Mini Project Report

on

Interactive Online Quiz Platform

Submitted by

Ashritha Azmeera (22BCS019)
Pj Vineeth Kumar (22BCS086)
Rishika P (22BCS099)

Under the guidance of

Dr. Shirshendu Layek

Assistant Professor, Data Science and Artificial Intelligence

Department of Computer Science and Engineering
Indian Institute of Information Technology Dharwad

10 /04 /2025

Certificate

This is to certify that the project, entitled **Interactive Online Quiz Platform**, is a bonafide record of the Mini Project coursework presented by the students whose names are given below during 2024-2025 in partial fulfilment of the requirements of the degree of Bachelor of Technology in Computer Science and Engineering.

Roll No	Names of Students
22BCS019	Ashritha Azmeera
22BCS086	PJ Vineeth Kumar
22BCS099	Rishika P

Dr. Shirshendu Layek
(Project Supervisor)

Contents

List of Figures

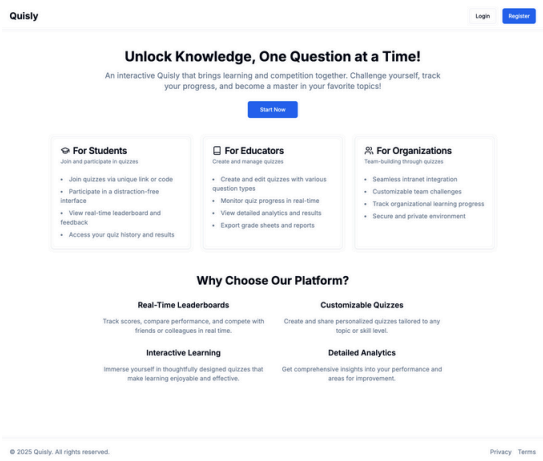

Fig 1.1: Quisly Homepage

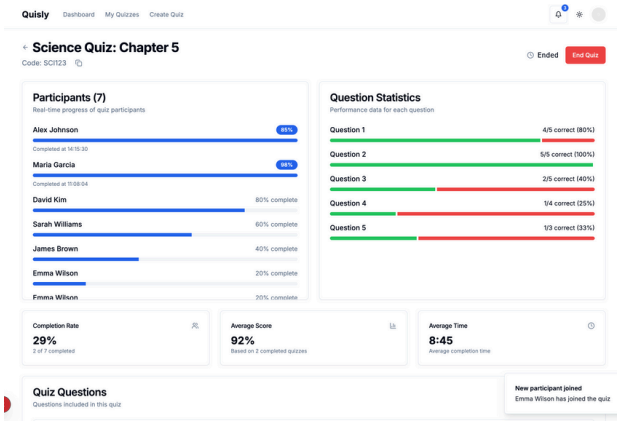

Fig 1.2: Realtime Statistics for the Faculty

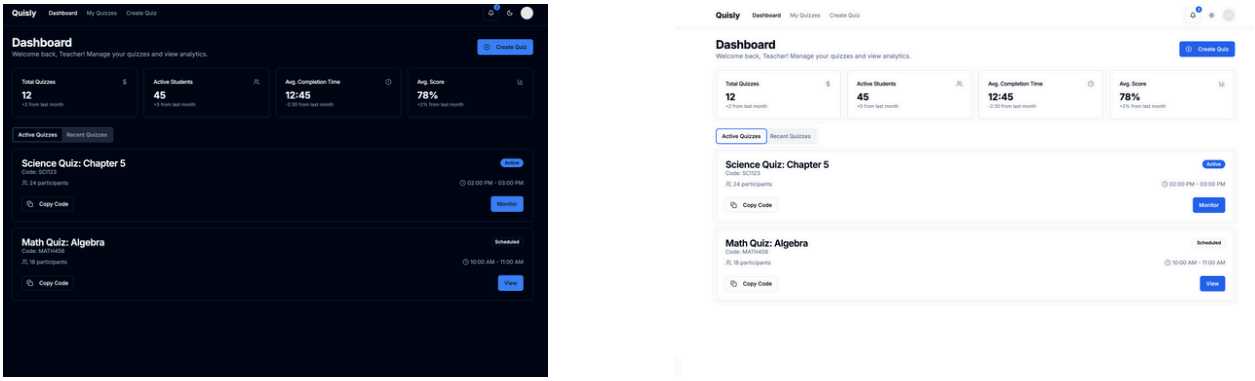

Fig 1.3: Faculty Dashboard pages in Light and Dark modes (left-dark mode, right-light mode)

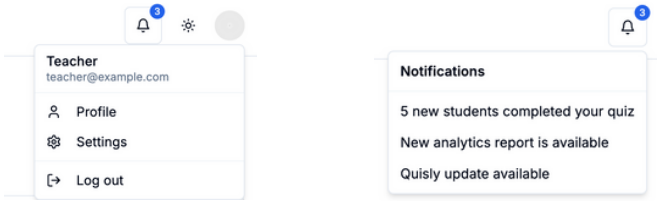

Fig 1.4: Settings and Notifications in Navabar

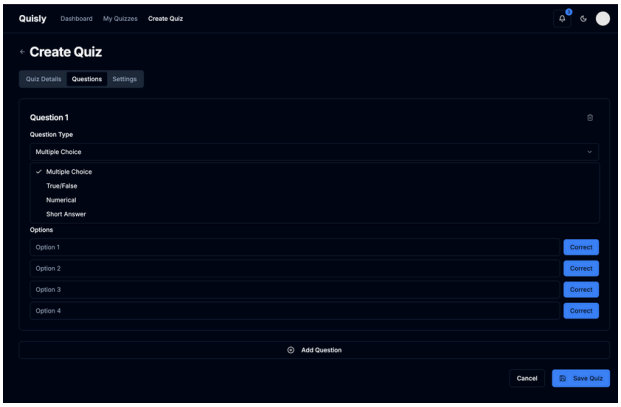

Fig 1.5: Faculty Quiz Creation page

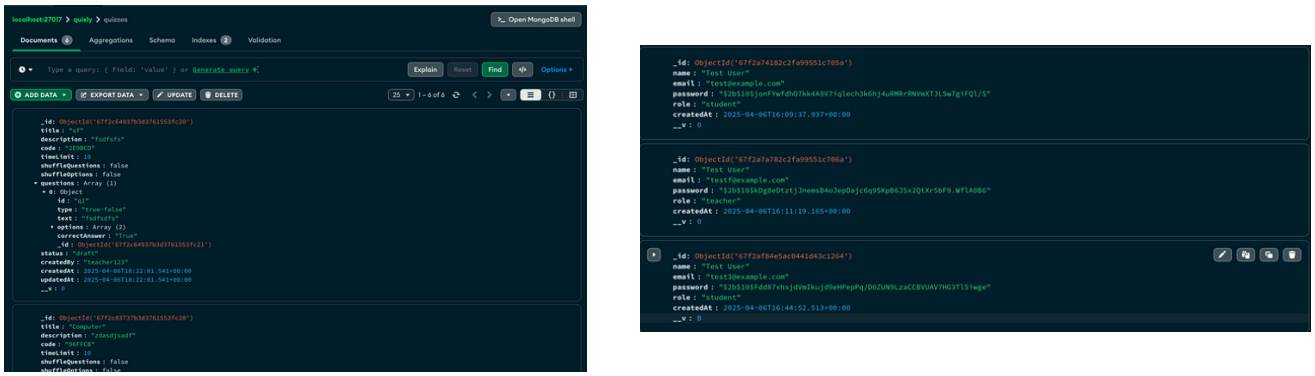

Fig 1.6: Database of the created quizzes

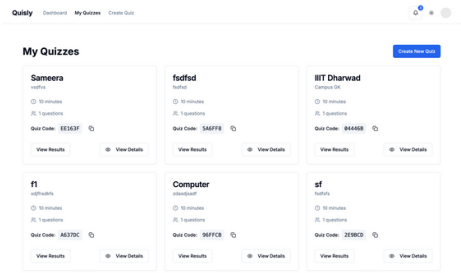

Fig 1.7: My Quizzes page showcasing all the quizzes created by the faculty

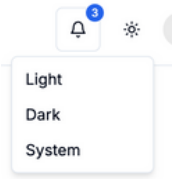

Fig 1.8: Different Mode options present in the navbar

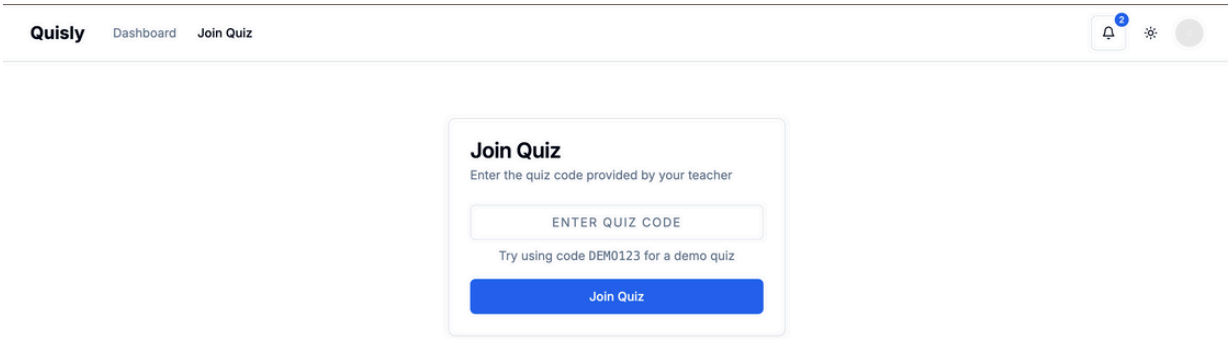

Fig 1.9: Join Quiz Pop up for the student

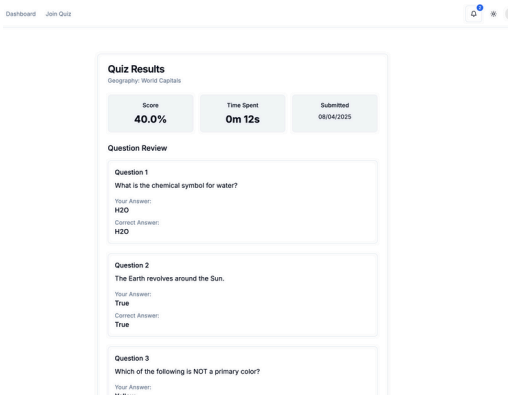

Fig 1.10: Real-Time quiz results shown to the student

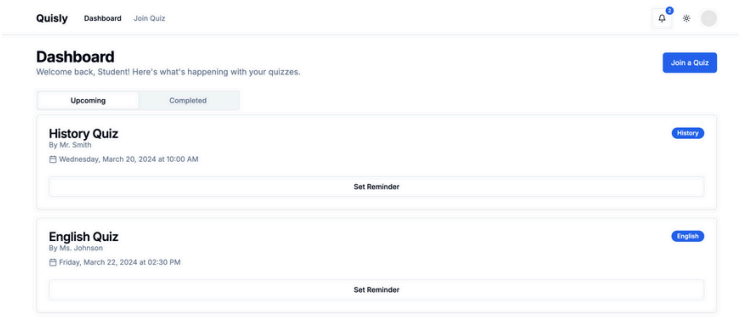

Fig 1.11: Student Dashboard

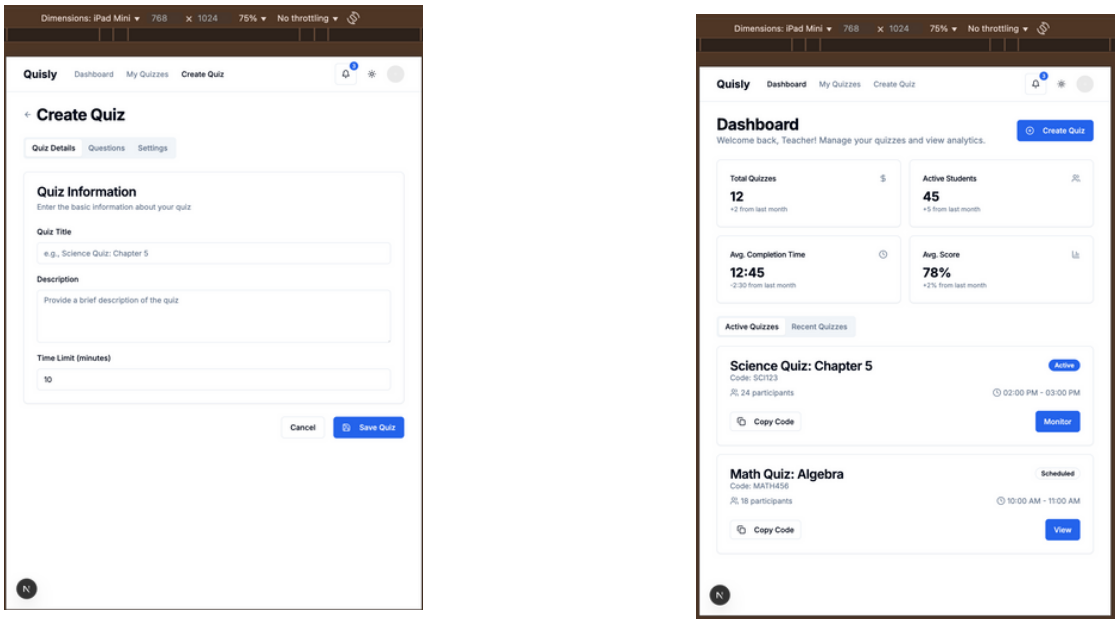

Fig 1.12: Faculty dashboard and create quiz pages in the iPad Mini versions showcasing the responsiveness of the website

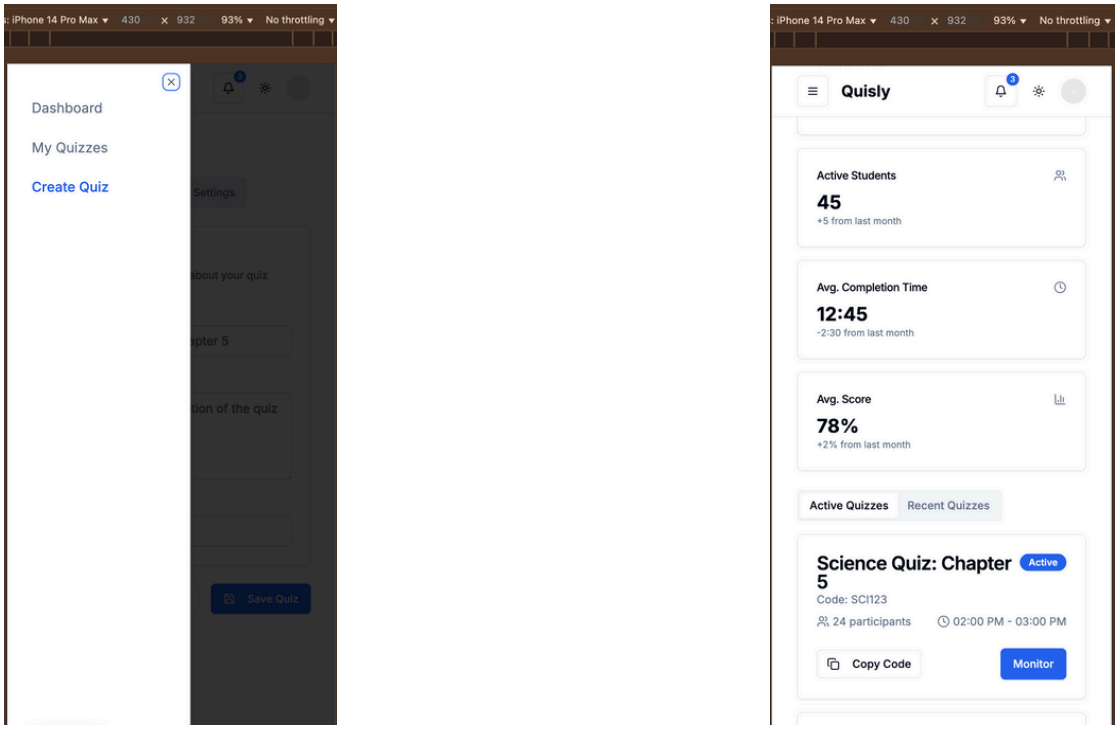

Fig 1.13: Faculty dashboard and create quiz pages in the iPhone versions showcasing the responsiveness of the website

1. Introduction

1.1. Overview of the project

With the increase in usage of digital platforms in education, online assessments have become an important part of the teaching-learning. The traditional everyday methods of conducting quizzes and tests often come with a lot of challenges like logistical constraints, manual evaluation mistakes, and lack of real-time engagement. To tackle these challenges, we have developed an Interactive Online Quiz Platform which is a web-based system that allows teachers to create and manage quizzes while allowing students to participate without any hassle.This platform is designed to be user-friendly, efficient, and scalable, ensuring that both students and teachers have a smooth experience. Teachers can create quizzes with various question types, set time limits, monitor real-time responses, and analyze results with real time leaderboards and automated grading system. On the other hand, students can join quizzes through a unique code or link, attempt questions in a structured interface, and view their scores on a real time leaderboard.
The frontend of this platform is built majorly using HTML and CSS, ensuring a creative and responsive interface. The backend is powered by JavaScript, Node.js, Express, and Socket.io, enabling real-time interactions with the students and teachers such as leaderboard updates and quiz progress tracking.

1.2. Problem Statement

There are many existing online quiz platforms that have inherent restrictions that inhibit their usability in academia and professional settings. These restrictions include a low level of engagement, owing to the absence of dynamic features such as real-time feedback and leaderboards that would render the experience more interactive and motivational. The faculty, therefore, would have to go through the rigors of evaluating responses manually, which is tedious and inefficient. The diversity of question types further hinders one with an assessment of a wider range of skills, and the more common ones would typically be simple multiple-choice formats. One other common limitation is a bad user experience, with both educators and learners being put off by cluttered and non-intuitive interfaces that make navigation through the platform a challenge. Besides, many of these platforms struggle with scaling in an efficient manner, especially when multiple users are trying to access quizzes at the same time. In this context, ensuring fairness and preventing malpractice constitute another serious challenge, highlighting the need for secure and well-monitored environments that can achieve a transparent and equitable assessment for all involved.

2. Objectives

The main objective of this project is to develop a fully functional online quiz platform that provides a hassle free experience for both the teachers and the students.The key objectives of this project are:

2.1 Creating and Managing Quizzes

Multiple Choice, True/False, Numerical, and Short Answer are among the forms in which teachers may build quizzes.Choices for choosing answers, time limitations, and question shuffles are provided as features to the teacher.Performance tracking and detailed analytics make result management simple.

2.2 Student Involvement & Instantaneous Engagement

In a distraction-free interface, students participate in quizzes via special links or codes.Socket.io enables realtime progress tracking and real-time leaderboards.Teachers may engage in dynamic interaction and keep an eye on ongoing quizzes.This makes sure that there is a seamless student-teacher interaction.

2.3 Automated Assessment and User Interface

Real-time score computation with performance reports and thorough feedback is included as a prime feature.The results can be exported to Excel or PDF for grading.An additional feature which makes the user interface better is a streamlined, responsive interface that allows for individualization of branding and dark mode.

2.4 Project Scope

By bridging the gap between traditional quizzes and digital learning platforms, this project seeks to be an efficient evaluation tool for teachers and students. The features put in place will guarantee that quizzes are interesting, dynamic, and simple to administer, which makes the platform perfect for online courses, training facilities, colleges, and universities.

2.5 Primary Objectives and Intentions

2.5.1 Effective Quiz Development & Administration

Make it simple for educators to develop tests with multiple-choice, true/false, numerical, and short-answer question types.Offers a user-friendly dashboard for tracking student progress, managing quizzes, and viewing responses.Give quizzes the ability to be customized with time limitations, question shuffles, and randomization choices.

2.5.2 Smooth Student Involvement & Participation

To ensure secure access, let students join quizzes using a special code or link.To encourage participation and competition, offer a distraction-free interface with real-time leaderboards.
Use Socket.io to track student progress, monitor quiz results, and provide real-time leaderboard updates.

2.5.3 Automated Performance Analysis and Evaluation

Grade responses instantly and give feedback right away.Produce comprehensive analytics and performance reports for evaluation.Allow results to be exported in various forms (such as Excel and PDF) for documentation purposes.

2.5.4 Future Improvements, Scalability, and Accessibility

Make sure the design is clear, responsive, and user-friendly, and that it works well on a variety of platforms, including computers, tablets, and smartphones.To improve the user experience, use customized themes and dark mode.Create a system that is scalable so that it can accommodate many users and accommodate upcoming features like deeper analytics, adaptive assessments, and quizzes created by AI.

3. Data and Methods

The Interactive Online Quiz Platform was put together using a systematic and iterative process which included testing, frontend and backend programming, UI/UX design, and real-time interactions. An effective, scalable, and user-friendly platform was assured by this process.

3.1 Development Process

The **Agile technique**, which included iterative cycles for design, development, testing, and feedback, was used to create the project. The following crucial stages were part of the development process:

3.1.1 Analysis and Planning of Requirements

Determined the essential features for both teachers and students.Specified the real-time features, assessment system, quiz structure, and question forms.The tech stack was finalised according to project specifications.

3.1.2 Wireframing and UI/UX Design

The user interface (UI) and wireframes were designed using **Figma**.Prototypes of various screens were made, such as the landing page (role selection: teacher/student), the teacher dashboard (quiz creation, analytics, leaderboard, grade sheet), and the student quiz interface (quiz attempt, leaderboard, results). To guarantee a flawless user experience, responsiveness and accessibility were prioritized.

3.1.3 Development of Frontends

The application is built using **Next.js (v15.2.4)** as a **React-based framework** to take advantage of React 18.3.1 and **TypeScript** for maintaining type safety in the code base. This stack allows implementations of server-side rendering and API routes aside from a scalable frontend development.

Styling and UI: TailwindCSS, a utility-first CSS framework with rapid UI development, styles the application. For accessible and highly customizable UI components, the project integrates Radix UI, utilizing elements such as dialogs, dropdowns, and tabs. Lucide React is used in the project for a modern set of icons while Sonner is used to display the toast notifications in a friendly way.

3.1.4 Backend and Development

The backend logic is powered by Express.js, while MongoDB serves as the main database with an easy configuration via Mongoose across querying and data modeling. For real-time communication features, Socket.IO is integrated to give WebSocket support and a bidirectional means of communication.

The entire authentication is done using Clerk (v6.13.0), a full-fledged auth solution offering multi-factor authentication, social logins, and user/session management. Authentication via tokens is done with JWT (jsonwebtoken), while bcryptjs makes sure of a secure hash before storing the password into the database.

The project utilizes React Hook Form for extremely efficient form handling in feature-rich applications along with Zod for schema-based validation. Additionally, the validation logic is connected seamlessly between Zod and React Hook Form with the help of @hookform/resolvers.

3.1.5 Integration of Real-Time features

The application has real-time features through the **Socket.IO**. By using WebSocket technology, event-based communications generate real-time changes between the client and the server, thus resulting in a much smoother interactive experience**.** Many other features were integrated such as **Live participant tracking**, **Real-time progress updates**, **Instant score updates**, **Live leaderboard, Teacher monitoring dashboard and Quiz time synchronization.**

3.1.6 Testing and Debugging

Every module underwent unit testing.To guarantee a seamless and error-free experience, user testing was also carried out.To improve performance, debugging and browser development tools were used.

3.2 Technologies stack used

Next.js	React framework for production-grade applications with server-side rendering and routing
React	Frontend library for building user interfaces and components
TypeScript	Static typing and enhanced development experience
MongoDB & Mongoose	Database management and object modeling
Clerk	Authentication and user management system
Express.js	Real-time bidirectional communication
Zod	Schema validation and type inference
JWT & bcryptjs	Token-based authentication and password hashing

date-fns	Date manipulation and formatting
Sonner	Toast notifications system
React Hook Form	Form handling and validation
Next Themes	Theme management (dark/light mode)
CORS	Cross-Origin Resource Sharing middleware
dotenv	Environment variable management
Lucide React	Icon library for modern applications
date-fns	Date manipulation and formatting
Embla Carousel	Carousel/slider component

Table 3.2.1: Describes all the tech stacks used and their purpose in the project

4.Results and Discussions

4.1 Results

4.1.1 Technical Achievements

Implemented as a full-stack quiz application successfully, Quisly adopts a web-proficient modern stack. A resistible architecture was constructed on a scalable pattern using Next.js and Express.js. Real-time capabilities were brought to life through the use of Socket.IO, allowing live updates during quizzes with synchronized interaction. Clerk maintained user security and authentication, thus protecting user login and session. Composition of the user interface was done using Radix UI, thus making for a responsive and inclusive design on accessibility.

4.1.2 User Experience

Setting up the experience for users was a joy with the application featuring modern, clean interfaces that would fit in with the current design ethos-a pleasure to use. Leaderboard updates that happen in real-time keep the surroundings alive and foster competition. Responsive design makes it possible for it to work even on the smaller devices. Accessible components would allow a particular group of users to experience that inclusivity on the platform. Overall, the interface arguably would leave to smooth navigation, even more effortless interaction, and reasonable application states.

4.1.3 Performance Optimization was a prime focus during the lifetime of the application. Server-side rendering was implemented to ensure a quicker initial page load and better SEO. Database query optimization through Mongoose contributed to fast data retrieval and updates. Features requiring

real-time communication were powered by Socket.IO, with subsequent updates being made at the lowest possible latency. Optimization in state management and data flow have also contributed towards speeds.

4.2 Discussion

4.2.1 Strengths

Numerous strengths seem to characterize Quisly in architecture, user experience, and even security. There is also a clean separation between the frontend and backend logic while adopting a modular component structure, which will really simplify development and maintenance in the architectural part. TypeScript definitely type-safety development, which would minimize runtime-type errors and MongoDB will make a flexible scalable database schema. On the user experience front, the application has an intuitive quiz interface, real-time feedback loops, and a cohesive design language to make consistency with other screens. Furthermore, it makes the experience seamless across devices. Security really steals the show; Clerk is more than just naked, robust authentication; it also features protected API routes, bcrypt-enabled password handling, and JWT-based token management.

4.2.2 Technical Innovations

Quisly integrates innovative concepts, such as real-time monitoring capabilities for quizzes, while creating a live leaderboard that engages users. The effective proposition ensures proper state management for quizzes such that it has promises never to leave a user with old views. Joining would connect the established onboarding experience with secure access control as opposed to authentication in its various forms. Such features certainly exhibit how modern, real-time, and scalable applications should look for their architecture.

4.2.3 Future Enhancement

Future work on Quisly should include advanced analytics informed by user performance and behaviour. The addition of other types and formats of quizzes would further diversify the platform. Also, any social features or sharing options would increase engagement and reach. The addition of offline mode would increase usability in areas with unreliable access to the Internet.

4.2.4 Lessons Learned

A clear lesson learned during the development of Quisly was the importance of real-time implementation for user engagement and response. Using TypeScript in the project greatly helped with maintaining the quality coding and decreasing errors in complex applications. The component-based structure allows plus and minimizes disruption in scaling and maintaining the product, where new features could be easily installed or modified. Likewise, a human-centered design approach ensured that the platform was user-friendly, usable, and inclusive-the nice design increases user satisfaction.

4.2.5 Educational Impact and Value

Quisly greatly enhances educational value by fostering more interactive quizzes for learning purposes. The platform keeps all learners engaged with timely feedback, being aware of their performance along the way. Tracking the users' progress with embedded tools allows both users and educators to follow and track improvements, thus helping to achieve personalized and adaptive learning.

4.2.6 Technical Impact and Value

Technically, this project serves as a testimony to modern-day full-stack web development practices. From real-time communication to secure authentication and responsive design, it shows how different tools and technologies may be employed into a single solution. Its architecture shall be scalable for varying environmental size and use cases.

4.2.7 Challenges Faced

Various challenges were encountered through the development process, particularly in synchronizing multiple users in real-time and managing state for quiz flow. A smooth, secure authentication flow required care in integrating the Clerk and JWT handling process. Compatibility issues were fixed across the browsers to ensure the seamless running across various platforms and devices.

4.2.8 Target Audience

Quisly is aimed at a wide variety of target audiences, from educational institutions and corporate training programs to self-learners and quiz lovers. It is a feature-rich, real-time solution designed for any situation requiring engaging and scalable quiz delivery.

4.2.9 Competitive Advantage

The platform's real-time features, modern and user-friendly interface, scalable architecture, and robust security infrastructure provide a big advantage over conventional quiz systems. These competitive advantages make Quisly a strong contender in the evaluation tools and edtech market.

5. Conclusion

The Quisly Quiz Platform, which skillfully combines technology and conventional teaching techniques, is a revolutionary approach to contemporary education. It transforms how teachers design, administer, and assess tests while giving students an interesting and dynamic educational experience by offering a complete digital assessment solution. Because of the platform's real-time features, which allow for instant feedback and progress tracking, teachers and students can monitor performance in real time, resulting in more effective teaching methods and improved learning results.Because the platform streamlines the entire assessment process from conception to evaluation, the impact on teachers is especially noteworthy. Instructors may easily create personalized tests with a variety of question formats, track students' progress in real time, and get immediate data on how well their classes are performing. The automated grading system greatly lessens the administrative burden, and this instant knowledge enables prompt interventions and modifications to instructional strategies. Teachers can concentrate more on teaching and less on administrative duties thanks to the platform's user-friendly interface and extensive feature set, which eventually improves the standard of instruction.The platform provides students with an interactive and captivating educational experience that surpasses conventional evaluation techniques. While the progress tracking tools assist students in identifying their areas of strength and growth, the real-time feedback system offers instant reinforcement of learning. The gamification features of the site, such progress indicators and leaderboards, foster an inspiring atmosphere that promotes involvement and ongoing education. Throughout their educational journey, students are kept motivated and involved by this instant feedback loop.The platform's technical architecture supports both small classroom settings and major educational institutions, guaranteeing scalability and reliability. Modern web technologies like Next.js, Socket.io, and MongoDB are integrated to create a strong basis for data management and real-time interactions. The platform is appropriate for both in-person and remote learning

situations because of its responsive design, which guarantees accessibility across a range of devices. All users get a flawless experience because to the combination of this technical prowess and intuitive user interfaces.The Quisly Quiz Platform is ideally positioned to develop alongside new educational technologies and approaches in the future. Its modular design makes it simple to incorporate new features like sophisticated analytics and AI-powered insights. The platform will continue to be at the vanguard of educational technology, adjusting to the evolving demands of both instructors and students thanks to its dedication to ongoing innovation and user feedback. Because of this progressive approach, it serves as both a foundation for learning experiences in the future and a tool for teaching in the present.Beyond specific classrooms, the platform has an impact on the larger educational ecosystem. It assists educational institutions in making data-driven decisions on curriculum development and instructional practices by standardizing assessment procedures and offering thorough analytics. The platform's focus on inclusivity and accessibility guarantees that all students, irrespective of their learning requirements, can take advantage of its features. The Quisly Quiz Platform is a useful tool in the continuous development of contemporary education because of its all-encompassing approach to educational technology.

6. References

- **Next.js –** https://nextjs.org/docs
- **React –** https://reactjs.org/docs
- **TypeScript –** https://www.typescriptlang.org/docs
- **MongoDB –** https://www.mongodb.com/docs
- **Mongoose –** https://mongoosejs.com/docs
- **Clerk –** https://clerk.com/docs
- **Express.js –** https://expressjs.com
- **Socket.IO –** https://socket.io/docs
- **Radix UI –** https://www.radix-ui.com/docs
- **TailwindCSS –** https://tailwindcss.com/docs
- **Zod –** https://zod.dev
- **JWT (jsonwebtoken) –** https://github.com/auth0/node-jsonwebtoken
- **bcryptjs –** https://github.com/dcodeIO/bcrypt.js
- **Next Themes –** https://github.com/pacocoursey/next-themes
- **Cohort 2.0 -** https://100xdevs.com