MAD 1 Project Report: Quiz Master

March 25, 2025

Author

Name: Medha Sen
Roll number: 21f1000057
Student email: 21f1000057@ds.study.iitm.ac.in
Author Background: I am a final year Integrated BS-MS student at Indian Association for the Cultivation of Science, pursuing a major in Computer Science and a minor in Mathematics.

Description

The Quiz Master - V1 project is a multi-user quiz application that allows an admin (quiz master) to create and manage subjects, chapters, quizzes, and questions. At the same time, users can register, attempt quizzes, and view their scores. The project is built using Flask, SQLite, and Jinja2 templating to ensure a seamless user experience with role-based access control.

Frameworks and libraries used

- Flask – Backend framework for application logic and routing.
- Flask Sqlalchemy: Designing models, doing query operations on the database, and committing changes to the database.
- Jinja2, HTML, CSS, Bootstrap - Front-end technologies for dynamic rendering and responsive design.
- SQLite – Database for storing application data.

Project Details

Database Schema

All tables have an id as the primary key.

1. User Table

   - id (Integer, PK) – Unique user identifier.
   - username (String(100), Unique, Not Null) – User's email (login ID).
   - password (String(100), Not Null) – Hashed for security.
   - full_name (String(100), Not Null) – User's full name.
   - qualification (String(100), Nullable) – Educational qualification.
   - dob (Date, Nullable) – Date of birth.

   Constraints and Design Choices: Unique username prevents duplicates. Bcrypt for secure password hashing.

2. Admin Table

   - id (Integer, PK) – Unique admin identifier.
   - username (String(100), Unique, Not Null) – Admin login.
   - password (String(100), Not Null) – Hashed for security.

   Constraints and Design Choices: A primary key is still used even though there is typically only one admin, allowing for future scalability.

3. Subject Table

   - id (Integer, PK) – Unique subject identifier.
   - name (String(100), Unique, Not Null) – Subject name.
   - description (Text, Nullable) – Subject details

   Constraints and Design Choices: Unique constraint on name to avoid duplicate subjects. Text field for description to allow detailed subject information.

4. Chapter Table

   - id (Integer, PK) – Unique chapter identifier.
   - name (String(100), Not Null) – Chapter name.
   - description (Text, Nullable) – Chapter details.
   - subject_id (Integer, FK $\rightarrow$ Subject.id, Not Null) – Links to a subject.

   Constraints and Design Choices: Foreign key subject_id ensures a chapter belongs to one subject. Allows multiple chapters under a subject without duplicates.

5. Quiz Table

   - id (Integer, PK) – Unique quiz identifier.
   - chapter_id (Integer, FK → Chapter.id, Not Null) – Associated chapter.
   - date_of_quiz (Date, Nullable) – Quiz date.
   - time_duration (String(10), Nullable) – Duration (HH:MM).
   - remarks (Text, Nullable) – Additional comments.

   Constraints and Design Choices: A quiz is tied to a chapter for better subject-wise organization.Time_duration is stored as a string for flexibility in formatting (HH:MM).

6. Questions Table

   - id (Integer, PK) – Unique question identifier.
   - quiz_id (Integer, FK → Quiz.id, Not Null) – Associated quiz.
   - question_title (String(100), Not Null) – Short title.
   - question_statement (Text, Not Null) – Full question..
   - option1, option2, option3, option4 (String(300), Not Null) – MCQ options.
   - correct_option (Integer, Not Null) – Correct answer (1-4).

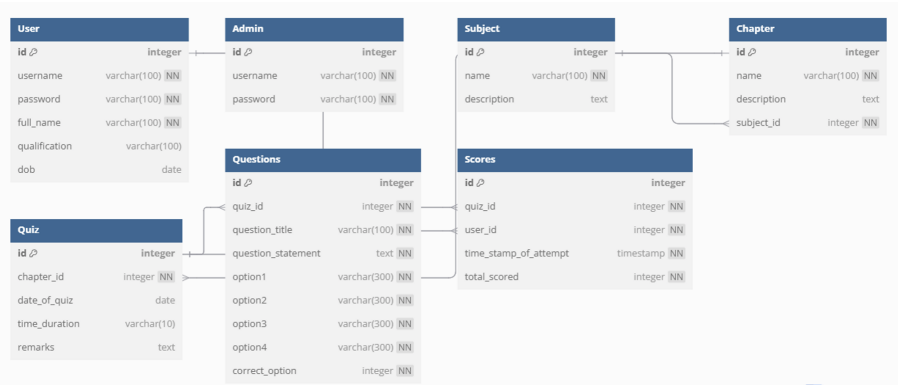
   Constraints and Design Choices: Foreign key quiz_id ensures a question belongs to a quiz. MCQ format with four fixed options ensures standardization. Integer correct_option (1-4) ensures structured answer validation.

7. Scores Table

   - id (Integer, PK) – Unique quiz attempt identifier.
   - quiz_id (Integer, FK → Quiz.id, Not Null) – Associated quiz.
   - user_id (Integer, FK → User.id, Not Null) – User attempting the quiz.
   - time_stamp_of_attempt (DateTime, Not Null) – Attempt time.
   - total_scored (Integer, Not Null) – User's score.

   Constraints and Design Choices: Tracks performance by linking a quiz attempt to a user. DateTime field ensures precise tracking of attempts. Foreign keys (quiz_id, user_id) establish relationships for analytics.

Entity Relationship diagram



API design

As per Milestone 8, I have developed API endpoints to manage subjects, chapters, quizzes, and user scores.

- *GET /api/subjects*: Retrieves all subjects, returning their ID, name, and description.
- *GET /api/subjects/subject_id/chapters*: Fetches all chapters under a given subject.
- *GET /api/chapters/chapter_id/quizzes*: Returns quizzes for a specific chapter, including metadata such as duration and number of questions
- *GET /api/users/user_id/scores*: Retrieves a user's quiz scores, showing quiz details, timestamps, and total scores

Architecture and Features

Architecture

The project follows a modular Flask structure for clarity and maintainability.

- models.py defines the database schema, while app.py handles views.
- Templates serve HTML files, and static houses CSS.
- Instance stores the SQLite database.
- venv is the virtual env folder containing all dependencies and required libraries.

Features

The app supports an Admin (predefined, no registration) and several Users (registration required).The homepage offers:

- User Login: Requires username and password validation.
- Admin Login: Credentials are preset at initialization.
- User Registration: Requires full name, email, qualification, DOB, and password. Duplicate usernames are prevented.

User Dashboard and Quiz Features

Users access a dashboard to browse and attempt timed quizzes before their expiration date. If the user fails to submit, the quiz auto-submits. Each quiz card has a *View* option displaying the Quiz details. A search bar allows filtering by chapter, subject, or keyword. The *Scores* dashboard displays the quiz attempt history of a user and the *Summary* dashboard presents the user performance summary with graphs displaying subject-wise comparison between highest scores and total number of attempts.

Admin Dashboard and Management Controls

Admins manage subjects, chapters, and quizzes via a dashboard, with options to add, edit, and delete entries. Search bars allow the Admin to search for Subjects, chapters, quizzes and users. The Quiz Dashboard allows quiz creation, including date, duration, and MCQ-based questions, all of which are editable. The Scores Summary Dashboard provides:

- User Summary Table: Displays user details with a searchable performance analysis.
- Quiz summary Table: Provides a concise overview of all quizzes.
- A graph comparing the number of quiz attempts for different chapters.

Video Link

MAD 1 Video presentation