**1. Tech Stack**

**1.1 Frontend:**

* **Streamlit**
  + **Use**: Streamlit is a Python-based web framework for building interactive applications. It is used to create the user interface of the quiz application, allowing users to log in, register, and take quizzes. Admins can also manage quizzes using the same interface.
  + **Features**: Streamlit provides a simple way to display forms, buttons, and select boxes. It handles user input and updates the UI based on the user’s actions.

**1.2 Backend:**

* **MongoDB**
  + **Use**: MongoDB is a NoSQL database used to store user details, quiz subjects, questions, and answers. Its flexibility makes it suitable for handling dynamic quiz data without requiring a rigid schema.
  + **Features**: MongoDB stores the user information (name, email, password), quiz subjects, and related questions. The admin panel allows CRUD (Create, Read, Update, Delete) operations on the quiz data.

**1.3 Security:**

* **Hashlib**
  + **Use**: Hashlib is used to hash user passwords before storing them in the database, ensuring password security.
  + **Features**: It provides a secure way to prevent password storage as plain text by converting passwords into cryptographic hash values.

**2. Features**

**2.1 User Functionality:**

* **User Registration**: Users can sign up with their name, email, and password. The password is securely hashed using hashlib before being stored in MongoDB.
* **Login**: Registered users can log in with their email and password. The system verifies the hashed password stored in the database.
* **Quiz Selection**: After logging in, users can choose from various subjects (e.g., Operating Systems, Data Structures, Machine Learning) and take the quiz.
* **Quiz Answering**: Each quiz consists of multiple-choice questions. The user's selected answers are stored temporarily during the session.
* **Score Calculation**: Once the quiz is submitted, the system checks the user’s answers against the correct ones and provides a final score, along with feedback on each question.

**2.2 Admin Functionality:**

* **Add/Edit Quizzes**: Admin users can create new quizzes or update existing ones by providing questions, choices, and the correct answer.
* **Delete Quizzes**: Admins can also delete quizzes for any subject.
* **Quiz Management**: Admins have access to CRUD operations to manage the quizzes for different subjects dynamically.

**3. Project Structure and Logic**

**3.1 User Registration and Authentication:**

* The user registration page allows new users to create accounts. The system checks if the email is already registered to avoid duplicates.
* During login, user credentials are checked against the database. The password is verified using a hashed version.

**3.2 Quiz Flow:**

* After logging in, the user selects a subject from a dropdown.
* Upon selecting a quiz subject, the corresponding quiz data is fetched from MongoDB.
* The quiz consists of multiple-choice questions with no pre-selected options. Users select answers and submit them.
* After submission, the correct answers are compared with the user’s answers to calculate the score, which is then displayed along with feedback for each question.

**3.3 Admin Section:**

* Admins can select a subject and provide a set of questions, choices, and correct answers.
* The admin panel allows updating or deleting a quiz for any subject.
* Admins can manage the entire quiz database through the CRUD interface provided by the application.

**4. Database Design (MongoDB)**

**Collections:**

1. **Users Collection**:
   * Fields: name, email, password, is\_admin
   * Purpose: Stores user details. The is\_admin field is used to identify admins.
2. **Quizzes Collection**:
   * Fields: subject, questions
   * Purpose: Stores quiz details. Each subject has multiple questions, and each question has a set of choices and a correct answer.
   * Example:

json

Copy code

{

"subject": "Operating Systems",

"questions": [

{

"question": "What is virtual memory?",

"choices": ["Physical memory", "An abstraction of memory", "Secondary storage", "None of the above"],

"answer": "An abstraction of memory"

}

]

}

**5. Key Challenges**

* **Session Management**: Streamlit’s session state is used to maintain user progress, ensuring that the user's login state and quiz answers persist as they navigate through the application.
* **Security**: Passwords are securely hashed before storage using hashlib to protect user credentials.
* **Dynamic Quiz Management**: Admins are given full control over quiz content, which is dynamically fetched and stored in MongoDB, allowing for flexible updates.

**6. Conclusion**

This project showcases a fully functional quiz application with login, registration, quiz-taking, and admin CRUD features. The combination of MongoDB and Streamlit enables efficient data storage and dynamic interaction, making it a scalable solution for educational or competitive quiz platforms.