### For the quiz app I will follow below approach. Backend Approach

* **API Design**: RESTful services to interact with the database. Users can fetch subjects, fetch questions by difficulty levels, and submit their answers.
* **Google OAuth**: Set Up Google API Credentials.
* **Database Design**: A relational database tables for subjects, questions, answers, levels, and users.
* **Question Selection Logic**: Fetch 4 beginner, 3 intermediate, and 3 professional questions per subject. Use a query that retrieves these based on the selected subject ID.
* **Answer Validation**: After the user submits answers, the backend will verify them against correct options and calculate a score.

### REST API <http://localhost/quizappbackend/login.php> : For Login <http://localhost/quizappbackend/api.php?action=get_subjects>: to get List of Subjects

### <http://localhost/quizappbackend/api.php?action=get_questions&subject_id=1> : To get questions ,options for answer and correct answer

**Frontend Approach (Vue.js)**

On the frontend side, using Vue.js for its component-based structure .

**Dynamic User Interface**: A clean UI where the user selects a subject, clicks "Start Quiz", and is navigated to the quiz page where questions are served one at a time.

* **State Management**: Managing quiz state (current question, selected answers, and score) as the user progresses through the quiz.
* **Routing**: Using Vue Router to navigate between the homepage, quiz page, and result page.

**Data Flow**

Data flow should work in the app:

1. **Subject Selection**: Fetch the available subjects and let the user pick one.
2. **Fetch Questions**: On selecting a subject, call the API to get questions based on the difficulty level.
3. **Display Questions**: Show one question at a time with four options, ensuring users can only proceed after selecting an answer.
4. **Score Calculation**: After completing 10 questions, submit the answers and calculate the score.
5. **Result Display**: Show the user’s performance in a simple and informative way.

**Database Design**

* **subjects:** Stores quiz subjects.
* **levels**: Stores difficulty levels (Beginner, Intermediate, Professional).
* **questions:** Stores quiz questions.
* **answers:** Stores possible answers and the correct one.