Project Overview

The project is a Streamlit-based web application named **PDF Dialogue Zone** that allows users to upload PDF documents and interactively ask questions about the content of those documents. The application leverages various machine learning and natural language processing tools, including LangChain, FAISS, and GooglePalm LLM, to enable efficient text extraction, embedding, and question-answering functionalities.

Here's a summary of the approach:

User Interface:

- Create a sidebar using Streamlit to display the application title and description.
- Provide a file uploader for users to upload their PDF documents.

PDF Text Processing:

- Extract text from the uploaded PDF document.
- Split the extracted text into smaller chunks using RecursiveCharacterTextSplitter for efficient embedding and search.

Embedding Management:

- Check if embeddings for the document already exist on disk. If they do, load them; otherwise, generate new embeddings using HuggingFaceInstructEmbeddings and save them using FAISS.

Question-Answering Functionality:

- Allow users to input questions about the uploaded PDF document.
- Perform a similarity search on the embedded text chunks to find relevant sections.
- Use the GooglePalm language model to generate answers to the questions based on the relevant text sections.

Execution:

- Run the main function to initialize and execute the application.

Libraries and modules

The project begins by importing necessary libraries and modules:

- **pypdf** for reading PDF files.
- **Streamlit** for creating the web application interface.
- **LangChain** for text splitting and question-answering.
- **FAISS** for vector storage and similarity search.
- **HuggingFaceInstructEmbeddings** for embedding text chunks.
- **InstructorEmbedding** for additional embedding functionalities.
- **dotenv** for environment variable management.
- **pickle** for saving and loading embeddings.

Problems faced and overcoming them during project:

- The first problem is extracting text from PDFs can be inconsistent due to the varying structures and formats of PDF documents. And I overcome it by utilizing the PdfReader from pypdf

- The embeddings which I have used is of OpenAI embeddings and I only know that the models of OpenAI are paid. But the embeddings also paid. So, that's why I used Hugging Face Instruct Embeddings

- If using Hugging Face Instruct Embeddings, we must have to install sentence-transformers. But still its not working and it gives error. Then I installed sentence-transformers==2.2.2. Then only it worked

- If using the google palm model, I have a problem in running it, because I have restricted my GOOGLE_API_KEY. Then it overcome by unrestricting the Api key

Future scope of this chatbot(what more we can do to it):

- Integrate more advanced embedding techniques, such as BERT, GPT embeddings, or other transformer-based models, for better text understanding.
- Add support for multiple languages
- Develop a mobile application version to provide easy access to users on the go.
- Integrate voice recognition and synthesis to allow users to interact with the chatbot using voice commands.
- Updating the model for better performance.