Software Requirements Specification

for

Fox Of Hood

Version 1.0 approved

Prepared by Grace Frizzell, Manila Aryal, Tamsyn Evezard, Ngoc Bui, Berkan Guven

Team 2

10/01/2024

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Grace Frizzell	09/17/24	Created & Initialized Document (all Sections)	1.0
Tamsyn Evezard	09/24/24	Updated Section 1	1.0
Tamsyn Evezard	09/25/24	Updated Section 2	1.0
Tamsyn Evezard	09/26/24	Updated Section 3, 5, Appendices	1.0
Grace Frizzell	09/30/24	Updated Section 4	1.0
Ngoc Bui	09/30/24	Created UI mock-ups	1.0
Manila Aryal	09/30/24	Updated Section 5	1.0
Tamsyn Evezard	09/30/24	Revised Sections 1-5	1.0
Berken Guven	10/01/24	Edited Section 4.3, 4.4, 5.4	1.0
Grace Frizzell	10/01/24	Editing Round	1.0
Tamsyn Evezard	10/01/24	Final Edit and Proof Read	1.0

1. Introduction

1.1 Purpose

This Software Requirements Specification document (SRS) defines all the features and requirements for the Fox of Hood (FOH) stock portfolio and trading simulator web application. This SRS document covers every subsystem of the FOH system, including the user management, stock trading, portfolio management, stock price management, administration, trend graphs and analysis, persistent storage, financial position, and API integration. FOH allows users to trade, sell, and adjust stocks whilst managing and viewing their own stock portfolio. FOH will simulate a stock trading platform whilst implementing multi-user login and an administration account.

The goal of this SRS is to define the elements of FOH in detail for the development team to build FOH successfully. This is version 1.0 of the FOH system specifically developed for the Hood College Principles of Software Engineering class of Fall 2024.

1.2 Document Conventions

From here on, the Fox of Hood system will be abbreviated as FOH. Each functional requirement is numbered as follows: REQ-1, REQ-2, etc., in order of priority. Requirements of special significance (high importance) will be bolded. Every requirement statement has its own priority.

“User” will be used to describe anyone who interacts with the system, while “admin” refers to the account run by the development team with administrative privileges that can create and delete user accounts. “Client” refers to the person or party that requested for this project to be created and provided the project description (in this case, Dr. Dimitoglou).

1.3 Intended Audience and Reading Suggestions

The intended audience and reading suggestions include the following in decreasing order of importance:

- **Developers and Project Managers:** to understand the requirements to successfully build FOH, and to track the project scope as development progresses.
    - See section 3 for details regarding system features.
- **Testers:** to verify the functionality of FOH compared to the SRS.
    - See section 3 for details regarding system features.
- **Client(s):** to evaluate the fulfillment of requirements and objectives of the project.
- **Users:** to understand the project features and how to use them.
    - See section 4 for a simplified representation of the user interface and features.

1.4 Project Scope

The FOH system is a web-based stock trading and portfolio simulator platform designed for the Principles of Software Engineering class at Hood College for Fall 2024. This interactive application will enable multiple users to trade stocks (buy, sell, and adjust), view and use up-to-date stock pricing (using an API), and analyze stock trends from their user portfolio, all through a web-based GUI. Each user can see 3 charts and graphs related to their financial position.

FOH provides detailed logging about activity (e.g. transaction and login history) during use and uses persistent storage. It incorporates a traditional authentication mechanism (e.g. username and password) and includes CAPTCHA functionality for an extra level of security against attacks. Administrator accounts will be able to manage (e.g. create and delete) user accounts. The ultimate goal is for our users to be exposed to a realistic experience of trading stocks through our simulator, deepening their understanding of the practice.

1.5 References

- Group, Term Project – Principles of Software Engineering – Fall 2024 *(George Dimitoglou, 27 August 2024, Blackboard)*
- API Documentation *(real-time stock data) (TBD)*

2. Overall Description

2.1 Product Perspective

The context and concepts specified in this SRS originated from other real-world stock trading and portfolio applications, however, FOH is a new, self-contained web-based application for Hood College's Fall 2024 Software Engineering class. The main functionality includes a stock portfolio simulator that allows users to trade (buy, sell, and adjust) stocks, view pricing (real-time), and analyze their stock activities (growth and decline) using charts and graphs. Detailed logging about user's activity is available. FOH will support multi-user logins with an administration account to align with appropriate standards for security.

See below for a simple diagram showing the major components of the overall system, subsystem interconnections, and external interfaces.

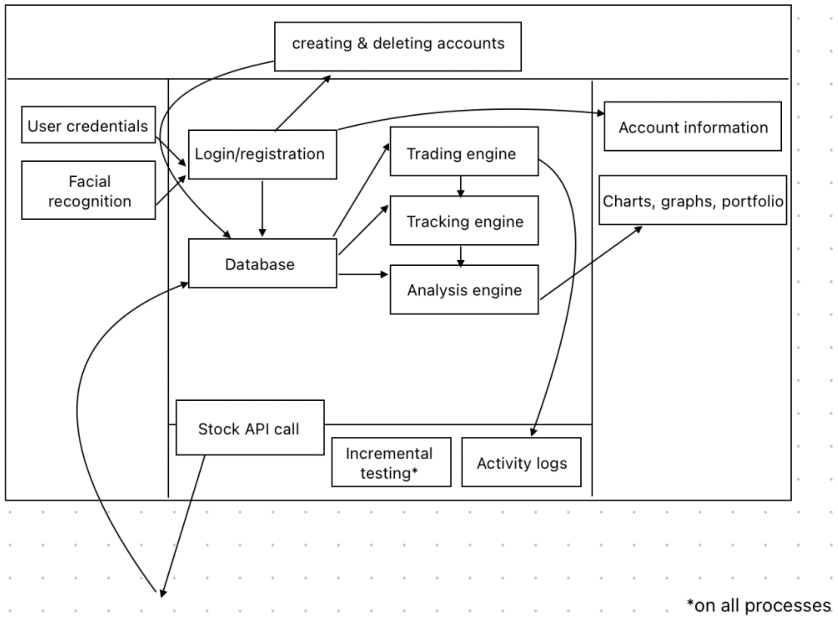

Figure 1 System Components and Connections

2.2 Product Features

The key features of FOH include:

- **Authentication:** a traditional username, password, and CAPTCHA.
- **Administration:** an administrative account with the permissions to create and delete user accounts.
- **Stock Portfolio Management:** viewing up-to-date stock prices and trading stocks.
- **Activity Logging:** login activities and transaction details (such as which stock was bought or sold, a time and date stamp, remaining balance and overall balance is available after each trade).
- **Trend Analysis:** charts and graphs to provide information about the user's financial position and stock growth and decline.

2.3 User Classes and Characteristics

FOH's primary user and client is Dr. Dimitoglou. He will have full access to the product features explained above, including administration. This will be essential for evaluation and assessment of the product's features and the level of objective fulfillment of FOH. On the other hand, the development team will have access to administration privileges to create and delete user accounts.

In addition, this product can be used by traders new to the practice with less experience, or casual traders, to get a feel of real-world trading with the basic features of the FOH simulator like buying and selling. These users will only have access to the user features and not the admin features. This is the most important user class to cater for, as the main objective of FOH is to be able to simulate stock trading for users. However, both classes of users are equally important to satisfy.

2.4 Operating Environment

The FOH simulator is a web-based platform that will run in a web browser. It is supported on desktop and mobile platforms. It requires browser compatibility, such as Chrome or Safari, and an operating system, such as Windows or macOS.

2.5 Design and Implementation Constraints

The components and design and implementation constraints associated with these components that may limit the options available to developers include:

- **Database:** SQL for storing user data, stock information, and activities.
- **Third-party API:** free API integration for up-to-date stock prices.
- **Security:** traditional username and password with a CAPTCHA feature for an additional authentication check.
- **Web technologies:** Python, HTML, CSS, JavaScript, Flask.

2.6 User Documentation

FOH will be designed so users are able to use the product without additional help. However, FOH documentation on GitHub will include various guides on how to use the software and its associated features as well as screenshots with stepwise instructions. We will include links to video demonstrations of certain functionalities to further help with ease of use.

2.7 Assumptions and Dependencies

- The free stock API used for the project will be free throughout the development and use of the project.
- Product features may be simplified due to the timeframe of development and academic environment.
- The developers and users must have a stable internet connection for up-to-date stock prices.

3. System Features

3.1 Authentication

3.1.1 Description and Priority

The authentication system for FOH will ensure users and admin can securely access the platform using their username, password, and CAPTCHA. This is a high priority feature.

3.1.2 Stimulus/Response Sequences

- When the user reaches the login page, they will be presented with text boxes labeled as username and password, with a CAPTCHA function to login to a preexisting account.
- Clicking “log in” after entering their credentials will take them to their portfolio page (home).
- If the user does not yet have an account, they will click “register” after entering their credentials to create an account and will be redirected to their portfolio page (home).
- For registration, user profiles will be created in the database and tied to the user’s information. For logging in, authentication information will be checked against that same database.

3.1.3 Functional Requirements

REQ-1: Implement a database for the persistent storage of user credentials (username and password).

REQ-2: Create secure sessions after a user logs in until they log out or the session times out after inactivity.

REQ-3: Implement logic for identical username prevention.

REQ-4: Validate password strength on the registration page to ensure the user’s chosen password satisfies the minimum length and character requirements (at least 8 characters including at least 1 special character and 1 number).

REQ-5: Implement a front-end login and registration page for secure access to the platform via username, password and CAPTCHA.

REQ-6: Give error feedback on invalid inputs to users when inputting username and passwords on the login and registration page.

3.2 Stock Portfolio Management

3.2.1 Description and Priority

The Stock Portfolio Management system will enable users to view and manage their stock portfolio, including their balance, stocks owned, and stock performance (stock growth and decline). The users will be able to view current stock prices, buy and sell stocks, and set up scheduled purchases. This is the highest priority feature.

3.2.2 Stimulus/Response Sequences

- After logging in, the user will be directed to their stock portfolio page where their overall balance and owned stock information will be displayed.
- Users can navigate to the trading page to view stock prices and trade stocks.
- Users can select stocks, buy, sell, and set up scheduled stock orders on the trade page.
- The portfolio will automatically update with new balance information after each transaction.

3.2.3 Functional requirements

REQ-1: Initialize each user's balance to $100,000 in their portfolio and do not allow users to spend more than their current balance.

REQ-2: Allow users to buy stocks with real-time prices.

REQ-3: Allow users to sell stocks.

REQ-4: Allow users to set and adjust scheduled stock purchases.

REQ-5: Display real-time stock prices (via API).

REQ-6: Display graphical representations of stock performance (see details in Trend Analysis).

3.3 Trend Analysis

3.1.1 Description and Priority

Trend Analysis will provide users with visual representations of the performance of their portfolio and individual stocks. This feature will be built into the stock portfolio page in the form of graphs and charts to track stock growth or decline over time. This is a medium-priority feature.

3.3.2 Stimulus/Response Sequences

- After each transaction (buying or selling stocks), the trend analysis charts will be updated to reflect changes in the user's financial position.
- The stock performance graphs will automatically update every 60 seconds, pulling real-time data from the API.

3.3.3 Functional requirements

REQ-1: Provide real-time graphical updates for stock price trends.

REQ-2: Provide a visual representation of the user's overall balance and how it changes over time.

REQ-3: Provide dynamic charts, showing stock growth or decline of stocks owned, that update every 60 seconds.

REQ-4: Provide graphs and charts that reflect transactions immediately after they occur.

3.4 Activity Logging

3.4.1 Description and Priority

The Activity Logging feature will record all user activities, including stock transactions (buy, sell, or update), login events, and account modifications. Details for each transaction will include the stock name, total price and amount, date and time of the transaction, and remaining balance after the transaction. This log will be accessible to the user. This is a high-priority feature.

3.4.2 Stimulus/Response Sequences

- After each action (buying or selling stocks, logging in or out), the user's logs will be updated, and a log entry will be created detailing the activity on the log page.

3.4.3 Functional requirements

REQ-1: Store event logs in a database.

REQ-2: Record and display user login and logout events.

REQ-3: Record and display all stock transactions (buy/sell).

REQ-4: Generate logs in chronological order.

REQ-5: Display a front-end log page for users and admins to view activities.

3.5 Administration

3.5.1 Description and Priority

Administrators will have limited access to user accounts, including the ability and privilege to create and delete user accounts. This is a medium-priority feature.

3.5.2 Stimulus/Response Sequences

- Administrators will log in with their own credentials to access the admin panel.
- From the admin panel, they will be able to create or delete user accounts.
- Admins will also be able to view logs of user activities and manage platform settings.

3.5.3 Functional requirements

REQ-1: Create a secure login for administration account.

REQ-2: Allow admin accounts to create new user accounts.

REQ-3: Allow admin accounts to delete user accounts.

4. External Interface

4.1 User Interfaces

All UI diagrams in this section are simplified and will be refined using Figma for prototyping as planning and visualizing progresses.

4.1.1 Login and Registration

The login and registration page will list username and password entry boxes as well as integrated CAPTCHA functionality, along with registration and log in buttons to trigger those events respectively. Tentatively, the login page will also feature a facial recognition option to be triggered by a button in lieu of the other information.

See the figure below for a simplified visual for the login and registration page.

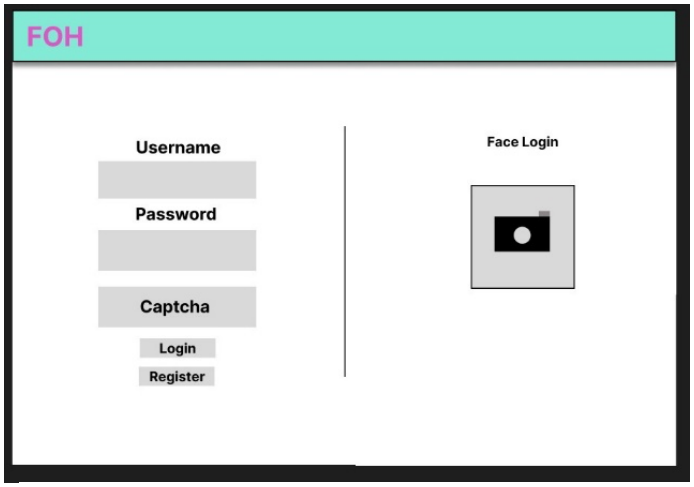

Figure 2 Login Page

4.1.2 Main (Home)

The main page will be the home page, featuring the user's portfolio of stocks and some analytical graphs and information to highlight particular trends in the behavior of each stock, specifically the stocks owned by the user (see Figure 3).

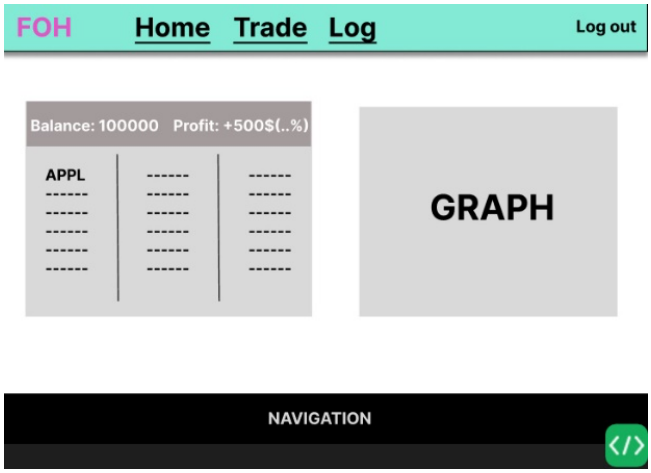

Figure 3 Home Page

4.1.3 Trade

The trading page will host all the trading functionality, featuring a list of stocks with up-to-date data, and the options to buy and sell stocks. Another field will be implemented for the user to choose a regular order of stocks. A simplified version is shown in Figure 4.

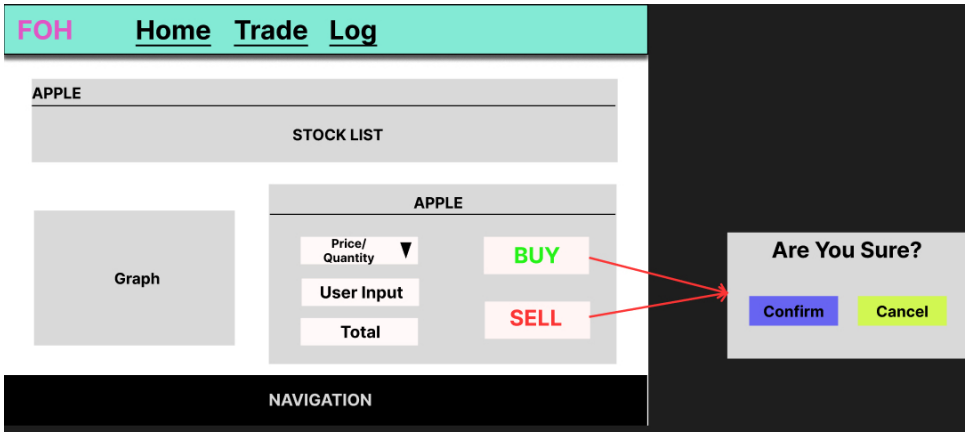

Figure 4 Trade Page

4.1.4 Log

The log or activity history page will feature a detailed log of user activity up to that point in time, including any transactional history.

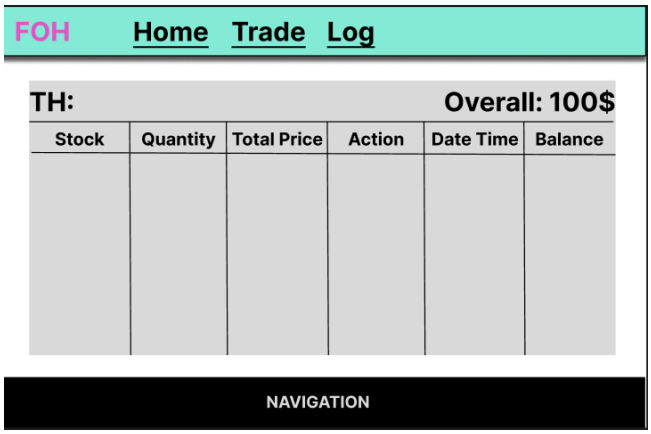

Figure 4 Log Page

The bottom of each page will feature an identical navigation list, wherein the user can navigate to the user documentation of the project (labeled as help), the contact information of each developer, and other pertinent information.

In the event of invalid user input, pop-up screens will be implemented to identify the error. For example, if they try to purchase more shares than their balance can afford, a message will be generated to identify this issue.

4.2 Hardware Interfaces

This is a web application designed to be accessible via the browser on a range of hardware devices. This includes standard desktop devices running operating systems including Windows and macOS, as well as mobile devices with iOS or Android systems. Any device with a modern web browser that supports HTML5, CSS, and JavaScript will be able to access FOH.

4.3 Software Interfaces

FOH will follow a typical client-server model, where the frontend interacts with the backend (which then communicates with the database and APIs).

- **Backend:** Python/Flask
- **Frontend:** HTML/CSS/JS with Bootstrap
- **Database:** SQL
- **API for stocks:** TBD

The design and building of the web application will be done using Python with Flask's framework. The app will use a SQL database to store data. The front-end of the app will use CSS with Bootstrap for styling. It will include popup forms, loading animations, and confirmation alerts to improve user experience.

4.4 Communications Interfaces

The Stock Simulator app will use **HTTPS** to keep all data exchanges secure, protecting user information like passwords, stock data, and transactions. Users will interact with the app through a web browser, and the app will connect to a stock price API to get real-time prices. The app will connect to a **PostgreSQL** database to store data. Communication between the app and users will use **HTTPS**, and data will be sent in **JSON** format to update stock prices and portfolios without refreshing the page. Passwords will be stored securely using **salted hashes**, and the app will use **SSL/TLS certificates** to ensure communication is safe.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

REQ-1: The system should handle up to 100 users simultaneously without the degradation of performance (open requirement) and should ensure fast recovery in case of system failures.

REQ-2: Stock prices must update every 60 seconds.

REQ-3: The system should allow for integration with external trading platforms, financial data providers, and other third-party systems.

REQ-4: The system should have an error handling or displaying mechanism which should provide meaningful error messages when an error occurs, or the system fails.

REQ-5: The system should provide the ability for users to interactively manage their portfolios and visualize portfolio metrics dynamically using charts and graphs.

REQ-6: The program must maintain the historical integrity of the stocks and transaction data.

5.2 Safety Requirements

There are no safety requirements for this project due to the simulation program involving faux (fake) money trading.

5.3 Security Requirements

REQ-1: Passwords should be encrypted in the database following the password policies such as minimum length, complexity, and periodic changes.

REQ-2: CAPTCHA should prevent hacking attacks (for example, brute-force login attacks).

REQ-3: SQL injection should be prevented through specific mechanisms.

REQ-4: Use of database encryption for transaction history and ensure secure access to databases with proper authentication and auditing is required.

5.4 Software Quality Attributes

For the **customer**, the most important attributes are:

- **Correctness**: The application operates according to customer specifications.
- **Availability**: The application will not exceed more than 1 minute of downtime per day.
- **Learnability**: The application will require less than 2 minutes of user training (i.e. reading the user documentation) to operate with fewer than 1 error for every hour of use.

For the **developer**, the most important attributes are:

- **Correctness**: The application's functionalities will operate according to the guidelines set in this document.
- **Maintainability**: The application will be modular in order to enable future functionalities beyond what is described in this document.
- **Reliability**: The application will not exceed more than 1 minute of downtime per day.

Other important attributes:

- **Security**: The app must protect user data, using HTTPS for communication and encrypting passwords.
- **Maintainability**: The app should be easy to update and fix, with clear and simple code for future developers.
- **Performance**: The app should handle many users and API requests without slowing down, even when there is a lot of activity.

6. Other Requirements

N/A

Appendix A: Glossary

Admin: the administration account that has access to creating and deleting user accounts.

API: Application Programming Interface (used to communicate external information with the application).

Figma: an online platform used for the prototyping of websites and other web-based platforms.

FOH: abbreviation for "Fox of Hood", the development codename for the project.

Portfolio: the collection of the stocks the user has invested in.

Stock: partial ownership of a company.

Stock Trading: Buying or selling stocks as a user.

UI: User Interface.

Appendix B: Analysis Models

N/A

Appendix C: Issues List

Pending requirements and issues to address in decreasing order of importance include:

- Details about setting a scheduled purchase of stocks as a user will be refined in the team meeting on **10/08/2024**.
- The system should handle a certain number of users simultaneously without the degradation of performance. Specifics of this pending requirement will be discussed in the team meeting on **10/01/2024**.
- Facial recognition should be an alternative option for user login. Implementation of this requirement is optional, and shall thus be decided at our development midpoint based on progress (by **10/25/2024)**.