Quiz Master - V1

Project Report

Name: Ritesh V

Student ID: 22f3002177

Email: 22f3002177@ds.study.iitm.ac.in

Course: Modern Application Development I

Project Title: Quiz Master - V1

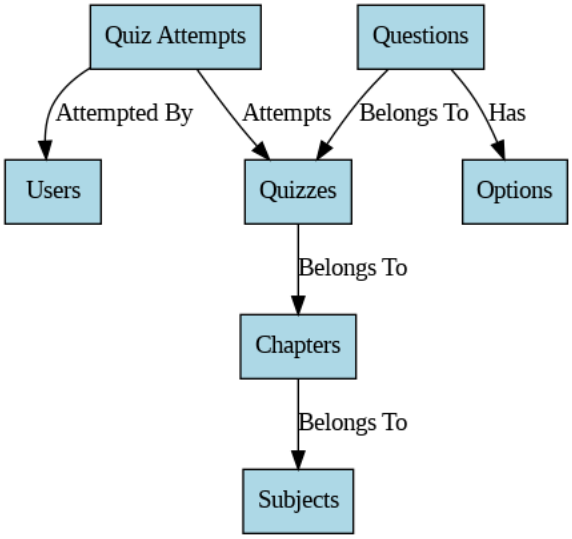
Problem Statement: The project aims to develop an interactive, multi-user exam preparation platform that allows administrators to create quizzes and users to attempt them. The application must support role-based access, quiz management, result tracking, and basic analytics.

Approach: I started by designing the home page and structuring the frontend, ensuring the admin's interface was clear and functional. After finalizing most of the HTML templates with Bootstrap and Jinja2, I moved on to implementing the backend routes, initially using arrays instead of SQLite for quick development and testing. Once the core functionality was in place, I integrated SQLite for data persistence, ensuring proper database structure and relationships. Finally, I focused on error correction and testing, refining user interactions, fixing bugs, and validating quiz attempts to ensure a smooth experience.

Frameworks and Libraries Used

Component	Technology Used
Backend	Flask (Python)
Frontend	HTML, CSS, Bootstrap, Jinja2
Database	SQLite
Authentication	Flask-Login (optional)
API Management	Flask-RESTful (if used)
Charts	Chart.js (for summary visualization)

ER Diagram
The main tables include:



1. **Users**: Stores user details such as ID, email, password, and role (admin or regular user).
2. **Subjects**: Contains the subjects created by the admin.
3. **Chapters**: Stores chapters under each subject.
4. **Quizzes**: Holds quiz details, including availability timestamps and duration.
5. **Questions**: Stores multiple-choice questions (MCQs) linked to quizzes, along with the correct answer.
6. **Options**: Stores answer choices for each question.
7. **Quiz Attempts**: Records users' quiz attempts, scores, and completion times.

API Resource Endpoints

Method	Endpoint	Description	Access
POST	/register	User registration	User
GET	/subjects	Retrieve all subjects	User
POST	/api/subjects/add	Add a new subject	Admin
POST	/api/subjects/update	Update an existing subject name	Admin
POST	/api/subjects/remove	Delete a subject	Admin
GET	/subjects/<subject_name>	Retrieve chapters under a subject	User
POST	/api/chapters/add	Add a new chapter to a subject	Admin
POST	/api/chapters/update	Update an existing chapter name	Admin
POST	/api/chapters/remove	Delete a chapter	Admin
GET	/subjects/<subject_name>/<chapter_name>	Retrieve quizzes under a chapter	User

POST	/api/quizzes/create	Create a new quiz	Admin
POST	/api/quizzes/update	Update an existing quiz	Admin
POST	/api/quizzes/delete	Delete a quiz	Admin
GET	/quiz/<quiz_id>/history	Retrieve quiz attempt history	User

Architecture
The Quiz Master - V1 project follows the MVC (Model-View-Controller) pattern to keep the code modular and organized. The controllers (route handlers) are located in app.py, managing API requests and rendering templates. The templates (HTML files) are stored in the templates/ directory, using Jinja2 for dynamic content rendering. Static files such as CSS, JavaScript, and images are in the static/ directory. The database schema and models are handled in init_db.py, which initializes the SQLite database.

Features
- **User Authentication:** Users can register, log in, and manage their accounts. Admins have additional privileges.
- **Quiz Management:** Admins can create subjects, chapters, quizzes, and questions.
- **Quiz Attempt & Evaluation:** Users can attempt quizzes with multiple-choice questions and get immediate feedback.
- **Data Persistence:** All quiz and user data is stored in an SQLite database, ensuring reliability.
- **API Endpoints:** A RESTful API is available, defined in api.yaml, for interaction with quiz data.

Presentation Video Link:
https://drive.google.com/file/d/1HerRmhJPEav8cAsdE1T8Vhi8ol7vB-OL/view?usp=share_link