e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:07/July-2024 Impact Factor- 7.868 www.irjmets.com

CHAT WITH YOUR DOCUMENTS

Sumit[*1], Aman Raj[*2], Rohit Sah[*3]

[*1,2,3]AI & ML Student, Vivekananda School Of Engineering And Technology, India.

ABSTRACT

In modern world, data has become incredibly valuable, sometimes it leading to apprehension about sharing personal information with online AI models or websites. One of the most widely used AI functionalities is Retrieval Augmented Generation (RAG), which serves as a question & answer chatbot for documents. With RAG, users can submit various document formats like PDFs, Excel sheets and ask questions related to the content and get the relevant answer. Rather than providing the same answer, which is already present in document, RAG generates customized answers using Large Language Models (LLMs) based on the user's queries.

RAG falls under the category of Generative AI, uses Large Language Models (LLMs) to generate responses based on their vast knowledge base. This capability enables RAG to provide answers about those queries which do not even specify in the document. Given the critical importance of data privacy, our project aims to develop a secure and locally hosted solution to protect user confidentiality.

Our proposed solution involves using a local LLM model that operates on the user's Personal Computer (PC), which ensures the data privacy and security. More important, the project is designed to function independently without relying on an internet connection. Additionally, for users whose PCs may not support the LLM model or have low hardware configuration, we offer an alternative solution—a user-friendly Application Programming Interface (API) provided by Open AI or Hugging Face. Through this API, users can seamlessly integrate their own private API key to run the RAG application on their system, ensuring accessibility and convenience while prioritizing data privacy.

I. INTRODUCTION

In the digital landscape, where data have the highest priority, the utilization of Artificial Intelligence (AI) models for document analysis and summarization presents both opportunities and challenges. As individuals or an organization wants to extract valuable information from their large size documents in minimum time while preserving sensitive information. In this context, Retrieval Augmented Generation (RAG) emerges as a promising AI feature, offering a dynamic and private approach to document interaction in real time.

The amount of data available in any document format is voluminous, from PDFs to Excel sheets and word files, hence the need for efficient document management techniques. Traditional methods mostly don't answer the challenges of modern-day information retrieval, which they rely on static search or manual scanning methods for searching in the documents.

RAG represents a transformative advancement in document analysis, mainly serving as a question & answer bot to documents. By using the vast knowledge base of the Large Language Models (LLMs), RAG enables users to engage and chat with their document. Unlike old or traditional methods that provide static responses, RAG generates dynamic answers by using the power of (LLMs) based on user queries, enhancing the engagement of users to their documents.

More importantly, RAG functions with the assurance of user privacy and data security. In ensuring that document analysis is locally and confidentially done, RAG Online reduces the risks related to providing sensitive information to AI assistants or online chatbot. This approach will be in line with the increased emphasis on data privacy rights laws that allow people and organizations to leverage the power of AI without infringing on privacy.

In this research paper, we explore the development and implementation of a private and locally hosted RAG solution. The objective is to give users a secure and efficient mechanism for working with their documents while keeping their data private. We propose a system architecture that enables local LLM models on users' Personal Computers to process the documents, thus eliminating the need for data transfer to an external server. Furthermore, for those who have low hardware PCs, we offer an API option for these users, hence making it something everybody can benefit from without compromising on privacy.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:07/July-2024 Impact Factor- 7.868 www.irjmets.com

This research paper will add to the existing debate over AI-driven document analysis through in-depth analysis of RAG's capabilities, privacy concerns, and strategies of implementation.

We want to empower each user and organization to realize the full power of artificial intelligence in document interaction while preserving the most valuable asset—data. The deployment of Artificial Intelligence (AI) models to analyze documents is beset with challenges and opportunities. While individuals and organizations want to extract important information from their documents without having their sensitive information exposed, innovative and privacy-sensitive AI solutions are at the forefront of these challenges. In that context, one of the most promising AI features is the Retrieval Augmented Generation, which has just emerged as a dynamic and private approach to the interaction with documents.

Figure 1: Global data generated annually, Image source: datapine blog

II. WHAT IS THE RAG SYSTEM?

The retrieval-augmented generation, it is a technique used in natural language processing and machine learning, aims at fusing the benefits of retrieval-based and generative models mainly (LLMs) to improve the quality of the generated text. This technique, therefore, finds critical applications in question answering with documents, document summarization, and for chatbot-like conversations.

The RAG approach represents an approach to the enhancement of response quality by using large language models, such as ChatGPT, Mixtral, and phi. It combines two main elements: retrieval and generation. Therefore, the responsibility of the retrieval component is to go through a huge knowledge base or database in order to retrieve relevant information. That means a search under the huge knowledge database of LLMs. What's the goal? The goal is to retrieve the most relevant content that has the best chance of containing information to fill a user's query. So essentially, it is like RAG system first searching for the answers relevant to a user's query in the given document. After the retrieval component retrieves the needed information, it feeds it into the generation component with the input prompt. And then magic will happen! The generation component which has a LLMs crafts a response taking the input prompt and the retrieved documents. The generation component can use context provided by the retrieved documents in order to construct responses which not only are accurate but also more informative and relevant for the conversation.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:07/July-2024 Impact Factor- 7.868 www.irjmets.com

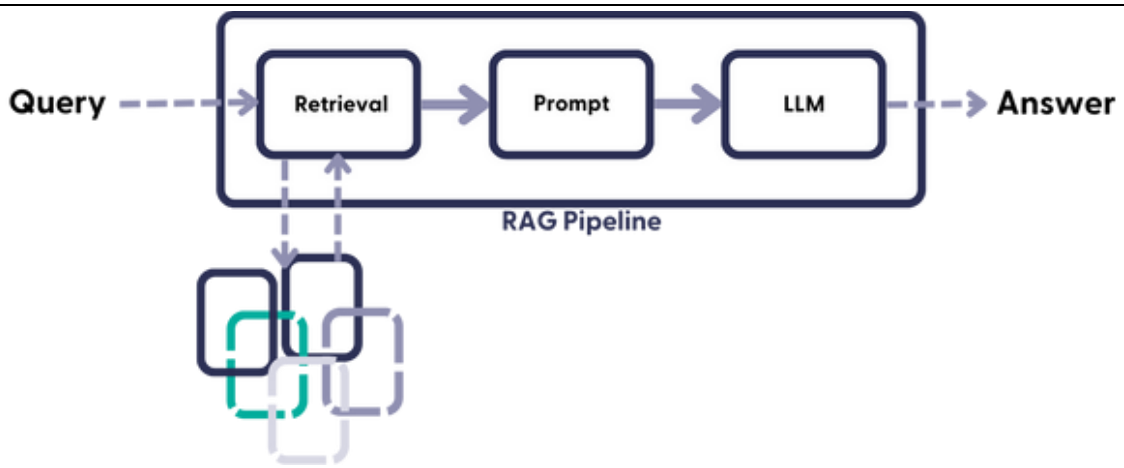

Figure 2: Overview of RAG Pipeline, Image Source: haystack.deepset.ai/blog/rag-deployment

2.1 How RAG Application works?

Data Chunking: Data chunking involves breaking down the given documents into small chunks. which means if chunk size is 200, the first 200 words is considered as chunk 1 and next 200 words considered as chunk 2 and so on. The chunks, therefore, become the blocks of retrieval and generation.

Vector Embeddings: Machine can only understand in terms of numbers so, in this part each word is converted into the form of numbers, each data chunk is vectorized as an embedding. These embeddings capture the semantic meaning of the content within each chunk. Techniques which generally used are word embeddings, sentence embeddings, or paragraph embeddings.

Vector Database: After the vector embedding all the vectors are stored inside the vector database. Vector database stores all the vectors or data points in multi-dimensional space. When a user types a query, the system processes it do vector embedding of the user's query and find the relevant data chunks inside the vector database based on the context of the query. They get stored based on the similarities of the words.

Semantic Search: Given the user's query and the vector embeddings of the data chunks, semantic searching is performed. It discovers the most relevant chunks for a given query in terms of semantic similarity. Techniques can be cosine similarity or semantic hashing for matching in embeddings.

Ranked results: The most relevant chunks of data are recovered from the vector database, carrying certain information on the user query.

LLM-based Generative Answer: The generative part; it fuses the retrieved chunks of data with the knowledge of the pre-trained LLM, such as PHI or Mixral and provide the AI generated answers.

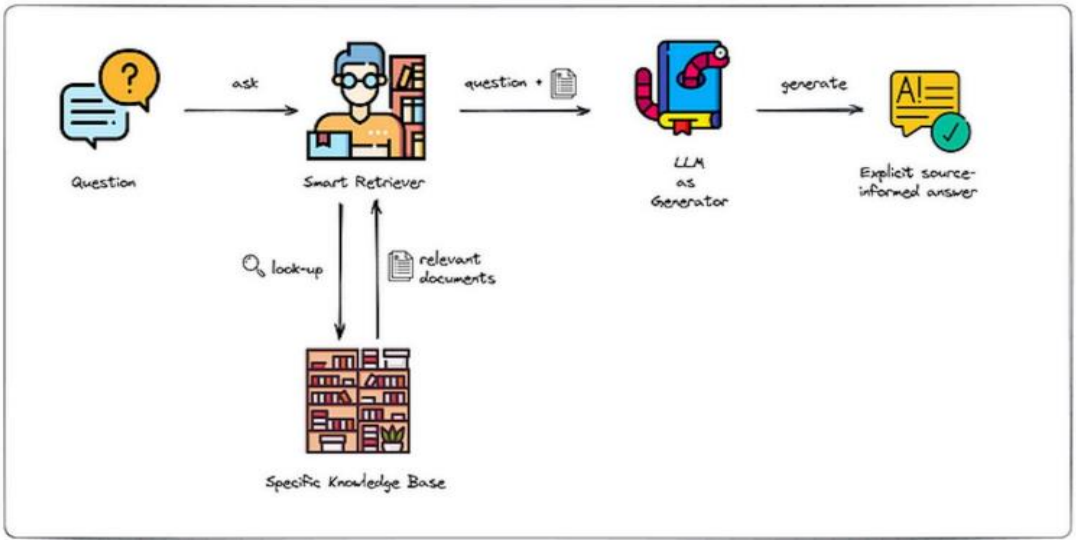

Figure 3: Overview of approach, Image source: deci.ai blog

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:07/July-2024 Impact Factor- 7.868 www.irjmets.com

2.2 Workflow

User Query: For instance, a user makes a query—a question "What is a neural network?" This question can be input into the RAG system.

Inside the RAG system:

Embeddings: Vector embeddings are applied to users question and convert it into the form of 0 & 1s. These are semantic representations; this can be information in the form of word embeddings, sentence embeddings, or paragraph embeddings.

Semantic Search: After the vector embedding of the user query semantic search will happen. In the semantic search, the system locates the most relevant chunks in the vector database based on the user question by semantic similarity. This is done through techniques of comparison, among which are cosine similarity or semantic hashing.

Retrieve: The most relevant chunks of data are recovered from the vector database based on the similarity score. These chunks carrying certain information on the user query Generative Answer with LLM: In this part, the user query and the relevant chunks passes to the LLM. The LLM uses its vast knowledge database to generate a response. LLM also ensures that the answer should be based on the information specify in the given document.

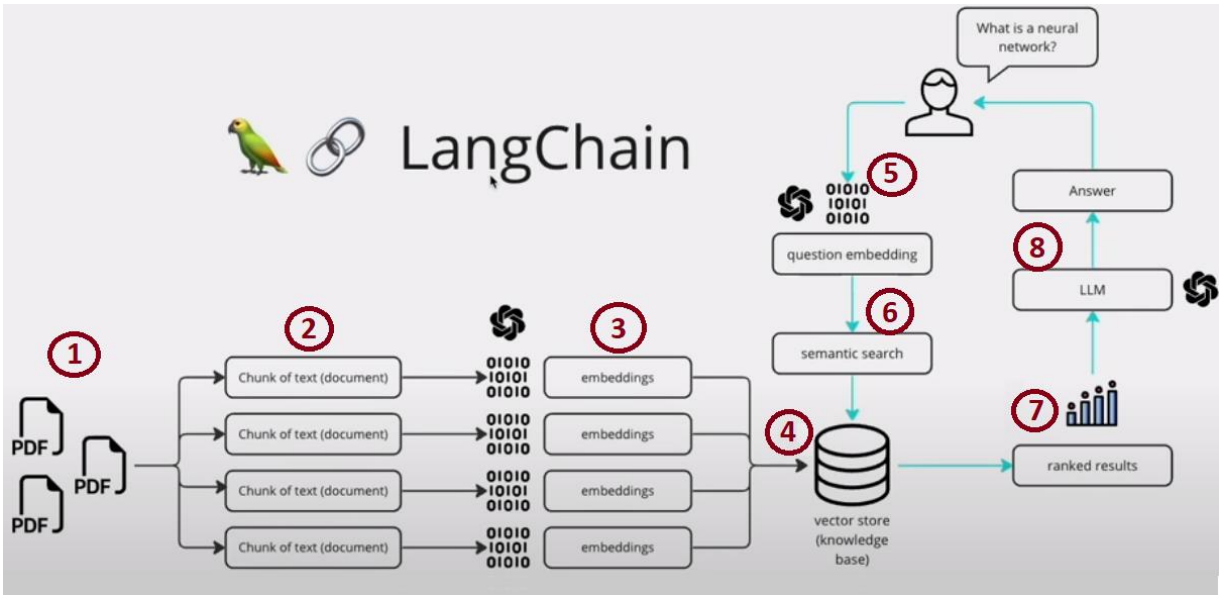

Figure 4: Simple working diagram, Image source: InternSystems RAG

2.3 How to build a RAG System?

For code please visit: https://github.com/Sumit-Pluto/Private_GPT

Prerequisite:

Python

Open AI API key (if not want to install LLM model weight or embedding model to your system)

LLM model weights (if not want to use LLM model through API key)

Hugging Face Instructor-large (for doing vector embedding locally without using API)

ChromaDB or any other vector database

Install the required packages: There is lots of libraries or packages required for building RAG system which perform all the operation from text embedding to retrieval the chunks from the vector database.

One of the most widely used framework for building RAG system is LangChain, which is designed to simplify the creation of applications using LLMs.

Load and Process Documents: The LLMs are capable of handling the various documents format like PDF, DOCX and TXT using LangChain loaders for e.g. PyPDFLoader, Docx2txtLoader and TextLoader

Chunking data -- when the document is successfully load, it splits into the chunks using LangChain text splitter. The size of the chunks is defined inside the code by the programmer.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:07/July-2024 Impact Factor- 7.868 www.irjmets.com

Creating Embeddings: Embeddings are basically vector representation of the text for efficient data understanding.

There are two ways to create embedding:

1. By using OpenAI API key for embedding
2. By installing the embedding model from the Hugging face in local device

After the embedding process is over, all the chunks go to the vector database, for this ChromaDB Or any other vector database can be used.

Build the chat interface: To deploy the RAG system, we mainly use flask or streamlit because they are very easy to programme and easily integrate with the AI model. With flask or streamlit web layout, user can upload their documents and ask their query.

Retrieve answers: The RAG system uses LangChain's RetrievalQA and vector database (ChromaDB) to efficiently response to the users query with relevant and accurate information from the ChromaDB embedded data.

Figure 5.1: Frame works & LLM model , Image source: Medium blog

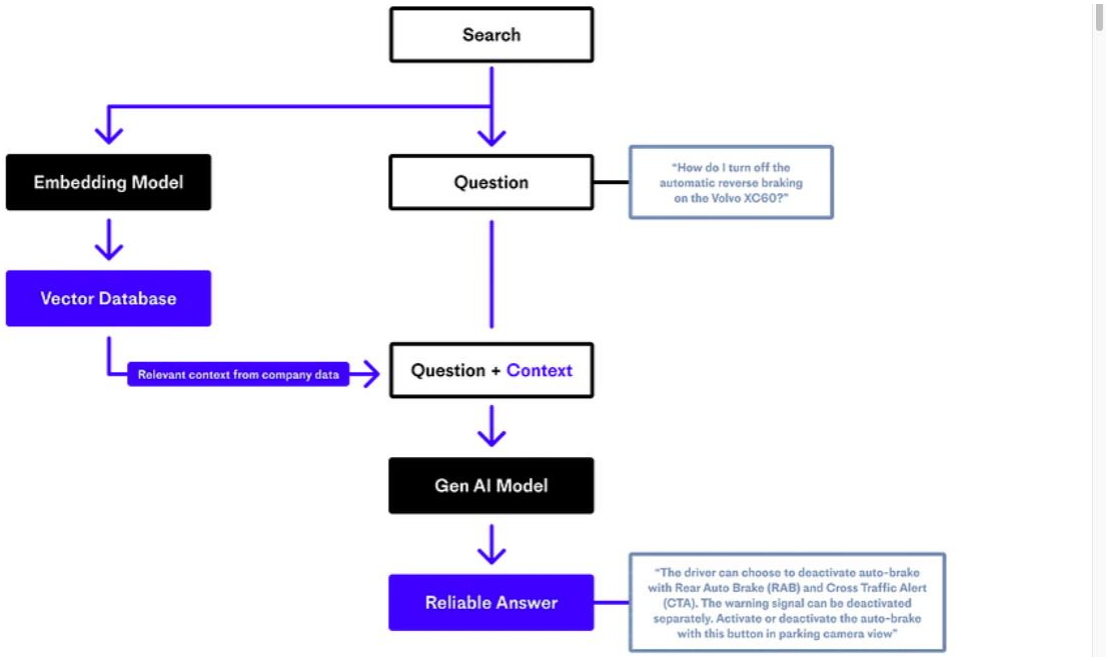

Figure 5.2: Flow Diagram, Image source: Medium blog

III. APPLICATION OF RAG

In this world, the data keeps increasing, and searching or extracting the important information from this data or from the large document is a very time-taking and complex task. For this, we can depend on the RAG application.

Summarization:

- Summarize long documents or articles.
- Produce short summaries combining retrieved chunks with generative language models.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:07/July-2024 Impact Factor- 7.868 www.irjmets.com

- User can quickly grasp the main ideas without reading the entire document.

Question Answering (Q/A):

- Provision of accurate answers for a user's queries.
- Retrieve the relevant information from a knowledge base and generate responses that are context sensitive.
- Handles a wide range of questions across various domains.

Content Generation:

- Expand short topic descriptions into full-fledged content.
- Can generate additional content based on the user's query.
- The allowance for detailed articles, blog posts, or reports.

Image and Graph Reading:(using Multimodal LLM)

- Read the images which is present inside the document and provide answer.
- Read the Graph and provide the important insights about the graph.

3.1 Advantages and limitation

The discussion section also critically evaluates the advantages and limitations of the proposed RAG system, shedding light on its potential implications for document analysis, privacy preservation, and user experience.

Precision with Context: The ability of RAG to combine retrieval and generative models of language enables it to provide answers sensitive to context; therefore, it allows for a balance between factual accuracy and fluency. Precision of this nature enhances the relevance and usefulness of the responses provided for an enhanced document interaction experience.

Customization: The flexibility of RAG to be fine-tuned towards specific domains or applications allows for tailoring the retrieval process to needs. This capability for customization enables users to tweak the system for an increased accuracy and relevance of the generated responses for specific use cases.

Data Integration: The integration of domain-relevant information from external data sources enables the RAG to mix domain-specific knowledge into the answers it generates. This integration enhances the depth and comprehensiveness of the answers it provides, allowing it to address complex questions and provide users with valuable insights.

Nuanced Answers: RAG excels at generating nuanced answers that are grounded in facts yet maintain the fluency of natural language. That capability facilitates the generation of responses that are insightful, going beyond simple statements of facts, and which enhance the overall experience for the user while helping in informed decision-making.

Complexity: Technically, RAG requires both retrieval techniques and generative language models expertise for it to be deployed and maintained, hence making it a non-trivial solution. The intricacy of the system poses challenges for less experienced users or organizations that require dedicated resources and support for successful implementation.

Quality of Data: The performance of RAG essentially depends on the quality of the external data that it retrieves. Bad quality or obsolete data can bring about a distortion in the reliability and accuracy of the responses by RAG system. It is very important to ensure that the inputs to this system are of high quality if optimal performance is needed.

Hallucinations: RAG, like other generative models of language, has the tendency to generate very plausible incorrect information, in which case it means inaccuracies or "hallucinations" in its responses are possible. Such phenomena underscore the need to have a human to verify and validate in order to ensure the accuracy and reliability of the outputs of the system, more so in critical or sensitive applications.

Resource-Intensive: The multi-step process in RAG, including retrieval, embedding, and generation, makes the whole process quite computationally expensive. Resource intensity may decrease system scalability, especially for large-scale deployments or applications that require high throughput. Efficient resource management and optimization strategies are necessary to reduce this limitation and guarantee efficient performance.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:07/July-2024 Impact Factor- 7.868 www.irjmets.com

IV. DISCUSSION

There are few resources currently available that document research related specifically to conversational AI that can extract information from the provided documents using RAG. Of those currently available, most of the research fails to provide integration of RAG with conversational AI and that too with knowledge base of one's source documents.

The concept RAG was first introduced by Lewis et al. in their paper titled "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks" published in 2020. Lead author Patrick Lewis of the 2020 article that first introduced the term apologized for the ugly acronym, which today refers to a developing family of techniques spanning hundreds of publications and dozens of for-profit services that he feels will shape generative AI in the future.

Lewis and colleagues created retrieval-augmented generation to connect generative AI systems to external resources, particularly those rich in current technical details. The study refers to RAG as "a general-purpose fine-tuning recipe" because it can be used by almost any LLM to interact with almost any external resource. Co-authors from University College London, New York University, and the former Facebook AI Research (now Meta AI) also contributed to the paper.

Conversations between users and their own data repositories are possible with retrieval-augmented generation.

For instance, a medical index added to a generative AI model might be a very useful tool for a physician or nurse. Having an assistant with access to market data would be beneficial for financial experts. Companies including AWS, IBM, Glean, Google, Microsoft, NVIDIA, Oracle, and Pinecone are implementing RAG because of its extensive potential.

Despite the potential, RAG-based chat systems face several challenges that need to be addressed. One challenge is the effective integration of retrieval and generation components to ensure seamless interaction and coherence in the conversation. Another major challenge is balancing the trade-offs between relevance and diversity in the generated answer, because both relevance and diversity are inversely proportional to each other. The generated answer is highly dependent upon the Embedding and LLM model which is used in RAG system.

Future research in RAG-based chat systems should focus on overcoming the challenges while exploring new avenues for improvement. This includes investigating the new embedding techniques or models and the most important part, which LLM should be used in RAG system.

V. CONCLUSION

The paper has discussed this innovative concept of leveraging Retrieval Augmented Generation (RAG) technology to develop a private GPT system so that one may interact with documents in a safe and efficient manner. We have discussed advantages and limitations of this approach, as well as possible implications, in this paper for document analysis, preservation of privacy, and user experience.

The private GPT ensures the users to enchance the interaction with their document without compromising the privacy.

The paper gives you the brief overview of how RAG based application work and how you can built it. Furthermore, the system design in the privacy and security considered will be part of the current concerns on how data privacy is to be considered in the digital era. Since it runs locally on the personal computer of the user or through a secure API, it keeps sensitive information private to the user, hence eliminating the risks involved with data exposure and unauthorized access. It will, therefore, ensure that users can leverage AI without compromising confidentiality and will develop trust and confidence in the system.

In essence, the private GPT system has great potential to evolve the document interaction workflows across different domains, such as research, business, and education, into an altogether new and completely different dimension. This will ensure that users are provided with a safe and effective means of interaction with their documents and will increase the productivity of the document workflow, facilitate knowledge discovery, and support informed decision-making. Moreover, the implementation of RAG technology gives a view into the future of document analysis by AI-driven types: it should be possible to incorporate the features of context

awareness, customization, and preservation of privacy into AI systems. In a nutshell, discovery of a private GPT system powered by RAG technology will go down as a turning point in the history of technologies dealing with document interaction. The system is full of promise for the safe and effective analysis of user documents. As AI is going to continuously evolve in the future, the possibilities for private GPT systems are endless and will further open new vistas for innovation, collaboration, and discovery in the digital age.

VI. REFERENCES

[1] Retrieval-Augmented Generation for Large Language Models: A Survey Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Qianyu Guo, Meng Wang, Haofen Wang

[2] Retrieval Augmented Generation: Streamlining the creation of intelligent natural language processing models Written By Sebastian Riedel ,Douwe Kiela ,Patrick Lewis ,Aleksandra Piktus

[3] Large Language Models: A Comprehensive Survey of its Applications, Challenges, Limitations, and Future Prospects.

[4] Muhammad Usman Hadi1, Qasem Al-Tashi,, Rizwan Qureshi2, Abbas Shah, Amgad Muneer2, Muhammad Irfan, Anas Zafar, Muhammad Bilal Shaikh, Naveed Akhtar, Mohammed Ali Al-Garadi8 , Jia Wu, Seyedali Mirjalili.

[5] A Survey of Large Language Models Wayne Xin Zhao, Kun Zhou*, Junyi Li*, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie and Ji-Rong Wen

[6] https://blogs.nvidia.com/blog/what-is-retrieval-augmented-generation/

[7] https://aws.amazon.com/what-is/langchain/

[8] Image Source: https://deepgram.com/ai-glossary/retrieval-augmented-generation