System Requirements Document

1. Introduction

This document outlines the key requirements for our stock analysis and sentiment evaluation project. The goal is to create an interactive platform that helps users analyze stock prices and assess the impact of news article on these prices. This document will guide the development process, ensuring that we stay on track to deliver a high-quality prototype and final product. The first semester focuses on building a prototype that demonstrates how news affects stock prices, while the second semester will expand this to include real-time data collection and predictive analysis of future stock price trends.

Stakeholders:

Project Team: Backend and frontend developers, document manager.
Users: Investors, financial analysts, data enthusiasts.
Mentors/Supervisors: Providing feedback and evaluating the project.

2. Functional Requirements (Lab 9)

These are the core functionalities our system needs to deliver:

1. **User Roles and Permissions**

- Admin: Admin users will have full control of system settings, manage API integrations, configure sentiment models, and oversee user accounts. They'll also monitor backend processes like data retrieval and performance.

- Standard User: Regular users will access stock data, view trends, and interact with different visual graphs. They'll be able to generate reports and filter data by date of the articles published, company, sentiment scores etc.

2. **Data Collection and Integration**

- API Integration: We will pull stock data from Yahoo finance and news articles from NewsAPI. This data will be securely stored in MongoDB. Integration will be smooth using the requests library, with error-handling mechanisms in place if APIs fail.

- How It Works: Stock data, such as daily closing prices for the past years, will be fetched through Yahoo finance. News articles tied to each company will come from NewsAPI, filtered by keywords. We'll store all this data in MongoDB and clean it to ensure it's ready for analysis.

- Managing Inaccuracies: To deal with inaccuracies, our system will flag articles with inconsistent sentiment scores, allowing us to refine and validate our results. We'll test and verify multiple NLP models to ensure better accuracy in sentiment detection.

3. **Sentiment Analysis**

- NLP Model Integration: We'll use sentiment analysis to process news articles. We're testing models like VADER and Text Blob or Hugging Face Transformers libraries for BERT or GPT model to pick the one that works best in our contexts.

- How It Works: The system will analyze the sentiment (positive, neutral, or negative) of news headlines and content, storing the results in MongoDB. If two models give conflicting results, we'll flag those articles for further inspection.

- Handling Edge Cases: Articles that give unexpected or outlier sentiment scores will be flagged for review. We'll refine or check out more advanced models to account for financial-specific terms and concepts.

4. **Dashboard and Data Display**

- User Dashboard: The dashboard will be the main interface for users to interact with the data. It will display stock price trends and sentiment analysis graphs, with filtering options to focus on specific dates, companies, or keywords.

- UI Design: The UI will have a left column listing company name. Clicking a company will display related news articles on the right, each showing the headline, a brief description, sentiment score, and stock price effect (high, medium, low). Users can click on an article to log in or sign up to view detailed charts and graphs. Filters above the articles allow users to sort by sentiment score or year for more focused results.

5. **Reporting and Analysis**

- Automated Reports: Users can generate reports comparing stock price trends and sentiment over time. These reports will include graphs and numerical values.

- Error Handling: If data is missing or an API call fails, users will be notified, and the system will provide options to retry or display cached data if available.

3. Non-functional Requirements (Lab 10)

These describe the qualities the system needs to have beyond just its functionality:

1. **Performance**

- Response Times: API calls should return data in under 3 seconds, and the dashboard should load stock trends in less than 2 seconds for typical data sets.
- Pre-processed data: Since we are dealing with the historic data, we plan to analyze and create the trends beforehand and push it on the webpage, giving smooth experience without any delays in calculations.

2. **Scalability**

- Growing with Demand: The system should be able to handle more users and data as it grows. MongoDB's flexible setup will allow us to add more companies and store more data without hitting performance issues.

3. **Security**

- Data Protection: All data transmissions will be encrypted. Sensitive data, like API keys, will be secured and not visible to users.

- User Authentication: Admins will use multi-factor authentication, while standard users will have strong password protection.

4. **Usability**

- User-Friendly Design: The platform will be easy to navigate, with intuitive tools for interacting with stock data and analysis graphs.

- Accessibility: The dashboard will be designed to meet accessibility standards, ensuring that all users can use the platform effectively.

5. **Reliability**

- System Uptime: We aim for 99% uptime, meaning the system will be available almost all the time. In case an API fails, the system will attempt to retrieve data again or show cached information.

- Data Backups: Daily backups of our data will ensure that nothing is lost in case of a system failure.

6. **Maintainability**

- Easy Updates: The system will be modular, so adding new features or fixing bugs won't be too difficult. We'll also maintain thorough documentation.

- Error Logging: We'll have comprehensive logs to track and resolve any system errors quickly.

4. Priority and Structure of Features

We've prioritized the most essential features and set a timeline for completing them:

High Priority:

- API integration for stock data and news
- Sentiment analysis
- Basic dashboard with trend visualization

Medium Priority:

- User roles and permissions
- Advanced filters for data analysis

Low Priority:

- Custom reports and export options
- Sentiment score adjustments for users

Feature Timeline:

End of September: MongoDB setup, API integration, and initial NLP model testing.
Mid-October: Dashboard with data visualization and sentiment analysis.
End of October: Full analysis of two companies.
Mid-November: Analysis of all remaining companies.
End of November: Final testing, documentation, and project delivery.

5. Conclusion

This document provides a clear outline of the system's requirements, from core functionality to the technical qualities that will make it reliable, scalable, and easy to use. We'll focus on integrating the APIs and ensuring accurate sentiment analysis while also making the platform intuitive and user-friendly. By following this roadmap, we're confident that we'll deliver a polished and valuable tool by the end of the project.