**INFS 740: Final Project (StockSphere)**

Abhishek Jallawaram (G01373042)

Teja Sajja (G01395837)

Table of Contents

[**Project Overview:** 2](#_Toc164827040)

[**Summary:** 2](#_Toc164827041)

[**Key Objectives:** 2](#_Toc164827042)

[**Project Tech Stack:** 3](#_Toc164827043)

[**React** 3](#_Toc164827044)

[**FastAPI** 3](#_Toc164827045)

[**MongoDB** 4](#_Toc164827046)

[**System Architecture:** 4](#_Toc164827047)

[**Admins:** 4](#_Toc164827048)

[**Customers:** 4](#_Toc164827049)

[**Database Schema:** 6](#_Toc164827050)

[**Collections:** 6](#_Toc164827051)

[**Relationship between Collections:** 6](#_Toc164827052)

[**API Documentation:** 6](#_Toc164827053)

[**All END POINTS:** 6](#_Toc164827054)

[**SAMPLE DATA:** 6](#_Toc164827055)

[**Functionalities** 6](#_Toc164827056)

[**CRUD Operations:** 6](#_Toc164827057)

[**Complex Queries:** 6](#_Toc164827058)

[**Visualizations:** 6](#_Toc164827059)

[**ML Prediction Model:** 6](#_Toc164827060)

[**UI Design And Screenshots** 6](#_Toc164827061)

[**Member Contributions** 6](#_Toc164827062)

[**Challenges & Future Work** 6](#_Toc164827063)

[**Conclusion:** 6](#_Toc164827064)

GitHub link: <https://github.com/abhishekjallawaram/StockSphere>

**Project Overview:**

### **Summary:**

The purpose of this stock and crypto management platform is to provide an efficient and user-friendly system for monitoring and managing stock and cryptocurrency transactions. The platform enables users perform buy and sell stocks & cryptocurrencies. The admin can analyze the stocks, cryptocurrencies, agents, customers, and transactions to retain the customers and ensure to maintain a comprehensive tool for financial data analysis and management.

### **Key Objectives:**

1. **Data Management:** Implemented data management capabilities that allow users to access the data. (CRUD operations).
2. **Search Queries:** Advanced search queries implemented that allows users to retrieve data from multiple collections based on specific parameters. This feature aims to facilitate deeper insights into transactions and relationships between different data points.
3. **Data Visualization:** Integrated data visualizations that offer graphical representations of financial data, enhancing the user's ability to analyze trends over time and make informed decisions.
4. **User Interface:** Developed a responsive and intuitive user interface that caters to both novice and experienced users, ensuring ease of navigation and accessibility of features.
5. **Real-Time Updates:** Utilized yfinance API to retrieve real-time data which fetches and updates the data, providing users with the most current information available and enhancing the platform's responsiveness.
6. **ML prediction:** A prediction model to predict the close price for a given stock.

### **Project Tech Stack:**

#### **React**

1. We have utilized the react components to build the frontend components of the application.
2. React has a component-based structure allows for reusable UI components, making the development process more efficient and the codebase more maintainable.
3. React provides virtual DOM which minimizes the number of costly DOM manipulations by batching updates, enhancing the performance of the application.
4. React has a vast community and a rich ecosystem of libraries and tools, providing abundant resources for solving common development challenges.

#### **FastAPI**

1. We have utilized the fastAPI to build the backed APIs needed to communicate with the MongoDB database and fetch the complex queriers.
2. FastAPI is one of the fastest web frameworks for Python, thanks to its Starlette framework for the web parts and Pydantic for the data parts.
3. Native support for asynchronous request handling, which is essential for handling multiple data-intensive operations like fetching large datasets from a database.
4. FastAPI automatically generates interactive API documentation (using Swagger UI and ReDoc), which simplifies API testing and frontend integration.

#### 

#### 

#### **MongoDB**

1. We have utilized the MongoDB as the primary database to store and manage structured and unstructured data of the platform.
2. MongoDB's schema-less nature allows for flexible and dynamic data models, which is beneficial for handling diverse datasets such as stocks and cryptocurrencies.
3. MongoDB offers horizontal scalability through sharding, effectively managing large volumes and high throughput of data.
4. MongoDB provides powerful querying capabilities, making it suitable for executing complex searches and aggregations required by the platform.

## **System Architecture:**

**Admins:**a) The admin role has access to all the records and performs all the operations on the resources accessed.

b) Manages the system by performing CRUD (creating, updating, or removing user customers, agents, stocks, crypto and transactions.)

c) Visualize data to retain customers, provide incentives to agents based on performance and offers to premium customers.

**Customers:**

Customers can log in to view their own balance, check out the stocks or cryptocurrencies they wish to buy or sell.

A diagram of a data flow

Description automatically generated

Figure1: System Architecture

## **Database Schema:**

### **Collections:**

### **Relationship between Collections:**

## **API Documentation:**

### **All END POINTS:**

### **SAMPLE DATA:**

## **Functionalities**

### **CRUD Operations:**

### **Complex Queries:**

### **Visualizations:**

### **ML Prediction Model:**

## **UI Design And Screenshots**

## **Member Contributions**

## **Challenges & Future Work**

## **Conclusion:**