Project System Design

Overview:

This project is an Article Retrieval System equipped with a question-answering mechanism. The system utilizes a gemma-7b-it language model, LangChain, and Chroma vector database. To enhance the efficiency of data retrieval, the content is initially segmented into chunks, embedded, and then stored in our vector database. This approach enables fast identification of relevant articles in response to our queries using similarity_search. The article search function encompasses variables that determine the number of relevant articles we wish to retrieve and the quantity of characters from each article to be utilized. The chars count restriction is implemented due to the limited scope of the model, which lacks a sufficiently large context window to comprehend entire articles. Moreover, the question-answering system employs an article_search function to find relevant articles fragments, which are subsequently utilized by the gemma model through a prompt. The model uses the provided data to create responses to given questions.

Technologies and their benefits:

- **Google Colab**: I used it because it makes it easy to describe different parts of the code and allows users to navigate the code smoothly. It also simplifies the process of preparing the environment and ensures that the code will work at the same speed for everyone when used with basic GPU Collab offers.
- **HuggingFace**: It provides a wide range of models to test and makes it easy to load and use them.
- **Gemma-7b-it**: After testing a few models, it performed the best at comprehending large prompts containing article fragments.
- **LangChain**: It provides access to easy-to-use functions that are helpful in data framing and chunking data.
- **ChromaDB**: It can store vectors with metadata and allows for similarity search queries. It is super easy to set up and use.

Challenges:

The biggest challenge was finding a model that would work quickly, comprehend long prompts, and provide sensible answers. Deciding on the best way to pass input to the model was also a dilemma. One approach was to use LLMChain with LangChain, utilizing a prompt template, while the other involved simply passing the article to the prompt. Ultimately, the simple second solution with the gemma-7b-it model worked quite well.

Potential Areas for Future Development:

- **Caching**: Implementing caching functionality to store previously generated answers for similar queries would enhance system efficiency and response time.
- **User-Friendly Interface**: Developing a chat-like interface to improve user interaction and accessibility.
- **Model Fine-Tuning**: Fine-tuning the gemma-7b-it model on the provided dataset could improve its performance and accuracy for specific tasks.
- **Utilizing Stronger Models**: Incorporating more powerful models such as GPT-4 and leveraging their APIs could enhance the system's overall speed and performance.