ONLINE QUIZ APPLICATION

- SHREYANSH AGRAWAL

Problem:

There is a need for a console-based quiz application in Python that provides a robust and user-friendly platform for creating, managing, and taking quizzes.

Objectives:

To develop a console-based Python quiz application that enables users to create, manage, and take quizzes efficiently while ensuring data security and user satisfaction.

Requirements:

1. Functional Requirements -
    a. User Authentication:
        i. Super Admin, Admins, and Users should have separate login credentials to access their respective functionalities.
        ii. Admins should be assigned a default password when created by Super Admin
        iii. Users should be able to Sign Up
    b. User Management:
        i. Super Admin can create, view, and delete admin accounts.
        ii. Admin can view user data and delete user accounts.
        iii. Users can view their past scores and leaderboard.
    c. Quiz Management:
        i. Admins can manage quiz categories, including creating, viewing, updating, and deleting them.
        ii. Admins can manage questions, including viewing and adding them.
        iii. Admins can view questions by quiz category.
        iv. Admins should be able to import a set of questions with options from a JSON file into the system.
    d. Quiz Taking:
        i. Users can take quizzes and receive scores.
        ii. Questions should be displayed one at a time, and users should be able to respond to the questions one by one.
        iii. Quiz should be automatically graded, and the score should be recorded for the user.
    e. Leaderboard:
        i. Users can view a leaderboard to see the top scores achieved by other users.

2. Non-Functional Requirements –
    a. Security:
        i. User passwords must be securely stored using encryption.
        ii. Role-based access control should be implemented, ensuring that only authorized users can perform specific actions.
    b. Data Integrity
        i. All data entered and stored in the system should be accurate and reliable.
        ii. The system should provide data validation to prevent erroneous data entry.
    c. Logging:
        i. The system should maintain a logging of significant events and actions taken by users for accountability and tracking purposes.
    d. Documentation
        i. Comprehensive and up-to-date documentation should be available for system users and administrators.

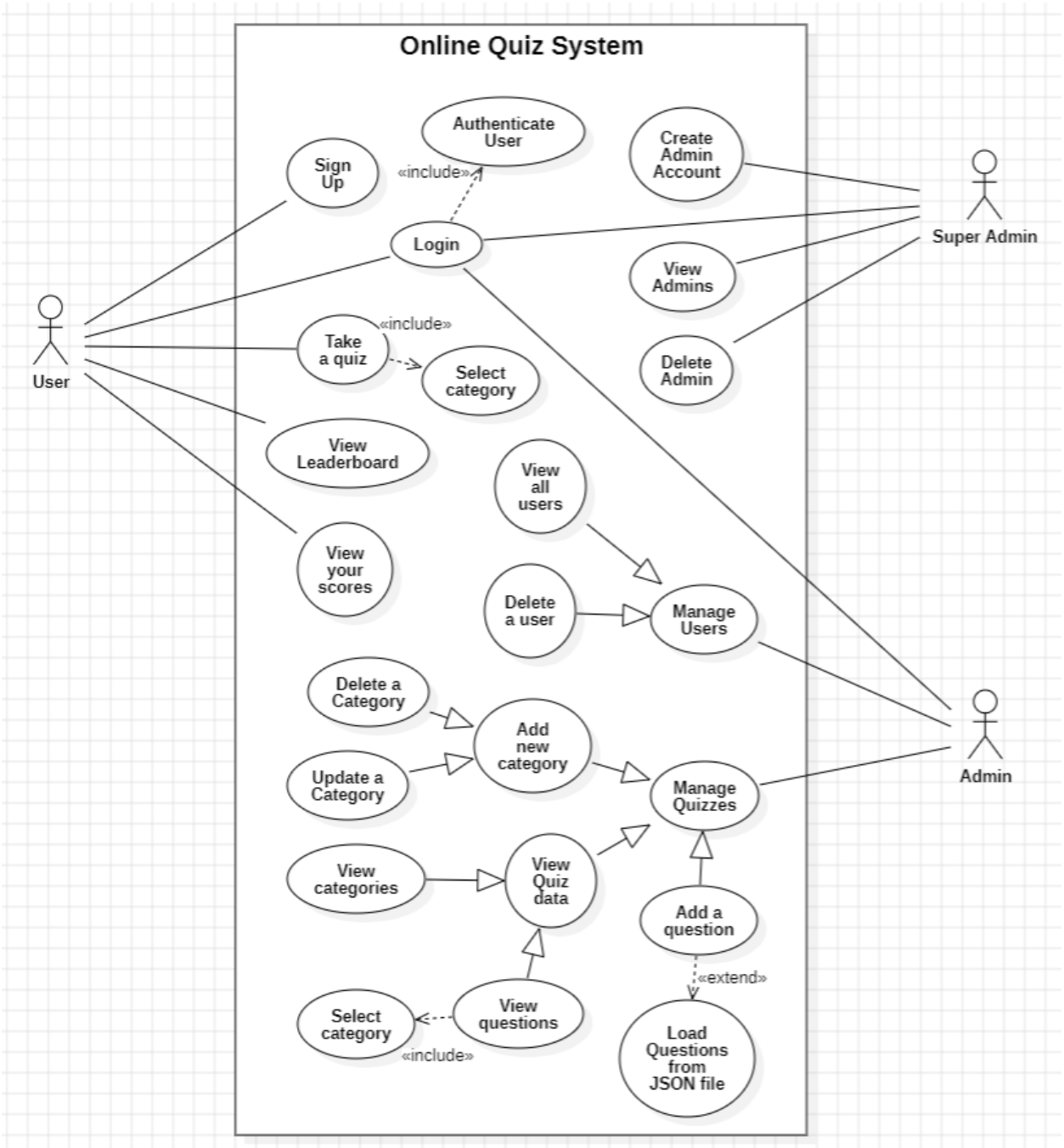
Stakeholders:

1. Internal Stakeholders
    - Developers
    - Mentors
2. External Stakeholders
    - Super Admin
    - Admin
    - Users (Quiz Takers)

Technologies Used:

- Python 3.11.5
- Libraries –
    - logging
    - sqlite3
    - hashlib
    - re (RegEx)
    - typing
    - pylint
    - shortuuid
    - maskpass
    - random-password-generator
    - tabulate
    - python-dotenv
- Git and GitHub

Use case Diagram



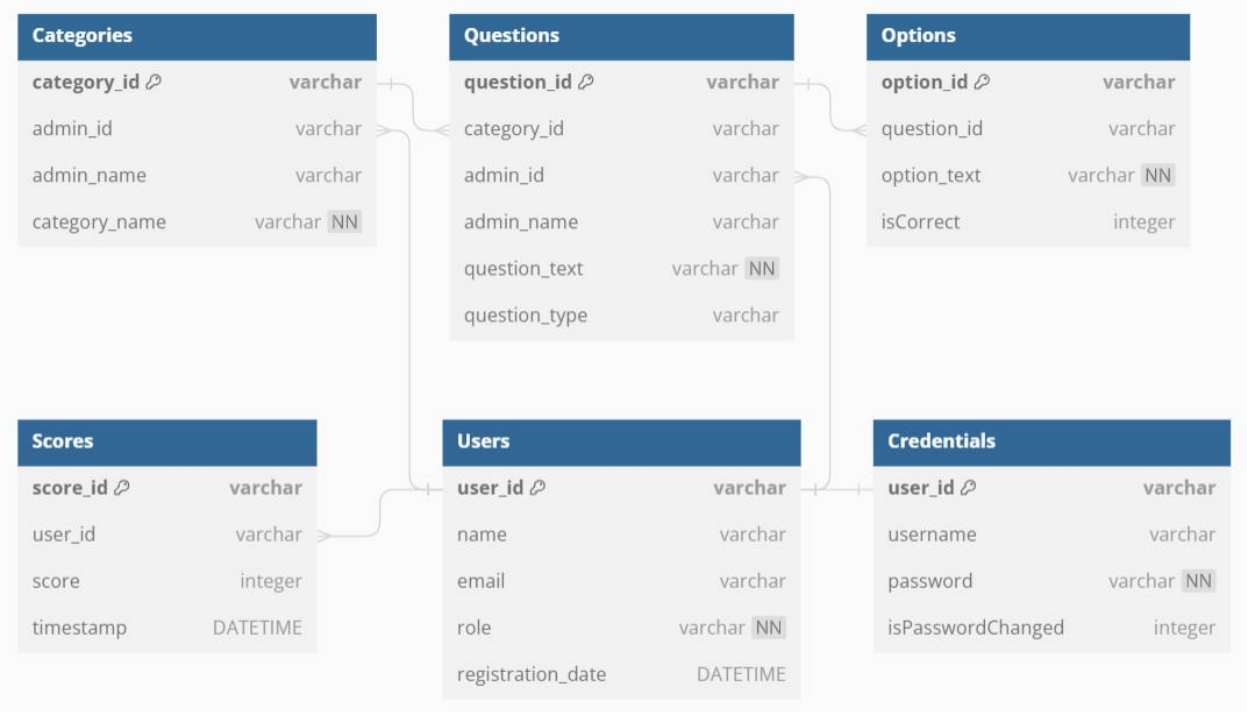
Actors & Use Cases

1. Super Admin –
    - Create an admin account
    - View admins
        - View All
        - View by admin id
    - Delete admin details

2. Admin –
    - Manage users
        - View user data
        - Delete user
    - Manage Quizzes
        - Manage Categories
            - CRUD on Categories
        - Manage Questions
            - View Questions by Category
            - View All Ques
            - Add a Question
            - Load Questions from JSON file

3. User –
    - Take a quiz
    - View leaderboard
    - View your scores

Database Schema



- One category has many questions
- One question has many options
- One user has many scores

Flow Diagram