**A**

**PROJECT SCHOOL REPORT**

**ON**

# Wizkids: Quiz-app

**Submitted By**

**Murtuza mohammad 245322733167**

**Giri sai ponogoti 245322733144**

**Srikaran Boyini 245322733136**

**HariHaran Gonela 245322733146**

**Mounika Sargari 245322733166**

**Abdul khayum 245322733170**

**Under the guidance**

**of**

**B. Kiran Kumar**

**Assistant Professor, CSE(AIML)**



# NEIL GOGTE INSTITUTE OF TECHNOLOGY

Kachavanisingaram Village, Hyderabad, Telangana 500058.

**AUGUST, 2024**

# NEIL GOGTE INSTITUTE OF TECHNOLOGY

A Unit of Keshav Memorial Technical Education (KMTES)

Approved by AICTE, New Delhi & Affiliated to Osmania University, Hyderabad

# CERTIFICATE

*This is to certify that the project work entitled* “Wizkids: Quiz-app**”** *is a bonafide work carried out by* **“Giri Sai, Murtuza Mohammed , Sri karan , HariHaran Gonela,Mounika Sargari, Abdul khayum ”** of II-year IV semester **Bachelor of Engineering** *in* **CSE** *during the academic year* **2023-2024 and** *is a record of bonafide work carried out by them*.

## Project Mentor

B. Kiran Kumar

Assistant Professor

CSE

**ABSTRACT**

The main goal of this project is to develop an interactive quiz application that enables users to test their knowledge in a fun and educational manner. The system, named **WizKids**, allows users to securely log in, select a subject, choose a difficulty level (easy, medium, or hard), and attempt quizzes in a user-friendly environment. After each quiz, users can view their scores and restart the quiz if desired, promoting continuous learning and self-improvement.

The development process begins with the design of a structured and responsive user interface using **React.js**, **HTML**, and **CSS**. Firebase is integrated as the backend service for managing user authentication and storing quiz data such as questions and scores. Firebase Authentication ensures secure login/signup, while the Firebase Realtime Database is used for storing subject-wise, level-based questions and tracking user performance.

React components manage quiz rendering, option selection, and scoring logic on the client side, while Firebase handles data retrieval and storage in real time. This separation ensures fast, efficient performance and scalability.

The interface is designed to be simple, intuitive, and engaging, making it accessible to users of all age groups. The application serves both educational and recreational purposes, transforming learning into a game-like experience through visually rich and interactive elements.

**CONTENTS**

| **S. NO.** | **TITLE** | | **PAGE NO.** |
| --- | --- | --- | --- |
|  | **ABSTRACT**  **TABLE OF CONTENTS** | | 3  4 |
| **1** | **Introduction**  1.1 Introduction  1.2 Limitations | | 6  6 |
| **2** | **Literature Survey** | |  |
|  | 2.1 | Frontend Development Technologies | 7 |
|  | 2.2 | Backend Services and Authentication | 8 |
| **3** | **Proposed Work & Technology Stack** | |  |
|  | 3.1  3.2  3.3 | Technology stack  Proposed work  Architecture | 9  10  12 |
| **4** | **Results & Discussions** | |  |
|  | 4.1 | Result | 14 |
|  | 4.2 | Discussions | 16 |
| **5** | **Conclusion & Future Scope** | | 17 |
| **6** | **References** | | 19 |

**LIST OF FIGURES**

| **Fig. No.** | **Figure Name** | **Page No.** |
| --- | --- | --- |
| 3.3.1 | Architecture Diagram | 12 |
| 4.1.1 | UI for user inputs | 14 |
| 4.1.2 | UI of Wizkids | 15 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**CHAPTER-1**

**INTRODUCTION**

* 1. **Introduction**

WizKids is an interactive and engaging quiz application designed to blend education and entertainment into a seamless gaming experience. Users of all ages can easily create their accounts and log in to begin their quiz journey. The app provides multiple subjects for users to choose from, ensuring a personalized and diverse learning experience. Each subject offers quizzes categorized into three difficulty levels—easy, medium, and hard—to accommodate varying skill levels and encourage progressive learning.

Participants can enjoy instant feedback, viewing their scores at the end of each quiz session, allowing them to track their performance and progress over time. WizKids also encourages continuous learning through its user-friendly interface that enables users to restart quizzes and attempt challenges multiple times. By combining the excitement of gaming with educational content, WizKids provides a fun, competitive, and rewarding environment that motivates users to enhance their knowledge and critical thinking skills.

* 1. **LIMITATIONS OF OUR PROJECT:**
* User inputs are mandatory for quiz initiation.
* Currently limited to predefined subjects and questions.
* Limited number of subjects available.
* Only three predefined difficulty levels (easy, medium, hard).

**CHAPTER-2**

**LITERATURE SURVEY**

#### ****1.**** Frontend Development Technologies:

**React.js:**

* **Overview:** React.js is a popular JavaScript library for creating interactive, responsive, and component-based user interfaces. It efficiently manages dynamic content and application state, significantly enhancing user experiences (React Documentation, 2024).
* **Usage in the Project:** React.js is utilized for building modular and reusable components such as login, signup forms, quiz interfaces, and score displays, enabling dynamic content updates and efficient rendering.

**HTML (HyperText Markup Language):**

* **Purpose:** HTML provides the structural foundation for web applications, defining elements and content layout, essential for accessibility and browser compatibility (MDN Web Docs, 2024).
* **Usage in the Project:** HTML structures the WizKids app content, including forms, buttons, and quiz components, facilitating clear navigation and content organization.

**CSS (Cascading Style Sheets):**

* **Purpose:** CSS enhances web application aesthetics by controlling layout, design elements, responsive behavior, and visual styling, improving user engagement and interface intuitiveness.
* **Usage in the Project:** CSS styles the WizKids application, ensuring a visually appealing, responsive, and user-friendly interface across various devices and screen sizes.

**JavaScript (JS):**

* **Purpose:** JavaScript is an essential programming language for creating interactive and dynamic web applications, enabling client-side scripting for improved functionality and real-time content manipulation (Flanagan, 2020).
* **Usage in the Project:** JavaScript powers the interactive features of WizKids, such as quiz logic, state management, user interactions, real-time feedback, and score calculation.

#### Backend Services and Authentication:

**Firebase Authentication:**

* **Overview:** Firebase Authentication offers secure, easy-to-integrate solutions for user authentication and session management, supporting multiple login methods (Google Firebase Docs, 2024).
* **Usage in the Project:** Firebase Authentication manages secure user registration, login, and session persistence, ensuring robust user management and secure data access.

**Firebase Realtime Database:**

* **Overview:** Firebase Realtime Database is a cloud-hosted NoSQL database allowing real-time synchronization of data across clients, facilitating seamless user experiences with minimal backend configuration (Google Firebase Docs, 2024).
* **Usage in the Project:** Firebase Realtime Database stores quiz questions, user progress, and scores, providing instant updates and synchronization between frontend components and backend storage.

### Proposed Work, Architecture, Technology Stack & Implementation Details

### 3.1 TECHNOLOGY STACK:

The technologies which we have used in this project are detailed given below:

**1. Programming Language: JavaScript**• JavaScript is the primary programming language used due to its versatility in creating interactive, dynamic, and responsive web applications. It provides seamless client-side interactivity and integration with frontend frameworks.

**2. Frontend Library: React.js**  
• React.js is a powerful JavaScript library used to build modular, reusable, and efficient user interface components. It simplifies state management and ensures responsive user experiences, making it ideal for interactive quiz applications.

**3. Markup Language: HTML**  
• HTML structures the web content effectively, providing semantic elements crucial for layout, accessibility, and compatibility across various browsers and devices.

**4. Styling: CSS**  
• CSS enhances the application's visual presentation by controlling layout, colors, typography, responsiveness, and other stylistic elements, significantly improving user interface appeal and usability.

**5. Backend and Authentication: Firebase**  
• Firebase Authentication: Offers secure user login, registration, and session management, ensuring authenticated and secure access to the quiz content and user data.

**3.2 PROPOSED WORK:**

1. **User Authentication and Session Management**

* Integrate Firebase Authentication to securely manage user signup, login, email verification, and session persistence.
* Implement user-friendly login/signup forms using React.js for enhanced user interaction.

2. **Interactive Quiz Selection**

* Design subject selection and difficulty-level selection components that dynamically present available quizzes.
* Allow users to clearly view and select their desired quiz category and difficulty seamlessly.

3. **Dynamic Quiz Execution**

* Develop quiz logic using React.js to dynamically load and render questions based on chosen subject and difficulty.
* Implement timer functionalities to manage quiz duration, enhancing competitiveness and engagement.

4. **Real-Time Scoring and Feedback**

* Implement real-time scoring mechanisms to instantly calculate and display quiz results upon completion.
* Provide detailed feedback after each quiz attempt, including correct answers, user-selected responses, and performance summaries.

5. **Data Storage and Synchronization**

* Utilize Firebase Realtime Database to store and manage quiz questions, user scores, performance history, and quiz metadata.
* Enable real-time synchronization for seamless updates and consistent user experiences across different sessions and devices.

6. **Responsive and Intuitive User Interface**

* Create responsive React components and interfaces optimized for various devices (desktop, tablet, mobile).
* Utilize HTML/CSS for structured layout, appealing aesthetics, responsive behaviour, and intuitive navigation.

7. **Error Handling and User Feedback**

* Implement comprehensive error handling across frontend and backend interactions to gracefully manage unexpected scenarios.
* Provide meaningful, clear, and actionable feedback messages to enhance user experience and app reliability.

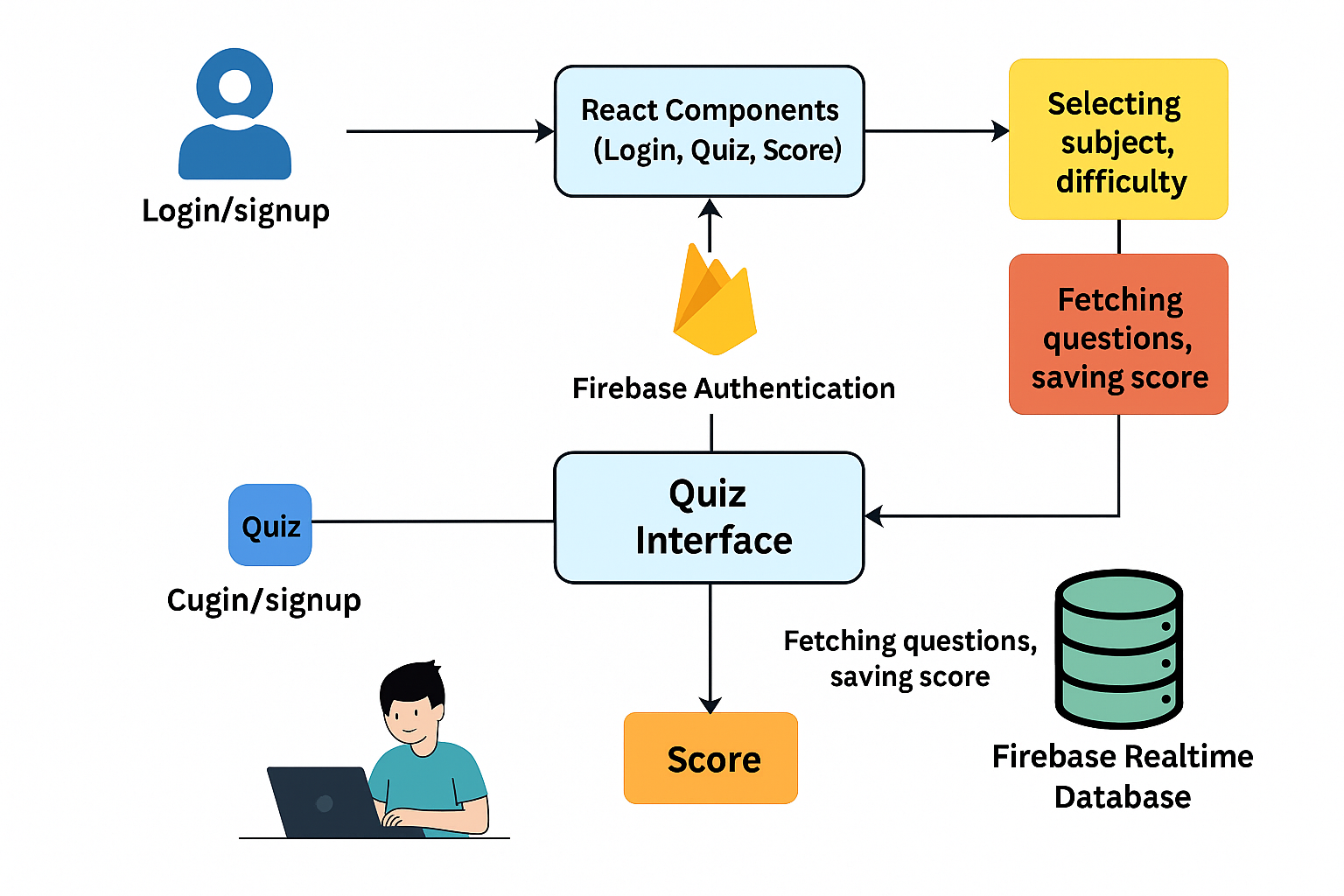
8. **Integration and Testing**

* Conduct functional and integration testing to ensure robust interaction between React frontend components and Firebase backend services.
* Verify all app features including authentication, quiz execution, scoring, and data management operate correctly and consistently.

9. **Deployment and Maintenance**

* Deploy the application using reliable cloud hosting platforms, ensuring scalability, performance, and secure access.

**3.3 Architecture:**

****

**3.3.1 Architecture Diagram**

### 1. Presentation Layer (Client-side):

### This layer is responsible for displaying the interface to the user and collecting inputs like login credentials, subject selection, and quiz answers. Built using React.js, it handles dynamic rendering and smooth user interaction. Technologies: React.js, HTML, CSS Components:

### Login/Signup UI: For user authentication

### Quiz UI: Displays questions, options, and captures responses

### Score Display: Shows quiz results and allows restart

### 2. Application Layer (Business Logic):

### This layer manages application logic, state transitions, user authentication handling, and interaction flow between components. Technologies: JavaScript (React.js), Firebase Authentication API Components:

### React Components: Functional components like Login.jsx, SignUp.jsx, QuestionCard.jsx, and Score.jsx

### Authentication Handler: Connects with Firebase to verify users and maintain secure sessions

### Routing Logic: Determines which screen (login, quiz, score) to display based on app state

### 3. Data Layer (Backend - Firebase):

### Responsible for persistent data storage, including user scores and quiz questions, and real-time data syncing with the frontend. Technologies: Firebase Realtime Database, Firebase Authentication Components:

### Authentication Service: Manages secure login/signup through email and password

### Realtime Database:

### Stores quiz questions organized by subject and difficulty level

### Stores user scores and quiz history

### Data Sync: Automatically updates data and UI when changes occur in the database

### Workflow Summary (Based on Diagram):

### Login/Signup Process: Users enter credentials → Firebase Authentication verifies and grants access.

### Subject & Difficulty Selection: React component renders options → User selection is stored in the app state.

### Quiz Execution: Selected subject and level are used to fetch relevant questions from Firebase → Questions are rendered using React → User answers are submitted and scored.

### Score Management: Final scores are calculated and saved to Firebase → React Score component displays the result → Option to restart quiz is provided.

### Firebase Integration: Ensures secure authentication and consistent real-time updates between frontend and backend.

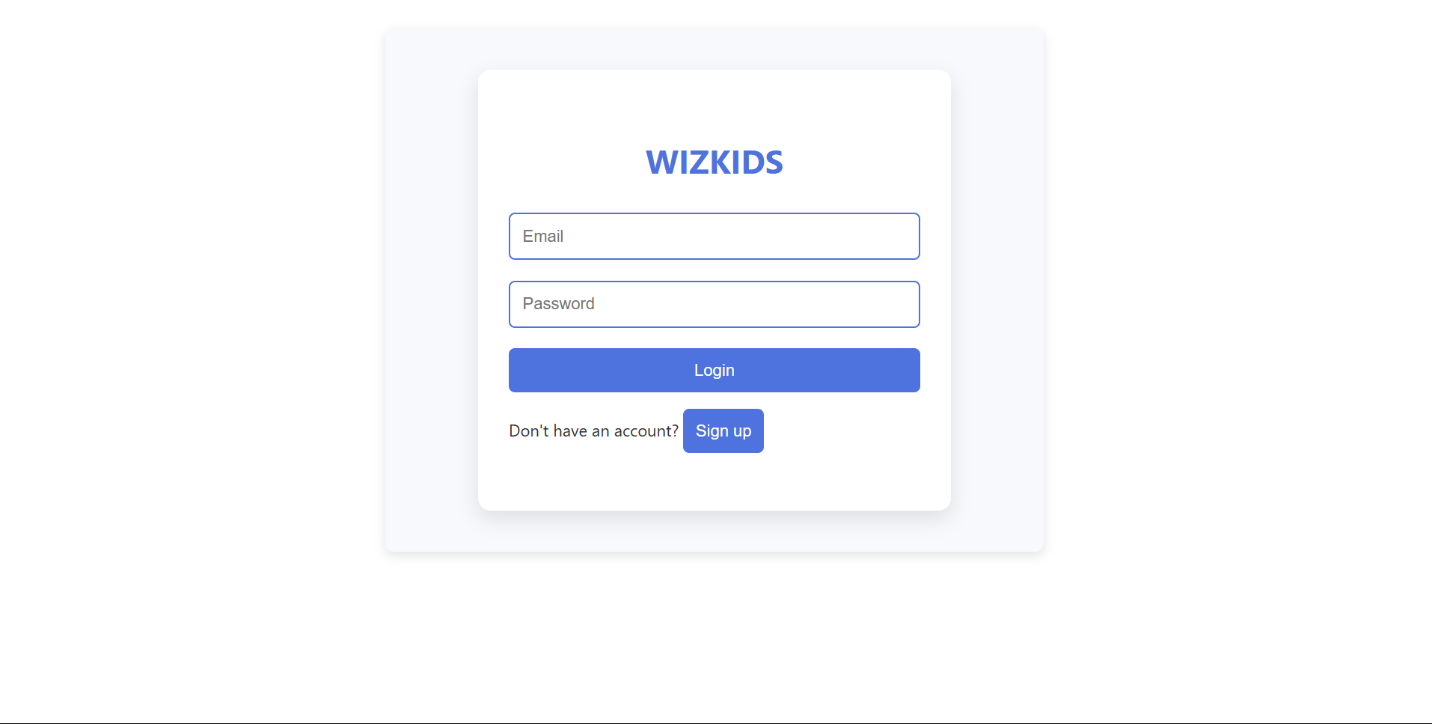
**4.RESULTS AND DISCUSSION**

**4.1 RESULT**

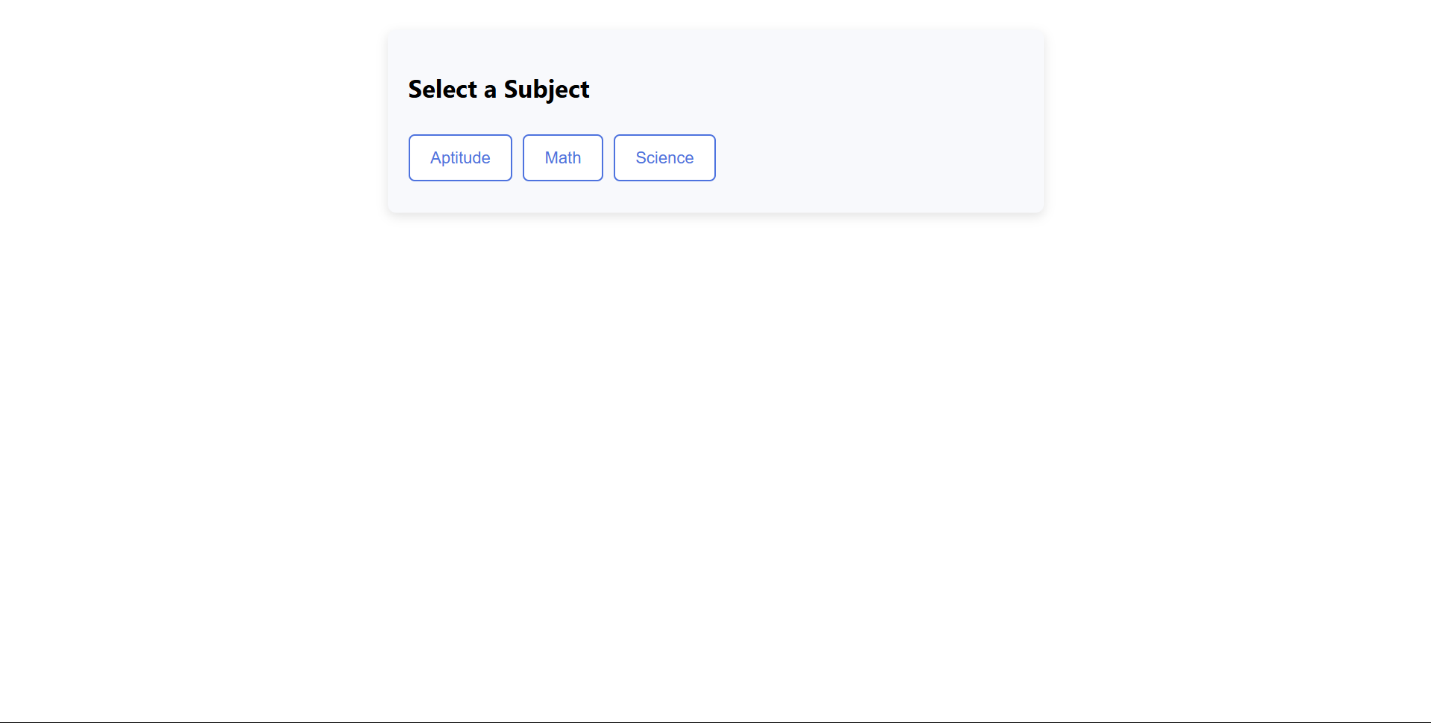
**Quiz Handling:**

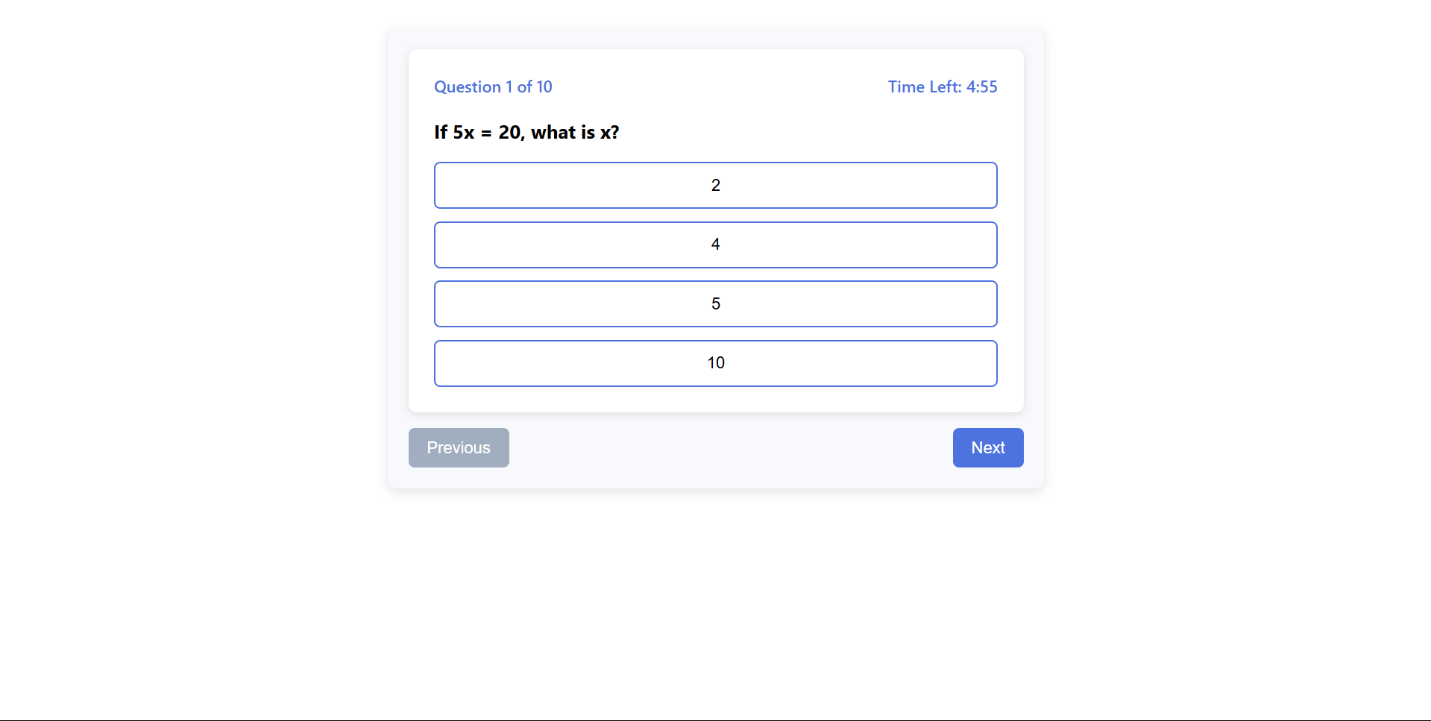
1. Users can register and log in securely using Firebase Authentication.
2. Subjects and difficulty levels (easy, medium, hard) can be selected interactively through the UI.
3. The application accurately calculates and displays scores at the end of each quiz.
4. Users can view their performance summary and choose to restart the quiz.

**Front-end Integration:** A user-friendly React-based front-end interface for interacting with the back-end API s.



**4.1.1 UI for user inputs**





**4.1.2 UI of Wizkids**

**4.2** **Discussions**

**Integration and Performance:**

The integration of React components with Firebase services proved to be effective in managing user authentication, quiz logic, and real-time data storage. Firebase Authentication ensured secure login/signup, while Firebase Realtime Database allowed seamless fetching and updating of quiz questions and scores. The overall architecture provided a smooth and responsive user experience with minimal latency during quiz interaction.

**Challenges Faced:**

* Limited Question Pool: Initially, the availability of questions was restricted to a small number of subjects and difficulties, limiting user engagement.
* Data Structuring: Organizing quiz questions in a scalable, difficulty-based format within Firebase required careful structuring and testing.
* Score Accuracy: Implementing accurate and real-time scoring logic was challenging, especially while maintaining component reusability.
* UI Responsiveness: Ensuring the app was responsive across various screen sizes and devices demanded thorough CSS customization and testing.

**5.CONCLUSION AND FUTURE SCOPE**

**5.1 Conclusion**

The WizKids Quiz App successfully demonstrated the development and integration of modern web technologies such as React.js, Firebase Authentication, and Firebase Realtime Database to build an interactive, educational, and user-friendly quiz platform. By implementing real-time data synchronization, secure login systems, and dynamic question rendering, the app provides users with a smooth and engaging quiz experience.

The project achieved its core objectives—allowing users to log in, select subjects and difficulty levels, attempt quizzes, view scores instantly, and restart quizzes with ease. Users responded positively to the app’s intuitive design, quick feedback, and simple navigation, making it both a fun and educational tool.

Despite challenges such as managing quiz structure, ensuring accurate scoring, and optimizing frontend responsiveness, the team successfully delivered a stable and scalable system. The use of Firebase minimized backend complexity while ensuring data security and performance.

The WizKids platform lays a strong foundation for continued development, and with future enhancements, it has the potential to become a widely used educational game application for students of all ages.

**5.2 Future Scope**

The WizKids Quiz App establishes a solid base for further enhancements and features that can significantly enrich user experience and broaden its educational impact. Key areas for future development include:

* **Expanded Subject Categories:** Introducing more subjects and topics will cater to a broader user base and enhance learning diversity.
* **Custom Quiz Creation:** Allowing users or educators to create and upload their own quizzes for classroom or personalized use.
* **User Progress Dashboard:** Implementing a personal dashboard where users can track their quiz history, scores, and performance trends.
* **Leaderboards and Achievements:** Adding gamification elements such as leaderboards, badges, and rewards to boost motivation and engagement.
* **Multiplayer and Timed Challenges:** Developing a multiplayer mode for real-time quiz battles and timed quizzes for competitive gameplay.
* **AI-Powered Question Generation:** Integrating AI to dynamically generate questions based on user level and performance patterns.
* **Offline Access:** Enabling quiz access in offline mode, with data syncing once the user reconnects to the internet.
* **Feedback and Suggestion System:** Including a user feedback feature to collect suggestions and improve quiz quality and app functionality.

In conclusion, WizKids has the potential to evolve into a comprehensive and adaptive learning platform. With planned enhancements and ongoing improvements, it can become a powerful educational tool that balances fun, challenge, and learning for users of all ages.

**6. REFERENCES**

**1. React.js Framework:**  
• React Documentation. (n.d.). Retrieved from <https://reactjs.org/>  
• Abramov, D., & Clark, A. (2015). The Road to Learn React. Leanpub.

**2. HTML and CSS:**  
• MDN Web Docs. (n.d.). HTML Reference. Retrieved from <https://developer.mozilla.org/en-US/docs/Web/HTML>  
• MDN Web Docs. (n.d.). CSS Reference. Retrieved from <https://developer.mozilla.org/en-US/docs/Web/CSS>

**3. JavaScript:**  
• Flanagan, D. (2020). *JavaScript: The Definitive Guide* (7th ed.). O'Reilly Media. ISBN: 978-1491952023  
• JavaScript Documentation. (n.d.). Retrieved from <https://developer.mozilla.org/en-US/docs/Web/JavaScript>

**4. Firebase Platform:**  
• Firebase Authentication Documentation. (n.d.). Retrieved from https://firebase.google.com/docs/auth  
• Firebase Realtime Database Documentation. (n.d.). Retrieved from https://firebase.google.com/docs/database