Quiz Master Project Report

1. Student & Project Details

Student Name : Nidhi Heer
Roll No : 24f1002483

Project Name: Quiz Master 2
Project Type: Full-Stack Web Application

Technology Stack:

- Backend: Python Flask (REST API)
- Frontend: Vue.js 3
- Database: SQLite (SQLAlchemy ORM)

Additional Technologies:

- Redis
- Celery

2. Problem Statement

Develop an online quiz platform with:

- Admin Features: Create/manage subjects, chapters, quizzes, and questions.
- User Features: Take quizzes, view scores, track progress.
- Core Functionality:
- Real-time quiz timer & auto-submission
- Score analytics & progress tracking
- Search functionality
- Session management with timeout

3. Approach & Solution

3.1 Architecture Design

- Backend: Flask REST API (modular blueprints)
- Frontend: Vue.js 3 (component-based)
- Database: SQLite with relational models
- Caching: Redis for performance
- Background Tasks: Celery

3.2 Key Features

Feature	Description
User Auth	JWT-based login with role-based access (Admin/User)
Quiz Management	CRUD operations for subjects → chapters → quizzes → questions
Real-time Quiz	Timer-based interface with auto-submission
Analytics	Score tracking, attempt history, and detailed results

4. Technologies Used

Backend

- Flask, SQLAlchemy, Flask-JWT, Redis, Celery

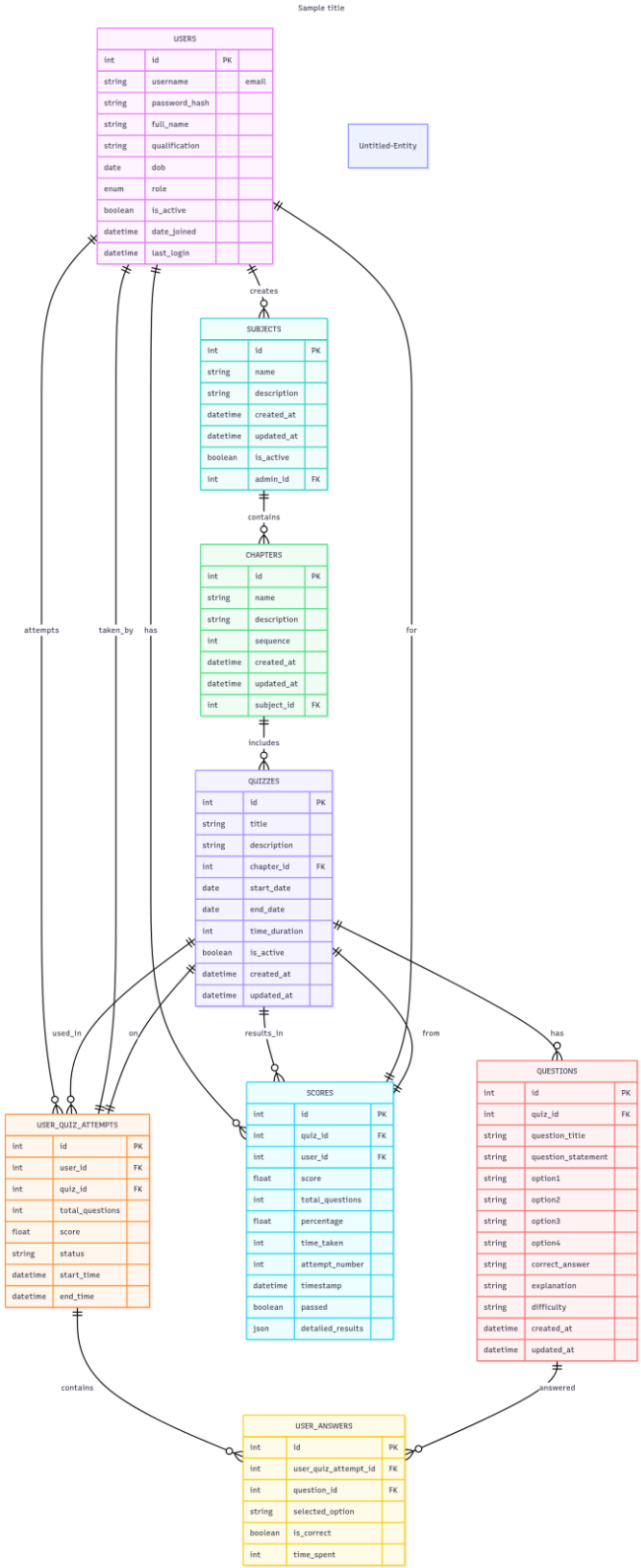
Frontend

- Vue.js 3, Vue Router, Axios

Deployment

- Docker, Docker Compose

5. Database Schema (ER Diagram)



Core Tables
- Users (`id`, `username`, `role`, `is_active`)
- Subjects (`id`, `name`, `admin_id`)
- Chapters (`id`, `name`, `subject_id`)
- Quizzes (`id`, `title`, `chapter_id`, `time_duration`)
- Questions (`id`, `quiz_id`, `options`, `correct_answer`)
- Scores (`user_id`, `quiz_id`, `score`, `timestamp`)

Relationships
- One-to-Many: Subjects → Chapters → Quizzes → Questions
- Many-to-Many: Users ↔ Quizzes (via Scores)

6. API Endpoints

Authentication
- POST /login, POST /register, GET /me

Admin Routes
- POST /subjects, DELETE /quizzes/<id>

User Routes
- POST /quiz/start, GET /scores/history

Search
- GET /search/questions?query=...

7. Key Features
- User Management: Registration, role-based access.
- Quiz System: Timer, auto-submission, multi-choice questions.
- Analytics: Score tracking, progress reports.
- Security: JWT, password hashing, rate limiting.

8. Deployment
Dockerized with containers for:

- Flask backend
- Vue.js frontend
- Redis cache
- Celery workers

9. Conclusion
The Quiz Master project delivers a scalable, secure, and user-friendly quiz platform with:

- Full-featured admin & user dashboards
- Real-time quiz experience
- Robust analytics
- Production-ready deployment

Drive Link to Presentation Video:
https://drive.google.com/file/d/1W0Fvw8qC3_bgXQBHKSuGEo1SfOXxRFJv/view?usp=drivesdk