HelpMateAI-Project

Introduction

HelpMateAI is an **AI-powered Retrieval-Augmented Generation (RAG) system** designed to assist with **insurance-related queries**. It efficiently retrieves information from insurance documents and generates precise responses using **LLMs like GPT-3.5**. The system incorporates **semantic search**, **cross-encoder re-ranking**, and **LLM-based response generation** to ensure that answers are fact-based and contextually accurate.

The project leverages **vector embeddings** and **retrieval mechanisms** to enhance the accuracy of insurance-related information retrieval. It is designed to work with a **predefined set of insurance documents** that are preprocessed and stored in a **vector database**. Users can ask **natural language queries**, and the system fetches the most relevant sections, refines the ranking using a **cross-encoder**, and generates a well-structured answer with citations.

Problem Statement

Insurance policies and legal documents are often lengthy and difficult to navigate. Users, including **policyholders, agents, and legal teams**, may need quick access to **specific details** regarding coverage, claims, and policy terms.

Challenges:

1. **Complexity of Documents**: Insurance documents contain technical jargon and are often **hundreds of pages long**.
2. **Time-Consuming Manual Search**: Finding specific clauses in large documents requires **manual reading** and **keyword-based searches**, which may **miss context**.
3. **Inconsistent Answers**: Traditional **search engines** return **entire documents**, leaving users to **extract relevant information manually**.
4. **Contextual Relevance**: Simple keyword-based searches lack **semantic understanding** and may return **irrelevant sections**.

Goal: Develop an AI-driven system that **automates the extraction of relevant insurance policy details**, **ranks retrieved information accurately**, and **generates precise answers with citations**.

Approach

The project uses **Retrieval-Augmented Generation (RAG)**, which combines **document retrieval** and **text generation**.

1. **Document Processing & Storage**
    - Extracts text from **PDF insurance documents** using **pdfplumber**.
    - Splits documents into **fixed-size text chunks**.
    - Converts chunks into **vector embeddings** using **SentenceTransformer**.
    - Stores embeddings in a **vector database (ChromaDB)** for **fast retrieval**.
2. **Query Processing & Document Retrieval**
    - Encodes the **user's query** into an **embedding** using the same model.
    - Performs **similarity search** in **ChromaDB** to retrieve the **top-K relevant chunks**.
    - Applies a **Cross-Encoder model** to **re-rank** the retrieved results for **higher accuracy**.
3. **Answer Generation & Citation**
    - The **top 3 most relevant chunks** are passed to **GPT-3.5**.
    - The model generates a **well-structured answer**, **grounded in retrieved documents**.
    - Includes **citations** (document name and page number).
4. **User Query Handling & API Support**
    - Allows users to **send queries via a Python script or API**.
    - Retrieves **precise** insurance-related information.
    - Provides a **concise answer** instead of entire documents.

System Design

The system consists of three key components:

1. Document Processing & Vector Store

- **Preprocessing**: Extract text from PDFs.
- **Chunking**: Divide text into **fixed-size blocks**.
- **Embedding**: Convert chunks into **numerical vector representations**.
- **Storage**: Save vectors in **ChromaDB**.

2. Query Processing & Retrieval

- **Query Embedding**: Convert user query into a **vector representation**.
- **Vector Search**: Find similar embeddings in **ChromaDB**.
- **Re-ranking**: Use a **Cross-Encoder model** to rank retrieved results.

3. Answer Generation

- **Input**: User query + retrieved document chunks.
- **LLM Processing**: GPT-3.5 generates an answer based on the provided context.
- **Output**: A precise, **cited** response.

Current Implementation

The system follows a **structured workflow** to **retrieve and generate responses**. Below is the core logic:

1. Query Processing

```
query = "What is the life insurance coverage for disability?"
df = search(query)  # Retrieve relevant documents
df = apply_cross_encoder(query, df)  # Re-rank results
df = get_topn(3, df)  # Select the top 3 most relevant chunks
response = generate_response(query, df)  # Generate final response
print("\n".join(response))  # Print the response
```

2. Optimized Query Handling Using Loops

```
queries = [
    "what is the life insurance coverage for disability",
    "what is the Proof of ADL Disability or Total Disability",
    "what is condition of death while not wearing Seat Belt"
]

def process_queries(queries):
    results = {}
    for query in queries:
        df = search(query)
        df = apply_cross_encoder(query, df)
        df = get_topn(3, df)
        response = generate_response(query, df)
        results[query] = "\n".join(response)
        print(f"\nQuery: {query}\n")
        print(results[query])

    return results

responses = process_queries(queries)
```

3. API Endpoint for External Integration

```
from fastapi import FastAPI
from pydantic import BaseModel

app = FastAPI()

class QueryRequest(BaseModel):
    query: str

@app.post("/query")
def answer_query(request: QueryRequest):
    df = search(request.query)
    df = apply_cross_encoder(request.query, df)
    df = get_topn(3, df)
    response = generate_response(request.query, df)
    return {"query": request.query, "response": response}
```

Project Setup

1. Clone the Repository

```
git clone https://github.com/ivineettiwari/HelpMateAI-Project.git
cd HelpMateAI-Project
```

2. Create a Virtual Environment

```
python -m venv env
source env/bin/activate  # On Mac/Linux
env\Scripts\activate     # On Windows
```

3. Install Dependencies

```
pip install -r requirements.txt
```

4. Set Up API Keys

Create a `.env` file and add your **OpenAI API Key**:

```
OPENAI_API_KEY=your-api-key-here
```

5. Run the Application

```
View Jupyter Notebook
```

Future Scope

- **Support for Additional Domains**: Extend beyond insurance to **legal, finance, and healthcare documents**.
- **Hybrid Search**: Combine **vector search + keyword-based search** for enhanced accuracy.
- **Model Fine-Tuning**: Fine-tune **cross-encoders** and **LLMs** on domain-specific datasets.
- **Multi-Language Support**: Expand support for **queries in multiple languages**.
- **Improved UI & API**: Build a **web-based front-end** for easier interaction.

Conclusion

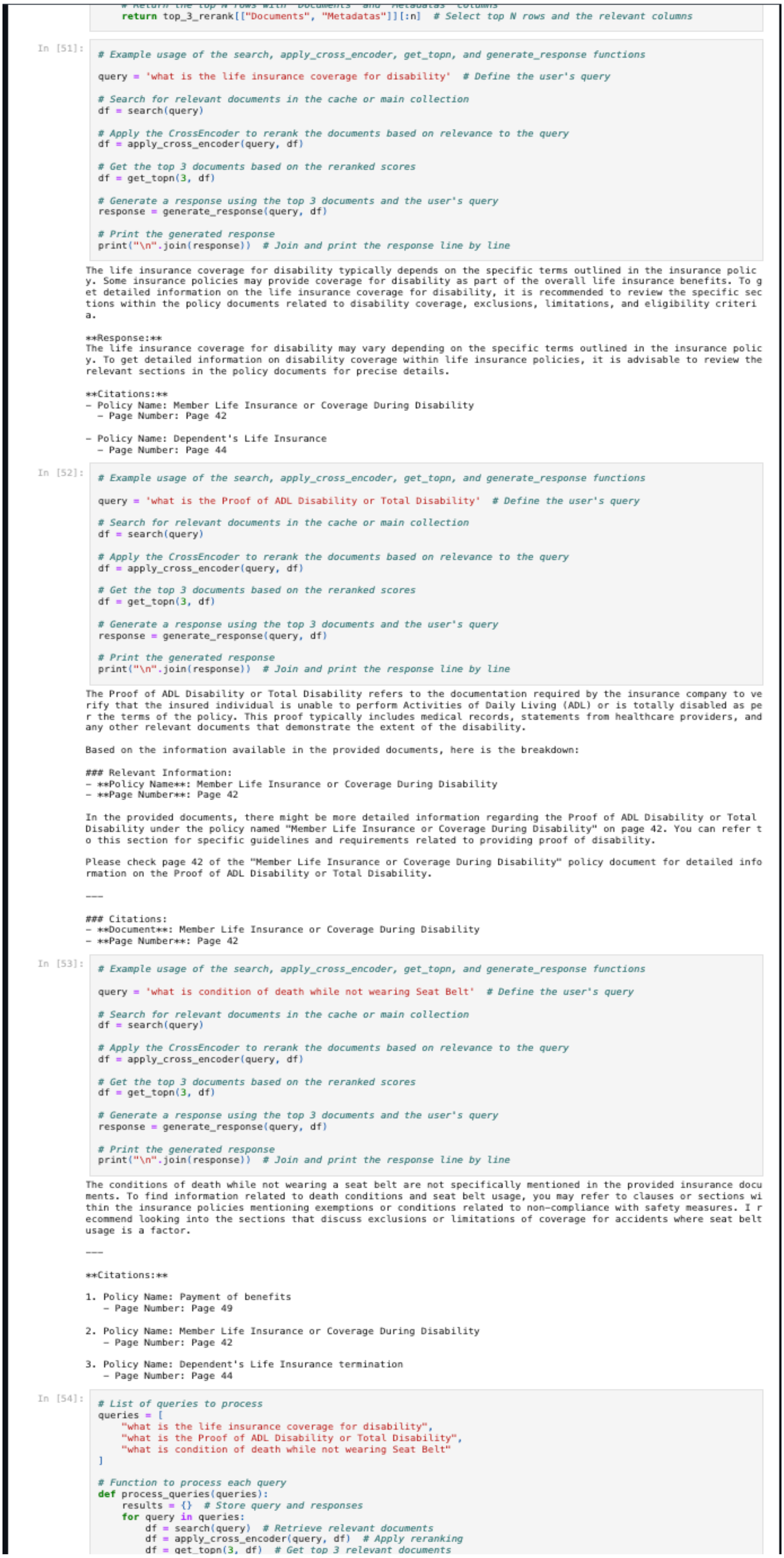
The **HelpMateAI-Project** provides a powerful **AI-driven document retrieval system** tailored for insurance-related queries. By leveraging **vector embeddings, cross-encoder ranking, and GPT-3.5**, it offers:

1. **Fast and accurate information retrieval**.
2. **Well-structured answers with citations**.
3. **Scalability for large insurance datasets**.

This **open-source project** can be further enhanced by integrating **real-time policy updates, UI improvements, and domain-specific model optimizations**.

For more details and contributions, visit the GitHub repository:
HelpMateAI-Project

Screen Shots.



```
    # Return the top N rows with 'Documents' and 'Metadatas' columns
    return top_3_rerank[["Documents", "Metadatas"]][:n]  # Select top N rows and the relevant columns
```

In [51]:

```
# Example usage of the search, apply_cross_encoder, get_topn, and generate_response functions

query = 'what is the life insurance coverage for disability'  # Define the user's query

# Search for relevant documents in the cache or main collection
df = search(query)

# Apply the CrossEncoder to rerank the documents based on relevance to the query
df = apply_cross_encoder(query, df)

# Get the top 3 documents based on the reranked scores
df = get_topn(3, df)

# Generate a response using the top 3 documents and the user's query
response = generate_response(query, df)

# Print the generated response
print("\n".join(response))  # Join and print the response line by line
```

```
The life insurance coverage for disability typically depends on the specific terms outlined in the insurance polic
y. Some insurance policies may provide coverage for disability as part of the overall life insurance benefits. To g
et detailed information on the life insurance coverage for disability, it is recommended to review the specific sec
tions within the policy documents related to disability coverage, exclusions, limitations, and eligibility criteri
a.

**Response:**
The life insurance coverage for disability may vary depending on the specific terms outlined in the insurance polic
y. To get detailed information on disability coverage within life insurance policies, it is advisable to review the
relevant sections in the policy documents for precise details.

**Citations:**
- Policy Name: Member Life Insurance or Coverage During Disability
  - Page Number: Page 42

- Policy Name: Dependent's Life Insurance
  - Page Number: Page 44
```

In [52]:

```
# Example usage of the search, apply_cross_encoder, get_topn, and generate_response functions

query = 'what is the Proof of ADL Disability or Total Disability'  # Define the user's query

# Search for relevant documents in the cache or main collection
df = search(query)

# Apply the CrossEncoder to rerank the documents based on relevance to the query
df = apply_cross_encoder(query, df)

# Get the top 3 documents based on the reranked scores
df = get_topn(3, df)

# Generate a response using the top 3 documents and the user's query
response = generate_response(query, df)

# Print the generated response
print("\n".join(response))  # Join and print the response line by line
```

```
The Proof of ADL Disability or Total Disability refers to the documentation required by the insurance company to ve
rify that the insured individual is unable to perform Activities of Daily Living (ADL) or is totally disabled as pe
r the terms of the policy. This proof typically includes medical records, statements from healthcare providers, and
any other relevant documents that demonstrate the extent of the disability.

Based on the information available in the provided documents, here is the breakdown:

### Relevant Information:
- **Policy Name**: Member Life Insurance or Coverage During Disability
- **Page Number**: Page 42

In the provided documents, there might be more detailed information regarding the Proof of ADL Disability or Total
Disability under the policy named "Member Life Insurance or Coverage During Disability" on page 42. You can refer t
o this section for specific guidelines and requirements related to providing proof of disability.

Please check page 42 of the "Member Life Insurance or Coverage During Disability" policy document for detailed info
rmation on the Proof of ADL Disability or Total Disability.

---

### Citations:
- **Document**: Member Life Insurance or Coverage During Disability
- **Page Number**: Page 42
```

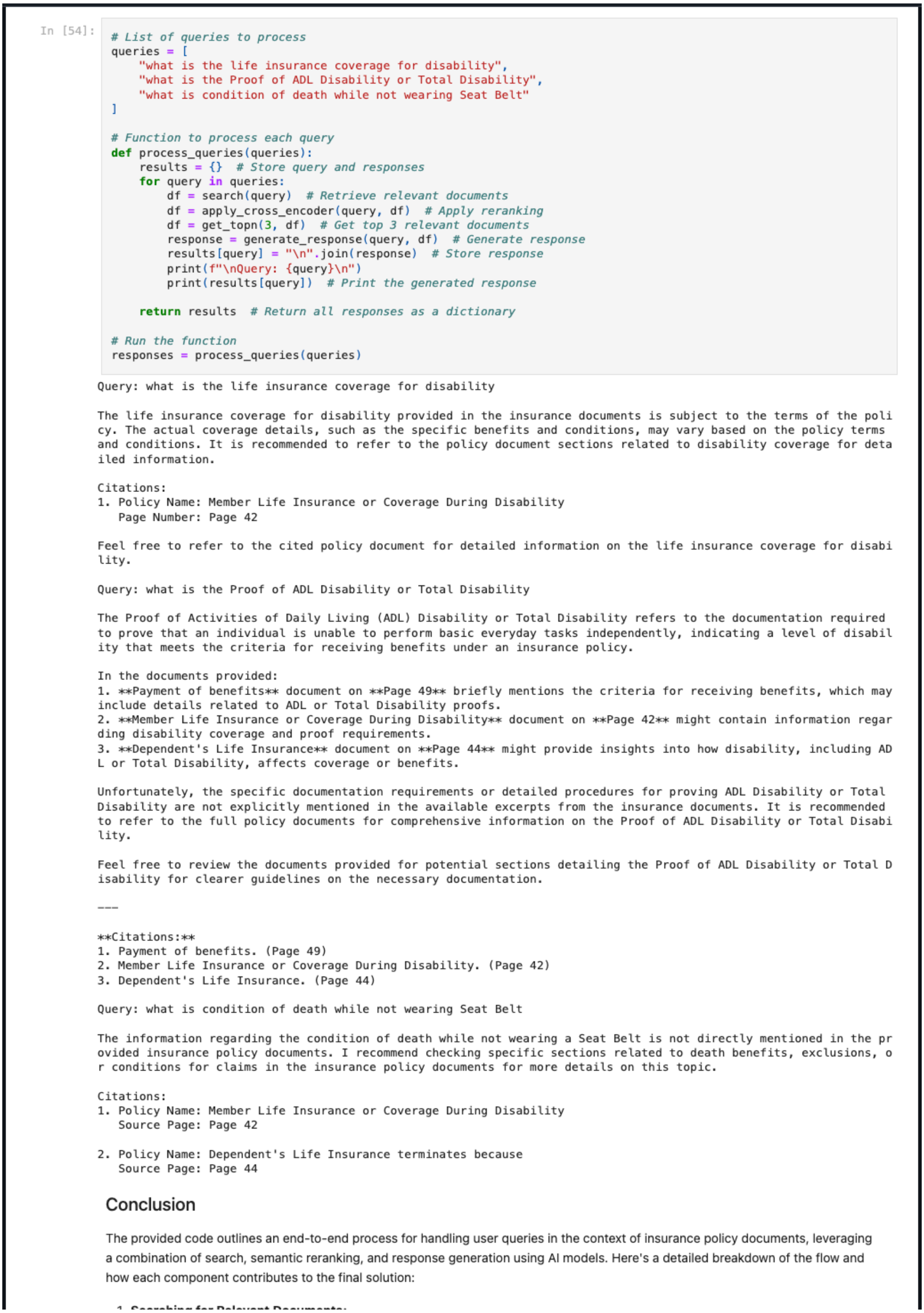
In [53]:

```
# Example usage of the search, apply_cross_encoder, get_topn, and generate_response functions

query = 'what is condition of death while not wearing Seat Belt'  # Define the user's query

# Search for relevant documents in the cache or main collection
df = search(query)

# Apply the CrossEncoder to rerank the documents based on relevance to the query
df = apply_cross_encoder(query, df)

# Get the top 3 documents based on the reranked scores
df = get_topn(3, df)

# Generate a response using the top 3 documents and the user's query
response = generate_response(query, df)

# Print the generated response
print("\n".join(response))  # Join and print the response line by line
```

```
The conditions of death while not wearing a seat belt are not specifically mentioned in the provided insurance docu
ments. To find information related to death conditions and seat belt usage, you may refer to clauses or sections wi
thin the insurance policies mentioning exemptions or conditions related to non-compliance with safety measures. I r
ecommend looking into the sections that discuss exclusions or limitations of coverage for accidents where seat belt
usage is a factor.

---

**Citations:**

1. Policy Name: Payment of benefits
   - Page Number: Page 49

2. Policy Name: Member Life Insurance or Coverage During Disability
   - Page Number: Page 42

3. Policy Name: Dependent's Life Insurance termination
   - Page Number: Page 44
```

In [54]:

```
# List of queries to process
queries = [
    "what is the life insurance coverage for disability",
    "what is the Proof of ADL Disability or Total Disability",
    "what is condition of death while not wearing Seat Belt"
]

# Function to process each query
def process_queries(queries):
    results = {}  # Store query and responses
    for query in queries:
        df = search(query)  # Retrieve relevant documents
        df = apply_cross_encoder(query, df)  # Apply reranking
        df = get_topn(3, df)  # Get top 3 relevant documents
```



In [54]:

```python
# List of queries to process
queries = [
    "what is the life insurance coverage for disability",
    "what is the Proof of ADL Disability or Total Disability",
    "what is condition of death while not wearing Seat Belt"
]

# Function to process each query
def process_queries(queries):
    results = {}  # Store query and responses
    for query in queries:
        df = search(query)  # Retrieve relevant documents
        df = apply_cross_encoder(query, df)  # Apply reranking
        df = get_topn(3, df)  # Get top 3 relevant documents
        response = generate_response(query, df)  # Generate response
        results[query] = "\n".join(response)  # Store response
        print(f"\nQuery: {query}\n")
        print(results[query])  # Print the generated response

    return results  # Return all responses as a dictionary

# Run the function
responses = process_queries(queries)
```

```
Query: what is the life insurance coverage for disability

The life insurance coverage for disability provided in the insurance documents is subject to the terms of the poli
cy. The actual coverage details, such as the specific benefits and conditions, may vary based on the policy terms
and conditions. It is recommended to refer to the policy document sections related to disability coverage for deta
iled information.

Citations:
1. Policy Name: Member Life Insurance or Coverage During Disability
   Page Number: Page 42

Feel free to refer to the cited policy document for detailed information on the life insurance coverage for disabi
lity.

Query: what is the Proof of ADL Disability or Total Disability

The Proof of Activities of Daily Living (ADL) Disability or Total Disability refers to the documentation required
to prove that an individual is unable to perform basic everyday tasks independently, indicating a level of disabil
ity that meets the criteria for receiving benefits under an insurance policy.

In the documents provided:
1. **Payment of benefits** document on **Page 49** briefly mentions the criteria for receiving benefits, which may
include details related to ADL or Total Disability proofs.
2. **Member Life Insurance or Coverage During Disability** document on **Page 42** might contain information regar
ding disability coverage and proof requirements.
3. **Dependent's Life Insurance** document on **Page 44** might provide insights into how disability, including AD
L or Total Disability, affects coverage or benefits.

Unfortunately, the specific documentation requirements or detailed procedures for proving ADL Disability or Total
Disability are not explicitly mentioned in the available excerpts from the insurance documents. It is recommended
to refer to the full policy documents for comprehensive information on the Proof of ADL Disability or Total Disabi
lity.

Feel free to review the documents provided for potential sections detailing the Proof of ADL Disability or Total D
isability for clearer guidelines on the necessary documentation.

---

**Citations:**
1. Payment of benefits. (Page 49)
2. Member Life Insurance or Coverage During Disability. (Page 42)
3. Dependent's Life Insurance. (Page 44)

Query: what is condition of death while not wearing Seat Belt

The information regarding the condition of death while not wearing a Seat Belt is not directly mentioned in the pr
ovided insurance policy documents. I recommend checking specific sections related to death benefits, exclusions, o
r conditions for claims in the insurance policy documents for more details on this topic.

Citations:
1. Policy Name: Member Life Insurance or Coverage During Disability
   Source Page: Page 42

2. Policy Name: Dependent's Life Insurance terminates because
   Source Page: Page 44
```

Conclusion

The provided code outlines an end-to-end process for handling user queries in the context of insurance policy documents, leveraging a combination of search, semantic reranking, and response generation using AI models. Here's a detailed breakdown of the flow and how each component contributes to the final solution:

1. **Searching for Relevant Documents:**