Stock Pulse Monitoring

Khanbu Arshiya Farheen[1], Mandala Shirisha[2], Kodmur Sailaja[3], Golla Indraja[4]
[1,2,3,4]*Dept. of Computer Science and Engineering, St. Johns College of Engineering and Technology, Yemmiganur 518301, India*

Abstract: **The rapid growth of financial markets and digital trading platforms has increased the demand for efficient real-time stock monitoring systems. Investors rely on live stock prices, technical indicators, and market-related news to make informed investment decisions. However, continuous manual tracking of stock prices across multiple platforms is time-consuming and inefficient. This paper presents STOCK PULSE, a real-time stock monitoring and intelligent alert system developed using the MERN stack with Typescripts. The system provides live stock price tracking, automated alerts based on price thresholds and technical indicators, real-time updates using Web-sockets, and news-based sentiment analysis. By integrating real-time data streaming and automation, STOCK PULSE offers a centralized, scalable, and user-friendly platform for effective stock market monitoring.**

Index Terms: **Stock Monitoring, Real-Time Systems, Intelligent Alerts, MERN Stack, WebSockets, Technical Indicators, Sentiment Analysis, Financial Dashboard**

I.INTRODUCTION

The rapid growth of financial markets and digital trading platforms has increased the need for efficient, real-time stock monitoring systems. Investors today rely not only on historical data but also on real-time price movements, technical indicators, and market-related news to make informed decisions. However, continuously tracking stock prices across multiple platforms is time-consuming and inefficient, especially for beginners and individual investors.

STOCK PULSE is a real-time stock price monitoring and intelligent alert system designed to address these challenges. The application provides live stock price updates, automated alerts based on price thresholds and technical indicators, and news-based sentiment analysis to support timely and informed investment decisions. By integrating real-time data streaming, technical analysis, and financial news, the system offer a centralized and user-friendly platform for stock tracking.

The project is developed using the MERN stack (MongoDB, Express.js, React.js, Node.js) with Typescript, ensuring scalability, maintainability, and industry-standard coding practices. STOCK PULSE demonstrates how modern web technologies can be applied to build a robust financial monitoring system suitable for real-world applications.

II. SYSTEM ARCHITECTURE

The system architecture of STOCK PULSE follows a modular and scalable client-server model. It is designed to efficiently handle real-time data, user interactions, and background processing.

Architecture Overview

1. The front-end is built using React with Typescripts to provide a responsive and interactive user interface.
2. The back-end is developed using Node.js and Express.js, responsible for handling business logic, API communication, authentication, and alert processing.
3. MongoDB is used as the database to store user details, watch-lists, alert configurations, and notification logs. Real-time stock price updates are delivered using WebSockets (Socket.IO).
4. External API s are integrated for stock market data and financial news.
5. Cron jobs handle background tasks such as periodic data fetching and alert evaluation

Component Interaction

1. The user interacts with the front-end dashboard.
2. Requests are sent to the back-end via REST API s.
3. The back-end fetches stock and news data from

third-party API s.

4. Real-time updates are pushed to the front-end using Web-sockets. Alerts are evaluated and notifications are sent via email. This architecture ensures low latency, real-time responsiveness, and scalability

III. METHODOLOGY

The methodology adopted for the development of STOCK PULSE follows a systematic and modular approach, ensuring clarity, scalability, and ease of maintenance. The system is divided into multiple functional modules, each responsible for a specific task.The development of STOCK PULSE follows a structured and iterative methodology. Initially, system requirements were analyzed to identify core features such as real-time tracking, alerts, and dashboards. The system architecture and database schema were then designed to support scalability and performance. Frontend components were developed to ensure smooth user interaction, while backend APIs were implemented to manage data flow and alert logic. Real-time data streaming was integrated using WebSockets, followed by testing to ensure accuracy, reliability, and responsiveness. Finally, the system was prepared for deployment with future enhancement possibilities.

A. Requirement Analysis

Identifying the needs of individual investors and beginners

Understanding limitations of existing stock tracking systems

Defining core features such as real-time updates, alerts, and sentiment analysis

B. *Data Collection*

Stock price data is fetched from third-party stock market API's. Financial news data is retrieved using News API's. The data includes prices, timestamps, headlines, descriptions, and sources.

C. Data Processing

Stock price data is processed to calculate technical indicators such as SMA, EMA, and RSI News data is analyzed using sentiment analysis techniques to classify sentiment as positive, negative, or neutral

D. Alert Evaluation

User-defined alert conditions are stored in the database. Background jobs continuously evaluate price and indicator conditions. Alerts are triggered when conditions are satisfied.

E. Real-Time Communication

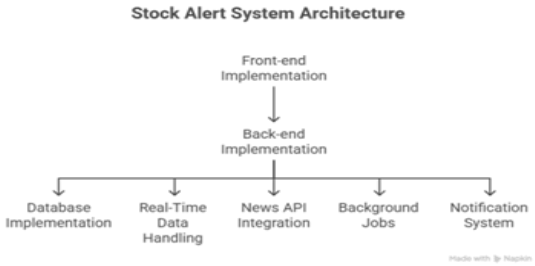
Web Sockets are used to push live stock price updates to the front end. This eliminates the need for constant manual refreshing.

F. Notification System

Email notifications are sent using Node-Mailer & Users receive instant alerts for significant market movements.

G. Security and Authentication

JWT-based authentication is implemented. Secure access to user-specific data and dashboards is ensured This structured methodology ensures efficient system operation and reliable performance.



IV. IMPLEMENTATION

The implementation of STOCK PULSE is carried out by dividing the system into front-end, back-end, database, and integration layers.

1. *Front-end Implementation*

Developed using React and Typescript .Interactive dashboards display stock prices, charts, and sentiment indicators. Chart libraries are used for visualizing historical data, Real-time updates are handled using Socket.IO client.

2. Back-end Implementation

Node.js and Express.js handle API requests Restful API s manage user authentication, watch lists, and

alerts Business logic processes stock data and evaluates alert conditions.

3. Database Implementation

MongoDB stores user profiles, watch-lists, alert rules, and notification logs. Schema design ensures flexibility and scalability.

4. Real-Time Data Handling

Socket.IO enables bi-directional communication Livestock updates are pushed instantly to connected users.

5. News API Integration

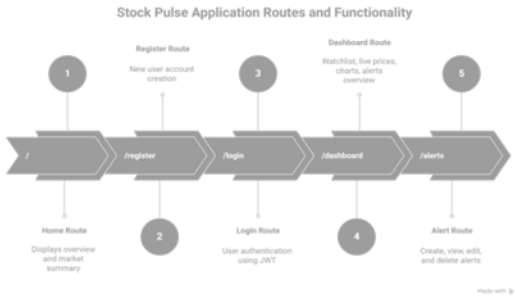
Financial news is fetched using News API s Relevant articles are filtered based on stock keywords. Sentiment analysis is applied to headlines and descriptions.

6. Background Jobs

Cron jobs periodically fetch stock data. Alerts are evaluated at regular intervals. Ensures timely notification delivery

7. Notification System

Node-Mailer is used to send email alerts. Notifications include stock name, trigger condition, and current price.



V. RESULTS AND DISCUSSION

The implementation of STOCK PULSE successfully achieved its objectives. The system provides real-time stock price updates with minimal latency and accurately triggers alerts based on user-defined conditions. The integration of technical indicators enhances analytical capabilities, while news sentiment analysis provides additional market context.

Users can efficiently monitor multiple stocks through a single dashboard, reducing manual effort. The realtime update mechanism ensures that users receive timely information, minimizing the risk of missed investment opportunities. Overall, the system demonstrates reliability, responsiveness, and scalability suitable for real-world usage

VI. ADVANTAGES AND APPLICATIONS

Advantages

Real-time monitoring reduces manual effort. Automated alerts improve decision-making. Beginner-friendly user interface Scalable and modular architecture. Industry-relevant technology stack TOCK PULSE offers several advantages including time savings, reduced manual effort, improved investment decision accuracy, real-time notifications, and a beginner-friendly interface. The system follows an industry-level architecture that ensures scalability and security. It can be applied in stock price monitoring platforms, trading and investment analysis systems, educational financial tools, and real-time market alert applications used by investors and analysts.

Applications

Individual investors and traders. Educational institutions for learning stock analysis Financial analytical platforms. Can be extended to crypto and forex markets

VII. CONCLUSION

The STOCK PULSE is a comprehensive real-time stock monitoring and alert system that integrates live market data, technical analysis, and news sentiment into a single platform. By leveraging the MERN stack with Type Script, the system ensures scalability, performance, and maintainability. The project successfully demonstrates the practical application of modern web technologies in the financial domain and provides a strong foundation for future enhancements such as AI-based predictions and portfolio management.

ACKNOWLEDGMENT

I would like to express my sincere gratitude to my project guide and faculty members for their continuous

support and guidance throughout the development of this project. I am also thankful to my institution for providing the necessary resources and environment to carry out this work.

Finally, I extend my heartfelt thanks to my friends and family for their encouragement and support during the completion of this project for their support and technical assistance during the project development.

I would like to thank my family and friends for continuous motivation, patience, and support without which it was not possible to remain focused and successfully complete this venture.

REFERENCES

[1] Dal Pozzolo, A., Bontempi, G., Snoeck, M., & Snoeck, C., "Adversarial drift detection for fraud detection," IEEE International Conference on Data Mining, 2015.
[2] Bahnsen, A. C., Aouada, D., & Ottersten, B., "Costsensitive decision trees for fraud detection," Expert Systems with Applications, vol. 39, no. 5, 2012.
[3] Carcillo, F., Dal Pozzolo, A., Snoeck, M., Bontempi, G., & Snoeck, C., "Scarff: A scalable framework for streaming credit card fraud detection," Information Fusion, 2021.
[4] Phua, C., Lee, V., Smith, K., & Gayler, R., "A comprehensive survey of data mining-based fraud detection research," arXiv preprint arXiv:1009.6119, 2010.
[5] Dal Pozzolo, A., Boracchi, G., Bontempi, G., & Snoeck, C., "Credit card fraud detection: A realistic modeling and new public dataset," IEEE Intelligent Systems, 2015.
[6] Breiman, L., "Random forests," Machine Learning, vol. 45, no. 1, 2001.
[7] Chandola, V., Banerjee, A., & Kumar, V., "Anomaly detection: A survey," ACM Computing Surveys, vol. 41, no. 3, 2009.
[8] Kaggle, "Credit Card Fraud Detection Dataset," Available: Kaggle Repository, 2018.