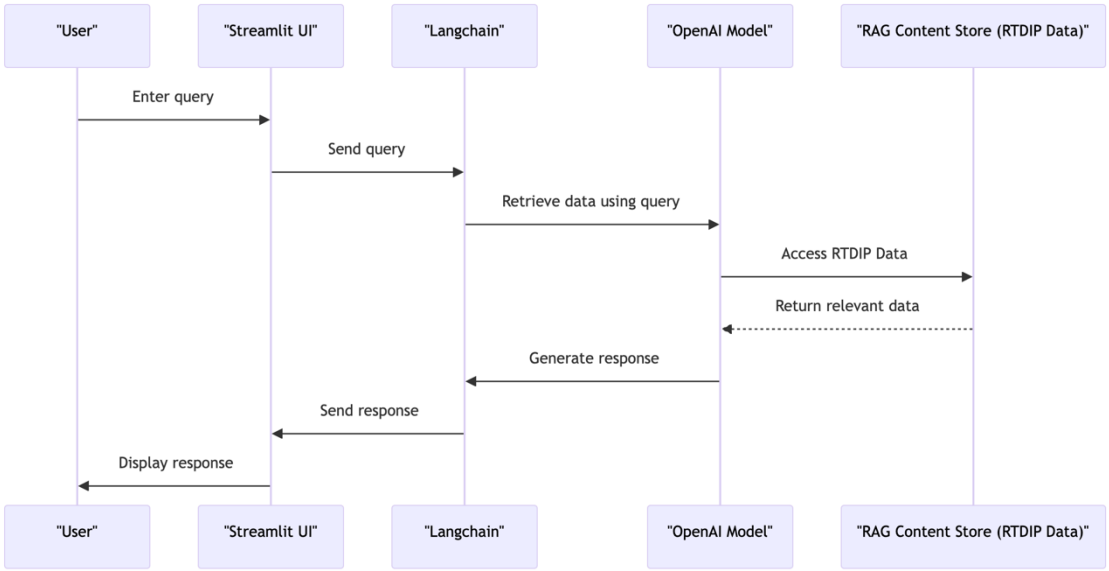
Software Architecture Design Documentation

Overview

This software is designed to provide users with information from the RTDIP Data using a chat interface. It uses Langchain for processing language model applications, OpenAI's language model for generating responses, and Streamlit for the user interface.



Components

- **Streamlit UI:** This is the front end of the application where users interact with the system. It's a simple chat interface where users can type their queries.

- **Langchain:** This component acts as a bridge between the Streamlit UI and the OpenAI model and the content store. It processes user queries and handles the logic for chaining different components together.

- **OpenAI Model:** This is the core of the response generation system. It uses the data retrieved from the RAG Content Store to generate answers to user queries.

- **RAG Content Store (RTDIP Data):** This is the database where the RTDIP Data is stored. It's used by the OpenAI model through Langchain to retrieve information relevant to the user's query.

Workflow

- **User Query:** The process starts when a user types a query into the Streamlit UI. Query Processing: The Streamlit UI sends this query to Langchain.
- **Data Retrieval:** Langchain communicates with the OpenAI model, which then accesses the RAG Content Store to fetch relevant RTDIP Data.
- **Response Generation:** The OpenAI model uses this data to generate a response to the user's query.
- **Display Response:** This response is sent back through Langchain to the Streamlit UI, which then displays it to the user.
  Key Features
- **Simple User Interface:** The Streamlit UI is user-friendly, making it easy for users to input their queries and receive responses.
- **Efficient Data Handling:** Langchain efficiently manages the flow of data between the UI, the OpenAI model, and the RAG Content Store.