Overview

The RAG chatbot is a natural language processing (NLP) application that uses a combination of information retrieval and text generation techniques to provide accurate and contextualized answers to user queries. The chatbot is designed to ingest documents, index them, and use the indexed information to generate responses to user questions.

Process Flow

The RAG chatbot process flow involves the following steps:

1. **Document Ingestion**: The chatbot ingests documents in various formats (e.g., PDF, text, Word documents) and processes them to extract relevant information.

2. **Indexing**: The extracted information is indexed using a vector database, which enables efficient similarity search and retrieval of relevant documents.

3. **Query Processing**: When a user submits a query, the chatbot uses the indexed information to retrieve relevant documents and generate a response.

4. **Response Generation:** The chatbot uses a text generation model to generate a response based on the retrieved documents and the user's query.

5. **Post-processing**: The generated response is post-processed to ensure that it is accurate, relevant, and contextualized.

Key Components

The RAG chatbot relies on several key components, including:

1. **Document Loader:** A module that loads and processes documents in various formats.
2. **Vector Database:** A database that stores the indexed information and enables efficient similarity search and retrieval.
3. **Text Generation Model:** A model that generates text based on the retrieved documents and user query.
4. **Query Processing Module:** A module that processes user queries and generates responses.

Required Libraries

The following libraries are required to build and run the RAG chatbot:

1. **LangChain:** LangChain is a library that provides a simple and efficient way to build and integrate language models into applications. It is used to implement the RAG model and provide a interface for querying and retrieving documents.
2. **Streamlit**: Streamlit is a library that allows users to build and deploy web applications with a simple and intuitive API. It is used to build the user interface of the chatbot and provide a way for users to interact with the chatbot.
3. **Pyngrok**: Pyngrok is a library that provides a simple way to create tunnels to localhost, allowing users to expose their local development environment to the internet. It is used to expose the chatbot to the internet and allow users to access it remotely.
4. **OpenAI**: OpenAI is a library that provides access to powerful language models, including the GPT-3 model used in this application. It is used to generate text and provide responses to user queries.
5. **Milvus**: Milvus is a vector database that provides efficient similarity search and retrieval capabilities. It is used to store and retrieve documents based on their semantic meaning.
6. **Unstructured**: Unstructured is a library that provides a simple way to load and process unstructured data, such as text documents and PDFs. It is used to load and process the documents used in the chatbot.