RAG - Retrieval Augmented Generation

LLMs are trained only with data and LLMs are limited by the recency of data. How good the data is, that good is only LLM.

Also, lets say we have a chatbot and it has to respond only from the data we have with us, not something very generic

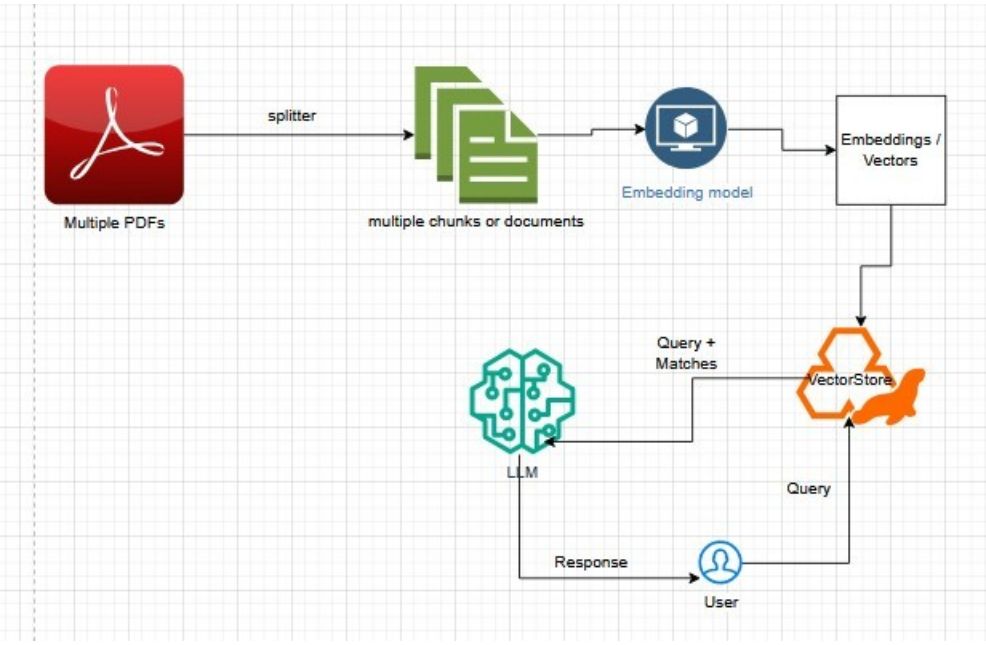
User ---> sends request ---> LLM

LLM is trained on some generic internet data but we want LLM to respond based on our data only
1. One option is to fine-tune LLM to make it respond in the way we want. We can train LLM more to respond to our data
2. Second option is send the entire dataset for every request but the problem is, we cant keep sending TBs of data for every request, it could consume bandwidth, computation, token cost. Most important is cost based on the tokens we are consuming from the data
3. Third option is, RAG. Reference documents could be PDF or docs, you dont want to send the entire private data to LLM (that's insecure)

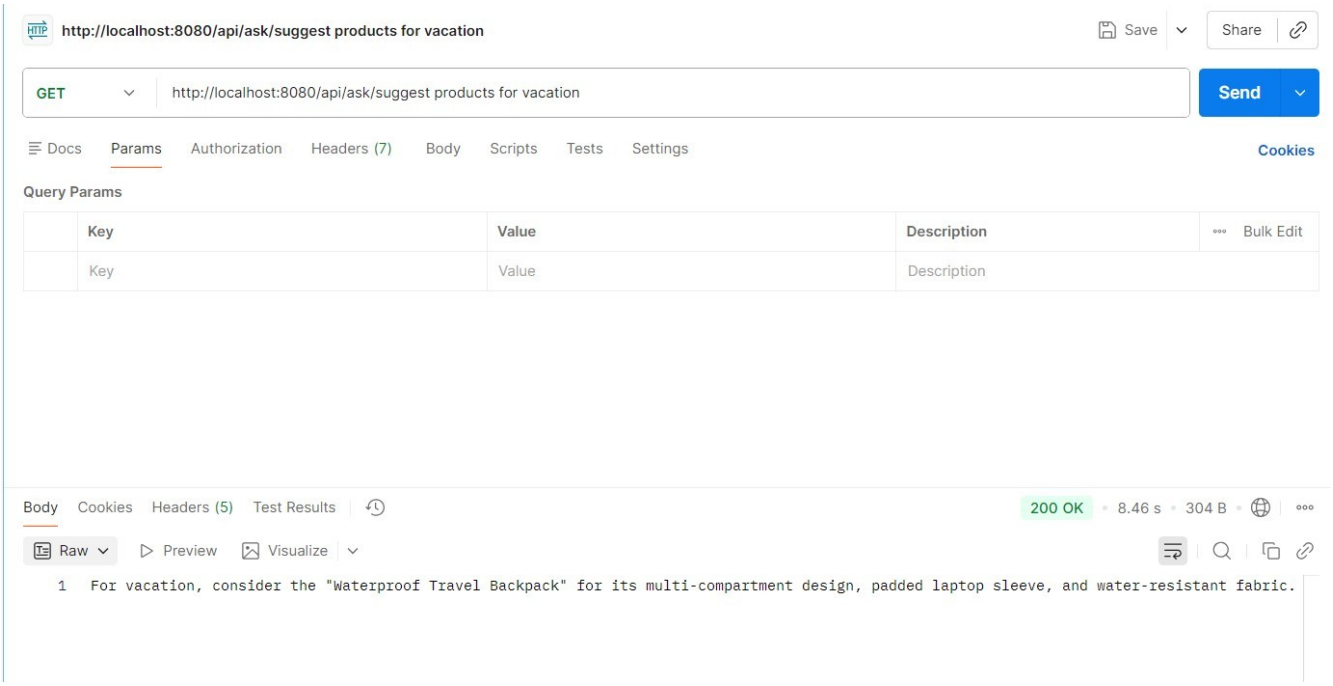
PDFs are converted into multiple chunks, then Embeddings and stored in VectorStore. When user sends a query it goes to VectorStore. VectorStore returns matches. So the User query + matches go into LLM and LLM responds to User finally.

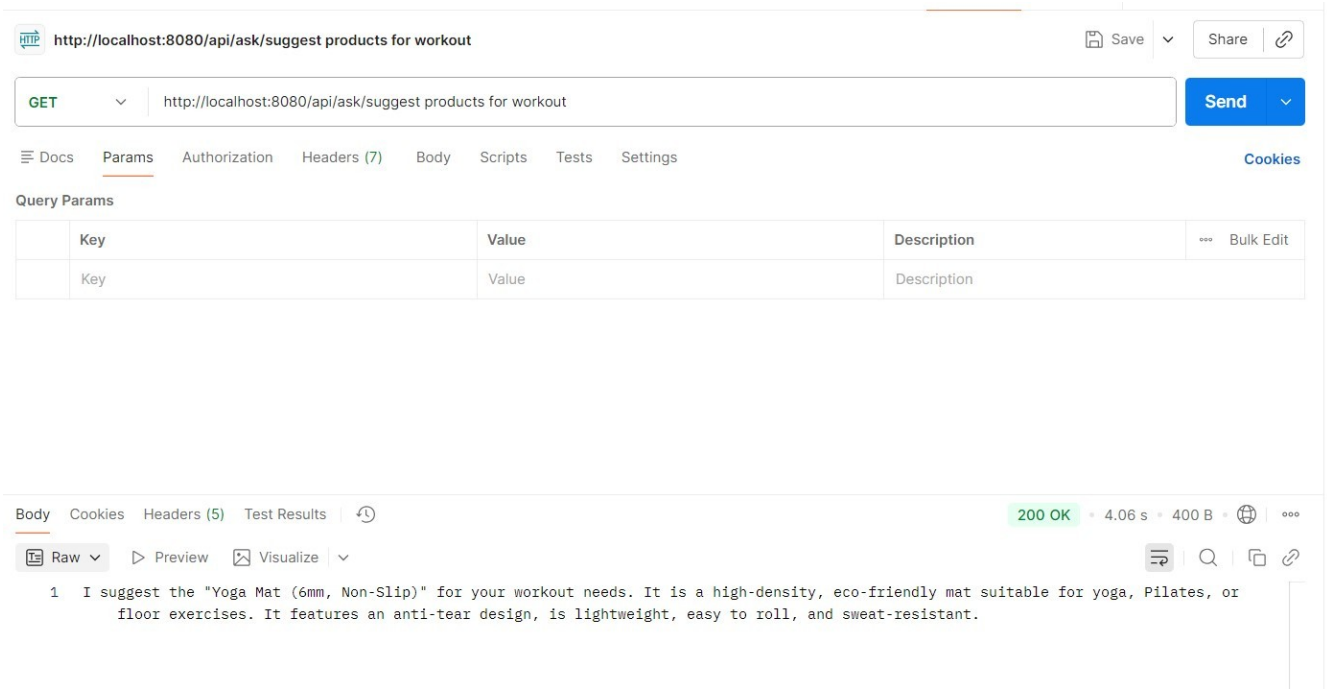
How do we include our documents into LLM?
PDF/docs ---> splitter ---> multiple chunks or documents ---> Embedding model ----> Embeddings/Vectors ---> VectorStore

User ---> Query ---> VectorStore (SimilaritySearch) ---> matches + query ---> LLM
LLM ---> responds ---> User
Check RAG_Concept pic

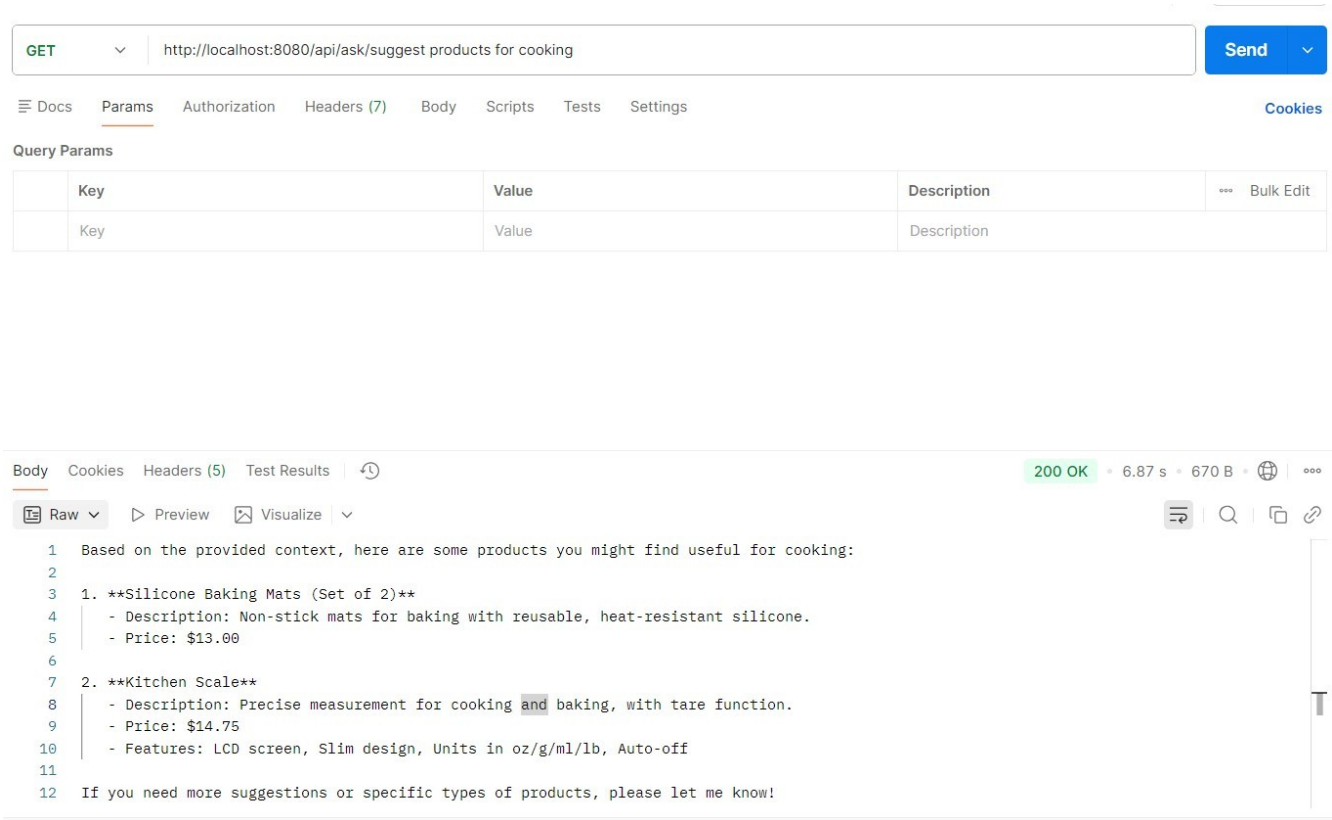


// you are talking to LLM but you are getting response based on your Product list
// for RAG





// with chat memory advisors
// GET http://localhost:8080/api/ask/suggest products for cooking



// After implementing PromptTemplate