QUIZ - MASTER - V2

Student Details

Name: Hitesh
Roll Number: 22f2001256
I am a student of IIT Madras BS Degree in Data Science & Applications. I am currently in the Diploma level and pursuing this degree as a standalone.

Description

The aim of this project is to develop a **Quiz Management System** that allows students to participate in quizzes, track their scores, and receive performance reports. The system should allow **admin** to create quizzes, manage users, and export user performance data asynchronously. Additionally, a timer must be included in the quiz interface, and correct answers should be highlighted. A user/student is able to give the quiz that is available within the time range of today's date and can also see the performance in each quiz.

Approach to the Problem Statement

1. **User Authentication:** Implemented user roles (admin, student) using Flask-Security.
2. **Quiz Management:** Created models for quizzes, questions, scores, and chapters.
3. **Frontend Development:** Used Vue.js for an interactive quiz-taking experience.
4. **Timer & Score Tracking:** Implemented a countdown timer for each quiz attempt.
5. **Asynchronous Processing:** Used Celery and Redis to send email reports.

Technologies Used

- Used Flask for backend Api
- Used Redis & Celery for backend jobs
- For frontend Vue.js, Html, CSS and JS was used
- SQLAlchemy was used for Database generation
- Flask_security for Authentication of the User

DB Schema Design

1. **Users Table** - Stores user information (Admin/Student)
2. **Roles Table** - Defines roles (admin/student)
3. **Subjects Table** - Stores subjects like Mathematics, MAD-I
4. **Chapters Table** - Chapters linked to subjects
5. **Quiz Table** - Stores quiz details linked to chapters
6. **Questions Table** - Stores quiz questions
7. **Scores Table** - Tracks quiz attempts with timestamps

This database design is chosen very strategically. No extra columns were taken that were of no use or would complicate the project.

API Design

The API was designed for RBAC and token authentication. A few API endpoints are given below

Authentication

- POST /login→ User authentication
- `POST /register` → Register a new user

User Dashboard

- `GET /user-dashboard` → Fetch upcoming quizzes for students

Quiz Management

- `GET /get-quiz-questions/<quiz_id>` → Fetch questions for a quiz
- `POST /store-score` → Store user score after quiz completion

Admin Panel

- `POST /create-quiz` → Admin creates a new quiz
- `GET /export-performance` → Admin exports user performance data

Architecture and Features

The app is in the main folder named as main.py. In the application directory, all the python code for the app such as Api.py for api, tasks.py for backend tasks, models.py that define the database for the app etc. are added.

The admin has all the CRUD functionalities for Subject, Chapter, Quiz as well as the Questions. Further the app is loaded with features like instant feedback of answers whether it is correct or not.

Drive Link of the Presentation Video

Quiz-Master-V2