MAD II Quiz Master Application

Author

Tisya Ahuja
23f1001323
23f1001323@ds.study.iitm.ac.in
I’m Currently in 3rd term of diploma, I'm always keen to explore and learn new things.

Project Overview

Quiz Master-V2 is a multi-user exam preparation platform. Admins manage courses, chapters, and quizzes, creating questions. Users register to attempt quizzes, track scores, and view performance. Built with Flask, VueJS, and SQLite, it features scheduled reminders, activity reports, and data exports, ensuring a comprehensive learning experience.

Technologies Used

Backend

- **Flask**: The core Python web framework for building the application's server-side logic and handling HTTP requests.
- **Flask-RESTful**: An extension for Flask that simplifies the creation of REST APIs.
- **Flask-Security**: Provides comprehensive security features, including authentication (token-based) and role-based access control.
- **Flask-SQLAlchemy**: An ORM (Object Relational Mapper) for interacting with the SQLite database.
- **Celery**: A distributed task queue used for handling asynchronous and scheduled background jobs (e.g., daily reminders, monthly reports, CSV exports).
- **Redis**: Used as a backend for Celery (broker) and for caching data to improve application performance.

Utilities

- **datetime module**: For handling date and time operations, such as formatting and parsing dates.
- **werkzeug.security**: Specifically generate_password_hash and check_password_hash for secure password hashing and verification.
- **smtplib, email.mime.***: Standard Python libraries for sending emails, including multipart messages and attachments (e.g., for monthly reports).
- **csv**: For generating CSV formatted data (e.g., for data exports).
- **requests**: A powerful HTTP library for making requests to external services (e.g., Google Chat Webhooks for reminders).

- **json**: For working with JSON data, both for API requests/responses and potentially for structured data handling.
- **celery.result.AsyncResult**: For managing and checking the status of asynchronous tasks.

Frontend Integration

- **Jinja2**: Flask's default templating engine, used for rendering HTML templates on the server-side, particularly for the application's entry point or email templates.
- **flask.jsonify, flask.send_from_directory**: Flask utilities that facilitate the interaction between the backend and frontend, by sending JSON responses,and serving static files.

Frontend

- **VueJS**: The JavaScript framework used for building the dynamic and interactive user interface.
- **Bootstrap**: (As per project statement) A popular CSS framework used for styling and ensuring a responsive design across various devices.
- **Chart.js**: JavaScript charting library for creating interactive charts.

API Overview

This API provides a comprehensive set of endpoints for managing an educational or quiz-based system. It allows for the management of subjects, chapters, quizzes, questions, user data, and user scores. The API follows a RESTful design, enabling standard CRUD (Create, Read, Update, Delete) operations on various resources, and also includes specific endpoints for administrative summaries and search functionality.

API Endpoint Overview

Below is a detailed breakdown of each API resource, its associated endpoints, the HTTP methods supported, and a brief description of its purpose.

1. User API (UserApi)

Manages user accounts.

HTTP Method	**Endpoint**	**Description**
GET	/api/getuser	Retrieves all users or specific users.
POST	/api/createuser	Creates a new user.
PUT	/api/updateuser/<int:user_id>	Updates an existing user by user_id.
DELETE	/api/deleteuser/<int:user_id>	Deletes a user by user_id.

2. Score API (ScoreApi)

Manages user scores and quiz submission.

HTTP Method	Endpoint	Description
GET	/api/getscore	Retrieves the quiz history/scores for a student.
POST	/api/createscore	Submits quiz data after a student completes a quiz.

3. Question API (QuestionApi)

Manages quiz questions.

HTTP Method	Endpoint	Description
GET	/api/getquestion/<int:quiz_id>	Retrieves questions associated with a specific quiz_id.
POST	/api/createquestion	Creates a new question.
PUT	/api/updatequestion/<int:question_id>	Updates an existing question by question_id.
DELETE	/api/deletequestion/<int:question_id>	Deletes a question by question_id.

4. Quiz API (QuizApi)

Manages quizzes.

HTTP Method	Endpoint	Description
GET	/api/getquiz/<int:chapter_id>	Retrieves quizzes associated with a specific chapter_id.
POST	/api/createquiz	Creates a new quiz.
PUT	/api/updatequiz/<int:quiz_id>	Updates an existing quiz by quiz_id.
DELETE	/api/deletequiz/<int:quiz_id>	Deletes a quiz by quiz_id.

5. Chapter API (ChapterApi)

Manages chapters within subjects.

HTTP Method	Endpoint	Description
GET	/api/getchapter/<int:subject_id>	Retrieves chapters associated with a specific subject_id.
POST	/api/createchapter	Creates a new chapter.
PUT	/api/updatechapter/<int:chapter_id>	Updates an existing chapter by chapter_id.
DELETE	/api/deletechapter/<int:chapter_id>	Deletes a chapter by chapter_id.

6. Subject API (SubjectApi)

Manages subjects.

HTTP Method	Endpoint	Description
GET	/api/getsubject	Retrieves all subjects or specific subjects.
POST	/api/createsubject	Creates a new subject.
PUT	/api/updatesubject/<int:subject_id>	Updates an existing subject by subject_id.
DELETE	/api/deletesubject/<int:subject_id>	Deletes a subject by subject_id.

7. Quiz Info API (QuizInfoApi)

Provides specific information about a quiz.

HTTP Method	Endpoint	Description
GET	/api/getquizinfo/<int:quiz_id>	Retrieves detailed information for a specific quiz_id.

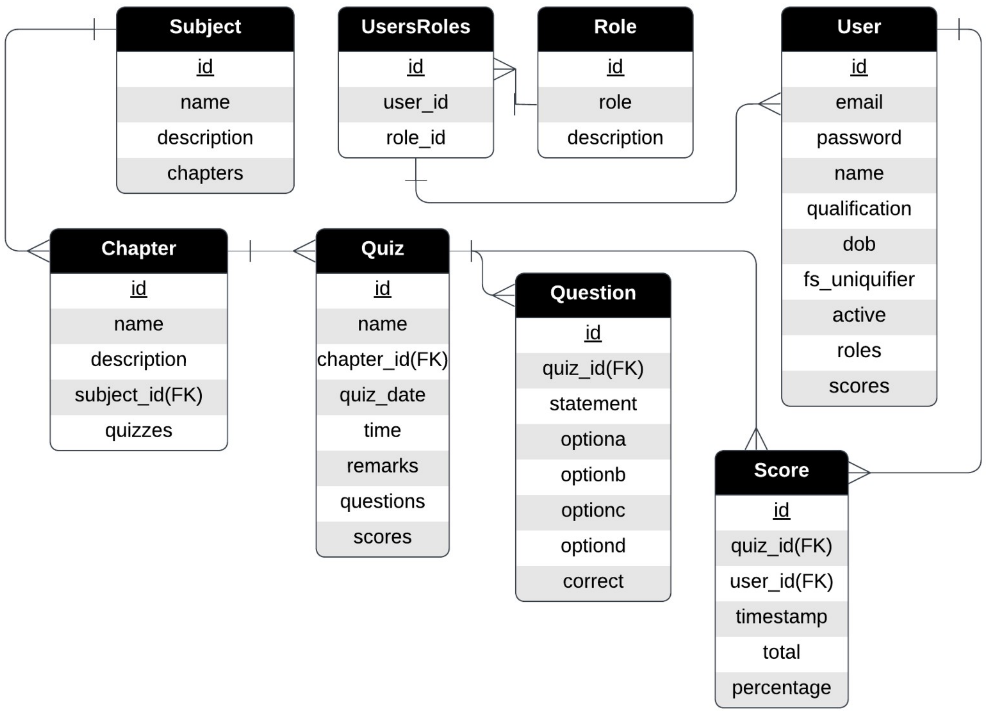
8. Admin API

Provides administrative summary and search functionalities.

HTTP Method	Endpoint	Description
GET	/admin/ShortSummary	Fetches a short summary for the admin page (Total users, quizzes, subjects).
GET	/admin/TopStudents	Fetches the top 3 students by their average score.

GET	/admin/TopQuizes	Fetches subjects of which quizzes are most attempted by students.
GET	/search/<string:query>	Provides search functionality within the admin portal.

ER Diagram for database



Demo Video Link

- https://drive.google.com/file/d/1SQbYMajvCl_eXsvpmOCxTZ6QyR-DTCz2/view?usp=sharing