VIDYAVARDHAKA COLLEGE OF ENGINEERING

Autonomous institution affiliated to

Visvesvaraya Technological University, Belagavi

PROJECT SYNOPSIS

Title: DocMind - An AI-Powered PDF Analysis System

Submitted by

Rakshitha B J 4VV22CS119

Renil Dharuni 4VV22CS123

Likhitha M 4VV22CS075

Lakshmi S 4VV23CS406

Under the Guidance of

Dr Ravikumar V
Professor
Department of Computer Science & Engineering

Project Title : DocMind - An AI-Powered PDF Analysis System

Abstract

DocMind is an AI-powered system that allows users to upload PDF documents and interact with them using natural language queries. Instead of manually reading through lengthy files, users can simply ask questions — and the system intelligently finds and presents accurate answers from the content. Using a technique called Retrieval-Augmented Generation (RAG), it combines modern AI tools like LangChain, LangGraph, vector databases, and Large Language Models (LLMs). DocMind is especially useful for students, researchers, legal professionals, and business users who frequently work with large PDF files and need quick access to specific information.

1. Introduction

In today's digital world, people deal with large amounts of PDF documents — reports, legal agreements, research papers, and manuals. Searching through these documents for specific answers can be time-consuming and frustrating. Traditional tools like Ctrl+F or keyword search often fail to understand the context or answer complex questions.

DocMind solves this problem by allowing users to chat with their PDF files. Using a combination of AI technologies, the system reads and understands uploaded documents, and then provides contextual answers in real time. For example, you can upload a 100-page policy document and ask: "What are the features of this System ?" — and DocMind will give you a precise answer with references from the original file.

This is made possible using a smart method called Retrieval-Augmented Generation (RAG), where the system first finds the most relevant parts of the document and then uses a language model to generate a clear and meaningful answer.

2. Objectives

The main goals of the DocMind project are:

- Allow users to upload PDF documents via a simple web interface.
- Break down and process the document into small, searchable chunks.
- Convert those chunks into vector embeddings using AI techniques.
- Use LangChain and vector search to find the most relevant parts for any user query.
- Use a Large Language Model (LLM) to answer questions based on retrieved content.
- Display answers clearly with source highlighting so users know where the information came from.
- Make the interface simple and intuitive so that non-technical users can use it easily.

3. Scope

- Uploading and analyzing PDF files only.
- Chunking document content and converting it into vector form.
- Using LangChain and LangGraph to build a conversational flow.
- Using a vector database (like FAISS or Chroma) for semantic search.
- Generating answers using a pre-trained LLM.
- Showing clear answers with references to original document text.

 Providing a chat-based interface for easy interaction**.**

4. Methodology

The methodology of this research is centered around the implementation of a Retrieval-Augmented Generation (RAG) framework to enable intelligent question-answering over PDF documents. The approach combines document preprocessing, semantic vector indexing, and large language model (LLM) inference to generate contextually grounded answers in response to user queries. The system architecture is modular and leverages state-of-the-art tools including LangChain, LangGraph, and vector databases.

4.1 Document Ingestion and Text Preprocessing

Upon upload, the system parses PDF documents using Python-based PDF extraction libraries such as PyMuPDF or pdfminer.six. The extracted text is then segmented into smaller coherent chunks (typically 200–500 tokens) to preserve semantic integrity. Each chunk is annotated with metadata such as page number and section heading for traceability.

4.2 Embedding Generation and Vector Storage

Each text chunk is transformed into a dense numerical vector using pre-trained sentence embedding models, such as OpenAI's text-embedding-ada-002 or Sentence-BERT. These embeddings capture the semantic content of the chunks and enable similarity-based retrieval. The resulting vectors are indexed and stored using a vector database (e.g., FAISS or ChromaDB) to support high-speed, approximate nearest neighbor search.

4.3 Query Processing and Semantic Retrieval

When a user submits a query, the system first converts it into an embedding using the same model as used for the document chunks. A vector similarity search is then performed to retrieve the top-*k* most relevant chunks from the embedded document corpus. These chunks represent the most semantically relevant contexts for answering the user's question.

4.4 Answer Generation via Retrieval-Augmented Generation (RAG)

The retrieved document contexts, along with the user query, are passed as input to a Large Language Model (LLM) such as GPT-3.5 or a local LLM deployed via LangChain. The LLM uses the provided context to generate a response that is both informative and **grounded in the**

original document. This RAG pipeline ensures factual accuracy and minimizes hallucination by the model. Where possible, source references (e.g., page numbers) are highlighted in the output to improve transparency and traceability**.**

4.5 Conversational Orchestration using LangChain and LangGraph

The interaction flow is managed through LangChain, a framework designed for chaining together LLM-based tasks. Additionally, LangGraph is employed to model conversational workflows as dynamic graphs, allowing for session management, prompt history, and future extension to multi-turn interactions.

4.6 User Interface and Deployment

The system is deployed as a web application using lightweight frameworks such as Streamlit or Gradio. The interface allows users to upload documents, pose natural language queries, and receive AI-generated answers along with highlighted source excerpts. This makes the system accessible to non-technical users in academic, legal, and enterprise contexts.

5. Expected Outcomes

By the end of the project, the following outcomes are expected:

- A fully functional web-based chatbot that users can use to chat with their PDFs.
- Accurate and relevant answers to user questions, backed by AI and document retrieval.
- Improved speed and ease of accessing information within long PDF documents.
- A transparent system that shows exact source references from the document.
- A flexible backend that can be extended for future use cases, like multi-document chat or summarization**.**

6. Conclusion

DocMind provides an innovative way to interact with documents by combining the power of AI and natural language processing. It saves time, reduces effort, and makes information inside PDFs more accessible and conversational. Whether it's legal documents, research papers, or business reports, DocMind makes it easier to ask questions and get meaningful answers — just like chatting with a knowledgeable assistant.

This system demonstrates how LangChain, vector search, and LLMs can work together to solve real-world problems using Retrieval-Augmented Generation (RAG). With further development, DocMind can evolve into a powerful tool for education, enterprise, and research workflows.

7. References:

[1] Z. Najwa, G. Mohamed, and N. Chafiq, “Revolutionizing Information Retrieval: Unveiling a Next-Generation AI-Powered Question-Answer System for Comprehensive Document Analysis,” Faculty of Science Ben M’sik, Hassan II University, Casablanca, Morocco,2024.

[2] H. Chase, “LangChain: Modular Abstractions for LLM Applications,” Whitepaper, 2023.

[3] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, Y. Wu, S. Edunov, and S. Riedel, "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, pp. 9459–9474, 2020.

[4] A. Khan, R. S. Kumar, and P. Bhargava, “Developing RAG-Based LLM Systems from PDFs: An Experience Report,” in *Proceedings of the International Workshop on Applied RAG Systems (AppRAG)*, Oct. 2024.

[5] A. D. Adimi, “Building a Multi-Agent Research Assistant with LangChain and LangGraph,” in *LangChain Community Blog*, Apr. 2025.

[6] Composio Research, “Building a Deep Research Agent Using LangGraph and Ollama,” *Composio Blog*, Apr. 2025.

[7] D. Lin, “Revolutionizing Retrieval-Augmented Generation with Enhanced PDF Structure Recognition,” , Jan. 2024.