Zosh Trading: A Very Secure and User-Friendly Crypto Trading Platform

Akhilesh Kumar; Prince Singh; Akhilesh Kumar; Abhishek Kumar Shrivastav

CSE Department NIET, Greater Noida 2201330100028

Supervisor: Mr. Ibrar Ahmed, Assistant Professor, CSE Department, NIET Greater Noida

Abstract— The evolution of financial technology has a redefined conventional investment frameworks, yet many modern digital trading tools remain overly complex for the average user. While their are most current systems cater primarily to professional audiences—often featuring steep learning curves, high costs, and minimal educational resources—this research introduces Zosh Trading. Developed using a MERN stack and a microservices-based approach, Zosh Trading offers a secure, intuitive environment for cryptocurrency and stock exchange. By integrating a streamlined user interface, live data visualization, automated portfolio management, and JWT-secured authentication within a scalable three-tier architecture, the platform successfully balances analytical depth with ease of use for entry-level and intermediate investors.

Keywords—MERN Stack, Cryptocurrency Trading, Microservices, REST API, JWT Authentication, Portfolio Management, FinTech, React.js, Node.js, MongoDB

I. Introduction

The proliferation of internet-based financial services has democratized access to global markets, enabling millions of users to participate in cryptocurrency and stock trading. However, this expansion has exposed a critical gap: while sophisticated platforms exist for experienced traders, the majority of beginner and intermediate investors struggle with complex interfaces, opaque fee structures, and insufficient analytical guidance [1].

Zosh Trading is a web-based trading platform designed to address these challenges. Built on the MERN stack— MongoDB, Express.js, React.js, and Node.js—the system provides real-time market data visualization, portfolio management, and transaction tracking within a secure and intuitive environment. The platform adopts a microservices-inspired modular architecture to ensure scalability, maintainability, and performance.

The primary objectives of the system are: (1) to develop a secure and scalable trading web application; (2) to provide real-time stock and crypto data visualization through APIs;
(3) to enable effective portfolio management with automated profit/loss computation; and (4) to guarantee secure session management using JSON Web Tokens (JWT) [5].

This paper is organized as follows: Section II surveys existing trading platforms and their limitations. Section III defines the problem statement. Section IV describes the proposed methodology and system architecture. Section V presents feasibility analysis. Section VI discusses results and implementation details. Section VII concludes with future directions.

II. Related Work and Existing Systems

Several digital trading platforms currently dominate the market. Zerodha [2] is a leading Indian brokerage platform offering stock, commodity, and derivatives trading. Although reliable, its interface presents a steep learning curve for new users. Groww targets beginner investors with a simplified UI but lacks advanced analytical tools needed by intermediate traders. Upstox provides real-time charting and analytical capabilities but imposes brokerage charges that deter small investors. Binance [3] is a globally recognized cryptocurrency exchange offering extensive trading pairs; however, the high volatility of its assets and complex order types remain challenges for entry-level users.

DOI: https://doi.org/10.5281/zenodo.20079493

A comparative analysis (Table I) summarizes the features and limitations of these platforms:

Table I. Comparative Analysis of Existing Trading Platforms

Platform	Key Features	Advantages	Limitations
Zerodha	Stock Trading	Reliable & fast	Complex interface
Groww	Simplified UI	Beginner- friendly	Limited analytics
Upstox	Real-time charts	Analytical tools	Brokerage charges
Binance	Crypto trading	Global access	High volatility risk

The analysis reveals a consistent pattern: platforms prioritizing simplicity sacrifice analytical depth, while those offering advanced tools alienate beginner users. Additionally, none of the surveyed platforms provide an integrated, fee-free simulation environment for learning- oriented traders. Zosh Trading is designed to bridge this gap [7].

III.Problem Statement

Despite significant advances in digital trading technology, users continue to face critical practical and technical challenges. The core problems identified are as follows:

- Interface Complexity: Most platforms expose beginners to advanced charts, technical indicators, and financial terminology without progressive disclosure, discouraging participation and increasing the likelihood of uninformed decisions.
- Inadequate Portfolio Management: While holdings tracking is commonly available, real-time profit/loss computation, percentage returns, and investment growth visualization are either absent or poorly presented.
- Security Vulnerabilities: Rising cybersecurity threats, phishing attacks, and session hijacking demand robust authentication and secure session management mechanisms that many platforms inadequately address.
- Cost Barriers: Brokerage fees, hidden transaction charges, and platform levies reduce profitability for small investors and create friction for educational use cases.
- Inconsistent UX: Many platforms lack responsive design, leading to degraded experiences across devices.

Zosh Trading addresses these challenges by providing a secure, modular, and user-centric trading environment that combines simplicity with analytical capability, suitable for both desktop and mobile environments.

IV.Proposed Methodology

A. System Architecture

Zosh Trading adopts a three-tier architecture comprising a Presentation Layer (React.js frontend), an Application Layer (Node.js/Express.js backend with modular microservices), and a Data Layer (MongoDB). This separation of concerns promotes scalability, independent module deployment, and ease of maintenance [8].

The system architecture follows a RESTful API design pattern. Each backend service exposes endpoints consumed by the React.js SPA. JWT tokens are issued at login and validated on every protected route, enforcing stateless, role- based access control.

B. Core Modules

The platform is decomposed into five primary modules:

1) User Authentication Module: Handles user registration and login. Passwords are stored as bcrypt hashes. Upon successful authentication, the server generates a signed JWT token with a configurable expiration. All subsequent requests include this token in the Authorization header for validation.

DOI: https://doi.org/10.5281/zenodo.20079493

2) Dashboard Module: Aggregates and displays trending assets, live price data retrieved via external market APIs, candlestick charts rendered through Chart.js or Recharts, and real-time account balance. Data refresh is managed through periodic API polling.

3)Trading Module: Implements buy and sell operations. The buy algorithm (Algorithm 1) validates user balance before executing a trade; the sell algorithm verifies asset ownership before transferring funds. All transactions are atomically recorded to ensure data consistency.

```
Input: AssetID, Quantity
Algorithm 1: Buy Asset
1. Retrieve current price P for AssetID
2. Compute cost C = P × Quantity
3. IF user.balance ≥ C:
a. Deduct C from user.balance
b. Create transaction record
c. Update portfolio holdings
4. ELSE: Return "Insufficient Funds" error
```

4)Portfolio Module: Continuously computes total investment, current market value, and profit/loss using the formula:
P/L = Current Market Value – Total Investment

Results are displayed as absolute values and percentage returns to provide intuitive investment insight.

5)Admin Module: Provides system administrators with user management, transaction monitoring, and system health dashboards, supporting operational oversight and compliance.

C.Security Design
Security is enforced at multiple layers. JWT tokens are signed using HMAC-SHA256 and carry an expiration claim to limit session duration. Passwords are hashed with bcrypt (work factor 12). Input validation is applied at both frontend and backend layers to mitigate injection attacks. Role-Based Access Control (RBAC) distinguishes user and admin privileges.

D.**Development Methodology**
The project follows the Agile development model, enabling iterative delivery across six phases: Requirement Analysis, Database Design, Backend Development, Frontend Development, Integration Testing, and Deployment. This approach allows continuous refinement based on testing feedback and facilitates parallel frontend-backend development.

V.Feasibility Study
A.multi-dimensional feasibility assessment was conducted prior to system development:

Technical Feasibility: The system relies exclusively on open-source technologies (React.js, Node.js, MongoDB) with extensive community support and documented APIs. Hardware requirements are modest—an i5 processor with 8 GB RAM suffices for development and testing [4].

Economic Feasibility: Zero licensing costs are incurred, as all frameworks and tools are open-source. Hosting can be achieved through free-tier cloud services during initial deployment, minimizing capital expenditure.

Operational Feasibility: The interface is designed for minimal onboarding friction. Automated portfolio computation and real-time dashboards reduce the cognitive load on end users, requiring minimal training.

Legal Feasibility: Operating as a simulation or paper-trading platform avoids regulatory requirements applicable to live brokerage services. User data is protected through authentication tokens and hashed credentials. Future real- market integration would require SEBI/financial regulatory compliance.

Schedule Feasibility: The modular architecture enables parallel development of frontend and backend services, allowing completion within a standard academic semester timeline.

DOI: https://doi.org/10.5281/zenodo.20079493

VI.Results and Discussion

The Zosh Trading platform was implemented and tested across all functional modules. Black-box testing validated login authentication, invalid credential rejection, insufficient balance handling, and transaction integrity. White-box testing covered API endpoint response validation and code path verification for the buy/sell algorithms.

The user authentication module demonstrated consistent JWT generation and token expiration enforcement. The trading module correctly validated balance constraints in all tested scenarios, returning descriptive error messages on failure. Portfolio tracking accurately reflected real-time profit/loss values consistent with the defined formula.

Compared to existing platforms (Table I), Zosh Trading offers a competitive combination of simplicity, security, and analytical functionality without brokerage fees or interface complexity. The modular three-tier architecture ensures that individual services can be independently scaled, addressing performance demands during peak trading activity.

The platform's responsive React.js frontend ensures consistent UX across desktop and mobile browsers, addressing the cross-device inconsistency observed in competing platforms.

VII.Conclusion

This paper presented Zosh Trading, a web-based trading platform built on the MERN stack with a microservices- inspired modular architecture. The system successfully addresses the principal limitations of existing platforms: interface complexity, inadequate portfolio management, security vulnerabilities, and excessive cost barriers.

Through the implementation of JWT-based authentication, RESTful API integration, automated profit/loss computation, and real-time data visualization, Zosh Trading delivers a balanced platform accessible to beginner and intermediate investors alike.

Future enhancements include: (1) integration of AI-based stock prediction models using LSTM neural networks; (2) development of native Android and iOS applications; (3) live brokerage API integration for real-market trading; and

(4) social trading features enabling portfolio sharing and copy-trading.

Acknowledgment

The authors express sincere gratitude to Mr. Ibrar Ahmed, Assistant Professor, Department of Computer Science and Engineering, Noida Institute of Engineering and Technology (NIET), Greater Noida, for his invaluable guidance, continuous support, and constructive feedback throughout the development of this project.

Refernces

[1] I. Sommerville, Software Engineering, 10th ed. Pearson Education, 2016.

[2] Zerodha, "Zerodha – India's largest stock broker," [Online]. Available: https://zerodha.com

[3] Binance,"BinanceCryptocurrency Exchange," [Online]. Available: https://binance.com

[4] MongoDBInc., "MongoDB Documentation," [Online]. Available: https://www.mongodb.com

[5] Node.jsFoundation,"Node.js Documentation," [Online]. Available: https://nodejs.org

[6] Meta Open Source, "React.js OfficialDocumentation," [Online].Available: https://react.dev

[7] Express.js Team, "Express.js Guide," [Online].Available: https://expressjs.com

[8] R. T. Fielding, "Architectural Styles and the Design of Network-based Software Architectures," Doctoral Dissertation, Univ. of California, 2000.

DOI: https://doi.org/10.5281/zenodo.20079493