**Overall Approach**

The custom chatbot utilizes Streamlit for the user interface, integrates ChatGroq for natural language processing(llm selection), and uses GoogleGenerativeAIEmbeddings for embedding text. The chatbot is designed to process user queries by retrieving relevant documents, generating context-aware responses, and maintaining a conversational history. The workflow involves loading documents, splitting them into manageable chunks, embedding these chunks, and using a language model to answer user queries based on the provided context.

**Frameworks/Libraries/Tools Used**

1. **Streamlit**
   * **Purpose**: To create the user interface for the chatbot.
   * **Usage**: Used to build the web app interface, manage session states, and handle user input/output.
2. **Langchain**
   * **ChatGroq**
     + **Purpose**: Natural language processing and response generation.
     + **Usage**: Utilized for generating responses based on the user's input and retrieved context.
   * **RecursiveCharacterTextSplitter**
     + **Purpose**: Text splitting.
     + **Usage**: Splits large documents into smaller, manageable chunks for efficient processing.
   * **FAISS**
     + **Purpose**: Vector storage and retrieval.
     + **Usage**: Stores the vector embeddings of document chunks and retrieves relevant documents based on user queries.
   * **GoogleGenerativeAIEmbeddings**
     + **Purpose**: Text embedding.
     + **Usage**: Embeds text from documents to create vector representations for similarity searches.
   * **PyPDFLoader**
     + **Purpose**: Document loading.
     + **Usage**: Loads PDF documents which serve as the knowledge base for the chatbot.
3. **dotenv**
   * **Purpose**: Environment variable management.
   * **Usage**: Loads API keys from environment variables securely.
4. **Time**
   * **Purpose**: Performance measurement.
   * **Usage**: Measures the time taken for the chatbot to process and respond to queries.

**Problems Faced and Solutions**

1. **Context Management**
   * **Problem**: Maintaining conversational context for generating relevant responses.
   * **Solution**: Stored user messages and responses in the session state, concatenating them to provide historical context for each query.
2. **Response Accuracy**
   * **Problem**: Ensuring accurate responses based on the provided context.
   * **Solution**: Designed a custom prompt template to instruct the language model to base its answers strictly on the given context and to avoid fabricating information.

**Future Scope**

1. **Improved User Interface**
   * Enhance the UI with additional features such as buttons for common actions, a history sidebar for easy navigation, and visual indicators for processing status.
2. **Expanded Knowledge Base**
   * Integrate additional document formats (e.g., DOCX, TXT) and sources (e.g., web scraping) to broaden the chatbot’s knowledge base.
3. **Multi-language Support**
   * Extend the chatbot to support multiple languages, making it accessible to a wider audience.