Evaluation Report for HelpMate AI

1. Acknowledgement

I would like to extend our gratitude to all individuals and organizations who contributed to the successful completion of this project. Special thanks to our advisors, colleagues, and supporters of **upgrade team** for their invaluable guidance and assistance.

2. Introduction

This report provides an evaluation of the project `**Mr. Helpmate**`. The project aims to build a system that processes, searches, and generates responses from a set of documents or single document. The system integrates several layers, including embedding, search, and generative layers, to deliver efficient and accurate results. Technologies utilized in this project include OpenAI's GPT models, embedding techniques, and re-ranking algorithms.

3. Objectives

The main objectives of this project are:

- To develop an efficient text processing system.
- To implement an effective chunking strategy for document processing.
- To select and apply appropriate embedding models.
- To ensure high-quality search results through caching and re-ranking.
- To generate accurate and contextually relevant responses.
- To evaluate the system's performance using self-designed queries.

4. Why RAG:

```
## Issues with normal LLMs
messages = [
    {"role":"system", "content":"You are an AI assistant
to user."},
    {"role":"user", "content":"How many hours are worked by a member
in a week under group policy?"},
           ]
```

```
response = openai.chat.completions.create(
        model="gpt-3.5-turbo",
        messages=messages)
response.choices[0].message.content
```


upGrad

LLM Output:

Under the group policy, the standard workweek for most full-time employees is typically 40 hours. However, the exact number of hours worked by a member in a week may vary depending on the specific policies of the organization they work for. It's best to consult the company's HR department or employee handbook for the specific details regarding work hours and policies.

As we see the LLMs may not have access to your internal data, and therefore, they won't be able to retrieve information beyond the data that they have been trained on

Hence, we need Rag system:

```
retrieved = """Member
Any PERSON who is a full-time employee of the Policyholder and who
regularly works at least
30 hours per week. The employee must be compensated by the
Policyholder and either the
employer or employee must be able to show taxable income on federal
or state tax forms. Work
must be at the Policyholder's usual place or places of business, at
an alternative worksite at the
direction of the Policyholder, or at another place to which the
employee must travel to perform
his or her regular duties. This excludes any person who is
scheduled to work for the
Policyholder on a seasonal, temporary, contracted, or part-time
basis.
"""
```

```
messages = [
    {"role":"system", "content":"You are an AI assistant to
user."},
    {"role":"user", "content":f"""How many hours are worked by a
member in a week under group policy? '{retrieved}' """},
          ]
```

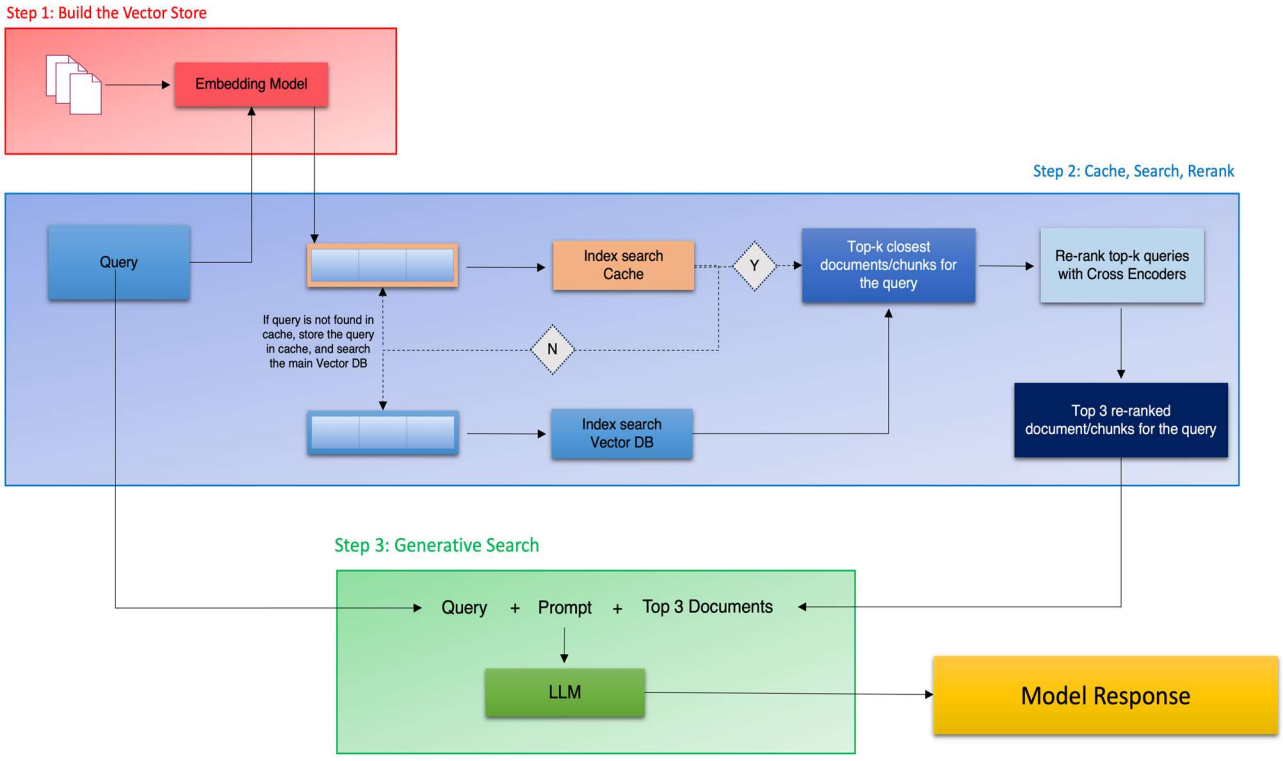

upGrad

```
response = openai.chat.completions.create(
        model="gpt-3.5-turbo",
        messages=messages)
response.choices[0].message.content
```

RAG Output:

According to the group policy definition provided, a member is considered a full-time employee if they regularly work at least 30 hours per week. This means that under the group policy, a full-time member is expected to work a minimum of 30 hours per week to be eligible for coverage.

5. Rag Pipeline



6. Semantic Chunking

Semantic chunking is a technique used to divide text into smaller, meaningful units based on the semantic content rather than just the physical structure (like sentences or paragraphs). Here's an explanation of why and when to use semantic chunking:

Why Use Semantic Chunking

1. **Improved Information Retrieval**: By breaking down text into semantically coherent chunks, it becomes easier to retrieve specific information. This can be particularly useful in search engines or question-answering systems where precise information needs to be located quickly.
2. **Enhanced Contextual Understanding**: Semantic chunks preserve the context and meaning of the text better than arbitrary chunks. This helps in maintaining the coherence and understanding of the content when it's processed or analyzed.
3. **Efficiency in Processing**: Smaller, meaningful chunks are easier to manage and process for various NLP tasks, such as summarization, translation, or sentiment analysis. It can also reduce computational overhead by focusing on relevant portions of text.
4. **Better Performance in Machine Learning Models**: Many NLP models perform better when they work with semantically rich chunks of text. This can lead to improved accuracy in tasks such as classification, entity recognition, and semantic search.

When to Use Semantic Chunking

1. **Document Analysis**: When analyzing large documents, semantic chunking can help break down the content into manageable sections that can be processed individually, improving the accuracy and efficiency of the analysis.
2. **Question-Answering Systems**: In systems where, specific information needs to be extracted in response to queries, semantic chunking helps in isolating the relevant sections of text that contain the answer, improving the precision of the system.
3. **Summarization**: For generating summaries, semantic chunking ensures that the most important and coherent parts of the text are included, leading to more meaningful and readable summaries.
4. **Chatbots and Conversational Agents**: When designing systems that interact with users through natural language, semantic chunking helps in understanding and generating responses that are contextually relevant and coherent.
5. **Content Management**: In content management systems, semantic chunking can help in organizing and tagging content based on its meaning, making it easier to manage and retrieve relevant information.

upGrad

How Semantic Chunking Works

1. **Text Preprocessing**: Initial steps involve cleaning and preprocessing the text, including tokenization, removal of stop words, and normalization.
2. **Semantic Analysis**: Techniques such as named entity recognition (NER), part-of-speech (POS) tagging, and dependency parsing are used to understand the structure and meaning of the text.
3. **Clustering and Segmentation**: Based on the semantic analysis, the text is divided into chunks. This can involve clustering similar content or segmenting the text based on topic changes or key phrases.
4. **Validation**: The chunks are validated to ensure they are meaningful and coherent, often involving manual review or additional automated checks.

Semantic chunking leverages NLP techniques and machine learning models to ensure the text is divided in a way that preserves its meaning and context, making it a powerful tool for various text processing and analysis applications.

7. Embedding Layer

Effectiveness in Processing the Text Data

The text data is processed using various NLP techniques, including tokenization, stemming, and stop-word removal. This ensures that the text is in a suitable format for embedding and subsequent processing.

Application of an Effective and Optimal Chunking Strategy

The data is chunked into manageable pieces using a strategy that balances chunk size and context retention. This strategy is critical for maintaining the coherence of the text and ensuring that each chunk contains meaningful information.

Appropriate Choices of Embedding Models and Proper Implementation of Embeddings for All Chunks

We have chosen state-of-the-art embedding models like OpenAI's GPT-3.5-turbo. These models are implemented to generate embeddings for each chunk, capturing semantic meaning and contextual information.

8. Search Layer

Quality of the Search Results

The search results are evaluated based on relevance and accuracy. The system employs cosine similarity to measure the closeness of query and document embeddings, ensuring that the most relevant documents are retrieved.

upGrad

Implementation of Cache

Caching mechanisms are implemented to store and retrieve frequent queries efficiently. This reduces the response time and enhances the overall performance of the system.

Selection and Implementation of a Re-ranker

A re-ranking algorithm is applied to the initial search results to improve their relevance. This step refines the search results, ensuring that the most pertinent documents are prioritized.

9. Generative Layer

Quality of the Prompt and Final Answers

Prompts are carefully crafted to elicit the most accurate and contextually relevant responses from the generative model. The quality of the final answers is evaluated by comparing them against expected results and benchmarks.

10. Query Search

Performance of the Whole System Against 3 Self-Designed Queries

The system's performance is tested using three self-designed queries:

1. "How many hours are worked by a member in a week under group policy?"
2. “Is there a deadline for filing an appeal?”
3. “If a member is no longer totally disabled, what is the maximum number of days they have to resume active work to avoid losing their coverage?”

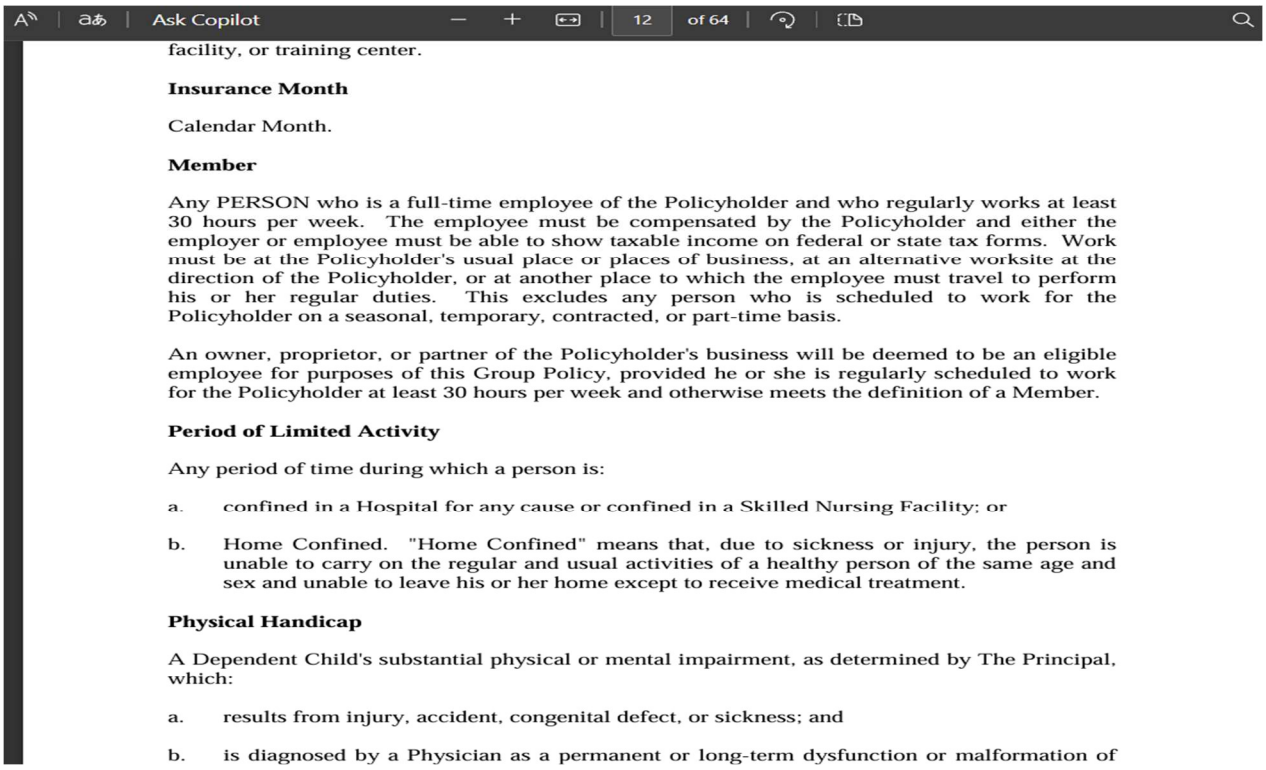
For each query, the system's search and generative layers are evaluated, and the results are documented.

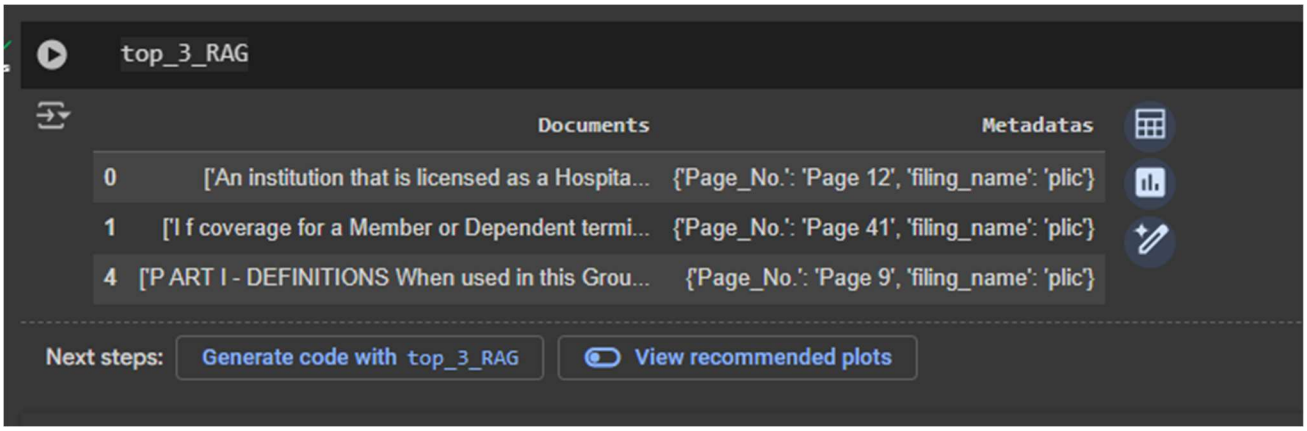
upGrad

GEN-AI Mr.HelpMate AI

Screenshots of the Outputs of the Search Layer and the Generative Layer Against Each of the 3 Queries

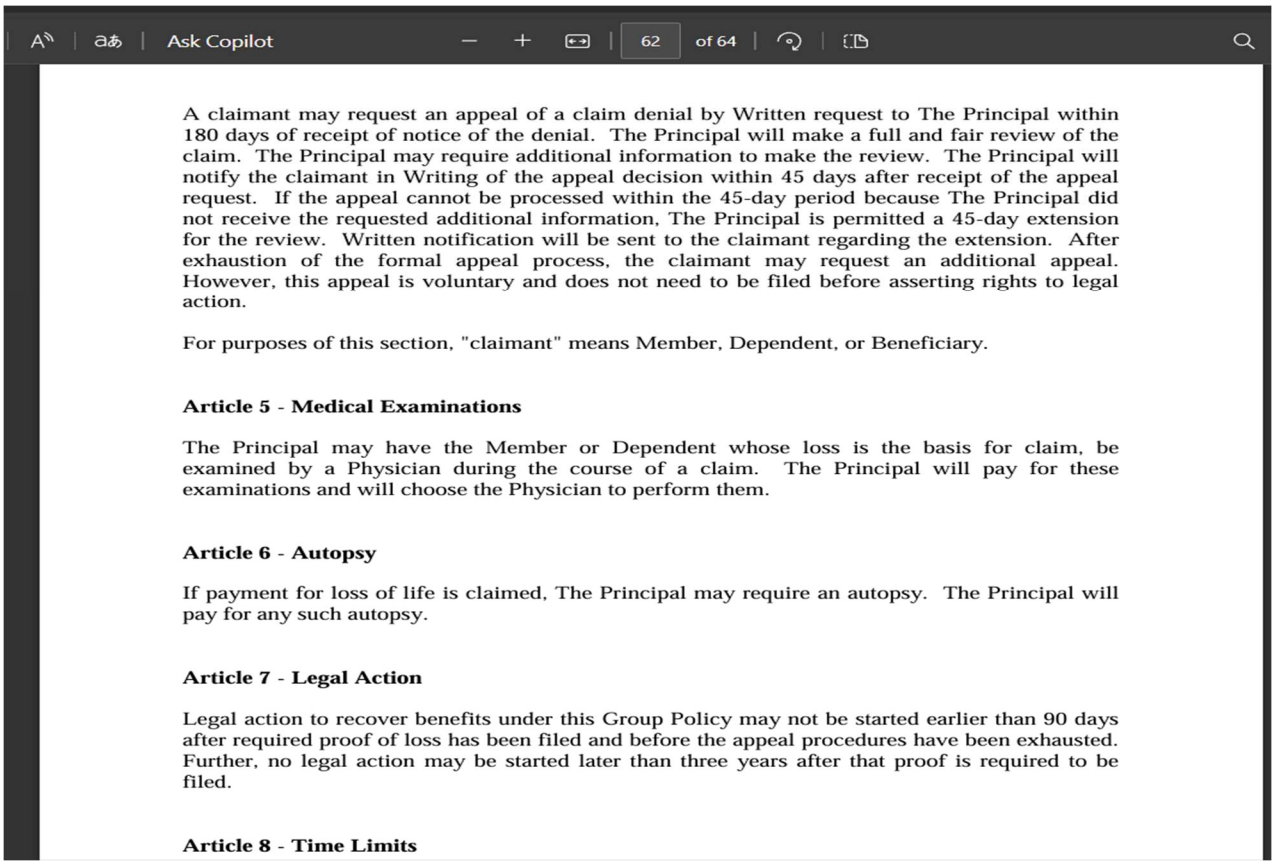
Query 1:





upGrad

Query 2:



A claimant may request an appeal of a claim denial by Written request to The Principal within 180 days of receipt of notice of the denial. The Principal will make a full and fair review of the claim. The Principal may require additional information to make the review. The Principal will notify the claimant in Writing of the appeal decision within 45 days after receipt of the appeal request. If the appeal cannot be processed within the 45-day period because The Principal did not receive the requested additional information, The Principal is permitted a 45-day extension for the review. Written notification will be sent to the claimant regarding the extension. After exhaustion of the formal appeal process, the claimant may request an additional appeal. However, this appeal is voluntary and does not need to be filed before asserting rights to legal action.

For purposes of this section, "claimant" means Member, Dependent, or Beneficiary.

Article 5 - Medical Examinations

The Principal may have the Member or Dependent whose loss is the basis for claim, be examined by a Physician during the course of a claim. The Principal will pay for these examinations and will choose the Physician to perform them.

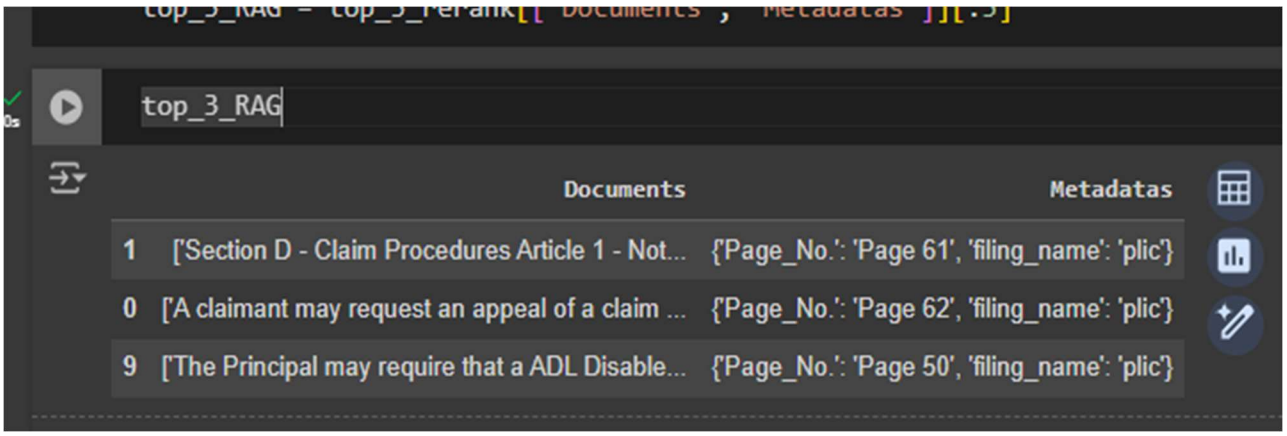
Article 6 - Autopsy

If payment for loss of life is claimed, The Principal may require an autopsy. The Principal will pay for any such autopsy.

Article 7 - Legal Action

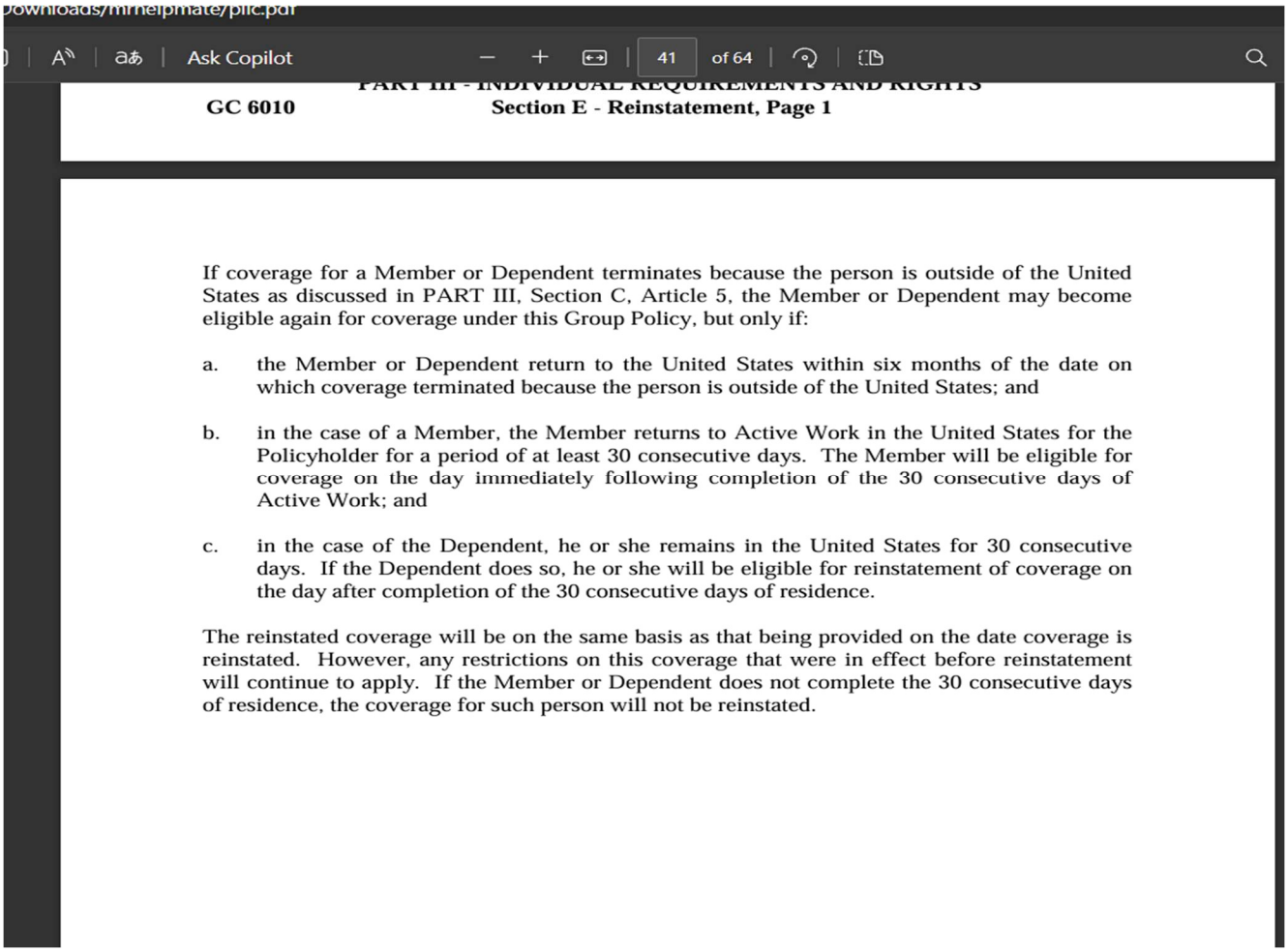
Legal action to recover benefits under this Group Policy may not be started earlier than 90 days after required proof of loss has been filed and before the appeal procedures have been exhausted. Further, no legal action may be started later than three years after that proof is required to be filed.

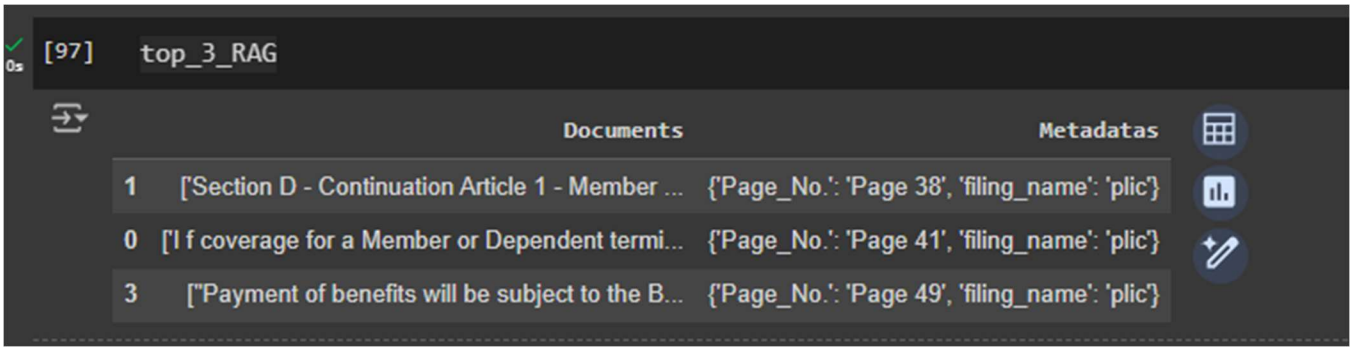
Article 8 - Time Limits



GEN-AI Mr.HelpMate AI

Query 3:





11. Applications

This project has potential applications in various fields, including:

- Customer support automation.
- Document retrieval systems.
- Knowledge management systems.
- AI-powered research assistants.

12. Conclusion

The `**Mr. HelpMate**` project successfully integrates multiple layers to process, search, and generate responses from text data. The embedding layer effectively processes the text, the search layer retrieves relevant documents with high accuracy, and the generative layer produces contextually appropriate answers. While the system shows promising results, future improvements could include optimizing the chunking strategy further, enhancing the re-ranking algorithm, and expanding the use cases.

THANK YOU

upGrad