Quiz Master V1

Author

Name : Soham Maity
Roll No : 24f2008474
Email ID : 24f2008474@ds.study.iitm.ac.in

Dual degree student at IIT Madras (BS Data Science and Programming) and Bengal Institute of Technology (BTech Computer Science). Passionate about coding and exploring emerging technologies.

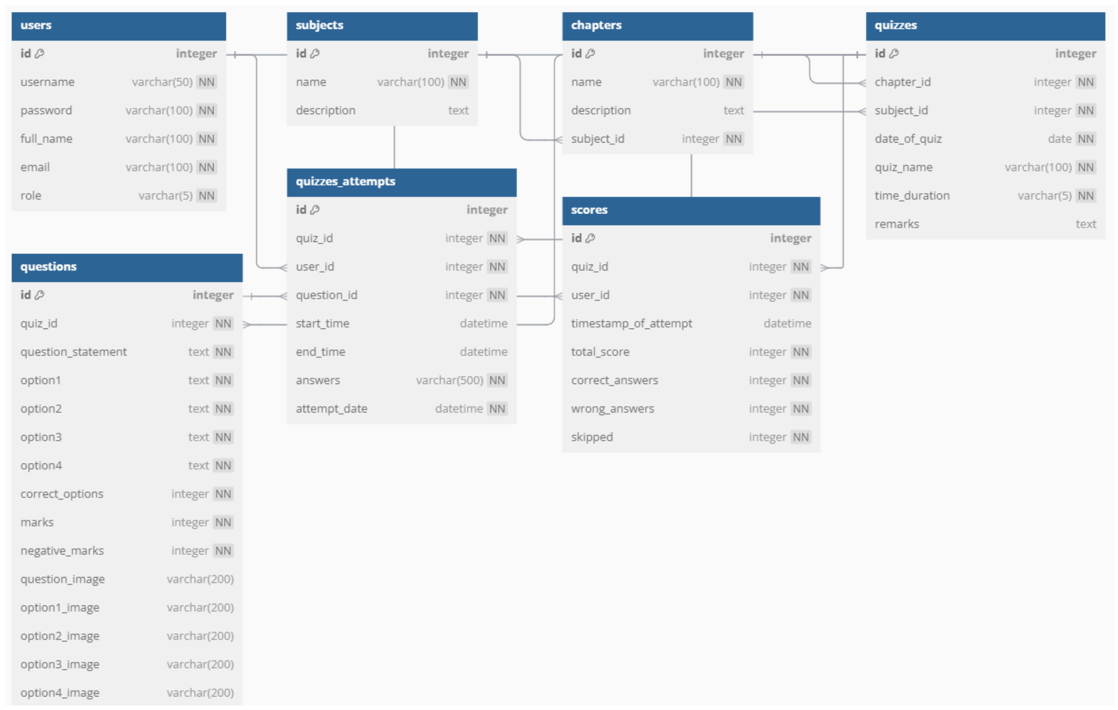
Description

QuizMaster is a comprehensive web-based quiz management platform that enables administrators to create and manage subjects, chapters, quizzes, and questions, while allowing users to take quizzes and track their performance. The application uses a role-based access system that differentiates between admin and regular user capabilities.

Technologies Used

1. Flask : Core web framework for routing and request handling
2. Flask-SQLAlchemy : ORM for database interactions and model definitions
3. Flask-Login : Handles user authentication and session management
4. Flask-Restful : JSON endpoints for programmatic data access
5. Jinja2 : Template engine for rendering dynamic HTML content
6. Bootstrap : Frontend framework for responsive design and UI components
7. SQLite : Lightweight database for data persistence
8. Matplotlib : For generating performance visualization charts

DB Schema Design



1. Hierarchical Organization :
   The schema follows a natural educational structure (Subject → Chapter → Quiz → Question), making queries logical and efficient.
2. Referential Integrity :
   CASCADE deletion constraints ensure that when a parent entity is deleted, all its dependent entities are removed, preventing orphaned records.
3. Separation of Concerns :
   QuizAttempt tracks individual question responses for detailed analysis
   Score provides summary statistics without needing to query all attempt records
4. Flexible Content Support :
   Image fields enable rich multimedia quiz content while remaining optional.
5. Performance Analytics :
   Various timestamp fields and performance metrics enable comprehensive analysis of user learning patterns.
6. Role-Based Access :
   Simple role field enables differentiation between regular users and administrators, controlling feature access.
7. Data Validation :
   Constraints like unique usernames/emails and not-null fields enforce data integrity at the database level.

API Design

The QuizMaster application implements a RESTful API using Flask-RESTful that provides programmatic access to core application data. The API offers endpoints for four primary resources :

Subjects API:
GET /api/subjects - Retrieves all available subjects

POST /api/subjects - Creates a new subject (admin only)
GET /api/subjects/<subject_id> - Retrieves details of a specific subject

Chapters API:
GET /api/chapters - Retrieves chapters with optional filtering by subject_id

Quizzes API:
GET /api/quizzes - Retrieves quizzes with optional filtering by subject_id and/or chapter_id

Scores API:
GET /api/scores - Retrieves scores with role-based access control (admins can view all scores or filter by user_id; regular users can only view their own scores)

The QuizMaster API is built using Flask-RESTful within a dedicated Flask Blueprint, providing structured access to subjects, chapters, quizzes, and scores through class-based resources. Authentication leverages Flask-Login with role-based permissions that restrict certain operations to administrators only.
The implementation supports flexible query parameter filtering and maintains a consistent JSON response format with appropriate HTTP status codes for success and error conditions. All responses include a standard structure with a success indicator and relevant data payload while comprehensive error handling ensures informative feedback for API consumers.

Project Architecture

The QuizMaster application follows an MVC pattern with clear separation of concerns. Models are defined in the models directory using SQLAlchemy ORM, controllers are organized as Flask blueprints in the controllers directory (including api.py for REST endpoints), templates are stored in a dedicated templates folder, and static assets reside in a static directory. This modular structure promotes maintainability and facilitates the separation of business logic, data access, and presentation concerns.

Feature Implementation

QuizMaster delivers a comprehensive quiz management system with role-based access control (admin/user).
Key features include hierarchical content organization (subjects → chapters → quizzes → questions), multimedia question support, timed quiz-taking with advanced scoring (positive/negative marking), detailed performance analytics, and a RESTful API for programmatic data access.
The application implements responsive design for cross-device compatibility and robust data validation throughout. Administrators benefit from dashboard analytics while regular users enjoy an intuitive interface for quiz attempts and performance tracking.

Video Demonstration

Project Demonstration.mp4