Rupi Corp

Final Report

(https://github.com/ChubsB/AI-Based-Stock-Investments)

(Provide KayDrive link to the project here)

BY

Team Lead: Syed Hussain Imam/19773

Member(s): ***Ahmed Abdul Ghafoor 19735***

***Huzaifa Farooq 19766***

***Mohammad Razi Moosa 18663***

UNDER THE SUPERVISION OF

SUPERVISOR: Dr. Tahir Syed

CO-SUPERVISOR: -

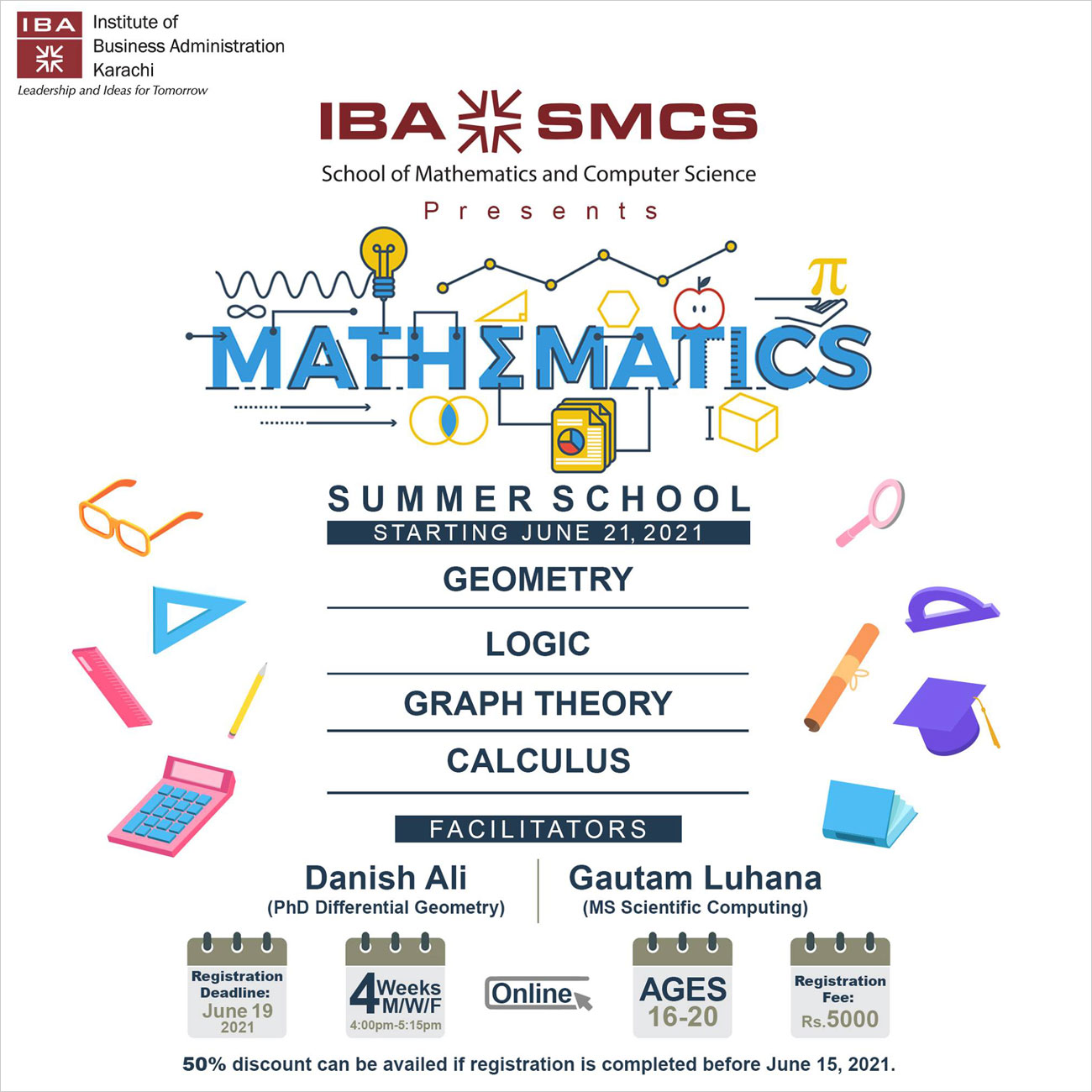
Member(s): NAME(S)

**SUBMITTED TO**

pROJECTS Manager – FYP

**ON**

DATE (01/05/2023)



(TO BE LEFT BLANK)

CONTRIBUTIONS

Each FYP team member should present their contributions to the project here in a separate matrix followed by their name in BOLD.

Details regarding:

* requirements specification (use cases and non-functional requirements),
* domain modeling (whole system or list the specific modules),
* software design (whole system or list the specific modules),
* report preparation (whole report or list the specific sections/diagrams),
* Other: any other relevant contribution.

Use extra pages if required. If more than one person has worked on a component jointly then specify the percentage each has contributed to that component.

Table of Contents

Proposal Defence Report

Insert earlier submitted and approved (final version) of project defense report here. (including title page and everything)

Software Requirement Specification (SRS)

# Problem Statement

The challenge of making well-informed investment decisions in the stock market has grown increasingly difficult due to the vast amount of information required to consistently outperform the market. Individual investors, in particular, heavily rely on publicly available news sources and sentiment analysis. Although the Efficient Market Hypothesis contends that all available information is already reflected in current stock prices, there is significant evidence suggesting that markets are not entirely efficient.

To capitalize on this inefficiency, our project aims to develop a stock advisory platform utilizing artificial intelligence techniques and a technical analysis approach, focusing on the psychological factors that influence future stock prices and trends. By evaluating and refining existing techniques, such as Moving Averages (MA), Auto Regressive Integrated Moving Average (ARIMA), and other AI-based methods, we strive to enhance the accuracy of stock price predictions and provide valuable insights for users.

Initially launched as a publicly available web application, the advisory platform will transition to a subscription-based model for individual and institutional investors. Offering comprehensive stock information for all stocks in the KSE-100 regular index, the platform will visualize prediction outcomes and furnish investment advice tailored to users' preferences. Furthermore, users can filter, select, and follow stocks of their choice.

Given the low level of individual investor engagement in the Pakistan Stock Exchange (PSX) compared to other countries, this platform also seeks to encourage greater participation in the PSX by delivering crucial information and guidance that is currently difficult for novice users to comprehend or locate in existing data portals. Our goal is to create an accessible and user-friendly tool that empowers individuals to make informed decisions in the stock market.

# System Requirement

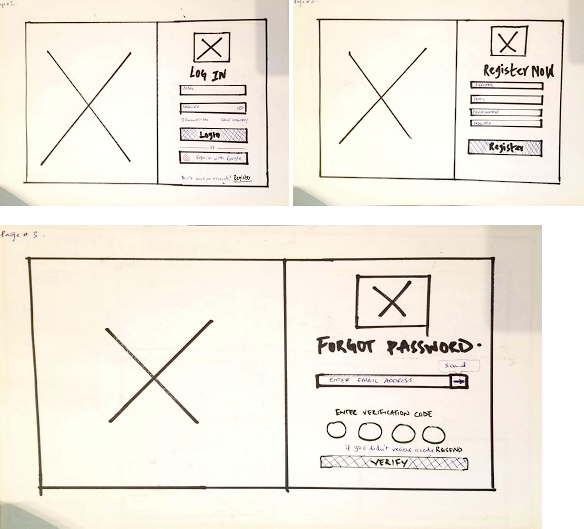
## List of Functional Requirements

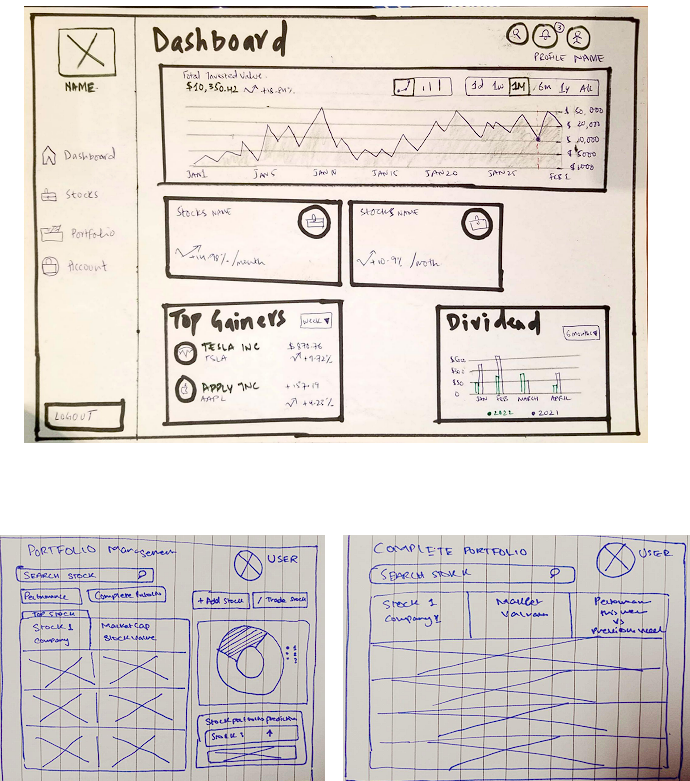
|  |  |
| --- | --- |
| **Functional Requirement No.** | **Functional Requirement Description** |
| FR01 | User Authentication - The system will be able to authenticate users through user accounts (Username/Email + Password), ensuring secure access to personalized data and features. |
| FR02 | User Registration - The system will provide a seamless registration process for new users, enabling them to create their account and access the platform. |
| FR03 | Stock Price Data Display - The system will display comprehensive stock price data for all stocks in the KSE-100 index, providing users with the information necessary for informed decision-making. |
| FR04 | Portfolio Creation - The system will allow users to create a custom portfolio, enabling them to manage and track their preferred stocks more efficiently. |
| FR05 | Portfolio Management - The system will enable users to add or remove stocks from their portfolio, ensuring that their investment preferences are always up to date. |
| FR06 | Stock Price Prediction - The system will offer stock price predictions for all stocks in the KSE-100 index, leveraging AI-based techniques and technical analysis to provide users with valuable insights. |
| FR07 | Portfolio Value Tracking - The system will display both the current and predicted future values of users' portfolios, enabling them to make informed decisions regarding their investments. |
| FR08 | Daily Stock Price Data Retrieval - The system will retrieve daily stock price data from reliable data sources, ensuring that users have access to the most recent and accurate information. |

## List of Non-functional Requirements

|  |  |
| --- | --- |
| Non-Functional Requirement No. | Non-Functional Requirement Description |
| NFR01 | Usability - The system will offer an intuitive and user-friendly interface, allowing users with varying levels of investment experience to easily navigate and utilize the platform. |
| NFR02 | Performance - The system will deliver prompt responses and maintain optimal performance levels, ensuring that users can access stock information, predictions, and portfolio data with minimal latency. |
| NFR03 | Scalability - The system will be designed to accommodate an increasing number of users and data processing requirements, enabling it to scale as the platform's user base and features expand. |
| NFR04 | Security - The system will implement robust security measures, including encryption and secure communication protocols, to protect users' personal and financial data from unauthorized access. |
| NFR05 | Reliability - The system will be designed with redundancy and error-handling capabilities to minimize downtime and ensure consistent availability of the platform. |
| NFR06 | Data Accuracy - The system will incorporate rigorous data validation and quality control processes to ensure that stock price information, predictions, and other data are accurate and up to date. |
| NFR07 | Maintainability - The system will be designed with modularity and adherence to best practices, enabling easy updates, bug fixes, and feature enhancements without causing significant disruptions to the platform's operation. |
| NFR08 | Compatibility - The system will be compatible with a wide range of devices and browsers, ensuring that users can access the platform from their preferred device and browser. |
| NFR09 | Responsiveness - The system will be designed with a responsive user interface, adapting to different screen sizes and resolutions, providing an optimal user experience across various devices. |

## User Interface Requirements





# Functional Requirements Specification

## Stakeholders

* Individual Investors - Users who will access the platform to manage their investments and seek reliable stock price predictions.
* Institutional Investors - Organizations that will use the platform for research and investment management purposes.
* Platform Developers - The team responsible for designing, implementing, and maintaining the stock advisory platform.
* Data Providers - Third-party sources that provide stock market data and relevant financial information.

## Actors and Goals

Actors:

* Registered Users - Users who have completed the registration process and can access the platform's features.
* Administrator - The individual(s) responsible for managing the platform, including user accounts, data updates, and system maintenance.

Goals:

* Provide users with accurate stock price predictions and investment advice.
* Enable users to create, manage, and track custom portfolios.
* Offer comprehensive stock information for all stocks in the KSE-100 index.
* Increase individual investor engagement in the Pakistan Stock Exchange (PSX).

## Use Cases

**Use Case 1: User Authentication**

Actor: Registered User

Description: The user logs into the platform using their credentials (Username/Email + Password).

Preconditions: The user has a valid account.

Postconditions: The user is granted access to personalized features, such as portfolio management and stock price predictions.

**Use Case 2: User Registration**

Actor: Guest User

Description: The user registers for a new account, providing necessary information such as username, email, and password.

Preconditions: The user does not have an existing account.

Postconditions: The user creates a new account and gains access to the platform's features.

**Use Case 3: Portfolio Creation**

Actor: Registered User

Description: The user creates a custom portfolio to manage and track their preferred stocks.

Preconditions: The user is logged in.

Postconditions: The user has a new portfolio to monitor their investments.

**Use Case 4: Portfolio Management**

Actor: Registered User

Description: The user adds or removes stocks from their portfolio.

Preconditions: The user has an existing portfolio.

Postconditions: The user's portfolio is updated to reflect their investment preferences.

**Use Case 5: Accessing Stock Price Predictions**

Actor: Registered User, Guest User

Description: The user views stock price predictions for all stocks in the KSE-100 index.

Preconditions: None.

Postconditions: The user obtains valuable insights for potential investments.

# User Interface Specification

For each use case specify the user interface. All reports generated by the system are also part of user interface specification.

# Domain Analysis

**Domain Model:**

**User:**

Attributes: UserID, Username, Email, Password, Portfolio

Relationships: Owns a Portfolio, Interacts with Stock, Views StockPricePrediction

**Portfolio:**

Attributes: PortfolioID, Name, User, Stocks

Relationships: Belongs to User, Contains multiple Stocks

**Stock:**

Attributes: StockID, StockName, Symbol, StockPrice, StockPricePrediction

Relationships: Belongs to Portfolio, Associated with StockPricePrediction

**StockPricePrediction:**

Attributes: PredictionID, Date, PredictedPrice

Relationships: Linked to Stock

**Mathematical Models:**

Moving Averages (MA): A simple method to smoothen stock price data, making it easier to identify trends.

Simple Moving Average (SMA): SMA = (P1 + P2 + ... + Pn) / n

Exponential Moving Average (EMA): EMA = (Close - Previous EMA) \* (2 / (n + 1)) + Previous EMA

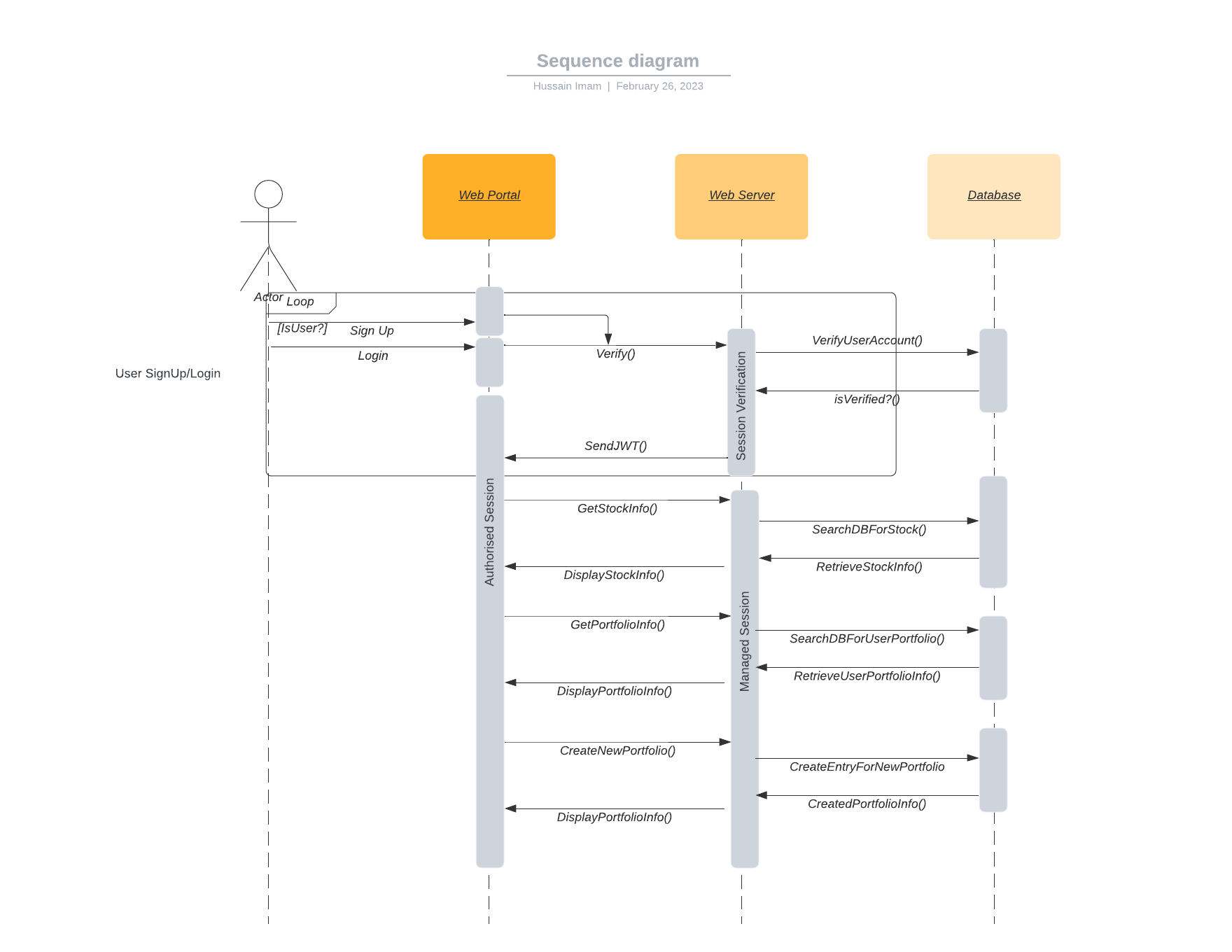
Auto Regressive Integrated Moving Average (ARIMA): A time series forecasting model that combines autoregression, differencing, and moving averages.

ARIMA(p, d, q): p = order of autoregression, d = degree of differencing, q = order of moving average.

AI-based Techniques: Various machine learning and deep learning algorithms can be used to predict stock prices, such as Long Short-Term Memory (LSTM) networks, Support Vector Machines (SVM), or Random Forests.

System Design

# Interaction Diagrams

.

# Class Diagram and Interface Specification

Show all classes and their associations. If cannot fit on a single-page, make an overview diagram showing all classes and their associations with just name. Details with attributes can be depicted in partial diagrams.

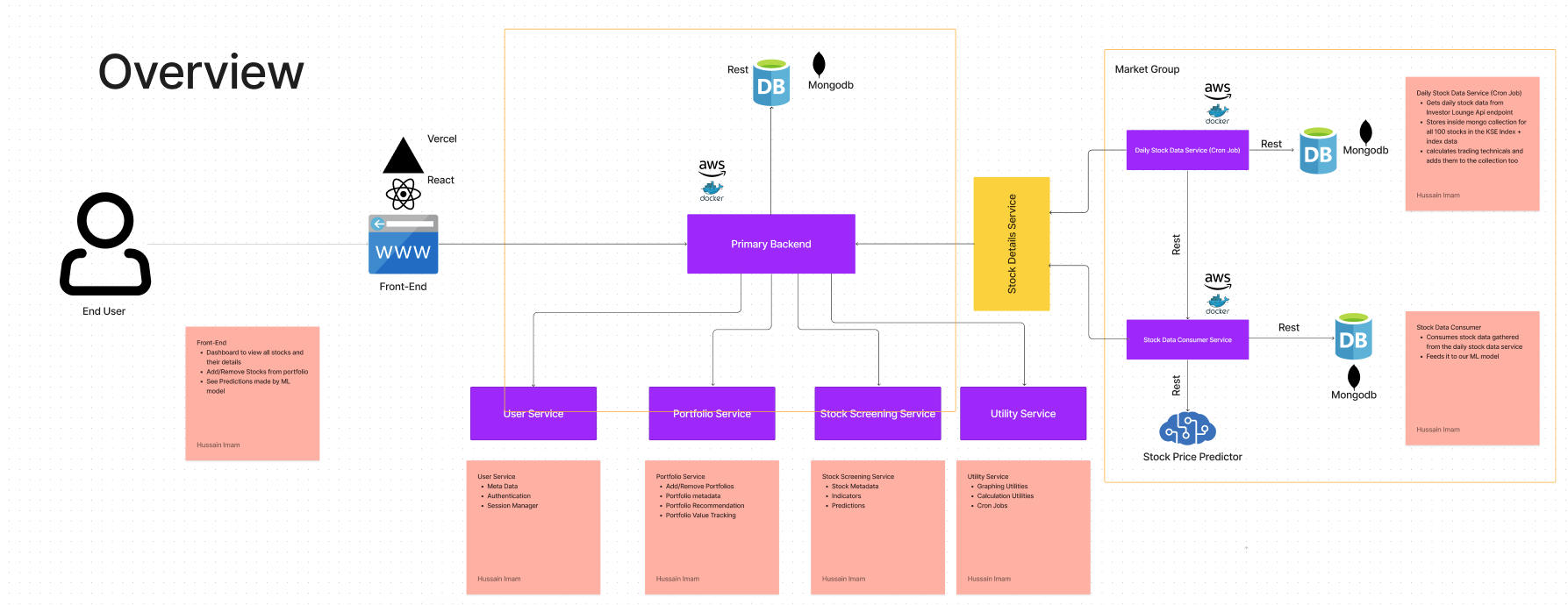
Use UML notation for class specification with datatypes for each class.

# System Architecture and System Design

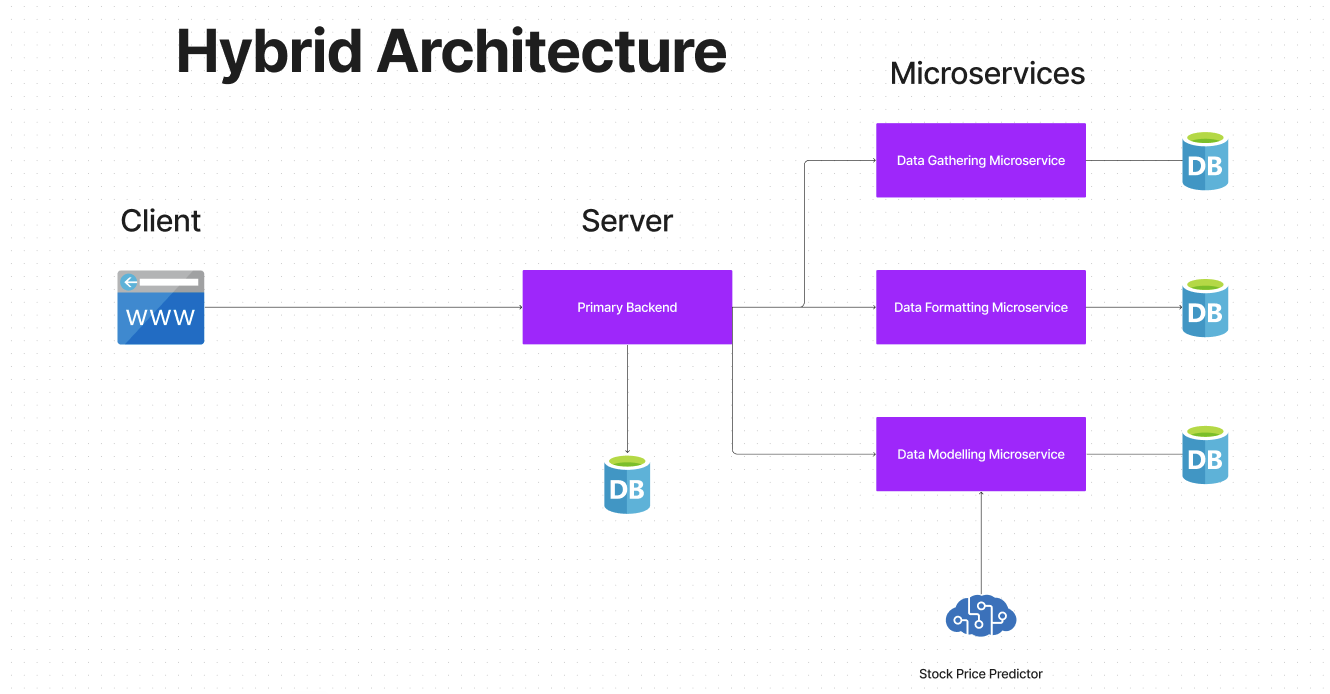
## Architectural Style

Client – Server Architecture: Our system is segregated into two major components our client (Browser Portal) and our server (Nodejs Backed) + Database (MongoDB/MySQL). Although for the most part our architecture is heavily centered around Client – Server fundamentals. Some of our services fall outside our monolithic backend and follow the microservice pattern i.e. Data collection, Data Formatting and Data modelling services

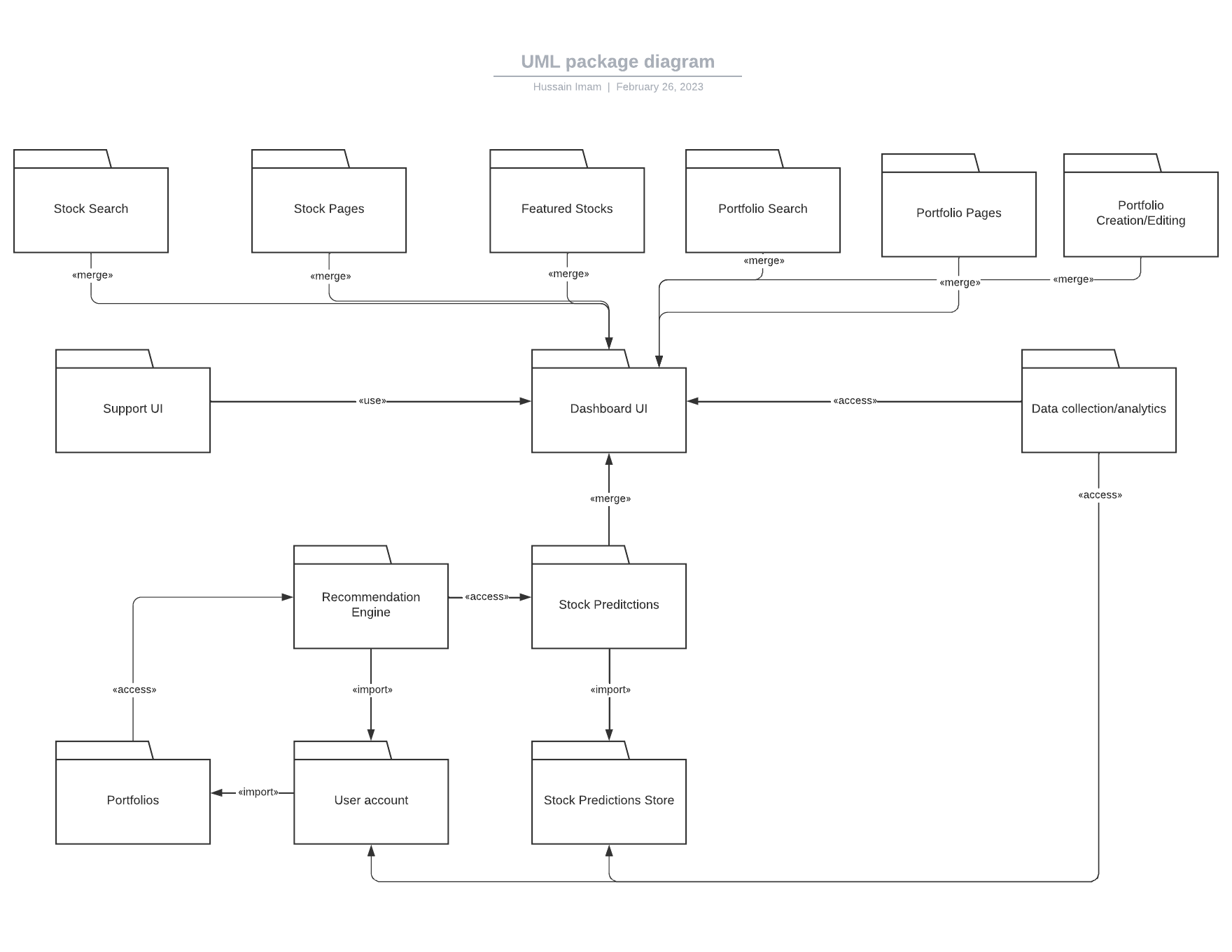
**Complete Architecture:**

****

**Client – Server + Microservices:**



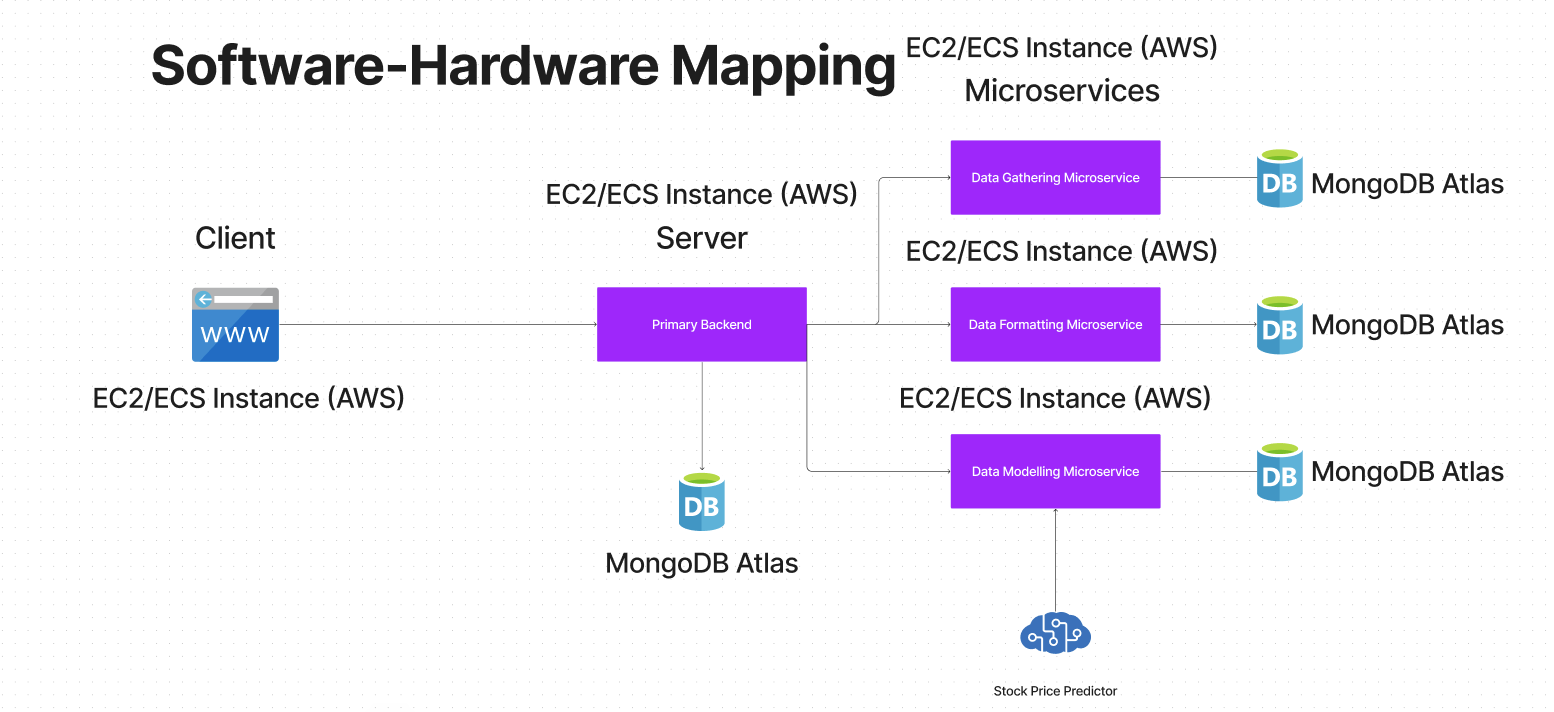
## Identifying Subsystems



## Mapping Subsystems to Hardware

Components in our architecture:

* Client (Reactjs Frontend): EC2 instance (AWS)
* Primary Backend (Nodejs): EC2 instance (AWS)
* Microservices: EC2 instance (AWS)
* Database: EC2 instance (AWS) / MongoDB Atlas



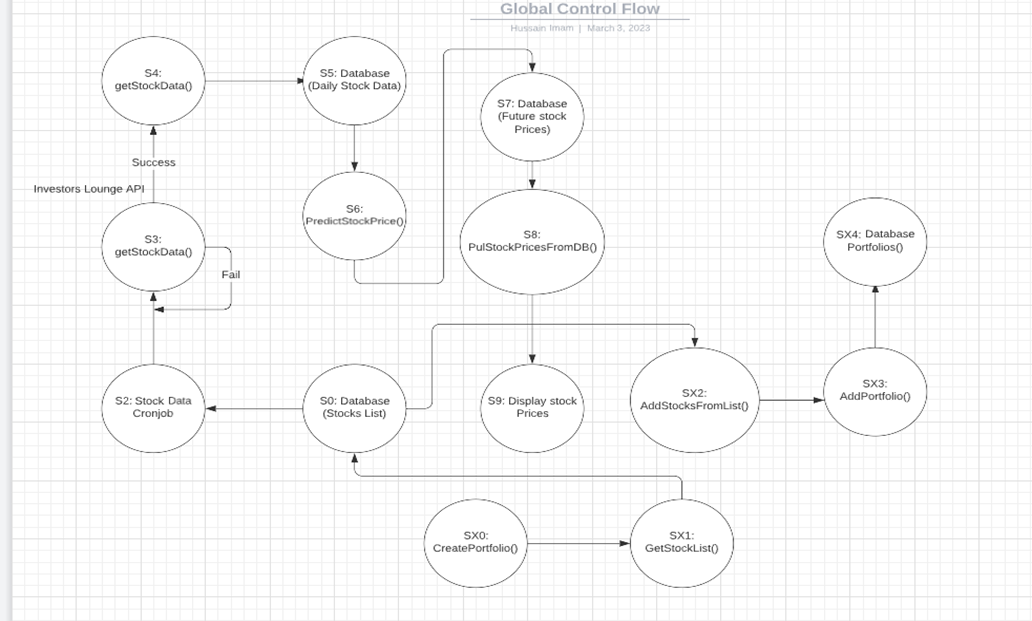
## Persistent Data Storage

Attach the description of the file format and/or database schema (format of database tables, printed by the command description)

## Network Protocol

Our application will be following the principles of RESTful web architectures; hence all communication will use HTTP/HTTPS.

## Global Control Flow



## Hardware/Sofware Requirements

**Recommended Operating Systems:**

Windows: 7 or newer

MAC: OS X v10.7 or higher

Linux: Ubuntu

**Hardware Requirements:**

We strongly recommend a computer fewer than 5 years old.

* Processor: Minimum 1 GHz; Recommended 2GHz or more
* Ethernet connection (LAN) OR a wireless adapter (Wi-Fi)
* Hard Drive: Minimum 32 GB;Recommended 64 GB or more
* Memory (RAM): Minimum 1 GB; Recommended 4 GB or above

**Supported Browsers:**

People often ask what browser they should use. There is no single answer for this. Use whichever browser works best on your computer. However, we recommend downloading Firefox and/or Chrome in addition to having Internet Explorer or Safari.

* FireFox
* Chrome
* Edge
* Safari
* Opera
* Brave

# Algorithms and Data Structures

Describe the complete algorithms your system use. Also specify and complex data structure (arrays, link lists, hash tables, trees etc.) your system uses.

# User Interface Design and Implementation

Describe how and if you modified the initial user design used earlier for ease of use.

# Design of Tests

Tests will be conducting using the Nodejs testing library called Jest (Testing Framework)

|  |  |  |  |
| --- | --- | --- | --- |
| Id | Test Case | Input Value | Expected Output |
| 1 | Sign Up | Email: [xyz@gmail.com](mailto:xyz@gmail.com)  Password: 1234Admin | Auth Token: JWT \*\*\*\* |
| 2 | Login | Email: [xyz@gmail.com](mailto:xyz@gmail.com)  Password: 1234Admin | Auth Token: JWT \*\*\*\* |
| 3 | Get stock data | Ticker: ABL | ABL Stock Data Object |
| 4  5 | Get Portfolio  Create Portfolio | Portfolio\_id: 3A21  PortfolioName: XYZ  Stocks: {  ABL: 25%  ENGRO: 25%  PIBTL: 50%  } | User Portfolio Object  Success: 200 (Reponse) |
| 6 | Edit Portfolio | Portfolio\_id: 3A21  Stocks: {  ABL: 55%  PIBTL: 45%  } | Success: 203 (Reponse) |

\*Frontend testing will consist of manual end-end testing according to specified user journeys.

User Manual

Not Required

Project Management and Plan

Present detailed comparison of the proposed timelines and the timeslines after execution of the project in Gantt charts here.

Current Status and Future Work

The current iteration of the project is an MVP for what can be, future iterations of our product would be making the current tooling better, obtaining licenses to hold and invest customer money to automate management of their portfolios.

References

Specify all resources including journal articles, books, reports and any other resource(s), including online, consulted during the course of work.