Project - Develop a RAG Application Using Knowledge Graph and Vector Search

Goal:

Many RAG applications are built today using vector search. Still, the latest research shows that incorporating knowledge graphs can help improve the performance and quality of the Retrieval-Augmented Generation (RAG) application. You will develop an application that leverages Knowledge Graphs and vector search. This application will use 10,0000 Wikipedia articles as inputs and create a knowledge graph and vector search database for these articles. Then you will take a query as input, retrieve relevant information from the structured knowledge base you created, and generate accurate and contextually appropriate responses.

Project Expectations:

1. **Understanding RAG Architecture:**
    - Students are expected to understand the RAG model, including its components: retriever and generator.
    - They should learn how Knowledge Graphs can be used to enhance retrieval by providing structured, contextual information.
2. **Implementing Vector Search:**
    - Implement a vector search mechanism using tools such as FAISS, Annoy, Pinecone, or other vector search libraries to retrieve semantically relevant information from unstructured data sources, in this case, Wikipedia articles
    - Understand how to generate and index embeddings, chunking, and perform efficient similarity searches
3. **Integration of Knowledge Graphs:**
    - Integrate a Knowledge Graph into the RAG pipeline. You will use DBpedia, YAGO, or other knowledge graphs (KG) of your choice, and use SPARQL to query the KG.
    - You will incorporate retrieval using KG in your RAG pipeline.
4. **Integration of LLM for generation of responses:**
    - Use a publicly available LLM to generate responses from the data retrieved for a given query. While you can use a paid LLM such as OpenAI's models, you are also free to use free models such as Microsoft Phi, Smaller version of LLaMA, or others
5. **Application Development:**
    - Develop a functional application that uses the RAG framework to answer queries or generate content based on the retrieved data.
    - The application should be able to handle a variety of inputs, demonstrating the flexibility and effectiveness of combining Knowledge Graphs and vector search in a RAG setup.
6. **Evaluation and Optimization:**

    - Evaluate the performance of the RAG application in terms of accuracy, relevance, and response quality.
    - Optimize both the retriever (vector search) and generator components to improve the overall application performance.
7. **Documentation and Presentation:**
    - Maintain thorough documentation of the project, detailing the design decisions, implementation steps, challenges faced, and solutions developed.
    - Prepare a presentation or demonstration of the final application, showcasing its capabilities and explaining the underlying technical details.

Steps to Achieve the Goal:

1. **Initial Research and Learning:**
    - Study the RAG model architecture and its components (retriever and generator).
    - Learn about Knowledge Graphs: their creation, querying (using SPARQL, for instance), and integration into NLP tasks.
    - Understand vector search fundamentals and tools available for implementation.
2. **Data Collection and Knowledge Graph Construction:**
    - Gather the dataset for your application, in this case Wikipedia articles
    - Utilize an existing Knowledge Graph that can be used for retrieval purposes.
3. **Implementation of Vector Search:**
    - Generate embeddings for data points using pre-trained models (e.g., BERT).
    - Index the embeddings using a vector search library and implement the retrieval mechanism.
4. **Integration of Components:**
    - Connect the retriever (vector search) and integrate the Knowledge Graph to enhance the retrieval process.
    - Connect the output of the retriever with the generator (language model) to create structured responses. Ensure the generator has access to structured information.
5. **Application Development:**
    - Implement the logic for query processing, retrieval, and response generation.
    - Command line query input and piping the response to a text file is adequate for this project
6. **Testing and Optimization:**
    - Test the application with various inputs to assess its performance.
    - Refine the retrieval algorithms, tweak the Knowledge Graph integration, and adjust the generator model to optimize results.
7. **Documentation and Final Presentation:**
    - Document the entire process, including code, methodologies, and findings.
    - Prepare a final presentation showcasing the application, discussing the technical challenges, solutions, and potential future improvements.