**OpenAI Workshop Student Project**

**Objective:** Web-based application for interactive exam generation and document-based learning.

**Features and Functionalities:**

**1. Question Generation**

* Generate 5 questions based on a topic provided by the user.
* Support the addition of difficulty levels: Basic, Intermediate, and Advanced.

**2. Answer Submission and Evaluation**

* Students submit their answers to the generated questions.
* Display results on a new page, including the correct answers and detailed explanations.

**3. Prompt Engineering**

* Optimize queries for generating questions.
* Include functionality to refresh questions based on a temperature value (controlled via a slider).

**4. Context Management**

* Maintain conversation context while generating and evaluating questions.

**5. Multiple Choice Questions (MCQs)**

* Generate structured outputs in JSON schema.
* Display MCQs with the ability to show correct answers and explanations.

**6. Retrieval-Augmented Generation (RAG)**

* Use courseware documents uploaded by users to generate contextually relevant questions.
* Implement file upload functionality and embedding generation for question preparation.

**7. Fine-Tuning**

* Train and fine-tune a model using the dataset of questions for a specific module.

**Technology Stack**

* Backend: Python, FastAPI
* Frontend: ReactJS
* Database: MySQL
* Vector Database: pgvector (PostgreSQL extension)
  + **Option1**: PostgreSQL (with pgvector) <https://supabase.com/pricing>
  + **Option2**: Amazon Relational Database Service (Amazon RDS) for PostgreSQL support the pgvector extension to store embeddings.
  + **Option3**: Amazon OpenSearch Service
* AI Integration: OpenAI API

**Technical Requirements:**

**Frontend (UI)**

* Topic selection.
* Temperature slider control.
* File upload and embedding initiation.
* Answer submission and result display.

**Backend**

* Integration with OpenAI API for question generation and fine-tuning.
* Embedding creation using courseware documents.
* Maintenance of conversation context.

**Data Storage**

* Store user-uploaded files and embeddings securely.
* Save submitted answers and evaluation results for review.

**Deployment**

* Cloud-hosted with scalable infrastructure.
* Secure access for students and instructors.

**Exercise 1:**

A screenshot of a computer

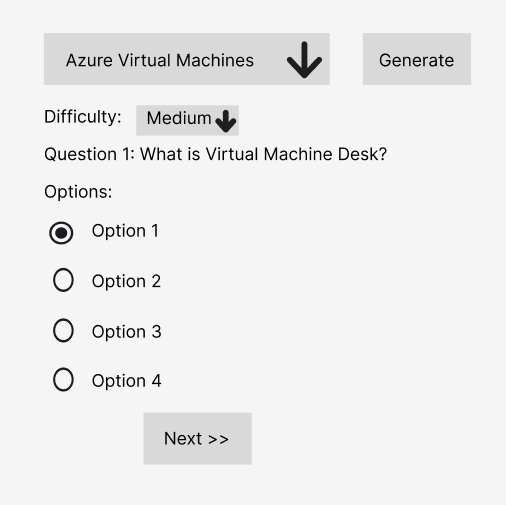
Description automatically generated

**Exercise 2:**

A screenshot of a computer

Description automatically generated

**Exercise 3:**



**File Upload:**

A screenshot of a computer

Description automatically generated

**Deliverables:**

1. A boilerplate starter application.
2. Documentation including:
   * Installation and deployment guide.
   * API references.
   * User manual for students and instructors.
3. Presentation demonstrating the application’s features.
4. Dataset prepared for fine-tuning

**Notes:**

* Students should have basic knowledge of python.