Text

Description automatically generated with medium confidence

Finance Analysis System

Service Oriented Architecture

Nazif Selimi(129248)

Dardan Osmani(129239)

1. **Introduction**

* **Problem Statement:**

Managing a stock portfolio and analyzing financial data can be difficult for both beginner and experienced investors. Many existing tools are either too complicated or don’t provide enough useful features. This makes it hard for investors to keep track of their stocks, understand their performance, and make informed decisions. There is a need for a platform that is easy to use but still powerful enough to help investors manage their portfolios effectively and analyze their financial data.

* **Motivation:**

The goal of creating this Financial Data Platform is to make portfolio management and financial analysis easier. As the stock market becomes more complex, investors need tools that help them quickly track their stocks, analyze trends, and make smart investment choices. This platform is designed to provide a simple yet powerful solution for investors at any level, combining ease of use with advanced features.

* **Background:**

This platform was built using .NET Web API for the backend and React for the frontend. It uses modern technology to create a secure, fast, and scalable platform for users. The platform includes real-time stock data, detailed financial analysis, and portfolio management features that let users easily interact with their investments. It’s designed to be flexible and high-performing, giving users the tools they need to make well-informed decisions.

1. **Related Work**

There are various financial platforms and tools available today that help investors manage their portfolios and analyze financial data. Some well-known examples include:

* **Yahoo Finance**

Yahoo Finance is a widely used platform that offers stock tracking, real-time market data, financial news, and basic portfolio management. While it is a powerful tool for staying up-to-date with market trends and individual stock performance, it lacks more advanced financial analysis features and customization options for serious investors.

* **Morningstar**

Morningstar provides in-depth financial analysis, research reports, and portfolio management tools. It is well-regarded for its comprehensive data and research capabilities. However, the platform can be difficult for less experienced users to navigate, and the advanced features often come with a steep learning curve.

* **Personal Capital**

Personal Capital offers users the ability to manage their personal finances, including investments, with a focus on tracking overall net worth. While it provides good insights into portfolio performance, its main focus is on personal financial planning rather than in-depth stock analysis, limiting its usefulness for those looking to focus primarily on their investment portfolio.

* **Robinhood**

Robinhood is an easy-to-use app that allows users to buy and sell stocks with no commission fees. Its simple interface makes it popular among beginner investors. However, it lacks advanced financial analysis tools, making it less appealing to those who need more detailed insights into their investments.

* **Summary**

While these platforms provide various features for financial management and analysis, they either lack the simplicity for beginners or the advanced features needed by serious investors. Our Financial Data Platform aims to bridge this gap by providing a user-friendly interface combined with powerful analysis tools, making it suitable for a wide range of investors.

1. **Tools and Technologies**

* **.NET Web API**

The frontend of the platform is created using React, a tool that makes it easy to build user interfaces. We also used TypeScript, which helps catch errors early and makes the code easier to manage. React allows us to build reusable components, making the platform interactive and efficient.

* **React with TypeScript**

The frontend of the platform is created using React, a tool that makes it easy to build user interfaces. We also used TypeScript, which helps catch errors early and makes the code easier to manage. React allows us to build reusable components, making the platform interactive and efficient.

* **Microsoft SQL Server**

We use Microsoft SQL Server to store all the important data, like user information and stock details. SQL Server ensures that the data is safe, organized, and easy to retrieve when needed. It also allows us to handle complex data requests

quickly and reliably.

* **JWT (JSON Web Token) for Authorization**

For security, we use JWT (JSON Web Token) to manage user authentication. This makes sure that only authorized users can access their portfolios and personal data by using a token system. JWT helps keep the platform secure and protects user information.

1. Nonfunctional Mockup:

A cell phone with a screen showing a graph

Description automatically generated

A screenshot of a website

Description automatically generated

A screenshot of a web page

Description automatically generated

A screenshot of a website

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

1. ER Diagram:

Entity-Relationship Diagram:

A diagram of a computer

Description automatically generated

1. System Details and Diagrams

* **Abstract Implementation Details**

**System Architecture:** The platform is designed using a **service-oriented architecture** (SOA), where different components are organized in a modular fashion to ensure flexibility, scalability, and maintainability. The core structure includes:

1. **Models**: These define the structure of the data stored in the database, such as stock information, portfolio details, and user data.
2. **Repository Layer**: This layer is responsible for interacting with the database, providing methods to query, update, and delete data. It abstracts the database operations, ensuring that the service layer remains independent of the underlying database technology.
3. **Service Layer**: This is the core of the application logic. It processes the data received from the repository layer and interacts with the API to fetch new stock data when required. It also manages the business logic and rules of the platform.
4. **DTOs (Data Transfer Objects)**: These are used to transfer data between different layers of the application, ensuring that only the required data is passed around, improving security and performance.
5. **Controllers**: The controllers handle the incoming HTTP requests from the frontend, communicate with the service layer, and send appropriate responses back to the client.

* **Data Flow**

1. The **React frontend** sends requests to the backend API endpoints.
2. The **backend controllers** handle these requests and pass the data to the service layer.
3. The **service layer** retrieves data from the database using the repository layer. If the requested stock data is not available in the database, the service layer fetches it from the **Financial Modeling Prep API** using an API key.
4. The new stock data is then stored in the database, and the response is sent back to the frontend for display.

This architecture ensures a clear separation of concerns, making the system easier to maintain and scale over time.

1. User Types and Capabilities

In this platform, there are no distinct user roles or types. All users have the same capabilities, which include:

* **Adding and removing stocks** to and from their portfolio.
* **Viewing stock performance** and conducting detailed financial analysis.
* **Interacting with real-time financial data**, which is either fetched from the database or, if missing, retrieved from an external API and stored for future use.

1. Discussion

**System Capabilities**

The Financial Data Platform effectively solves the key problems identified earlier by making it easier for users to manage their portfolios and access accurate stock data.

**Problem-Solving Capabilities**

The platform simplifies portfolio management by allowing users to quickly add and remove stocks from their portfolio. It also ensures that users have up-to-date stock data by automatically retrieving missing information from the **Financial Modeling Prep API** and storing it in the database.

**Key Features**

Some of the platform’s important features include:

* **Real-Time Stock Data**: The platform integrates with the Financial Modeling Prep API to provide users with the latest stock data, ensuring they always have accurate information.
* **Simple Portfolio Management**: Users can easily manage their portfolios by adding or removing stocks, with the system tracking the performance of these investments.
* **Financial Metrics**: The platform gives users access to key financial data like dividends and market capitalization to help them make smarter investment decisions.
* **Database Storage**: All stock data and user portfolios are securely stored in a **Microsoft SQL Server** database, making the system reliable and efficient.

These features make the platform a valuable tool for investors, helping them manage their portfolios more easily and make informed decisions.

1. Future Work & Conclusion

**Limitations**

While the Financial Data Platform provides robust tools for portfolio management and financial analysis, there are some limitations in the current system:

* **Limited User Roles**: Currently, there are no user roles such as admin or premium users, which could offer advanced features or controls.
* **Real-Time Updates**: Although the platform fetches real-time stock data when missing, it does not continuously update stock prices after the initial retrieval.
* **Basic Financial Analysis**: The platform provides useful financial metrics, but more advanced analysis tools, such as predictive models or detailed performance comparisons, are not yet implemented.

**Future Work**

To enhance the platform further, several areas of improvement and expansion can be explored:

* **User Roles and Permissions**: Introduce different user roles, such as administrators or premium users, to provide varying levels of access and more advanced features for specific user types.
* **Real-Time Stock Updates**: Implement real-time stock price updates and notifications to keep users constantly informed of changes in their portfolio.
* **Advanced Financial Tools**: Add more advanced financial analysis tools, such as predictive modeling, trend analysis, and portfolio optimization recommendations.
* **Mobile Application**: Develop a mobile version of the platform to allow users to manage their portfolios and access stock data on the go.
* **Integration with More APIs**: Expand the platform by integrating with additional financial APIs to offer even more data sources and insights.

1. Conclusion

The Financial Data Platform successfully addresses the need for a simple yet powerful tool for managing stock portfolios and conducting financial analysis. By providing users with an easy-to-use interface, real-time stock data, and the ability to manage their portfolios effectively, the platform has made it easier for investors to track their investments and make informed decisions.

This project has demonstrated the importance of combining usability with advanced features in financial software. The platform has the potential to grow further by adding more user roles, real-time updates, and advanced tools, ensuring that it continues to meet the evolving needs of its users.