DocuSense AI: Document AI conversational platform using GenAI.

Jyoti Madake,
Dept of E&TC Engineering,
Vishwakarma Institute of Technology,
Pune, India,
jyoti.madake@vit.edu

Kritika Raina,
Dept of E&TC Engineering,
Vishwakarma Institute of Technology,
Pune, India,
raina.kritika22@vit.edu

Akanksha Kamble
Dept of E&TC Engineering,
Vishwakarma Institute of Technology,
Pune, India,
akanksha.kamble22@vit.edu

Abstract

In modern academic, legal, and business settings, users frequently encounter the problem of reading extensive documents in PDF format. This problem is resolved by DocuSense AI, which powers a sophisticated web application with Generative AI, allowing users to engage with their documents using natural language queries. Built with Python, Lang Chain, and Llama 3.3, the application transforms static documents into interactive and intelligent dialogues. The system generates vector embeddings of the PDFs for quick querying, while the frontend interface is user-friendly and consists of HTML, CSS, and JavaScript. PostgreSQL is employed for secure username and password registration. DocuSense AI is tailored for students, scholars, and working professionals to enhance their comprehension and productivity. Literature reviews support the implementation of Generative AI in conjunction with semantic search and layout-aware models in the analysis of documents centred around human users. This solution changes the way documents are engaged with—making the process quicker, smarter, and more efficient.

Keywords-Generative AI, Llama 3.3, Lang Chain, PDF Querying, Natural Language Processing, Document Summarization, AI Chatbot, Document Interaction.

I. Introduction

In various academic, legal, and professional settings, individuals are frequently required to analyse large volumes of text within PDF documents such as research papers, contracts, reports, and technical manuals. Traditional methods of document analysis rely heavily on manual reading or simple keyword searches, which are often inefficient and cognitively exhausting. These methods fail to capture the contextual meaning behind user queries, leading to time-consuming and error-prone information extraction.

DocuSense AI presents a novel solution that leverages the capabilities of Generative Artificial Intelligence (AI) to enable conversational interaction with documents. Users can upload PDF files and interact with them using natural language queries, receiving intelligent and contextually relevant answers in real time. The system is developed using Python, Lang Chain, and the Llama 3.3 language model, which work in tandem to process documents, generate vector embeddings, and provide accurate responses.

The platform's architecture includes a clean and responsive frontend designed with HTML, CSS, and JavaScript, along with a secure login and registration system implemented using PostgreSQL. This ensures not only a user-friendly experience but also secure and scalable access to the application. By transforming static documents into interactive experiences, DocuSense AI simplifies information retrieval and enhances productivity across domains.

Existing literature highlights the importance of combining semantic search, layout understanding, and advanced NLP techniques to create efficient and human-centric document analysis tools. Frameworks like Lang Chain and models such as Llama and Layout LM have demonstrated significant promise in enabling context-aware, multimodal document interaction. This project builds on these advancements to provide a practical, real-world solution for intelligent document analysis.

DocuSense AI which is a new Generative AI driven platform enabling users to talk to PDF documents via natural language questions. It uses Lang Chain, Llama 3.3 and vectors embeddings to provide responses that are contextually accurate. The backend is developed in HTML, CSS and JavaScript while data security is achieved through the use of PostgreSQL. The solution improves understanding and searching of documents within academic and professional settings enhancing document analysis intelligence solutions gaining new levels of agility and responsiveness.

II. Literature Review

Liam Melin-Higgins [1] investigated AI-enhanced PDF readers based on an interaction-centred design methodology. Through prototyping and user testing, the research found that AI facilitates reading by enabling summarization, critique, and contextualization. Yet, users continue to rely on visual cues, indicating that AI should augment—not supplant—conventional ways of reading. The research guides user-oriented design of AI-supported reading technologies. Yeh et al. [2] suggested a future document reader with NLP-driven plug-ins to turn static documents into active, networked ones.

Their paper contains 18 suggested features, contextual pop-up menus, and an in-central plug-in store. The paper introduces UI prototypes that illustrate how customizable AI tools can facilitate greater understanding and user engagement while reading documents. Topsakal and Akinci [3] had suggested Lang Chain framework for quick application development for applications based on LLM. Authors described important components of Lang Chain—prompts, chains, memory, and agents—and demonstrated its modularity and efficacy through a number of real-world examples, which reflected its capability in creating scalable, context-sensitive, AI-driven solutions. [4] had suggested V-MoE, a sparse Vision Transformer with Mixture of Experts (MoE) for scaling vision models effectively. Their Batch Prioritized Routing algorithm suppresses computation through choosing salient image patches. V-MoE attains state-of-the-art performance while having lower inference costs, boasting excellent scalability, transfer learning, and fine-tuning across huge datasets such as ImageNet and JFT-300M. Chan and Lee [5] also studied differences in generations between Gen Z students and Gen X/Y teachers when it comes to embracing generative AI tools such as ChatGPT in tertiary education. The research determined that Gen Z was more hopeful and prolific users, while the educators were worried about abuse and academic honesty. Both cohorts highlighted the importance of proper instructions and AI literacy education. Khoramnejad and Hossain [6] review the use of generative AI (GAI) methods, such as GFlowNets, GANs, and diffusion models, for xG wireless network optimization. They discuss GAI contributions to mobile AIGC, ISAC, semantic communication, and network security, and provide a case study illustrating GAI-augmented resource allocation in non-terrestrial networks, and its future potential for data-driven xG network optimization. Sheremet et al.

[7] proposed an AI-driven system integrating ChatGPT with PDFs via LangChain that facilitates dynamic, conversation-like interactions between static documents. The system utilizes vector embeddings, retrieval-augmented generation, and conversational agents to improve user experience, data analysis, and literature review and is beneficial in customer support, academics, and secure document interaction. Mansurova et al. [8] created a domain-specialized chatbot for the blockchain community using GPT-3.5 as part of LangChain, Pinecone, and web search APIs. The system relies on both retrieval-augmented generation and external knowledge bases to improve response accuracy. It was found to be effective in user education and knowledge dissemination at scale when evaluated on the Digital Tenge project. Devaraj et al. [9] suggested a chatbot legal document using LangChain, GPT, and Flask-based REST APIs with an Android frontend. The framework semantically processes user queries using cosine similarity and provides relevant legal content. Optimized for legal documents such as court judgements and the Indian Constitution, it provides scalable, context-sensitive legal aid through an easy-to-use interface. Hammarström [10] developed a domain-specific QA system that integrates ChatGPT with hybrid document retrieval methods to assist factory workers in traversing in-house manuals. The system achieved 92% accuracy in response generation with semantic and TF-IDF retrievers, dynamic chunking, and retrieval-augmented generation. It demonstrates viable application of LLMs in manufacturing settings for carrying out document-based information retrieval effectively. Zhang et al. [11] suggested DialoGPT, a conversational model based on GPT-2 that was trained on 147M Reddit dialogue samples. It enhances response diversity and relevance with mutual information maximization and surpasses current baselines in human and automatic evaluations with robust open-domain dialogue generation capability. Bird et al. [12] presented the CI-AI framework, which allows the execution of tasks using chatbots through natural language. They paraphrased 483 human-annotated commands using T5 and trained seven transformer models. RoBERTa had 98.96% accuracy, while an ensemble had 99.59%, showing enhanced classification and ease of use for non-experts. Safari and Shamsfard [13] introduced PerInfEx, a Persian open-domain chatbot which extracts personal information implicitly from chit-chat. They introduced new data collection techniques, such as gamification and Conditional BERT-based augmentation, and achieved high performance on intent identification and slot filling with a ParsBERT-based NLU model .

Pandya and Holia [14] introduced Sahaay, an open-source chatbot platform based on LangChain and Flan-T5 models for customer service automation. It scrapes organizational information, embeds it with Instructor-large, and provides context-aware, real-time feedback. Flan-T5-XXL achieved better performance with scalable, customized support across sectors. Zafar et al. [15] introduced a privacy-conscious, explainable conversational AI architecture integrating LLMs, Knowledge Graphs, and Role-Based Access Control. They presented LLMXplorer—a cataloging of 150+ LLMs—and certified their system on journalism data, promoting trust, transparency, and contextual accuracy in real-world conversation applications. In [16], the authors introduce an end-to-end deep learning framework with Bi-LSTM and CRF layers for ADE extraction from social media text, remedying data imbalance and noisy language with transfer learning and loss function adaptation, with performance enhanced over multiple benchmark datasets. In [17], Ghaddar et al. make a comparative analysis of data augmentation methods in NLP, pointing out that text length affects the effectiveness of the augmentation. Lexical approaches are best suited to short-text classification, whereas contextual approaches such as back-translation and BERT-based augmentation best suit long-text classification. In [18], Sayeed et al. introduce a new method for semi-automated peer grading through complete fine-tuning of text-to-text generation models (e.g., Flan-T5) as discriminative classifiers, with 83% accuracy for identifying correct student answers on a low-resource, real-world educational dataset.

There are currently AI and other related PDF Readers that fail to show sufficient domain adaptability and to deliver real-time, interactive, and context-responsive user experiences. Many are currently static retrieval in a subset of domains, they deliver no dynamical summarization, contextual memory, or personalized learning. Existing systems also have limited modularization and user control. In this paper, we identify these issues, while introducing a domain-adaptive and scalable PDF assistant through the use of Lang Chain that combines semantic search, conversational artificial intelligence, and customizable plug-ins. The system allows users to interact and engage with documents in natural dialogue while maintaining contextual engagement and understanding, and affording greater user customization, validating the use of semi-automation in education, research, and enterprise applications.

III. Design and Methodology

DocuSense AI's methodology focuses on creating an interactive document experience from what is traditionally a document experience based on human interaction with documents. The tool uses modern technologies, such as Lang Chain, Llama 3.3, and vector embeddings to process the PDF document and interpret the user's queries to provide precise, meaningful answers. This methodology involves multiple steps that execute different pieces of the process to create an interactive platform for retrieving information instead of a static platform.

A. System level block diagram

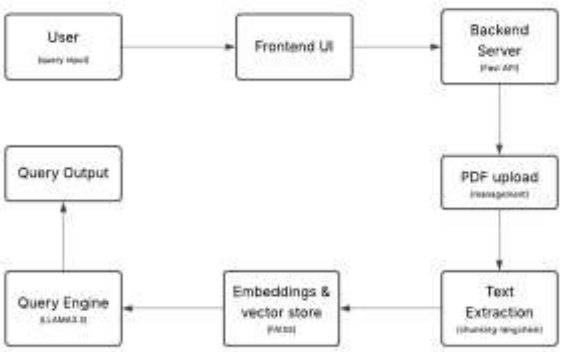

Fig. 1. System level block diagram of DocuSense AI

B. System Workflow Overview

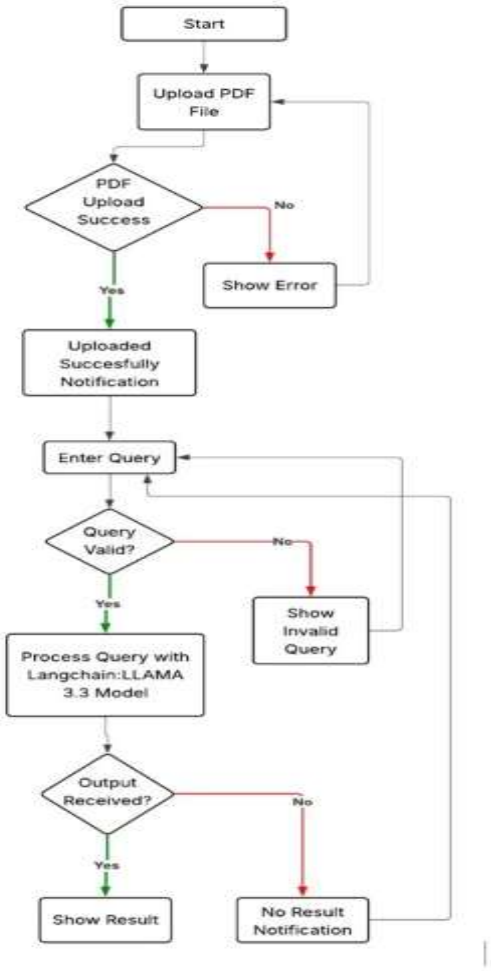

Fig. 2. Workflow of DocuSense AI

The DocuSense AI workflow starts with uploading a PDF document by the user and is completed is a workflow of steps that provides query able, actionable content. The workflow can described as having four components.

1.User Interaction: The user can upload their PDF document via a user-friendly web-based interface.

2.Document Processing: When the document is uploaded, it is parsed and the text is extracted, and then the document text is parsed into smaller sections of text to make it manageable for future queries.

3.User Querying: Users can then step up to the system and take some action (question) in natural language format (such as English) and the backend of the system finds the best match to the divided sections of the document for the queries.

4.Answer Generation: In this last phase, the response is generated from the content extracted from the PDF document, and answers are returned, based on contextually precise information to the query.

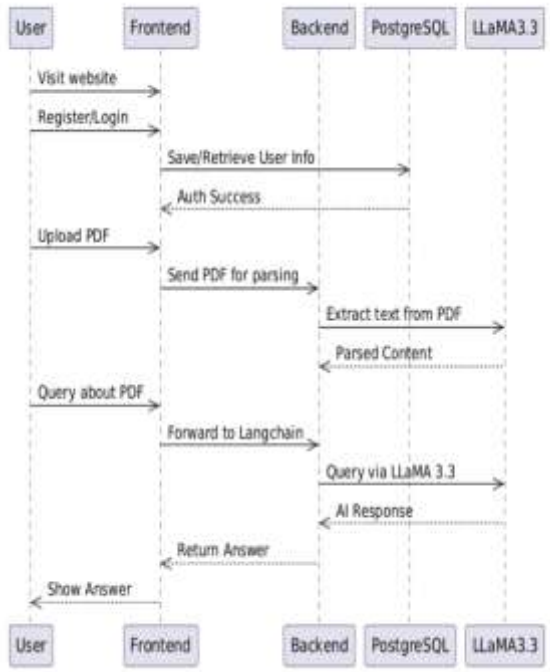

Fig. 3. Sequence diagram of DocuSense AI

B. PDF Parsing and Document Chunking

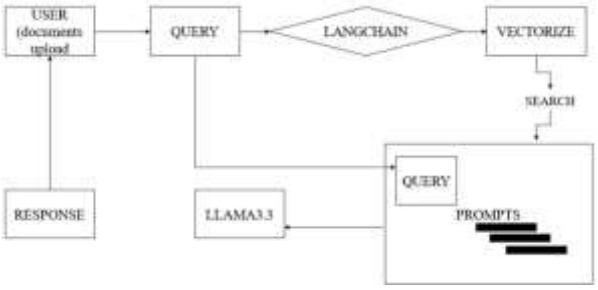

Fig. 4. Block diagram of PDF parsing

PDF parsing and document chunking, or dividing documents into smaller, semantically unimpaired sections, makes a large PDF document manageable for the system to work with. This process preserves the original meaning of the document while enabling the system to parse through the document as manageable chunks of text.

1.PDF Extraction: The first step is for the system to extract text from the uploaded PDF. Anything from PyMuPDF, as well as pdfminer work well, as these libraries can generally extract text from formatted and unformatted documents. With a substantially large PDF with extreme formatting, the extraction process can become cumbersome and tricky.

2.Text Chunking: After the completed extraction, the system will then chunk up the data into smaller sub-chunks, usually between 200–500 tokens. It is important to chunk the document as in its original state, the document length and semantic structure can be overwhelming for the system. While it is best for the system to generate long-form responses, separating large document into smaller manageable text, while ensuring that the sub-chunks have enough context to stand on their own for future retrieval is necessary.

3.Overlapping Context: The next step is to manage the overlap between the document's sections for maintaining context between the various sections of text that were extracted. Any managed transition between sections of text should be limited as document context is important for generating accurate results during retrieval of information.

System utilizes PyMuPDF tools to extract text while also preserving structural features such as tables and headings, from a PDF file. It then splits the extracted text into chunks of 500 tokens. In order to ensure continuity, it added an overlapping 50 token window on each chunk. This ensures continuity and captures important information like definitions that start in the previous chunk and continue in the next chunk. The system successfully provides factual and contextual responses to user queries.

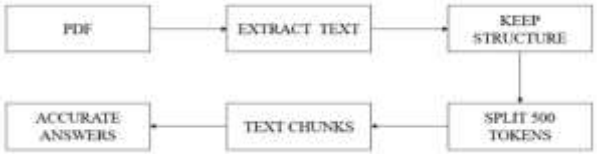

Fig. 5. Block diagram for text extraction in DocuSense AI

C. Embedding Generation Using Lang Chain

Once the document is chunked, each chunk is then vector embedded. The document embeddings represent the contextually meaningful information captured in the text of the document, allowing the system to find relevant content based on context and not merely matching keywords. The ability to generate meaningful and contextually aware responses to user queries relies on this step.

1.Text Embedding: Each chunk of text is converted using Lang Chain into a vector embedding. The vector embedding encodes the text as a high-dimensional vector that reflects what it means in order to compare chunks of text associated with proximity and semantic relevance.

When any chunk of text is embedded, each chunk is represented as a vector in some high dimensional space that encodes its meaning. A sentence about LangChain, for example, is encoded to some number vector representation. This implies that semantic relations can be made, which means the system has the capacity to identify relevant content even if the user queries it more abstractly - for example that "tools for developing LLM apps" refers to LangChain.

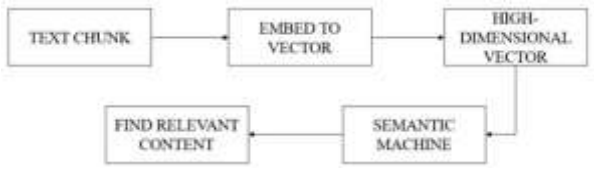

Fig. 6. Block diagram for text embeddings in DocuSense AI

2. Vector Database: The vectorized text chunks are then stored in purpose-built vector databases (e.g. FAISS or Chroma DB) for the sole purpose of retrieval functionality and design. These are optimized for locating and processing the data that would fulfill a user request.

A vector embedding is stored in vector databases such as FAISS or Chroma DB that allow for rapid searching based on similarity. When a user submits a query, its embedding is searched against the stored chunks. Consider the case when archived chunks on Lang Chain and FAISS were stored. A semantically relevant query would retrieve those chunks ,based on cosine distance or something similar.

Fig. 7. Flow of storing vectors

3.Semantic Search: When the user executes a query, the system conducts a semantic search by comparing the vectorized query to document embeddings that are stored. This allows the system to find the most relevant chunks based on what the user is asking which allows it to extract content that is contextually appropriate.

Semantic search identifies the most relevant content by turning the user's query into a vector in order to compare it to document embeddings. For instance, given a user query that does not use the exact same words, it will locate related chunks, if the query is still about the same issue, like integrating LangChain with vector databases. This approach is focused on the purpose of the question. It will ensure accurate responses, when intent is indicated, not just keywords.

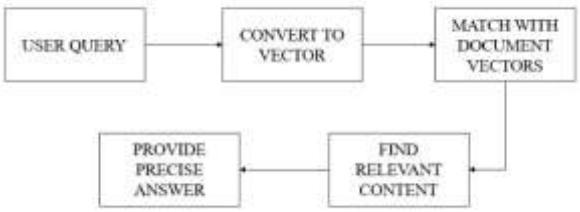

Fig. 8. Flow of query response in DocuSense AI

D. Natural Language Query Processing

Query processing phase allows the system to take the user's query typically in natural language, understand the intent behind the query and relate it to content in documents. During this phase, the query is transformed into a vector embedding - the document chunks will also have been converted into a vector embedding - and the vector embedding of the query can be compared against the vector embeddings of the various document chunks to execute a search and retrieval.

1. Query Vectorization: The user's query is first represented as a vector embedding, as well as the document embeddings are represented, representations that are used for the document chunks.

So when a user queries for example, "What are the benefits of Lang Chain in LLM systems?", the system is converting it to a high-dimensional vector using the same embedding model of the document chunks. Thus, it can relate and compare ideas in different but similar phrasing, e.g. "Lang Chain augments the LLM workflow by enabling structured pipelines."

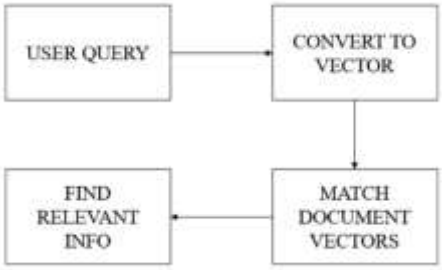

Fig. 9. Vectorization in DocuSense AI

2.Similarity Search: The system takes the query vector to perform a search in the vector database and return the document chunks that are most similar semantically. This is the step where the system is searching for the information that is most closely relevant, even if it was presented in a very different way from the document text.

The system does more than just make the query a vector and put that in the document vectors to find semantically related content. So, if given the prompt "How do Lang Chain interact with vector databases?", it could pull the chunks on Lang Chain using FAISS or Chroma DB, even if those words did not exist - meaning having some match with other different words, and taking meaning and not words.

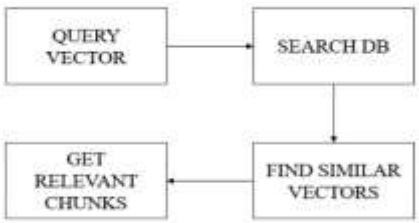

Fig. 10. Similarity search in DocuSense AI

3. Contextual Answering: Once the relevant chunks are identified these are passed to the llama 3.3 model to create an answer using the surrounding context of the text selected adjacent to the identified relevant text. This approach enables the system to create answers which will be specific, relevant and informed by the context of the user's query. Llama 3.3 takes the retrieved relevant chunks to create context relevant answer. An example we could use would be if the chunks mentioned Lang Chain embeds vectors to enable semantic search, Llama 3.3 could identify this and respond by saying, "Lang Chain allows for intelligent searching of documents by embedding the documents and the queries and semantically matching them".

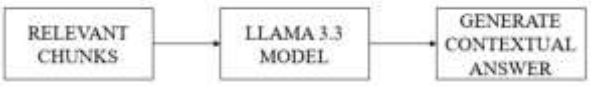

Fig. 11. Response generation in DocuSense AI

E. Generative Answering with Llama 3.3

The strength of DocuSense AI's ability to generate relevant and accurate answers lays in the Llama 3.3 language model. This model processes complicated textual information and produces answers that demonstrate a full understanding of the content. It is also incorporated into the system to ensure that the two-variable response is variable and contextual. Llama 3.3 has been released in a number of configurations, with various model sizes ranging from 8 billion to 65 billion parameters, affording freedom to deploy models suitable individually for the application in terms of resources and performance. The model has been trained on a large and diverse corpus of over 15 trillion tokens across several domains (scientific articles, technical reports, codebases, etc.). To support generalization ability across domains, it has been trained on a very large corpus.

The architecture is transformer-based with attention mechanisms, rotary positional embeddings (RoPE), grouped-query attention (GQA) for efficient inference, and does use sliding-window attention in certain configurations for long-context circumstances. Llama 3.3 has also been tuned for multi-turn dialogue to support multi-step reasoning in a coherent and contextually accurate way.

For deployment, Llama 3.3 has FP16 and INT8 quantization, which greatly reduces memory usage and inference time, making it ideal for use cases in both cloud and edge-based applications. This release uses Byte Pair Encoding (BPE) for a tokenizer that effectively handles various input languages and formats. Due to these advanced features, cumulatively, Llama 3.3 is composed to produce high-quality responses with a richer insight of user intent, thus being a vigorous generative applications engine such as DocuSense AI.

1. Model Integration: llama 3.3 helps-to-process the document portions acquired from the semantic search and outputs accurate, contextual response to user questions. For example, suppose a user asks "how does DocuSense AI keep their documents private when analyzing them?" In this case, document chunks that referiness certain privacy = safeguarding methods (e.g., on-device embeddings, secure vector storage) would be pulled from the document, and the model would synthesize a response from these different chunks - maybe one chunk refers to encryption and another refers to access control, then the model reads and relates that information such that the model produces a coherent response e.g., "DocuSense AI keeps privacy by running documents on devices and only storing vector representations in security aware databases - such that we are kept to minimum exposure to raw text content". This is a clear example of how llama 3.3 pulls and combines information from different parts of the document to produce a sensible response.

2.Prompt Engineering: Lang Chain is used to do prompt engineering in order to build relevant, high-quality output responses that are, structured, and context-rich to instruct this Llama 3.3 model during inference. Using this approach allows us to provide logical reasoning to the model so that it can produce contextually accurate responses to the user based on their question. The content may not be instructive, for example, instead of sending a simple prompt like "Explain vector databases," the system will make a better prompt dynamically, like: "You are an AI assistant helping users understand technical systems." From the following text chunks on FAISS and Chroma DB, explain in simple terms what a vector database is and how it relates to semantic search." The prompt provided with context, role, and goal, allows the user prompted the model to respond with what would be well structured output response. The example output may resemble the following: "A vector database is a database that stores numeric representations (called embeddings) of text, which allows the system to find and retrieve the most relevant/meaningful information based on meaning and not just on keywords. Facilitators like FAISS and Chroma DB help to perform the semantic search efficiently and quickly. So this is an example of how the prompt engineer technique allows llama 3.3 to not only understand the user's intent but to also provide a refined, targeted, and informative response.

IV. Results and Discussions

The DocuSense AI platform set out to look at how one can make interactions with documents more efficient using Generative AI and natural language processing capabilities (NLP). The platform aims to improve how users manipulate documents with an easy-to-use interface, efficient query processing, and document evaluation. The following section outlines conclusions drawn from testing and evaluation of the DocuSense AI platform, and then detailed discussion about the core functionalities and the contributions from this work.

A. Performance of Core Features

1.Login and Registration Process

The login and registration page was kept simple to allow for easy authentication. The website follows standard security protocols and encrypts user credentials and data to keep it

private. We tested the system by attempting to log in multiple times to see if our system was able to handle multiple logins and how quickly they were authenticated. Overall, our testing showed that many login attempts did not affect the ability of the system to let users access in a timely manner.

Fig. 12. Sign in page of DocuSense AI

2.PDF Upload and Processing

Users could upload PDF files with ease, and the system monitored their uploads in real-time. Different kinds of PDF files were accepted by the system—basic text or more advanced files that had images and tables within them. After uploading the PDF, the document was processed where the text was extracted from the document and broken into pieces of manageable text. The process of chunking attempts to maintain context while enabling users to query documents that are of varying structures. During the test phases, it was always correct in processing documents, be it the size and formatting.

Fig. 13. Home page of DocuSense AI

3.Query Interface and Answer Generation

The query interface allows users to ask questions using natural language phrase. The system finds the most relevant document segments and provides answers based on the question submitted. In tests, this system can handle a wide range of queries and return appropriate and accurate responses with relevant context. For example, we were able to query legal documents and find specific clauses and terms. The integrated NLP models provided an effective way to return content to the user based on their query.

Fig. 14. Chat interface of DocuSense AI

4.Answer Display

The answers are displayed clearly and outright below the user's inquiry. Where a particular source had to be cited, the platform provides the citation with a link to the specific document section. This feature brings transparency to users and creates trust in the answers made by the system. According to user feedback during testing, the answers were easy to comprehend, thus signifying the appropriateness of the language model (Llama 3.3).

B. System Performance Evaluation

The performance evaluation of the system was based on response time, accuracy, and scalability:

1. Response Time: On average, the time taken by the system to provide an answer is less than 2 seconds, which has established an easy and efficient user experience. For records that are significantly larger, we found that this response time was robust due to the superior efficiency of the vector database and the approaches taken to create their embeddings.

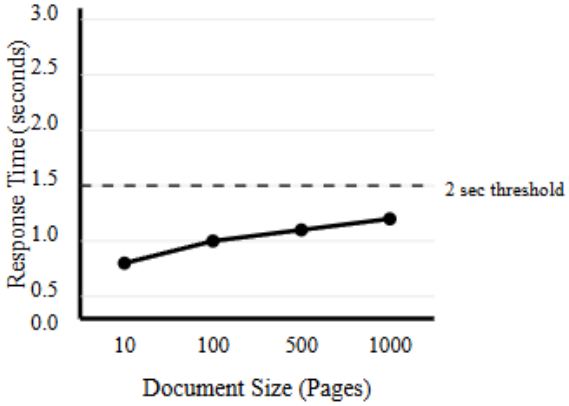

Fig. 15. Response time graph for DocuSense AI

2. Accuracy: The evaluation of accuracy refers to how well the system is able to respond to a request based on the query issued to the system. The documents under consideration were research papers, agreements, and technical reports, and in all cases the system was able to provide appropriate and accurate replies/answers, even with extensive and complex documents. We believe this can be attributed to the techniques employed in creating chunks of the relative records and the method for semantic search.

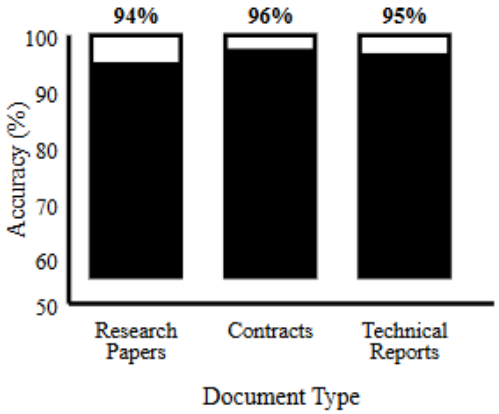

Fig. 16. Accuracy graph for DocuSense AI

3. Scalability: During internal testing, the system proved capable of maintaining performance with documents up to 1000 pages in length, with large datasets as a result. While we still believe it could be made more efficient, we also believe we have made it viable for varying use-case applications in both academic and professional environments. FAISS implementation for vector databases makes chunk retrieval efficient based on the user's preferences.

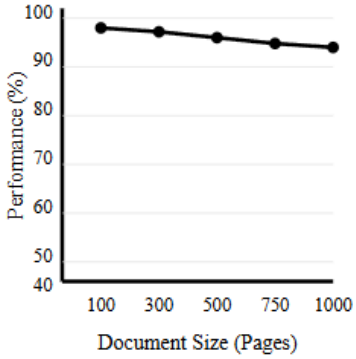

Fig. 17. Scalability graph for DocuSense AI

C. User Experience and Interface Design

The Responsive UI of the platform is one of its major strengths, because it provides full usability across multiple devices; for example, it provides functionality and usability on desktops, tablets, and mobile phones. During our testing, responders noted that the interface was smooth and they did not have difficulty navigating through the interface. It is important to also mention that the platform works on multiple screen sizes, devices, and still works clearly and was very easy to use in every case examined. The format that the platform is designed in, looks modern and clean for end users, and this is a benefit. End users are able to upload documents easily, query the content and then view the results without difficulty.

1.Key Features Evaluated:

1.1 Simple Navigation: The interface is simple and intuitive, allowing users to upload documents, submit queries, and view results quickly and without issue.

1.2 Real-Time Feedback: The system also provides progress indicators when document uploads and queries are handled which keeps users informed and satisfied.

D. Limitations and Areas for Improvement

DocuSense AI has strong capabilities for a user, there are some improvements that could be made:

1.Document Types that are Complex: The system can correctly identify text-based documents. However, it may struggle with PDFs that contain many images or that have a complex layout (e.g., scanned documents). Improvements to optical character recognition (OCR) or incorporating multimodal processing could possibly mitigate this.

2.Ambiguous Queries: The platform is best suited for queries that are specific and clear. It may struggle with vague queries or poorly worded questions. Improvements for processing ambiguous queries would improve its robustness.

3.Multi-Language Capability: While designed for English, it may be possible extend its use into other languages, thus enhancing its globalization capabilities.

V. Conclusion and Future scope

The DocuSense AI project successfully delivers an innovative, easy-to-use document query option that transforms the way users converse with PDF documents using natural language. With a stylish interface, secure authentication process, and cutting-edge AI technology, including LangChain and Llama 3.3, DocuSense AI—a document query platform—allows users to converse about documents using natural language; merging static content with dynamic comprehension. The platform eliminates the hassle of manual searching by providing real-time, context-sensitive response based on the uploaded material, it also supports multiple files uploaded simultaneously, history of session queries, and solid data privacy. DocuSense AI is envisioned to be helpful for students, researchers, professionals, or anyone who has to manage big amounts of information and is effective and easy to use. In the next iteration of the DocuSense AI document query platform, more functionality will be added with additional file types like

DOCX, TXT and image documents with OCR capability. There are mobile applications planned for Android and iOS to further reduce barriers to use and access, and to have voice query interaction for hands free searching. Additional future iterations will see features like sharing documents and group document chats, as well as localized offline capability through intelligent caching. These substantive improvements will solidify DocuSense AI as a holistic, AI-based assistant for documents search and information extraction.

VI. References

1. Melin-Higgins, L. (2024). AI in Document Interaction: An Interaction Design Approach to Enhancing Reading Practices.

2. Yeh, C., Lipka, N., & Dernoncourt, F. (2023). Envisioning the Next-Gen Document Reader. arXiv preprint arXiv:2302.07492.

3. Topsakal, O., & Akinci, T. C. (2023, July). Creating large language model applications utilizing langchain: A primer on developing llm apps fast. In International Conference on Applied Engineering and Natural Sciences (Vol. 1, No. 1, pp. 1050-1056).

4. Riquelme, C., Puigcerver, J., Mustafa, B., Neumann, M., Jenatton, R., Susano Pinto, A., ... & Houlsby, N. (2021). Scaling vision with sparse mixture of experts. *Advances in Neural Information Processing Systems*, *34*, 8583-8595.

5. Chan, C. K. Y., & Lee, K. K. (2023). The AI generation gap: Are Gen Z students more interested in adopting generative AI such as ChatGPT in teaching and learning than their Gen X and millennial generation teachers?. *Smart learning environments*, *10*(1), 60.

6. Khoramnejad, F., & Hossain, E. (2025). Generative AI for the optimization of next-generation wireless networks: Basics, state-of-the-art, and open challenges. *IEEE Communications Surveys & Tutorials*.

7. Sheremet, O. I., Sadovoi, O. V., Sheremet, K. S., & Sokhina, Y. V. (2024). Effective documentation practices for enhancing user interaction through GPT-powered conversational interfaces. AAIT, 7(2), 135-150.

8. Mansurova, A., Nugumanova, A., & Makhambetova, Z. (2023). Development of a question answering chatbot for blockchain domain. Scientific Journal of Astana IT University, 27-40.

9. Devaraj, P. N., PV, R. T., & Gangrade, A. (2023). Development of a Legal Document AI-Chatbot. arXiv preprint arXiv:2311.12719.

10. Hammarström, M. (2023). Exploring Knowledge Vaults with ChatGPT: A Domain-Driven Natural Language Approach to Document-Based Answer Retrieval.

11. Zhang, Y., Sun, S., Galley, M., Chen, Y. C., Brockett, C., Gao, X., ... & Dolan, B. (2019). Dialogpt: Large-scale generative pre-training for conversational response generation. arXiv preprint arXiv:1911.00536

12. Bird, J. J., Ekárt, A., & Faria, D. R. (2023). Chatbot Interaction with Artificial Intelligence: human data augmentation with T5 and language transformer ensemble for text classification. Journal of Ambient Intelligence and Humanized Computing, 14(4), 3129-3144.

13. Safari, P., & Shamsfard, M. (2024). Data augmentation and preparation process of perinfex: A persian chatbot with the ability of information extraction. IEEE Access, 12, 19158-19180.

14. Pandya, K., & Holia, M. (2023). Automating Customer Service using LangChain: Building custom open-source GPT Chatbot for organizations. arXiv preprint arXiv:2310.05421

15. Zafar, A., Parthasarathy, V. B., Van, C. L., Shahid, S., & Shahid, A. (2023). Building trust in conversational ai: A comprehensive review and solution architecture for explainable, privacy-aware systems using llms and knowledge graph. arXiv preprint arXiv:2308.13534

16. Taneja, K., & Goel, A. K. (2025). MuDoC: An Interactive Multimodal Document-grounded Conversational AI System. arXiv preprint arXiv:2502.09843.

17. Bayer, M., Kaufhold, M. A., Buchhold, B., Keller, M., Dallmeyer, J., & Reuter, C. (2023). Data augmentation in natural language processing: a novel text generation approach for long and short text classifiers. International journal of machine learning and cybernetics, 14(1), 135-150.

18. Sayeed, M. A., Gupta, D., & Kanjirangat, V. (2025). Engineering Text-to-text Generation Language Models as Discriminative Classifiers for Accurate Answer Detection. Procedia Computer Science, 258, 2930-2947.