Project Report: Quiz Master

Student Details

Name: Mohd Afsar
Degree: B.S. Degree in Data Science and Applications, IIT Madras

---

Project Details

Project Title: Quiz Master
Course: Modern Application Development - 2 (MAD 2)
Project Description: Quiz Master is an interactive web-based platform designed to facilitate the creation, management, and evaluation of quizzes. The platform allows users to register, attempt quizzes, and track their performance. It supports subject- and chapter-based quizzes, ensuring a structured learning experience.

Approach to Problem Statement:

1. **Understanding Requirements:** Initially, I analyzed the project requirements to define the core functionalities and user flows.

2. **Database Design:** Designed an efficient relational schema to manage users, quizzes, questions, and scores.

3. **Backend Development:** Implemented a RESTful API using Flask to handle authentication, quiz management, and result tracking.

4. **Frontend & UI Integration:** Created a user-friendly interface with dynamic quiz functionality.

5. **Testing & Debugging:** Performed rigorous testing to ensure smooth user experience and functionality.

6. **Documentation & Submission:** Referred to official Flask and SQLAlchemy documentation to validate the implementation.

Technologies & Frameworks Used

- **Backend:** Flask, Flask-SQLAlchemy, Flask-Security
- **Database:** SQLite
- **Frontend:** Vue-CLI, CSS
- **Other Libraries:** datetime, bcrypt (for password hashing)

ER Diagram

The database consists of multiple tables with the following relationships:

1. **User** (has a many-to-many relationship with **Role** and **Subject**)
2. **Qualification** (one-to-many with **User** and **Subject**)
3. **Subject** (one-to-many with **Chapter** and **Question**)
4. **Chapter** (one-to-many with **Quiz** and **Question**)
5. **Quiz** (one-to-many with **Score** and **Question**)
6. **UserQuizzes** (tracks user quiz attempts)
7. **Score** (stores user performance data)
8. **RecentActivity** (logs user interactions)

(ER diagram not available)

API Resource Endpoints

(API documentation YAML file attached separately)

YAML FILE NAME: api-doc.yaml

Architecture & Features

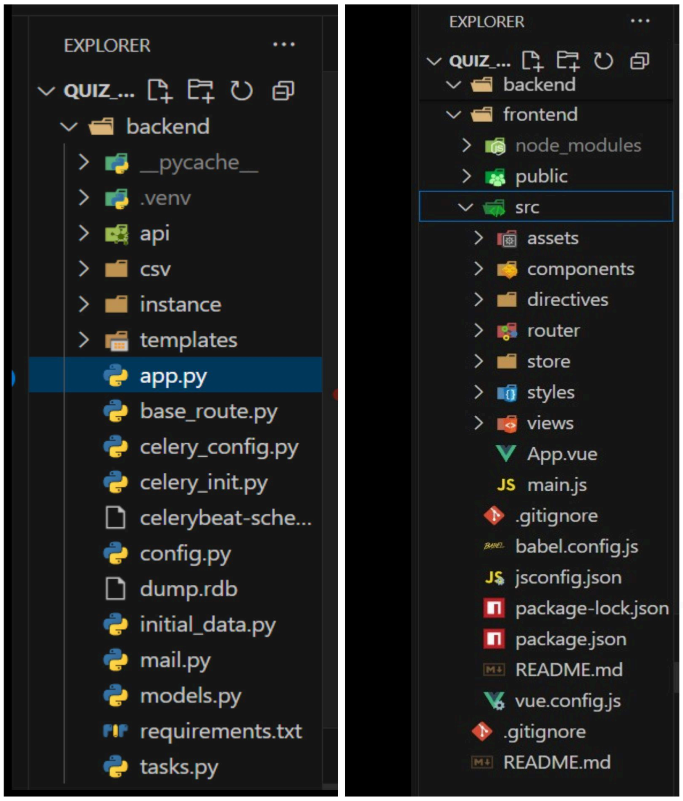
System Architecture:

- **MVC Structure** (Model-View-Controller) followed for structured development.
- **Token-Based Authentication** implemented for secure access.
- **Role-Based Access Control (RBAC)** for admin and user functionalities.

Key Features:

- **User Authentication** with role management.
- **Quiz Creation & Management** (by admin/authorized users).
- **Subject & Chapter-based quiz categorization.**
- **Quiz Attempt Tracking** and performance evaluation.
- **Interactive UI** for a seamless user experience.

My Project directory:



```
EXPLORER
QUIZ_...
  backend
    __pycache__
    .venv
    api
    csv
    instance
    templates
    app.py
    base_route.py
    celery_config.py
    celery_init.py
    celerybeat-sche...
    config.py
    dump.rdb
    initial_data.py
    mail.py
    models.py
    requirements.txt
    tasks.py
```

```
EXPLORER
QUIZ_...
  backend
  frontend
    node_modules
    public
    src
      assets
      components
      directives
      router
      store
      styles
      views
      App.vue
      main.js
    .gitignore
    babel.config.js
    jsconfig.json
    package-lock.json
    package.json
    README.md
    vue.config.js
  .gitignore
  README.md
```

Presentation Video Link

Video Link: Click Here or ⬇️

Web Application Development || Python || Vue || QuizMaster

Submitted by:
Mohd Afsar
IIT Madras, B.S. in Data Science & Applications