Journal of Frontiers in Multidisciplinary Research

RAG-Based Query Engine using LLM and Vector DB for College Details

Arshia Tahseen [1*], Joanna Sumathi [2], Shriya Reddy [3], Dr. D. Shravani [4]
[1-4] Department of Computer Engineering Stanley College of Engineering and Technology for Women, Hyderabad, India

* Corresponding Author: **Arshia Tahseen**

Article Info

E-ISSN: 3050-9726
P-ISSN: 3050-9718
Volume: 06
Issue: 01
January-June 2025
Received: 01-01-2025
Accepted: 03-02-2025
Published: 05-03-2025
Page No: 86-91

Abstract
Large Language Models (LLMs) and other developments in generative AI technology have broadened the application of AI in a number of fields in recent years. This paper suggests a multipronged strategy to address this, utilising LLM's strengths to minimise data insufficiency through specific solutions, including document integration and fine-tuning. The creation and implementation of a Retrieval-Augmented Generation (RAG) model, a chat application intended to enhance information retrieval and storage procedures and support the production of high-quality material, is the main goal. The work offers a thorough examination of the improved information retrieval phases by integrating a RAG model, highlighting their functions in getting over data constraints. This research presents the development of an efficient chatbot utilizing large language models (LLMs) such as LLaMA and Gemini, integrated with the Retrieval-Augmented Generation (RAG) framework to enhance response accuracy and contextual understanding and also uses Groq AI to reduce the response time. The chatbot is supported by a Facebook AI Similarity Search (FAISS) for efficient similarity search and clustering of dense vectors, enabling fast and scalable searches. Using Beautiful Soup, data was gathered via web scraping college website to extract important information such as course offers, faculty biographies, admissions details.In order to create an intelligent, domain-specific chatbot, this paper demonstrates how well LLMs, vectorized search, and web scraping can be integrated with the RAG framework. The results show how such a system could improve user involvement in academic settings, automate institutional communication, and reduce human labor.

DOI: https://doi.org/10.54660/.IJFMR.2025.6.1.86-91

Keywords: Chat Application, LLMs (Large Language Models), Gemini, LLaMA, RAG (Retrieval-Augmented Generation), FAISS (Facebook AI Similarity Search), Vector Database, Semantic Search, Beautiful Soup, Semantic Search, Web Scraping, College Information System

1. Introduction

Recent developments in Generative Artificial Intelligence (AI) have been fueled by the creation of models such as OpenAI's ChatGPT, which has established this technology as a disruptive force in industry, academia, and creativity. Large Language Models (LLMs) have shown tremendous promise in a variety of applications, including context-aware problem-solving and natural language creation. The development of LLMs like LLaMA, Gemini, and Groq AI, which are excellent at comprehending, processing, and producing responses that resemble those of a human, is a noteworthy development in this area. Nevertheless, these models are inherently limited, especially with regard to their dependence on static knowledge, real-time flexibility, and input size restrictions. Retrieval-Augmented Generation (RAG) frameworks have become a viable answer to these problems, allowing LLMs to dynamically access external knowledge sources. Instead of depending just on pre-trained data, intelligent and adaptable chatbots that can retrieve information in real-time can be created by combining LLMs with effective vector-based search engines. By effectively indexing and transforming text data into numerical embeddings, FAISS is essential to enabling

fast similarity searches. In order to retrieve the most pertinent information, a user's query is vectorized and then compared to the stored embeddings using FAISS's nearest-neighbor search. After this data has been received, it is run via the RAG framework, where an LLM (LLaMA, Gemini, or Groq AI produces a natural-language answer that is contextually appropriate. Instead of depending just on previously learned information, this integration guarantees that the chatbot may dynamically retrieve, process, and provide responses based on the most recent information available. This method allows for smooth answers to a variety of college-specific questions, ranging from admissions information to academic program requirements, and is designed to satisfy the distinct informational demands of students, teachers, and administrative personnel. This chat application offers a scalable, effective, and intelligent way to automate information access at educational institutions by fusing LLMs, FAISS-based vector retrieval, and the RAG framework. This study examines the system's technological architecture, implementation procedure, and performance advantages, highlighting how it might improve institutional communication and the user experience for academic staff, administrative staff, and students. In the upcoming sections we'll have a clear understanding on the following

- Data Collection
- Vectorization in Database using FAISS
- Query Processing and Retrieval
- Response Generation using RAG and LLMs
- Delivering Contextually Accurate Responses

2. Related Work

This section reviews literature on existing works in the field of query-answering chatbots utilizing LLMs for document and image-based information retrieval. Processing document images efficiently is a challenge in automated information retrieval. Mathew *et al.* [1] introduced DocVQA, which explores Visual Question Answering (VQA) for images in documents, emphasizing the importance of extracting meaningful text from scanned documents. Advanced deep learning-based models enhance accuracy in recognizing embedded text, a crucial component of our chatbot system. Cui *et al.* [2] developed SuperAgent, an e-commerce chatbot capable of handling real-time customer queries using NLP techniques. While their work focuses on commerce, it validates the potential of AI-powered chatbots for structured question answering, similar to our system, which retrieves answers from PDFs. Kim & Lee [3] designed an educational chatbot specifically for answering university-related queries. Their study demonstrates that chatbots can effectively respond to structured domain-specific queries, reinforcing the feasibility of our chatbot for academic document-based question answering. Brown *et al.* [4] demonstrated that GPT-3 excels in few-shot learning scenarios, making it effective for chatbot-based applications. However, they also highlight the dependence of LLMs on well-structured prompts, an issue addressed in our chatbot's query-processing mechanism. Lewis *et al.* [5] proposed Retrieval-Augmented Generation (RAG), which improves chatbot response quality by retrieving relevant document segments before generating answers. This approach is essential for handling knowledge-intensive tasks, ensuring that responses are both factually accurate and contextually relevant.

Yih *et al.* [6] introduced Fusion-in-Decoder, which improves chatbot responses by integrating retrieved knowledge into the model's decoder. This method is crucial for context-aware response generation, as it ensures that answers to PDF-related queries remain fact-based and coherent. Floridi & Chiriatti [7] analyzed GPT-based chatbots, pointing out their strengths and weaknesses. While these models excel at natural language understanding, they sometimes produce hallucinated responses when lacking sufficient retrieval mechanisms. Our work mitigates this issue using context based retrieval and vector embeddings. Johnson & Zhao [8] emphasized the role of prompt engineering in optimizing LLM performance. Their work suggests that structured and well-formulated prompts significantly improve chatbot accuracy, which is relevant to our system's query refinement module. Guzman & Van der Maaten [9] explored prompt optimization techniques, showing that fine-tuned prompts can enhance chatbot performance in knowledge-intensive tasks. Our chatbot incorporates these insights by structuring queries effectively. Johnson *et al.* [10] demonstrated the impact of GPU-accelerated similarity search, which aligns with our chatbot's use of embedding-based retrieval to match user queries with document content.

Xu *et al.* [11] focused on optimizing vector search in resource-limited settings, a relevant aspect of our work since efficient retrieval mechanisms are necessary for scaling chatbot applications without excessive computational overhead. Ananya G and Dr. Vanishree K [12] explored RAG-based chatbots that enhance accuracy by combining retrieval with generative models. Our chatbot adopts this method, using vector embeddings based retrieval and LLMs to generate precise responses for image-based queries. Dr. Snehal Rathi *et al.* [13] studied hybrid chatbots that integrate rule-based logic with LLMs to improve factual accuracy. Our chatbot applies this approach by using structured retrieval alongside AI-generated responses for more reliable answers. Arijit Khan [14] examined the role of knowledge graphs in improving search and question-answering accuracy. The chatbot applies similar retrieval techniques using embeddings to enhance response precision from document images. Soha Khazaeli *et al.* [15] developed a legal question-answering system that retrieves and ranks answers using embeddings and BERT-based models. Their approach improves accuracy in free-format legal queries by leveraging both general and legal domain data. From the literature, it is observed that existing chatbot systems struggle with accurately answering queries related to images in PDFs due to limited multimodal integration. This paper overcomes this challenge by combining embeddings and LLMs for improved response accuracy.

3. Proposed Framework

The Methodology involves collecting data from Website, preprocessing this data, segmenting text, generating embeddings for semantic understanding. Retrievers connect generative models to information sources by utilizing Retrieval Augmented Generation (RAG). A web application that connects to the database and the generative model is how users engage with the LLM and hence provides us with an accurate output according to the context of the query.

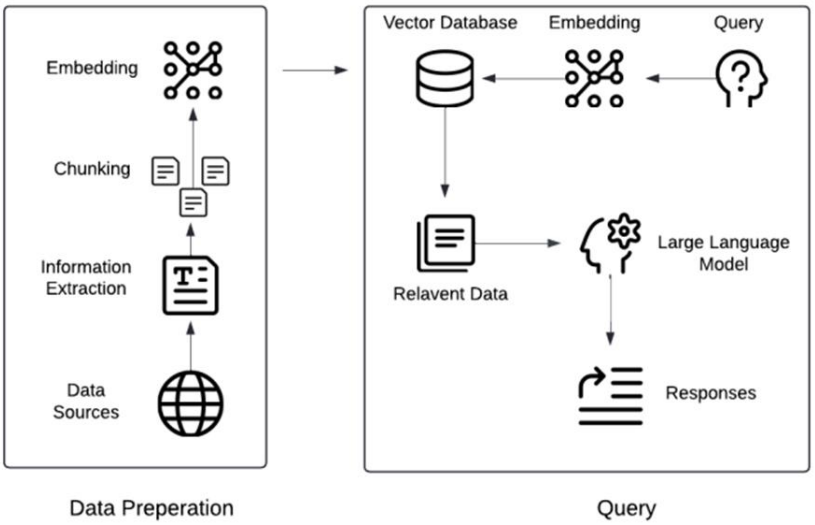

Fig 1: Flowchart of Working

A. Collecting data and preparing database

In this project, structured data from a college website is scraped using Beautiful Soup. It is first used to extract all links available on the homepage, it then extracts important information including course offers, faculty profiles, admissions information, and institutional policies from these links. After being gathered, the data is cleaned and organized so that the chatbot may quickly access it for later processing and retrieval. This data is stored in the form of PDF.

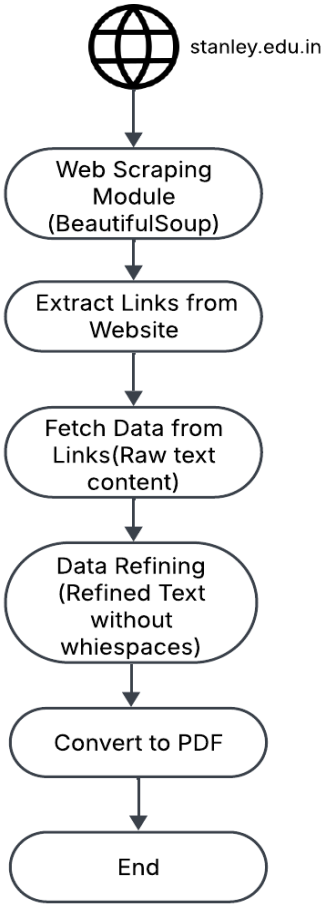

Fig 2: Flowchart for Data Collection

B. PDF data preprocessing

The input data consists of PDF files containing textual content and visual content which must be extracted for processing. Text extraction techniques convert PDF content into a machine-readable format. Preprocessing steps clean and standardize the extracted text by removing irrelevant elements (headers, footers, non-text components) and resolving formatting inconsistencies.

C. Text segmentation and embedding generation

The extracted text is divided into smaller segments to improve processing efficiency. These segments are then

converted into embeddings which are numerical representations that capture semantic meaning. This is done using the Embedding-001 model developed by Google. By encoding contextual relationships, embeddings enhance the system's ability to interpret and analyze text, supporting tasks like semantic search and context-aware content generation.

D. Vector store database creation

Vector databases store text embeddings, enabling efficient retrieval and processing. These databases support Retriever-Augmented Generation (RAG) systems by allowing the retrieval of relevant information during response generation. All of these Embeddings stored in FAISS RAG combine retrieval-based and generative approaches to produce more accurate and contextually rich outputs than traditional models.

E. Retrievers in the RAG framework

Retrievers act as intermediaries between generative models and external knowledge sources. They enhance content generation by performing semantic search and retrieving relevant information based on user queries. This integration improves response accuracy and depth by leveraging external data.

Fig 3: User interface for chatbot

F. User interaction with the LLM

Users engage with the LLM through a web-based interface. When a query is submitted, the LLM processes it into an embedding, performs a semantic search to retrieve relevant context, and generates a response. This iterative process ensures context-aware, relevant interactions, improving usability and user satisfaction.

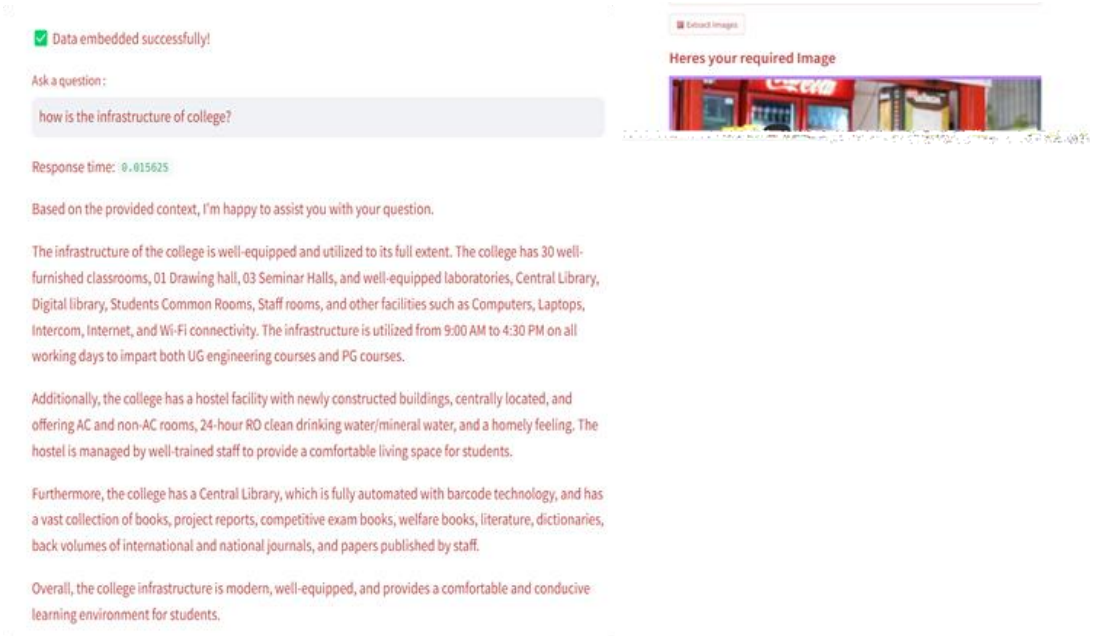

Fig 4: Response with Text and Images

4. Experimental Results

4.1 Vector databases: Enhancing data retrieval & search

Vector databases represent a paradigm shift in data storage, specializing in high-dimensional numerical representations rather than conventional row-column structures. These systems excel in similarity-driven applications like NLP, computer vision, and user behavior analysis by employing advanced indexing techniques for vector comparisons. Their core functionality lies in the ability to perform rapid proximity searches across massive embedding datasets, overcoming the limitations of traditional exact-match query systems.

Core Advantages

- **Optimized Embedding Storage:** Efficiently handles complex, high-dimensional data structures
- **Semantic Search Capabilities:** Delivers lightning-fast nearest-neighbor query performance
- **Elastic Scalability:** Adapts seamlessly to exponentially growing AI datasets
- **Native AI Integration:** Provides built-in support for machine learning workflows and model serving

Table 1: Comparative study of vector databases

Feature	Chroma DB	FAISS
Query Speed	15-60ms (1M vectors, CPU)	0.5-5ms (1M vectors, GPU)
Indexing Speed	10-30 min (1M vectors)	2-5 min (1M vectors, GPU)
Max Scale Tested	100M vectors (with performance drop)	1B+ vectors (production deployments)
Scalability	Limited by CPU	Highly Scalable

4.2 Large Language Models (LLMs) in Retrieval-Augmented Generation (RAG)

Large Language Models (LLMs) have revolutionized natural language processing (NLP) applications, ranging from conversational AI to information retrieval and code generation. Among the most prominent LLMs, Meta's LLaMA, OpenAI's GPT, and Google's Gemini have emerged as leading contenders, each offering distinct advantages in terms of architecture, scalability, and application. This paper presents a comparative analysis of these models, evaluating their design, capabilities, and performance.

4.3 Overview of language models

4.3.1 LLaMA (Large Language Model Meta AI)

LLaMA, developed by Meta, is an open-weight model series aimed at fostering AI research and democratizing access to powerful language models. The latest versions, LLaMA 2 and anticipated LLaMA 3, provide multiple model sizes (7B, 13B, 65B parameters), optimized for efficiency and fine-tuning. Unlike proprietary models such as GPT and Gemini, LLaMA allows direct modification and adaptation by researchers and developers.

4.3.2 GPT (Generative Pre-trained Transformer)

Developed by OpenAI, the GPT series has set benchmarks in natural language understanding and generation. GPT-4, including its optimized variant GPT-4-turbo, employs deep transformer networks trained on extensive datasets. GPT is widely used in conversational AI (e.g., ChatGPT), content generation, and programming assistance. While proprietary, GPT models are available through APIs, making them accessible for commercial applications.

4.3.3 Google Gemini

Google's Gemini series, developed by DeepMind, represents the next step in multimodal AI, integrating text, image, video, and audio processing. Gemini 1.5, the latest iteration, supports long-context understanding (over 1 million tokens), enabling advanced reasoning and retrieval-based tasks. Integrated with Google's ecosystem (e.g., Search, Workspace), Gemini enhances multimodal interactions beyond traditional text-based LLMs.

4.4 Comparative Analysis

Table 2: Comparison between LLM models

Feature	LLaMA (Meta)	GPT (OpenAI)	Gemini (Google)
Model Size	7B-65B params	Various (GPT-3.5, GPT-4, Turbo)	Scalable (1M+ token context)
Architecture	Transformer-based, optimized for research	Transformer-based, deep learning with large-scale pretraining	Multimodal with long-context retrieval
Multimodal Support	No (text only)	Limited (GPT-4V for images)	Yes (text, images, video, audio)
Open-Source	Partially (weights available)	No (proprietary)	No (proprietary)
Best Use Cases	Research, on-device AI	General AI, reasoning, coding	Multimodal tasks, enterprise applications
Efficiency	Optimized for local deployment	Cloud-based, high-performance	High scalability with Google services

5. Conclusion and future scope

This project presents a chatbot designed to efficiently retrieve and respond to queries from PDFs containing both text and embedded images. By leveraging FAISS vector databases and Large Language Models (LLMs), the system ensures accurate and context-aware responses. Unlike traditional chatbots, which struggle with extracting meaning from visual data, our approach enhances retrieval by embedding both text and image-related information, enabling precise and efficient search. The chatbot is implemented with a Streamlit interface, allowing users to seamlessly upload documents and ask questions about college-related information, such as

course details, admission procedures, and faculty information. FAISS-based vector search optimizes retrieval, ensuring relevant text and image-based responses. Future improvements include advanced image analysis with deep learning, domain-specific fine-tuning for academic inquiries, multilingual support to assist a diverse student body, and knowledge graph integration to enhance factual accuracy. Deployment as a cloud-based solution will further improve accessibility and scalability, making it a comprehensive AI-driven inquiry system.

6. References

1. Mathew M, Nayak T, Jawahar C. DocVQA: A Dataset for VQA on Document Images. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2020;2200–10.
2. Cui L, Hu Y, Zhang Y, Wang Y. SuperAgent: A Customer Service Chatbot for E-Commerce. Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2021;3121–32.
3. Kim J, Lee S. AI-Powered Chatbots for University Information Systems: Enhancing Student Engagement and Response Accuracy. International Journal of Artificial Intelligence in Education. 2021;31(2):198–215.
4. Brown T, Mann B, Ryder N, Subbiah M, Kaplan J, Dhariwal P, *et al*. Language Models are Few-Shot Learners. Advances in Neural Information Processing Systems (NeurIPS). 2020;33:1877–901.
5. Lewis P, Perez E, Piktus A, Petroni F, Karpukhin V, Goyal N, *et al*. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. Advances in Neural Information Processing Systems (NeurIPS). 2020;33:9459–74.
6. Yih W-T, Chang M, He X, Gao J. Fusion-in-Decoder: Enhanced Knowledge Injection into Language Models. Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT). 2021;3851–62.
7. Floridi L, Chiriatti M. GPT-3: Its Nature, Scope, Limits, and Consequences. Minds and Machines. 2020;30(4):681–94.
8. Johnson M, Zhao X. Optimizing LLM Performance with Advanced Prompt Engineering. Journal of Artificial Intelligence Research. 2022;75:157–76.
9. Guzman R, Van der Maaten L. Enhancing Chatbot Performance through Fine-Tuned Prompt Optimization. Proceedings of the 2022 Conference on Computational Linguistics and Language Modeling (CLLM). 2022;245–60.
10. Johnson A, Patel V, Chen L. GPU-Accelerated Similarity Search for Large-Scale Information Retrieval. Proceedings of the 2023 IEEE International Conference on Big Data (BigData). 2023;1120–31.
11. Xu Y, He S, Li Q. Efficient Vector Search in Resource-Limited Environments. Proceedings of the International Conference on Data Engineering (ICDE). 2022;548–59.
12. Ananya G, Vanishree K. Enhancing Chatbot Accuracy with Retrieval-Augmented Generation (RAG). International Journal of Recent Technology and Engineering (IJRTE). 2023;11(3):124–30.
13. Rathi S, Chate P, Desai G, Gangji O, Kale V, Kalbhor A. Hybrid Chatbots: Integrating Rule-Based and AI Models for Enhanced Factual Accuracy. Journal of Emerging Technologies and Innovative Research (JETIR). 2023;10(11):245–55.
14. Khan A. Knowledge Graphs for Query Answering. arXiv preprint arXiv:2305.14485. 2023.
15. Khazaeli S, Arora A, Gupta P. Legal Question-Answering in the Indian Context: Leveraging Embeddings and BERT Models. arXiv preprint arXiv:2309.14735. 2023.
16. Johnson J. An Overview of Retrieval-Augmented Generation (RAG) for AI Applications. Medium Blog on AI & NLP. 2023. Available from: https://medium.com/
17. Groq. Groq API Documentation. 2024. Available from: https://groq.com/docs
18. LangChain. LangChain Documentation. 2024. Available from: https://python.langchain.com/docs/
19. Facebook AI Research (FAIR). FAISS: A Library for Efficient Similarity Search. 2021. Available from: https://github.com/facebookresearch/faiss
20. Google AI Blog. Understanding Word Embeddings in NLP. 2023. Available from: https://ai.googleblog.com