

**Project Title: Chat With Multiple PDFs**

Generative AI and Prompt Engineering (Fintech)

RV University

Rachana M S, Meghana H

20-05-2024

Mentor: Manjul Krishna Gupta

**Abstract**

Our project involves the development of a platform that enables users to engage in interactive chat sessions with multiple PDF documents. By leveraging Langchain and Google Gemini Pro Vision, the system allows users to ask questions and receive answers based on the content of the uploaded PDFs. This innovative solution is designed to enhance communication and facilitate comprehensive discussions and inquiries, providing users with an efficient and seamless experience in extracting knowledge from PDF documents.

The methods employed include:

* **Text Extraction**: PyPDF2 extracts text from uploaded PDFs.
* **Text Chunking**: RecursiveCharacterTextSplitter segments the extracted text into manageable chunks.
* **Vector Embeddings:** Google Generative AI's embedding model converts text chunks into high-dimensional vectors, stored in a FAISS vector store for efficient similarity searches.
* **Conversational AI Integration:** ChatGoogleGenerativeAI processes user queries against the vector store and generates contextually relevant responses using a custom prompt template, ensuring accuracy or stating if the answer is unavailable in the context.

Key findings from the project include successfully integrating these technologies to provide an easy and robust solution for querying information from PDFs. The system demonstrates high accuracy in retrieving relevant information and providing detailed responses, thereby enhancing the usability and accessibility of document-based information. The project showcases the potential of combining text processing and AI technologies to solve complex information retrieval challenges effectively.

**Table of contents**

|  |  |  |
| --- | --- | --- |
| Sl.No | Section | Page |
| 1.1 | Cover Page & Title Page | 1 |
| 1.2 | Abstract | 2 |
| 1.3 | Table of contents | 3 |
| 1.4 | Chapter-1: Introduction | 4-15 |
| Chapter-2: Literature Review |
| Chapter-3: Architecture, Design, Methadology |
| Chapter-4: Implementation |
| Chapter-5: Test Results and Discussion |
| 1.6 | Conclusion | 16 |
| 1.7 | References | 17 |
| 1.8 | Appendices | 17 |

**Chapter-1: Introduction**

The rapid advancement in artificial intelligence and natural language processing has enabled the development of innovative solutions for managing and extracting information from large volumes of unstructured text. In this context, the project uses advanced text processing techniques, vector embeddings, and conversational AI to facilitate interactive querying of multiple PDF documents.

**Background Information**

PDFs are widely used for document distribution and archiving, but extracting and querying information from them can be challenging due to their unstructured nature. Traditional methods of information retrieval from PDFs are often inefficient and time-consuming, necessitating the development of more sophisticated tools that can handle large text volumes and provide accurate, contextually relevant answers.

**Objectives and Goals**

The primary objective of this project is to create a robust system that allows users to engage in conversational interactions with multiple PDF documents. The key goals include:

1. Efficient Text Extraction: Develop a method to accurately extract text from PDFs.
2. User-Friendly Interface: To create an intuitive and interactive user interface using Streamlit, making it easy for users to upload, manage, and interact with their documents.
3. Scalability: To support the simultaneous upload and interaction with up to 15 PDF files, each up to 200 MB in size, ensuring the platform can handle substantial volumes of data.
4. Integration of Advanced Tools: To integrate Langchain and Google Gemini Pro Vision for advanced document analysis and question-answering capabilities.
5. Seamless Interaction: To provide a seamless and efficient user experience where users can ask questions and receive accurate answers related to the content of the PDFs.

**Scope and Limitations**

**Scope:**

* PDF Document Handling: The platform supports upload and management of up to 15 PDF documents, with a combined size limit of 200 MB.
* Interactive Q&A: Users can ask questions related to PDF content and receive relevant answers.
* Technology Integration: Utilization of Langchain for backend processing and Google Gemini Pro Vision for document analysis, with Streamlit serving as the frontend framework.
* User Experience: Focus on creating a user-friendly, interactive experience facilitating easy document interaction.

**Limitations:**

* Document Types: The system specifically supports PDF documents and may not accommodate other file formats.
* File Size and Number Limitations: Handling up to 15 PDF files totaling 200 MB may be restrictive for users dealing with larger or numerous documents.
* Processing Time: The size and complexity of PDF documents can result in processing delays for generating responses.
* Dependency on External APIs: Reliance on Langchain and Google Gemini Pro Vision APIs means any changes or service downtime may impact platform functionality.

**Chapter 2: Literature Review**

The field of interactive document processing and natural language understanding has seen significant advancements in recent years. Key technologies such as Langchain and Google Gemini Pro Vision have played pivotal roles in enhancing the capabilities of document analysis and interaction. This section reviews existing literature and technologies relevant to the project, highlighting how our project builds upon or differs from previous work.

**Existing Literature and Technologies**

1. **Document Processing and Information Retrieval:** Traditional methods often rely on rule-based approaches or OCR for text extraction from PDFs, with recent advancements in libraries like PyPDF2 and Tesseract OCR improving accuracy.

Various techniques for text chunking and organization, including NLP-based segmentation, have been explored to enhance efficiency in information retrieval tasks.

1. **Vector Embeddings and AI Integration:** Vector embeddings like Word2Vec, GloVe, and advanced models such as BERT and GPT have revolutionized NLP tasks by capturing semantic relationships and context.

Conversational AI models like the GPT series and Dialogflow enable human-like interactions through text or speech, leveraging pre-trained language models and fine-tuning techniques.

**Project Contribution**

Our project integrates advanced text processing, vector embeddings, and conversational AI into a unified platform for querying multiple PDFs. Key differentiators include:

1. Unlike traditional methods, our project combines text extraction, chunking, vectorization, and conversational AI for interactive querying.
2. Integration of Advanced Tools has been done by incorporating Langchain and Google Gemini Pro Vision enhances document analysis, enabling more accurate and contextually relevant responses.
3. Utilizing Streamlit offers an intuitive platform for users to upload, manage, and interact with PDFs, enhancing the overall experience.
4. Our project handles multiple PDFs simultaneously, focusing on scalability to support substantial data volumes.
5. Seamless integration of text processing and conversational AI allows users to dynamically engage, receiving accurate answers based on PDF content.

This synthesis of technologies and methodologies represents a significant advancement in the fields of document processing and conversational AI, facilitating efficient knowledge extraction and interaction with textual data.

**Chapter-3: Architecture, Design, Methodology**

**Software Development Methodology:**

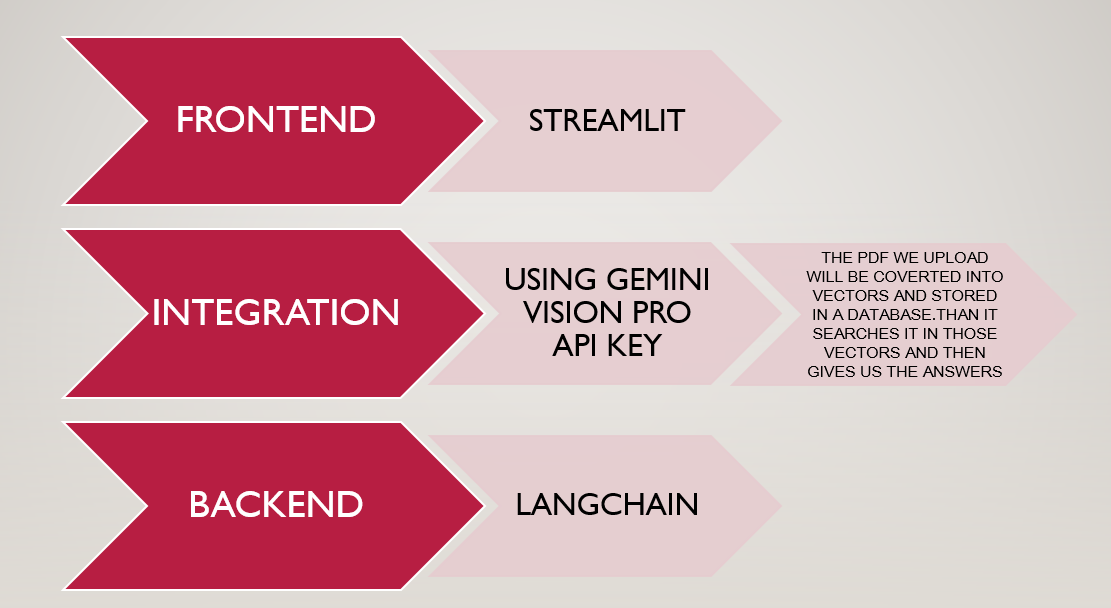
Our project adopts an Agile software development methodology. Agile allows for iterative development, enabling us to respond to feedback and adapt to changing requirements efficiently. It emphasizes collaboration, flexibility, and customer satisfaction throughout the development process.

**Tools, Languages, and Technologies:**

* Language: Python is the primary programming language used for development due to its versatility and extensive library support.
* Frameworks: Streamlit is utilized for building the user interface, providing an intuitive and interactive platform for users to interact with PDF documents.
* Libraries: PyPDF2 for text extraction from PDFs, Langchain for advanced text processing, Google Gemini Pro Vision for document analysis and question-answering capabilities.
* Development Environment: Visual Studio Code (VS Code) serves as the primary development environment

**Overview of the Project's Architecture and Design:**

* Frontend: The frontend is developed using Streamlit, providing a user-friendly interface for uploading, managing, and interacting with PDF documents. Users can input queries and receive responses seamlessly.
* Backend: The backend architecture includes modules for text extraction, chunking, vectorization, and conversational AI integration. PyPDF2 is employed for extracting text from PDFs, followed by Langchain for advanced text processing and Google Gemini Pro Vision for document analysis and question-answering capabilities.
* Integration: Streamlit interacts with the backend modules to facilitate a smooth user experience. Queries from users are processed through the backend modules, and responses are generated based on the content of the PDF documents.
* Scalability: The architecture is designed to handle multiple PDF documents simultaneously, with a focus on scalability to support substantial data volumes. Each component is optimized for efficiency and performance, ensuring seamless interaction with PDF content.



**Diagram: Architecture Diagram**

**Chapter-4: Implementation**

1. **Dependencies**

**Module Imports:**

1. streamlit for building the user interface.
2. PyPDF2 for extracting text from PDF files.
3. langchain for advanced text processing and vector embeddings.
4. langchain\_google\_genai for integrating Google Generative AI for conversational capabilities.
5. google.generativeai for configuring Google API key.
6. dotenv for loading environment variables.

**2) Frontend Code (Streamlit):**

The frontend is developed using Streamlit, a Python library for creating web applications.

The interface includes a text input field for users to input questions related to the content of uploaded PDF files.

Users can interact with the application by uploading PDF files and submitting questions.

**3) Model Code:**

The code includes functions for text extraction, chunking, vectorization, and conversational AI integration.

**get\_pdf\_text(pdf\_docs):** Extracts text from uploaded PDF files using PyPDF2.

**get\_text\_chunks(text):** Splits the extracted text into manageable chunks.

**get\_vector\_store(text\_chunks):** Converts text chunks into vectors using GoogleGenerativeAIEmbeddings and stores them in a FAISS vector store.

**get\_conversational\_chain():** Configures the conversational AI model using ChatGoogleGenerativeAI.

**4) Model Access Using API:**

The model uses the Gemini Vision Pro api key.

**APP LOGIC:**

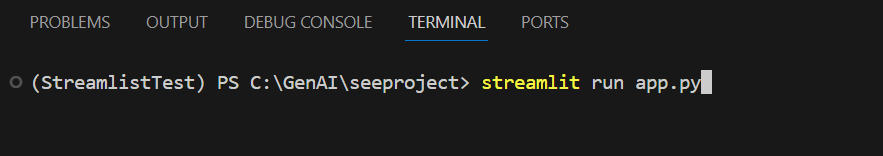
Upon uploading PDF files and submitting questions, the application executes the following steps -

* The uploaded PDF files are processed to extract text and segment it into chunks.
* Text chunks are converted into vectors and stored in FAISS. Then using pre-trained embeddings facilitates efficient similarity searches.
* User queries are processed against the generated vectors to retrieve contextually relevant answers from the configured conversational AI model.
* The application displays the retrieved answers to the user, providing a seamless interaction experience.

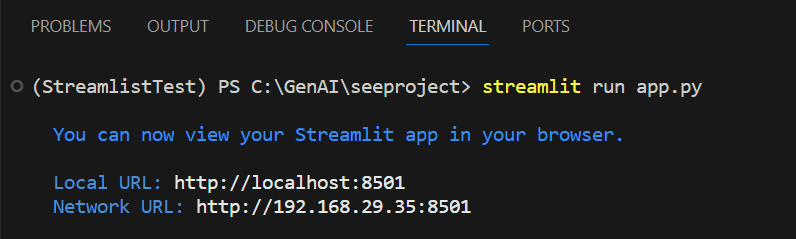
**How to Execute:**

1) To run the application, execute the following command in the terminal:

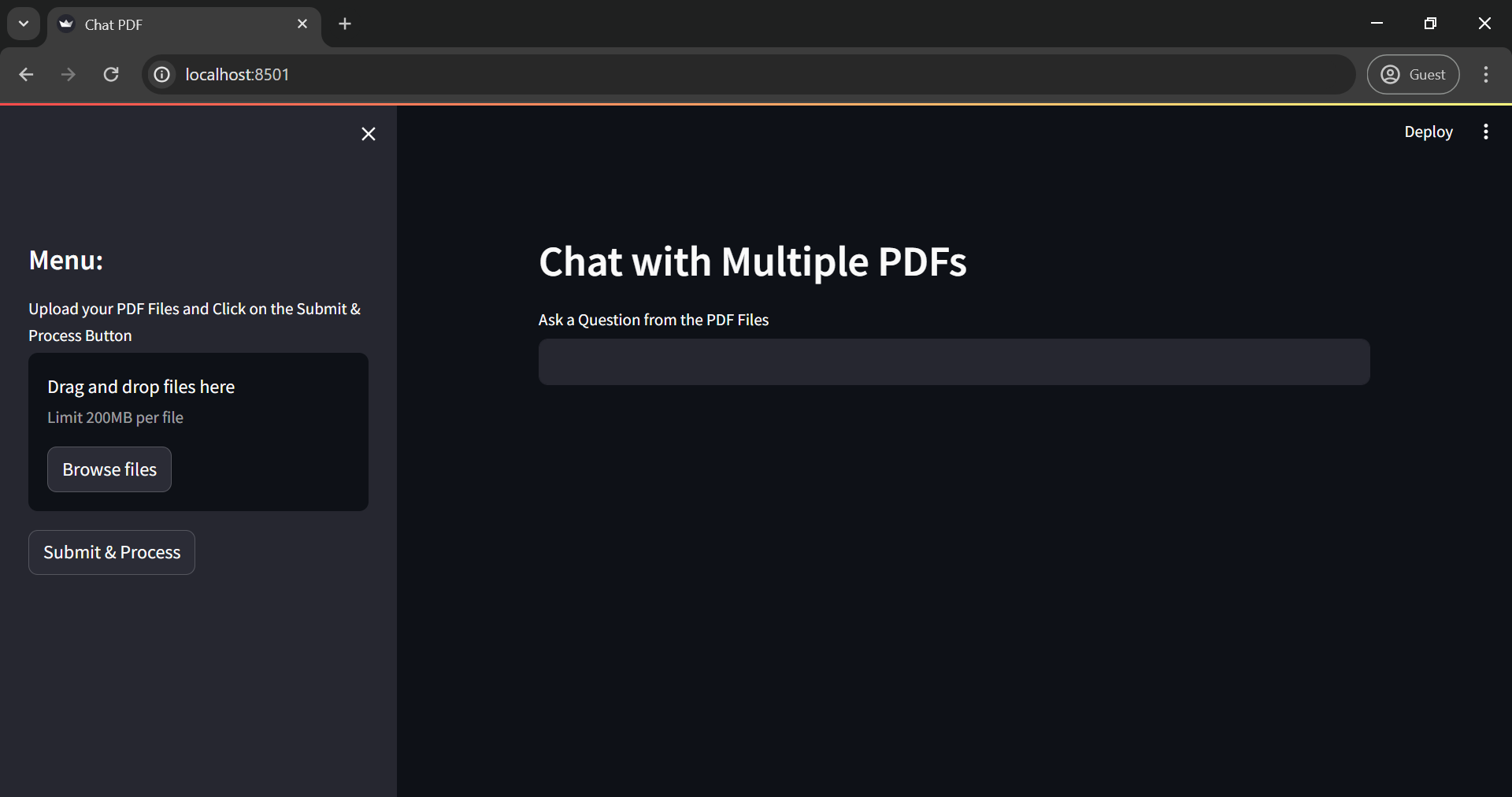
streamlit run <filename>.py



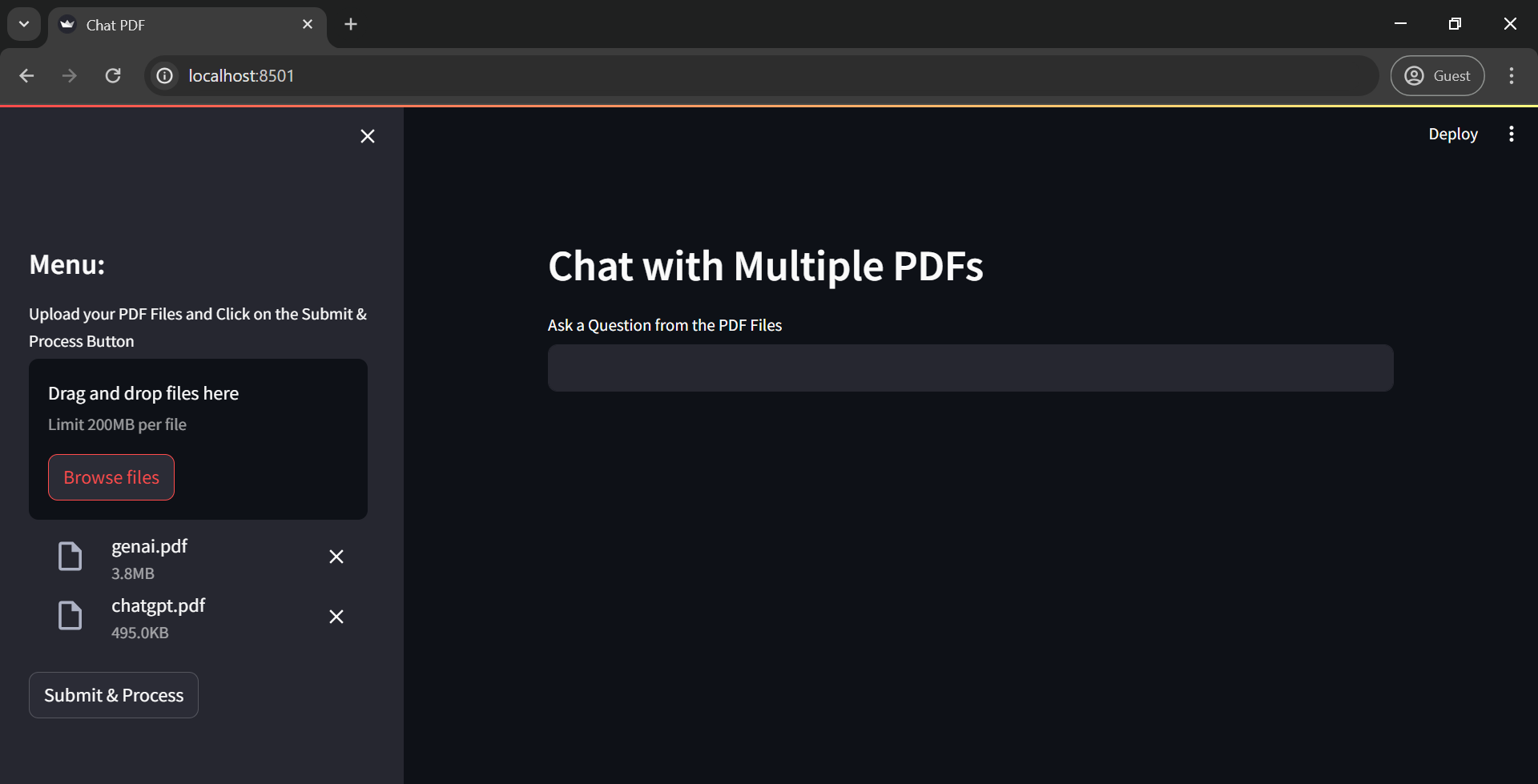
2) Access the application through the provided URL in the terminal or web browser.



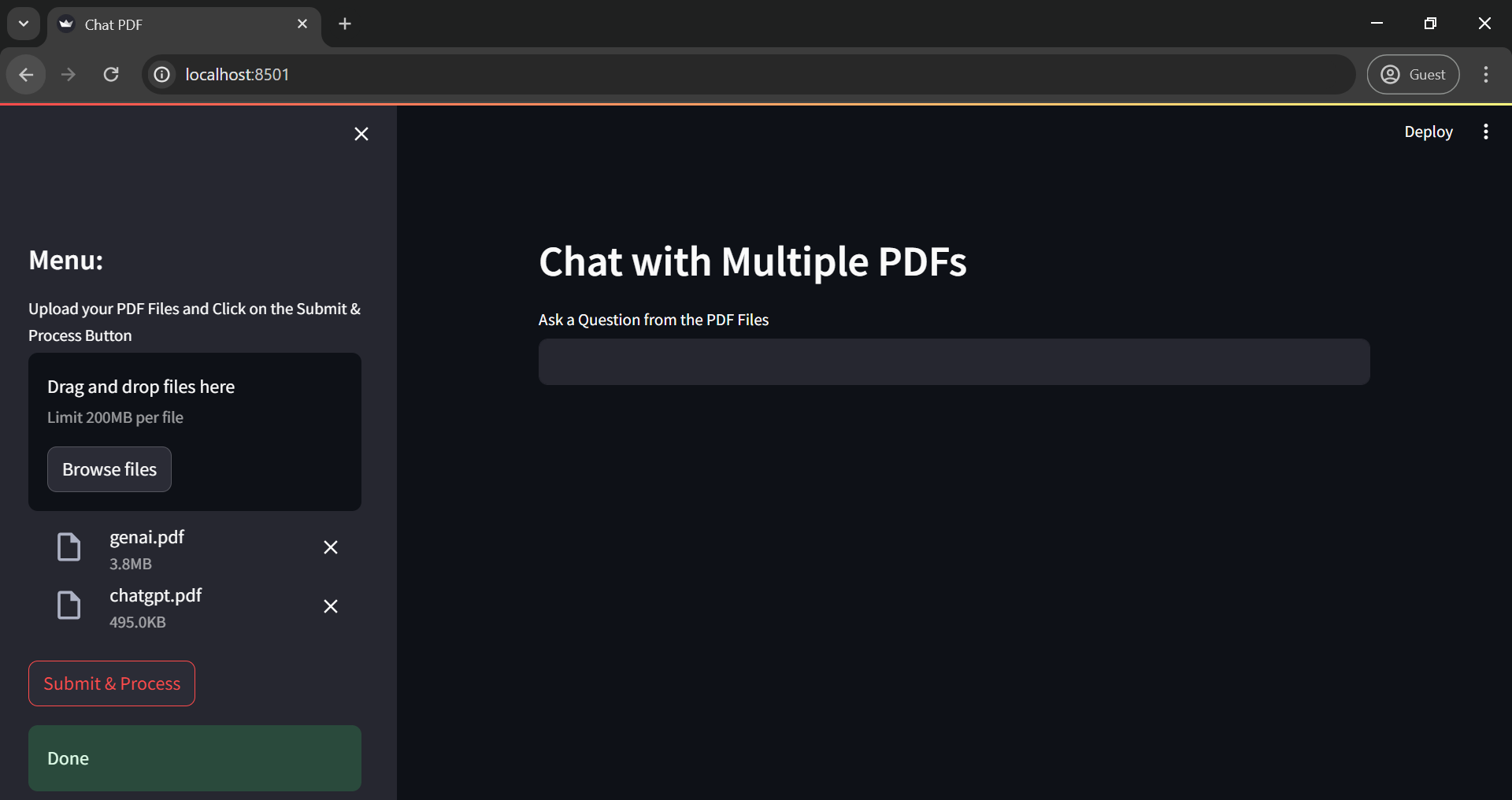
3) The User Interface of the application will be visible.



4) Upload PDF files

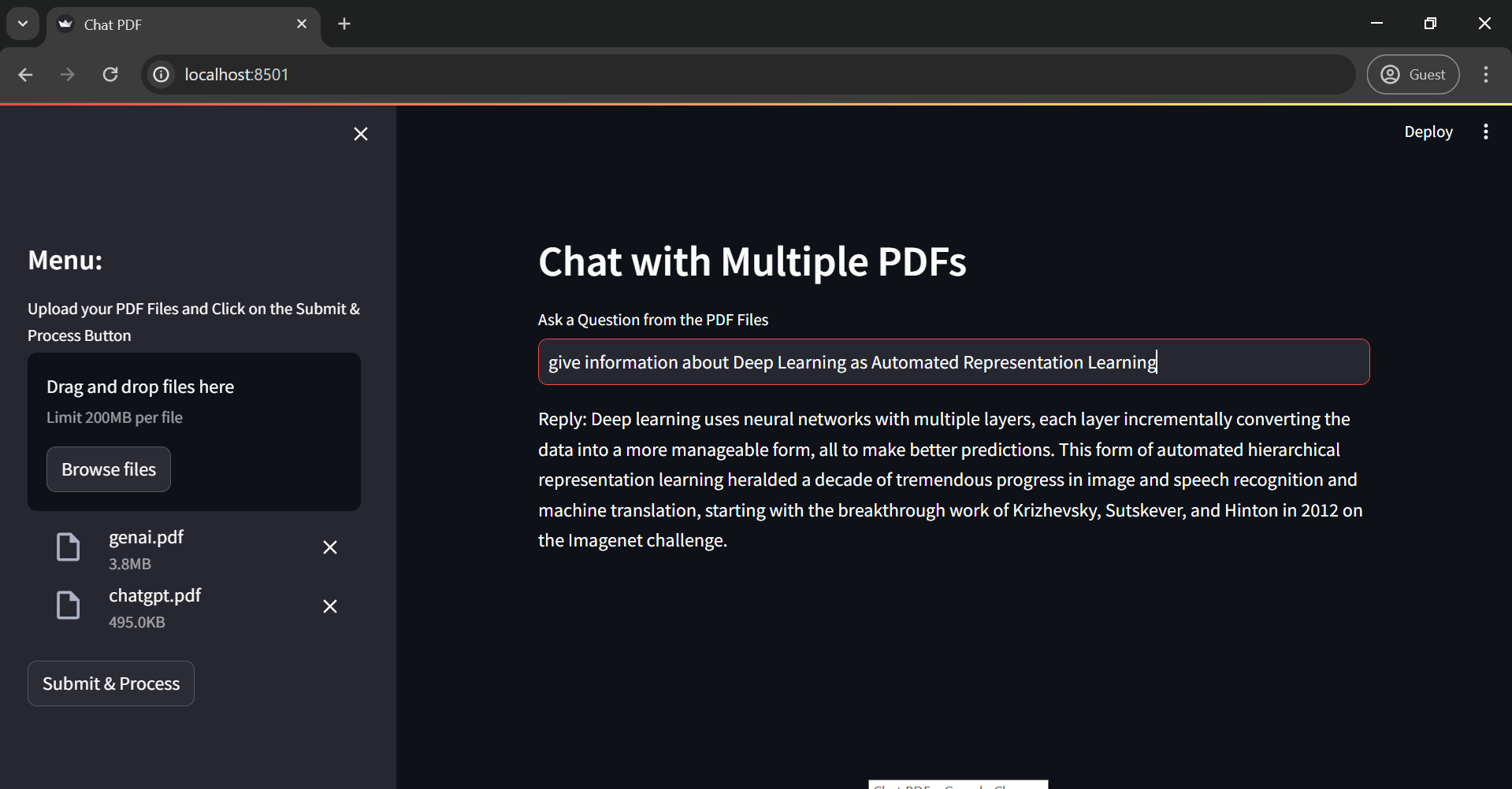


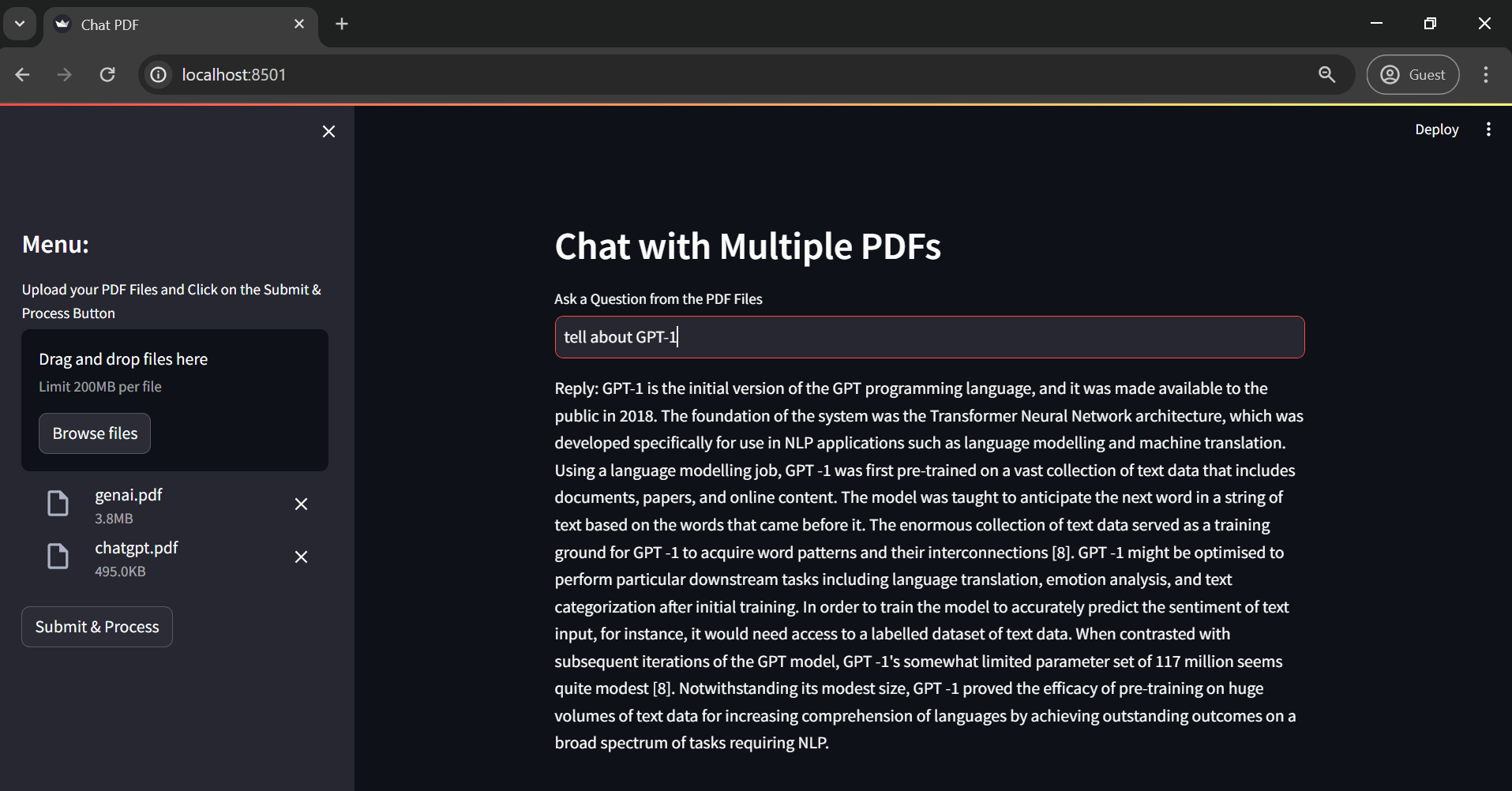
5) Click on the Submit and Process Button



After uploading the PDFs and clicking on “Submit and Process” button it shows as Done.

6) Input questions and interact with the application





**Challenges faced and solutions:**

During implementation, we encountered several challenges and devised solutions to overcome them:

1. **Text Extraction Complexity**

Extracting text from PDFs, which often have diverse structures and formats, proved challenging.

To address this, we relied on the PyPDF2 library and implemented robust error handling for consistent and reliable text extraction.

1. **Optimizing Text Chunking**

Determining the optimal chunk size and overlap for efficient text processing presented a hurdle.

We tackled this challenge by experimenting with different parameters using RecursiveCharacterTextSplitter, ensuring adaptable and effective text chunking.

1. **Integration of Google Generative AI**

Integrating Google Generative AI for conversational capabilities required careful configuration and seamless integration.

To achieve this, we extensively consulted documentation and community resources, ensuring a smooth integration process.

**Chapter-5: Test Result and Discussion**

**Presentation of Results by Feature:**

The performance of each implemented feature is presented systematically, by highlighting its functionality and its effectiveness in achieving the intended goals. This includes the accuracy of text extraction, the efficiency of text chunking, the quality of vector embeddings, and the responsiveness of the conversational AI model.

**Comparison with Project Objectives:**

The achieved results are compared against the predefined project objectives to evaluate the extent to which each objective has been met

**Discussion of Unexpected Findings:**

The model might sometimes say the answer is not found in the pdf which has to be resolved.

**Conclusion**

The project has successfully achieved its objectives and delivered valuable outcomes. Key highlights are:

**Summary of Project and Outcomes:**

The project aimed to develop a platform for interactive querying of multiple PDF documents, integrating advanced text processing, vector embeddings, and conversational AI technology. Through the systematic implementation, the platform now allows users to extract knowledge from PDFs through natural language queries.

**Achievements and Contributions:**

The project has successfully implemented features for text extraction, chunking, vectorization, and conversational AI integration, providing users with a seamless and intuitive experience. By leveraging advanced technologies and methodologies, the platform facilitates efficient information retrieval and fosters interactive engagement with PDF content.

**Suggestions for Future Work or Improvements:**

While the current version of the platform fulfills its primary objectives, there are opportunities for further enhancement. Future work could focus on refining the accuracy and efficiency, expanding the capabilities of the conversational AI model, and optimizing the user interface. Additionally, integration with additional document formats and language support could broaden the platform's applicability and user base.

In summary, the project represents a significant advancement in the field of document processing and conversational AI, offering a valuable tool for knowledge extraction and interactive querying of PDF documents.

**REFERENCES**

**Online tutorials:**

1. <https://www.analyticsvidhya.com/blog/2023/08/chat-with-pdfs/>
2. <https://anuragkmr.medium.com/building-a-simple-talk-to-pdf-chatbot-338f7ef1231f>
3. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10400373/>

**APPENDICES**

It can take upto 15 PDFs of size 200 MB each.

