iJRASET
International Journal For Research in Applied Science and Engineering Technology

# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

**Volume:** 14 **Issue:** IV **Month of publication:** April 2026

**DOI:** https://doi.org/10.22214/ijraset.2026.81200

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
*ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538*
*Volume 14 Issue IV Apr 2026- Available at www.ijraset.com*

# Zenstock: An Interactive Stock Trading Platform Using MERN Stack

Ankita Verma[1], Ms. Neha Singh[2], Ashi Mishra[3]
[1, 3]*B.Tech Student* , [2]*Assistant Professor, Department of Information Technology, Goel Institute of Technology and Management, Lucknow, India*

***Abstract:*** ***With the rapid advancement of digital financial technologies, stock trading has become increasingly accessible to a wide range of users. However, many existing trading platforms remain complex, feature-heavy, and difficult for beginners to understand and use effectively. This creates a significant gap between theoretical knowledge and practical trading experience.***
***This paper presents Zenstock, a full-stack web-based stock trading and order management system developed using the MERN stack (MongoDB, Express.js, React.js, and Node.js). The proposed system is designed to simulate real-world stock trading operations in a simplified and interactive environment. It allows users to perform essential trading activities such as buying and selling stocks, tracking order history, and managing their investment portfolio.***
***The system architecture ensures seamless communication between the frontend, backend, and database using RESTful APIs, enabling efficient data processing and real-time updates. Secure authentication mechanisms using JSON Web Tokens (JWT) and password encryption techniques ensure data privacy and user security. The modular design enhances scalability and maintainability, making the system adaptable for future improvements.***
***Keywords: MERN Stack, Stock Trading, Order Management, Full Stack Development, MongoDB, React, Node.js.***

## I. INTRODUCTION

In recent years, the rapid advancement of digital technologies has significantly transformed the financial sector, particularly in the domain of stock trading. Online trading platforms have made it possible for individuals to participate in financial markets from anywhere, eliminating traditional barriers such as physical brokerage services and geographical limitations. Platforms like Zerodha have played a major role in democratizing stock trading by providing cost-effective and accessible solutions to users.

Despite these advancements, many existing trading platforms remain complex and difficult for beginners to navigate. The presence of advanced tools, technical charts, and dense user interfaces often creates confusion for new users, limiting their ability to understand basic trading operations. As a result, there is a growing need for simplified and educational trading systems that can bridge the gap between theoretical knowledge and practical implementation.

To address these challenges, this paper presents Zenstock, a full-stack stock trading and order management system designed to provide a simplified and interactive trading experience. The platform allows users to simulate real-world trading activities such as buying and selling stocks, tracking order history, and managing portfolios in a controlled environment. This simulation-based approach helps users gain practical exposure without the risk associated with real financial transactions.

The primary objectives of this project are to design a user-friendly trading platform, enable simulation of stock trading operations, and demonstrate the integration of frontend, backend, and database systems in a full-stack application. The system is developed using the MERN stack, which includes MongoDB for database management, Express.js and Node.js for backend development, and React.js for building an interactive user interface.

Furthermore, the system follows a modular and scalable architecture, ensuring efficient data handling, secure user authentication, and smooth communication between system components through RESTful APIs. This not only enhances performance but also allows future expansion of the platform, such as integration with real-time stock APIs and advanced analytics.

In conclusion, Zenstock serves as both an educational tool for beginners and a foundational prototype for real-world stock trading platforms, contributing to the development of accessible and user-centric financial technologies.

## II. PROBLEM STATEMENT

In recent years, stock trading platforms have become widely accessible due to digital transformation. However, most traditional and modern trading systems are primarily designed for experienced users and investors. These platforms often include complex dashboards, advanced analytical tools, and technical charts that can be overwhelming for beginners.

One of the major challenges faced by new users is the lack of a simple and intuitive interface that allows them to understand basic trading operations. The steep learning curve discourages users from actively participating in stock markets. Additionally, existing platforms generally involve real financial transactions, which increases the risk for beginners who are still learning and may lack proper knowledge of market dynamics.

Another significant limitation is the absence of simulation-based learning environments. Most platforms do not provide a safe space where users can practice trading without financial risk. As a result, users are unable to gain practical experience, which is essential for building confidence and understanding trading strategies.

Furthermore, many systems lack integrated features for simplified order management and portfolio tracking in a beginner-friendly format. This makes it difficult for users to monitor their investments and analyze their trading performance effectively.

## III. OBJECTIVE

The primary objective of this project is to design and develop Zenstock, an interactive and user-friendly stock trading and order management system that simplifies the trading experience for beginners.

The specific objectives include:

1) To develop a simple and intuitive user interface that enhances usability and reduces complexity.
2) To simulate real-time stock trading operations, including buying and selling of stocks.
3) To implement an efficient order management system for tracking and managing trades.
4) To provide a portfolio management feature that allows users to monitor their investments and performance.
5) To demonstrate full-stack development using the MERN stack with seamless integration between frontend, backend, and database.
6) To create a risk-free learning environment where users can gain practical trading experience.

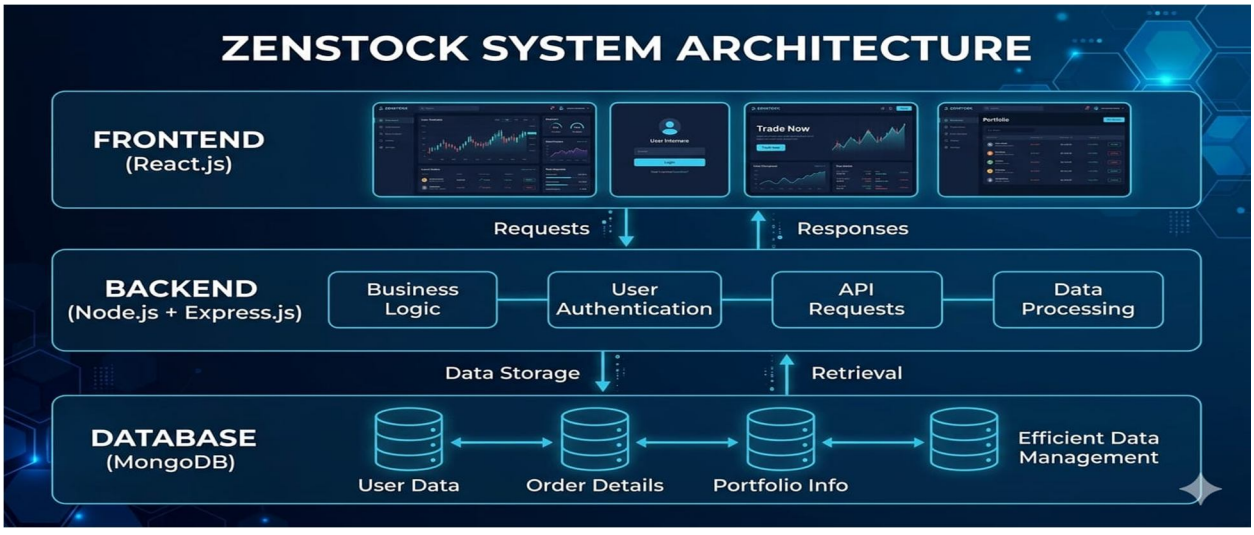
## IV. SYSTEM ARCHITECTURE

### A. *Architecture Overview*

The Zenstock system follows a three-tier architecture consisting of the frontend, backend, and database layers. This architecture ensures separation of concerns, scalability, and efficient data management.

1) Frontend (React.js): The frontend layer is responsible for user interaction and interface design. It provides a dynamic and responsive user experience where users can perform actions such as login, stock trading, viewing portfolio, and tracking orders.
2) Backend (Node.js + Express.js): The backend acts as the core processing unit of the system. It handles business logic, user authentication, API requests, and communication between the frontend and the database.
3) Database (MongoDB): MongoDB is used to store user data, order details, and portfolio information. Its flexible schema design allows efficient handling of dynamic data.

### B. *System Architecture Diagram*



*C. Working Flow*

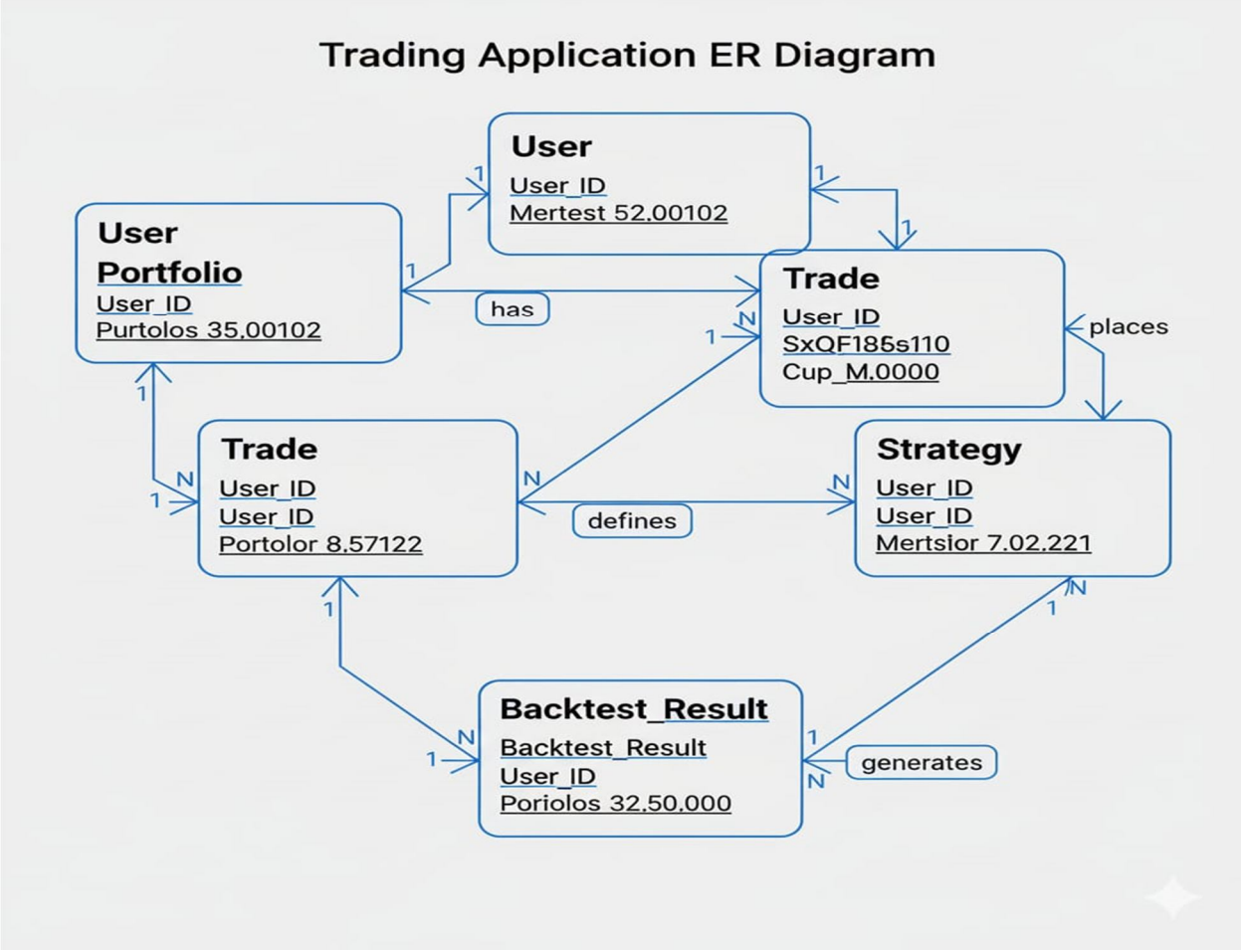
The system operates through a sequence of steps that ensure smooth interaction between components:

1) User Interaction: The user interacts with the React-based frontend to perform actions such as login, buying or selling stocks, and viewing portfolio details.
2) API Request: The frontend sends HTTP requests (GET, POST, PUT, DELETE) to the backend server through RESTful APIs.
3) Backend Processing: The Node.js and Express.js backend processes these requests by executing the required business logic, such as validating user input, handling authentication, and processing trading operations.
4) Database Communication: The backend interacts with MongoDB to store or retrieve relevant data, including user information, orders, and portfolio records.
5) Response Generation: After processing, the backend sends a response back to the frontend, which updates the user interface accordingly in real time.

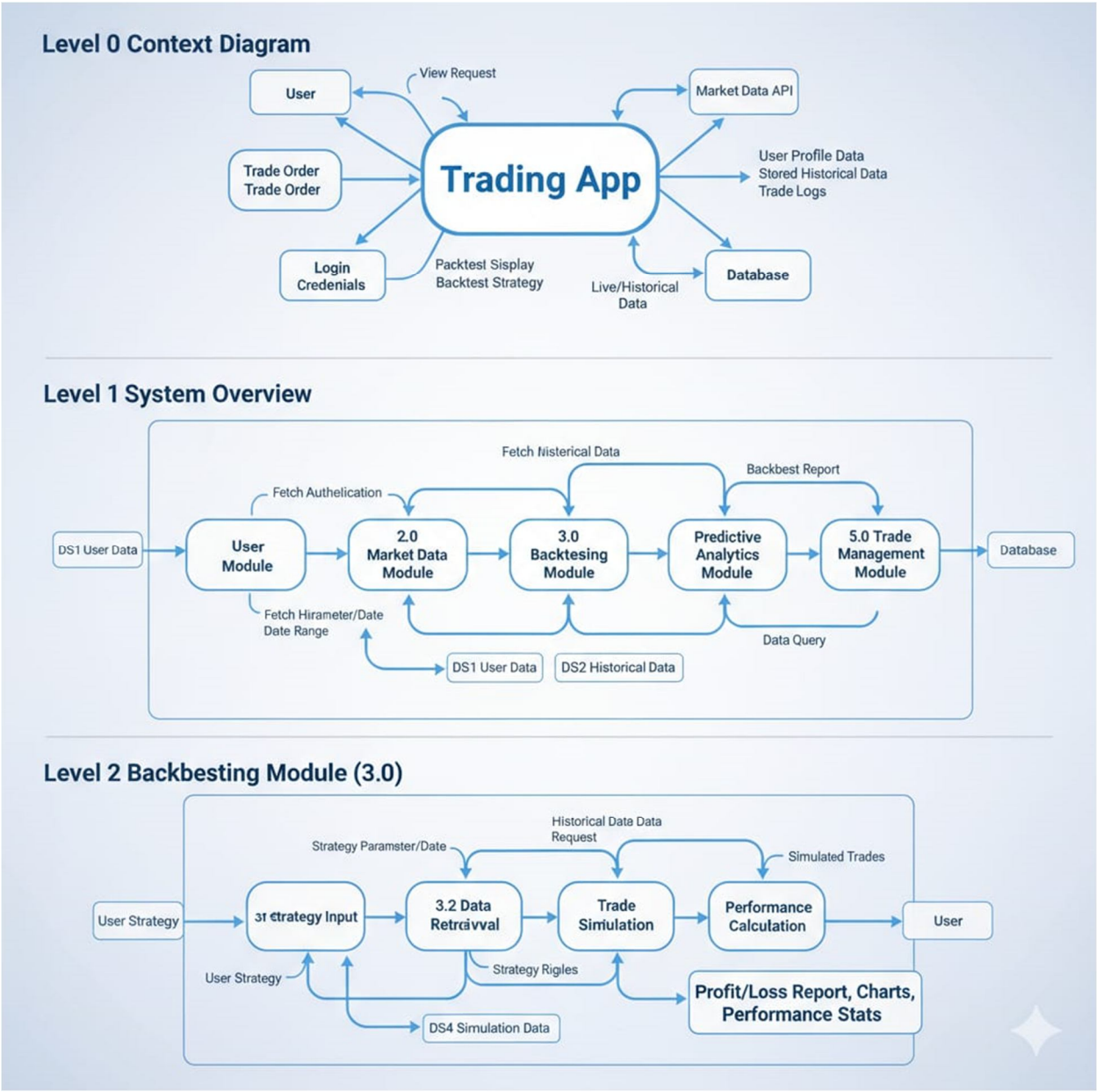
*D. Key Features of Architecture*

1) Scalability: Easily supports additional features and modules.
2) Modularity: Each component works independently.
3) Efficiency: Fast data processing using REST APIs.
4) Security: Secure communication and authentication mechanisms.
5) Flexibility: MongoDB allows dynamic data handling.

*E. ER Diagram*



*F. DFD Diagram*




## V. TECHNOLOGY STACK

The development of Zenstock utilizes the MERN stack along with supporting technologies to ensure efficient performance, scalability, and secure data handling. Each component of the technology stack plays a crucial role in the overall functioning of the system.

*A. Frontend*

The frontend is developed using modern web technologies to provide an interactive and responsive user interface.

- React.js: React.js is used to build a dynamic and component-based user interface. It enables efficient rendering through a virtual DOM and enhances user experience with fast and responsive design.
- Axios: Axios is a promise-based HTTP client used to send API requests from the frontend to the backend. It facilitates smooth communication and data exchange between the client and server.
- HTML & CSS: HTML is used for structuring web pages, while CSS is used for styling and layout design. Together, they ensure a clean and user-friendly interface.

### B. Backend

The backend is responsible for handling application logic, processing requests, and managing communication between the frontend and database.

- Node.js: Node.js provides a runtime environment for executing JavaScript on the server side. It enables asynchronous and event-driven programming, ensuring high performance and scalability.
- Express.js: Express.js is a lightweight web framework used to build RESTful APIs. It simplifies routing, request handling, and middleware integration, making backend development efficient.

### C. Database

- MongoDB: MongoDB is a NoSQL database used to store application data such as user details, orders, and portfolio information. Its flexible schema and document-based structure make it suitable for handling dynamic and scalable data.

### D. Authentication and Security

Security is a critical aspect of the system, and the following technologies are used to ensure safe user access:

- JWT (JSON Web Token): JWT is used for secure user authentication. It generates tokens that verify user identity and allow access to protected routes without repeatedly sending credentials.
- bcrypt: bcrypt is used for hashing user passwords before storing them in the database. This ensures that sensitive information remains secure even in case of data breaches.

### E. Advantages of Technology Stack

- Enables full-stack JavaScript development.
- Provides high scalability and performance.
- Ensures secure authentication and data protection.
- Supports real-time and efficient data communication.
- Facilitates easy maintenance and future enhancements.

## VI. KEY MODULES

The Zenstock system is divided into multiple functional modules, each responsible for handling specific operations within the platform. This modular approach ensures better organization, scalability, and maintainability of the system.

### A. Authentication Module

The Authentication Module is responsible for managing user access and ensuring system security.

- Allows users to register (signup) and login to the platform.
- Uses JWT (JSON Web Token) for secure session management.
- Implements bcrypt hashing to securely store user passwords.
- Protects sensitive routes and ensures only authorized users can access system features.
- This module ensures data privacy and prevents unauthorized access.

### B. Trading Module

The Trading Module handles all stock trading operations within the system.

- Enables users to buy and sell stocks in a simulated environment.
- Processes user requests and validates trading inputs.
- Updates order and portfolio data based on transactions.
- Mimics real-world trading behavior without financial risk.
- This module provides practical exposure to trading activities.

### C. Order Management Module

The Order Management Module is responsible for tracking and managing all trading transactions.

- Stores details of each transaction such as stock name, quantity, price, and type (buy/sell).
- Allows users to view order history.

- Helps in analyzing past trading activities.
- Ensures proper record maintenance of all operations.
- This module improves transparency and tracking of user actions.

*D. Portfolio Module*

The Portfolio Module manages the user's stock holdings and investment data.

- Displays current stock holdings of the user.
- Calculates total portfolio value based on transactions.
- Updates portfolio dynamically after each buy/sell operation.
- Helps users track their investment performance.
- This module provides insights into user investments and growth.

*E. Dashboard Module*

The Dashboard Module acts as the main interface for users to interact with the system.

- Displays user information and account details.
- Shows recent trading activity and portfolio summary.
- Provides quick access to trading, orders, and portfolio sections.
- Enhances user experience through a clean and organized interface.
- This module serves as the central hub of the application..

## VII. WORKING PRINCIPLE

The working of the Zenstock system follows a structured sequence of operations that ensures smooth interaction between the user interface, backend processing, and database management. The system is designed to simulate real-world stock trading in a simplified and efficient manner.

*A. Step-by-Step Process*

- User Authentication: The process begins when the user logs into the system using valid credentials. The Authentication Module verifies the user using secure methods such as JWT tokens and encrypted passwords.
- User Interaction: After successful login, the user interacts with the dashboard to select a stock and choose an action such as buying or selling.
- Request Transmission: The frontend sends the user's request to the backend server through RESTful API calls using HTTP methods such as POST or GET.
- Backend Processing: The Node.js and Express.js backend receives the request and performs necessary validations, such as checking user authenticity, verifying input values, and processing trading logic.
- Business Logic Execution: Based on the request type (buy/sell), the system calculates updated values, modifies portfolio data, and prepares order details.
- Database Update: The backend interacts with MongoDB to store or update data in the relevant collections, including Users, Orders, and Portfolio.
- Response Generation: After processing, the backend sends a response back to the frontend containing updated data and confirmation of the transaction.
- User Interface Update: The frontend dynamically updates the dashboard, displaying the latest portfolio value, order history, and transaction status to the user.

*B. Key Characteristics:*

- Real-time Simulation: Provides instant updates after each transaction.
- Data Consistency: Ensures accurate synchronization between modules.
- Secure Processing: Maintains data security throughout the workflow.
- Efficient Communication: Uses REST APIs for seamless interaction.

*ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538*
*Volume 14 Issue IV Apr 2026- Available at www.ijraset.com*

## VIII. RESULTS AND DISCUSSION

### A. *Results*

The implementation of Zenstock demonstrates successful integration of frontend, backend, and database components, resulting in a fully functional stock trading simulation platform. The system performs efficiently under normal usage conditions and provides a seamless user experience.

- Smooth Frontend-Backend Integration: The use of React.js for the frontend and Node.js with Express.js for the backend ensures efficient communication through RESTful APIs. User requests are processed quickly, and responses are reflected dynamically on the interface.
- Efficient Data Handling: MongoDB enables fast storage and retrieval of data related to users, orders, and portfolios. The system maintains consistency and accuracy in updating records after each transaction.
- Secure Authentication: The implementation of JWT-based authentication along with bcrypt password hashing ensures that user data is protected. Unauthorized access is prevented through secure validation mechanisms.
- Responsive User Interface: The platform provides real-time updates, allowing users to instantly view changes in their portfolio and transaction history after performing any action.

### B. *Discussion*

The developed system successfully addresses the challenges identified in traditional trading platforms by simplifying user interaction and providing a simulation-based learning environment. Unlike conventional systems, Zenstock focuses on usability and accessibility, making it suitable for beginners.

The modular architecture of the system enhances flexibility, allowing new features to be integrated without affecting existing functionality. Additionally, the use of the MERN stack ensures that the application remains scalable and maintainable for future improvements.

### C. *Advantages*

- Beginner-Friendly Interface: The system is designed with a simple and intuitive interface, making it easy for new users to understand trading concepts.
- Real-Time Simulation: Users can perform trading operations and immediately see the results, which enhances learning and engagement.
- Scalable Architecture: The modular design allows the system to be extended with additional features such as real-time APIs and analytics.
- Full-Stack Implementation: Demonstrates complete integration of frontend, backend, and database technologies, making it a strong practical application of full-stack development.

## IX. FUTURE SCOPE

The current implementation of Zenstock provides a strong foundation for a stock trading simulation platform. However, there are several opportunities for future enhancements that can significantly improve the system's functionality, performance, and real-world applicability**.**

### A. *Integration with Real-Time Stock APIs*

The system can be enhanced by integrating real-time stock market APIs such as Alpha Vantage or Yahoo Finance. This would allow users to access live stock prices and market data, making the platform more realistic and closer to real-world trading environments.

### B. *Advanced Analytics and Visualization*

Future versions of Zenstock can include advanced analytical tools such as:

- Interactive charts and graphs.
- Technical indicators (moving averages, RSI, etc.).
- Performance analysis dashboards.

These features will help users better understand market trends and make informed trading decisions.

### *C. AI-Based Stock Prediction*

Artificial Intelligence and Machine Learning models can be integrated to predict stock price movements based on historical data and market trends. This would provide users with intelligent insights and enhance decision-making capabilities.

### *D. Mobile Application Development*

Developing a mobile application version of Zenstock for Android and iOS platforms would improve accessibility and user engagement. A mobile app would allow users to trade and monitor their portfolio anytime and anywhere.

### *E. Enhanced Security and Features*

Future improvements can also focus on:

- Multi-factor authentication (MFA).
- Role-based access control.
- Notification systems for trade alerts.
- Cloud deployment for better scalability.

## X. CONCLUSION

This paper presented Zenstock, a full-stack stock trading and order management system developed using the MERN stack. The project successfully demonstrates the integration of frontend, backend, and database technologies to create a seamless and interactive trading platform.

The system provides a simplified environment where users can simulate stock trading activities such as buying, selling, and portfolio management. By focusing on usability and accessibility, Zenstock effectively addresses the challenges faced by beginners in understanding complex trading platforms.

Furthermore, the application ensures efficient data handling, secure authentication, and real-time updates, making it both reliable and scalable. The modular architecture allows easy extension of features, enabling future enhancements such as real-time data integration, advanced analytics, and AI-based prediction models.

In conclusion, Zenstock not only serves as an effective educational tool for learning stock market operations but also acts as a strong foundation for developing real-world trading applications, contributing to the advancement of user-friendly financial technologies.

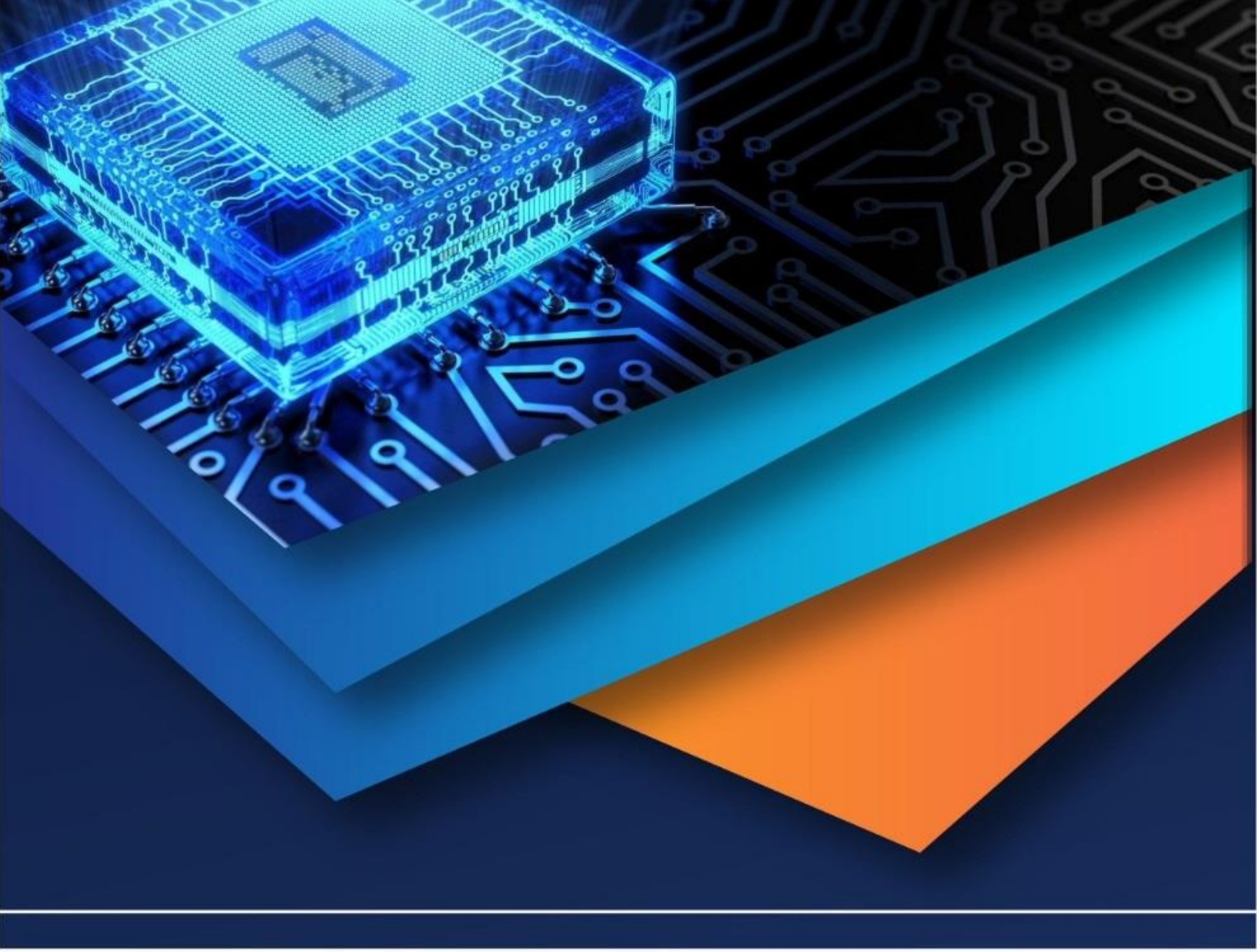
## XI. ACKNOWLEDGEMENTS

The authors express sincere gratitude to Ms. Neha Singh , Assistant Professor, Department of Information Technology, GITM Lucknow, for expert guidance, laboratory access, and continuous encouragement throughout Session 2025–2026. The authors also thank the GITM IT Department staff for technical support during prototype assembly and testing.

## REFERENCES

[1] MongoDB Inc., “MongoDB Documentation,” [Online]. Available: https://www.mongodb.com/docs/
[2] Meta Open Source, “React.js Documentation,” [Online]. Available: https://reactjs.org/docs/
[3] OpenJS Foundation, “Node.js Documentation,” [Online]. Available: https://nodejs.org/en/docs/
[4] Express.js, “Express Web Framework Documentation,” [Online]. Available: https://expressjs.com/
[5] Auth0, “JSON Web Token (JWT) Introduction,” [Online]. Available: https://jwt.io/introduction/
[6] Provos, N. and Mazières, D., “A Future-Adaptable Password Scheme,” USENIX, 1999.
[7] Alpha Vantage Inc., “Stock Market API Documentation,” [Online]. Available: https://www.alphavantage.co/documentation/
[8] Yahoo Finance, “Financial Data Platform,” [Online]. Available: https://finance.yahoo.com/
[9] M. McKinney, “Python for Data Analysis,” O’Reilly Media, 2017.
[10] T. M. Mitchell, “Machine Learning,” McGraw-Hill, 1997.
[11] I. Sommerville, “Software Engineering,” 10th ed., Pearson, 2015.
[12] R. S. Pressman, “Software Engineering: A Practitioner’s Approach,” McGraw-Hill, 2014.
[13] D. Flanagan, “JavaScript: The Definitive Guide,” O’Reilly Media, 2020.
[14] [14] K. C. Laudon and J. P. Laudon, “Management Information Systems,” Pearson, 2018.
[15] Investopedia, “Stock Trading Basics,” [Online]. Available: https://www.investopedia.com/
[16] W3Schools, “Web Development Tutorials,” [Online]. Available: https://www.w3schools.com/



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY