HelpMateAI – Generative Search using RAG on Insurance Data

### Objective

The objective of this project is to build an efficient Generative Search system for insurance domain. A robust system capable of effectively and accurately answering questions from a policy document. We will be using a single long life insurance policy document (Principal-Sample-Life-Insurance-Policy.pdf) for this project.

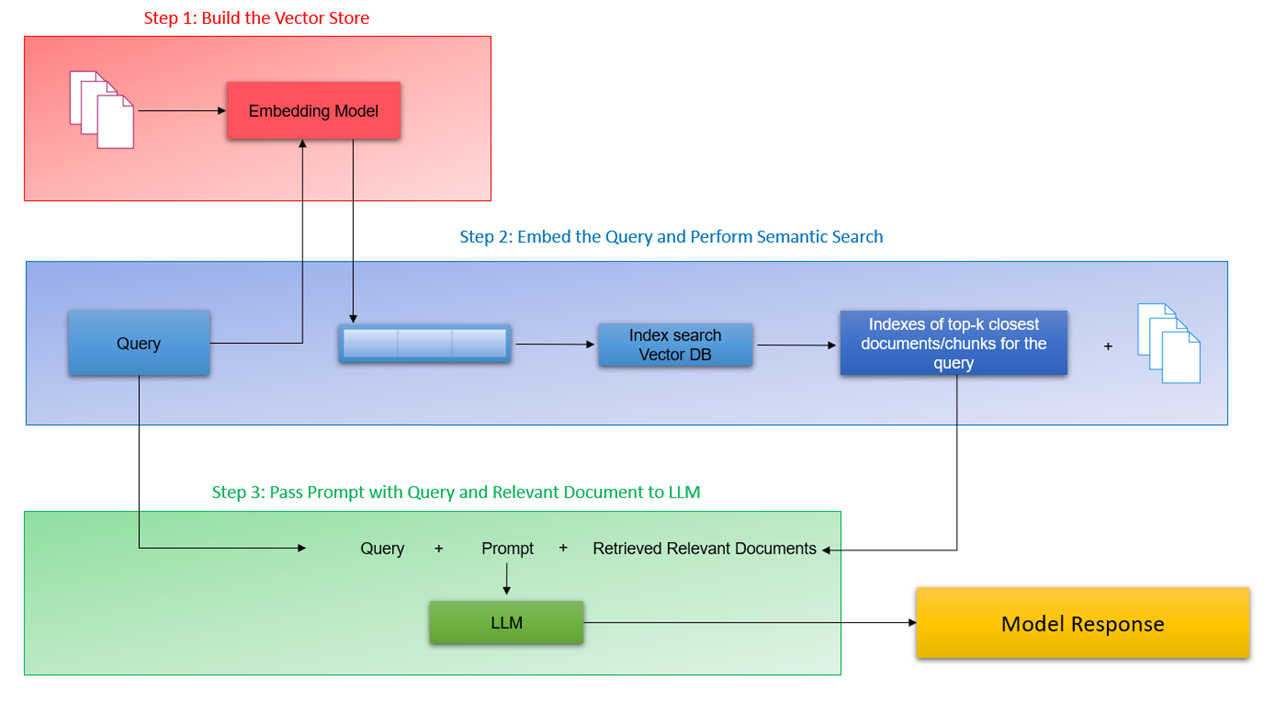
### System Design

The Generative AI search system has the following key layers.

1. **Embedding Layer:** The PDF document is effectively processed, cleaned, and chunked for the embeddings. The choice of the chunking strategy will have a large impact on the final quality of the retrieved results. Two chunking strategies – Page level chunking and Fixed size chunking are tried to verify the results.   
   Another important aspect in the embedding layer is the choice of the embedding model. We tried using OpenAI's *text-embedding-ada-002* model to generate embeddings. And *sentence-transformers/all-MiniLM-L6-v2* model from the [SentenceTransformers library](https://huggingface.co/sentence-transformers) on HuggingFace.
2. **Search Layer:** The search layer is built using ChromaDB. We will embed the chunks and store them in a ChromaDB collection. We will then perform a semantic search of a query in the collections embeddings to get several top semantically similar results.  A Cache layer is also implemented in Semantic Search, so that results can be fetched easily if a query indeed matches to a query in cache.

Re-Ranking is also implemented using Cross Encoder. Re-ranking the results obtained from semantic search can sometime significantly improve the relevance of the retrieved results. We tried two cross encoder models in CrossEncoder library from sentence\_transformers – *ms-marco-MiniLM-L6-v2* and *ms-marco-MiniLM-L12-v2*.

1. **Generation Layer:** In this layer, we pass the final top search results to OpenAI’s GPT 3.5 LLM along with the user query and a well-engineered prompt. To generate a direct answer to the query along with citations, rather than returning whole pages/chunks.



**Retrieval Augmented Generation - System Design**

Two experiments were conducted.

**Approach 1**

Chunking Strategy - Page Level Chunking

Embeddings - Using OpenAI's text-embedding-ada-002 model

Cross Encoder - Using cross-encoder ms-marco-MiniLM-L6-v2

**Approach 2**

Chunking Strategy - Fixed-Size Chunking

Embeddings - Default Embedding of Chroma DB (sentence-transformers/all-MiniLM-L6-v2 model)

Cross Encoder - Using cross-encoder ms-marco-MiniLM-L12-v2

**Major functions implemented in the System:**

* extract\_text\_from\_pdf(): Function to extract text from a PDF file.
* search\_vectordb(): Performs a semantic search of a query in the collections embeddings to get several top semantically similar results. Includes implementing Cache in Semantic Search.
* get\_rerank\_scores(): Utility function to fetch rerank scores using cross encoder.
* generate\_response(): Generates a response using GPT-3.5's ChatCompletion based on the user query and retrieved information (the top 3 results after reranking)

### Output

The following are the screenshots of results from the system. 3 screenshots against 3 self-designed queries that clearly showcase the top 3 results/chunks retrieved from the search layer. 3 screenshots of the same 3 queries with the final output generated by the LLM in the generation layer.

One observation is that the citations need improvement. The policy name returned by the LLM is not accurate, not picked the exact name from the metadata. Instead of relying on LLM response, we probably might have to use techniques like function calling to provide the citations exactly.

A screenshot of a computer

AI-generated content may be incorrect.

A close up of a text

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A close up of text

AI-generated content may be incorrect.