Retrieval-Augmented Generation

(RAG) System for Document-based Question Answering

**1. Project Overview**

The goal of this project was to design and develop a **Retrieval-Augmented Generation (RAG)** system that integrates document retrieval and natural language generation to answer questions based on a given set of documents, specifically insurance-related FAQs. We utilized the **LlamaIndex** framework for document indexing and retrieval and **OpenAI's GPT models** to generate relevant and accurate responses.

This project aims to address the need for efficient document-based question-answering systems, particularly in industries that deal with large amounts of textual information like insurance, healthcare, and legal sectors.

**2. Objectives**

* **Develop a Retrieval-Augmented Generation (RAG) system** that can accurately answer user queries from a set of documents.
* **Implement an efficient document indexing mechanism** using **LlamaIndex**.
* **Integrate OpenAI's GPT model** to generate human-like responses based on the retrieved documents.
* **Test the system's performance** using predefined questions and evaluate its accuracy and relevance.
* **Enhance the system with future improvements** such as sub-question handling and customized nodes.

**3.1. System Overview**

The system is divided into the following key modules:

1. **Document Loader**: This module loads and processes input documents (e.g., insurance FAQs).
2. **Document Indexing**: The loaded documents are converted into a structured format (nodes) that can be indexed and queried.
3. **Query Engine**: The query engine retrieves relevant documents and generates responses using the OpenAI API.
4. **Response Generation**: Using OpenAI's GPT model, the system generates contextually relevant responses based on retrieved data.
5. **Testing Pipeline**: A feedback loop for testing the accuracy of the responses and gathering insights to refine the system.

**3.2. Architecture Diagram**

A diagram of a process

Description automatically generated

A diagram of a dog

Description automatically generated

**4. Implementation**

**4.1. Libraries and Tools Used**

**LlamaIndex:** For document indexing and retrieval.

OpenAI API: For language model integration (GPT).

Pandas: For handling tabular data (insurance FAQs).

**Google Colab:** For running the Jupyter notebook and leveraging cloud resources.

**Python:** The primary programming language.

**4.2. Code Walkthrough**

**Step 1: Setup and Imports**

We installed necessary libraries like LlamaIndex, OpenAI, and other supporting libraries for document handling.

**Step 2: Loading the Data**

Insurance FAQs were loaded from a CSV file. This was then converted into documents that could be indexed by the system.

**Step 3: Document Indexing**

Documents were indexed using LlamaIndex, which allowed us to retrieve relevant nodes (chunks of text) based on a query.

**Step 4: Query Engine and Response Generation**

A query was processed by the query\_engine, which retrieved documents related to the question. The retrieved documents were then passed to OpenAI's GPT model to generate a natural language response**.**

**Step 5: Testing Pipeline**

A series of predefined questions were tested to evaluate the system's performance and response accuracy. The feedback was used to refine the response generation process.

**Project By:**  
**Archana K  
Dhruv Gaur  
Akanksha kumari**