

HelpMate AI

Tejus B



Objectives



Develop a Semantic Search System

Utilize the RAG pipeline for efficient document retrieval, integrating embedding, search, and generation layers.



Implement a Cache Layer

Enhance performance by caching previous queries and results using ChromaDB.



Extract and Structure Information

Process and store PDF documents in a structured format, generating vector representations with OpenAI's models.



Build a Generative Search System

Develop a robust system capable of accurately answering questions from a policy document.



System Design

RAG Pipeline & Cache Implementation



RAG Pipeline Overview

Integrates Embedding, Search, and Generation layers to retrieve and generate accurate responses.



Search & Rank Layer

Performs semantic similarity searches and re-ranks results using cross-encoders.



Embedding Layer

Processes and chunks PDFs, generating vector embeddings stored in ChromaDB.



Cache Implementation

Enhances performance by caching queries and results, utilizing a similarity threshold.



Implementation

- **Document Processing:** Utilized pdfplumber for text and table extraction, followed by chunking and vector embedding with OpenAI's models.
- **Semantic Search:** Implemented semantic similarity searches using the RAG pipeline and vector database ChromaDB.
- **Cache System:** Developed a cache system with ChromaDB, optimizing the retrieval of previous queries and results.



Photo by Maik Jonietz on Unsplash

Challenges Faced

- **Data Quality & Preprocessing:** Extracting relevant information from complex insurance documents proved challenging due to varied text structures.
- **Chunking Strategies:** Optimizing chunk size and overlap to maintain context without losing coherence was critical but difficult.
- **Query Understanding & Matching:** Designing relevant queries that required sophisticated understanding and reasoning posed a significant challenge.

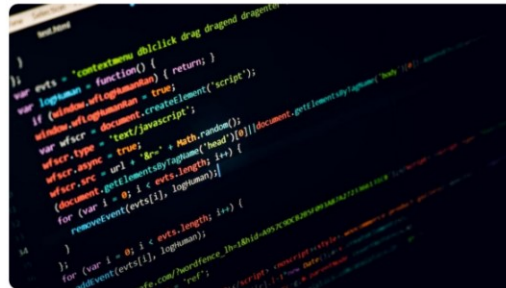


Photo by RoonZ nl on Unsplash

Conclusion



Successful Implementation

HelpMate AI successfully achieved its objectives, implementing a robust semantic search system with the RAG pipeline.



Challenges Overcome

The project addressed significant challenges in data processing, chunking strategies, and query design.



Scalable and Efficient System

The final system is scalable, efficient, and provides accurate information retrieval from complex documents.

Lessons Learned



Efficient Document Processing

Utilizing tools like pdfplumber is crucial for handling complex PDF documents efficiently.



Semantic Search Optimization

Fine-tuning search parameters and thresholds is essential for achieving optimal results.



Cache Management

Implementing an effective caching strategy significantly improves system performance.

