Task 8 Write-up: LangChain Integration with Symbolic Knowledge Base

For Task 8, I explored LangChain by integrating it with a symbolic Prolog-style knowledge base using a Retrieval-Augmented Generation (RAG) approach. The goal was to reimplement core reasoning behavior—similar to Logic-LM or LINC—using LangChain's tooling.

Overview

I created a small logical knowledge base containing:

- **Facts** such as `animal(sparrow)` and `flies(eagle)`
- **Rules** such as `bird(X) :- animal(X), flies(X)`

These were converted into LangChain-compatible `Document` objects and split using a text splitter. I used HuggingFace's `sentence-transformers` for embeddings and indexed the documents using FAISS. A retriever was then built to query the indexed knowledge base.

Testing

To test logical inference retrieval, I queried the system with natural language prompts like:

"Can a sparrow fly?"

The retriever successfully returned both relevant **facts** and **rules**, demonstrating symbolic logic retrieval in a lightweight RAG setup.

Technologies Used

- LangChain (Python)
- HuggingFace Embeddings
- FAISS vector store
- Google Colab (runtime environment)

Repository

GitHub Repo – Task 8 LangChain

All code, the notebook (`.ipynb`), and this write-up are included in the repository.

Conclusion

This implementation serves as a lightweight prototype for integrating symbolic reasoning with LLM pipelines using LangChain's RAG capabilities.