Quiz Computation MVC Project Documentation

- ## Project Overview

  This project is a web application designed to allow users to take quizzes and receive immediate feedback on their performance. It will use the MVC architectural pattern to separate concerns and ensure scalability and maintainability.

- ## Functional Requirements

1. **User Management**
   - Sign Up: Users can register by providing a username, email, and password.
   - Login/Logout: Users can log in using their credentials and log out securely.
   - Profile Management: Users can view and update their profile information.

2. **Quiz Management**
   - Quiz Creation: Admin users can create and manage quizzes, including adding, updating, or deleting questions.
   - Quiz Listing: Users can view a list of available quizzes.
   - Quiz Taking: Users can select and take quizzes, answering multiple-choice questions.
   - Quiz Results: Users receive immediate feedback after submitting a quiz, including their score and correct answers.

3. **Scoring and Analytics**
   - Score Calculation: Automatically calculate the score based on the user's answers.
   - Result Storage: Store the results of each quiz attempt for future reference.
   - User Performance: Provide users with an overview of their performance across different quizzes.

- ## Database Structure

1. **Users Table**
   - `user_id` (Primary Key, Integer, Auto Increment)
   - `username` (VARCHAR)
   - `email` (VARCHAR)
   - `password_hash` (VARCHAR)
   - `created_at` (DATETIME)
   - `updated_at` (DATETIME)

2. **Quizzes Table**
   - `quiz_id` (Primary Key, Integer, Auto Increment)
   - `title` (VARCHAR)
   - `description` (TEXT)
   - `created_by` (Foreign Key, References `user_id`)
   - `created_at` (DATETIME)
   - `updated_at` (DATETIME)

3. **Questions Table**
   - `question_id` (Primary Key, Integer, Auto Increment)
   - `quiz_id` (Foreign Key, References `quiz_id`)
   - `question_text` (TEXT)
   - `created_at` (DATETIME)
   - `updated_at` (DATETIME)

4. **Options Table**
    - \`option_id\` (Primary Key, Integer, Auto Increment)
    - \`question_id\` (Foreign Key, References \`question_id\`)
    - \`option_text\` (TEXT)
    - \`is_correct\` (BOOLEAN)
    - \`created_at\` (DATETIME)
    - \`updated_at\` (DATETIME)

5. **User_Answers Table**
    - \`answer_id\` (Primary Key, Integer, Auto Increment)
    - \`user_id\` (Foreign Key, References \`user_id\`)
    - \`quiz_id\` (Foreign Key, References \`quiz_id\`)
    - \`question_id\` (Foreign Key, References \`question_id\`)
    - \`selected_option_id\` (Foreign Key, References \`option_id\`)
    - \`created_at\` (DATETIME)

6. **Results Table**
    - \`result_id\` (Primary Key, Integer, Auto Increment)
    - \`user_id\` (Foreign Key, References \`user_id\`)
    - \`quiz_id\` (Foreign Key, References \`quiz_id\`)
    - \`score\` (INTEGER)
    - \`created_at\` (DATETIME)

- **Workflow**

1. **User Registration and Login**
    - Users register and log in to the application.
    - Authentication is managed using secure password hashing.

2. **Creating and Managing Quizzes**
    - Admin users create quizzes by adding questions and options.
    - Questions can have multiple options, with one or more marked as correct.

3. **Taking a Quiz**
    - Users select a quiz to take from the list of available quizzes.
    - Users answer each question and submit the quiz.
    - The system calculates the score based on correct answers and stores the result.

4. **Viewing Results**
    - Users can view their quiz results immediately after submission.
    - Historical results are stored and can be viewed later.