Project Report: QuizzyVerse

Author
Vaishnavi Mishra
Roll Number: 23f2003607
Email: 23f2003607@ds.study.iitm.ac.in

I am currently pursuing the Diploma level in the BS in Data Science program at IIT Madras. I am passionate about building interactive and scalable web applications.

Description

QuizzyVerse is a multi-user quiz platform for exam preparation across various subjects. It supports a super-admin (Quiz Master) and multiple users who can register, attempt quizzes, and view their performance reports.

AI/LLM Used
Minimal use of LLM (under 5%) for writing support, prompt structuring, and Bootstrap component inspiration. Code was written manually.

Technologies Used

Backend: Flask, Flask-CORS, Flask-JWT, Celery, Redis
Frontend: Vue.js, Axios, Bootstrap
Database: SQLite (via SQLAlchemy ORM)
Caching/Jobs: Redis and Celery

Purpose: These tools enable secure user handling, asynchronous job execution, fast UI, and lightweight local development.

DB Schema Design

Users
- id (PK), name, email, password, role, dob, qualification
- Role differentiates Admin and Users.

Subjects
- id (PK), name, description

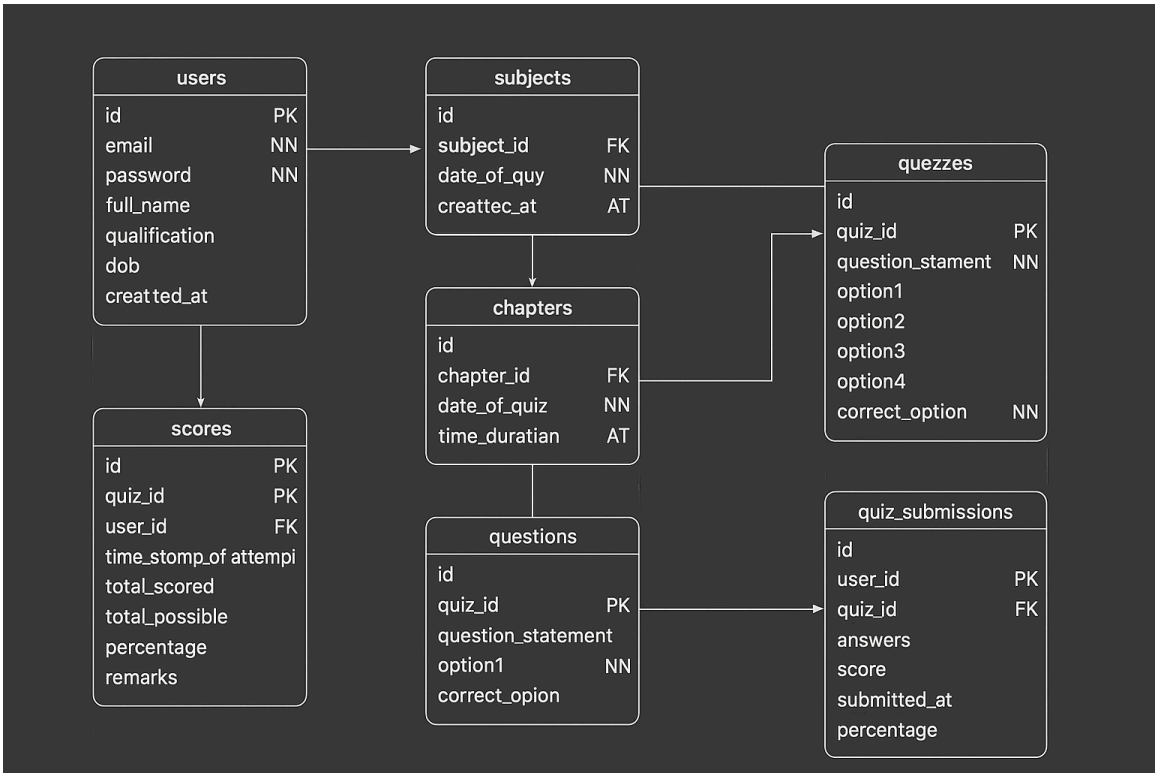
Chapters
- id (PK), name, description, subject_id (FK)

Quizzes
- id (PK), chapter_id (FK), date_of_quiz, time_duration, remarks

Questions
- id (PK), quiz_id (FK), question_statement, option1-option4, correct_option

QuizSubmission
- id (PK), user_id (FK), quiz_id (FK), timestamp, total_score

Design Reasoning: Normalized schema ensures clear linkage and scalability. QuizSubmission table supports historical tracking and performance reports.



API Design

QuizzyVerse includes RESTful APIs for:
- Authentication: User registration/login (JWT-based), Admin login (pre-seeded)
- Admin Operations: CRUD on subjects, chapters, quizzes, questions
- User Operations: Attempt quizzes, view scores, download CSVs
- Background Tasks: Trigger reminders, monthly reports, CSV exports

Architecture and Features

Architecture
- Backend (Flask): Contains controllers, models, Celery jobs, and caching
- Frontend (Vue.js): Interacts with backend via Axios and shows UI dynamically
- Bootstrap: Used for styling across pages
- Celery + Redis: Handle background jobs and performance caching

Implemented Features
- ✅ Role-based Auth (JWT): Admin/User separation

- ✅ Admin Dashboard: Manage users, subjects, chapters, quizzes
- ✅ Quiz Management: Create quizzes, add MCQs, set timers
- ✅ User Dashboard: View upcoming quizzes, attempt, view score
- ✅ Quiz Attempt Timer: Each quiz has a timer
- ✅ Caching: Quiz data cached for 5 minutes to reduce DB load
- ✅ Daily Reminders: Sent via Google Chat webhook
- ✅ Monthly Email Report: HTML report sent via email on the 1st of each month
- ✅ CSV Export: Users/Admins can trigger async CSV download of quiz performance via Celery

Video

QuizzyVerse_video.mp4