Software Requirements Specification

for

Stock Market Analysis

Version 1.0 approved

Prepared by

Ojasvni Jain

Vamshidhar Dawoor

Neeraj Shyam Sundar

Anuj Mishra

Aviral Sharma

8 July 2024

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

The Stock Market Analysis System's software requirements, including the revision or release number, are detailed in this document. This document's scope encompasses the entire system, emphasizing both business management procedures and features unique to individual users. Data storage in a MySQL database, data manipulation with Pandas, data visualization with Matplotlib/Plotly, and integration of real-time financial data from the API are all included in this. It describes the features that end users may use to analyze the market and the CRUD functions that are accessible to manage organizations in the system.

1.2 Document Conventions

This SRS follows a standard format with specific typographical conventions:

Headings: Bold and numbered for clear section separation.
Requirements: Numbered and prioritized with a unique identifier.
Code and API references: Monospaced font for differentiation.
Highlights and notes: Italicized for emphasis. Higher-level requirements are inherited by detailed requirements unless specified otherwise.

1.3 Intended Audience and Reading Suggestions

This document is intended for various stakeholders involved in the project:

- **Developers**: For understanding the technical requirements and system functionalities.
- **Project Managers**: For tracking project progress and ensuring requirements alignment.
- **Marketing Staff**: For identifying key features and benefits for promotional activities.
- **Users**: For understanding the capabilities and usage of the system.
- **Testers**: For designing test cases based on specified requirements.
- **Documentation Writers**: For creating user manuals and help guides.

The document is organized into sections, starting with an overview, followed by detailed requirements, system architecture, and data handling processes. Readers are advised to begin with the overview sections and proceed to the detailed requirements that are most relevant to their roles.

1.4 Project Scope

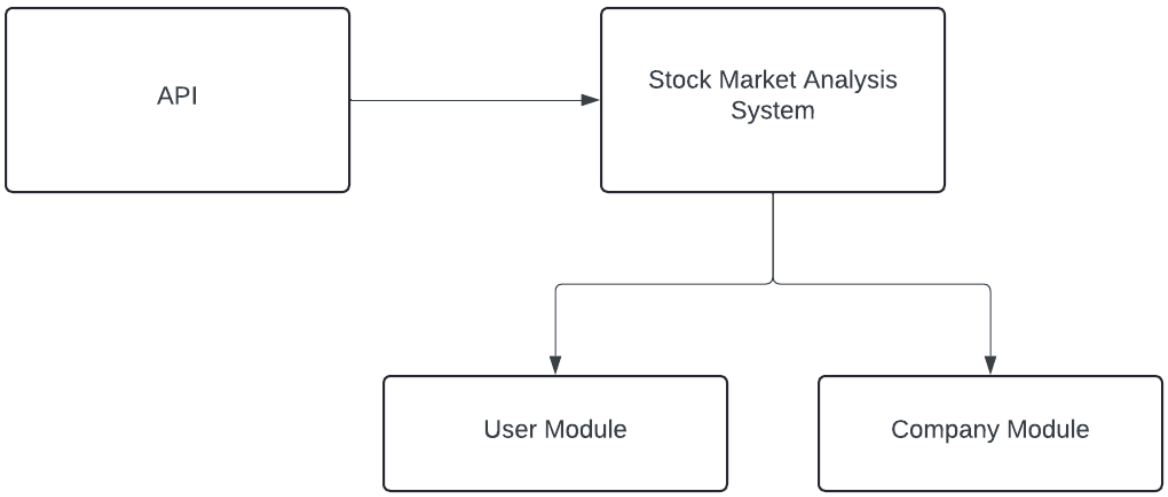
The Stock Market Analysis System has been designed to offer functionalities for firm management and extensive market analysis tools. It allows users to leverage real-time data from the API to retrieve, visualize, and forecast stock performance. With interactive charts and graphs, users may compare stock values across several periods (daily, weekly, and monthly). The solution provides CRUD activities in a MySQL database for business management, guaranteeing smooth data integration and upkeep. The project's use of cutting-edge data visualization and analysis tools to improve user engagement and facilitate data-driven decision-making is by company objectives. This document outlines the functionalities and

system requirements needed to accomplish these goals, acting as a thorough roadmap for the upcoming release of this developing product.

2. Overall Description

2.1 Product Perspective

The Stock Market Analysis System is a new, self-contained system that offers real-time stock market analysis and business management capabilities. It is neither a continuation of a product family nor a replacement for existing systems. The system combines several technologies to provide a comprehensive solution for financial analysis. It consists of a RESTful API (developed with Flask), data manipulation (using Pandas), data storage (MySQL), and data presentation (Matplotlib/Plotly). The system uses the API to retrieve real-time financial data and communicate with a MySQL database for company management. Here's a simplified diagram of the system architecture:



2.2 Product Features

The Stock Market Analysis System has the following key features:
Real-Time Data Retrieval: Obtain real-time financial data using the API.
Market Analysis: Allows detailed market analysis by inputting individual business keys to retrieve, visualize, and anticipate stock performance.
Comparative Analysis: Interactive charts and graphs allow for a comparison analysis of daily, weekly, and monthly stock prices.
Company Management: Provides CRUD functions to manage company data in a MySQL database.
Data Visualization: Uses Matplotlib/Plotly to visualize data straightforwardly and exactly.

RESTful API: Allows for data retrieval and manipulation using a RESTful API created with Flask.

2.3 User Classes and Characteristics

The system expects the following user classes:
End-Users: Individuals or organizations conducting market research. They may use the system regularly and need access to interactive charts and stock forecasts.
Company Administrators: Users that manage CRUD processes for companies. They require a greater level of access to update and manage firm information.
Developers: Developers are users who will maintain and improve the system. They require thorough technical documentation and access to the underlying software.
Testers: Users in charge of ensuring the system's functionality and stability. They need access to testing environments and thorough requirement descriptions.

2.4 Operating Environment

The software will run in the following environments:
Hardware Platform: Standard server hardware that can run web and database servers.
Operating System: Compatible with major operating systems such as Windows, macOS, and Linux.
Database: MySQL is the database used for robust data storage.
Other Software Components: Python is required for data manipulation (Pandas), web server (Flask), and data visualization (Matplotlib/Plotly).

3. System Features

3.1 Data Fetching

The system will use the API to retrieve real-time stock data. This functionality will be created to enable efficient and reliable data fetching, as well as to gracefully handle potential API errors and retain data integrity. Fetched data will be converted to CSV format for storage and analysis, making it easier to manipulate and integrate with other programs.

3.2 CRUD Operations

CREATE:

The system will allow for the development of new stock data entries in a MySQL database. A well-defined database schema will be created to record critical information including the firm name, stock price, trading volume, and date of data input. This module will allow authorized users to accurately insert new data into the database, ensuring that all recorded information is comprehensive and accurate.

READ:

Users will be able to retrieve stock data from the MySQL database using specified criteria. Queries can be created with parameters such as firm name or date

range, allowing users to obtain historical data or specific stock information as needed for research or reporting.

UPDATE:

The technology will allow users to change existing stock data entries. This involves resolving errors in previously entered data and making changes for occurrences like stock splits. Updates will be carried out securely to ensure data consistency and accuracy across the database.

DELETE:

To maintain data relevance and integrity, users will have the ability to delete old or irrelevant stock data entries from the database. This feature will help to streamline data management operations and ensure that only current and relevant information is stored in the system.

3.3 Comparative Analysis

The system will allow for comparative stock price analysis across a variety of periods, including daily, weekly, and monthly. Users will be able to see and compare financial data such as moving averages, the Relative Strength Index (RSI), and Moving Average Convergence Divergence (MACD). These comparisons will provide insight into long-term stock performance trends, which will help with investment decisions and strategic planning.

3.4 Visualization

Graphs and tables created with Matplotlib and Plotly will present analytical insights derived from stock data. Users can examine data trends and patterns dynamically using interactive charts, and static visualizations provide clear representations of key metrics and performance indicators. These visualizations will be customizable based on user inputs, such as selecting specific organizations for examination, hence improving the clarity and usability of the analytical results.

4. External Interface Requirements

4.1 User Interfaces

The user interface for the Stock Market Analysis System will have the following logical features and components:

- **Sample Screen Images**: Screens will be created to give users straightforward and intuitive access to capabilities like data collecting, CRUD operations, and visualizations.
- **Screen Layout Constraints**: The layout will prioritize usability, making critical features easy to reach. Menus and navigation bars will be positioned for easy use.
- **Error Message Display Standards**: Error messages shall be clean and succinct, giving the user useful information. Errors will be displayed consistently, usually as a pop-up or inline notice near the source of the mistake.
- **User Interface Components**: The system will need interfaces for:
    - **Data Fetching**: Screens for entering API keys, choosing companies, and specifying date ranges for data retrieval.

- **CRUD Operations**: Interfaces for storing, retrieving, updating, and removing stock information.
- **Comparative Analysis and Visualization**: Interactive charts and tables for financial metrics created using user input.

4.2 Hardware Interfaces

The Hardware Interface requirements for the Stock Market Analysis System are as follows:
Supported Device Types: The system will work with computers, laptops, tablets, and possibly mobile phones.
Data and Control Interactions: The system will largely communicate with hardware components via standard input (keyboard and mouse) and output (display screens) modes.
Communication Protocols: The system will use HTTP/HTTPS to communicate with the API and other external services. It will also use conventional database connectivity protocols (such as JDBC for MySQL).

4.3 Software Interfaces

The Stock Market Analysis System's software interfaces will connect to the following software components and services:
Databases: The system will connect to a MySQL database to store and retrieve stock data. It will use SQL queries to perform CRUD tasks.
Operating Systems: The system will be interoperable with the Windows, macOS, and Linux operating systems.
Tools and Libraries: Key libraries include:
Matplotlib and Plotly: Matplotlib and Plotly are key libraries for creating static and interactive visualizations.
Financial Analysis Libraries: Financial analysis libraries include pandas and NumPy for data manipulation and analysis.
Integrated Commercial Components: The system will work with the API to retrieve stock data.
Data Items and Messages:
Incoming: API answers with stock data.
Outgoing: Outgoing operations include database queries and API calls.
Services Needed: The system will require services to facilitate secure API connectivity, database access, and data presentation.
Communication Details: Detailed application programming interface protocols will be specified separately, outlining how the system will communicate with the API and the database.

4.4 Communications Interfaces

The Stock Market Analysis System has the following communication requirements:
Web Browser: The user interface will be accessible through web browsers (such as Chrome, Firefox, and Safari).
Network Server Communications: The system will employ HTTP/HTTPS for secure communication with the API and client-server interactions.
Message Formatting: JSON will be the standard format for messages sent to the API.
Communication Standards: The system will use standard protocols such as REST for API interactions and SQL for database queries.

Security and Encryption: To maintain data security, all communications will be sent over HTTPS. Sensitive information, like API keys and user credentials, will be encrypted.
Data Transfer Rates and Synchronization: The system will be built to handle different data transfer rates, as well as procedures that successfully sync data fetching and database updates.

5. Nonfunctional Requirements

5.1 Performance Requirements

The Stock Market Analysis System must meet the following performance requirements to ensure efficient and responsive operation:
Data Retrieval Efficiency: The system should be capable of retrieving stock data from the API within 5 seconds for individual company requests and within 30 seconds for batch requests (up to 10 companies).
CRUD Operations Responsiveness: Database operations (Create, Read, Update, Delete) should complete within 2 seconds for individual records and within 10 seconds for batch operations (up to 100 records).
Real-Time Data Processing: When analyzing and visualizing data, the system should process and display results within 3 seconds for daily data, 5 seconds for weekly data, and 10 seconds for monthly data intervals.
Scalability: The system should handle up to 1,000 simultaneous users without degradation in performance, with an average response time of less than 2 seconds for all operations.
Concurrent Users: The system must support at least 500 concurrent users performing various operations without significant performance degradation.

5.2 Safety Requirements

The Stock Market Analysis System must address safety concerns to prevent loss, damage, or harm:

- **Data Integrity Safeguards:** Implement checks to prevent data corruption during CRUD operations. Regular backups should be performed to safeguard against data loss.
- **Error Handling:** Gracefully handle API errors and database connectivity issues, providing clear error messages and fallback mechanisms to prevent system crashes.
- **External Policies and Regulations:** Comply with financial data handling regulations and standards to ensure data safety and integrity.
- **Safety Certifications:** If applicable, obtain certifications for data handling and processing to meet industry standards and regulatory requirements.

These safety requirements help mitigate risks associated with data loss or corruption and ensure compliance with relevant regulations.

5.3 Software Quality Attributes

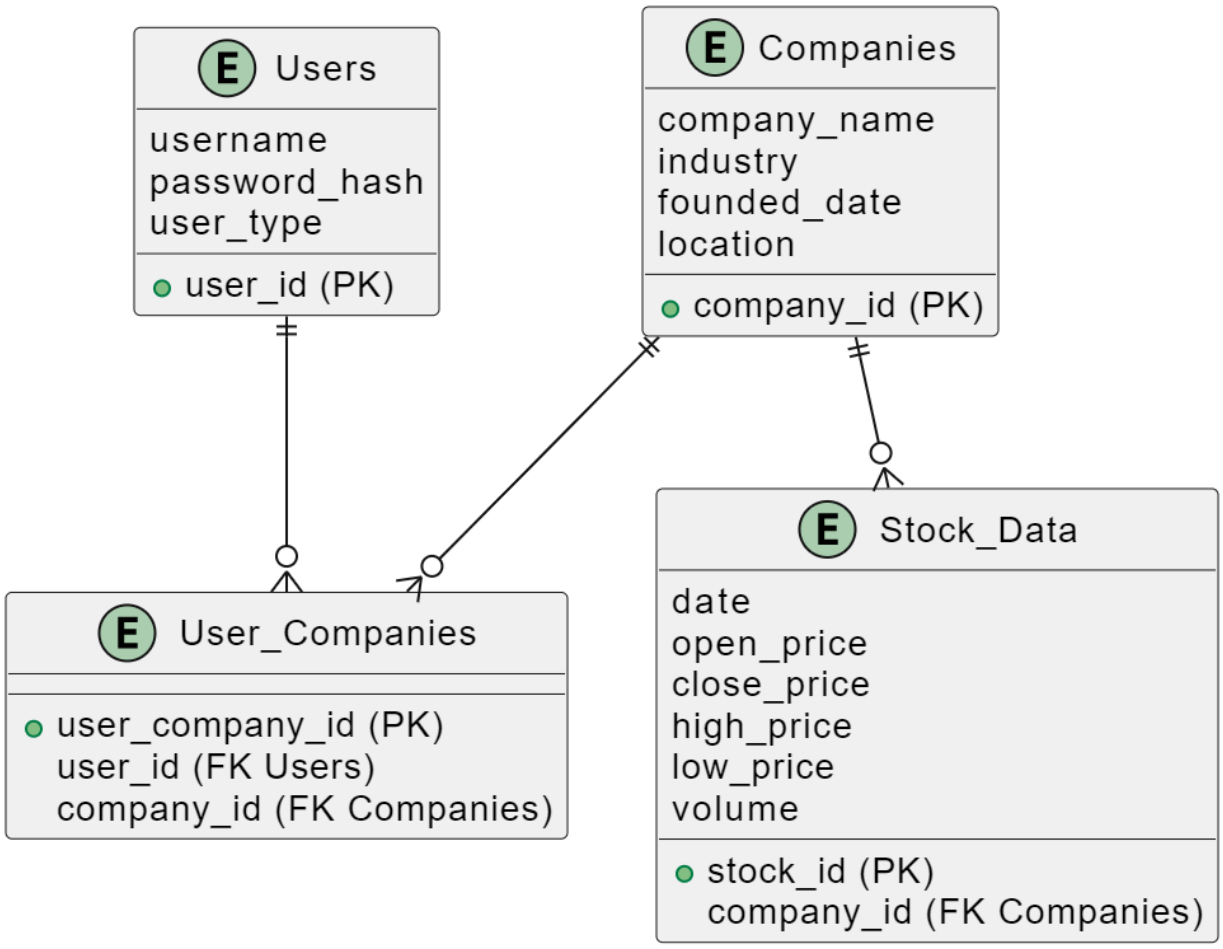
The Stock Market Analysis System must exhibit the following software quality attributes:

- **Adaptability**: The system should be easily adaptable to incorporate new features or changes in requirements without major rework.
- **Availability**: Ensure high availability with an uptime of at least 99.9%, minimizing downtime and ensuring continuous access for users.
- **Correctness**: The system must accurately perform all specified functions, ensuring the correctness of data retrieval, processing, and visualization.
- **Flexibility**: The system should be flexible to support various data sources and integrate with other financial analysis tools or APIs.
- **Interoperability**: Ensure compatibility with different operating systems, web browsers, and external APIs for seamless integration and user experience.
- **Maintainability**: Use modular and well-documented code to facilitate easy maintenance and updates. The system should support straightforward bug fixes and enhancements.
- **Portability**: Ensure the system can be easily deployed on different hardware and software environments without requiring significant changes.
- **Reliability**: The system should consistently perform its intended functions under specified conditions without failure.
- **Reusability**: Design components to be reusable across different modules or projects to enhance development efficiency.
- **Robustness**: The system should handle unexpected inputs or conditions gracefully, without crashing or producing incorrect results.
- **Testability**: Ensure the system is designed for easy testing, with comprehensive test coverage for all functions and features.
- **Usability**: Provide an intuitive user interface with clear navigation and user-friendly features to enhance the user experience.

These quality attributes ensure that the system is reliable, maintainable, and user-friendly, providing a high-quality experience for both users and developers.

Appendix A: Analysis Models

ER Diagram



The ER diagram we provided consists of four entities: Users, Companies, User_Companies, and Stock_Data. Here's a detailed description:

Users:

- Attributes: username, password_hash, user_id (PK).

Companies:

- Attributes: company_name, industry, founded_date, location, company_id (PK).

User_Companies:

- Attributes: user_company_id (PK), user_id (FK referencing Users), company_id (FK referencing Companies).

Stock_Data:

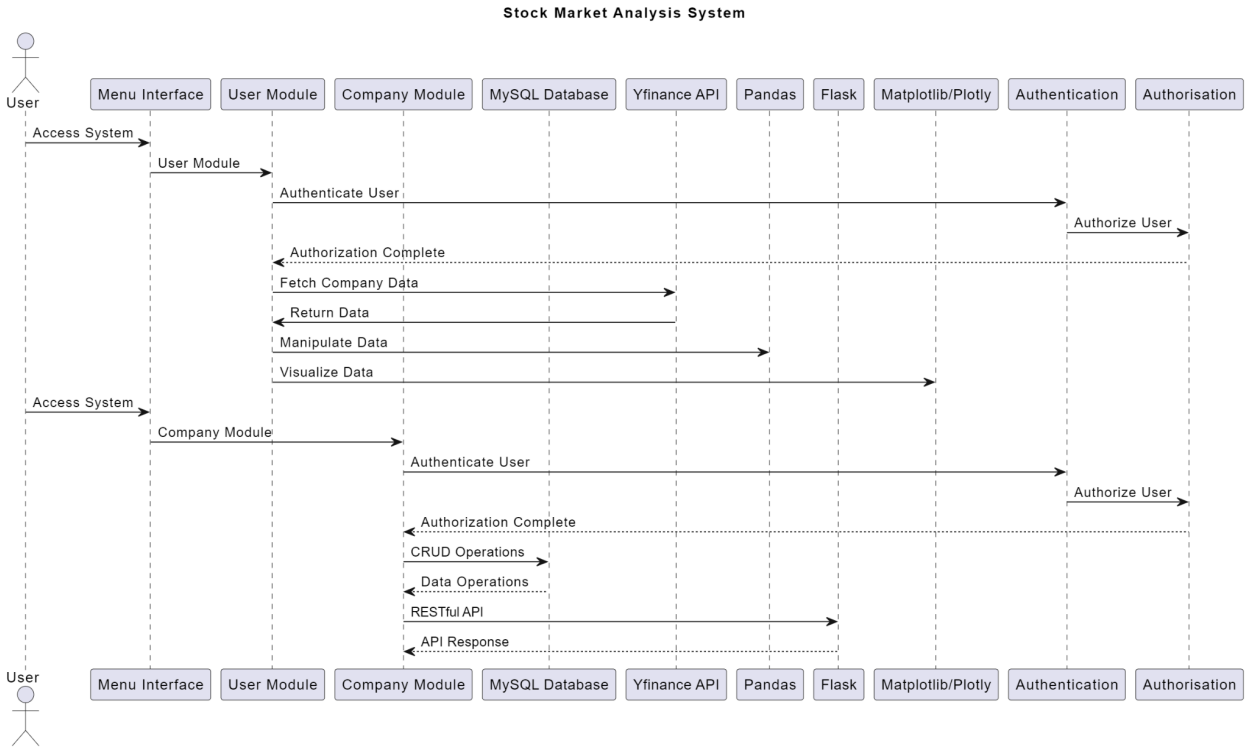
- Attributes: date, open_price, high_price, close_price, low_price, volume, stock_id (PK), company_id (FK referencing Companies).

Relationships:

- **Users to User_Companies:** One-to-many via user_id.
- **Companies to User_Companies:** One-to-many via company_id.
- **Companies to Stock_Data:** One-to-many via company_id.

This diagram represents a database structure for a stock trading or investment platform where users can be associated with multiple companies and track their stock data over time.

Sequence Diagram



In the Stock Market Analysis System, the sequence of actions involves two primary actors: the User and the Company Authority. Here's a concise breakdown of their interactions:

User Interaction:

1. Accessing the System: The User accesses the system through the Menu Interface.
2. User Module: After accessing, the User proceeds to interact with the User Module.
3. Authentication and Authorization: Authentication and authorization processes ensure secure access for the User.
4. Data Fetching: The User requests stock data using the Yfinance API.
5. Data Manipulation: Data fetched from Yfinance API is manipulated using Pandas for further analysis.
6. Data Visualization: Manipulated data is visualized using Matplotlib/Plotly, providing clear and interactive charts to the User.

Company Authority Interaction:

1. Accessing the System: The Company Authority accesses the system through similar initial steps.
2. Company Module: Interactions for the Company Authority are managed through the Company Module.
3. Authentication and Authorization: Similar to the User, authentication and authorization are crucial steps here too.
4. CRUD Operations: Company Authority performs CRUD operations (Create, Read, Update, Delete) on stock data stored in the MySQL database.
5. Data Management: CRUD operations ensure seamless integration and maintenance of stock data.
6. RESTful API Testing: Utilizing Flask, the system tests the functionality of GET and POST methods for the RESTful API.
7. API Response: The system generates API responses confirming the successful operation of GET and POST methods.