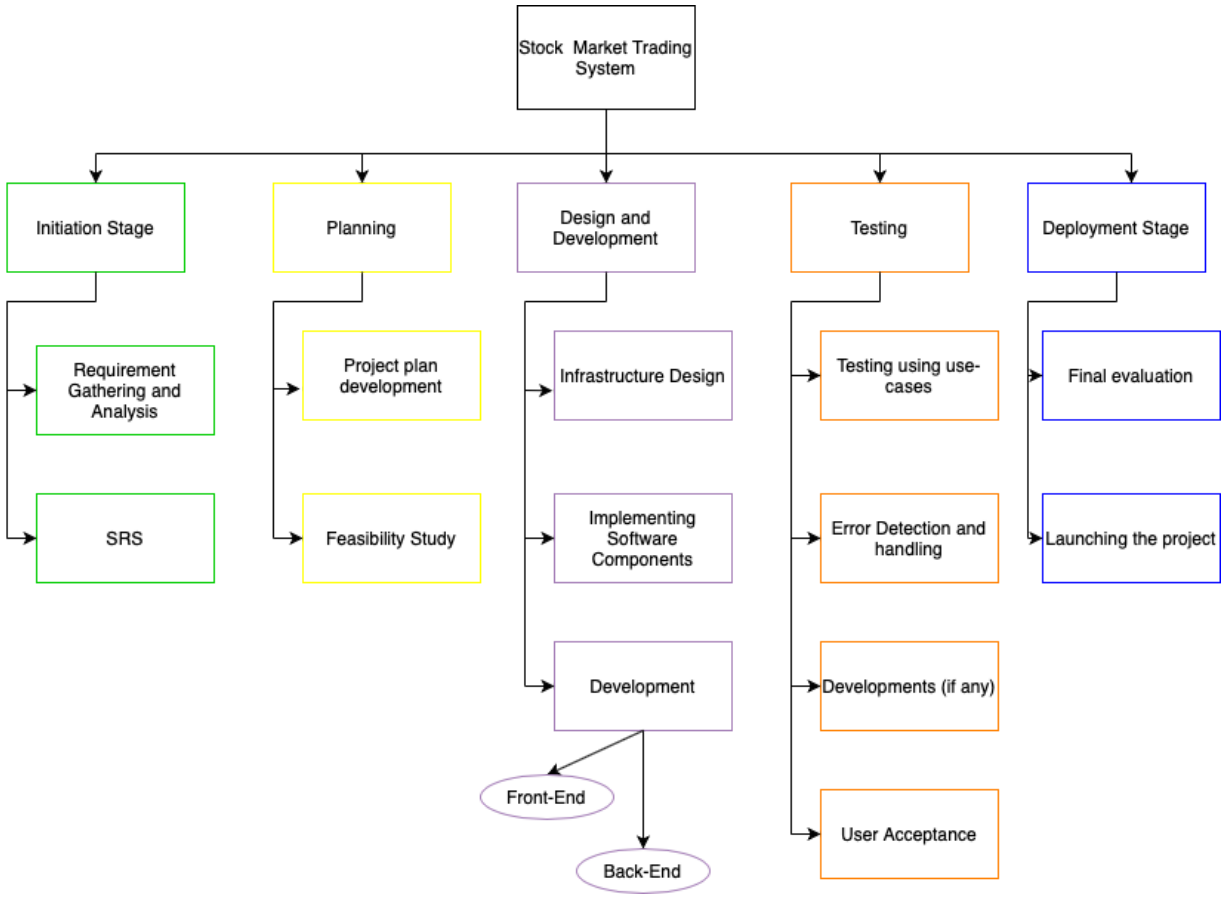
Selenium
This is a testing framework to perform testing of web applications across various browsers and platforms like Windows, Mac, and Linux. This platform allows us to write tastes in many programming languages like Java, PHP, C#, Python, Groovy, Ruby, and Perl. Record and playback features are offered in order to write tests without learning Selenium IDE

Postman
This is an API testing tool. API developers can use this tool to test the REST APIs that they develop. This tool is very easy to use and gives quick results of the queries. Protocols can be changed and the body can be sent easily based on the API definition.

3. Work Breakdown Structure

The work breakdown structure has been splitted into 5 parts:

- The initiation addresses all the activities that contribute to the project's design stage, such as gathering the required things and analysing what the project needs and designing a SRS document.
- The planning stage indicates doing some study required for the project and developing a plan for the project.
- The design stage manages the fundamental diagrams that should be made for the designers to have a superior understanding of the classes, programming segments and framework engineering. The development stage is where all the features are implemented by a team of developers who write code.
- We then move on to the stage of testing. This is where the code is checked by running it against several system generated test cases to search for any errors.
- Finally, it is rolled out into production after the software has passed all quality tests. It needs to be tracked and maintained continuously.



4. Project Deliverables

4.1 Project Management

- Management Plan (Build)

There is a collection of mechanisms and priorities to ensure that the software can be accessed and reused in the long term. It includes identifying a series of sub-tasks, assigning teams, long- and short-term targets.

- SRS (Build)

A comprehensive project plan has been made in light of the timeline and resources.

4.2 Initiation

- Project Planning (Build)

A project plan has been made, provided the timeline and resources.

4.3 Requirement Analysis

- Requirement Gathering (Build)

This project inherits customer requirements that were taken into account when the Bombay Stock Exchange (BSE) was originally constructed.

- Requirement Analysis (Build)

The criteria collected are evaluated for feasibility and are equivalent to those of the Bombay Stock Exchange.

4.4 Design and Development

- Implementing Software Components (Reuse)

A software like Bombay Stock Exchange will be implemented.

- Front-End (Reuse)

The front-end will look alike as the BSE front-end.

- Back-end (Build)

Based on the scope of the application, the database is designed from scratch.

- Integration (Build)

Using suitable frameworks, the frontend and backend will be combined, the details of which have been given in the SRS document.

4.5 Testing

- Error detection and Handling (Reuse)

With auto use-case generators and evaluators, several open-source testing tools exist. We're going to make use of tools like Selenium and Postman.

- User - acceptance (Build)

User acceptance testing will be carried out in the vicinity of the team with a selection of participants, and any improvements to be made would be derived from their responses.

4.6 Deployment

- Rollout

The product will be rolled-out for general use after all the quality tests and the satisfaction of the users.

5. Effort Estimation in terms of Person Months

The effort Estimation technique used in this Plan is the Three-point estimate. Three-point Estimation looks at three values −

- the most optimistic estimate (O),
- a most likely estimate (M)
- a pessimistic estimate (least likely estimate (L)).

Three-point Estimate (E) is based on the simple average and follows triangular distribution.

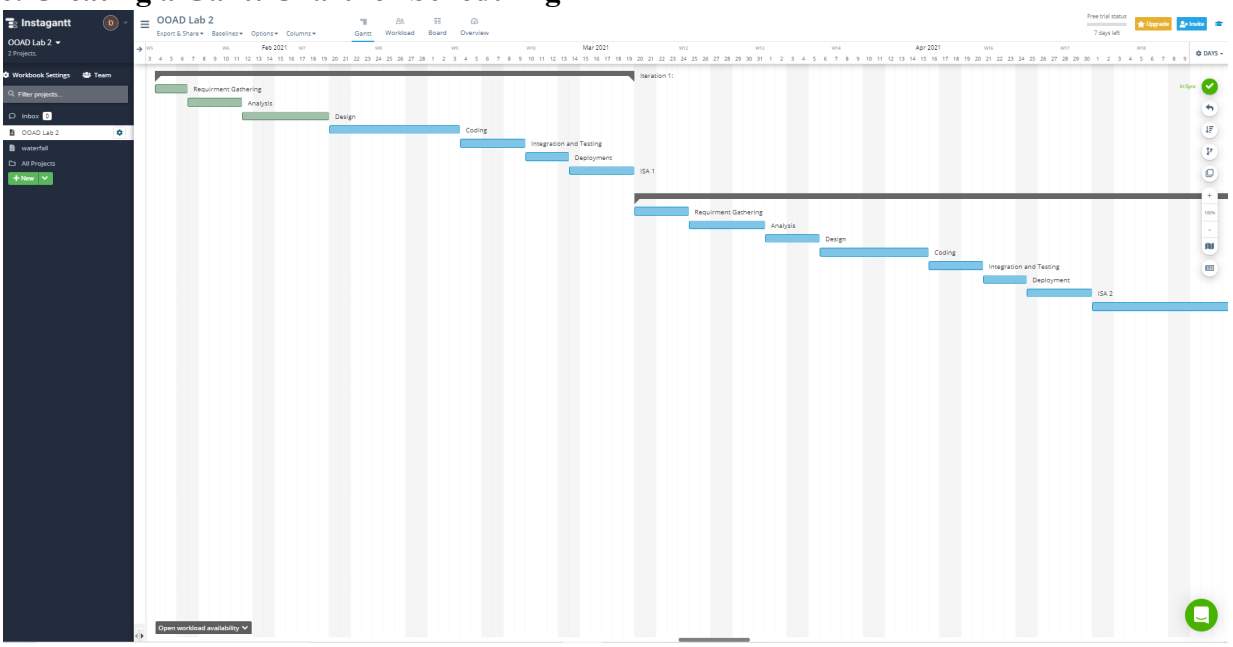
- E = (O + M + L) / 3

The tasks that we divide out effort into are:

Front End:-
Most Optimistic Estimate:- 0.4 person Months
Most Likely Estimate:- 0.6 person Months
Pessimistic Estimate:- 1 Person Month
Final Estimate for the task:- (0.4 + 0.6 + 1) / 3 = 0.67 Person Months

Back End :-
Most Optimistic Estimate:- 0.8 person Months
Most Likely Estimate:- 1 person Months
Pessimistic Estimate:- 1.5 Person Month
Final Estimate for the task:- (0.8 + 1 + 1.5) / 3 = 1.1 Person Months

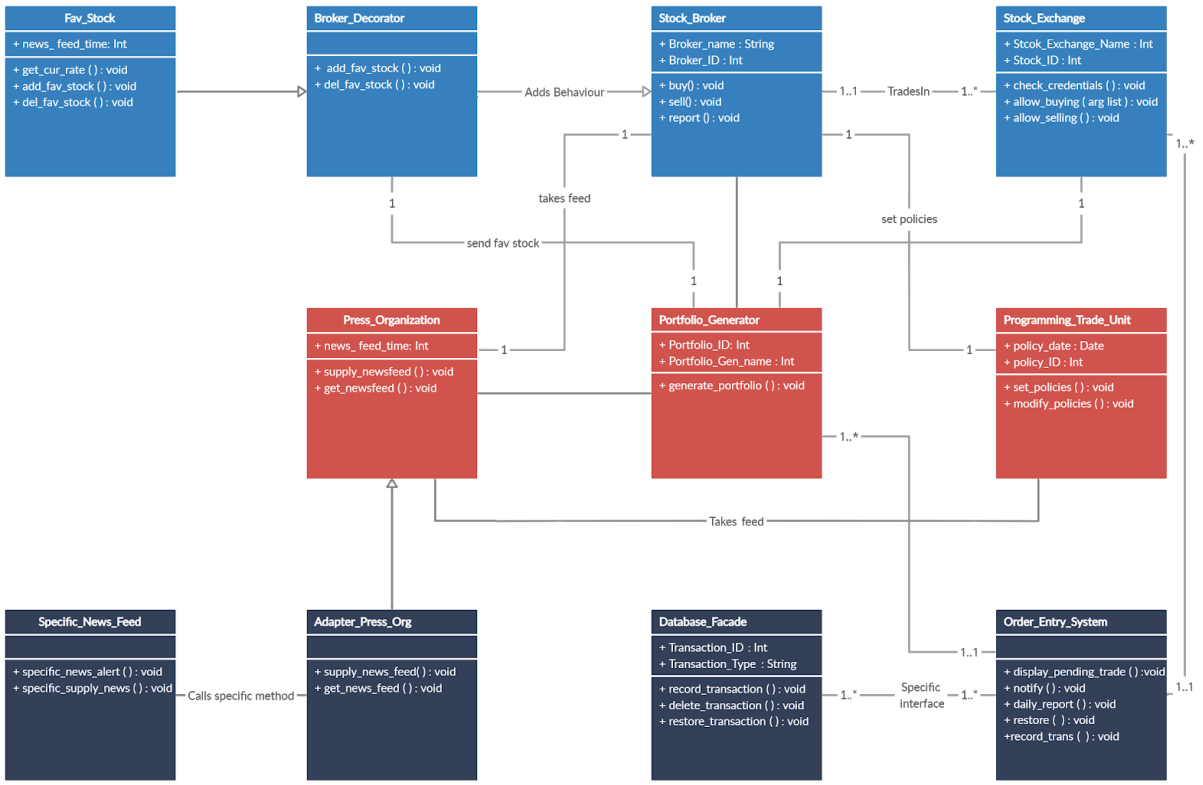
6. Creating a Gantt Chart for Scheduling



CHAPTER 3 :

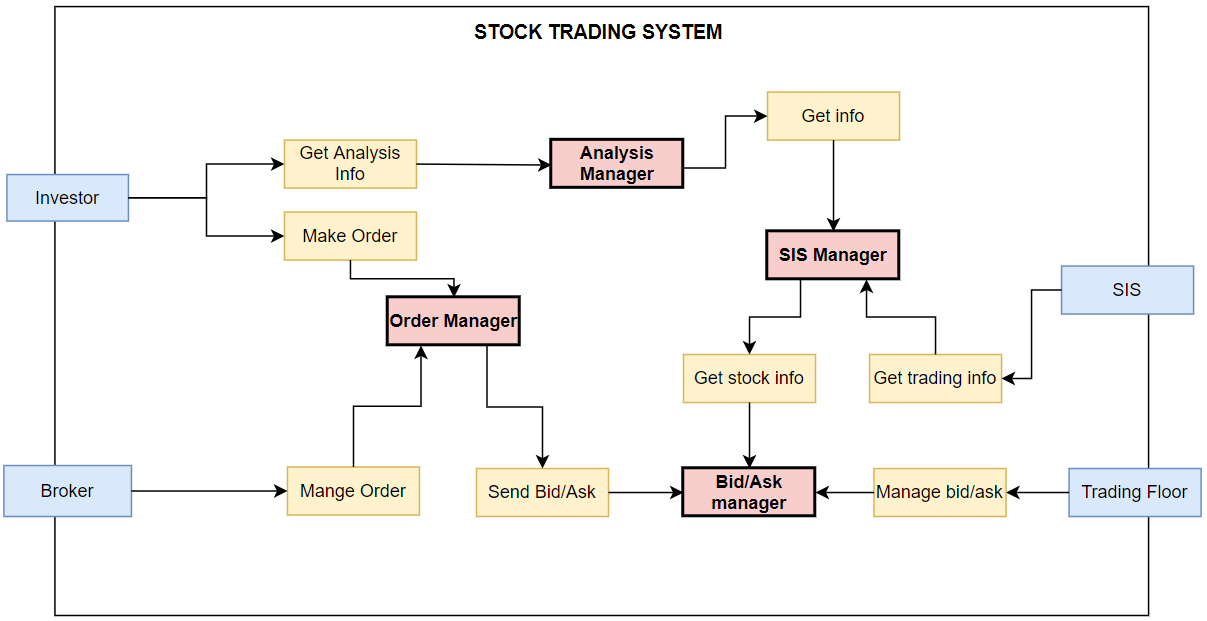
Design Diagrams

Class Diagram:



This class diagram describes the structure of a stock market trading system by showing the system's classes, their attributes, operations and relationships between the objects.
There are super classes like Fav_Stock (favourable stocks), Broker_Decorator, Stock_Broker and sub classes like Portfolio_Generator, Order_Entry_System etc.
The behavioral feature added to Stock_Broker defines the way in which the objects of the class can interact.
As the Broker_Decorator receives a favourable stock, a portfolio is generated based on the objects present in Stock_Broker.
The favoured stocks are added to the Stock_Exchange class.
The Database_Facade class stores transaction details based on the Order_Entry_System class.
Specific_News_Feed and Adapter_Press_Org classes supple newsfeed.
Stock_Broker and Programming_Trade_Unit classes take feed from Press_Organisation. Class Stock_Broker inturn sets policies and all these classes play a role in portfolio generation.

System Architecture:



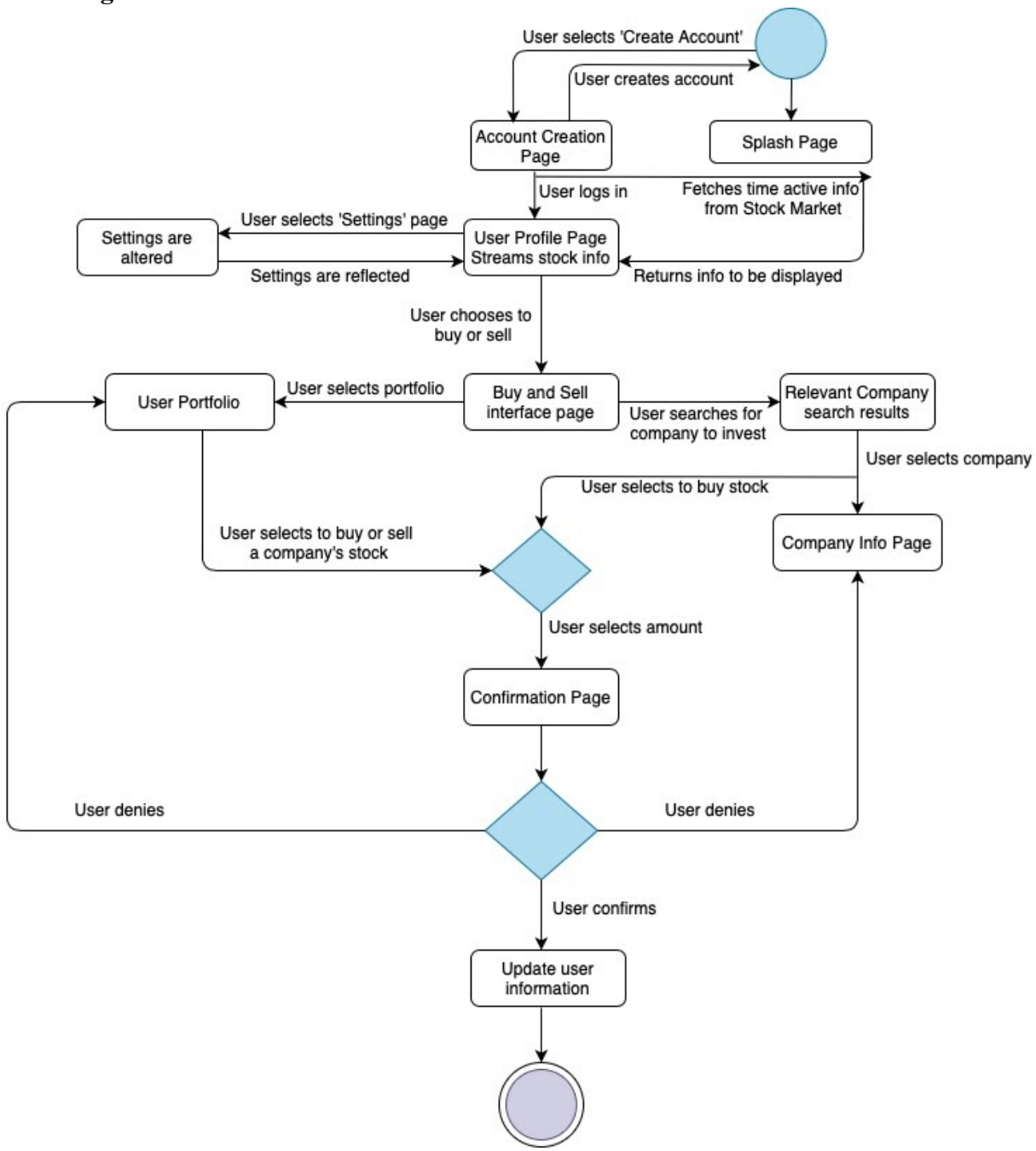
The investor decides to buy or sell a stock according to quantity, price, time and negotiation tolerance, its current strategy and information from SIS.
To do so, the investor sends a bid/ask order including name of the stock, quantity, price range, and time to the Broker, to apply it to the decision that the Broker made.

Broker makes bid(s)/ask(s) to Trading Floor according to parameters of incoming order from Investor and returns a positive or negative acknowledgment to Investor.

SIS(System Information System) fetches relevant information from the Internet or other media and instant stock data from Trading Floor and then places them in a database.

Trading Floor places all orders to a table according to price and time priority rules. It maps the bids and asks offers, performs transactions and updates the table. Finally, Trading Floor waits for new bids/asks.

State Diagram:

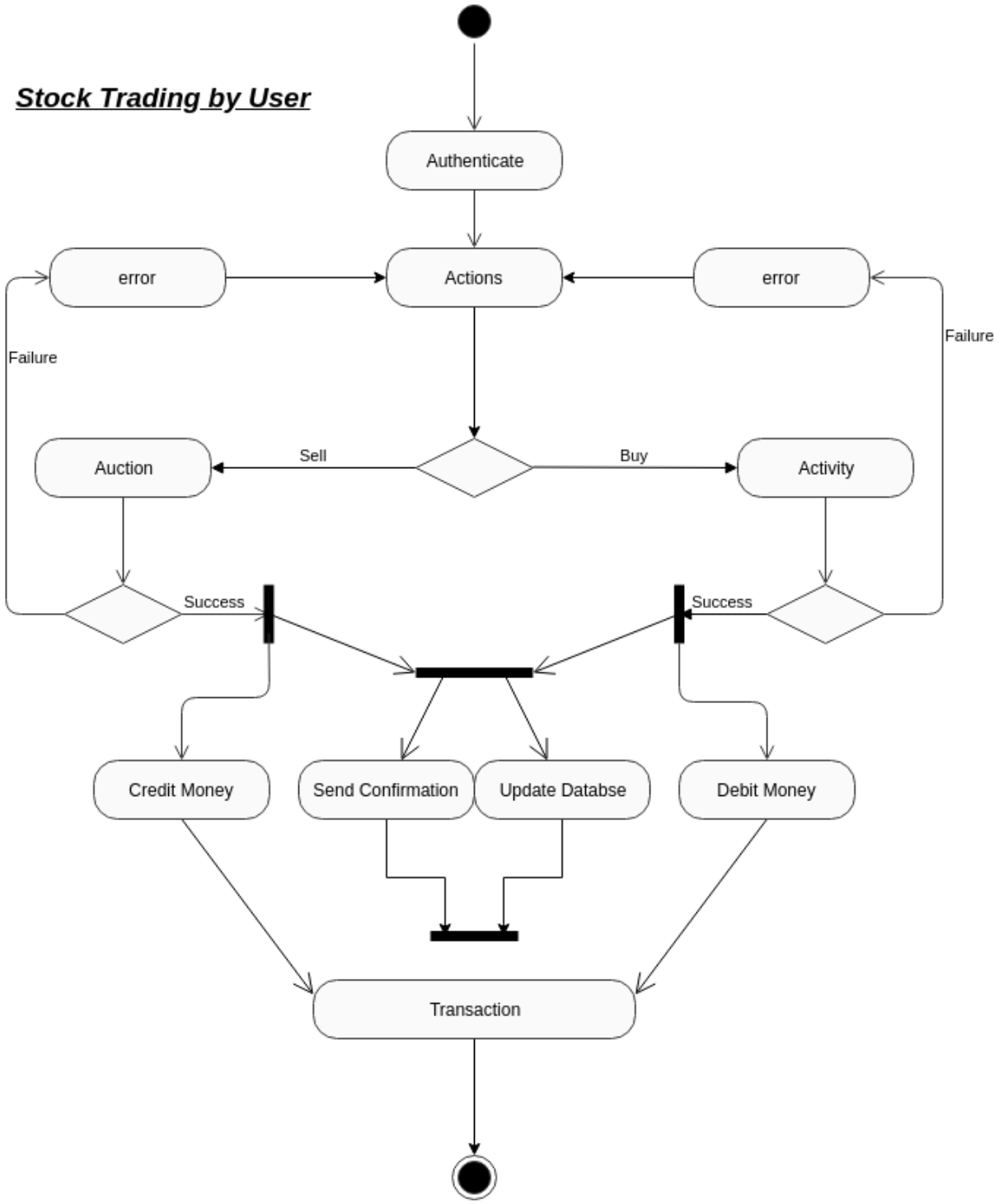


In the state diagram we can see that the first state is the user account creation state and then it is followed by a user login page the user logs in into the website and into the user's homepage over there. And as the user logs in all the info related to the user is fetched from the database and is displayed when the user clicks on the profile page.

Next the user has an option to buy, sell and view transactions related to the stocks all the related pages to the above mentioned activities have to be updated continuously as the stock value updation is real time and in the transaction and view history page we see all the previous transactions related to the user along with its generated ids. Finally the user has an option to logout. After which none of the users info can be accessed until he is logged back in again.

USE-CASES:

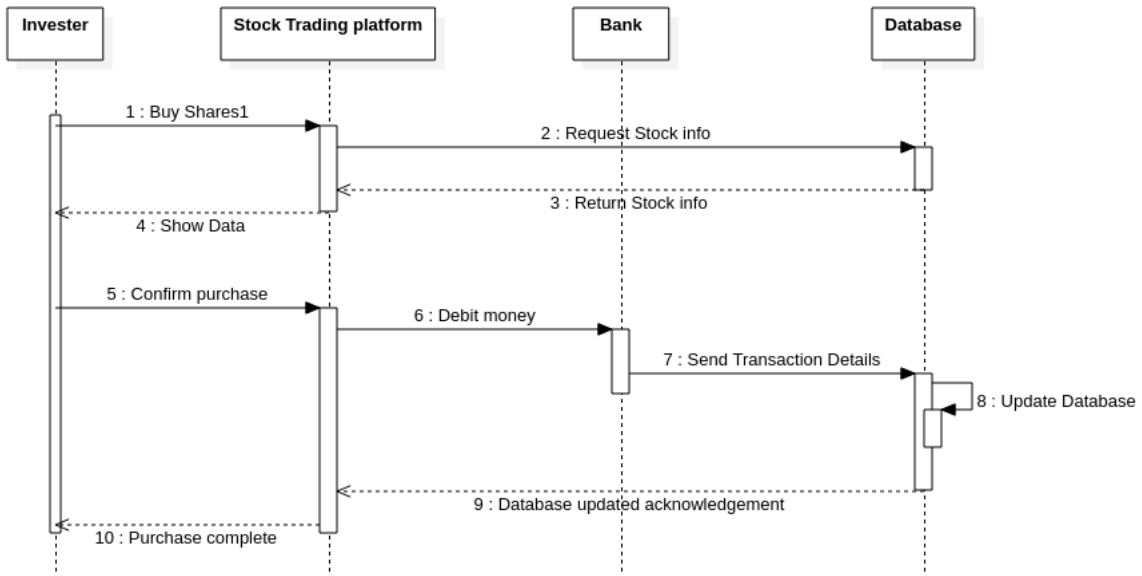
Stock Trading by user - Buy and Sell:
Activity diagram:

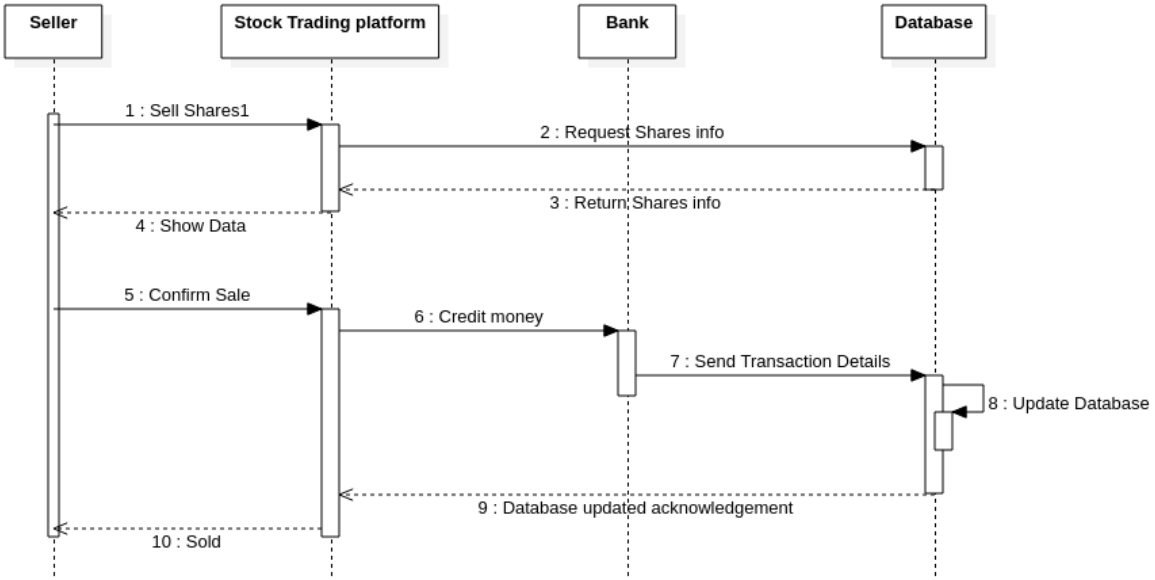


For this use case the user has to be registered with all the proper information and must be logged in into the stock trading platform and next the user will be logged in into his home page where he will have an option to buy and sell the stocks.

If the user chooses to buy stocks then a list of available stocks must be displayed. and must be updated continuously and the price of the stocks must keep varying and to simulate the real time flow of data. But if the user chooses to buy stocks then the user owned stocks must be displayed so that the user can decide on which stocks to sell.
And this feature must support multiple user's buy and sell features so that the integrity and availability is mained for the customers.

Sequence diagram:



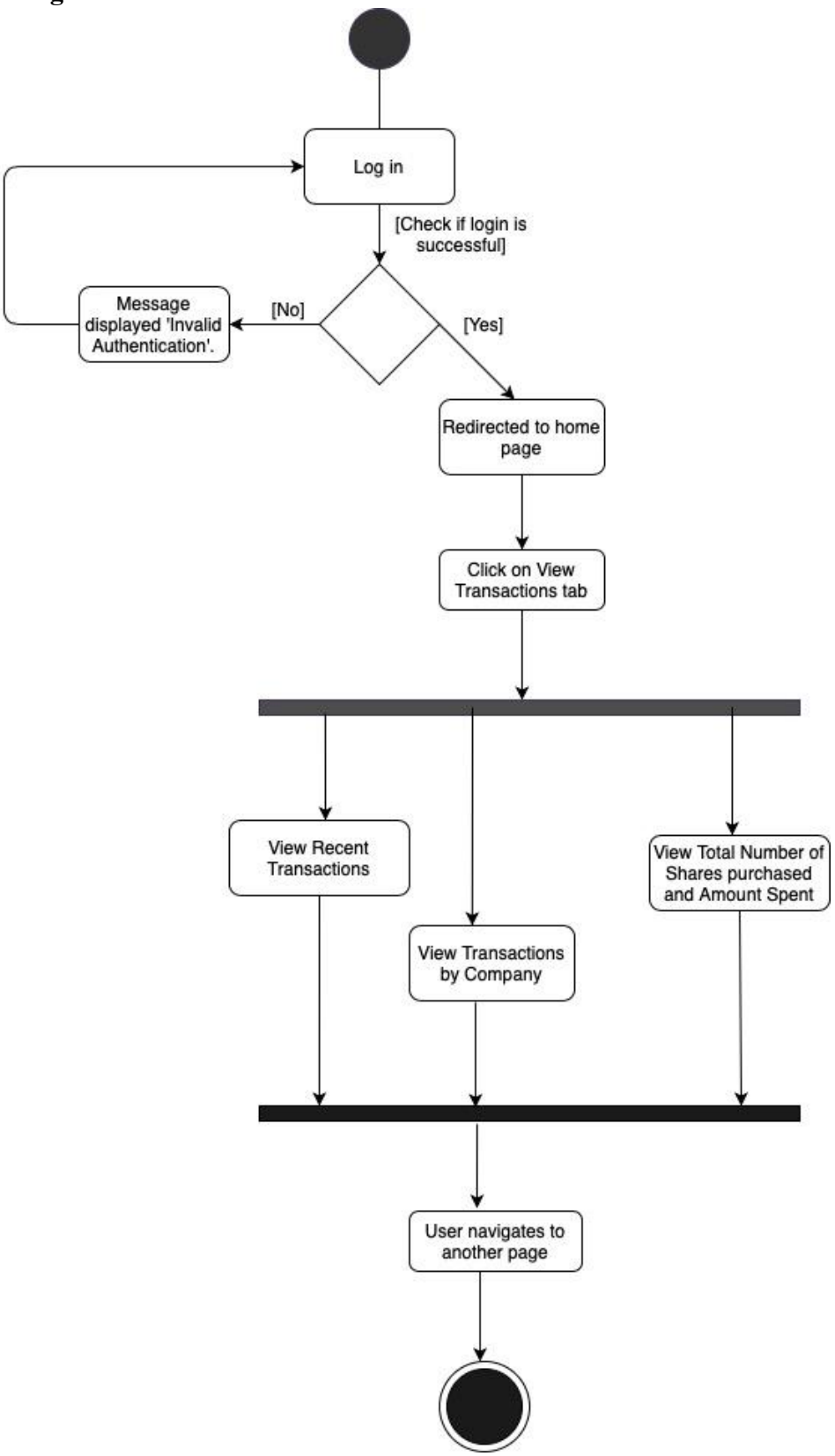


For Buying the stock the sequence flows like: Logging in the user ,then the user has to go to the buy page and as a response to this the website must query the database and return the required self updating webpage that continuously displays the cost of different available stocks. Then when the user clicks on buy stock the application must again update the database and update the available and the user owned stocks webpages.

For Selling the stock the sequence flows like: Logging in the user ,then the user has to go to the View/Sell page and as a response to this the website must query the database and return the required self updating webpage that continuously displays the cost of different owned stocks. Then when the user clicks on sell stock the application must again update the database and update the available and the user owned stocks webpages.

View Transactions:

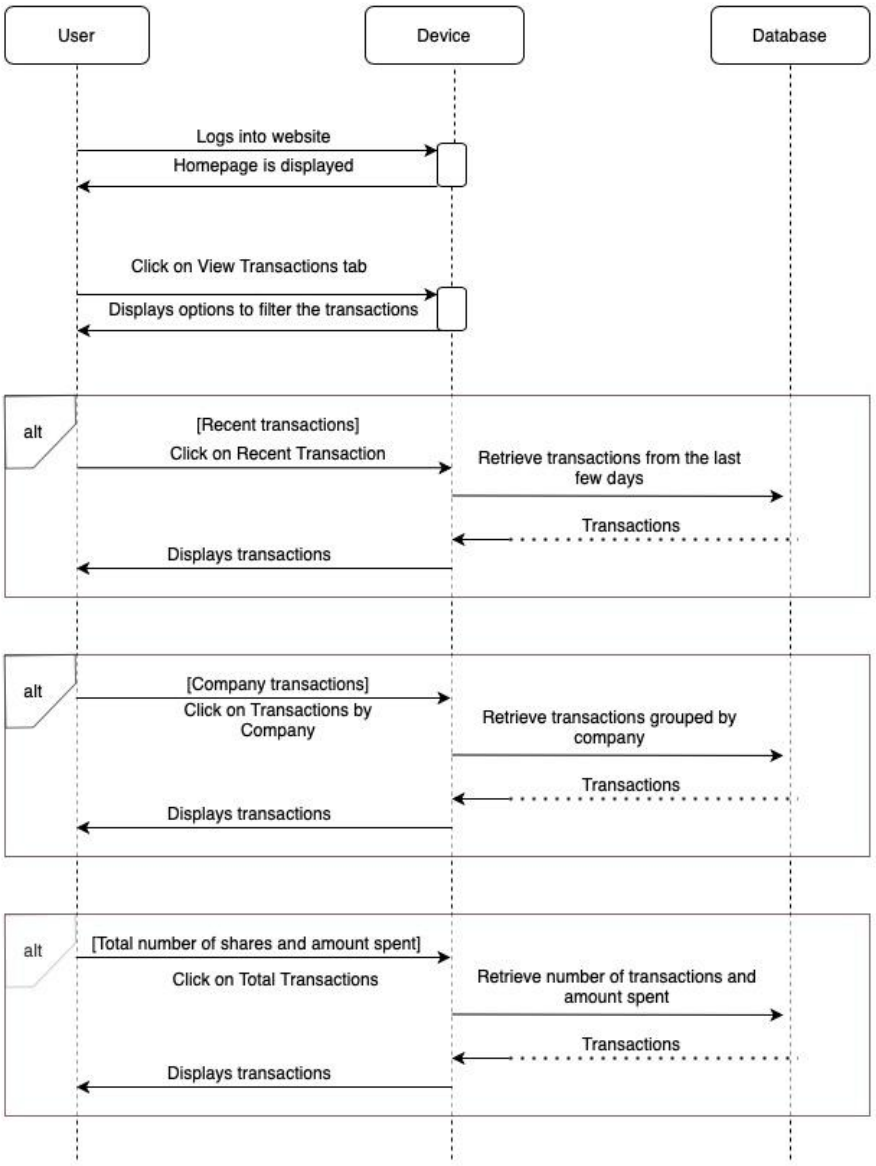
Activity Diagram:



This use case deals with the user viewing their entire transaction history, with the most recent transaction being shown first and then displayed in reverse chronological order.
For this the user must be logged in successfully. When the user navigates to the Transaction History page, all the transactions are displayed in reverse chronological order. When the user navigates to the Company-wise Transactions page, transactions are grouped based on company. Summary statistics such as number of shares bought and sold per company and amount spent and earned from each company is also displayed.

The user can navigate to other pages from these pages using the button at the bottom of the page.

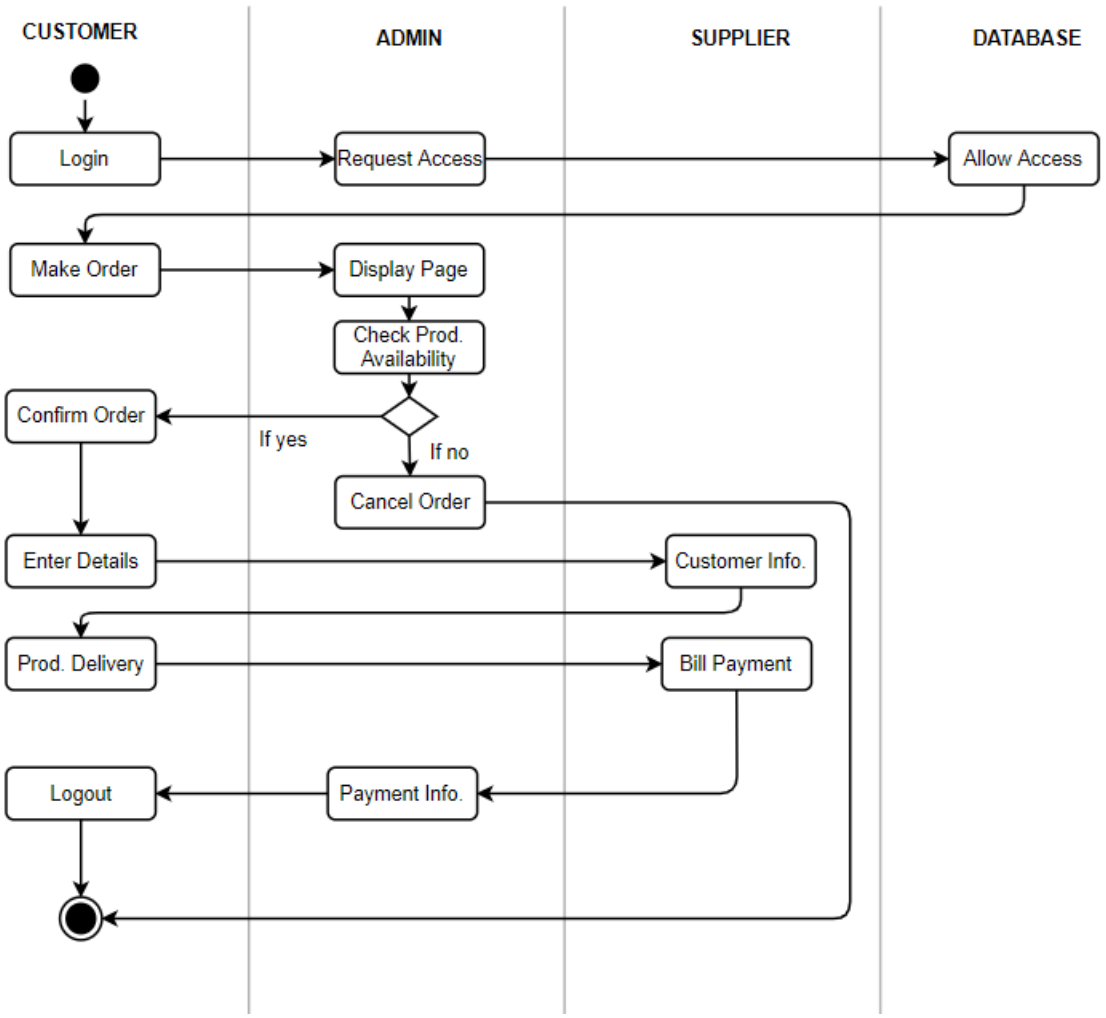
Sequence Diagram:



The sequence flows like this: When the user clicks on View History, the transactions of the user sorted in descending order by date must be fetched from the database. These are then displayed on the page.

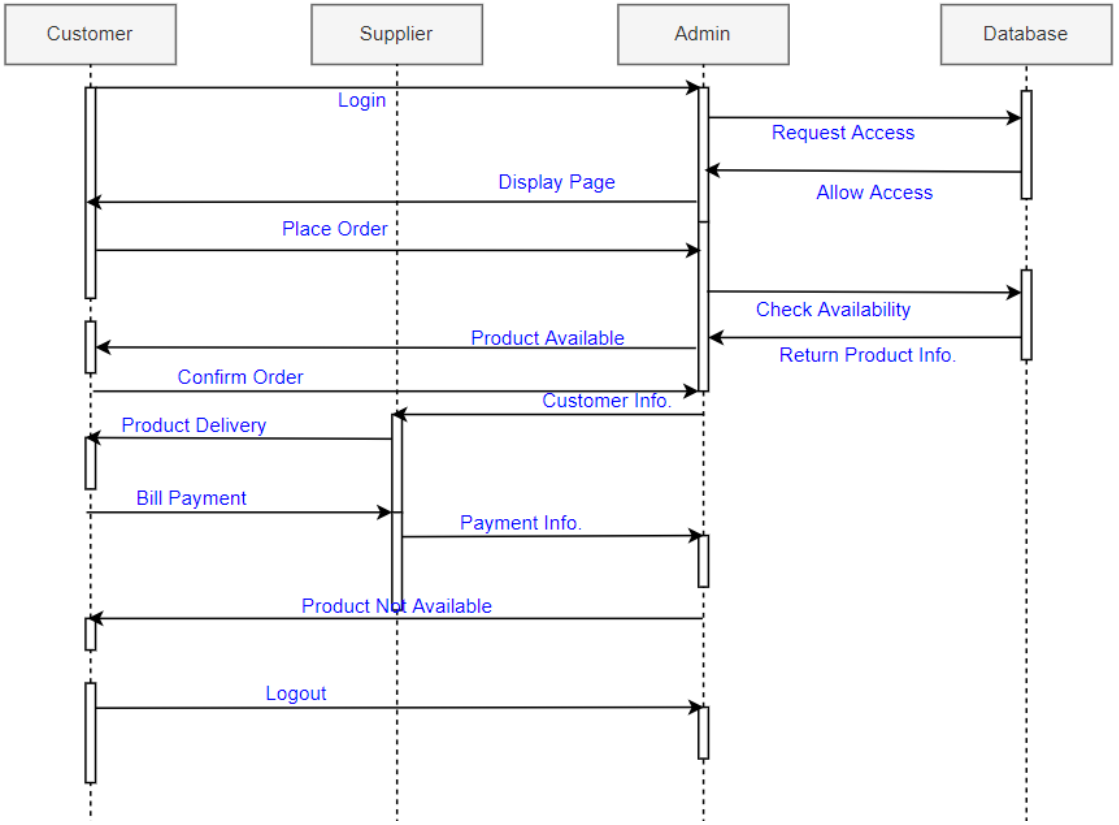
When the user clicks on View Transactions, the database should query the user's transaction details and group them based on company and these are then ordered based on the most recent date. Additional details such as company stock ID and company name are joined to these transactions using further database querying. Aggregation queries are run to determine the summary statistics such as number of shares and amount per company. These are then displayed on the Company-wise Transactions page.

Stock Management By Admin
Activity Diagram :



This use case shows how stocks are managed by the admin. The customer is logged in successfully and then given access from the database by the admin upon selecting favoured stocks.
Once the customer makes an order, the admin checks the availability of the favoured stocks. The order is either confirmed or canceled by the admin based on the availability of the stock. If the stocks are not available and the order is canceled, the customer is redirected to the logout page.
If confirmed, the customer has to then enter his/her details. This customer information is provided to the supplier who then supervises the product delivery and payment.
Once payment info is entered, the customer is then redirected to the logout page.

Sequence Diagram :



The sequence of this use case flows like this : Once the customer is logged in successfully, the database is queried and access is granted to the user, the customer is directed to the display page. Display page shows all the costs of all the stocks and the customer proceeds to buy/make an order for the favoured stocks.
After customer places an order, availability of the favoured stocks is checked by querying the database.
If available, stock information is returned for the customer to confirm the order. Customer is directed to a page to enter customer information. Another page displays the customer's payment information.
If stock is not available, the customer is redirected to the logout page.

CHAPTER 4 :

Test Cases

Use-case:
User Registration and login, Stock Buy and sell

Author:
N Sanketh Reddy

Tests:
2 System Test cases
2 Integration Test cases
15 Unit Test cases

Test Case ID	Name of Module	Test case description	Pre-conditions	Test Steps	Test data	Expected Results	Actual Result	Test Result
ST-01	NA	Interoperability - dependency testing	To run the application, flask and python must be installed.	NA	1) python3 -V 2)python3 import flask flask.__version__	The python version must be greater than 3.6, and the flask version must be greater than 1.0.0	Python version: 3.6.3 Flask version:1.0.2	PASS
ST-02	Database	Security testing	Other users if the server must not be able to alter the mysql database	NA	For prevent a normal user to alter the database we have to use to use the root user's mysql server	mysql -u root -p Then enter the root user's password	By using the root user's mysql server the guest users will not be able to alter the database. So the integrity of the database is maintained	PASS
UT-01	Registration	Register new user with proper credentials	Should be on the registration page of the website	Type in proper details for registration	username: skete1, Phone: 9876827367, DOB:1999-12-17, Address:10th cross Srinager, Password: 99	Should intimate user that they are successfully registered	Prints "You have successfully registered"	PASS
UT-02	Registration	Register new user with an existing username	Should be on the registration page of the website	Type in an existing user's username in the form along with other details	username: skete, Phone: 9876827367, DOB:1999-12-17, Address:10th cross Srinager, Password: 99	Should Intimate user that username is already used	Prints "Account already exists in this username!"	PASS
UT-03	Registration	Register new user with unaccepted username	Should be on the registration page of the website	Type an username with a special character or any other character other than letters and numbers	username: skete^1**$, Phone: 9876827367, DOB:1999-12-17, Address:10th cross Srinager, Password: 99	Should Intimate user that the username is not acceptable	Prints "Username must contain only characters and numbers!"	PASS
UT-04	Registration	Register new user with	Should be on the	Type an improper	username: skete1,	Should Intimate user	Prints "Phone Number format	PASS

		unaccepted phone number	registration page of the website	phone number	Phone: 0876827367, DOB:1999-12-17, Address:10th cross Srinager, Password: 99	that the phone number is not acceptable	is wrong”	
UT-05	Registration	Register new user with unaccepted DOB	Should be on the registration page of the website	Enter a DOB that is not at least 18 years from the date of account creation	username: skete1, Phone: 0876827367, DOB:2009-12-17, Address:10th cross Srinager, Password: 99	Should Intimate user that the that they are too young to register	Prints “You are not old enough to Register!”	PASS
UT-06	Registration	Trying to register with incomplete information	Should be on the registration page of the website	Enter some details without entering all the details of form and click on submit	username: skete1, Phone: 8765827367, DOB: ,Address:10th cross Srinager, Password: 99	Should intimate the user that the empty fields must be filled	Pops up a alert box next to the empty textbox with ”Please fill this field”	PASS
UT-07	Login	Trying to login as an Existing user	Should be on the login page of the website	Enter proper credentials of an existing user in the website	username:skete, password:99	Should intimate the user and redirect to the home page	Shows the user that he is successfully logged in by redirecting the user to home page with his session information	PASS
UT-08	Login	Trying to login with incorrect username or password	Should be on the login page of the website	Enter wrong username/ password	Username: sky, password:99	Should not login the user	Prints “Incorrect username/password!”	PASS
UT-09	Login	Security attack - SQL injection	Should be on the login page of the website	Enter “; OR “1==1” as password	Username: admin Password: “; OR “1==1”	Should not login the user	Prints “Incorrect username/password!”	PASS
IT-01	Home/Buy/ Sell/Profile	Reload page	Should be on a session activated page of a user	Click on the reload button of the browser or F5	NA	Should not terminate the user session and still stay in the same page	Reloads the same page without terminating the session	PASS
IT-02	Home/Buy/ Sell/Profile	Movement between different pages of the website	Should be on a session activated page of a user	Click on the Home/Buy/Sell/Profile button on the page	NA	Should go to the profile/ home page and load the contents on the page	Loads the contents of the required page	PASS
UT-09	Realtime nameture of stock	The stock prices should not be constant	NA	Start the website hosting server	NA	The stock prices should be fluctuating	A new thread is spawned in the application for fluctuating the stock prices to simulate the real stock price fluctuation	PASS
UT-10	Buy/Sell	The updated stock prices must be	Should be on the Buy/Sell page	NA	NA	The web page must reload	There is hot reload function implemented in	PASS

		displayed in the webpage				continuously so that the current prices of the stock is displayed	the Buy and Sell page so that he stock price is updated real time	
UT-11	Buy	Must display all the available stocks	Should be on the buy page of the website	NA	NA	Must display only the available stocks	If a user buys a stock it is sent to the View/Sell page and removed from the table in the Buy page	PASS
UT-12	Sell	Must display all the bought stocks	Should be on the View/sell page	NA	NA	Must display only the stocks bought by the user	If the user sells a stock it is removed from the buy sell page and again appended to the table in the Buy page so that other users can buy	PASS
UT-13	Buy	When a user buys a stock from the company and if the company has more than 1 stock	Should be on the buy page of the website	Click on “Buy” button next to the available company stocks that has more than 1 remaining stocks	NA	The company’s available stocks must reduce by 1	If Tesla has 7 stocks and if the user clicks on “Buy” one time then the available company stocks reduces to 6	PASS
UT-14	Buy	When a user buys a stock from the company which has only 1 stock remaining	Should be on the buy page of the website and there should be at least 1 company with exactly 1 stock remaining.	Click on “Buy” button next to the available company stocks that has exactly 1 remaining stock	NA	The company’s stocks must be removed from the buy page and available stocks	If google has only 1 stock remaining and if the user clicks on “Buy” then the google row is removed from the /buy webpage	PASS
UT-15	Application must support multiple users at the same time	Multiple user must be able to buy and sell stocks parallely	Multiple sessions of the users must be logged in and must be actively participating in the buy and sell options	NA	NA	Must display the updated stocks table continuously for all the users	All the users are connected to a single main database so the Integrity is always maintained and there will be no false copies across the users	PASS

Use-case:
View Transaction History and Grouped Transaction, managing user balance

Author:
Aditi Ahuja

Tests
1 System Test case
9 Unit Test cases

2 Integration Test cases

Test Case ID	Name of Module	Test case description	Pre-requisites	Test Steps	Test Input	Expected Results	Observed Result	Test Result
ST-03	NA - applicable to the overall system	Interoperability - dependency testing	To run the application, MySQL server, flask and python must be installed with sufficiently recent versions.	NA	1.mysql -V 2. python3 -V 3.python3 import flask flask.__version__	The MySQL version must be greater than 12, the python version must be greater than 3.6, and the flask version must be greater than 1.0.0	MySQL version: 14.14 Python version: 3.6.3 Flask version:1.0.2	PASS
UT-16	Transaction History	User must be able to view entire transaction history	Users must be logged in and on the Transactions History page.	NA	NA	Should display all transactions performed by the user in decreasing order by date	Displays all transactions sorted in decreasing order by date	PASS
UT-17	View Company-wise Transactions	Show all transactions grouped by company	User must be logged in and on the Company-wise transactions page.	NA	NA	Should display all transactions grouped by company	Displays all transactions groped by company chronologically reversed.	PASS
UT-18	Display low account balance message	Message stating Low Balance to be displayed if a user does not have enough account balance	User must be logged in and attempting to buy stocks from the Buy page	Click on Buy next to an available stock	NA	A message stating that the user has a low account balance must be displayed	An alert stating 'Your account balance is low' is displayed	PASS
UT-19	Update account balance after buy	Each buy transaction must deduct from the user's account balance	User must be logged in and must have performed a buy transaction.	Click on Buy next to an available stock	NA	Update the user balance.	The user balance is updated in the database.	PASS
UT-20	Update account balance after sell	Each sell transaction must add to the user's account balance	User must be logged in and must have performed a sell transaction.	Click on Sell next to an available stock	NA	Update the user balance	The user balance is updated in the database.	PASS
UT-21	Summarise amount spent and earned per company	The summary of company-wise transactions based on amount spent and earned should be displayed.	User must be logged in and on the company-wise transactions page.	NA	NA	Display buy and sell amount for each company	Buy and sell amounts are displayed in the summary details.	PASS
UT-22	Summarise number of transactions per company	The summary of company-wise transactions based on number of buy and sell transactions must be displayed.	User must be logged in and on the company-wise transactions page.	NA	NA	Display number of buy and sell transactions.	Number of buy and sell transactions are displayed in the summary details.	PASS

IT-03	Navigation between the Transaction History and Company-wise transactions pages.	Should be able to navigate between pages using session-specific user details	Users must be logged in and attempt to navigate between these two pages.	Click on either of these two pages.	NA	Should not terminate an existing session while navigating.	Navigates to another page without session termination.	PASS
IT-04	Reload History/Company-wise transactions pages	Must reload the page without terminating an existing session, based on the current user.	Users should be on the History/Company-wise transactions pages.	Reload the page according to system specifications.	NA	Should not terminate an existing session while reloading.	Reloads the page without session termination.	PASS
UT-25	Users require a minimum balance to start trading	Requires a sign up amount of 5000.0 at the time of registration.	NA	NA	NA	The initial minimum amount should reflect in the database.	The database is updated with the initial amount.	PASS
UT-26	Users are required to verify identity before Buy transactions	Only authorized users should be able to perform buy transactions.	User must be logged in.	Enter password for verification.	Card password	Authorized users only should be redirected to Buy.	Authorised users are shown a message Correct password and redirected.	PASS

Use case :
Stock Management by Admin

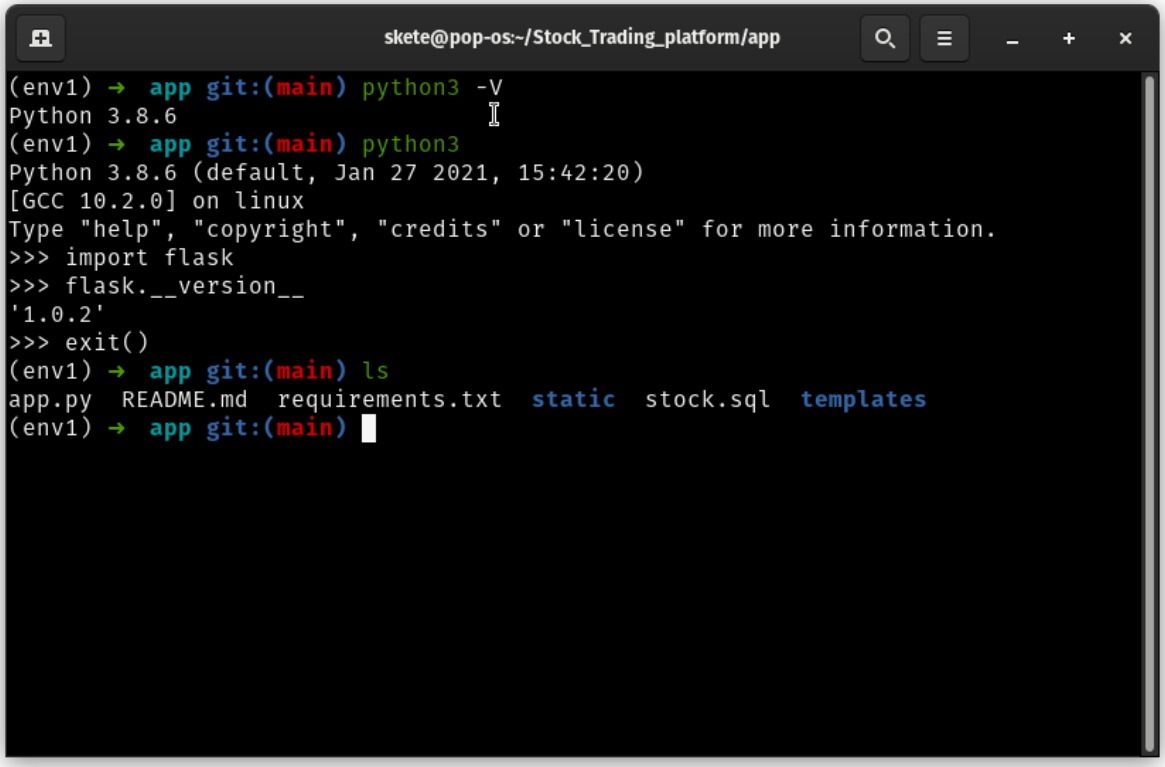
Author:
Bhavana Madhuri

Test Case ID	Name of Module	Test case description	Pre-requisites	Test Steps	Test Input	Expected Results	Observed Result	Test Result
ST-04	NA (applicable to the overall system)	Interoperability - dependency testing	To run the application, MySQL server, flask and python must be installed with sufficiently recent versions.	NA	1.mysql -V 2. python3 -V 3.python3 import flask flask.__version__	The MySQL version must be greater than 12, the python version must be greater than 3.6, and the flask version must be greater than 1.0.0	MySQL version: 14.14 Python version: 3.5.2 Flask version:1.0.2	PASS
IT-05	Available	Should be able to	User should have	NA	NA	Shows a list	Database is	PASS

	stocks	view all favoured, available stocks	selected page, “Available Stocks”			of available stocks or stocks that could be potential favoured stocks.	queried and available stocks shown.	

CHAPTER 5:

Screenshots



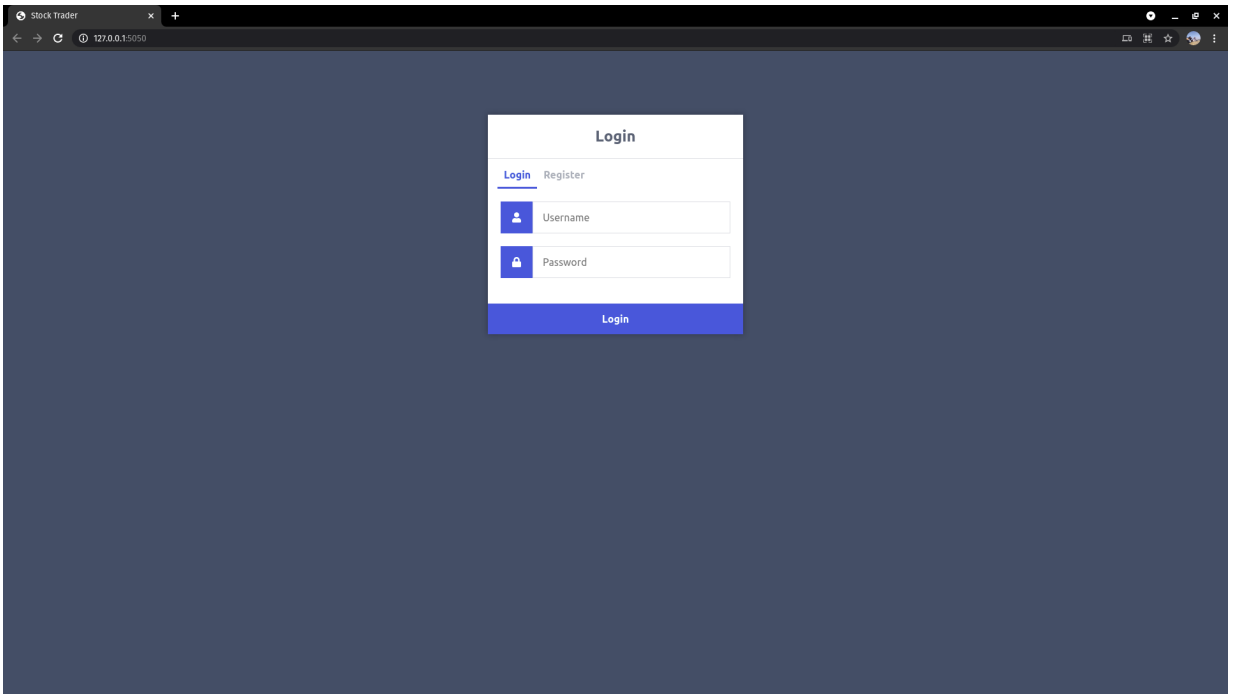
- These are the requirements for running the application.

```
(env1) ➜  app git:(main) ✗ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 4005
Server version: 8.0.23-0ubuntu0.20.10.1 (Ubuntu)

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> source stock.sql;
Query OK, 6 rows affected (0.02 sec)

Query OK, 1 row affected (0.00 sec)

Database changed
Query OK, 0 rows affected, 2 warnings (0.01 sec)

Query OK, 0 rows affected, 1 warning (0.01 sec)

Query OK, 0 rows affected, 2 warnings (0.01 sec)

Query OK, 0 rows affected, 2 warnings (0.02 sec)

Query OK, 0 rows affected, 3 warnings (0.01 sec)

Query OK, 0 rows affected (0.01 sec)

Query OK, 1 row affected (0.00 sec)

Query OK, 1 row affected (0.00 sec)

Query OK, 1 row affected (0.00 sec)

Query OK, 1 row affected (0.00 sec)

Query OK, 1 row affected (0.00 sec)

Query OK, 1 row affected (0.00 sec)

Query OK, 1 row affected (0.01 sec)

Query OK, 1 row affected (0.00 sec)

mysql> quit
Bye
(env1) ➜  app git:(main) ✗
```

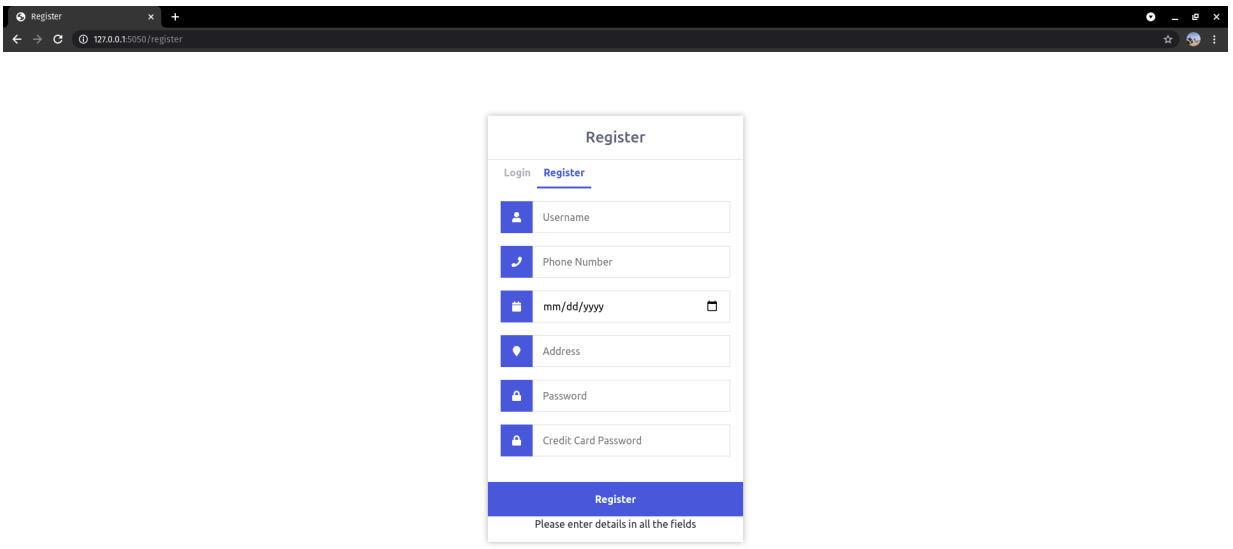
- Logging in into the root server for mysql and running the stock.sql file that loads all the necessary table information.

```
(env1) ➜  app git:(main) ✗ python3 app.py
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://127.0.0.1:5050/ (Press CTRL+C to quit)
 * Restarting with inotify reloader
 * Debugger is active!
 * Debugger PIN: 149-505-374
```

- Starting the flask application

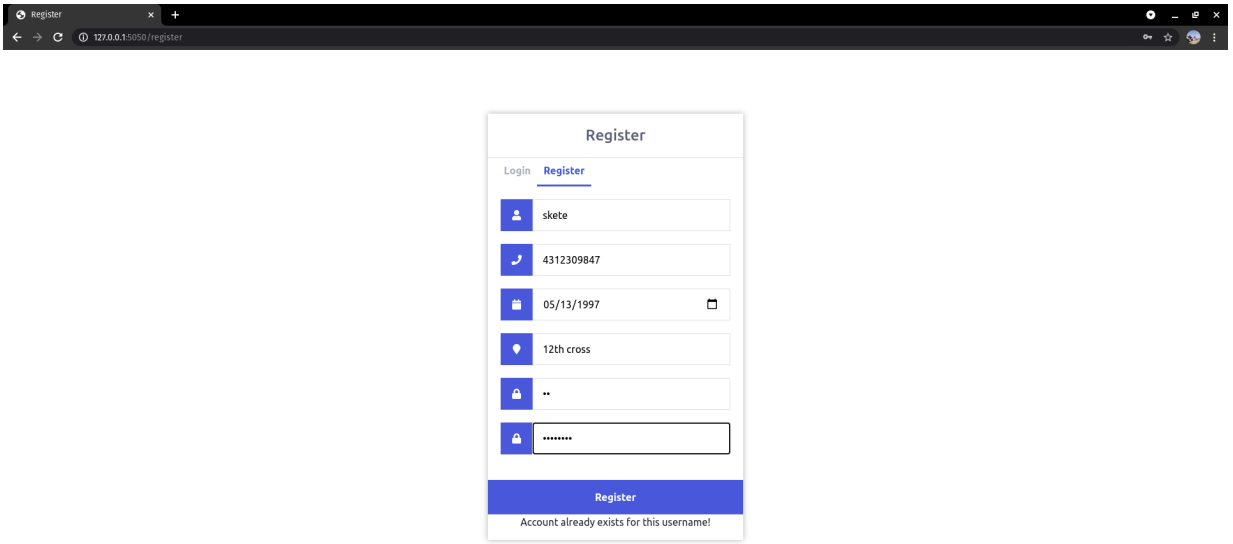


- By Ctrl+clicking on the link provided on the terminal after starting the application server we open the application on the default browser and we are greeted with the login page.



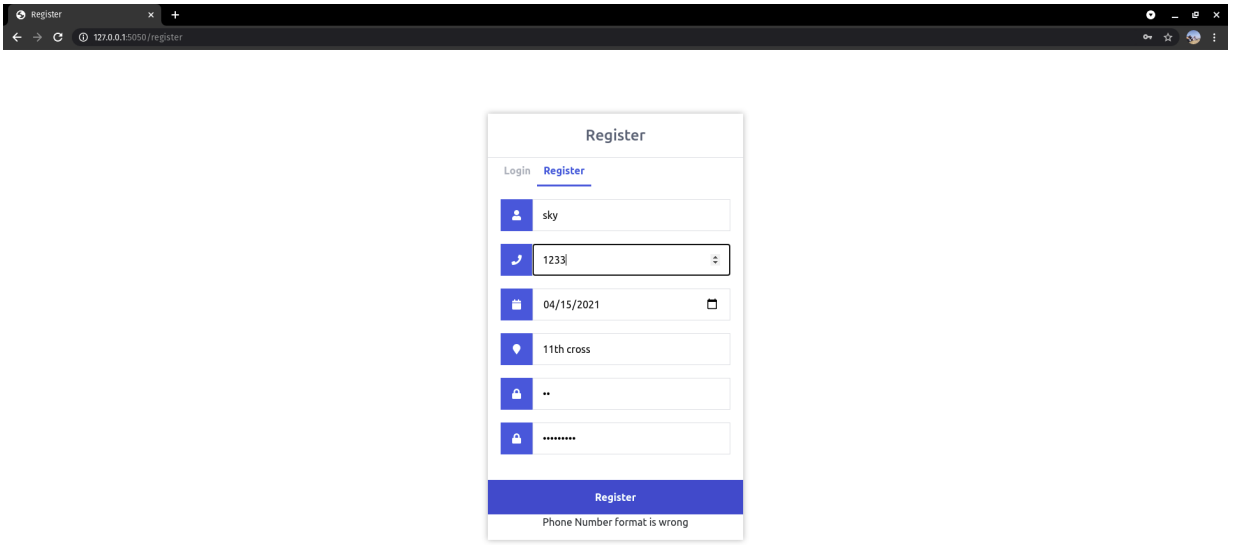
- By clicking on the register option we open the register page

---

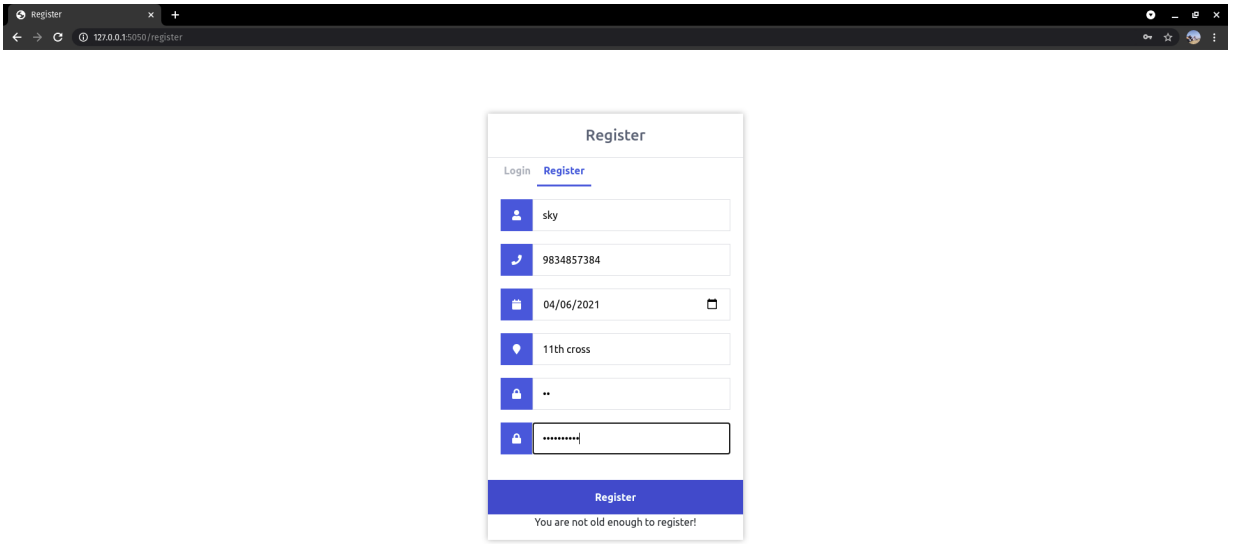


- If we try to register an account with an already existing username, an error message is prompted

---

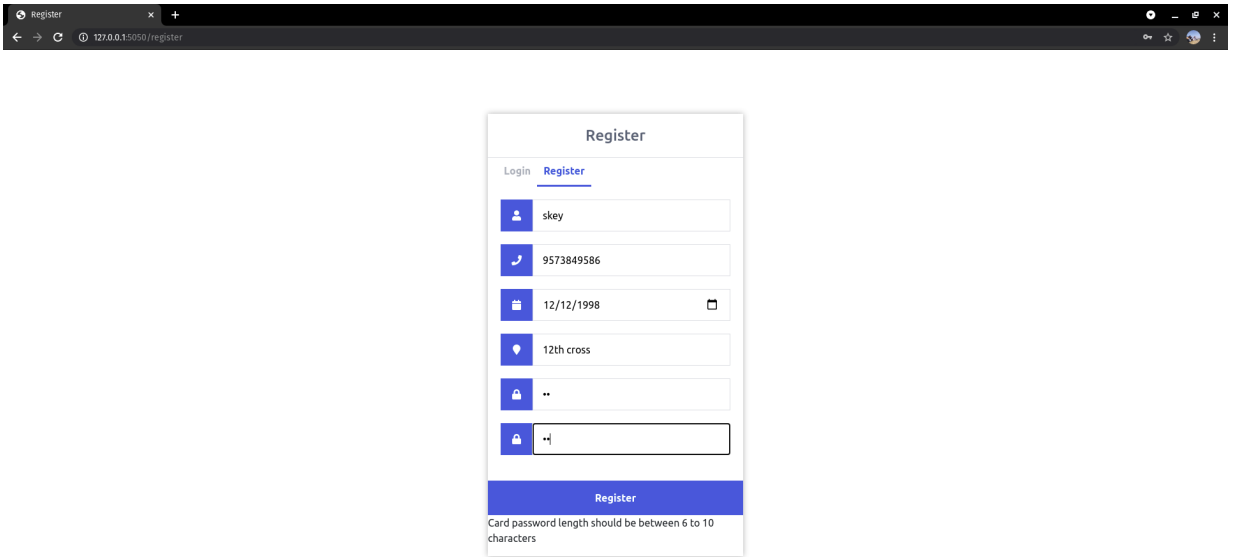


- If we fill in a wrong phone number then we get an error.



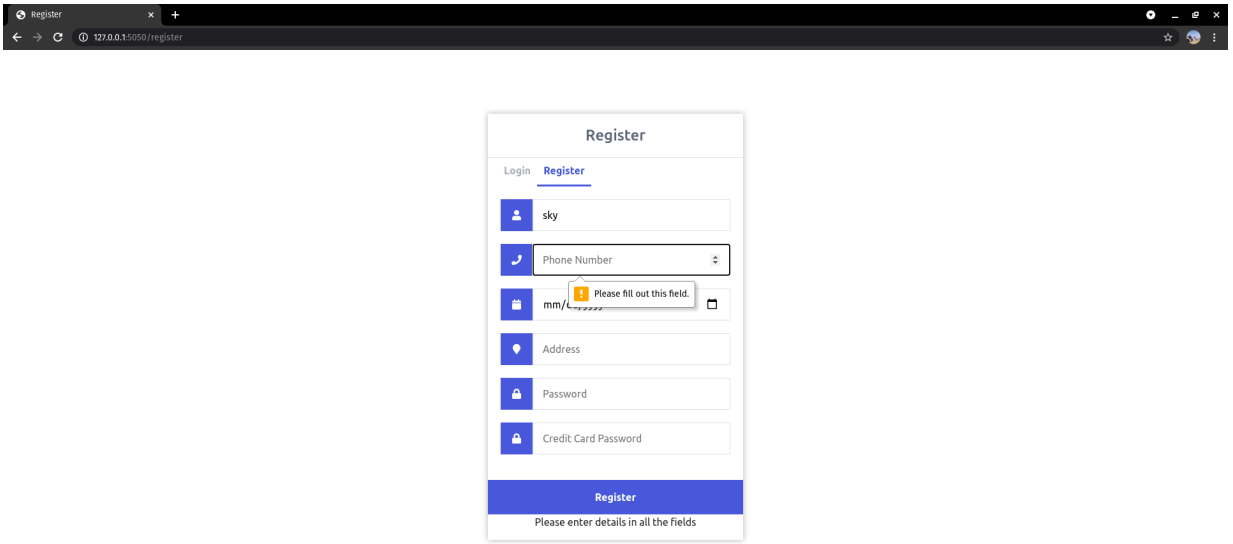
- When the user enters his DOB we check if the person is at least 18 y/o. If they are not we show the appropriate message and not register the account

---



- While filling the card password, we check if the password is between 6 to 10 characters long. If it's not then we trigger a error message

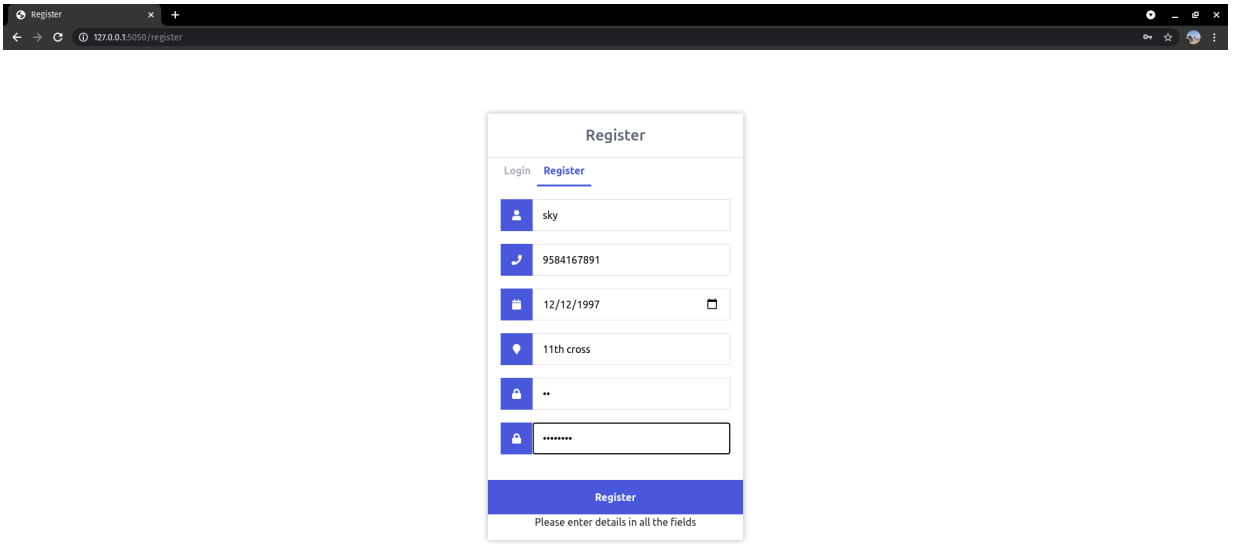
---

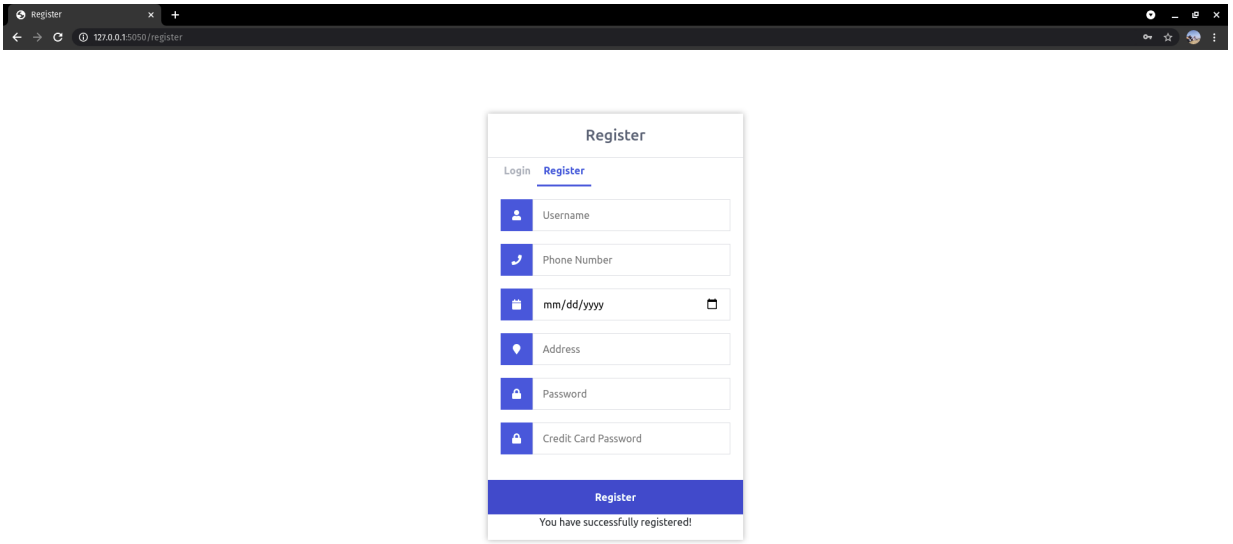


- If any of the fields are left empty we pop up a notification near the text box to intimate the user to fill the details

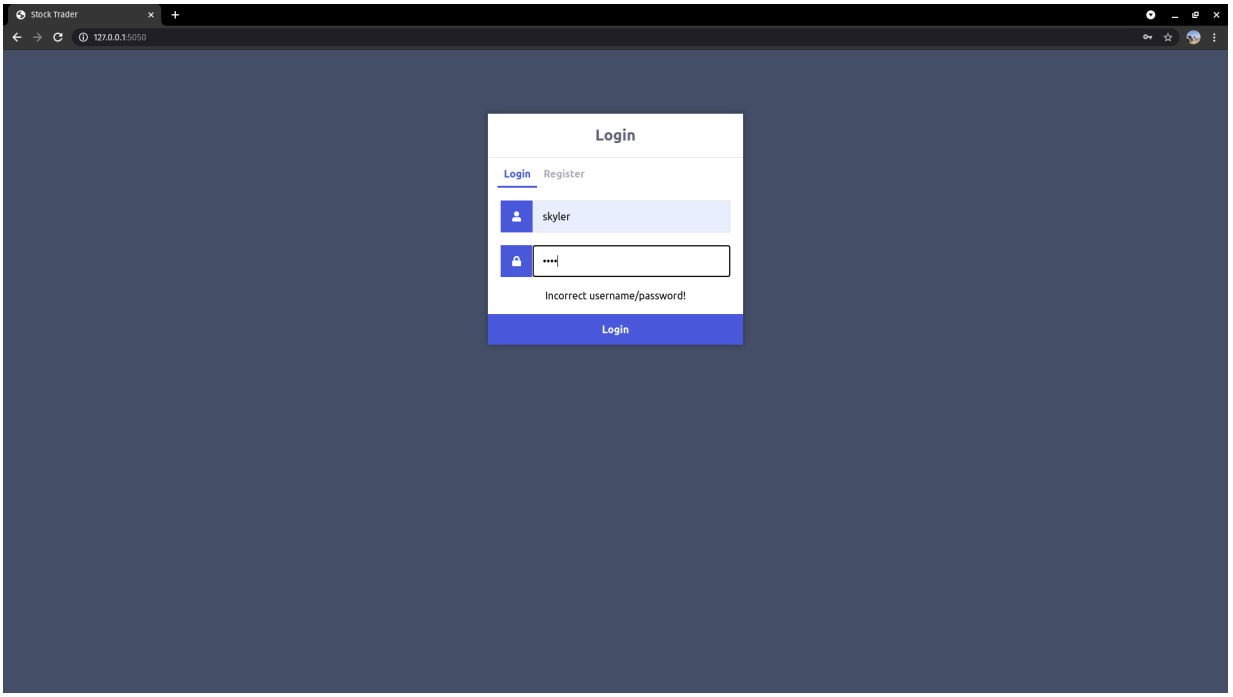
---

- If all the required fields are filled with proper information then the user is registered on the website

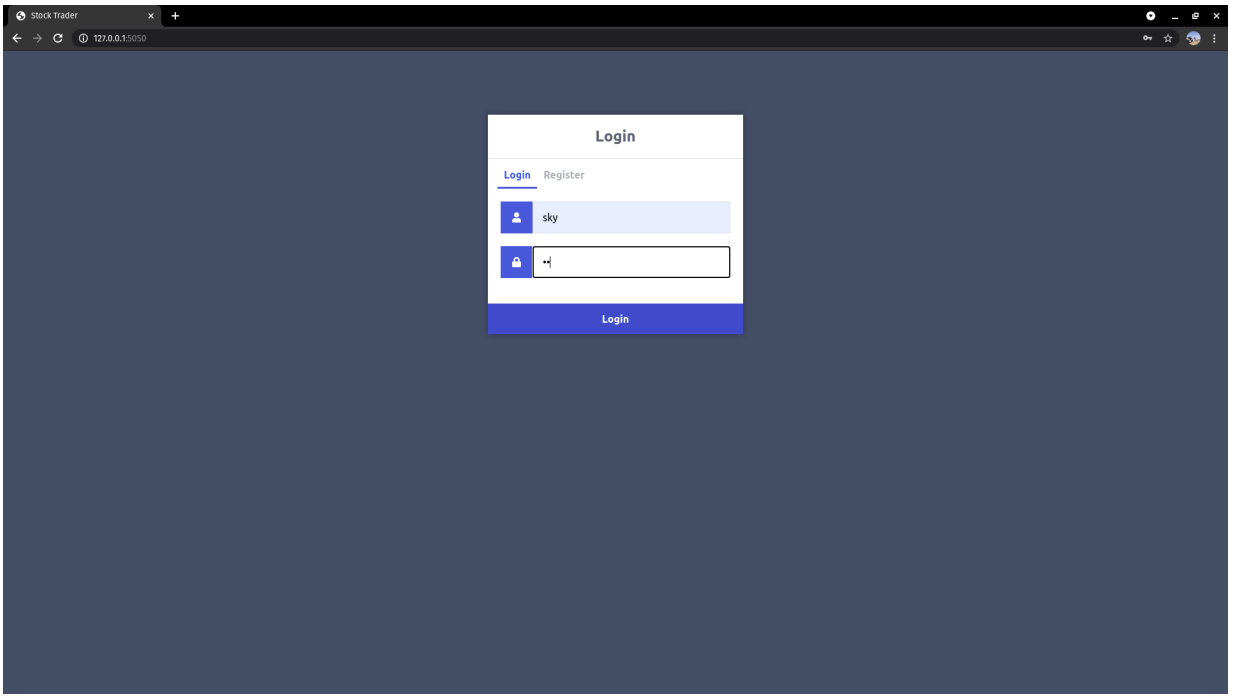


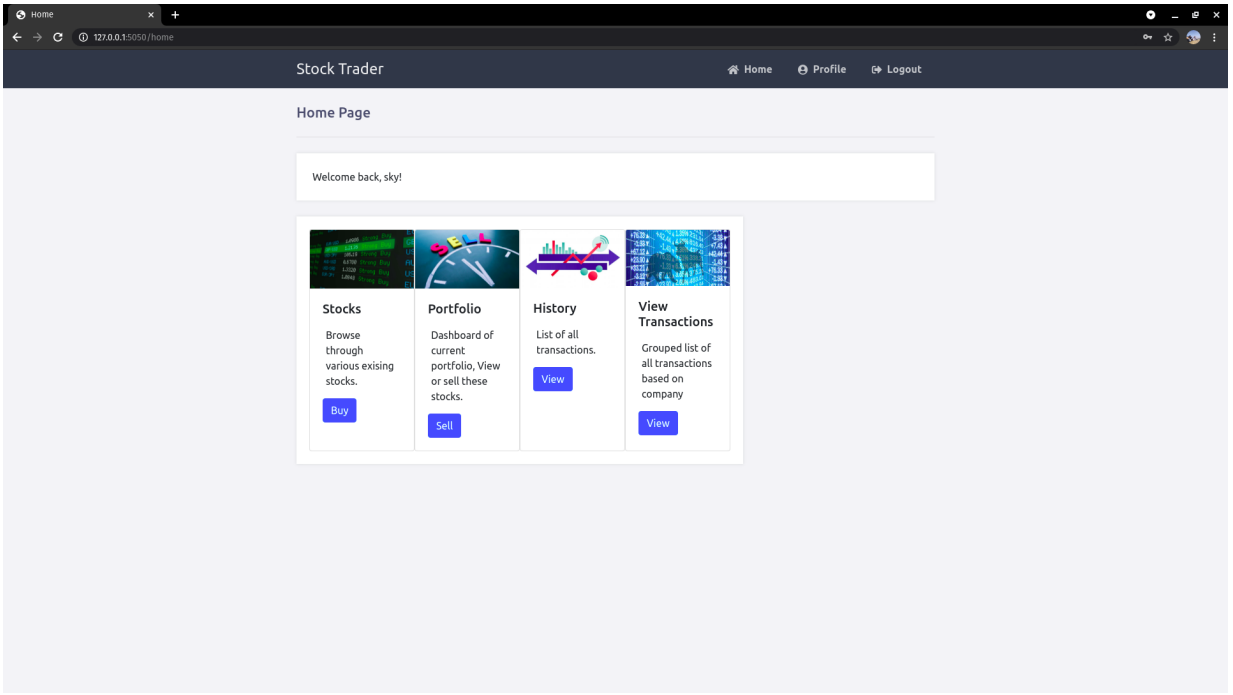


- After successfully registering the user we now test out the login, if the username or the password is wrong we show the following error message.

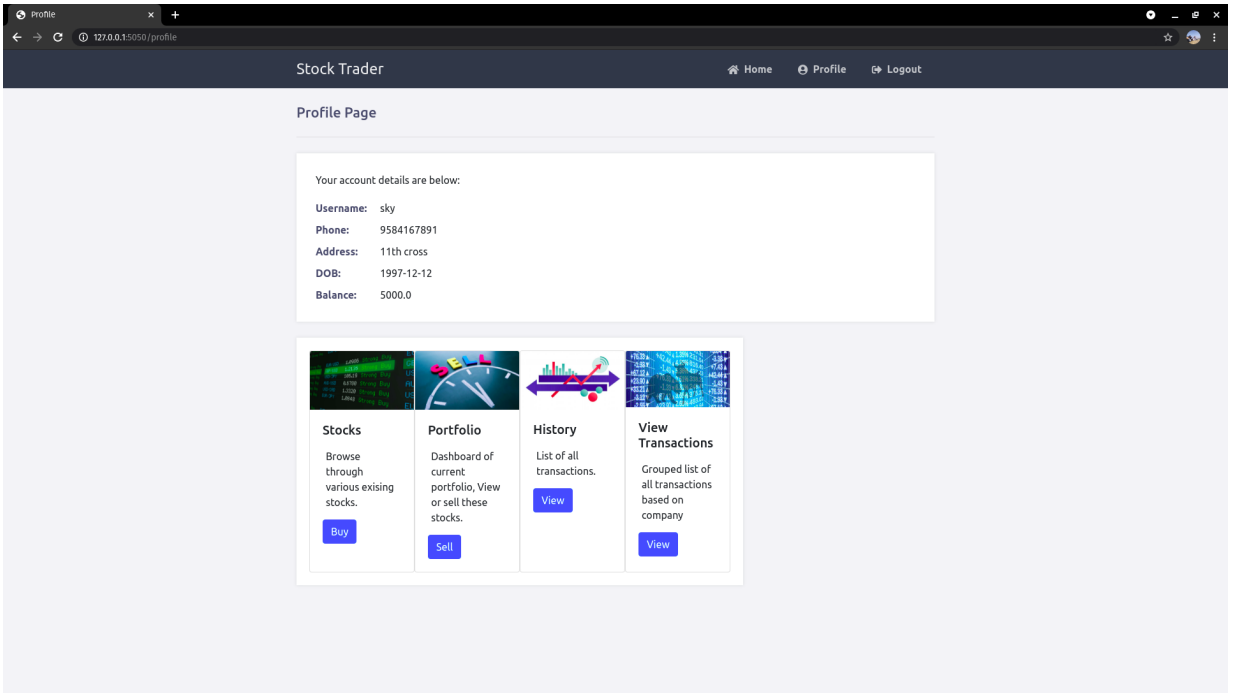


- If we enter the correct credentials in the login page, we will be redirect to the homepage of the user

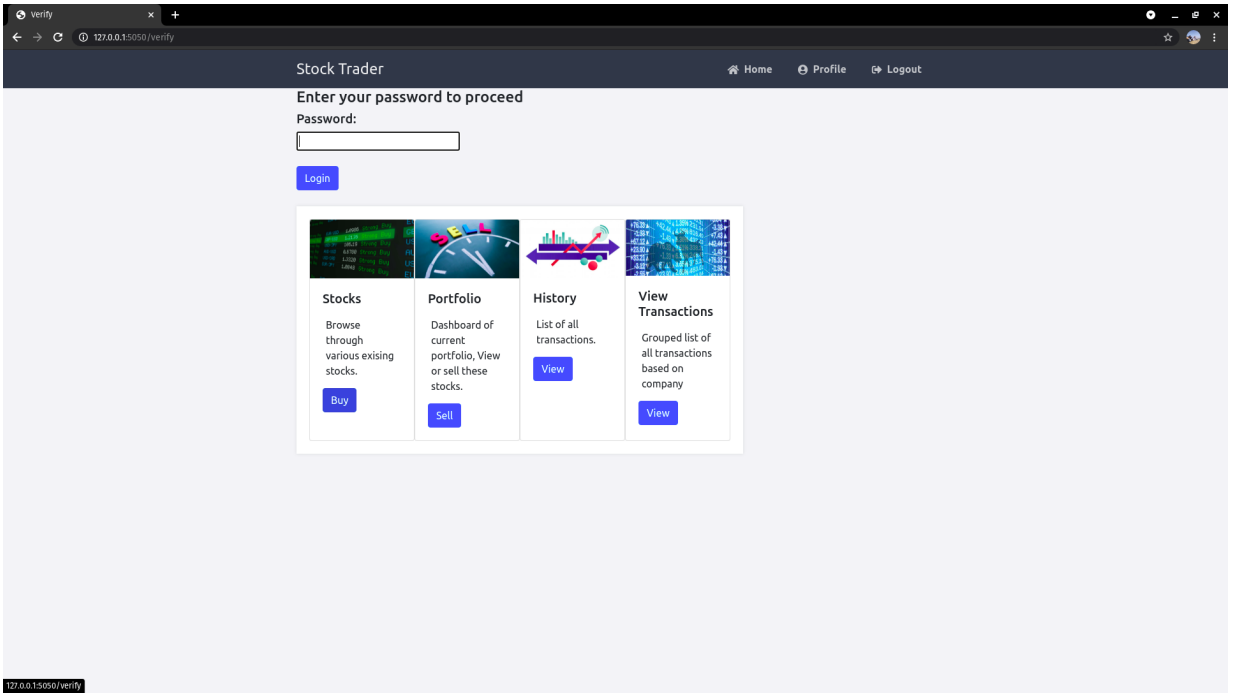




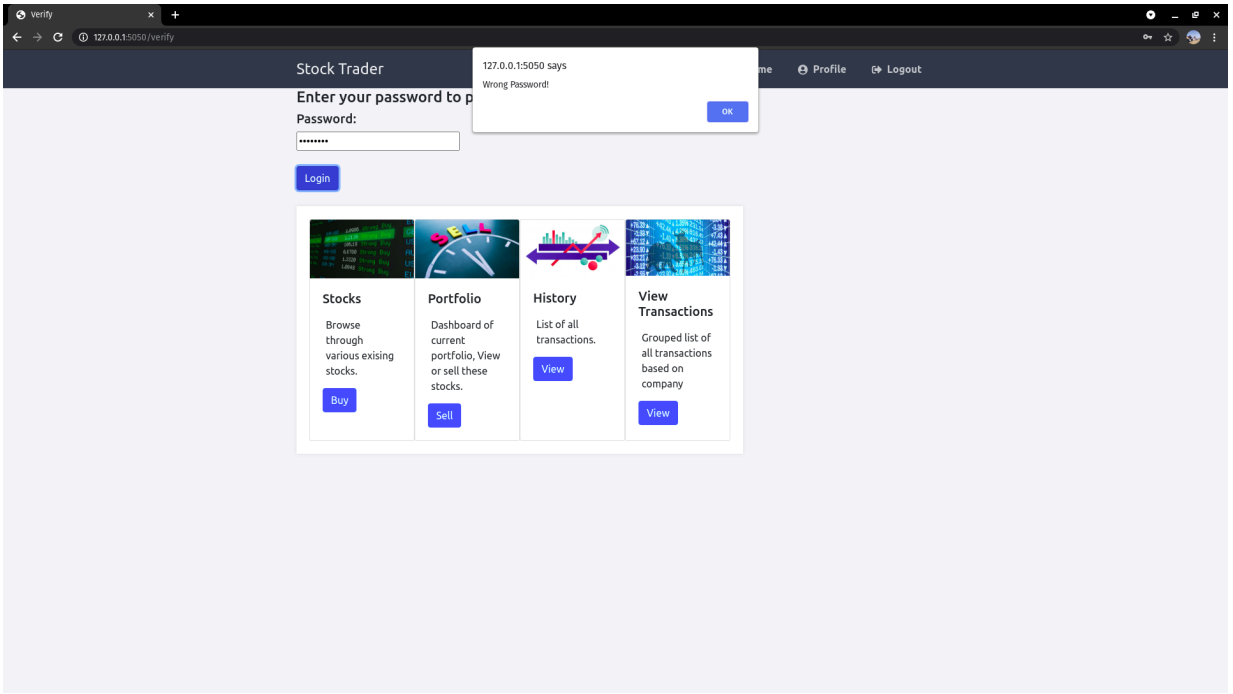
- When we click on the profile page of the user we can see the user info except for the passwords



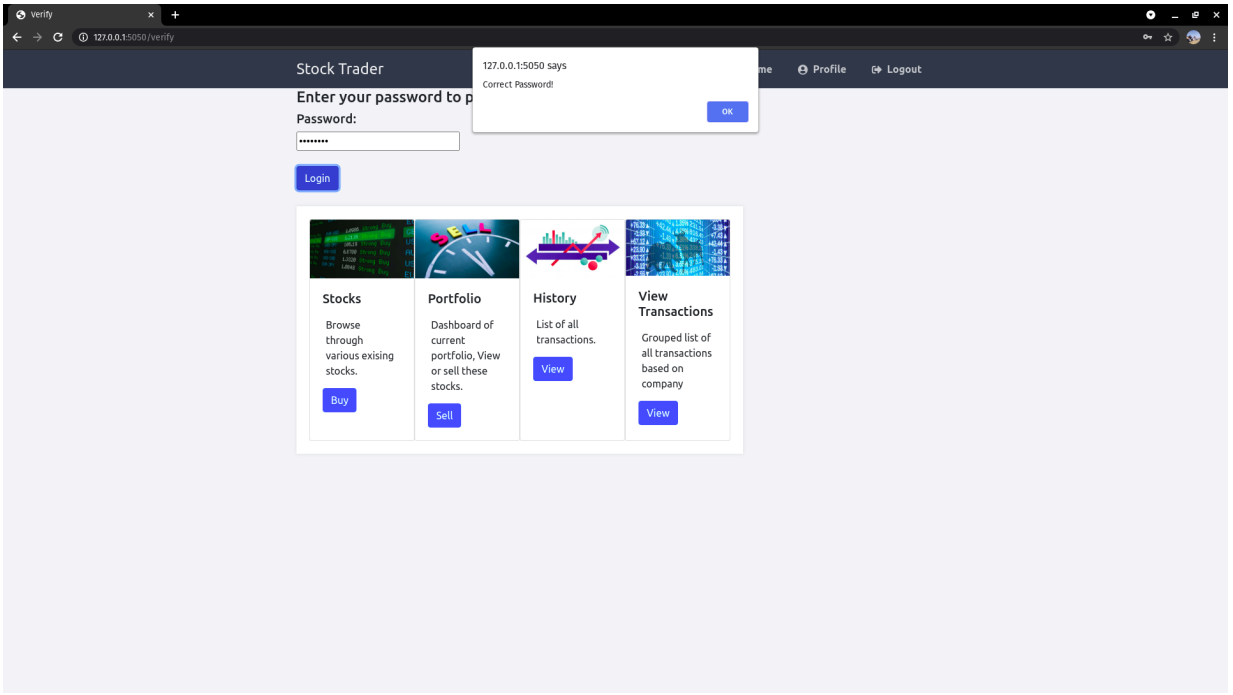
- By clicking on the “Buy” button we are prompted to enter the credit card password

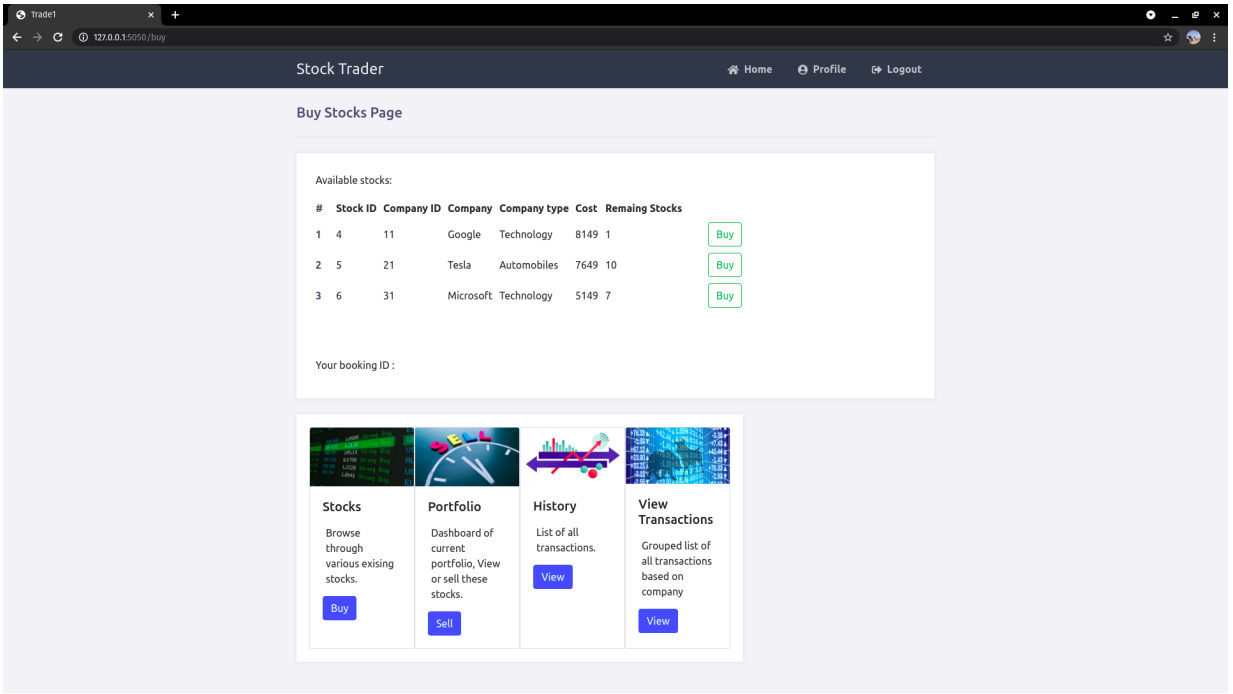


And if the password is wrong, we will get an alert saying “Wrong password”

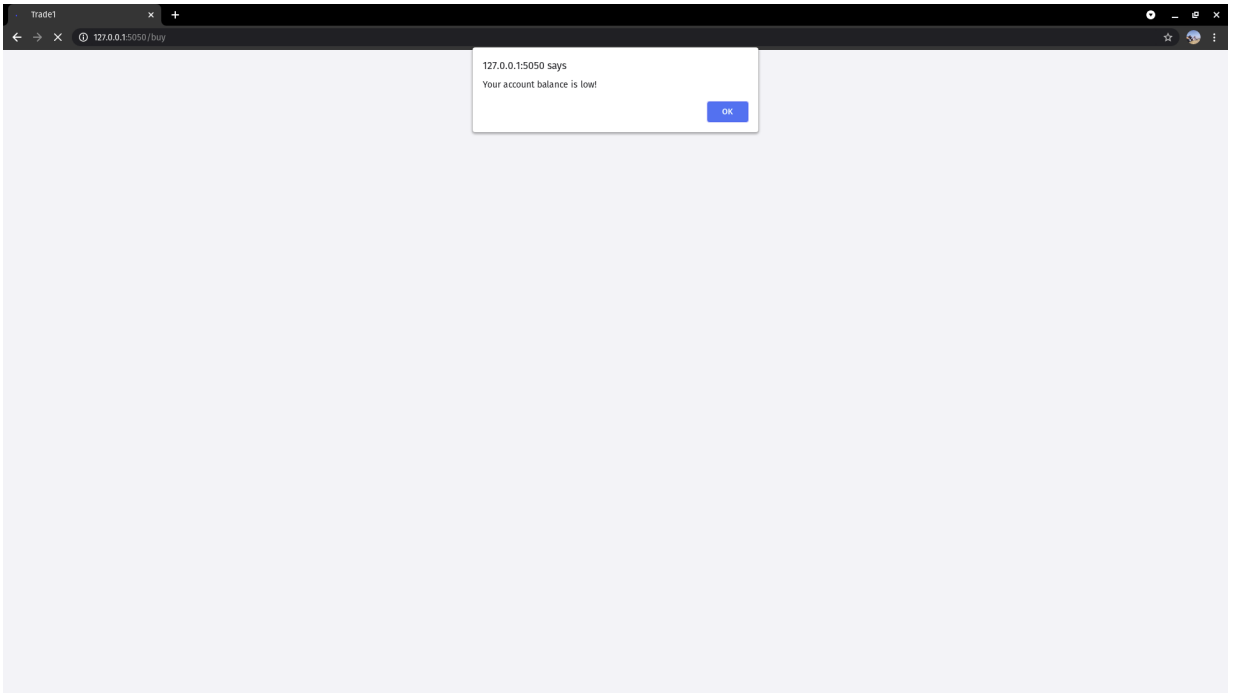


But if the password is correct we get a message saying “Correct password” and we get redirected to the buy page.





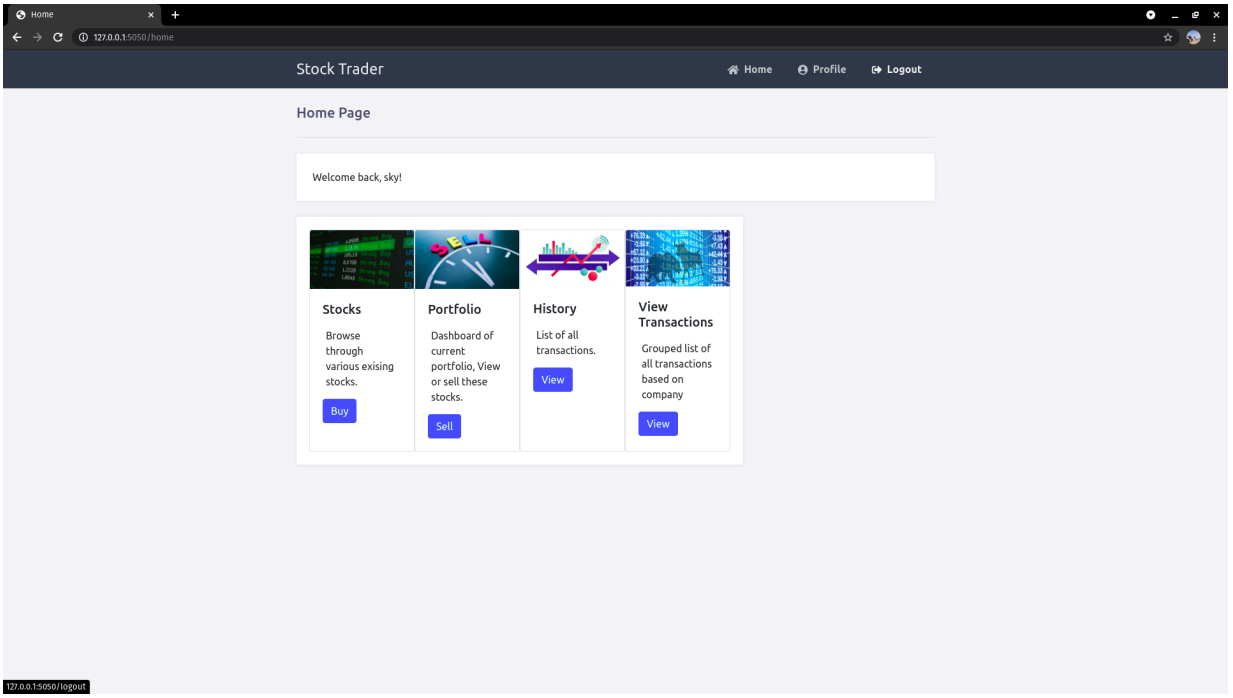
- Now when the user with username sky tries to buy a stock worth 7000+ we get the following alert

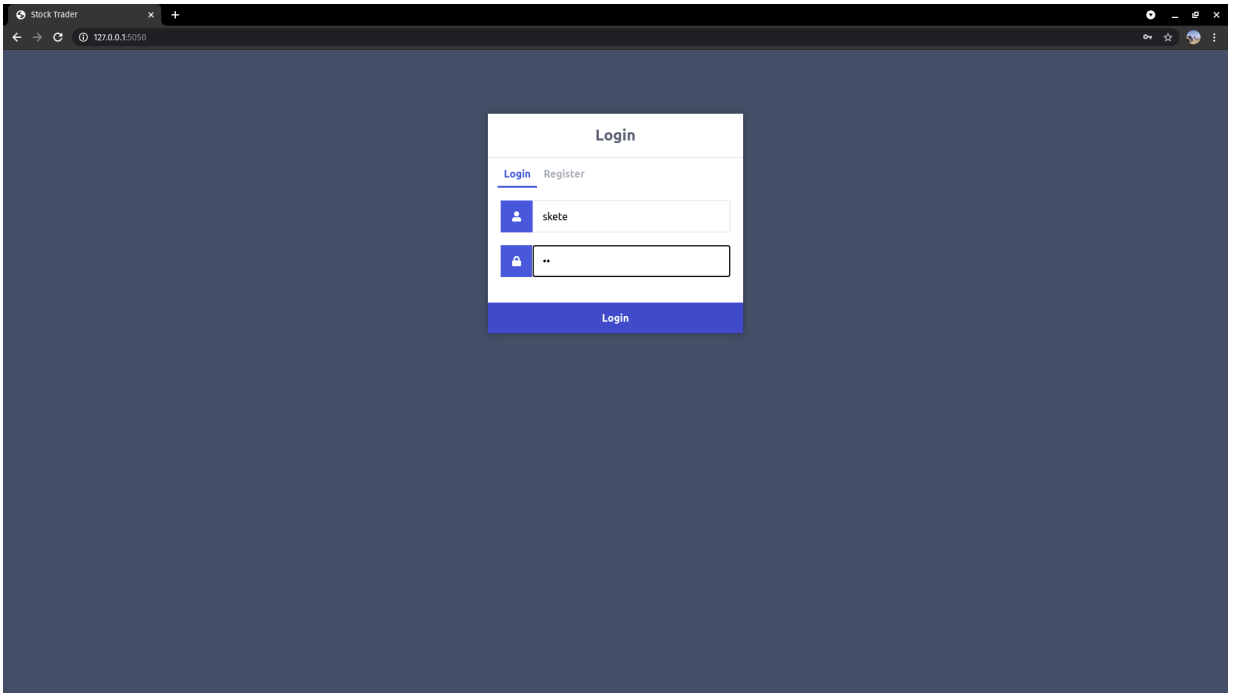


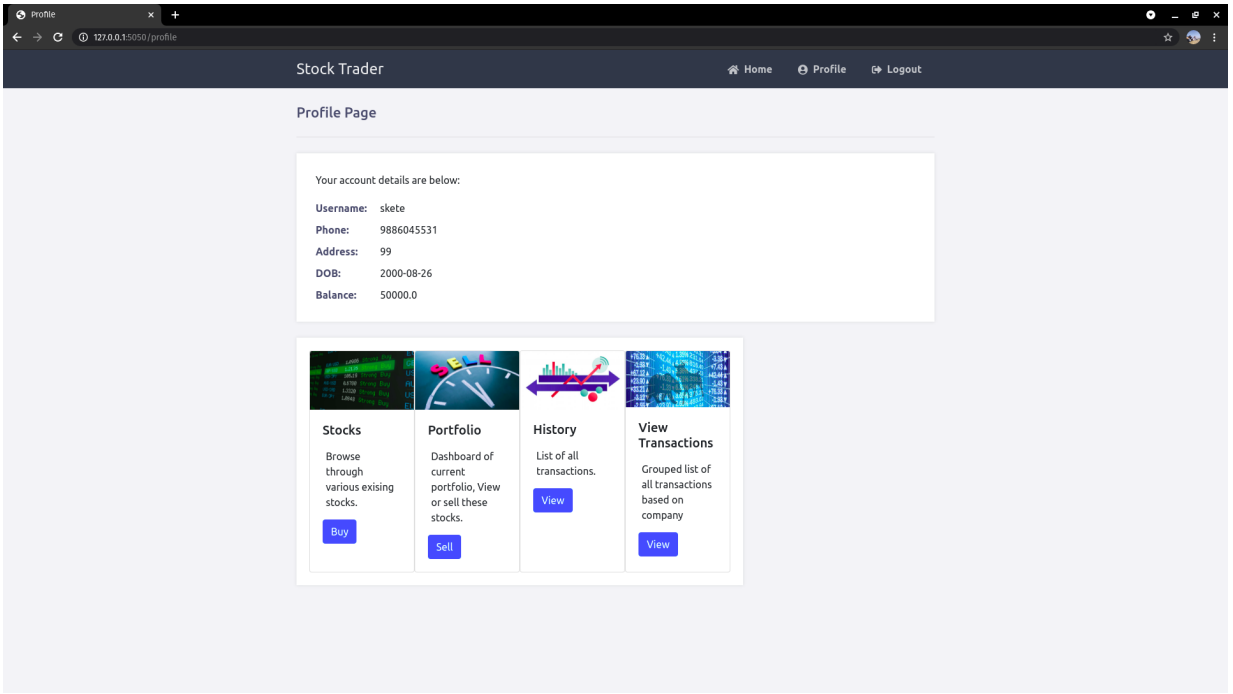
As the user only had 5000 in their account as seen on the profile page

---

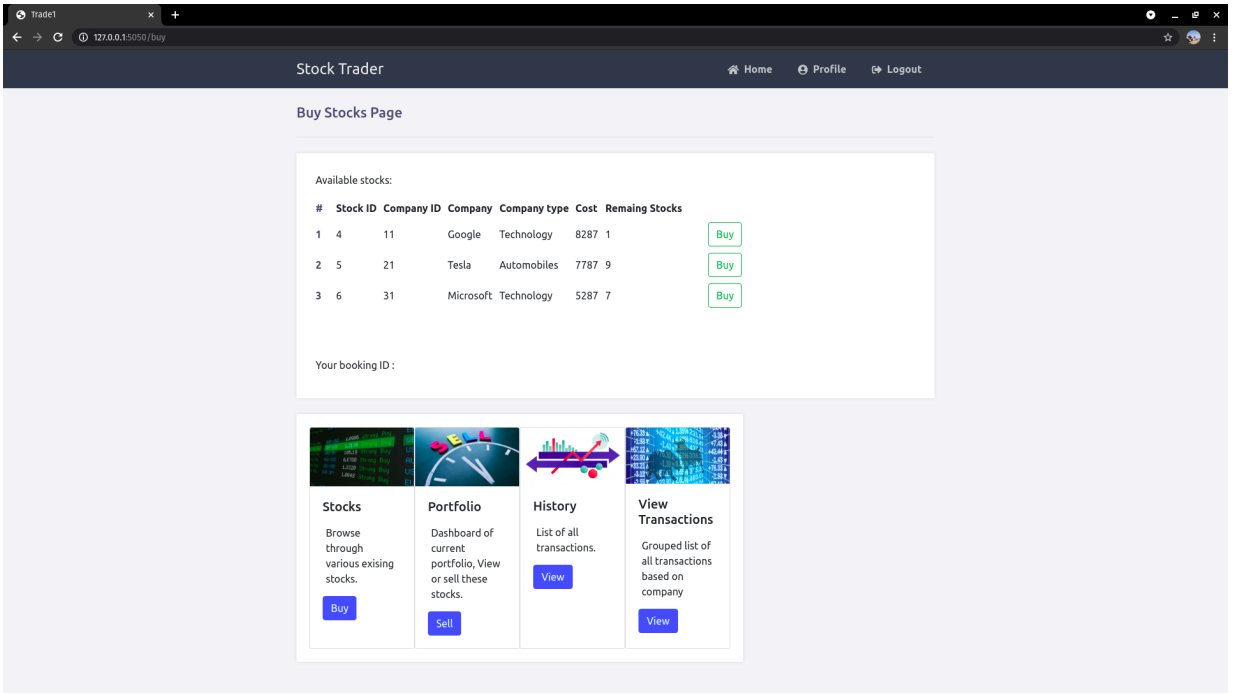
- So we log out as the user “sky” by clicking on the logout button on the home page and this redirects to the login page where we login as “Skete”







- Now we enter the buy page by clicking on the buy button and typing in the credit card password.



- After clicking on multiple stocks to buy ,the “Buy” page is auto update to show the remaining stocks

Trade1

127.0.0.1:5050/buy

Stock Trader

Home Profile Logout

Buy Stocks Page

Available stocks:

#	Stock ID	Company ID	Company	Company type	Cost	Remaing Stocks	
1	4	11	Google	Technology	8246	1	Buy
2	5	21	Tesla	Automobiles	7746	8	Buy
3	6	31	Microsoft	Technology	5246	7	Buy

Your booking ID :

Stocks

Browse through various exising stocks.

Buy

Portfolio

Dashboard of current portfolio, View or sell these stocks.

Sell

History

List of all transactions.

View

View Transactions

Grouped list of all transactions based on company

View

Trade1

127.0.0.1:5050/buy

Stock Trader

Home Profile Logout

Buy Stocks Page

Available stocks:

#	Stock ID	Company ID	Company	Company type	Cost	Remaing Stocks	
1	4	11	Google	Technology	8251	1	Buy
2	5	21	Tesla	Automobiles	7751	7	Buy
3	6	31	Microsoft	Technology	5251	7	Buy

Your booking ID :

Stocks

Browse through various exising stocks.

Buy

Portfolio

Dashboard of current portfolio, View or sell these stocks.

Sell

History

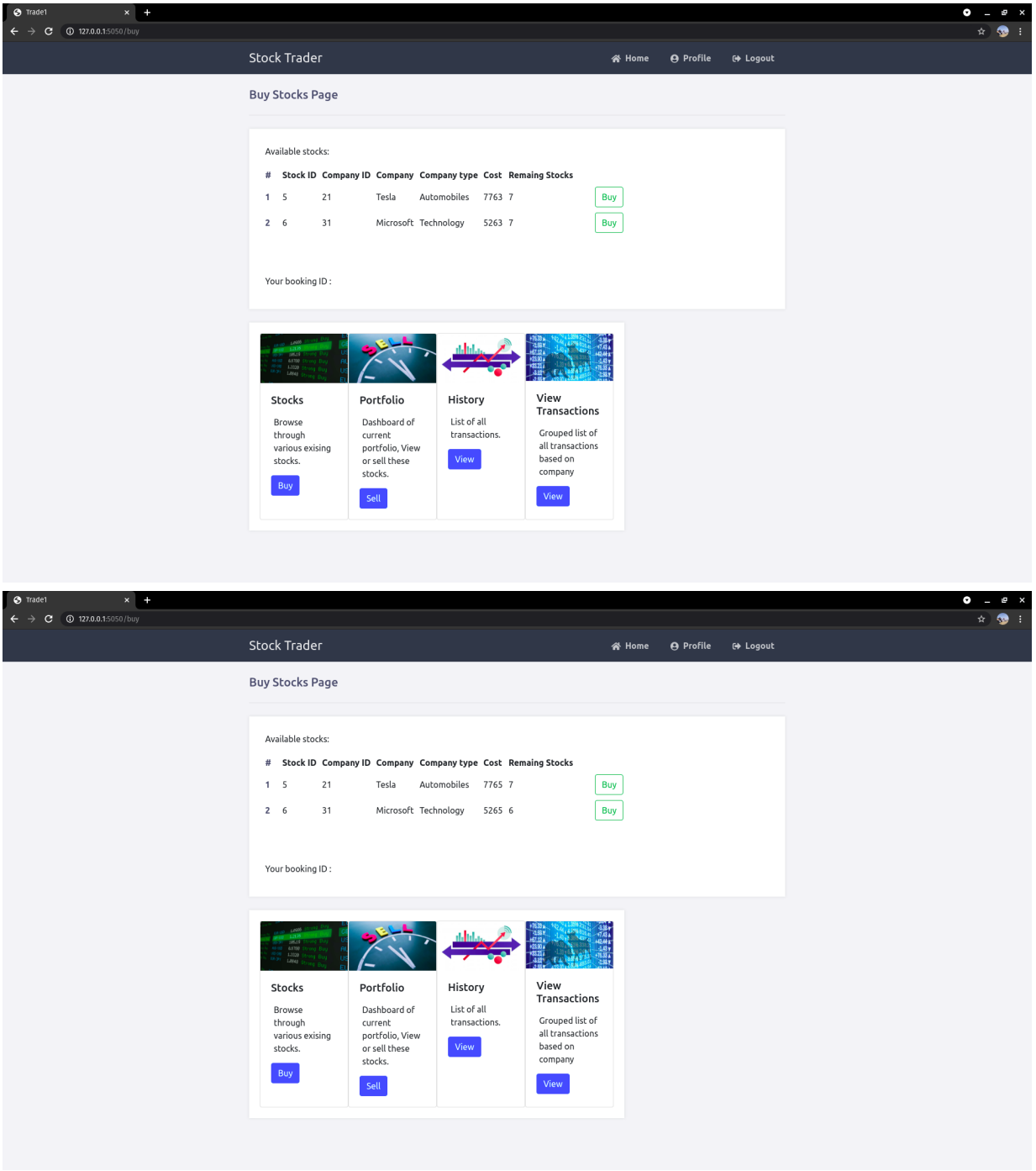
List of all transactions.

View

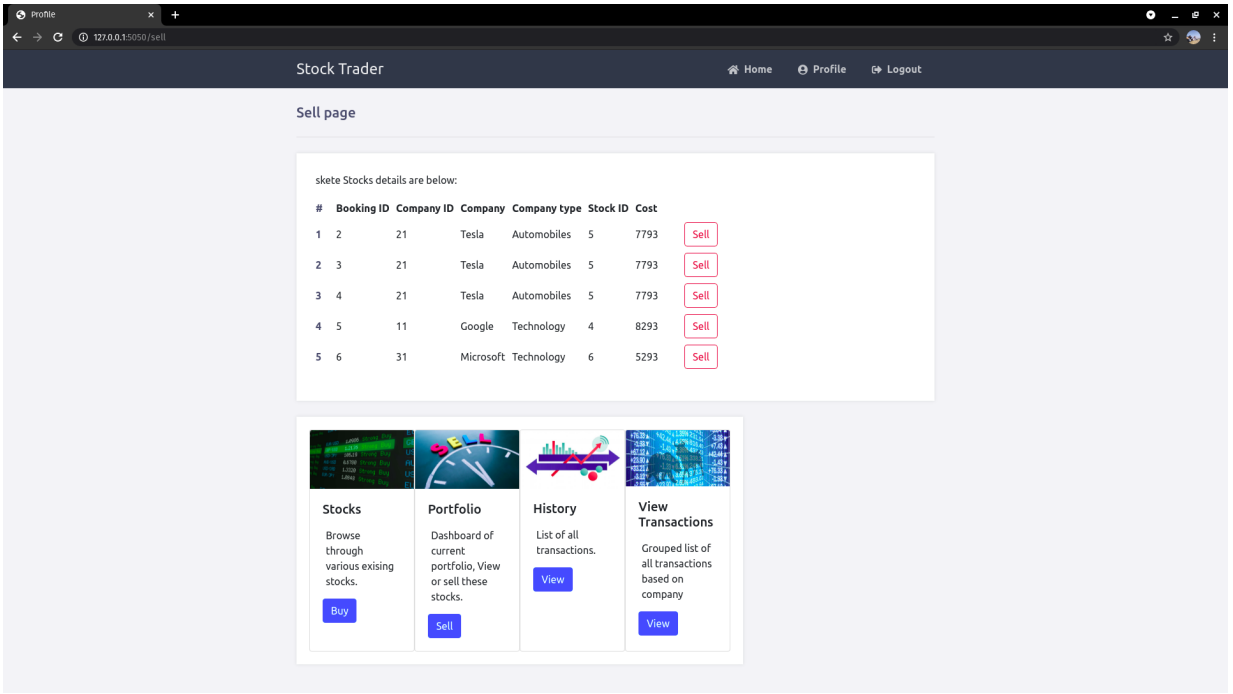
View Transactions

Grouped list of all transactions based on company

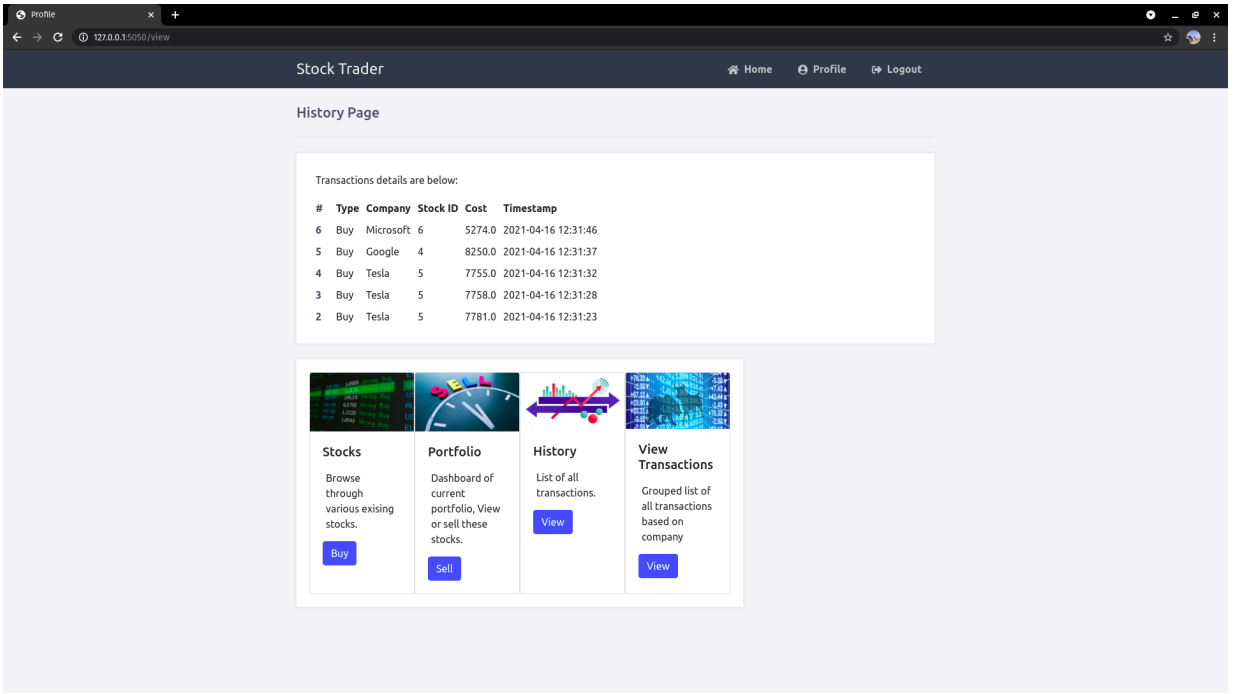
View



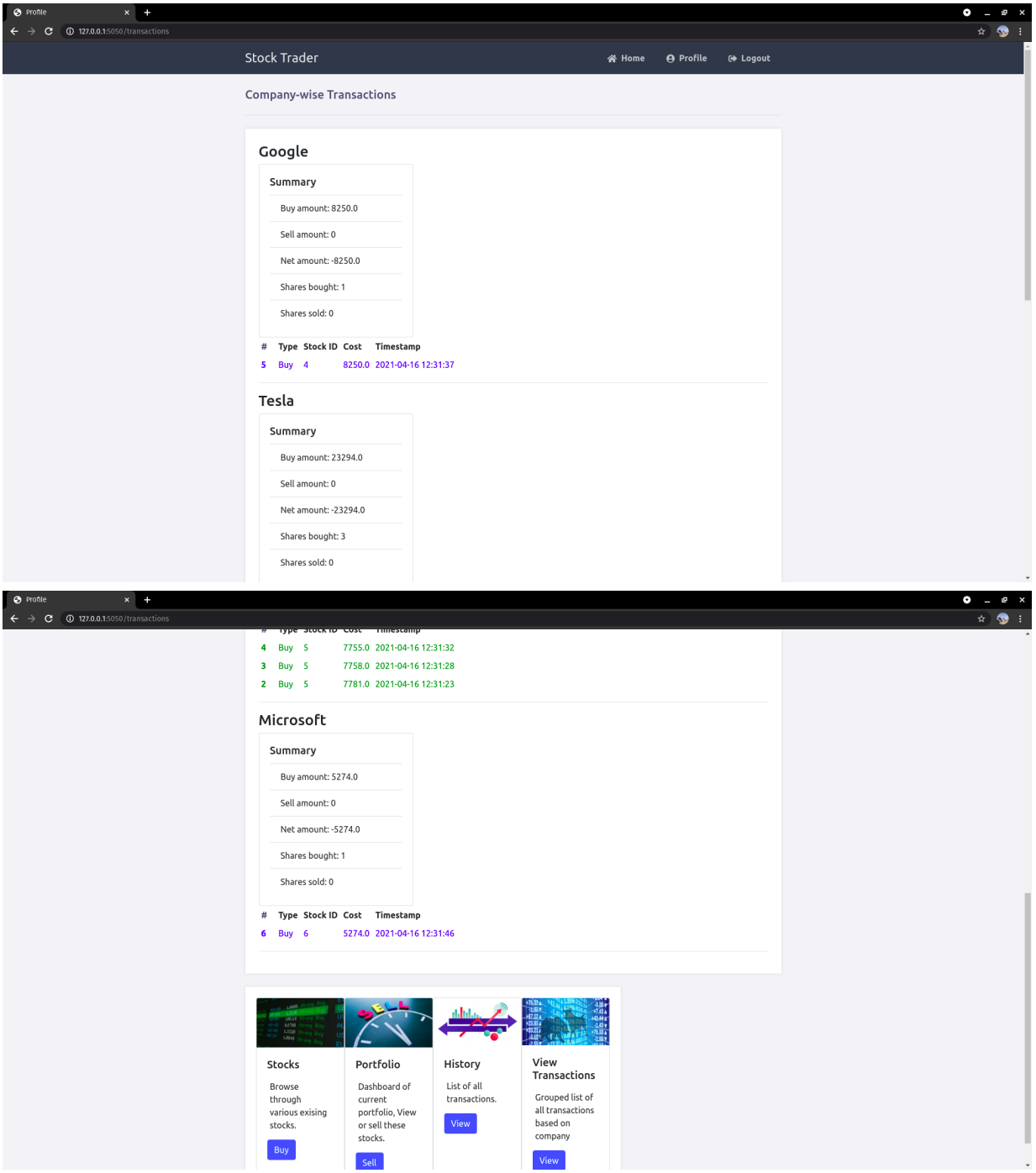
- And to view/sell the bought stocks we can go to the sell page



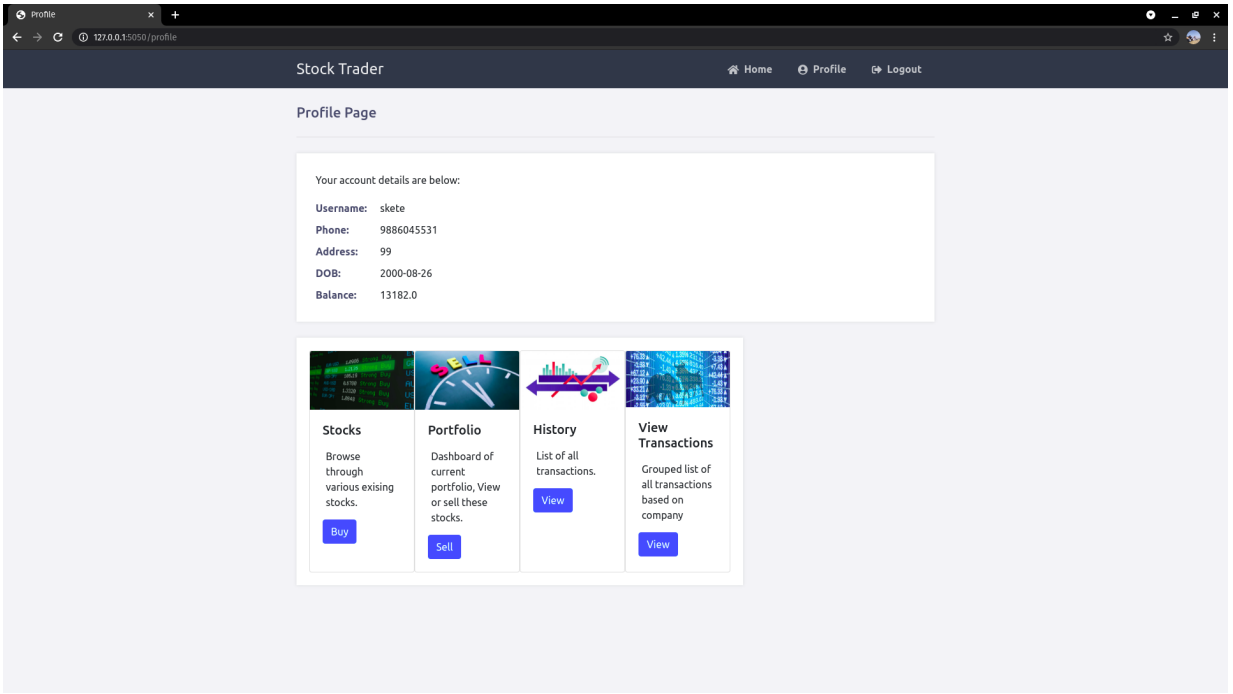
- To look at all the actions/transactions done by the user, we need to enter the view page by clicking on the “View” button.



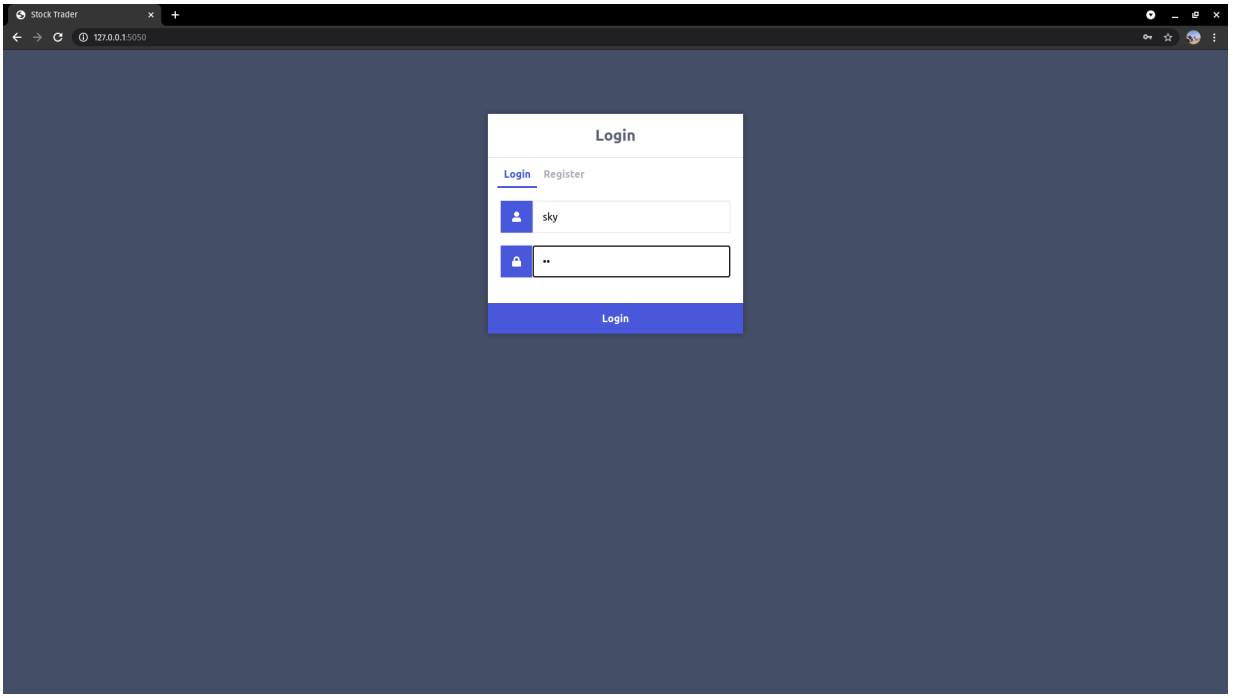
- And to look at the company wise transactions we need to visit the transactions page by clicking on the view button.

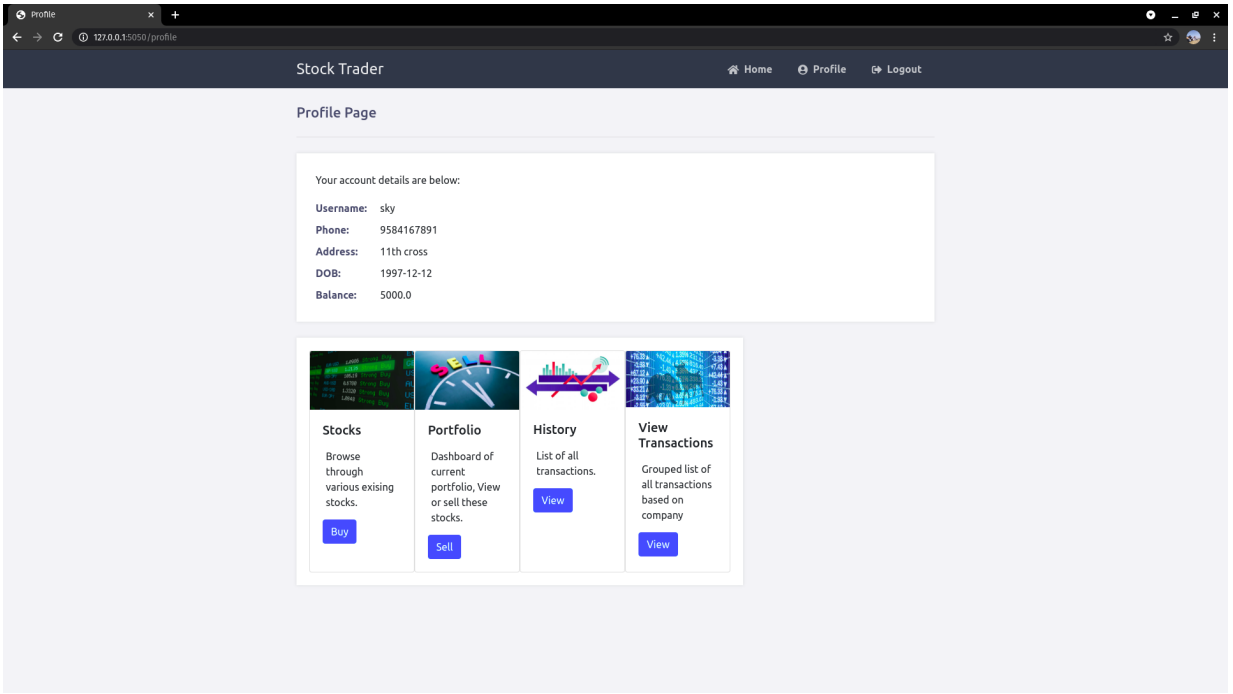


- Now we can also see the remaining balance of the user “skete” by visiting the profile page of the user.

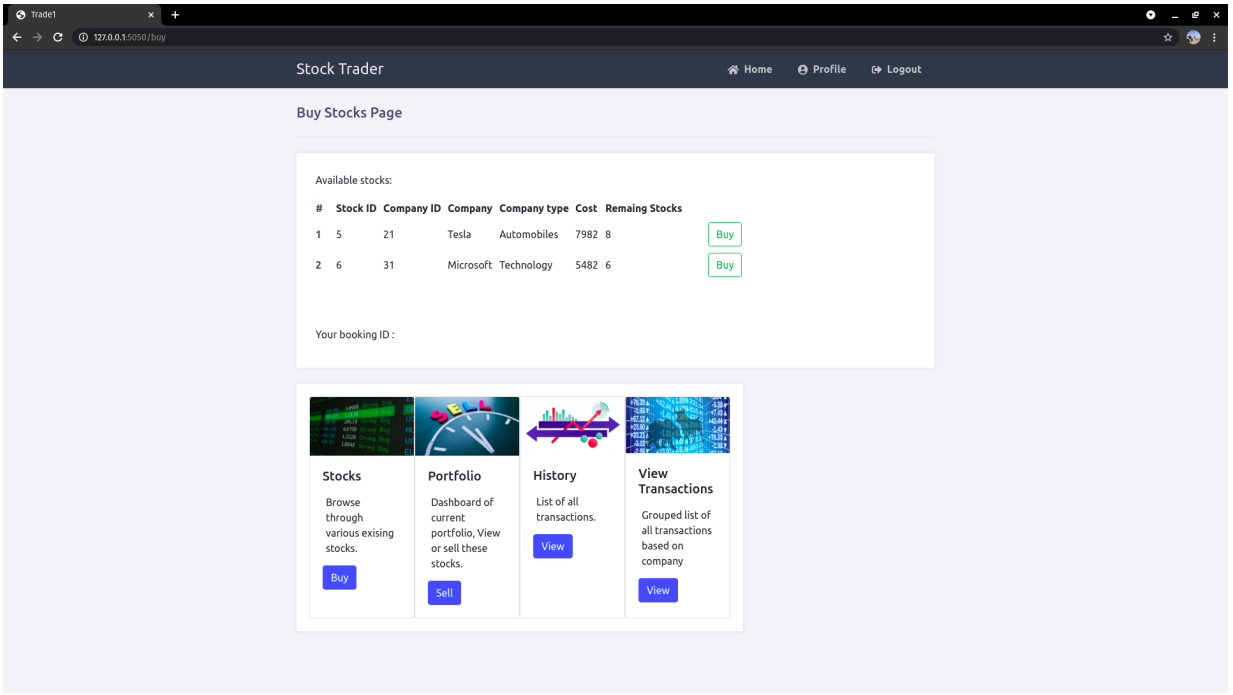


- Next we logout of “skete” and log back in as the user “sky”





- And now we visit and “buy” page



Here we can see that the stocks that “skete” bought are not available for “sky” to buy, hence we can say that the singularity and integrity of the stocks is maintained.

And also if we notice the price of the stocks on the website, it keeps fluctuating. This is to simulate the real time flow of stock share prices.