Quiz Master – V1 Project Report

Modern Application Development – 1 (MAD 1)

IIT MADRAS

1. Student Details

Name – VIVEK KUMAR

Roll no. – 21F3000834

Email Id – 21f3000834@ds.study.iitm.ac.in

2. Project Details

Problem Statement

Quiz Master - V1 is a multi-user application that serves as an exam preparation platform for multiple courses. The platform consists of two roles:

- **Admin** (Quiz Master): Has full control over the system, including user management, subject creation, chapter management, and quiz/question management.
- **User:** Can register, log in, select subjects/chapters, attempt quizzes, and view quiz scores.

Approach

1. **System Design:**
    - Defined user roles and database schema.
    - Created Entity-Relationship (ER) diagrams to structure the database.
2. **Backend Development:**
    - Implemented using **Flask** framework.

    - Created API routes for CRUD operations on subjects, chapters, quizzes, and users.

3. **Frontend Development:**
    - Designed the UI using **HTML, CSS, Bootstrap, and Jinja2 templating**.
    - Implemented form validations and interactive elements.
4. **Database:**
    - Used **SQLite** for data storage and management.
    - Created tables programmatically via Flask models.
5. **Testing & Deployment:**
    - Tested different user interactions.
    - Ensured database consistency and UI responsiveness.

3. Frameworks and Libraries Used

- **Flask:** Backend framework for building the web application.
- **SQL Alchemy:** ORM (Object-Relational Mapping) tool for database interactions.
- **SQLite:** Database management system for storing application data.
- **HTML/CSS/JavaScript:** Frontend technologies for user interface design and interactivity.
- **Flask-Login:** Extension for managing user sessions and authentication.
- **Datetime:** Python library for handling date and time operations.
- **Jinja2:** Template engine for rendering dynamic HTML content.
- **Werkzeug:** Utility for securely managing passwords and authentication.
- **ChartJS:** User for creating different types of charts on the admin dashboard.
- **Flask-WTF:** For Form validation.

4. DB Schema Design

Database Structure

1. User Management

- **Admin**: Stores admin login credentials with hashed passwords.
- **User**: Stores user details like name, contact, program, and authentication data.
    - Foreign keys: program_id, discipline_id, level_id (links to academic structure).

2. Academic Structure

- **Program** → Contains different educational programs.
- **Discipline** → Belongs to a program, contains multiple levels.
- **Level** → Represents different levels within a discipline (e.g., Foundation, Diploma).
- **Subject** → Connected to programs, disciplines, and levels.
- **Chapter** → Linked to subjects, contains quizzes.

3. Quiz System

- **Quiz** → Contains metadata like duration, total marks, and passing criteria.
- **Question** → Stores multiple-choice questions with options and correct answers.
- **Score** → Stores user performance in a quiz, with a unique constraint on user_id and quiz_id.

4. Additional Features

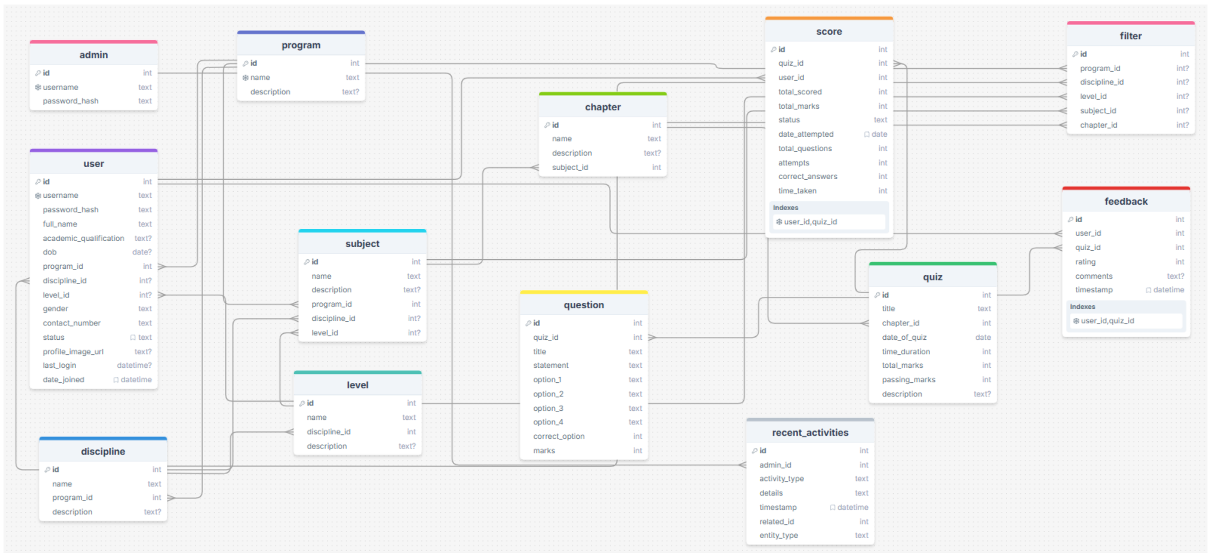
- **Feedback** → Allows users to rate and review quizzes.
- **RecentActivity** → Logs admin actions for tracking changes.
- **Filter** → Supports filtering quizzes based on academic parameters.

Reasons Behind the Design

1. **Normalization & Scalability**
    - Separate tables for Users, Programs, Disciplines, and Levels prevent data redundancy.

- Foreign key constraints maintain data integrity.

2. **Security & Data Integrity**
   - Passwords are hashed for security.
   - Constraints like unique=True and nullable=False ensure valid data input.
   - UniqueConstraint in Score prevents duplicate quiz attempts by a user.
3. **Efficient Querying**
   - Relationships allow easy querying of related data (e.g., all quizzes under a subject).
   - ondelete="CASCADE" ensures related data is removed when a parent record is deleted.
4. **Extensibility**
   - The modular structure allows adding new levels, subjects, and quizzes without restructuring.



5. API resource Endpoints

- /users/<id> --- Fetch, update, delete a specific user by ID
- /quizzes/<id> --- Fetch, update, delete a quiz by ID
- /attempts/<id> --- Fetch a specific attempt by ID

6. Presentation Video Link

Google Drive Link