Author

Suyog Manoj Parkhi
22f3000976
22f3000976@ds.study.iitm.ac.in

I am a dedicated computer science student with a passion for web development and creating user-friendly applications. My interests include building interactive web applications and exploring new technologies.

Description

This project involves creating a Quiz Master application with a responsive interface. The system allows users to create, manage, and take quizzes, with features for tracking performance, timing responses, and displaying results.

Technologies used

- Flask: Python web framework for building the backend API and handling requests
- SQLAlchemy: ORM for database interactions and model definitions
- Flask-RESTful: Extension for creating RESTful APIs in Flask
- HTML/CSS/JavaScript: Frontend implementation with responsive design
- Bootstrap: CSS framework for consistent styling and responsive components
- Chart.js: For visualizing quiz performance and statistics on the dashboard
- SQLite: Lightweight database for development (can be migrated to PostgreSQL for production)

These technologies were selected to create a robust, scalable application with clear separation between frontend and backend, while maintaining ease of development and deployment.

DB Schema Design

This schema supports one-to-many relationships between quizzes and questions, questions and options, users and quizzes/attempts, etc. The design enables efficient querying of quiz results, user performance, and quiz statistics.

API Design

The application provides a RESTful API for all core functionality:

- User APIs: Registration, authentication, profile management
- Quiz APIs: Create, read, update, delete quizzes and questions
- Attempt APIs: Start quiz, submit answers, retrieve results
- Dashboard APIs: Get performance statistics and recent activity

The API follows REST principles with proper HTTP methods (GET, POST, PUT, DELETE) and status codes. Authentication is handled via JWT tokens, with role-based access control for secured endpoints.

Architecture and Features

The project follows the MVC architecture pattern:

- Models: Located in /models directory, defining database schemas
- Views: HTML templates in /templates, with static assets in /static
- Controllers: API endpoints in /routes, handling business logic

Core Features:

- User registration and authentication system
- Quiz creation with multiple question types
- Quiz taking with timer functionality
- Immediate feedback on quiz completion
- Dashboard with performance statistics

The dashboard provides rich visualizations of user performance, with filters for time periods and quiz categories. The responsive design ensures a seamless experience across all devices.

Video

Video link here