**RAG Web Application Assignment**

**Objective:** To evaluate your ability to design, implement, and deploy a Retrieval-Augmented Generation (RAG) system with a user-friendly web interface. This assignment will assess your skills in data handling, vector database integration, prompt engineering, and system evaluation. **Instructions:**

1. This assignment must be completed **individually**. Collaboration or external assistance is strictly prohibited.
2. You are free to use any open-source libraries, frameworks, and tools to complete the assignment.
3. Document your code clearly with comments explaining the purpose and functionality of each component.
4. Submit a single ZIP file containing:
   * Source code for the RAG system and web application
   * A detailed report (as a PDF document) addressing the items listed in Part 4.
   * A README file with instructions on how to set up and run your application.

**Part 1: Data Understanding** You will be provided with a JSON file (CareerAdvisoryService.job\_profiles\_en\_100\_6thFeb.json). This file contains data related to various job profiles, including:

* **General Description:** Textual overview of the job role.
* **Day in the Life:** Description of typical daily activities.
* **Reasons Liked/Disliked:** Pros and cons of the job.
* **Preparation:** Education, training, and experience needed.
* **Aptitude, Interest, and Value Ratings:** Scores and reasons related to different attributes.
* **Career Pathways:** Potential career advancement opportunities.
* **Employer Information:** Details about companies hiring for this role.
* **Job Location Details:** Information about job availability and salary ranges in different regions of India.

The data includes diverse job roles such as "Data Entry Clerk" and "Phlebotomist". Each profile offers insights into required skills, education, and potential career paths. The dataset provides a basis for users to explore various career options based on different interests, skills, and geographical locations.

**Part 2: RAG System Implementation**

1. **Vector Database:** Choose a vector database of your choice to store embeddings of the job profile data.
2. **Data Ingestion:** Load the provided JSON data and pre-process/parse the text fields to create embeddings using a suitable embedding model.
3. **Indexing:** Build a vector index on the generated embeddings for efficient retrieval.
4. **Retrieval:** Implement a retrieval mechanism that takes a user query as input and retrieves the most relevant job profiles from the vector database based on semantic similarity.
5. **Generation:** Use a pre-trained language model (LLM) of your choice to generate an answer to the user's query, augmented with the information retrieved from the vector database.
6. **Prompt Engineering**: Design effective prompts that guide the LLM to provide informative and relevant answers based on the retrieved context, with effective guard rails.

**Part 3: Web Application Development**

1. **Frontend:** Create a simple web interface (using frameworks like Streamlit) that allows users to:
   * Enter a text-based query.
   * View the generated answer from the RAG system.
   * See the retrieved context/citations from the vector database.
2. **Backend:** Develop a backend API that connects the frontend to the RAG system.

**Part 4: Evaluation and Reporting**

Prepare a report (as a PDF document) that includes the following:

1. **Accuracy Analysis:**
   * Describe how you evaluated the accuracy of the answers generated by the RAG system.
   * Provide examples of queries and their corresponding answers, along with a subjective assessment of their accuracy and relevance.
   * Explain the metrics or methods you used to derive the accuracy (e.g., human evaluation, comparison against ground truth, etc.).
2. **Latency Measurement:**
   * Measure and report the average latency of the queries (i.e., the time it takes for the RAG system to generate an answer after receiving a query).
   * Analyse the factors that contribute to the latency and suggest potential optimizations.
3. **Cost Analysis:**
   * Estimate the cost of deploying your RAG service to a production environment (e.g., using cloud services like AWS, Azure, or GCP).
   * Consider the cost of infrastructure (e.g., servers, databases), LLM API usage, and other relevant factors.
   * Provide a breakdown of the estimated costs and justify your assumptions.
4. **Technology Stack**: List down the different open source softwares used to solve the problem and also their respective versions.

**Submission:**

Submit a ZIP file containing all the required components (source code, report, and README) to the designated submission platform by the deadline.

**Evaluation Criteria:**

Your submission will be evaluated based on the following criteria:

* Correctness and functionality of the RAG system and web application
* Effectiveness of the retrieval and generation mechanisms
* Quality of the prompt engineering
* Thoroughness and clarity of the evaluation report
* Code quality and documentation

Good luck!