A

PROJECT SYNOPSIS

ON

RAG and LLM-Based Intelligent Document and Web Question Answering System

Submitted In Partial Fulfillment of the Requirement for the

Degree of

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE AND ENGINEERING

PROJECT GUIDE:
Mr. ADITYA RATHI

STUDENT NAME:
Ayush Kr. Karn
220060101042

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

College of Engineering Roorkee

7th, KM Haridwar, National Highway Vardhmanpuram, Roorkee, Rehmadpur, Uttarakhand 247667

Session: 2025 - 26

ABSTRACT

With the exponential growth of digital information across documents and the web, the need for intelligent, context-aware question-answering systems has become increasingly important. Traditional search engines can only return keyword-based results, which often require users to manually sift through vast content. To overcome this limitation, this project proposes an advanced **Retrieval-Augmented Generation (RAG)** and **Large Language Model (LLM)-**based intelligent question answering system.

The system enables users to upload **PDF documents** and provide **website URLs**, from which it extracts and indexes content for future question answering. Using a combination of **semantic embeddings, vector databases, and LLM-based reasoning**, the model can retrieve and generate contextually accurate answers grounded in the provided data sources. The RAG architecture ensures that every generated answer is factually supported by retrieved text segments, minimizing hallucinations common in standalone LLMs.

To ensure a smooth and interactive user experience, the entire project is implemented using the **Streamlit framework**, which provides a responsive, web-based interface for uploading files, querying content, and viewing results dynamically. The integration of **LangChain**, **FAISS/Chroma**, and **Groq LLM API** makes the system both modular and efficient. This framework represents a modern AI-driven solution for personalized knowledge retrieval from diverse and unstructured information sources.

INTRODUCTION

The growing popularity of **Large Language Models (LLMs)** such as GPT, LLaMA, and Gemini has revolutionized natural language processing. These models demonstrate remarkable abilities to understand, reason, and generate human-like responses. However, a critical challenge lies in their dependency on static training data — they cannot inherently access new or external information after training. Consequently, they may produce inaccurate, outdated, or irrelevant responses, especially for domain-specific or document-based queries.

To solve this issue, **Retrieval-Augmented Generation (RAG)** has emerged as a hybrid architecture that enhances the reasoning capability of LLMs by grounding their responses in real, retrieved data. RAG systems combine the strengths of **information retrieval** and **natural language generation**, ensuring that the final output is both accurate and contextually aware.

This project, titled **"RAG and LLM-Based Intelligent Document and Web Question Answering System,"** aims to design an AI system capable of reading and understanding content from **uploaded PDF documents** and **websites**, and answering user queries based on them. The system supports multiple document types, processes textual information into embeddings, retrieves the most relevant content chunks, and finally generates human-like answers using an LLM.

The project's **frontend is implemented using Streamlit**, which allows real-time document uploads, website link submissions, and question-answer interactions in an intuitive and user-friendly way. The system can serve as an **AI research assistant**, **knowledge retrieval bot**, or **document analyzer**. The combination of RAG and LLM ensures factual correctness, while Streamlit enhances accessibility for both developers and non-technical users.

LITERATURE REVIEW

Over the past decade, the development of question-answering systems has evolved from traditional keyword-based retrieval to advanced semantic understanding using deep learning models. Earlier systems such as **TF-IDF** and **BM25** focused on word frequency and document ranking but lacked contextual comprehension. With the introduction of **transformer-based architectures** like **BERT**, **T5**, and **GPT**, systems gained the ability to understand context, semantics, and user intent.

However, standalone LLMs, despite their power, suffer from hallucination — generating plausible but incorrect information. To mitigate this, **Meta AI introduced the RAG framework**, which augments generative models with retrieved external knowledge. In RAG, the retrieval component searches a knowledge base (such as a vector database) for relevant text passages, which are then provided as context to the generator model. This process ensures factual accuracy, contextual richness, and grounded responses.

Frameworks such as **LangChain**, **LlamaIndex**, and **Haystack** have simplified the implementation of RAG pipelines, offering modular integration with embedding models and databases like **FAISS**, **Pinecone**, and **Chroma**. Research has demonstrated that RAG systems outperform traditional LLMs in specialized domains like legal document summarization, academic Q&A, and enterprise knowledge management.

For this project, the system leverages **FAISS/Chroma for vector storage**, **Groq API for LLM inference**, and **Streamlit** for frontend deployment. Studies from OpenAI, Meta, and Hugging Face emphasize that combining retrieval with LLMs reduces hallucination and improves factual grounding by up to 35%. Thus, this project adopts RAG as the backbone of its architecture to enable reliable, dynamic, and interactive AI question-answering across both static and web data.

OBJECTIVES

The objectives of this project are as follows:

1. To develop an **AI-based intelligent question answering system** capable of understanding and responding to user queries using **Retrieval-Augmented Generation (RAG)** and **Large Language Models (LLMs)**.
2. To enable data ingestion from **multiple sources**, including **uploaded PDF documents** and **websites**, for dynamic and personalized knowledge retrieval.
3. To implement **semantic search and vector-based retrieval** using **FAISS/Chroma** databases.
4. To integrate **LangChain** as the core framework for managing retrieval and generation pipelines.
5. To build an interactive and visually appealing **Streamlit-based frontend interface** that supports uploading, querying, and displaying results.
6. To optimize system response time and reduce hallucination by grounding answers in actual retrieved content.
7. To demonstrate practical applications of RAG in areas like research assistance, corporate document analysis, and educational Q&A systems.

METHODOLOGY

The proposed system follows a multi-stage architecture integrating retrieval and generation components within a user-friendly web interface. The workflow is as follows:

1. Data Ingestion

Users can upload **PDF files** or enter **website URLs** through the Streamlit interface. The system uses **PyMuPDF** or **pdfplumber** to extract text from PDFs and **BeautifulSoup** or **LangChain WebLoader** to scrape website content.

2. Preprocessing and Chunking

Extracted text is preprocessed by removing extra spaces, stop words, and irrelevant characters. It is then divided into meaningful text chunks (e.g., 500–1000 tokens) to make them manageable for embedding and retrieval processes.

3. Embedding Generation

Each chunk is transformed into a **dense vector representation** using an embedding model such as **OpenAI Embeddings** or **SentenceTransformers**. These embeddings capture semantic meaning, allowing the system to understand context rather than just keywords.

4. Vector Storage and Retrieval

The embeddings are stored in a **vector database** such as **FAISS** or **Chroma**, enabling efficient similarity search. When a user inputs a question, the system converts it into an embedding and retrieves the top relevant chunks based on cosine similarity.

5. Contextual Answer Generation

The retrieved content is sent along with the user's question to a **Large Language Model (LLM)** (through Groq or OpenAI API). The model then generates an answer that is both **contextually accurate and semantically coherent**.

6. Streamlit Frontend Implementation

The user interface is developed using **Streamlit**, offering:

- A sidebar for navigation (PDF Upload, Website Query, Settings)
- File upload and URL input widgets
- Real-time question answering area with formatted output
- Display of retrieved document snippets for transparency

- Responsive layout for both desktop and mobile devices

7. Evaluation

The system’s performance is evaluated based on:

- Response accuracy
- Factual grounding
- Retrieval precision
- Latency (response speed)
- User interface satisfaction

Testing is performed using diverse documents and web pages to assess system adaptability and reliability.

RESULT

The developed system successfully integrates **document retrieval, semantic search, and generative AI** into one cohesive framework. Experiments show that combining **RAG** with **LLMs** substantially improves factual accuracy and reduces hallucination. The system was tested using academic papers, news websites, and PDF reports; results indicated that the generated answers were contextually correct and concise.

The **Streamlit interface** significantly enhanced user interaction, allowing seamless document uploads, website-based queries, and answer visualization without technical expertise. The response generation time was optimized using the **Groq LLM API**, which provided fast inference with reduced latency compared to conventional APIs.

This project demonstrates the feasibility of deploying real-time RAG-based systems with minimal hardware and maximum interpretability, making it suitable for educational, professional, and enterprise use cases.

CONCLUSION AND FUTURE WORK

The **RAG and LLM-Based Intelligent Document and Web Question Answering System** represents a modern AI solution for efficient and grounded information retrieval. By combining **retrieval-based search** and **generative modeling**, the system bridges the gap between static knowledge storage and dynamic conversational reasoning.

The integration of **Streamlit** provides an intuitive, browser-based interface, making the system easily accessible and deployable in any environment. This work highlights how combining RAG with advanced embedding and LLM techniques can revolutionize the way users interact with unstructured data.

Future Enhancements

1. **Support for multi-format data:** DOCX, images, and spreadsheets.
2. **Integration of multimodal LLMs** for text and image-based Q&A.
3. **Personalized user sessions** with memory-based contextual awareness.
4. **GPU-accelerated inference** for real-time performance.
5. **Deployment on cloud platforms** (AWS/GCP) for scalability and remote accessibility.

This project lays the groundwork for future research into **context-grounded conversational AI systems** capable of reading, reasoning, and responding with human-level understanding.

REFERENCES

[1] Lewis, P. et al., “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks,” *NeurIPS*, 2020.

[2] LangChain Documentation, 2024 – https://www.langchain.com/

[3] FAISS: Facebook AI Similarity Search – https://faiss.ai/

[4] OpenAI API Documentation – https://platform.openai.com/

[5] ChromaDB: Open-source embedding database – https://www.trychroma.com/

[6] GroqCloud API Documentation – https://groq.com/

[7] LlamaIndex: “Connecting LLMs with External Data,” 2024.

[8] Yao, S., “ReAct: Reasoning and Acting in Language Models,” *ICLR*, 2023.

[9] Streamlit Documentation – https://streamlit.io/

[10] Sentence Transformers, 2024 – https://www.sbert.net/