Agentic RAG Using MCP

Problem statement

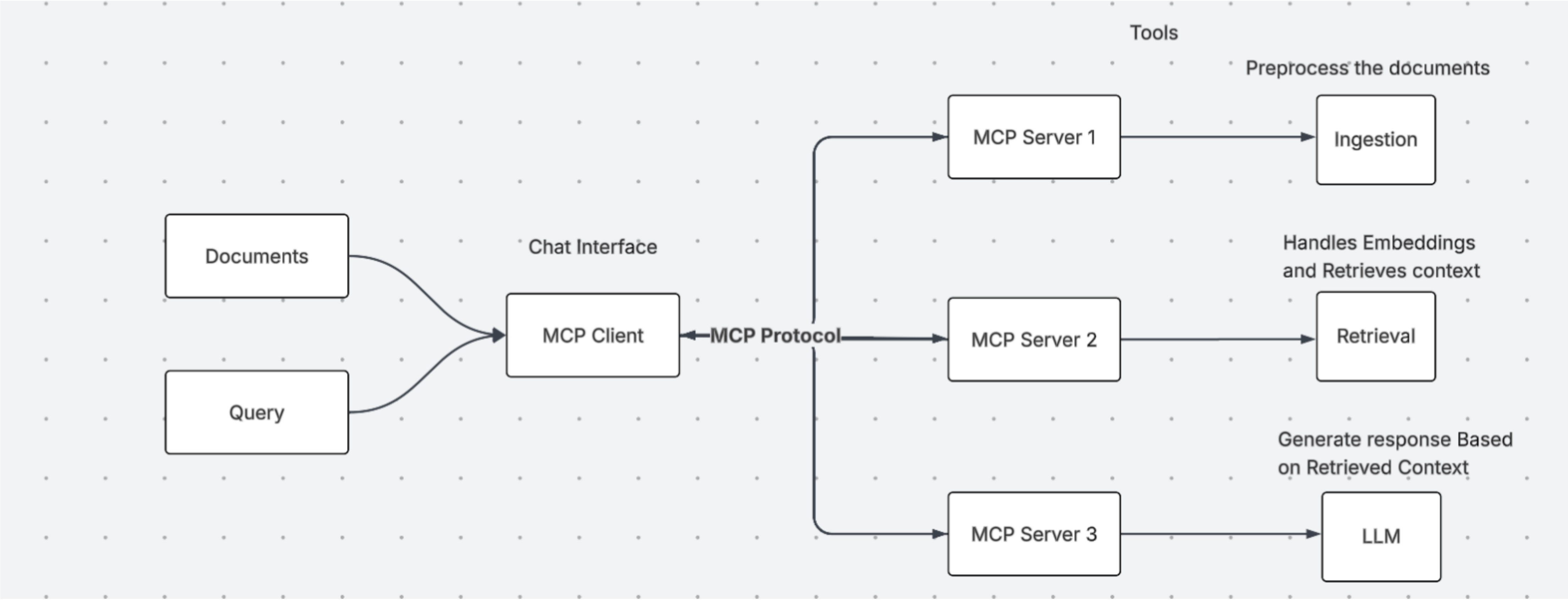
The Problem:

- Valuable information is often trapped within various document formats like PDFs, presentations, and spreadsheets.
- Manually searching through these documents is time-consuming and inefficient.
- Standard chatbots lack knowledge of private, domain-specific data, leading to generic or incorrect answers.

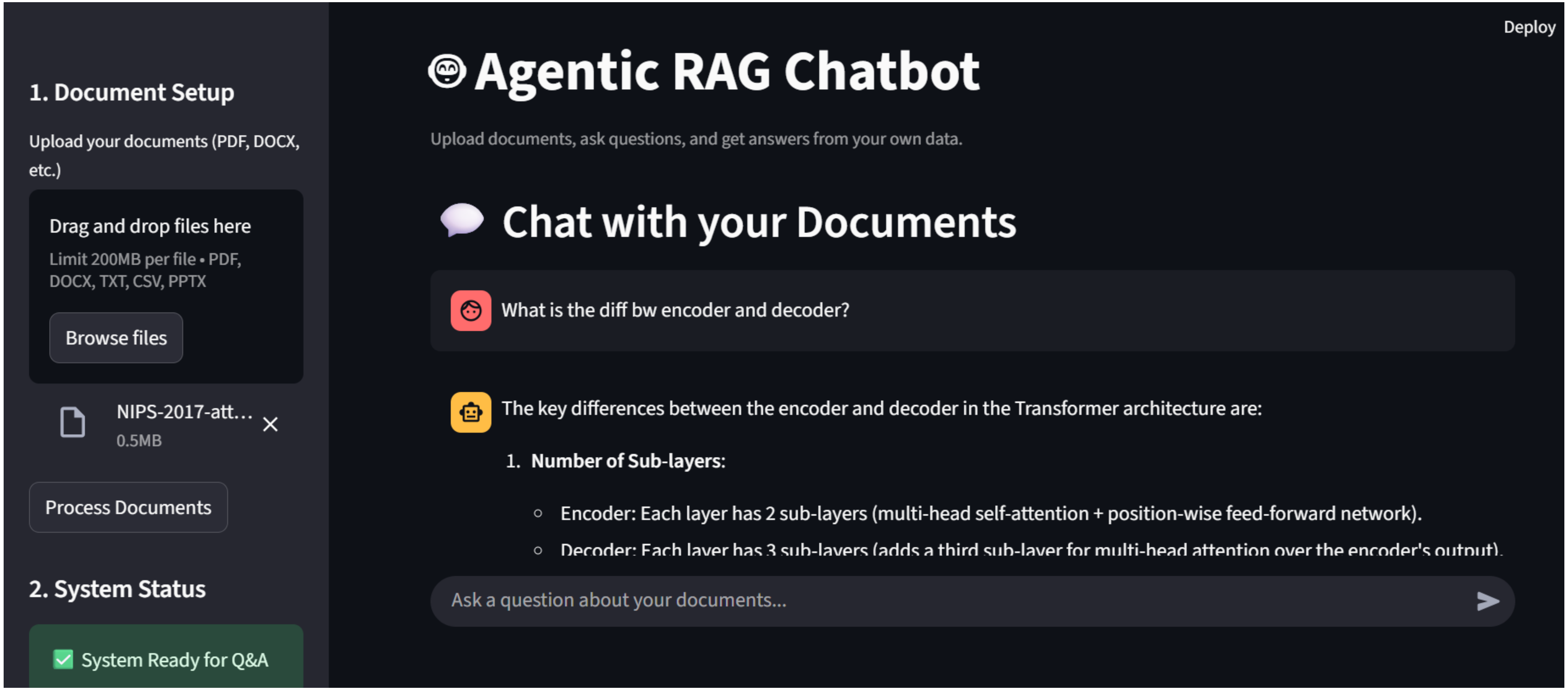
The Solution:

- Our goal is to build an agent-based chatbot that can read any document and provide factual, verifiable answers using RAG and the Model Context Protocol (MCP).

Architecture



Results


Deploy
1. Document Setup
Upload your documents (PDF, DOCX, etc.)
Drag and drop files here
Limit 200MB per file • PDF, DOCX, TXT, CSV, PPTX
Browse files
NIPS-2017-att...
0.5MB
Process Documents
2. System Status
System Ready for Q&A
Agentic RAG Chatbot
Upload documents, ask questions, and get answers from your own data.
Chat with your Documents
What is the diff bw encoder and decoder?
The key differences between the encoder and decoder in the Transformer architecture are:
1. Number of Sub-layers:
Encoder: Each layer has 2 sub-layers (multi-head self-attention + position-wise feed-forward network).
Ask a question about your documents...

Dep

What is encoder ans decoder?

Based on the context:

Encoder: The encoder is a stack of 6 identical layers. Each layer has two sub-layers:

1. A multi-head self-attention mechanism.
2. A position-wise fully connected feed-forward network. Residual connections and layer normalization are applied around each sub-layer. The encoder maps an input sequence to a sequence of continuous representations.

Decoder: The decoder is also a stack of 6 identical layers. Each layer has three sub-layers:

1. A multi-head self-attention mechanism (modified to prevent positions from attending to subsequent positions).
2. A multi-head attention mechanism over the encoder's output.
3. A position-wise fully connected feed-forward network. Like the encoder, residual connections and layer normalization are applied around each sub-layer. The decoder generates an output sequence one element at a time,

Retrieved contexts

Both share similarities like residual connections, layer normalization, and output dimensionality ((d_{model} = 512)).

View Sources

Source 1:

Here, the encoder maps an input sequence of symbol representations (x1,...,x n) to a sequence of continuous representations z = (z1,...,z n). Given z, the decoder then generates an output sequence (y1,...,y m) of symbols one element at a time. At each step the model is auto-regressive [9], consuming the previously generated symbols as additional input when generating the next. The Transformer follows this overall architecture using stacked self-attention and point-wise, fully connected layers for both the encoder and decoder, shown in the left and right halves of Figure 1, respectively. 3.1 Encoder and Decoder Stacks Encoder: The encoder is composed of a stack of N = 6 identical layers. Each layer has two sub-layers. The first is a multi-head self-attention mechanism, and the second is a simple, position- 2Figure 1: The Transformer - model architecture. wise fully connected feed-forward network. We employ a residual connection [10] around each of

Tech stack used:

- Python
- Streamlit
- mcp
- DeepSeek model
- Sentence Transformers
- FAISS
- LangChain
- And some other python libraries

Challenges faced

- Integrating asyncio-based MCP with Streamlit's synchronous script execution model.
- Debugging silent failures in agent subprocesses.