SRS requirements for:

BULLSTOCKS

A simple stock trading application

By:

Group 25

Aamina Mariam

(2210062)

Mohammad Yousuf

(22100289)

Sabahat Kashif

(22100248)

Table of Contents

1. Introduction

1.1 Purpose
This document specifies the software requirements of BullStocks. This app will help users buy and sell stocks conveniently, keeping in view their own buying and selling history, company's records and other helpful features.

1.2 Scope
The function of BullStocks is to provide an interactive software for stock trading. It provides its users a platform for stock trading with a stock broker.

1.3 Definitions

- **Portfolio:** A portfolio is a collection of financial investments like stocks, bonds, commodities, cash, and cash equivalents, including closed-end funds and exchange-traded funds. (ETFs)

- **Dashboard:** Dashboard is the page that the users see in the web application when they log in. It is the page that shows the analysis of the application's data, trends, summaries etc. From the Dashboard the users can drill down to get more information about a particular piece of data.

- **Simple Return:** The simple rate of return uses the following formula: Subtract the value of the portfolio at the end of the year from the value of the portfolio at the beginning of the year, then divide that number by the value at the beginning of the year.

- **Cash:** Currently available balance of the user, which they can use to buy stocks/ money gotten after selling a stock

- **Stock Trading:** Stock trading refers to the buying and selling of shares in a particular company; if you own the stock, you own a piece of the company.

- **S&P 500**: The S&P 500, or simply the S&P, is a stock market index that measures the stock performance of 500 large companies listed on stock exchanges in the United States. It is one of the most commonly followed equity indices

- **User history:** The history of a user's buying and selling stocks

- **Stock history:** The history of a stocks buying and selling, along with prices

- **Stock Broker:** A broker who buys and sells securities on a stock exchange on behalf

of clients.

- **Stock Exchange:** A market in which securities are bought and sold.

2. Overall Description:

2.1 Product Perspective

Bull Stocks is a comprehensive stock trading system which provides a trading platform to the user for trading securities.It would not only allow users to purchase stocks through their bank account but also provide the stock broker to view their clients' portfolios.

2.2 Product Features

BullStocks is an app-based platform which provides users with a detailed view of the stocks market and enables them to make their choices of buying and selling stocks thoroughly. The features which users can facilitate themselves with are viewing the stock market index, history of a particular stock, trade history, the cash available to them. The users can then make an informed choice according to their resources and profit interests.

2.3 User Classes and Characteristics:

- User: The user is the person or entity that wants to open an account with a stock broker and trade stocks using that account. The user will pay using their bank account and trade stocks using their account when the stock Exchange is open.

- Stock Broker/Admin: The stock broker is a company registered with the stock Exchange that allows the user to create an account and trade stocks using its platform.

2.4 Design and Implementation Constraints

The design of our product will be simple, user friendly and quite self explanatory, therefore, can be used by users without any difficulties.

2.5 Operating Environment:

The product is expected to be deployed as a python app which can run on Windows PCs and Laptops.

2.6 User Documentation:

User functionality has been described in detail in System requirements(chapter 3).

3. System Requirements

3.1 Functional Requirements

The functional requirements for the system are divided into two sections:

1. User Functional requirements
2. Admin/Stock Broker Functional requirements

3.1 Functional Requirements

This includes User Functional Requirements and Admin/Broker Functional requirements.

3.1.1 User Functional Requirements

User Functional Requirement 1

Buy Stock:

- Description: The user can buy a stock that they are interested in. using their account.
- Input: Login credentials
- Process: Information Authentication
- -Output: stock bought

User Functional Requirement 2

Sell Stock:

- Description: The user can sell a stock they wish to sell, using their account.
- - Input: Enter login details
- Process: Authentication
- Output: Stocks sold

User Functional Requirement 3

Trade history:A user will be able to view all the trades that have been performed from the trading account. Trade history includes all the stocks bought and sold through that account.

- Input: Click trade history
- Process: acces the the trades table
- Output: All transactions performed by the user

User Functional Requirement 4

Account Balance: A user will be allowed to view the account balance which will be updated after every trade.

User Functional Requirement 5

Stock History: A user will be able to see a graph of the stock's performance over a specified period of time. This helps the user decide whether to buy or sell the stock.
Input:User clicks the name of a stock
Process: Fetches the history of a stock given a specified period of time.
Output: The history of the stock in the shape of a graph.

User Functional Requirement 6

Performance of the portfolio relative to the S&P 500:
Description: Enables the user to compare his stocks value to the market one, and overall performance.
Input:Click Portfolio performance
Process:Fetches the performance of S&P 500.
Output: A graph of the portfolio's returns and S&P 500 returns.

3.1.2 Admin/Broker Functional Requirements

Admin Functional Requirement 1

View Clientele

Description: The Brokerage firm can view the accounts of its clients and view the whole database.
Input:Admin logs in using special id and password.
Process:The client table and portfolios of clients are fetched.
Output: Clients and their Portfolios.

3.2 Security requirements:

- Only users with an account can login.
- Stock brokers can only view their clients and their portfolios but can not change them.

3.3 External Interface requirements:

- **Software requirements:** The software provides an easy to use GUI for the user to interact with the stock market. It provides real time prices of the stocks and allows them to trade stock.

Chapter 4:Use cases:

Primary Actor	***Use Cases***
Admin/Brokerage firm	User portfolio(Authority to access the complete database)
User	1. Login/Sign up 2. Buy 3. Sell 4. User history 5. Stock History 6. User portfolio Profit/loss Comparison of portfolio with stock market

Use Case ID:	1		
Use Case Name:	Login/ Signup		
Created By:	Aamina Mariam(User)	Last Updated By:	
Date Created:	October 8, 2020	Date Last Updated:	

Actors:	Users, broker/admin
Description:	Users can create an account to begin trading.
Trigger:	User wants to login or create an account
Preconditions:	1. The Id and password of a new user signing should not match any in the database. 2. The user logging in should have a valid Id and password
Postconditions:	A correct login should direct the user to the dashboard. New users should also be directed to the dashboard.
Normal Flow:	**1.0 Login/Signup** 1. Normal login should direct the users to the dashboard.
Alternative Flows:	**1.1 Invalid login Information** 1. If the Id or password are invalid system should show a prompt and ask the user to login again.
Exceptions:	User does not have an account/has not signed up.
Includes:	Admin/brokerage firm
Priority:	High
Frequency of Use:	Login: frequently. Whenever the client needs to access his account. Signup: rarely. Whenever a user wants to make an account.
Business Rules:	None
Special Requirements:	None
Assumptions:	None-
Notes and Issues:	None

Use Case ID:	2		
Use Case Name:	Buying Stocks		
Created By:	Aamina Mariam	Last Updated By:	
Date Created:	October 8, 2020	Date Last Updated:	

Actors:	User
Description:	Users want to buy stocks of different companies.

Trigger:	User wants to buy stocks to gain profit,
Preconditions:	1. The user is registered with the brokerage firm. 2. The user has enough currency for trading.
Postconditions:	1. Users currency is updated. 2. Users portfolio is updated.
Normal Flow:	**2.0 Stock buying** 1. User selects a company from the dashboard. 2. User enters the amount of shares he/she is interested in buying. 3. User selects the buy option.
Alternative Flows:	None
Exceptions:	**2.0.E.1 User currency finished** 1. System informs the user that his/her currency has finished. 2. System cancels the transaction. 3. System terminates use case.
Includes:	1. User Portfolio 2. Admin/Brokerage firm
Priority:	High
Frequency of Use:	Whenever the user wants to buy stocks.
Business Rules:	None
Special Requirements:	Users should be able to cancel transactions prior to step 3 in normal flow.
Assumptions:	System has information about the user's account/currency. Specific amount is deducted from there.
Notes and Issues:	

Use Case ID:	3		
Use Case Name:	Selling Stocks		
Created By:	Aamina Mariam	Last Updated By:	
Date Created:	October 8, 2020	Date Last Updated:	

Actors:	User
Description:	The user needs to sell shares of a company.
Trigger:	Either the cshare price is going extremely high or going low
Preconditions:	
Postconditions:	1. The user portfolio is updated.
Normal Flow:	**3.0 Stock selling** 1. User is shown the stocks he owns. 2. User selects a company from his/ her portfolio.

	3. User enters the amount of shares he/she is interested in selling. 4. User selects the sell option.
Alternative Flows:	None
Exceptions:	**3.0.E.1**
Includes:	1. User portfolio 2. Admin/Brokerage firm
Priority:	High
Frequency of Use:	Whenever user wants to sell stocks
Business Rules:	
Special Requirements:	Users should be able to cancel transactions prior to step 3 in normal flow.
Assumptions:	System has information about the users bank account. The return is transferred to that account.
Notes and Issues:	

Use Case ID:	4		
Use Case Name:	Trade History		
Created By:	Sabahat	Last Updated By:	
Date Created:	October 8, 2020	Date Last Updated:	

Actors:	User
Description:	Users can view their history
Trigger:	User wants to view their history
Preconditions:	3. The user is logged in
Postconditions:	None
Normal Flow:	1. Customer logs in to their account 2. View their history
Alternative Flows:	None
Exceptions:	Not signed up/ not logged in
Includes:	no other use case
Priority:	Medium
Frequency of Use:	Sometimes
Business Rules:	None
Special Requirements:	None
Assumptions:	User has bought and sold some stocks

Notes and Issues:	None

Use Case ID:	5		
Use Case Name:	Stock History		
Created By:	Sabahat	Last Updated By:	
Date Created:	October 8, 2020	Date Last Updated:	

Actors:	User
Description:	User can view the history of a particular stock
Trigger:	User wants to view the history of a stock he is interested in buying/selling
Preconditions:	4. The user has signed up already and is logged in to his account.
Postconditions:	None
Normal Flow:	1. Logs in 2. goes to company's stock and checks stock history
Alternative Flows:	None
Exceptions:	Not signed up/ logged in
Includes:	no other use cases
Priority:	Medium
Frequency of Use:	Sometimes. Whenever the user wants to buy a stock (not necessarily always)
Business Rules:	None
Special Requirements:	None
Assumptions:	Stock has some buying and selling history. Users account already exists
Notes and Issues:	None

Use Case ID:	6		
Use Case Name:	User Portfolio		
Created By:		Last Updated By:	

Date Created:	October 8, 2020	Date Last Updated:	

Actors:	User
Description:	The user wants to view his/her own information. Most probably information about the stocks that he owns.
Trigger:	The user wants to see what stocks he/she owns and may want to buy or sell stocks.
Preconditions:	User has logged in and his her identity has been authenticated
Postconditions:	
Normal Flow:	**6.0 User portfolio** 1. The user logs in and wants to access his/her portfolio. 2. The user goes to his/her portfolio. 3. The user returns back to dashboard when he/she no longer wants to access the portfolio.
Alternative Flows:	**6.1**
Exceptions:	**6.0.E.1**
Includes:	
Priority:	Medium
Frequency of Use:	Whenever a user wants to access his/her portfolio.
Business Rules:	None
Special Requirements:	None
Assumptions:	User has an account
Notes and Issues:	None

Use Case ID:	7		
Use Case Name:	Brokerage firm/Admin		
Created By:	Yousuf	Last Updated By:	
Date Created:		Date Last Updated:	

Actors:	Admin
Description:	The brokerage firm can view their clients and their portfolios
Trigger:	brokerage firm wants to see/update client portfolios
Preconditions:	Needs to have id and password of admin
Postconditions:	None
Normal Flow:	Brokerage firm logs in as admin and list of clients and their portfolios shows up
Alternative Flows:	if wrong password for admin, they will be redirected to main page

Exceptions:	None
Includes:	None
Priority:	High
Frequency of Use:	low
Business Rules:	Admin can not buy/sell stocks on behalf of the client
Special Requirements:	Admin has no email id and has a specific id and password
Assumptions:	None
Notes and Issues:	None

Use Case ID:	8		
Use Case Name:	Portfolio returns		
Created By:	Yousuf	Last Updated By:	
Date Created:		Date Last Updated:	

Actors:	User
Description:	The return of the portfolio will be calculated
Trigger:	
Preconditions:	The user portfolio must have at least 1 stock.
Postconditions:	The return is displayed as a simple rate of return and updated regularly.
Normal Flow:	The user opens their portfolio and returns are displayed at the top.
Alternative Flows:	None
Exceptions:	The user doesn't have any stock in their portfolio.
Includes:	None
Priority:	High
Frequency of Use:	High
Business Rules:	Return decreases if the value of the stocks in the portfolio decreases.
Special Requirements:	None
Assumptions:	None
Notes and Issues:	None

Use Case ID:	9

Use Case Name:	Performance of the portfolio relative to the S&P 500 index		
Created By:	Yousuf	Last Updated By:	
Date Created:		Date Last Updated:	

Actors:	User
Description:	It compares the users return to the S&P 500
Trigger:	A user click Portfolio returns
Preconditions:	The portfolio should have a return
Postconditions:	A graph is generated
Normal Flow:	A graph of the user's return and S&P 500 is plotted
Alternative Flows:	If stock market is closed that day, return stays con
Exceptions:	None
Includes:	None
Priority:	Low
Frequency of Use:	Medium
Business Rules:	The S&P500 index and return stays constant when market is closed.
Special Requirements:	None
Assumptions:	None
Notes and Issues:	None

Use Case ID:	10		
Use Case Name:	Total Gain/Loss		
Created By:	Yousuf	Last Updated By:	
Date Created:		Date Last Updated:	

Actors:	User
Description:	Calculates the total gain/loss for a stock
Trigger:	A opens portfolio
Preconditions:	The portfolio should have a stock
Postconditions:	None
Normal Flow:	total gain/loss for a stock is calculated
Alternative Flows:	None
Exceptions:	None

Includes:	None
Priority:	High
Frequency of Use:	low
Business Rules:	The stock should be owned by the user
Special Requirements:	None
Assumptions:	None
Notes and Issues:	None

Use Case ID:	11		
Use Case Name:	Sector indexes		
Created By:		Last Updated By:	
Date Created:	Oct 20th, 2020	Date Last Updated:	Dec 12th, 2020

Actors:	User
Description:	indices of top companies of top 5 sectors
Trigger:	User wants the comparison of his stocks or portfolio with market indices
Preconditions:	user has an account
Postconditions:	None
Normal Flow:	user is shown the stock indices of the companies
Alternative Flows:	in case of the market being closed, latest updated indices would be shown.
Exceptions:	Market is closed.
Includes:	None
Priority:	Medium
Frequency of Use:	sometimes
Business Rules:	no specific rules
Special Requirements:	None
Assumptions:	None
Notes and Issues:	None

Use Case ID:	12		
Use Case Name:	Market indexes		
Created By:	Sabahat	Last Updated By:	
Date Created:	Oct 20th, 2020	Date Last Updated:	Dec 12th, 2020

Actors:	User
Description:	3 Market stock indices: the NASDAQ Composite, Russell2000, and S&P 500 Index.
Trigger:	User wants to gauge the market movement before investing
Preconditions:	User has an account already
Postconditions:	None
Normal Flow:	Particular market indices are shown to the user
Alternative Flows:	in case of the market being closed, last updated indices would be shown.
Exceptions:	Market is closed.
Includes:	None
Priority:	Medium
Frequency of Use:	sometimes
Business Rules:	no specific rules
Special Requirements:	None
Assumptions:	None
Notes and Issues:	None

Item	Item Description	Complexity	Count	Weight	Weighted Count
1	Number of Actors	Simple		1	
		Average		2	
		Complex	2	3	6
2	Number of Use Cases	Simple	2	5	10
		Average	3	10	30
		Complex	7	15	105
Unadjusted Use Case Points (UUCP)					**151**

Factor	Description	Rating (0=Irrelevant, 5=Essential)	Weight	Weighted Rating

T1	Distributed system	1	2	2
T2	Response performance objectives	5	1	5
T3	End-user efficiency	4	1	4
T4	Complex internal processing	5	1	5
T5	Code must be reusable	3	1	3
T6	Easy to install	5	0.5	2.5
T7	Easy to use	4	0.5	2
T8	Portable	5	2	10
T9	Easy to change	5	1	5
T10	Concurrent	5	1	5
T11	Secure	5	1	5
T12	Access to 3rd parties	0	1	0
T13	User training facilities	5	1	5
Technical Factor (TF) = sum of weighted ratings				**53.5**
Technical Complexity Factor (TCF) = 0.6 + (0.01 x TF)				**1.135**

Factor	**Description**	**Rating** 0=Lowest 5=Highest	**Weight**	**Weighted Rating**
F1	Familiar with Rational UP	0	1.5	0
F2	Application experience	0	0.5	0
F3	Object-oriented experience	1	1	1
F4	Lead analyst capability	1	0.5	0.5
F5	Motivation	0	1	0
F6	Stable requirements	0	2	0
F7	Part-time workers	0	-1	0
F8	Difficult programming language		-1	
Environmental Factor (EF) = sum of weighted ratings				**1.5**
Environmental Value (EV) = 1.4 - (0.03 * EF)				**1.355**

Use Case Points				
Use Case Points (UCP) = UUCP * TCF * EV				**232.226675**