Author

Samyabrata Roy
22f2001443
22f2001443@ds.study.iitm.ac.in

I am pursuing a BS in Data Science from IIT Madras and a BSc in Statistics from Sister Nivedita University, along with a diploma in Web Development from CITC Chandigarh. I have a strong interest in Non-Parametric Statistical Inference and Machine Learning and truly enjoyed the journey of developing the WebApp for my projects.

Description

In this project, I was tasked with developing a multi-user quiz application using Flask, Jinja2, SQLite, and Bootstrap. The application features an admin (Quiz Master) who manages users, subjects, chapters, and quizzes, while users can attempt quizzes and view their scores. It supports CRUD operations, user authentication, quiz management, and data visualization.

Technologies used

Flask – The core framework used for building the web application, handling routing, and managing HTTP requests. Also used for handling web rendering templates, managing redirects, displaying flash messages, and maintaining user sessions.
Flask-SQLAlchemy – An extension of Flask, used for database management and handling connections across the app.
Session – Used to store user IDs after login, ensuring secure authentication and maintaining session states.
HTML5 – Used for designing the structure of web pages.
CSS – Applied for styling and enhancing the visual appeal of the application.
JavaScript (JS) – Implemented to enable on-page search mechanisms.
Werkzeug Security (generate_password_hash, check_password_hash) – Used for securely hashing and verifying passwords.
Datetime (datetime) – Utilized for handling and storing timestamps related to quizzes and user activities.
Collections (Counter) – Used for counting occurrences, such as quiz attempts and scores.
Matplotlib (matplotlib) – Applied for data visualization, such as generating summary charts of quiz results

DB Schema Design

Entities and Their Relationships

1. **User Table (One-to-Many with Score, Many-to-Many with Role)**
    a. Stores user details such as name, email, password, date of birth, and qualification.
    b. Connected to **UserRole** (many-to-many with Role) and **Score** (one-to-many with Score).
2. **Role Table (Many-to-Many with User)**
    a. Defines roles (e.g., admin, student).
    b. Connected to **UserRole**, which bridges users and roles.
3. **UserRole Table (Many-to-Many Bridge between User and Role)**
    a. Links users to their assigned roles.
4. **Subject Table (One-to-Many with Chapter)**
    a. Represents subjects in which quizzes are conducted.

    b. Connected to **Chapter** (one subject can have multiple chapters).
5. **Chapter Table (One-to-Many with Quiz, Many-to-One with Subject)**
    a. Represents different chapters under a subject.
    b. Connected to **Subject** (many-to-one) and **Quiz** (one-to-many).
6. **Quiz Table (One-to-Many with Question, Many-to-One with Chapter, One-to-Many with Score)**
    a. Each quiz is associated with a chapter.
    b. Connected to **Question** (one-to-many) and **Score** (one-to-many).
7. **Question Table (Many-to-One with Quiz)**
    a. Stores questions along with multiple choices.
    b. Connected to **Quiz** (many-to-one).
8. **Score Table (Many-to-One with User and Quiz)**
    a. Tracks quiz scores for each user.
    b. Connected to **User** and **Quiz** (many-to-one for both).

Key Relationships and Constraints

- **User & Role (Many-to-Many)**
    - A user can have multiple roles, and a role can belong to multiple users.
- **User & Score (One-to-Many)**
    - A user can take multiple quizzes, and each attempt is recorded in the Score table.
- **Subject & Chapter (One-to-Many)**
    - A subject can have multiple chapters, but each chapter belongs to a single subject.
- **Chapter & Quiz (One-to-Many)**
    - A chapter can have multiple quizzes, but each quiz is tied to a single chapter.
- **Quiz & Question (One-to-Many)**
    - A quiz consists of multiple questions, but each question belongs to a single quiz.
- **Quiz & Score (One-to-Many)**
    - A quiz can have multiple attempts, tracked in the Score table.

Cascade Behavior

CASCADE DELETE is used for related records:

- If a user is deleted, their scores are deleted.
- If a quiz is deleted, associated questions and scores are removed.
- If a chapter is deleted, related quizzes are removed.

Architecture and Features

The project follows a structured organization for efficient management. The root directory contains key folders and files. The **controller** folder handles backend functionality with config.py for configurations, database.py for database interactions, model.py for defining data structures, and route.py for API routing. The **instance** folder stores the database file. The **html_temps** folder manages HTML templates, including navigation tabs. The **static** folder contains two subdirectories: **CSS** for stylesheets and **Img** for images. The **app.py** file initializes and runs the Flask application. A project report in PDF format is also present in the root directory.

The system features an **Admin Panel** that allows complete management of subjects, chapters, and quizzes. Admins can add, edit, and delete subjects, while also managing chapters within each subject. For every chapter, quizzes can be created and modified, with full control over

questions and multiple-choice options, including setting correct answers and scoring rules. On the **User Panel**, users can access subjects, attempt quizzes, and track their performance.

The controllers that were used:

```
@app.route('/admin_')
@app.route('/remove_sub', methods=['GET', 'POST'])
@app.route('/add_sub', methods=['GET', 'POST'])
@app.route('/user_')
@app.route('/user/quiz')
@app.route('/user/profile')
@app.route('/login', methods=['GET', 'POST'])
@app.route('/register', methods=['GET', 'POST'])
@app.route('/logout', methods=['GET', 'POST'])
@app.route('/forget', methods=['GET', 'POST'])
@app.route('/subject_dash')
@app.route('/quiz_dash')
@app.route('/quiz_dash/del_quiz',methods=['POST','GET'])
@app.route('/quiz_dash/add_quiz',methods=['POST','GET'])
@app.route('/question_delete',methods=['GET','POST'])
@app.route('/stat',methods=['GET','POST'])
@app.route('/question_details',methods=['GET','POST'])
@app.route('/question',methods=['GET','POST'])
@app.route('/remove_user',methods=['POST'])
@app.route('/quiz_question',methods=['GET','POST'])
@app.route('/score',methods=['Get','POST'])
@app.route('/chapter_dash')
@app.route('/add_chapter',methods=['Get','POST'])
@app.route('/remove_chap',methods=['Get','POST'])
@app.route('/edit',methods=['GET','POST'])
```

Video

Link: https://drive.google.com/file/d/1h2BPRwqGloJK-ouZrI7VdxC3k1vXNB42/view?usp=drive_link