**Report on Quiz APP (Frontend Project)**

# Project Overview

The Engineering Quiz Application is designed to provide users with an engaging platform to test their knowledge in engineering-related topics through quizzes. The application utilizes modern web

technologies including React, TypeScript, and Vite for a responsive and dynamic user experience.

# Technologies Used

* **Frontend Framework:** React
* **TypeScript:** For type safety and improved development experience.
* **Build Tool:** Vite for fast development and build processes.
* **CSS Framework:** Tailwind CSS for styling and layout.
* **Linting and Formatting:** ESLint and Prettier for code quality and consistency.
* **Version Control:** Git for source code management.

# Key Features

1. **User Authentication:** Users can sign up and log in to track their quiz progress.
2. **Quiz Creation:** Users can create custom quizzes with multiple-choice questions.
3. **Real-time Feedback:** Immediate feedback on answers to enhance learning.
4. **Leaderboard:** Displays top scorers to encourage competition.
5. **Responsive Design:** The application is fully responsive, providing a seamless experience across devices.

# Development Environment

The application is developed in a Node.js environment with specific version requirements:

* **Node.js Version:** >= 14.0.0

# Development Dependencies:

* + ESLint for linting JavaScript and TypeScript code.
  + TypeScript for static type checking.
  + Tailwind CSS for utility-first styling.
  + Vite for fast development server and build tool.

1. Access the application in your browser at http://localhost:5174/.

# Project Structure

The project follows a modular structure for better maintainability:

* **src/**: Contains all the source code.
  + **components/**: Reusable React components.
  + **pages/**: Page components for routing.
  + **styles/**: Global styles and Tailwind CSS configuration.
  + **utils/**: Utility functions and constants.

# Additional Files and Their Purpose

1. **Configuration Files:**
   * **eslint.config.js**: This file configures ESLint for linting JavaScript and TypeScript code, ensuring code quality and adherence to coding standards.
   * **config.json**: Contains the project template information, which can be useful for understanding the initial setup of the project.

# HTML File:

* + **index.html**: The main HTML file that serves as the entry point for the application. It includes the necessary meta tags and links to the main JavaScript file.

# Binary Files:

* + Several binary files (e.g., **semver**, **rollup**, **vite**, etc.) are included, which are essential for various build and runtime tasks. These files ensure that the application can be built and run correctly across different environments.

# Scripts:

* + **\*.cmd and \*.ps1 files**: These scripts are used to execute various commands in

Windows environments. They help in running the application and its components seamlessly.

# Package Lock File:

* + **.package-lock.json**: This file locks the versions of the dependencies used in the project, ensuring that the same versions are installed in future setups, which is crucial for maintaining consistency across different environments.

# Git Ignore File:

* + **.gitignore**: This file specifies which files and directories should be ignored by Git, preventing unnecessary files from being tracked in version control.

# Summary of Additional Mentions

* The project includes configuration files for ESLint and other tools, which are crucial for maintaining code quality.
* The presence of various scripts and binaries indicates a well-structured setup for running and building the application.
* The **.gitignore** file ensures that irrelevant files do not clutter the version control history.
* The package lock file guarantees consistent dependency versions across installations.

# Challenges and Solutions

* **Challenge:** Ensuring cross-browser compatibility for the quiz interface.
  + **Solution:** Utilized CSS resets and modern CSS practices to ensure consistent rendering across different browsers.
* **Challenge:** Managing state effectively in a complex application.
  + **Solution:** Leveraged React's Context API and hooks for state management, making it easier to share state across components.

# Future Enhancements

1. **Mobile App Development:** Create a mobile version of the quiz application using React Native.
2. **Question Bank Expansion:** Allow users to contribute questions to a shared question bank.
3. **Analytics Dashboard:** Provide users with insights into their performance over time.

# Conclusion

The Engineering Quiz Application is a robust platform for learning and testing knowledge in

engineering. With its modern tech stack and user-friendly interface, it aims to provide an enjoyable learning experience for users. Continuous improvements and feature enhancements will ensure its relevance and effectiveness in the educational domain.

## **Quiz APP- Frontend Project Progress Report**

### **Period: 23rd October 2024 – 15th November 2024**

### **23rd October 2024**

* **Initiated the project.**
* Finalized the basic structure and layout of the quiz application using HTML.
* Created a prototype for the quiz interface, including the following components:
  + Question display area.
  + Options as radio buttons.
  + Navigation buttons (Next, Previous, Submit).

### **25th October 2024**

* Enhanced UI using **CSS**:
  + Styled the quiz container with borders, padding, and rounded corners.
  + Added hover effects to buttons for better interactivity.
* Incorporated responsive design principles for mobile compatibility.

### **28th October 2024**

* Added JavaScript functionality:
  + Implemented **event listeners** for navigation buttons.
  + Created a basic function to calculate and display scores on quiz submission.
  + Ensured smooth question transitions with validation for unanswered questions.

### **1st November 2024**

* Developed the **timer feature**:
  + Added a countdown timer for quiz completion.
  + Displayed remaining time dynamically on the screen.
  + Added alerts for when time runs out.

### **4th November 2024**

* Enhanced user experience by:
  + Adding animations for question transitions.
  + Highlighting the selected answer option visually.
  + Displaying immediate feedback on correct or incorrect answers after quiz submission.

### **8th November 2024**

* Integrated **localStorage** to save user progress:
  + Allowed users to resume the quiz from where they left off.
  + Stored selected answers temporarily until submission.

### **11th November 2024**

* Added advanced features:
  + Randomized question order to prevent pattern recognition.
  + Enabled skip functionality for users to revisit unanswered questions later.

### **13th November 2024**

* Conducted testing:
  + Tested the application across multiple browsers (Chrome, Firefox, Edge).
  + Resolved minor bugs related to timer synchronization and option selection.
  + Improved performance by optimizing JavaScript code.

### **15th November 2024**

* Finalized and deployed the frontend project.
* Documented all features, functionalities, and limitations.
* Ready for potential backend integration and further enhancements.