Coding Exercise: Document Management and RAG-based Q&A Application

Overview

Candidates are required to develop an application that integrates backend services with **Retrieval-Augmented Generation (RAG)** capabilities. The application should manage users, documents, and an ingestion process that generates embeddings for document retrieval in a **Q&A setting**.

To complete this assignment, candidates may use **mocking services, JSON, TF-IDF, BM25, Scikit-learn, or any other retrieval algorithm** to implement the required functionality.

Application Components

1. Python Backend (Document Ingestion and RAG-driven Q&A)

Objective:

Develop a **backend application in Python** that handles:

- **Document ingestion**
- **Embedding generation**
- **Retrieval-based Q&A (RAG)**

Key APIs:

- **Document Ingestion API:**
  - Accepts document data
  - Generates embeddings using a **Large Language Model (LLM) library**
  - Stores embeddings for future retrieval
- **Q&A API:**
  - Accepts user questions
  - Retrieves relevant document embeddings
  - Generates answers based on the retrieved content using **RAG**
- **Document Selection API:**
  - Allows users to specify **which documents** should be considered in the RAG-based Q&A process

---

2. Tools & Libraries (Choose Any)

Candidates may use one or more of the following tools/libraries:

LLM & Embedding Models

- **Ollama Llama 3.18B model**
- **LangChain**
- **LlamaIndex**
- **OpenAI API**
- **Hugging Face Transformers**

Storage & Databases

- **PostgreSQL (preferred) for storing embeddings**

Backend Optimization

- **Asynchronous programming** for efficient API request handling

---

Notes:

- The application should be designed for **scalability and efficiency**.
- Proper error handling and logging should be implemented.
- The choice of retrieval algorithms and embedding models should be justified.