Gen AI driven FAQ chatbot using Advanced RAG Architecture for Querying Annual reports

Mehul K
Department of Information Technology
Sri Sairam Engineering College
Chennai, India
mehulkrishieee@gmail.com

Dr. V. R. Kanagavalli
Department of Humanities and Science
Sri Sairam Engineering College
Chennai, India
kanagavalli.math@sairam.edu.in

Ms. Saradha K R
Department of Information Technology
Sri Sairam Engineering College
Chennai, India
saradha.it@sairam.edu.in

Gowtham P N
Department of Information Technology
Sri Sairam Engineering College
Chennai, India
gowthamnaren2017@gmail.com

Sachin M P
Department of Computer Science and Engineering
(Artificial Intelligence and Machine Learning)
Sri Sairam Engineering College
Chennai, India
m.p.sachin265@gmail.com

Surya U
Department of Information Technology
Sri Sairam Engineering College
Chennai, India
suryajup17@gmail.com

Godhandaraman R
Department of Information Technology
Sri Sairam Engineering College
Chennai, India
godhandaraman313@gmail.com

Girish S
Department of Mechanical Engineering
Sri Sairam Engineering College
Chennai, India
girishofficial04@gmail.com

Naveen R
Department of Information Technology
Sri Sairam Engineering College
Chennai, India
naveen685685@gmail.com

***Abstract*—In the rapidly evolving business landscape, stakeholders require timely and accurate access to financial and operational information from annual reports. However, the extensive and complex nature of these reports makes it difficult to efficiently extract key analytical trends, financial performance measures, and comparative insights. The traditional manual approach to analyse these documents is time-consuming, error prone, and inefficient. Stakeholders—including creditors, customers, suppliers, employees, government agencies, and industry analysts—rely on this data to make informed decisions. The absence of an automated mechanism to handle frequent queries regarding financial performance, operational metrics, and sustainability targets impedes large-scale informed decision making. This paper presents a Gen AI-driven solution that leverages Natural Language Processing (NLP), LangChain Framework and Retrieval Augmented Generation (RAG) Architecture to efficiently extract and analyze financial annual reports and its data. The approach involves utilizing PyPDF2 for data extraction, segmenting content using RecursiveCharacterTextSplitter, generating embeddings via FastEmbedEmbeddings module, and storing information in Qdrant for rapid similarity search. The response generation is powered by DeepSeek LLM, deployed locally via Ollama, ensuring privacy and computational efficiency. A Gradio-based UI is implemented for an intuitive interface. This system enhances financial data accessibility, fostering transparency, compliance, and data-driven decision-making.**

***Keywords*—Financial Report Analysis, AI Query System,RAG Architecture, NLP, LangChain Framework, Qdrant, Ollama, Gradio**.

I. INTRODUCTION

Annual financial reports contain critical information on a company's financial health, operational efficiency, and strategic plans. However, stakeholders often struggle to extract precise insights due to the document's complexity. Traditional search mechanisms and manual data retrieval methods are inefficient, requiring significant time and effort to locate relevant information.

In response to this challenge, an AI-driven financial query system is proposed, that automates the extraction and analysis of annual reports. The system utilizes PyPDF2 for structured data extraction, an advanced chunking mechanism for text segmentation, and Qdrant for vector-based retrieval. The system also incorporates DeepSeek LLM, hosted locally via Ollama, to process user queries. A Gradio-based UI ensures seamless user interaction.

II. LITERATURE SURVEY

The authors in [1] tackle the shortcomings of current multimodal Retrieval-Augmented Generation (RAG) systems, which have difficulty effectively grasping and using the connections between text and images. Standard RAG models are great at retrieving and adding to text data, but they struggle to produce consistent replies when dealing with multimodal documents that have intricate text-image connections. To get around these issues, the authors suggest a better multimodal RAG structure that boosts retrieval and generation by explicitly factoring in image-text relationships. They assess how well this method works by comparing it against four question-answering datasets—Short-form QA, Long-form QA, MCQ-type QA, and True-False QA—showing significant advancements over current methods, with additional insights gained from testing with large multimodal models like OpenAI's and Gemini's.

The authors in [2], tackle the challenge of extracting information from unstructured documents, such as URLs and PDFs. These types of documents are difficult to search efficiently because of their complex and unorganized nature. Traditional systems for text analysis often struggle with interactivity, flexibility, and integrating data, which reduces their effectiveness for these kinds of tasks. To address this, the authors suggest a new approach that uses LangChain's advanced natural language processing, combined with Dash, Google Generative AI, and FAISS. This method aims to enhance the efficiency of extraction and analysis. The effectiveness of this system is demonstrated through experiments, showing considerable gains in both accuracy and speed. This is further enhanced by features like annotation, query storage, bookmarking, text-to-speech, and translation into ten different languages.

The authors of [3] tackle the growing problem of how awkward and slow it can be to work with PDF files. It's a hassle lots of people are running into, especially since digital documents are becoming the norm, but old-school PDF viewers just aren't cutting it when it comes to actually using them. Basic PDF readers can't really pull out important info, give a quick summary, or let easily search or ask questions about the text, leaving users with fewer options than they should have. To fix this, the authors are looking into something called LangChain, which is a language model built to make grabbing information, summarizing documents, searching through them, getting answers to questions, and even translating them way smoother. LangChain does this by combining really good natural language understanding with a deep dive into the documents themselves. The authors check out how well LangChain works by examining its inner workings and testing it out in real-life situations, showing that it can really boost how much people can get done, how easy it is to access information, and how well teams can work together.

The research described in [4] addresses the challenge of dealing with information overload in large documents, a problem where traditional methods fail to adequately capture the essential points for users. Existing tools generally can't handle summarization and question-answering tasks efficiently when it comes to large-scale documents, often leading to reduced productivity. To combat this issue, the authors propose a new framework for creating personalized chatbots that utilize Large Language Models (LLMs). They integrate OpenAI, LangChain, and Streamlit to enhance the document processing capabilities of these chatbots. The effectiveness of this framework is assessed by examining its design and real-world use cases. A detailed walkthrough showcases how the framework greatly improves the speed and accuracy of information retrieval, ultimately enhancing user productivity.

The authors of [5] delve into the shortcomings of early AI chatbots, noting their lack of versatility and responsiveness when compared to human cognitive abilities. This limitation confined their practical applications. Traditional models were designed for specific, narrow tasks and struggled with diverse content creation and the effective processing of multimodal data. To address this, the authors introduce modern generative AI chatbots, equipped with Large Language Models (LLMs) for textual tasks and Large Multimodal Models (LMMs) for handling various forms of data. These advanced chatbots can generate a wide range of content, including text, images, audio, video, code, virtual worlds, datasets, and web content. The study meticulously categorizes their capabilities into eight distinct groups, showcasing their adaptability and efficiency in real-world scenarios.

The authors of [6] introduce an AI system for document management with LangChain and Pinecone. The article describes how documents are divided into small pieces, transformed into numbers, and saved for rapid searching. When a user queries, the system retrieves the most relevant data and provides an answer through AI. This assists in rapid finding, summarizing, and comprehending documents. The research points to its application in fields such as law, education, and business to make information handling manageable on a large scale.

The authors of [7] introduce an AI system for document management with LangChain and Pinecone. The article describes how documents are divided into small pieces, transformed into numbers, and saved for rapid searching. When a user queries, the system retrieves the most relevant data and provides an answer through AI. This assists in rapid finding, summarizing, and comprehending documents. The research points to its application in fields such as law, education, and business to make information handling manageable on a large scale.

III. METHODOLOGY

The advanced tools and techniques such as Natural Language Processing (NLP), LangChain Framework and Retrieval Augmented Generation (RAG) Architecture is being employed for building Gen-AI Driven FAQ Chatbot, user-friendly-interface to improve access and decision-making tasks for stakeholder. It has five phases: data extraction using PyPDF2; text-processing chunking; embedding-generation and reranking of chunks; retrieval and query processing with the intervention of DeepSeek LLM; finally, and it is deployed using Gradio UI.

A. Data Extraction Using PyPDF2

The methodology's first phase emphasizes standard approaches to extract textual content from these financial reports. These finance assessments are readily available in PDF format. In doing so, PyPDF2-a standard Python library is used to parse and read PDF files. The system will use PyPDF2 to systematically extract raw text from these documents while maintaining the sequential order of content while being able to efficiently organise the multi-page reports. Annual and Quarterly Financial reports usually contain narrative text, numerical data, and publishing structure elements like tables that dictate a robust extraction rationale. The extraction procedure commences with loading the PDF into memory. Subsequently, the pages are looped through to fetch the text material. A lightweight instruction set drives the process, telling PyPDF2 to pull precise and pertinent content like nonaudited financial statements, management reports, SEC-required disclosure statements, and tabular data while ignoring irrelevant or redundant content such as headers, footers, or repetitively redundant boilerplate text. To validate the extraction procedure, a preliminary check is done on the output to ensure that certain critical sections, such as financial summaries or legal disclosures, are accurately captured. By applying precision principles in PyPDF2 extraction and enhancing the quality of subsequent data processing, the aim of providing reliable insights to system stakeholders-such as creditors, analysts, and regulators-is then facilitated.

B. Text Processing and Chunking

The next phase is one in which the raw text has been extracted; it now requires some form of processing and segmentation to suit the analysis. Most often, these financial reports include very long, unstructured narratives that are interspersed with structured elements; hence, a very systematic method for text handling is called for. This phase includes two important steps: loading the extracted text into a structured format; and segmenting it into small manageable pieces. Upon extraction, the document is first rendered into Markdown format in the process of parsing. Subsequently, the text is loaded into a document object using the UnstructuredMarkdownLoader module. This loader is crucial in preserving the integrity of formatting elements typical of financial reports, such as detail tables presenting revenue or expenses, succinct bullet points summarizing key metrics, and other structure types that encode truly complex financial information. In this way, it helps in keeping the structure intact and ensures that the meaning and context of the data are preserved. It thereby avoids tasks such as errant table entries, or broken lists that may actually happen for the raw text.

Once the loading process is completed, the structured document would undergo segmentation using the

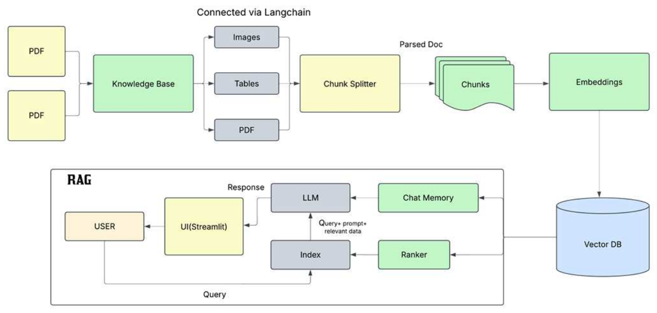


Fig. 3.1. System Architecture

RecursiveCharacterTextSplitter module. This splits the text into a number of smaller coherent pieces, establishing boundaries based on both the character limits and logical breaks such as paragraph breaks or section heads. This process aims to create chunks that are not only stand-alone in that they contain enough context and information, but they are also reasonably few in size—usually not exceeding a pre-established character length during processing such as 1000 characters, which will make indexing and searching for relevant information much easier later. The recursive character of the splitter makes sure that, in case of long sections, the splitting is done iteratively to maintain readibility and avoid unnaturally breaking words apart into obscure fragments.

C. Embedding Generation and Reranking of Chunks

The system converts segmented texts into numerical embeddings to facilitate effective financial data retrieval. The (POS) part-of-speech tagging of the text is performed utilizing the library of NLTK's Averaged Perceptron Tagger. This helps structure the text and process it in a more compliant manner. The level of semantic understanding is improved, and the quality of the resulting embeddings is thus enhanced aboard. The text chunks are then processed by fast embeddings, generating high-dimensional vector representations that capture contextual relationships within financial reports. The embeddings allow financial terminology to be contextualized, so that related terms like ”revenue” and ”profit” are located close in the vector space. This stage is critical for executing accurate similarity searches for improving the quality of retrieval.

The embeddings generated are stored in ChromaDB, an advanced vector database optimized for rapid similarity searches and structured indexing. ChromaDB organizes and relates every vector with relevant metadata-such as section title and page number-for efficient retrieval across large financial documents. Its indexing methods allow for low-latency searches, even among huge reports. For further precision, FlashrankRerank, the sophisticated reranking model which essentially rearranges results based on the relative importance of context, is used. FlashrankRerank therefore ranks the retrieved chunks from query intent such that the most relevant financial insights occupy the peak position. This helps eliminate noise and subsequently improve extracted data precision. Integration of POS tagging, embedding generation, efficient vector storage, and intelligent reranking determines the method of optimum retrieval of financial data, thus providing stakeholders with accurate, fast, and contextually aware insights. This ensures easy extraction of valuable information by financial professionals, investors, and analysts alike in support of informed decisionmaking that streamlines financial analysis workflows.

D. *Retrieval and Query Processing Using DeepSeek LLM*

The fourth phase integrates retrieval and response generation to effectively handle stakeholder queries. The process combines Qdrant's vector-based retrieval with the natural language understanding capability of DeepSeek LLM deployed in a local context via Ollama for better privacy and efficiency.

When a user sends a query (for example,"What are the key financial highlights?"), it passes through the FastEmbedEmbeddings module and gets turned into an embedding that is consistent with the vectors stored within the report. The embedding is then used to perform a similarity search in Qdrant to identify the top-k text chunks that are most relevant, either based on cosine similarity or Euclidean distance metrics. The retrieved chunks are basically the portions of the financial report that are reasonable candidates for doing what is being asked, such as revenue summaries or management commentaries. The extracted chunks go off to DeepSeek, a natural language-generation-oriented large language model, for further elaboration. DeepSeek, through Ollama, processes the chunks alongside the user query and generates a cohesive and accurately contextualized answer. A user-defined function formats the output so that it can actually be read with some ease. The function extracts the main content from the LLM's response dictionary and makes sure it reads well by using the textwrap library to wrap the text at 100 characters. This function takes care of empty lines, with proper spacing while not just breaking the words unnecessarily, thus keeping long summarizes of complex financial data clearly readable. This retrieval approach and processing of the user query ensure that the stakeholders get precise, comprehensible responses from the content of the report. DeepSeek being deployed locally through Ollama enhances computational efficiency and privacy of data, while the integration to retrieve quickly through Qdrant supports timely decision-making and meets the transparency and usability objective of the system.

E. *User Interface and Deployment*

The last phase will work on delivering the system in a user-friendly fashion via its UI and local deployment through Ollama. The UI is based on Gradio and is interactive and simple to work with so that the stakeholders can input financial queries and obtain structured responses. Gradio offers an extremely light framework for real-time interactions; users can upload financial reports, pose targeted inquiries, and obtain well-structured insights right in their web browsers-an essential feature that ensures even nontechnical users can effectively navigate and extract relevant financial information. The system is deployed locally through Ollama rather than being hosted on the cloud, with enhanced privacy, security, and computational efficiency. Since Ollama enables on-device execution of the DeepSeek LLM, there is no need for external servers, reducing reliance on third-party cloud infrastructure. This means that the system can run the model locally for very low-latency performance while guaranteeing full control over sensitive financial data. Furthermore, by deploying the system locally, operational costs will be reduced, providing a feasible scenario for organizations with strict data compliance requirements.

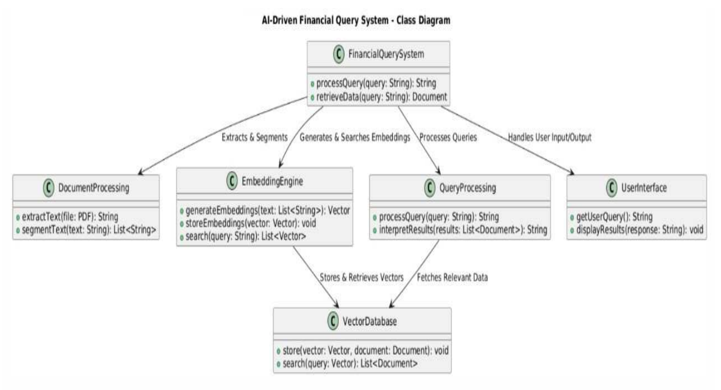
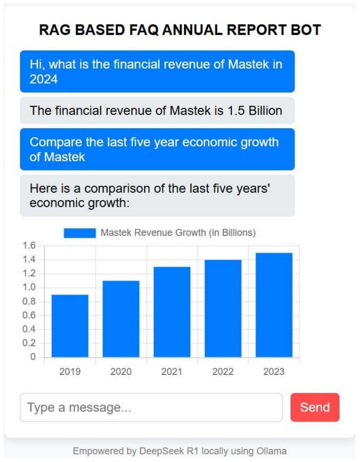

AI-Driven Financial Query System - Class Diagram
FinancialQuerySystem
processQuery(query: String): String
retrieveData(query: String): Document
Extracts & Segments
Generates & Searches Embeddings
Processes Queries
Handles User Input/Output
DocumentProcessing
extractText(file: PDF): String
segmentText(text: String): List<String>
EmbeddingEngine
generateEmbeddings(text: List<String>): Vector
storeEmbeddings(vector: Vector): void
search(query: String): List<Vector>
QueryProcessing
processQuery(query: String): String
interpretResults(results: List<Document>): String
UserInterface
getUserQuery(): String
displayResults(response: String): void
Stores & Retrieves Vectors
Fetches Relevant Data
VectorDatabase
store(vector: Vector, document: Document): void
search(query: Vector): List<Document>

Fig. 3.2. Class Diagram

This way of UI and deployment works as a bridge between advanced AI-driven analytics and the stakeholder needs. The infusion of user-friendly transactions through Gradio and secure, on-device Ollama deployment ensures rigorous, transparent, and data-driven decisions for financial analysts, industry professionals, and regulatory bodies alike.

IV. RESULTS AND DISCUSSION

Experiments demonstrate that the AI-driven query system significantly improves the efficiency and accuracy of financial data retrieval. The below figure 4.1 shows the User Interface of the chat bot.


RAG BASED FAQ ANNUAL REPORT BOT

Hi, what is the financial revenue of Mastek in 2024

The financial revenue of Mastek is 1.5 Billion

Compare the last five year economic growth of Mastek

Here is a comparison of the last five years' economic growth:

Mastek Revenue Growth (in Billions)
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0
2019
2020
2021
2022
2023
Type a message...
Send
Empowered by DeepSeek R1 locally using Ollama

Fig. 4.1. Output

Compared to traditional search mechanisms, the system delivers near-instant responses with higher relevance, reducing manual effort. The integration of DeepSeek LLM with Qdrant ensures context-aware and precise financial insights. Furthermore, hosting via Ollama and Gradio enhances user experience and accessibility.

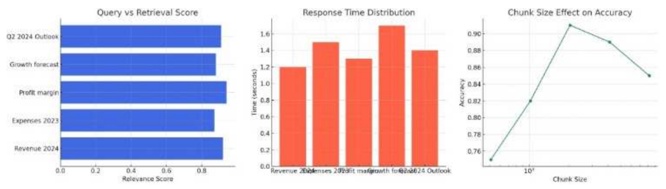


Fig 4.2. Performance of the System

Here, Figure 4.2 analyses the retrieval effectiveness and performance of the system. The first chart displays the relevance scores of retrieved responses for various questions, indicating accuracy with scores nearing 0.9. Similarly, the second graph displays the distribution of response time ranging from 1.2 seconds to 1.7 seconds , while maintaining an overall high level of accuracy (>0.85). The third graph evaluates how chunk size affects accuracy and suggests a maximum achievable value for accuracy at about 0.90, then falls into minimal decline.

V. CONCLUSION

The financial report analysis platform powered by Gen-AI has been successfully implemented, with financial and operational information made more accessible and usable. Through NLP and Retreival Augmented Generation (RAG) the platform streamlines the extraction, segmentation, and retrieval of key financial information, enabling finance teams to spend more time on analysis rather than data work. PyPDF2 was employed for parsing documents, the RecursiveCharacterTextSplitter module for text segmentation into sections, and FastEmbedEmbeddings with Qdrant for storage and retrieval efficiency. DeepSeek LLM, deployed on Ollama, delivers precise, context-specific answers to enhance financial insights. The user interface, developed using Gradio, provides an interactive means for investors, financial analysts, auditors, and regulators to view financial information.

The system enhances financial information access, facilitates transparency and compliance, and enables companies to make informed data-driven decisions. It automates financial documents processing, extracts meaningful insights, and enables stakeholders to identify trends, risks, and opportunities faster. The platform is now live, aiding in financial analysis and strategic planning.

VI. REFERENCE

[1] Joshi, P., Gupta, A., Kumar, P., & Sisodia, M. (2024, June). Robust multi model rag pipeline for documents containing text, table & images. In 2024 3rd International Conference on Applied Artificial Intelligence and Computing (ICAAIC) (pp. 993-999). IEEE.

[2] Priya, K., Kamath, A., Chandan, K. M., Praveen, C., Omkar, S. N., & Aaditya, S. J. (2024, November). Enhancing Q&A Systems with Multilingual Text Conversion and Speech Integration: Harnessing the Power of LangChain and Large Language Models. In 2024 8th International Conference on Computational System and Information Technology for Sustainable Solutions (CSITSS) (pp. 1-6). IEEE.

[3] Kotiyal, A., Gujjar, P., Prasad, G., Devadas, R. M., Hiremani, V., & Tangade, P. (2024, July). Chat With PDF Using LangChain Model. In 2024 Second International Conference on Advances in Information Technology (ICAIT) (Vol. 1, pp. 1-4). IEEE.

[4] Pokhrel, S., Ganesan, S., Akther, T., & Karunarathne, L. (2024). Building Customized Chatbots for Document Summarization and Question Answering using Large Language Models using a Framework with OpenAI, Lang chain, and Streamlit. Journal of Information Technology and Digital World, 6(1), 70-86.

[5] Naik, D., Naik, I., & Naik, N. (2024). The AI Engine of Creation: Exploring the Capabilities of AI Chatbots based on

Generative AI, Large Language Models and Large Multimodal Models. Authorea Preprints.

[6] Pesaru, A., Gill, T. S., & Tangella, A. R. (2023). AI assistant for document management Using Lang Chain and Pinecone. International Research Journal of Modernization in Engineering Technology and Science, 5(6), 3980-3983.

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