Quiz master application

Author

Vishal Vinayak Gaikwaad
23f3003477
23f3003477@ds.study.iitm.ac.in

I am currently at the diploma level and pursuing knowledge about the fundamentals of data science.

Description

Quiz Master is a quiz application where users can attempt quizzes on various subjects and chapters while tracking their progress. The Master (admin) can create and schedule quizzes, monitor student attempts, and analyze their performance.

Technologies used

The main technologies used in this application are **Flask, Flask-SQLAlchemy, SQLite, Jinja2, Chart.js, and Bootstrap**.

- **Flask** – A Python web framework used for handling requests, defining routes, and managing application logic.
- **Flask-SQLAlchemy** – Facilitates database interactions by defining models and executing queries on the SQLite database.
- **SQLite** – A relational database used for storing user data, quiz details, and submissions, making the application efficient.
- **Jinja2** – A templating engine used to dynamically render HTML pages, enhancing the user interface.
- **Chart.js** – A JavaScript library used for visualizing quiz performance and user progress with interactive charts.
- **Bootstrap** – A front-end framework that improves the application's design, making it responsive and visually appealing.
- **Flask Sessions & Flash Messages** – Used for user authentication, session management, and displaying notifications.
- **Functools (wraps)** – Helps create decorators for authentication and access control.

DB Schema Design

The database consists of 7 tables:

1. ***User*** (id INTEGER PRIMARY KEY, username STRING UNIQUE, password STRING, fullname STRING, qualification STRING, dob DATE, is_master BOOLEAN)

2. ***Subject*** (id INTEGER PRIMARY KEY, name STRING UNIQUE, description STRING)

3. ***Chapter*** (id INTEGER PRIMARY KEY, name STRING UNIQUE, description STRING, subject_id INTEGER FOREIGN KEY)

4. ***Question*** (id INTEGER PRIMARY KEY, question STRING, option1 STRING, option2 STRING, option3 STRING, option4 STRING, answer STRING, point INTEGER, chapter_id INTEGER FOREIGN KEY)

5. ***Quiz*** (id INTEGER PRIMARY KEY, chapter_id INTEGER FOREIGN KEY, date DATE, duration TIME)
6. ***Score*** (id INTEGER PRIMARY KEY, user_id INTEGER FOREIGN KEY, quiz_id INTEGER FOREIGN KEY, question_id INTEGER FOREIGN KEY, correct_opt INTEGER, submit_opt INTEGER, score INTEGER)

7. ***Submission*** (id INTEGER PRIMARY KEY, user_id INTEGER FOREIGN KEY, quiz_id INTEGER FOREIGN KEY, score INTEGER, total_questions INTEGER, correct_ans INTEGER, date DATE DEFAULT CURRENT_TIMESTAMP)

- USER —Stores user details, including login credentials, personal information, and admin status.
- SUBJECT—Stores details of subjects available for quizzes.
- CHAPTER—-Stores chapters associated with subjects.
- QUESTION—Stores quiz questions, their options, correct answers, and assigned points.
- QUIZ—Stores quizzes along with their associated chapter, date, and duration.
- SCORE—Stores scores of users for individual quiz questions.
- SUBMISSION—Keeps track of quiz submissions, including user performance and completion details

API Design

Authentication & Authorization:
– Secure session-based login and registration with flash messaging.
– Role-based access control using decorators to restrict actions.

Content Management (Master/Admin Only):
– CRUD operations for Subjects, Chapters, Questions, and Quizzes.
– Quizzes include dates and durations.

User Interaction:
– Users can browse and attempt quizzes with real-time scoring and validation.
– Performance tracking with score summaries and answer reviews.

Data Insights:
– Users can track progress through charts (quizzes per subject/month).
– Admins monitor high scores, user engagement, and quiz performance.

Security & Design:
– Flask templates for server-side rendering.
– SQLAlchemy models enforce data integrity.
– Secure session management and role-based restrictions.

Architecture and Features

Architecture

– Database: Stored in `instance/db.sqlite3` (SQLite).
– Templates: All frontend HTML files are in the `templates/` folder.
– Main Application: `app.py` initializes the Flask app.

– Routes (Controllers): Defined in `routes.py` to handle all CRUD operations, user authentication, and submissions.
– Models: Defined in `models.py` and managed using SQLAlchemy.
– Overall Architecture: Follows MVC principles—Models (`models.py`), Views (`templates/`), and Controllers (`routes.py`).

Features

Master (Admin):
– Create, update, and delete subjects, chapters, questions, and quizzes.
– Schedule quizzes and manage quiz details.
– View user performance through charts.

User (Student):
– Attempt quizzes from various subjects and chapters within the given timeframe.
– Track progress and view past quiz scores.

Authentication:
– Secure login and registration for all users.

Video

Video link :- project demo video.mp4
https://drive.google.com/file/d/1nh7RCJOQwP71pS7kE_LE2ShiOpnBPpHX/view?usp=sharing