1. Introduction

The goal of this project is to develop an online quiz platform where users can take tests, create tests, view their past attempts, and receive notifications about new quizzes. The application will be built using Laravel for the back-end and Vue.js for the front-end.

2. Project Objectives

- **Create a web application for managing and taking online quizzes.**
- **Allow users to create and manage their own quizzes.**
- **Allow administrators to create and manage quizzes, control users, and manage categories.**
- **Enable superadmins to do all of the above, as well as control user roles.**
- **Enable users to take quizzes and view their past attempts.**
- **Implement a notification system to inform users about new quizzes.**
- **Ensure security and role management for different types of users (superadmin, admin, user).**

3. Features

3.1. User Management

- **Registration and Login: Allow users to create an account and log in.**
- **Role-Based Access: Define roles for superadmin, admin, and regular users.**
  - **Superadmin: Full access to all features, including user role management.**
  - **Admin: Can create and manage quizzes, control users, and create new categories.**
  - **User: Can create quizzes, take quizzes, view past attempts, and receive notifications.**

3.2. Quiz Management

- **Create Quiz: Users, admins, and superadmins can create new quizzes with multiple questions.**
- **Edit Quiz: Users can edit their own quizzes; admins and superadmins can edit all quizzes.**
- **Delete Quiz: Users can delete their own quizzes; admins and superadmins can delete all quizzes.**
- **View Quiz List: Users can view a list of available quizzes.**

3.3. Question Management

- **Add Questions: Users can add questions to their quizzes; admins and superadmins can add questions to any quiz.**
- **Edit Questions: Users can edit their own questions; admins and superadmins can edit any question.**
- **Delete Questions: Users can delete their own questions; admins and superadmins can delete any question.**

3.4. Quiz Attempt

- **Take Quiz: Users can take quizzes.**

- **Submit Answers: Users can submit their answers for evaluation.**
- **View Results: Users can view their quiz results immediately after submission.**

3.5. Past Quiz Attempts

- **View Past Attempts: Users can view their past quiz attempts, including scores and dates.**

3.6. Notifications

- **Send Notifications: Admins and superadmins can send notifications to users about new quizzes.**
- **View Notifications: Users can view notifications in their dashboard.**
- **Mark as Read: Users can mark notifications as read.**

3.7. Category Management

- **Create Category: Admins and superadmins can create new quiz categories.**
- **Edit Category: Admins and superadmins can edit existing categories.**
- **Delete Category: Admins and superadmins can delete categories.**

4. Technical Specifications

- **Back-End: Laravel**
- **Front-End: Vue.js**
- **Database: MySQL**
- **Authentication: JWT (JSON Web Tokens)**
- **API Communication: Axios (for Vue.js to Laravel API calls)**
- **Version Control: Git and GitHub**

5. UML Diagrams

- **Use Case Diagram: To show different user interactions with the system.**
- **Class Diagram: To illustrate the structure of the database and relationships between classes.**

6. Project Phases

1. **Idea Selection and Requirement Gathering: Identify and document the project requirements.**
2. **Project Planning and Jira Setup: Plan project phases and configure Jira for task management.**
3. **Development with Laravel and Vue.js:**
    - **Set up the development environment.**
    - **Develop back-end with Laravel.**
    - **Develop front-end with Vue.js.**
4. **Security Implementation: Secure user data and application access.**
5. **Unit and End-to-End Testing: Ensure the application works as expected.**
6. **Continuous Integration: Automate testing and deployment with GitHub Actions.**
7. **Code Quality with ESLint: Maintain high code quality.**

8. **Dockerization: Use Docker for containerizing the application.**
9. **Documentation and Reports: Document the project details and create reports.**
10. **Deployment: Deploy the application on a platform like AWS or Heroku.**

7. Evaluation Criteria

- **Code Quality: Clarity, performance, and informative comments.**
- **Functional Performance: Reliability and adherence to the required features.**
- **User Experience: Accessibility and user engagement.**
- **Version Control: Organized and consistent versioning practices.**
- **Documentation: Clear and precise documentation as specified in the deliverables.**

8. Deliverables

- **GitHub Repository: Complete source code connected to Jira project.**
- **Project Documentation: PDF, PPT, or Canva file detailing the project.**
- **Docker Images: Backend and frontend images on DockerHub.**