CodeClash

A PROJECT REPORT
for
Mini Project-II (ID201B)
Session (2024-25)

Submitted by

Mohammad Daud
202410116100121
Mayank Srivastava
202410116100118

Submitted in partial fulfilment of the
Requirements for the Degree of

MASTER OF COMPUTER APPLICATION

Under the Supervision of
Ms. Shruti Aggarwal
Assistant Professor

Submitted to

DEPARTMENT OF COMPUTER APPLICATIONS
KIET Group of Institutions, Ghaziabad
Uttar Pradesh-201206
(MARCH - 2025)

Declaration

We, the undersigned, hereby declare that the Mini Project titled "**CodeClash**" is an original work completed by us as part of the curriculum requirement for the course under the Master of Computer Applications (MCA) program at **KIET Group Of Institutions**.

We affirm that we have undertaken this during the academic year 2024-25 under the guidance of **Ms. Shruti Aggarwal**. All the content and ideas presented in this report are the result of our own efforts, except where explicitly stated otherwise. Proper citations have been provided wherever references to external sources have been made.

We further declare that this project has not been submitted, either in part or in full, to any other university or institution for any degree or diploma.

Mohammad Daud

Mayank Srivastava

CERTIFICATE

Certified that **MOHAMMAD DAUD (202410116100121), MAYANK SRIVASTAVA (202410116100118)** has carried out the project work having "**CodeClash**" (**Mini Project-II, ID201B**) for **Master of Computer Application** from Dr. A.P.J. Abdul Kalam Technical University (AKTU**)** (formerly UPTU), Lucknow under my supervision. The project report embodies original work, and studies are carried out by the student himself, and the contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Ms. Shruti Aggarwal
Assistant Professor
Department of Computer Applications
KIET Group of Institutions, Ghaziabad

Dr. Akash Rajak
Dean
Department of Computer Applications
KIET Group of Institutions, Ghaziabad

CodeClash

Mohammad Daud
Mayank Srivastava

Abstract

The CodeClash is a web application designed for students, which is educational platform where more than one student can compete with each other. In CodeClash, student can create a room and invite his/her friends with the invitation code, and after putting the invitation code and start the test, after completion of the test a result is generated and student can also get to know about the rank. This project addresses inefficiencies in traditional educational model and ensures accuracy and quality education.

Key features of the system include:

- A user-friendly interface built using React and Tailwind CSS.
- Real-time ranking on leaderboard.
- Focused on quality education, feels like a game.
- Secure and scalable architecture for efficient performance.

This project demonstrates the use of modern web technologies to enhance administrative workflows in educational institutions.

Keywords: CodeClash, coding, React, Tailwind CSS, SpringBoot, Java

Acknowledgments

Success in life is never attained single-handedly. My deepest gratitude goes to my project supervisor, **Ms. Shruti Aggarwal,** for her guidance, help, and encouragement throughout my project work. Their enlightening ideas, comments, and suggestions.

Words are not enough to express my gratitude to **Dr. Akash Rajak**, Professor and Dean, Department of Computer Applications, for his insightful comments and administrative help on various occasions.

Fortunately, I have many understanding friends, who have helped me a lot on many critical conditions.

Finally, my sincere thanks go to my family members and all those who have directly and indirectly provided me with moral support and other kind of help. Without their support, completion of this work would not have been possible in time. They keep my life filled with enjoyment and happiness.

Name - Mohammad Daud

Roll No - 202410116100121

Name – Mayank Srivastava

Roll No - 202410116100118

TABLE OF CONTENTS

LIST OF FIGURES

1. Introduction

In today's fast-paced world of technology, competitive coding has become an essential skill for aspiring developers. While platforms like LeetCode, CodeChef, and HackerRank provide a rich collection of problems, they primarily focus on individual problem-solving rather than real-time competition. CodeClash is designed to bridge this gap by introducing a real-time multiplayer coding battle platform where users can compete with their friends in a dynamic, engaging environment.

CodeClash allows users to register, create a profile, and challenge peers in live coding battles. Users can create a custom challenge room, select a difficulty level and topic, and invite friends via an invitation code. Once all participants join, the competition begins. After the timer expires, results are displayed, ranking users based on their accuracy and speed.

The platform is not just about winning; it also encourages collaboration, skill improvement, and interactive learning. By integrating real-time communication, instant feedback, and automated result generation, CodeClash provides an experience that is both educational and highly competitive.

1.1 Overview

A good system must meet the following criteria:

- Performance: Fast and efficient operation.
- Reliability: Consistent and accurate results.
- Scalability: Ability to grow and support more users.

1.2 Basic Communication Model

CodeClash operates on a client-server model, where the frontend (React.js) communicates with the backend (Spring Boot) using REST APIs. Users interact with the web application through a seamless UI, while the backend handles user authentication, room management, challenge execution, and result processing.

1.3 Data Communication

The data communication model ensures that user actions are processed and stored efficiently. The platform follows a structured request-response cycle, where:

- Frontend (React.js) sends requests (user authentication, room creation, challenge submissions) to the backend.
- Backend (Spring Boot) processes these requests, validates data, and interacts with the database.
- MongoDB stores user profiles, challenge data, and rankings.
- WebSockets enable real-time updates for challenge progress, peer status, and live leaderboards.

By integrating Spring Security for authentication and MongoDB for data persistence, CodeClash ensures scalability, security, and efficiency in handling multiple real-time competitions.

2. Literature Review

CodeClash is a real-time multiplayer coding competition platform designed to bring an interactive and engaging experience to competitive programming. Unlike traditional coding platforms, where challenges are solved independently, CodeClash allows users to compete with their friends and peers in real-time battles. The platform is built to enhance learning, encourage collaboration, and improve problem-solving speed in a competitive setting.

2.1 Key Features of CodeClash

- User Registration & Profile Management – Users can create an account, log in, and manage their profiles, including editing their name and date of birth (but not email or username).
- Room Creation & Challenge Setup – Users can create custom challenge rooms by selecting a difficulty level and topic.
- Invitation System – Users can invite their friends to join a challenge room using a unique invitation code.
- Real-Time Competitive Coding – Once all participants enter the room, the test begins simultaneously, and users work on solving coding problems.
- Timer-Based Competitions – Each challenge has a fixed time limit, after which results are automatically generated.
- Ranking & Result Generation – Once the competition ends, user rankings are displayed based on performance, such as accuracy and time taken to solve the problems.

2.2 Role of Modern Web Technologies

Key Technologies Used in CodeClash

1. Frontend – React.js (React Framework)

- The frontend of CodeClash is built using React.js, a React-based framework that enhances performance and provides server-side rendering (SSR) and static site generation (SSG).

🔷 Why React.js?

- Improves SEO and page load speed with server-side rendering (SSR).
- Supports fast client-side navigation for a smooth user experience.
- Allows easy integration with real-time features (e.g., WebSockets for live updates).

🔷 User Interface (UI) Technologies

- Tailwind CSS for a responsive, modern, and customizable UI.
- React Hooks for state management and dynamic UI updates.

2. Backend – Spring Boot (Java Framework)

The backend is developed using Spring Boot, a powerful Java framework that simplifies backend development, authentication, and API handling.

🔷 Why Spring Boot?

- Provides a robust and scalable architecture for handling multiple user requests.
- Supports Spring Security for secure user authentication and authorization.
- Easily integrates with databases (MongoDB) and external services.

🔷 Core Backend Features in CodeClash

- RESTful APIs for user authentication, room creation, challenge execution, and result processing.
- Role-based authentication using Spring Security (JWT tokens for secure access).
- Efficient data handling with MongoDB.

3. Database – MongoDB (NoSQL Database)

The platform uses MongoDB, a NoSQL database, to store and manage dynamic data related to user profiles, challenges, results, and rankings.

🔷 Why MongoDB?

- Schema flexibility to store various types of data (users, challenges, results).
- High performance and scalability to handle large numbers of participants.
- Fast queries for real-time leaderboard updates and competition results.

4. Real-Time Communication

WebSockets, To ensure real-time updates, CodeClash can integrate WebSockets, which allow live interaction between users.

🔷 Benefits of WebSockets in CodeClash

- Enables real-time status updates (e.g., players joining a room).
- Updates leaderboards dynamically as users submit answers.
- Reduces server load compared to frequent HTTP polling.

5. Authentication & Security – Spring Security & JWT

Security is crucial for any multiplayer platform. CodeClash uses Spring Security and JWT (JSON Web Tokens) for user authentication and session management.

🔷 Security Features in CodeClash

- Password hashing & encryption for secure user data storage.
- JWT-based authentication to manage user sessions.

- Access control to ensure only authorized users can join specific rooms.

3. Project Flow

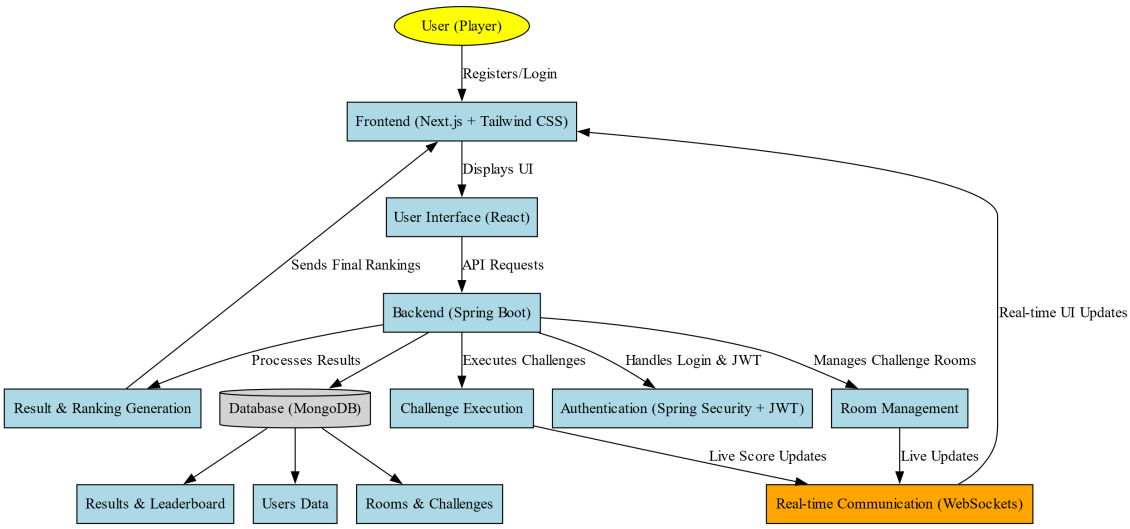

Figure – 3.1

3.1 Use Case Diagram

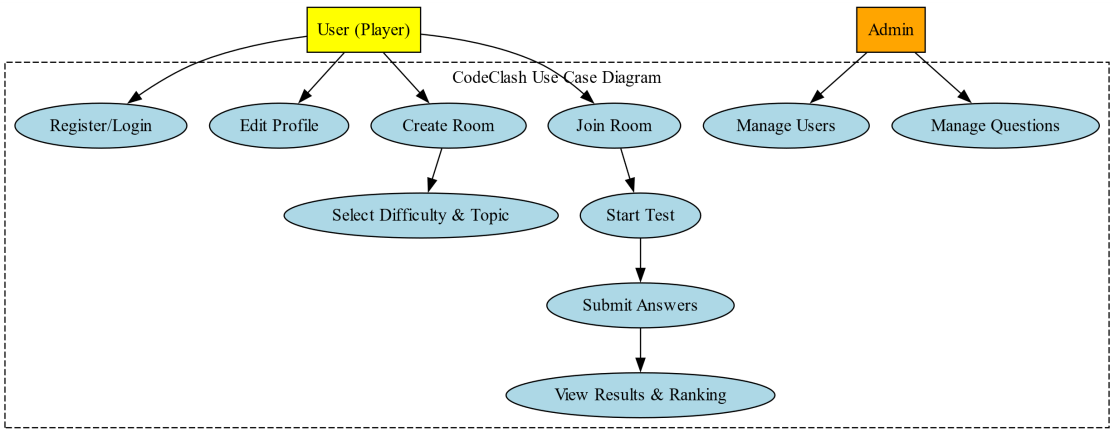

Figure – 3.2

3.2 Level 0 DFD

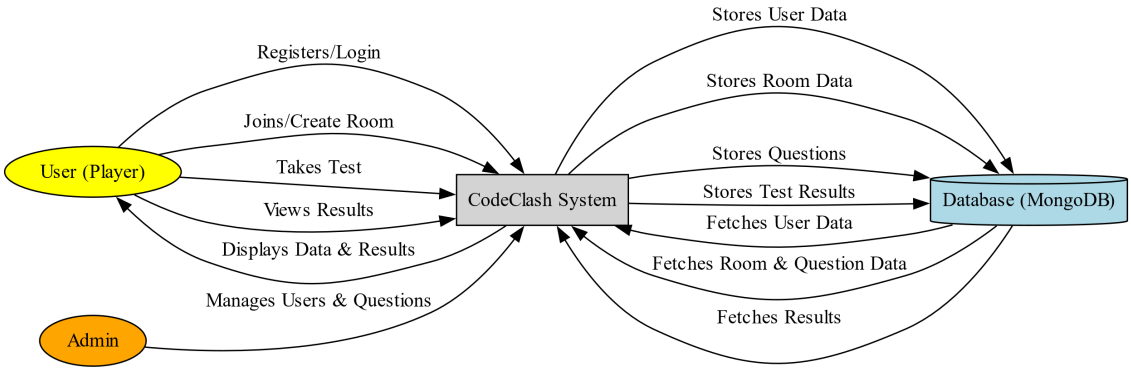

Figure – 3.3

3.3 Level 1 DFD

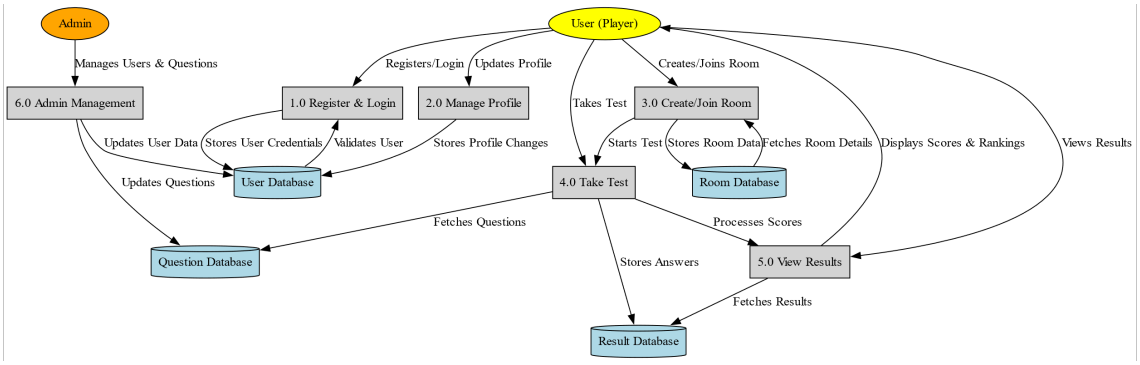

Figure – 3.4

3.4 Level 2 DFD

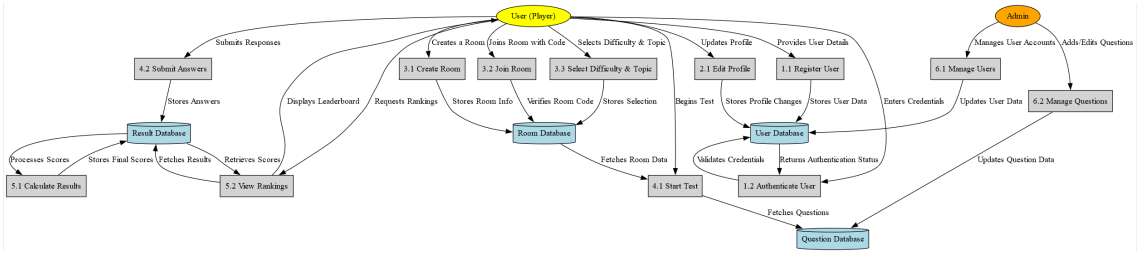

Figure – 3.5

3.2 ER Diagram

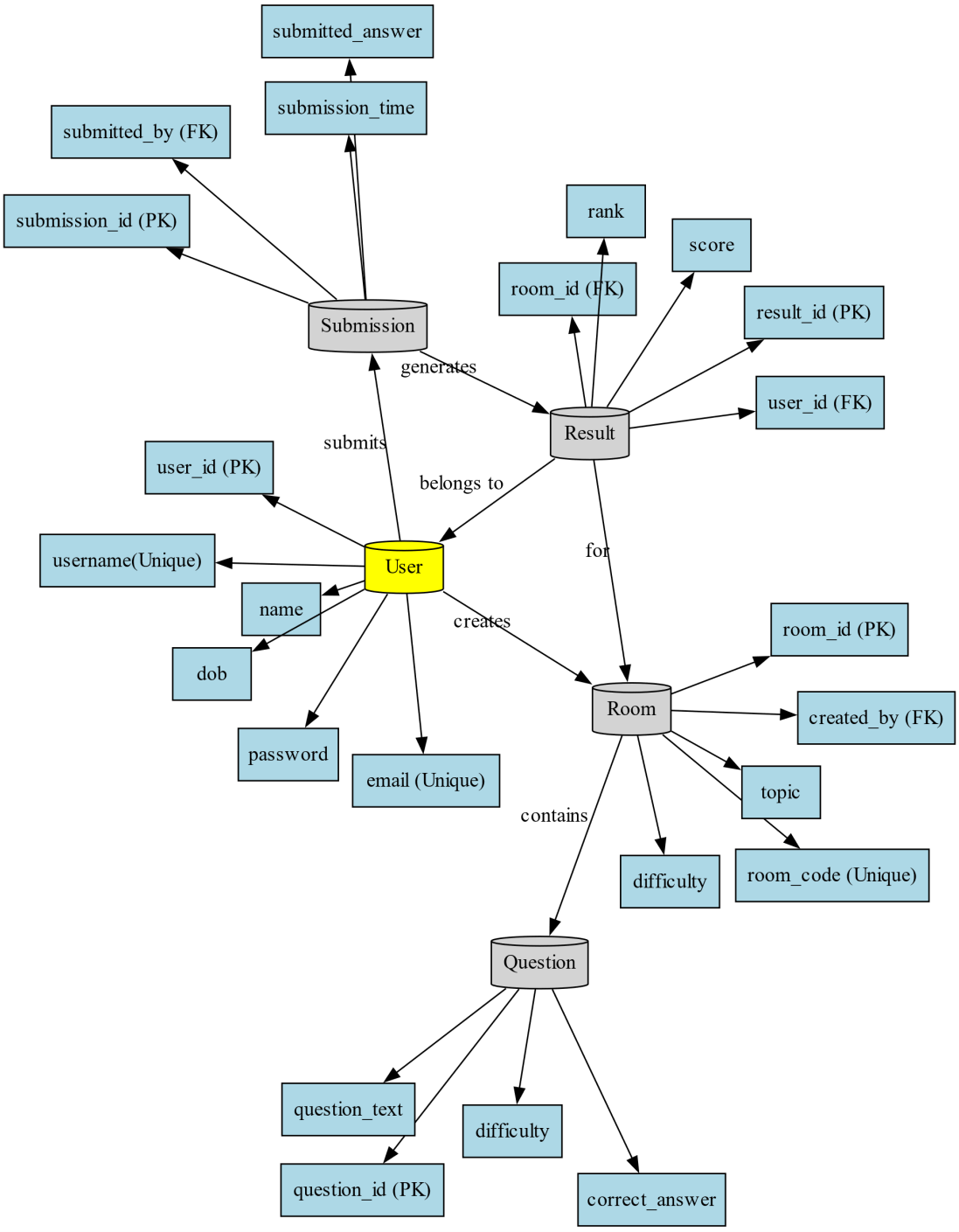

Figure – 3.6

4. Database Tables for CodeClash

4.1 Users Table

Field	Type	Description
UserID	String (PK)	Unique User ID
Name	String	Full name
Email	String	User's email address
Password	String	Hashed password
Role	String	Role (Student/Admin)
DOB	Date	Date of Birth

4.2 Rooms Table

Field	Type	Description
RoomID	String (PK)	Unique Room ID
HostID	String (FK)	User ID of the room creator
Topic	String	Selected topic
Difficulty	String	Selected difficulty level
CreatedAt	Timestamp	Timestamp of room creation

4.3 Room Participants Table

Field	Type	Description
ParticipantID	String (PK)	Unique Participant ID
RoomID	String (FK)	Room the user has joined
UserID	String (FK)	ID of the user joining the room

4.4 Questions Table

Field	Type	Description
QuestionID	String (PK)	Unique Question ID
Topic	String	Category or subject
Difficulty	String	Question difficulty level
Question	Text	The actual question
OptionA	String	Option A
OptionB	String	Option B
OptionC	String	Option C
OptionD	String	Option D
CorrectAns	String	Correct answer option (A/B/C/D)

4.5 Submissions Table

Field	Type	Description
SubmissionID	String (PK)	Unique submission ID
RoomID	String (FK)	Room where the test was taken
UserID	String (FK)	User who submitted answers
QuestionID	String (FK)	Question answered
SelectedAns	String	User's selected answer (A/B/C/D)
IsCorrect	Boolean	Whether the answer was correct

4.6 Results Table

Field	Type	Description
ResultID	String (PK)	Unique Result ID
RoomID	String (FK)	Room where the test was taken
UserID	String (FK)	User who participated
Score	Integer	Score obtained
Rank	Integer	User's rank in the test

5. System Structure

5.1 Overview

The system structure of CodeClash follows a client-server architecture where users interact with the platform via a web interface, and the backend processes requests, manages data, and ensures real-time interactions. The platform is designed using Spring Boot for the backend, React.js for the frontend, MongoDB as the database, and WebSockets for real-time communication.

5.2 System Architecture

The CodeClash platform consists of the following layers:

5.2.1 Frontend Layer (Client-Side)

Developed using React.js and Tailwind CSS.

- Provides an intuitive user interface for registration, login, profile management, quiz creation, and result visualization.
- Implements WebSockets for real-time communication.

5.2.2 Backend Layer (Server-Side)

- Built using Spring Boot and Spring Security for authentication.
- Handles user management, room creation, question selection, and leaderboard generation.
- Uses WebSockets to handle multiplayer quiz sessions in real time.

5.2.3 Database Layer

- Uses MongoDB for storing user data, quiz questions, submissions, and results.
- The database schema ensures data integrity and quick access.

5.2.4 Communication Layer

- Uses REST APIs for standard HTTP requests.
- Implements WebSockets for real-time synchronization between players during the quiz.

5.2.4 System Flow

- User Authentication
  - A user registers using email and password.
  - After login, Spring Security generates a JWT token for authentication.
  - The user is redirected to the home page.
- Room Creation & Quiz Setup
  - Users create a quiz room by selecting a topic and difficulty level.
  - A unique invitation code is generated for inviting friends.
  - Peers enter the invitation code to join the same quiz session.
- Quiz Execution
  - Once all users join, the quiz starts in real time.
  - Questions are displayed one by one, and users submit answers.
  - WebSockets handle synchronization, ensuring the same question appears for all users at the same time.

5.2.5 Result Processing

- After time runs out, the system evaluates the submissions.
- The leaderboard is generated based on correct answers and time taken.
- Users can view their ranks and scores.

6. Implementation Phases

Phase 1: Frontend Development

- User Registration & Authentication
  - Developed the login and signup pages using React.js.
  - Used React Hooks and Tailwind CSS for UI components.
  - Integrated JWT authentication for secure login/logout.
- Dashboard & Profile Management
  - Implemented a dashboard displaying user details.
  - Allowed users to edit name and DOB, but restricted email/username edits.

Phase 2: Backend Development

- User Authentication with Spring Security
  - Implemented JWT-based authentication.
  - Used Spring Security's SecurityContextHolder for session management.
- Room & Quiz Management
  - Created REST endpoints for creating, joining, and starting a quiz room.
  - Used WebSockets to sync real-time quiz sessions.

Phase 3: Real-time Quiz Handling

- WebSockets Integration
  - Used WebSockets for real-time updates between participants.
  - Implemented event-driven messages for quiz questions and answers.
- Leaderboard Calculation
  - Calculated scores based on speed and accuracy.
  - Stored rankings in the MongoDB database.

Phase 4: Testing & Deployment

- Unit & Integration Testing
  - Tested API endpoints using Postman and JUnit.

- Debugged WebSocket communication using Spring WebSocket logs.
- Deployment
- Deployed frontend on Vercel and backend on AWS.
- Used MongoDB Atlas for cloud-based database storage.

7. Project Outcome

- Successfully implemented real-time multiplayer quiz battles.
- Achieved secure authentication and leaderboard generation.
- Users can compete with friends in a fair & engaging way.

8. Testing Phase for CodeClash

- The testing phase ensures that CodeClash operates smoothly, providing a bug-free, secure, and efficient multiplayer quiz platform. The platform undergoes multiple testing methodologies, including unit testing, integration testing, functional testing, performance testing, and security testing.

Testing Methodologies:

8.1 Unit Testing

- Test individual components (controllers, services, repositories) to ensure they function correctly.

8.2 Integration Testing

- Verify that different modules work together correctly.
- Tools Used: Postman (API Testing), Spring Boot Integration Testing

8.3 Functional Testing

- Ensure that CodeClash meets the functional requirements.

Approach:

- Manual testing of the entire user journey.

- Automated UI testing using Cypress for React.js frontend.

8.4 Performance Testing

- Ensure CodeClash handles multiple users efficiently.

8.5 Security Testing

- Identify vulnerabilities in authentication, database access, and data security.

8.6 Bug Fixing & Debugging Process

- During testing, several issues were identified and resolved.

8.7 Deployment Readiness

- After successful testing, CodeClash was tested across different devices & browsers, optimized for scalability, and is now ready for deployment.

6. Conclusion

CodeClash successfully provides a real-time multiplayer quiz competition platform where users can challenge their peers in a structured and engaging environment. The project integrates modern web technologies, including React.js for the frontend, Spring Boot for the backend, and MongoDB for data storage.

Throughout the development lifecycle, various aspects such as user authentication, real-time quiz battles, leaderboard ranking, and invitation-based room creation were implemented with a focus on security and performance. The use of Spring Security ensures data protection, while WebSockets enable real-time interactions among users.

Comprehensive testing, including unit, integration, functional, performance, and security testing, was conducted to ensure CodeClash operates smoothly under different conditions. The platform was optimized for scalability and responsiveness, making it ready for deployment.

In conclusion, CodeClash successfully bridges the gap between learning and competition by offering an interactive and engaging quiz platform. Future enhancements may include AI-based question recommendations, adaptive difficulty levels, and integration with external learning platforms to further enrich the user experience.

7. References

Spring Boot Documentation – https://spring.io/projects/spring-boot

React.js Official Documentation – https://Reactjs.org/docs

MongoDB Documentation – https://www.mongodb.com/docs/manual/

WebSockets–https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API

Spring Security Documentation – Available at: https://docs.spring.io/spring-security/reference/

JUnit 5 User Guide – https://junit.org/junit5/docs/current/user-guide/

Postman API Testing – https://learning.postman.com/docs/getting-started/introduction/