Archives available at journals.mriindia.com

International Journal of Electrical, Electronics and Computer Systems

ISSN: 2347-2820
Volume 14 Issue 01, 2025

Quiz Rush: A Real-Time Tournament-Based Quiz Application

[1]Kamlesh Kelwade, [2]Saima Ansari, [3]Himanshu Gautam, [4]Md Saad, [5]Sahil Botre

[1] Associate Professor, Computer Science and Engineering, Anjuman College of Engineering and Technology, Sadar, Nagpur, India
[2] Professor, Computer Science and Engineering, Anjuman College of Engineering and Technology, Sadar, Nagpur, India
[3,4,5] B. tech Student, Computer Science and Engineering, Anjuman College of Engineering and Technology, Sadar, Nagpur, India

Peer Review Information

Submission: 07 Feb 2025
Revision: 16 Mar 2025
Acceptance: 18 April 2025

Keywords

Flutter
Leaderboard System
MySQL
Performance Tracking

Abstract

QuizRush is a structured, tournament-based quiz application designed to provide a real-time, competitive quizzing experience. Unlike traditional quiz applications that rely on random matchmaking, QuizRush ensures fairness by scheduling quizzes at fixed intervals, allowing all participants to answer the same set of questions simultaneously. The system is developed using Spring Boot (Java) for the backend and Flutter for the frontend, with MySQL as the database. Real-time quiz execution is enabled using WebSockets for instant question broadcasting and answer submission. QuizRush incorporates secure authentication, admin-controlled quiz management, and a leaderboard system for performance tracking. Future enhancements may include AI-generated questions, global tournaments, and detailed analytics for user engagement. The study highlights the advantages of structured tournament-based quizzing over traditional random matchmaking approaches, making QuizRush an ideal platform for educational institutions, corporate training, and competitive assessments.

INTRODUCTION

Quizzing has been an essential tool for education, entertainment, and knowledge assessment. With the digitalization of learning, quiz applications have gained popularity for various purposes, including academic learning, corporate training, and online competitions. Existing quiz applications primarily focus on either single-player experiences or random matchmaking-based multiplayer modes. However, these approaches lack structured competition, often resulting in imbalanced gameplay.

Structured tournament-based quiz applications address these issues by offering scheduled quizzes where all participants receive the same questions simultaneously. This format enhances engagement, ensures fairness, and fosters competitiveness among users. QuizRush introduces this concept through a robust and scalable technological infrastructure.

The primary motivation for QuizRush stems from the growing demand for structured competitive learning experiences in educational institutions, corporate training programs, and global trivia competitions.

AIM & OBJECTIVE

A. Aim

The primary aim is to design and develop a real-time tournament-based quiz application that enhances user engagement, ensures fair competition, and leverages WebSocket technology for instant question broadcasting and response collection.

B. Objective

- Develop a structured, tournament-based quiz system that schedules quizzes at fixed intervals, ensuring fair competition among players.
- Implement real-time question broadcasting and answer submission using WebSockets for low- latency multiplayer interactions.
- Ensure secure authentication and leaderboard ranking by integrating JWT-based authentication and automated scoring mechanisms.

METHODOLOGY

A. System Design and Development

QuizRush follows a structured software development lifecycle (SDLC) using an agile methodology. The system is designed as a **client-server architecture**, where the mobile frontend (Flutter) interacts with the backend (Spring Boot) through REST APIs and WebSockets. The database (MySQL) stores user data, quiz details, and results.

B. Technologies Used

- Backend: Spring Boot (Java) – Handles user authentication, quiz management, and real-time communication.
- Frontend: Flutter – Provides a cross-platform mobile application interface.
- Database: MySQL – Stores user profiles, quiz questions, scores, and rankings.
- Real-Time Communication: WebSockets – Ensures instant question broadcasting and answer submission.
- Security Mechanisms: JWT (JSON Web Tokens) – Secure authentication and access control.

C. System Workflow

1. User Registration & Authentication – Users sign up and log in securely using JWT-based authentication.
2. Quiz Creation & Scheduling – Admins create and schedule quizzes at fixed intervals.
3. Real-Time Quiz Execution – When a quiz starts, questions are broadcasted to all participants using WebSockets.
4. Answer Submission & Scoring – Players submit answers in real-time, and the system records responses.
5. Leaderboard & Performance Tracking – Scores are calculated instantly, and rankings are displayed on the leaderboard.

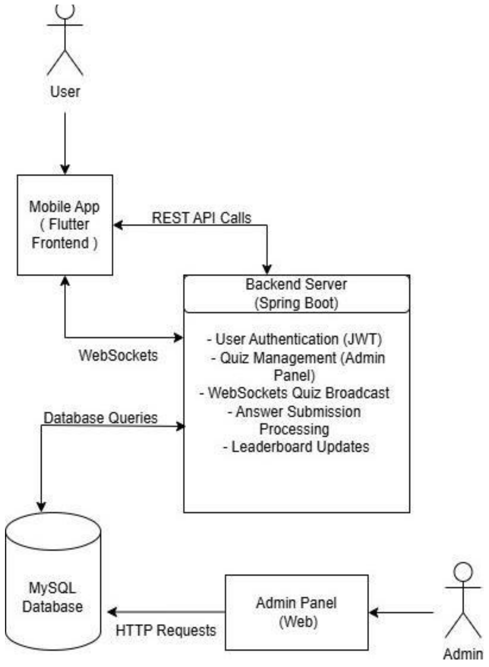

User
Mobile App (Flutter Frontend)
REST API Calls
Backend Server (Spring Boot)
- User Authentication (JWT)
- Quiz Management (Admin Panel)
- WebSockets Quiz Broadcast
- Answer Submission Processing
- Leaderboard Updates
WebSockets
Database Queries
MySQL Database
HTTP Requests
Admin Panel (Web)
Admin

Fig. Workflow of QuizRush

D. Testing & Evaluation

To ensure the efficiency, security, and user engagement of QuizRush, the system underwent rigorous testing under different conditions:

Performance Testing:

- Measured server response times and WebSocket latency under varying user loads.
- Conducted stress testing to evaluate system scalability with concurrent players.

Security Testing:

- Verified JWT authentication to prevent unauthorized access and session hijacking.
- Performed penetration testing to identify potential vulnerabilities in user authentication and data transmission.

User Experience & Engagement Analysis:

- Conducted usability testing with real users to assess interface intuitiveness and ease of navigation.
- Gathered user feedback through surveys to evaluate engagement levels and gameplay satisfaction.

IMPLEMENTATION

The implementation of QuizRush involved setting up the backend, frontend, database, and real-time communication functionalities.

A. Backend Development

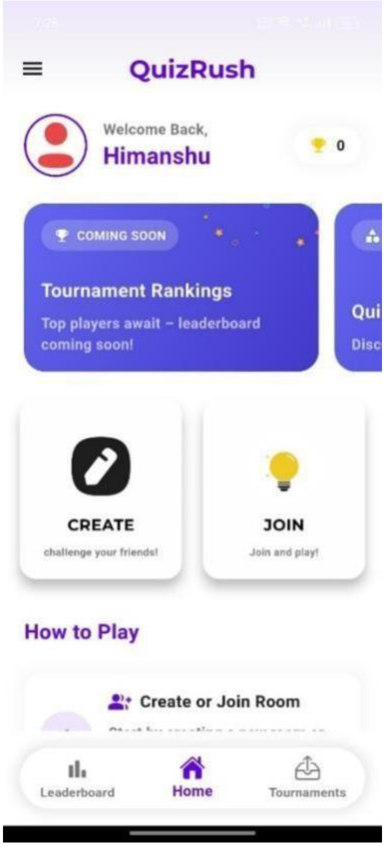
- Implemented using Spring Boot (Java) for managing user authentication, quiz logic, and data storage.
- WebSocket integration for real-time

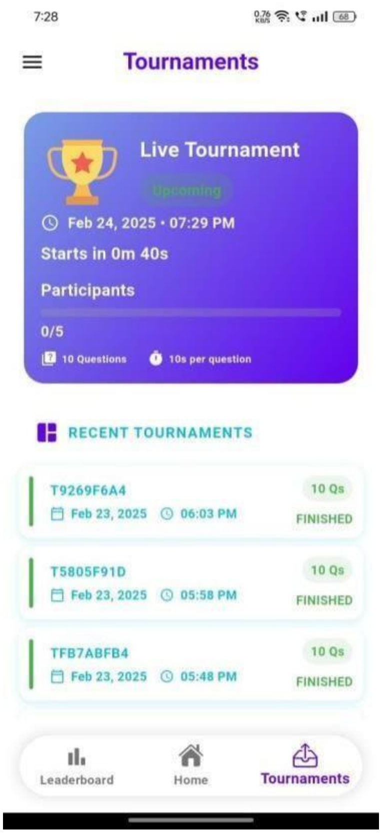
question broadcasting and answer collection.

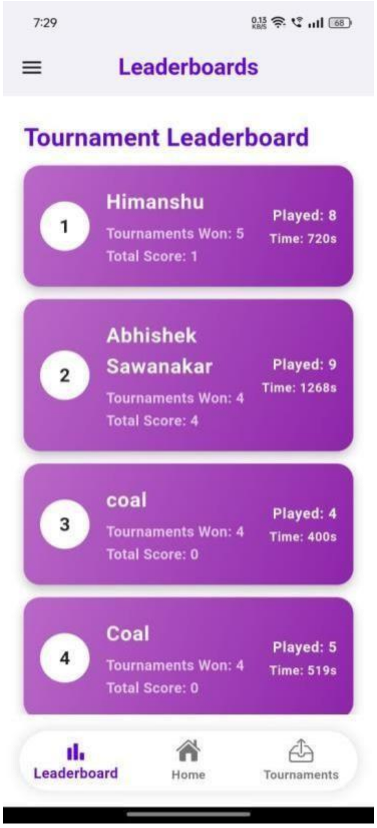
- JWT-based authentication for secure access control.

B. Frontend Development

- Built using Flutter for a responsive and interactive user experience.
- Features include quiz participation, leaderboard visualization, and profile management.







C. Database Management

- MySQL database for storing user accounts, quizzes, and scores.
- Implemented indexing and optimization for faster query execution.

RESULTS AND DISCUSSION

After implementing the core functionalities, QuizRush was tested for performance, accuracy, and scalability.

A. Key Observations

- Minimal Latency - Real-time quiz execution was successfully achieved using WebSockets.
- Scalability - The system maintained stable performance under high concurrent user loads.
- Security - Secure authentication mechanisms prevented unauthorized access.
- Leaderboard Functionality - Automated scoring and ranking system performed as expected.

B. Discussion

1. Effectiveness of Tournament-Based Quizzes: Users found structured quizzes more engaging compared to random matchmaking systems.
2. Real-Time Performance: WebSocket technology ensured low latency, making the quiz experience smoother.
3. Security & Authentication: JWT-based authentication provided robust protection against unauthorized access.
4. Future Enhancements: AI-driven question generation and adaptive difficulty levels can further improve user engagement.

CONCLUSION

The QuizRush system successfully addresses the limitations of traditional quiz applications by offering real- time, structured, and competitive quiz gameplay. It provides secure authentication, low-latency question broadcasting, and dynamic leaderboard updates, ensuring fairness and transparency in quiz participation.The system performed efficiently under high user loads, maintaining low response times and stable performance. Security mechanisms such as JWT authentication, encrypted password storage, and secure WebSocket connections were implemented to protect user data and prevent unauthorized access.While the system demonstrates strong real-time capabilities, certain limitations, such as internet dependency and the lack of AI-based question generation, present opportunities for further development.

References

Spring Framework Documentation (2023). WebSocket Messaging with Spring Boot: Performance and Scalability.

Rana, M., Pandey, A., Mishra, A., & Kandu, V. (2023). Enhancing Data Security: A Comprehensive Study on the Efficacy of JSON Web Token (JWT) and HMAC SHA-256 Algorithm for Web Application Security. International Journal on Recent and Innovation Trends in Computing and Communication, 11(9), 4409–4416.

IEEE Conference Paper (2022). Comparative Study of WebSockets vs. REST APIs for Real-Time Quiz Applications. IEEE International Conference on Cloud Computing and Intelligent Systems.

Brown, T., & Miller, P. (2022). Optimizing WebSocket Performance for Real-Time Multiplayer Applications. ACM Digital Library.

Gudes, E., & Shen, H. (2021). Efficient Database Indexing Strategies for Real-Time Web Applications. IEEE Transactions on Knowledge and Data Engineering, 33(8), 3567-3579.

Wang, A. I., & Tahir, R. (2020). The Effect of Kahoot! on Learning Outcomes: A Literature Review. Computers & Education, 149, 103818.

Felszeghy, S., et al. (2019). Using Online Game-Based Platforms to Improve Student Performance and Engagement in Histology Teaching. BMC Medical Education, 19, 273.

Bhatia, V., Joshi, S., & Chapaneri, R. (2019). WebSocket-Evented Real-Time Online Coding Collaborator. In Smart Intelligent Computing and Applications (pp. 325–334). Springer, Singapore