# Helpmate AI RAG Project

## Goals:

The goal of the project was to build a robust generative search system capable of effectively and accurately answering questions from a policy document.

## Sources:

Group member life insurance policy document provided by upGrad.

## Design Choices:

### Pre-processing:

Loaded and chunked a long life insurance policy document into manageable pieces, settled on a page-wise chunking strategy to optimize the embedding and search processes.

### Embedding Layer:

Implemented a process to clean and chunk the PDF document. Used the OpenAI's text-embedding-ada-002 model to embed the document.

### Search Layer:

Designed queries to be used for searching the document.Implemented a vector database (ChromaDB) to store and retrieve embeddings.

Developed a cache mechanism to improve search efficiency and reduce redundant computations.

Implemented a re-ranking mechanism using cross-encoding model ms-marco-MiniLM-L-6-v2 to enhance search result relevance.

**Generation Layer:**

Designed prompts for the large language model (LLM) GPT 3.5 to generate answers that are user friendly.

Tested the system with self-designed queries to evaluate the performance of the search and generation layers.

## Challenges:

**IndexError During Cache Implementation:**

Faced an IndexError due to attempting to access elements beyond the available range in lists. Resolved by adding checks to ensure the loop iterates only up to the length of the list.

**Kernel Crashes:**

Kernel repeatedly crashed when attempting to add large batches of documents to the ChromaDB collection.
Resolved by splitting the document addition process into smaller batches to manage memory usage effectively.

**PersistentClient Configuration:**

Encountered issues with configuring the PersistentClient for ChromaDB.
Resolved by ensuring the correct path was provided and initializing the client with the correct parameters.

**Module Import Errors:**

Faced errors related to importing CrossEncoder from sentence_transformers due to Keras version incompatibility.
Resolved by installing the compatible version of Keras using pip install keras==2.4.3.

**Output Formatting:**

Faced challenges with formatting the output for screenshots, ensuring the complete wording of documents was visible.
Resolved by adjusting the display settings and printing methods to ensure full visibility of the content.