Name: Muddada Satwik
Roll: 23f1001933
Email: 23f1001933@ds.study.iitm.ac.in

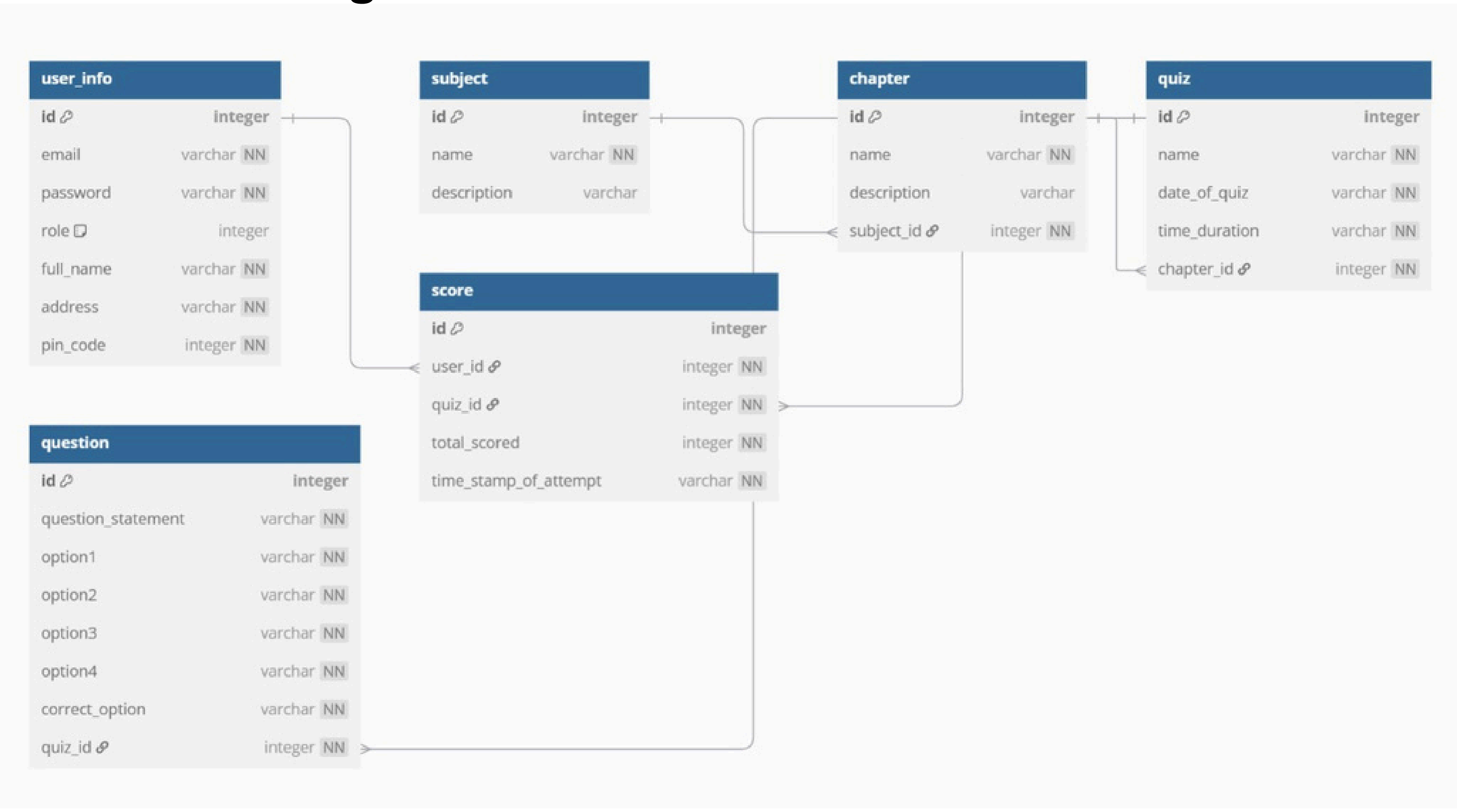
I am Satwik, an undergraduate in ECE, trying to gain knowledge in the field of data science

Description:

I have used the Flask framework to build the Quiz Master V1 application, which is a multi-user exam preparation web application. The application supports two roles: admin (quiz master) and regular users (students). It features multiple routes to serve different functionalities and dynamically renders templates using Jinja2, HTML, and Bootstrap for styling. All user interactions and data submissions are managed via Flask's routing and form handling, while SQLAlchemy is used for database modeling with SQLite as the backend.

The app allows users to register/login, choose subjects and chapters, and attempt chapter-wise quizzes created by the admin. Each quiz consists of multiple-choice questions, and the application tracks and stores quiz scores. The admin has complete control over the app for creating and updating subjects, chapters, and quizzes, and visualizing performance through charts generated using Matplotlib.

DB Schema Diagram:



Technologies Used:

Flask – For creating the application and routing.
render_template, request, url_for, redirect, session – Flask modules for rendering templates and handling form data.
Flask-SQLAlchemy – For defining database models and relationships.
HTML – For creating the web pages.
CSS – For styling the components.
Bootstrap – For responsive and clean UI design.
Jinja2 – For dynamic content rendering within templates.
SQLite – As the database to store all application data.
Matplotlib – For generating visual graphs of user performance.
datetime – For handling and storing quiz dates and timestamps.

Architecture and Features:

I have developed the Quiz Master V1 application using the Flask framework, serving as a multi-user exam preparation platform with two distinct roles: admin (quiz master) and regular users. The application features dynamic routing and rendering using Jinja2, with front-end development handled via HTML, CSS, and Bootstrap. Data handling and persistence are achieved through Flask-SQLAlchemy and SQLite, while user engagement and quiz performance are visualized using Matplotlib. Upon launching the app, users are greeted with a home page offering options to register or log in. The admin accesses a dedicated dashboard where they can create, edit, and delete subjects, chapters, and quizzes, as well as add MCQ-type questions, define quiz durations, and view user analytics. The admin also manages user accounts and can search for specific subjects or quizzes. Registered users can log in to their own dashboards, view available subjects and chapters, and attempt quizzes, with their performance automatically recorded and visualized over time. Each quiz consists of multiple-choice questions with one correct answer, and upon completion, the score is saved to the database along with a timestamp. Users can view their quiz history and overall performance trends through generated graphs. The application architecture is modular, with app.py handling the app initialization and controllers.py handling all routing and controller logic, models.py defining the database schema, templates stored in the templates folder, and static assets like CSS housed in the static folder. All database tables—including User, Subject, Chapter, Quiz, Question, and Score—are created programmatically using SQLAlchemy, with clearly defined relationships through foreign keys. The app runs entirely on a local machine at http://127.0.0.1:5000, providing a seamless and interactive learning experience.

Video link