

Help MAte AI



Submitted by: Shashi Khurana

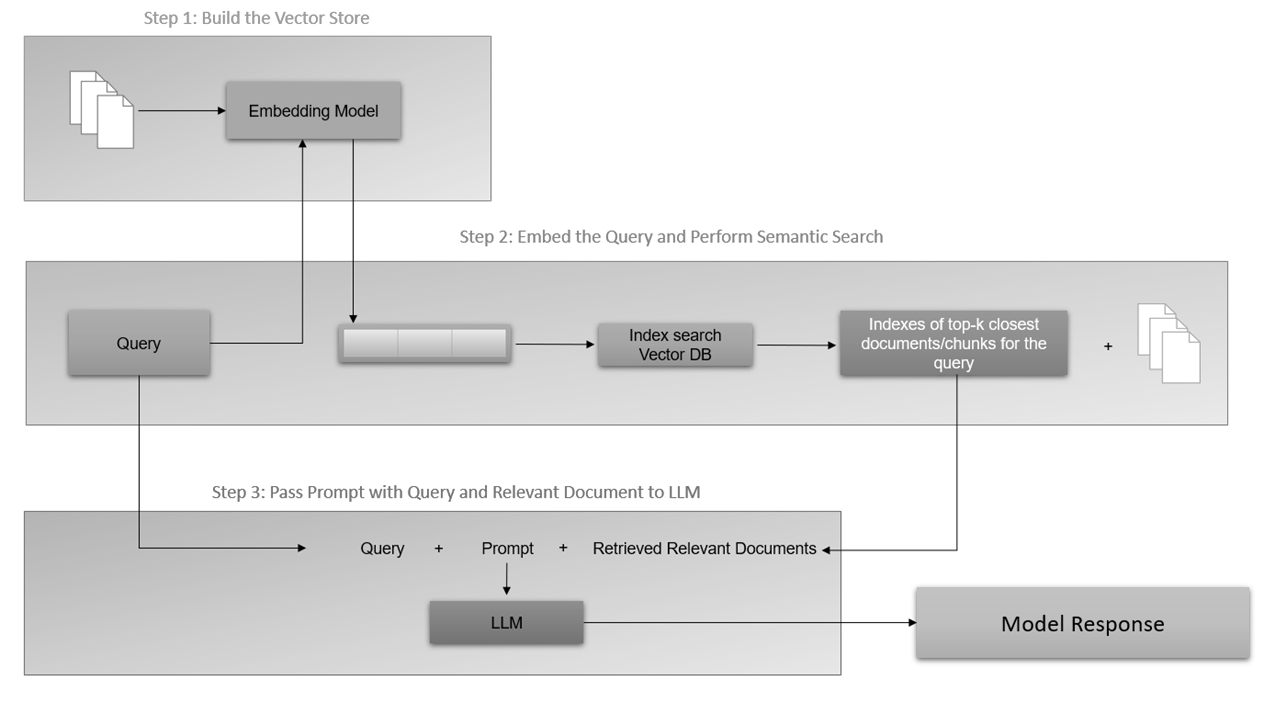
**Help Mate AI**

**Introduction**

The goal of the project is to build a robust generative search system that is capable for effectively answer question from one long pdf *document (*Principal-Sample-Life-Insurance-Policy.pdf). For this we will build the architecture as defined in the section below.

**Architecture**

The project should implement three layers effectively *a)* Embedding Layer, *b)* Search and Rank layer & *c)* Generation Layer. It will be key to try out various strategies and experiments in various layers in order to build an effective search system.

**

**Embedding Layer**

The embedding layer is typically the first layer of a RAG model, and it typically contains an embedding model that is trained on a massive data set of text and code. This data set is used to learn the relationships between words and phrases and to create embeddings that represent these relationships. The embedding layer is an important part of RAG models because it allows your system to understand the meaning of the text that it is processing and understand its semantic relationship to the query. The embedding layer generates embeddings for your text corpus and allows the RAG model to understand the meaning of the query and to generate a relevant and informative response. This is essential for a variety of tasks, such as question answering, summarization and machine translation.

**Search and Rank Layer**

The second layer is the search and rank or the re-rank layer. The search and re-rank layer *is crucial* component that is responsible for retrieving the relevant information from an external knowledge base, ranking it based on its relevance to the input query and presenting it to the generation layer for further processing. The search and re-rank layer *are* essential component of RAG, as it ensures that the retrieved text is accurate, relevant and contextually appropriate. The search and re-rank layer typically consist of two components:

1. A search component that uses various techniques to retrieve relevant documents from the knowledge base

The search component typically uses a technique called semantic similarity. As discussed in the previous session, semantic similarity is a measure of how similar two pieces of text are in terms of their meaning. The search component uses semantic similarity to retrieve documents from a knowledge base that are relevant to the user's query.

1. A re-rank component that uses a variety of techniques to re-rank the retrieved documents to produce the most relevant results

The re-rank component of the search typically uses a variety of techniques to re-rank the retrieved documents. These techniques can include the following:

* Ranking by relevance: The re-rank component can rank the retrieved documents based on how relevant they are to the user's query.
* Ranking by popularity: The re-rank component can rank the retrieved documents based on how popular they are, such as by measuring the number of times they have been viewed or shared.
* Ranking by freshness: The re-rank component can rank the retrieved documents based on how recent they are, such as by measuring the date on which they were published.

The search and re-rank layer is an important part of RAG models because it allows the model to retrieve and re-rank relevant documents from a knowledge base. This is essential for numerous tasks, such as question answering, summarization and machine translation. The search and re-rank layer is a powerful tool that can be used to improve the performance of a variety of AI tasks. It is an essential part of RAG models, and it plays a key role in helping these models retrieve and re-rank relevant information. The retrieval-based model is used to find relevant information from existing information sources. The re-rank layer is used to rank the retrieved information based on its relevance to the input query.

**Generation Layer**

The generation layer is typically the last layer of a RAG model which consists of a foundation large language model that is trained on a massive data set of text and code. As the name suggests, the generation layer allows the model to generate new text in response to a user's query. The generative model takes the retrieved information, synthesizes all the data and shapes it into a coherent and contextually appropriate response. This is essential for many tasks, such as question answering, summarization machine translation and generative search specifically RAG. In the context of search, this layer excels in providing context and natural language capabilities for generative search.

**RAG PIPELINE Steps**

**Step 1:** Build the vector store: The first step is to build a vector store that can store documents along with metadata. A vector store is a database that stores embeddings of text data in a vector space. The documents are converted to raw text and then split into chunks. Each chunk is then represented as a vector using an embedding model. The vector store is then populated with these vectors.

**Step 2:** Embed the query and perform semantic search: The next step is to embed the user query into the same vector space as the documents in the vector store. This is done using an embedding model. Once the query is embedded, a semantic search is performed to find the closest embedding from the vector store. The entries with the highest semantic overlap with the query are retrieved.

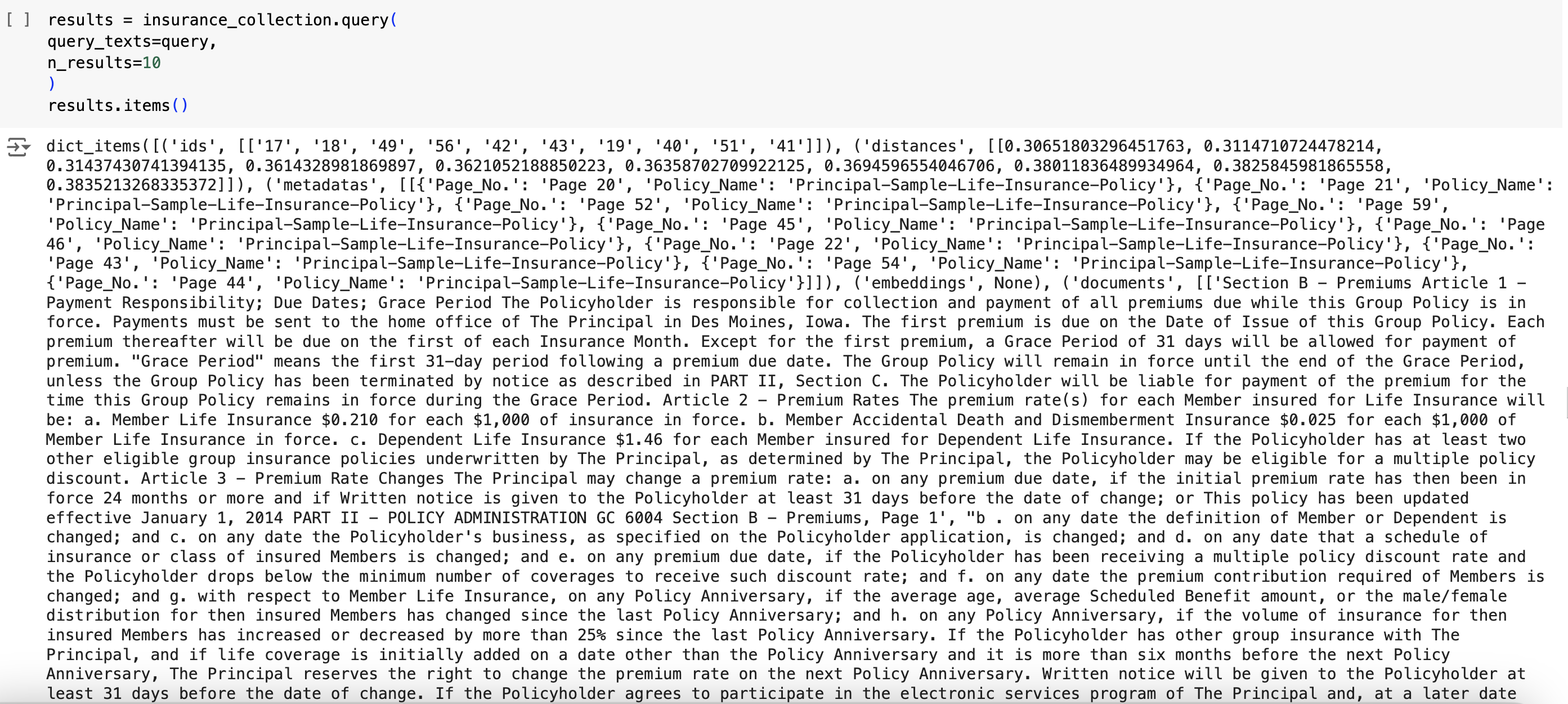
**Step 3**: Pass the prompt with the query and the relevant documents to the LLM: The final step is to pass the prompt, which is a concatenation of the query and the retrieved documents, to the LLM. The LLM generates a response based on the context of the query, the system prompt and the relevant documents passed from the search layer. The retrieved documents serve as the knowledge bank and provide the necessary context for the query to the LLM, which helps it generate a more accurate and relevant response.

**Top 3 Results from the Search Layer**: Sharing Search results against self-designed queries that clearly showcase the top 3 results/chunks retrieved from the search layer.

Below is the screenshot for user search query and its output.

**User Query** *A close-up of a white background

Description automatically generated*

**Search Result** **

**Top 3 Result based on rank**

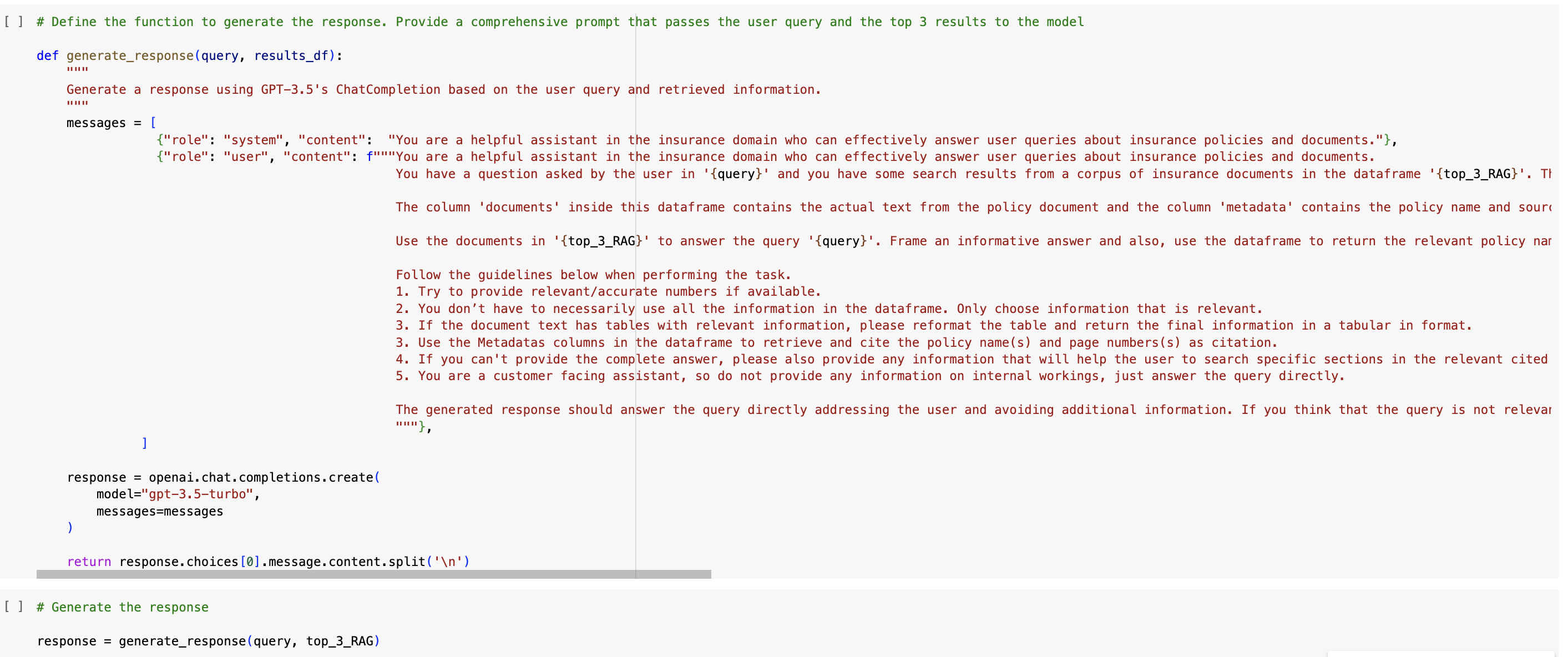
*A screenshot of a phone

Description automatically generated*

*A screenshot of a computer

Description automatically generated*

**Final Generated Answer from the Generation Layer**: Sharing screenshot of the same query above with the final output generated by the LLM in the generation layer.

**

LLM Output for the query

*A screenshot of a computer

Description automatically generated*

Below are the results with citations for the user query.

The premium amount for life insurance is not directly provided in the search results from the insurance documents provided. The detailed premium amounts specific to life insurance policies would typically be outlined in the premium section of the insurance policy document under a section related to premium calculation, payment schedules, and other associated details. To find the exact premium amount for life insurance, it is recommended to refer to the policy document directly by exploring the premium section of the document.

If you need to identify the premium amount for a specific life insurance policy, I suggest checking the premium calculation details and payment responsibilities outlined in the policy document. To locate this information, you can refer to the document's sections related to premiums, payment responsibility, or cost structures.

In case the premium amount details are not easily accessible, you may want to search for keywords like "premium amount," "cost of insurance," or "policy fees" within the document to pinpoint the relevant section that discusses the premium amount for life insurance.

---

\*\*Citations:\*\*

1. Policy Name: Principal Life Insurance Policy

- Page No.: Page 20

2. Policy Name: Principal Life Insurance Policy

- Page No.: Page 21

3. Policy Name: Principal Life Insurance Policy

- Page No.: Page 52

*Sample Results 2:*

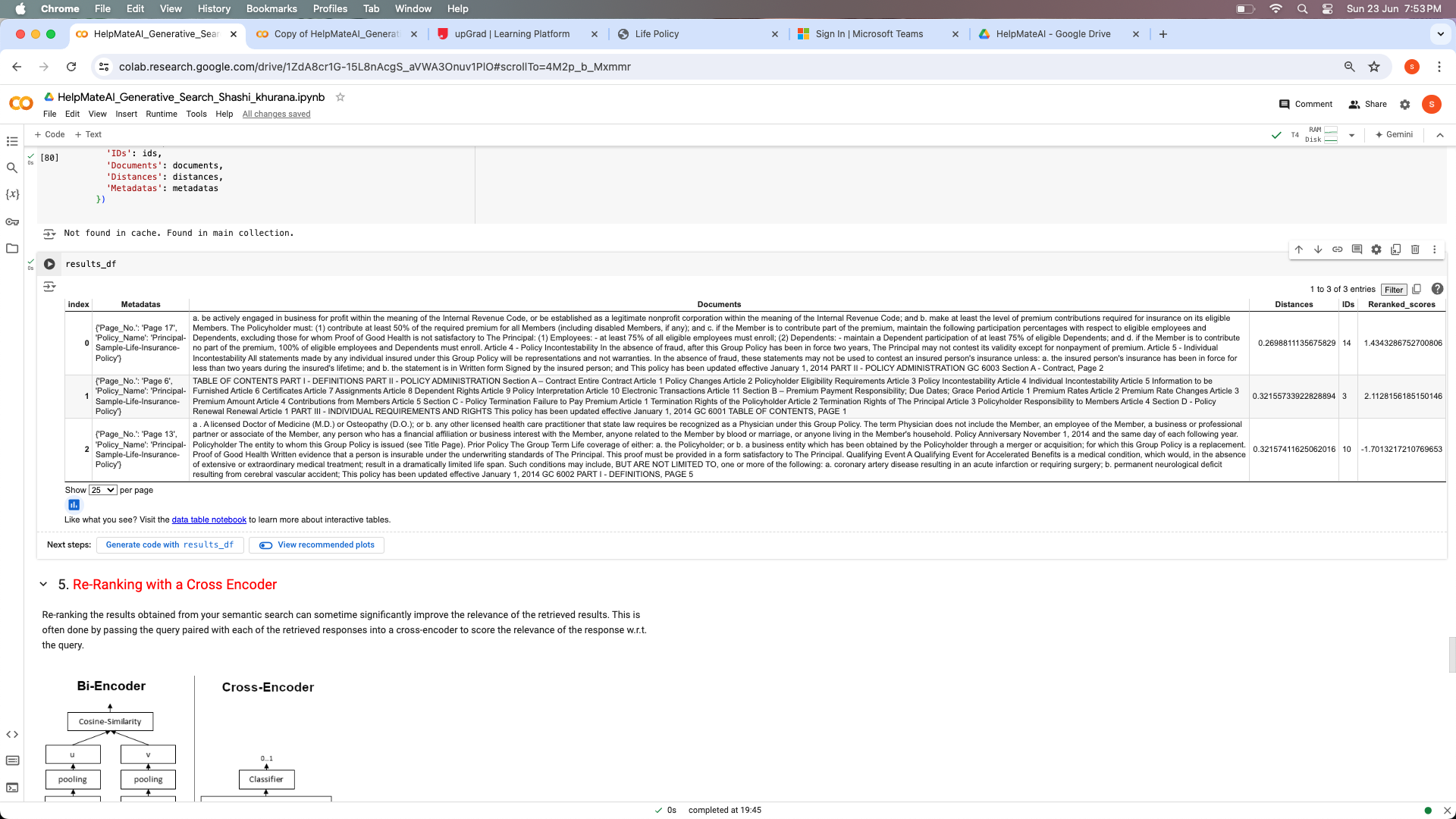
*See below the query, corresponding search results and finally LLM results (Screenshots attached in similar manner)*

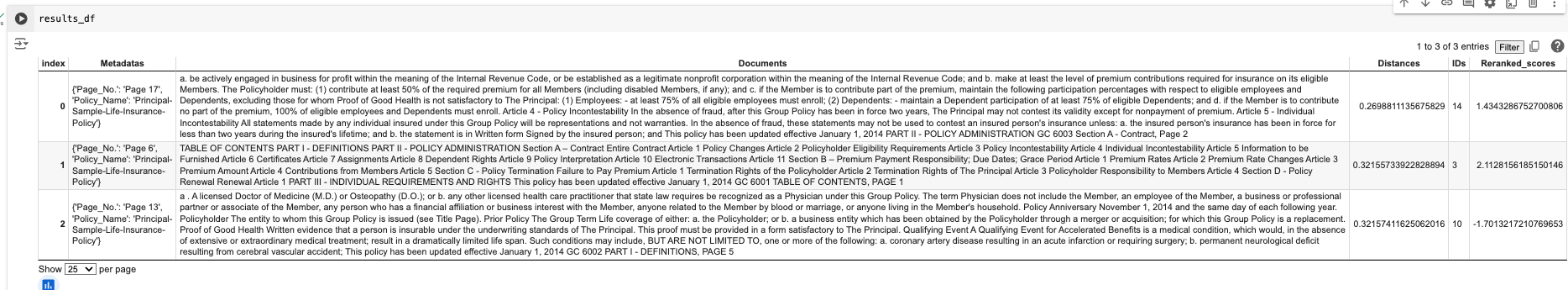
A close-up of a white background

Description automatically generated

A white brick wall with black text

Description automatically generated





A close-up of a text

Description automatically generated

A screenshot of a computer

Description automatically generated