COLLEGE CODE : 9623

COLLEGE NAME : Amrita College of Engineering And Technology

DEPARTMENT : Computer Science and Engineering

STUDENT NM-ID: E4405866C4E8C2C81634C314A03DF444

ROLL NO : 23CS028

DATE : 12-09-2025

Completed the project named as

Phase 1 Problem Understanding and Requirements

PROJECT NAME: INTERACTIVE QUIZ APP

SUBMITTED BY,

NAME : Brylin Shijo.S

MOBILE NO: 9600709838

Interactive Quiz App

Phase 1 – Problem Understanding & Requirements

1. Problem Statement

In the current digital age, assessing knowledge through traditional methods is often cumbersome, time-consuming, and lacks interactivity. The Interactive Quiz App aims to bridge this gap by providing a user-friendly platform where users can take quizzes across various subjects and difficulty levels. The system will support

instant feedback, progress tracking, and administrative controls to manage the quiz content dynamically. The primary problem this application addresses is the need for a scalable, easy-to-use quiz system that enhances learning through interactive engagement.

2. Users & Stakeholders

- Primary Users:
- Quiz Takers: These include students, professionals, and general knowledge enthusiasts who want to test their skills or prepare for exams through interactive quizzes.
- Admins: Responsible for managing the quiz content, monitoring user activities, and generating performance reports.
- Stakeholders:
- Educational Institutions: Schools,

universities, and coaching centers that require an efficient system to conduct periodic quizzes and assessments.

- Quiz Content Creators: Individuals or teams responsible for developing and curating high-quality quiz questions across different domains.
- End Users (Quiz Takers): Individuals seeking self-assessment tools for learning enhancement.
- Secondary Stakeholders:
- System developers and designers ensuring seamless integration of frontend and backend components.
- Database administrators managing data consistency and security.

3. User Stories

- As a user, I want to register an account so I can securely attempt quizzes and track my progress.

- As a user, I want to log in with my credentials so I can securely access my quizzes and results.
- As a user, I want to browse and filter quizzes by category (e.g., science, history, mathematics) and difficulty level (easy, medium, hard).
- As a user, I want to attempt quizzes with multiple-choice questions, receive immediate feedback, and view my final score.
- As a user, I want to view my historical performance data to track my improvement over time.
- As an admin, I want to create, update, and delete quizzes and questions to keep the content fresh and relevant.
- As an admin, I want to manage users, view their quiz participation, and analyze performance reports.
- As an admin, I want to manage

categories and question types for efficient organization of quiz content.

4. MVP Features

- Authentication:
- User registration and login system with role-based access control (User/ Admin).
- Quiz Management:
- Display list of available quizzes with categories and difficulty levels.
- Interface for users to attempt quizzes and receive scores instantly.
- Admin dashboard to manage quizzes and questions.
- User Profile:
- Track quiz history and performance metrics for each user.
- Admin Features:
- CRUD operations for quizzes and questions.

- User management functionalities.
- Basic analytics dashboard for performance tracking.

5. Wireframes / API Endpoint List

Wireframes:

1. Login/Registration Page: Simple forms for user authentication.
2. Home Page: Displays list of quizzes available with categories and difficulty.
3. Quiz Attempt Page: Displays questions one by one with multiple-choice options.
4. Result Page: Shows score and performance summary after quiz attempt.
5. Admin Dashboard: Provides options to create, update, or delete quizzes/ questions and view user data.

API Endpoint List:

- POST /api/auth/register: Register a new user.
- POST /api/auth/login: Login and return a JWT token.
- GET /api/quizzes: Retrieve a list of available quizzes.
- GET /api/quizzes/{id}: Retrieve details of a specific quiz.
- POST /api/quizzes/{id}/submit: Submit answers for a quiz and return the result.
- GET /api/users/{id}/performance: Retrieve user performance history.
- POST /api/admin/quizzes: Admin creates a new quiz.
- PUT /api/admin/quizzes/{id}: Admin updates a quiz.
- DELETE /api/admin/quizzes/{id}: Admin deletes a quiz.
- POST /api/admin/questions: Admin creates a new quiz question.

- PUT /api/admin/questions/{id}: Admin updates a quiz question.
- DELETE /api/admin/questions/{id}: Admin deletes a quiz question.

6. Acceptance Criteria

- The app must support user registration and login with secure password storage.
- Users should be able to browse quizzes filtered by category and difficulty.
- Users should attempt quizzes with multiple-choice questions and receive real-time score feedback.
- Admins must have access to a dashboard that allows them to create, update, and delete quizzes and questions.
- The app must store quiz attempts and scores for users to review their performance history.
- API responses must follow a

consistent structure (e.g., success flag, data payload, error messages).

- The database schema must ensure data consistency and enforce relational integrity (e.g., quizzes linked to categories).
- The system must prevent unauthorized access to admin endpoints.
- Data should be securely stored and protected against common vulnerabilities (e.g., SQL injection).