**Question answering chatbot -CPU**

1. Problem Statement

Design a LLM based question answering chatbot for enterprise/private data.

The application should take the question as input and generate the answer based on the knowledge provided in the document and personalize answer per the user question.

The model should be able to prevent hallucination.

2. Approach

This approach process of 2 steps:

1. Data Ingestion
2. LLM Based Answer Retrieval

**Data Ingestion Steps -**

1. Load and extract the txt file
2. Split the extracted data into smaller chunks
3. Embed the text chipmunks using any embedding model e.g.; hugging face instruct embeddings
4. Build and semantic index of each chunk
5. Ingest the index into a vector database as knowledge base

**LLM Based Answer Retrieval steps -**

1. Take the input from user with an UI (Gradio)
2. Embed the question using same embedding model which is used for data ingestion
3. Semantically search and retrieve the relevant text chunks using knowledge base (vector database)
4. Enhance the prompt using both question and retrieved doc
5. Call LLM model with enhanced prompt and generate the answer
6. Display the answer to user using UI (Gradio)

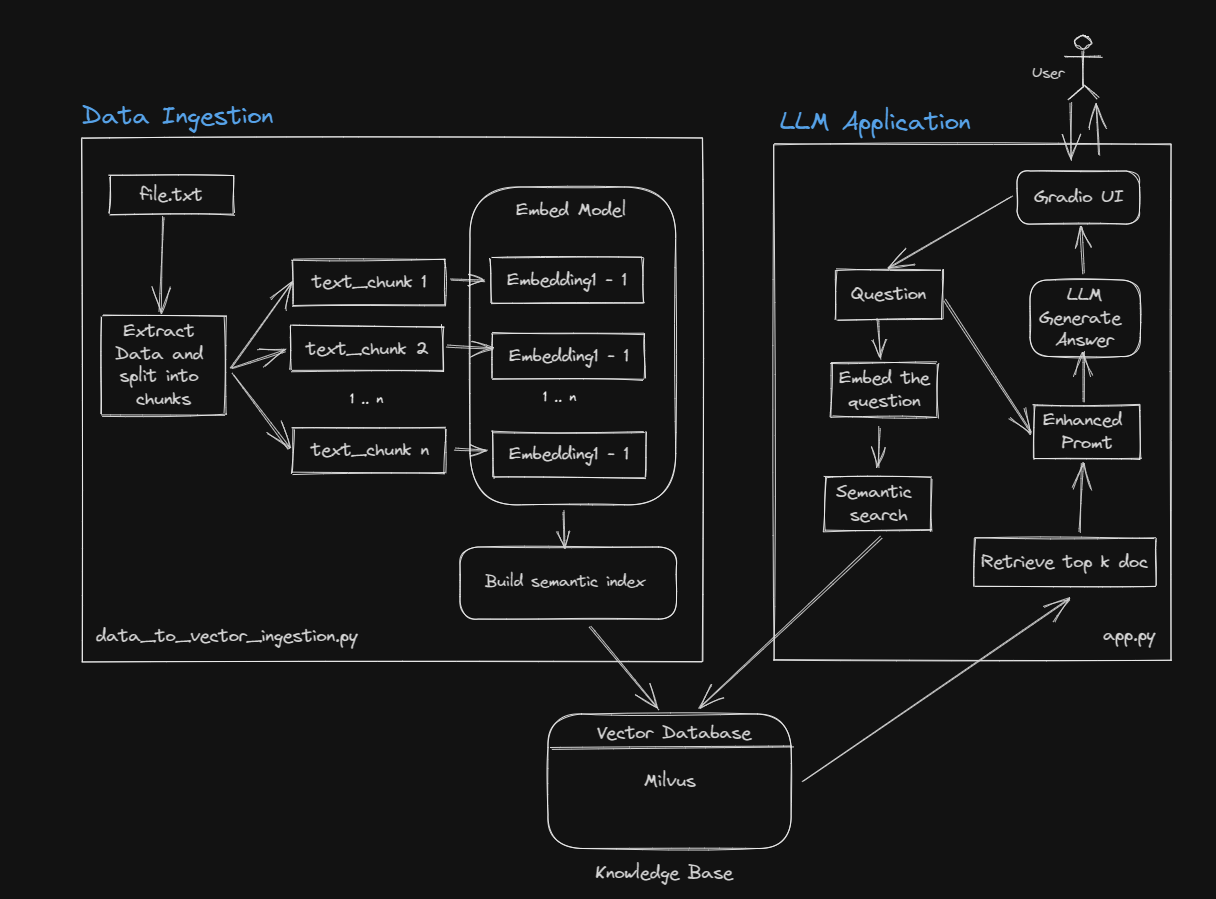
**Assumptions**

* No OpenAI LLM API call. Only using open source pretrained LLM.
* Model should run on CPU (because of compute constrain)

3. Solution

**Library used -**

* [LangChain](https://docs.langchain.com/docs/) - For LLM orchestration
* [Milvus](https://milvus.io/) Vector Database for embedding storage
* [InstructorEmbeddings](https://instructor-embedding.github.io/) - For vector embedding of docs
* [llamacpp](https://pypi.org/project/llamacpp/) - LLM model
* [Gradio](https://www.gradio.app/) - Web interface
* [W&B](https://docs.wandb.ai/guides/integrations/langchain) - For prompt/experiment tracking

**Application** [**Flow**](https://excalidraw.com/#json=vkTlm3fETi0mBZd37OZGT,eVLyX6wiEVp7EFEvoiSBAQ) **Diagram**  


**Performance Evaluation**

* Using weights and bias to keep track for all the prompts results and manually checking the output’s performance.
* Using [BertScore](https://huggingface.co/spaces/evaluate-metric/bertscore/blob/main/README.md) as a performance metric.

**Drawbacks for current model**

The performance of current model is not very great and the reason could be that this model is quantized model designed to run only on CPU therefore less number of learnt parameter and also the context window of this model is not large enough to pass all the relevant docs at once.

4. Future Scope

1. Use SOTA open-source model with larger context window (like falcon)
2. Finetune the model with instructions
3. Create one more layer of LLM to validate semantic search retrieval of docs.
4. Experiment with prompt template
5. Experiment with [guardrails](https://github.com/ShreyaR/guardrails) for edge cases
6. Experiment with chunk size of the doc and k number of retrieved documents for prompt.