International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

NABL-Query: Using Retrieval-Augmented Generation to Answer Questions from Accreditation Documents

Shivani S. Shelar
Reserach Scholar, Shree Mahaveer Education Society's
Sanghavi College of Engineering
Nashik, India

Sandeep R. Jadhav
Asistant Professor, Shree Mahaveer Education Society's
Sanghavi College of Engineering
Nashik, India

Abstract *- The process of locating pertinent data within accreditation documentation remains challenging for labs that adhere to NABL's guidelines. Long files, complicated documents, and unstructured formats (such as PDFs) create difficulties in finding information manually, as searches are both slow and error-prone. For this reason, we developed NABL-Query—a question-and-answer tool utilizing retrieval-augmented generation (RAG), which facilitates accurate and contextual responses by using both semantic embedding and vector-based retrieval processes alongside natural language generation techniques to provide answers to user questions based on uploaded documentation. Data sources include LangChain, FAISS, and embedding model tools developed for use in building an end-to-end pipeline for understanding document content and producing responses. While we tested the method's effectiveness, our results demonstrated that the RAG approach greatly enhances retrieval accuracy and decreases response times as opposed to traditional keyword-search methods. Ultimately, NABL-Query will provide scalable means for improving the accessibility and usability of NABL documents within laboratories.*

Keywords *- NABL, Retrieval-Augmented Generation, Semantic Search, FAISS, NLP, Document QA*

I. INTRODUCTION

Accreditation ensures that laboratories operate consistently and reliably, with the National Accreditation Board for Testing and Calibration Laboratories (NABL) establishing criteria for testing and conformity; however, these documentation is often lengthy and complicated due to the fact that they are typically non-structured PDF's, which makes finding information manually very difficult via conventional keyword searches because they do not consider user intent or context [1], thus often resulting in either incomplete or irrelevant documents to the user looking to access them [1]. Artificial Intelligence (AI) and Natural Language Processing (NLP) have made advances in document interactions; however, current chatbot technologies lack the ability to access domain-specific knowledge and have limited application in specialty areas [14][15]. Semantic searching using embeddings and vector databases, such as FAISS, can facilitate improved access by leveraging metrics through similarity-based matching of contextual meaning [9][18], however most of the time, it delivers fragmented data rather than full answers.Retrieval Augmented Generation (RAG) solves this problem because it combines the benefit of semantic searching with generative models to provide contextually accurate responses to queries [2][3]. Recent frameworks such as LangChain have extended and support this approach [10][11]. This work proposes to utilize RAG-based systems for answering queries regarding NABL documents so that users can receive concise and meaningful answers using natural language when searching for information.

II. RELATED WORK

Keyword search for retrieving documents has many drawbacks: searching for keywords does not give you the context of the document or understand what the user was really looking for [1], [2]. It is especially challenging with complex documents and results can often be inaccurate [2]. With AI-based chatbots, users can have much better interaction; however, the bot does not have access to all relevant domain-specific information to provide an accurate answer [2], [3].

When using semantic search with embeddings and vector databases (such as FAISS), however, returns will only give you the relevant text part rather than the entire document [3], [5]. Additionally, you can use frameworks such as LangChain to connect semantics to retrieval and generation workflows [6], but for the most part solutions available out there are not domain-specific. Recent RAG-based frameworks such as RAGLAB and FlashRAG have demonstrated improvements in modularity and efficiency for retrieval-augmented generation systems [3], [4]. Therefore there is a clear need to create a domain-specific solution, which can combine augmented retrieval with the generation of intelligent answers through RAG to deliver accurate and contextualized answers [1], [2].

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

Table 1: Comparative Analysis of Existing Approaches and Proposed System

Approach	Key Feature	Limitation	Advantage of Proposed System
Traditional Search	Keyword-based retrieval	Lacks semantic understanding	Enables context-aware information retrieval
AI Chatbots	Natural language interaction	Limited to pre-trained knowledge	Utilizes real-time document-based knowledge
Vector Databases	Semantic search using embeddings	Does not generate complete answers	Integrates retrieval with response generation
Existing QA Systems	Partial automation	Not domain-specific	Customized for NABL documentation
Proposed RAG System	Retrieval + generation	—	Accurate, fast, and context-aware responses

III. PROPOSED METHODOLOGY

The proposed system (an intelligent framework for retrieving information from NABL documents using Retrieval-Augmented Generation) combines the use of semantic retrieval and natural language generation to provide accurate and context-based answers to queries. This intelligent framework will function by integrating document preprocessing, vector similarity search, and generative modelling to overcome the deficiencies of keyword-based retrieval systems [1], [2]; Because the proposed system is modular in architecture, it can also efficiently handle retrieval of large document collections.

3.1 Architecture Components

Systems architecture includes the following components:

User Interface (UI):
Developed using Streamlit, the UI allows users to upload NABL PDF documents and enter queries in natural language.

Document Processing Module:
Responsible for taking the text extracted from the uploaded PDF's and performs the necessary pre-processing on it (cleaning, chunking, etc.) to ensure that retrieval of document-level text is efficient.

Embedding Module:
Embeds textual information from the uploaded PDF's into vector representations using OpenAI Embeddings, which enables the generation of answers based upon semantic understanding rather than just simple keyword matching [2].

Vector Storage Module:
Embeds vectors retrieved from the Embedding Module into the FAISS storage format, allowing for rapid (high-dimensional) indexing of document vectors to enable fast similarity-based retrieval[s][5][9].

Processing Pipeline:
Managed by LangChain, the Processing Pipeline will bring together the retrieval and generation processes as part of a single workflow [6][10].

RAG Engine:
Retrieves relevant chunks of documents from the Shared Memory Module and returns/converts this information to generate an accurate answer to a user's query based upon the information retrieved [1].

The combined capabilities of semantic search and Generative AI will allow for efficient and accurate retrieval/generation of answers, respectively.

$$P(a \mid q, D) = \sum_{c \in C} P(a \mid q, c) \cdot P(c \mid q, D)$$

……. Equation (1)

Where:

- a: Generated answer produced by the system
- q: User query (input question)
- D: Entire document collection (all uploaded NABL PDFs)
- C: Set of retrieved document chunks relevant to the query
- $c \in C$: Each individual chunk from the retrieved set
- $P(c \mid q, D)$: Probability that chunk $c$is relevant to query $q$given documents D
  $P(a \mid q, c)$: Probability of generating answer $a$using query $q$and chunk c
  $\sum_{c \in C}$:Summation over all retrieved chunks

This method uses semantic search and generative AI to quickly find the right information and provide accurate answers.

The system is a pipeline that incorporates three components of the entire document processing (semantic retrieval and response generation, by using RAG [1], [2]). The steps that the system goes through while processing a document are:

1. Document Upload - Users upload documents (NABL pdf) via the Streamlit interface.

2. Text Extraction - The system extracts & preprocesses text from the documents.

3. Chunking - The text from the documents is chunked into smaller parts for easier processing.

4. Embedding Generation - The system turns each chunked part into a vector embedding using OpenAI Embeddings [2].

5. Vector Storage - All vector embeddings are saved in FAISS for fast, similarity-based retrieval of text [5].

6. Query Input - The user submits a natural language query to the system.

7. Retrieval - The system retrieves any relevant chunks of text by performing a similarity search through FAISS [5].

8. Response Generation - The retrieved chunks of text are processed through LangChain with RAG, to create contextually relevant answers [1], [6].

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

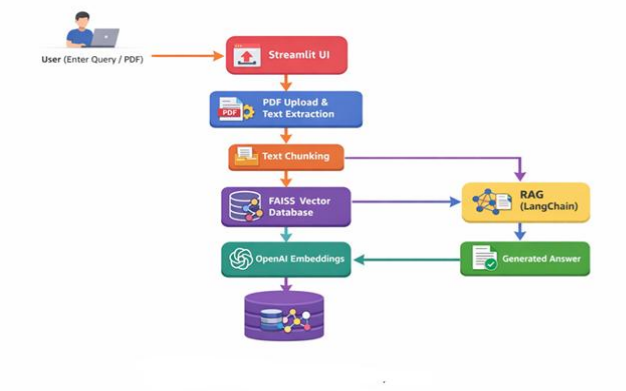

Figure. 1. NABL-Query System Architecture with Retrieval-Augmented Generation (RAG).

Streamlit is employed as the front-end interface, while Python serves as the back-end engine. LangChain is an orchestration tool that manages workflows when documents are uploaded, retrieved, and generated as responses. FAISS provides a vector database for carrying out rapid similarity searches to help find the relevant sections of a document in response to a user's query. There is a secure login for approved users to log into the system and upload their NABL-related PDF documents for processing after they log into the system (Figure 2).

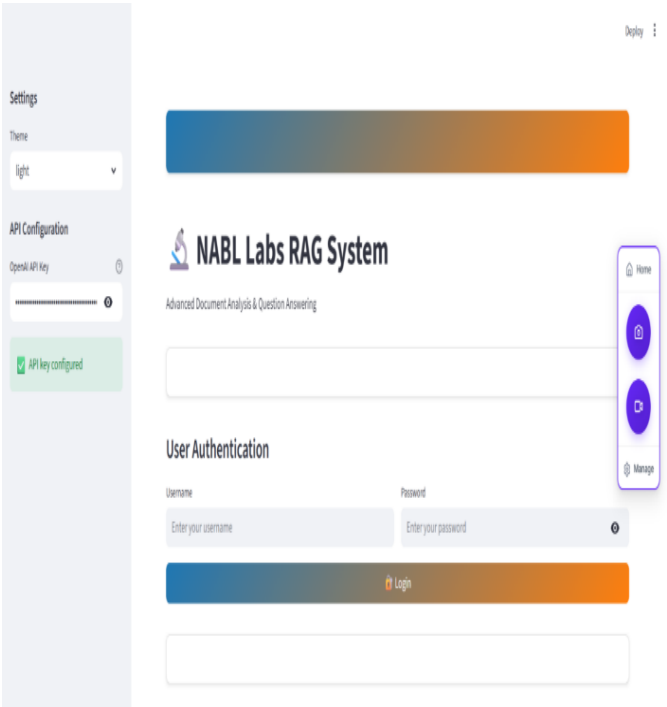

Figure 2 User Authentication (Login Interface)

The User Interface (see Figure 3) allows users to upload PDF files for processing purposes. Once uploaded, the system extracts the text from the files and creates pre-processing parameters necessary for embedding the text and performing any remaining processing steps.

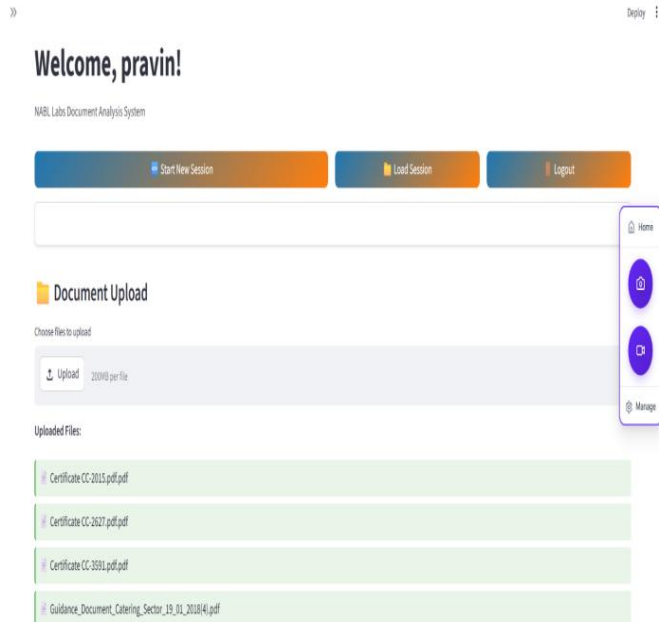

Figure 3: Document Processing and Upload Status

IV. RESULTS & DISCUSSION

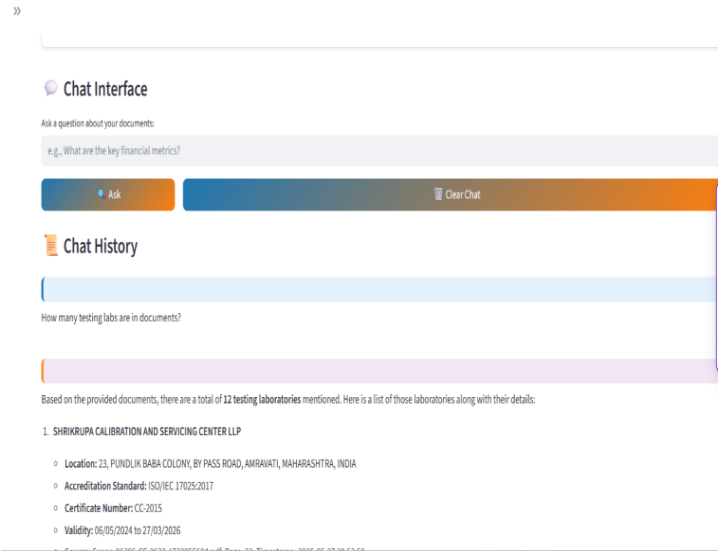

Figure 4: Query Interface for Document Question Answering

The system was evaluated on NABL documents and produced accurate, context-aware results. The query interface (Figure. 4) allows users to ask questions in natural language, improving usability. RAG enhances response quality by combining semantic retrieval and generation, with FAISS enabling fast retrieval and LangChain managing the workflow. Figure 5 is an Example of a Generated Answer with Laboratory Information. The generated answer provides accurate information about the laboratory, such as its location and accreditation standards, as well as the Laboratory's certificate number and valid through date. This has greatly reduced the time spent searching for laboratory information

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

using manual methods and has improved the relevance of results, demonstrating the System's accuracy and usability for NABL document analysis.

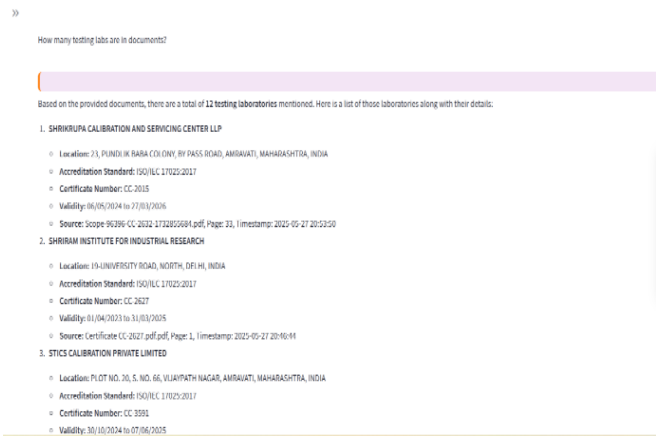

Figure 5: Sample system output showing the extracted laboratory information

CONCLUSION

This paper presents a RAG-based question answering system for NABL documents, integrating LangChain, FAISS, and OpenAI Embeddings to generate accurate and context-aware responses. The system significantly reduces manual search effort, improves retrieval efficiency, and enhances decision-making in laboratory environments. The results demonstrate reliable performance and scalability, making the system suitable for real-world applications. By combining semantic retrieval with natural language generation, it effectively understands user queries and delivers meaningful answers. The approach can be extended to other document-intensive domains, with future improvements such as support for multiple document formats and real-time deployment.

REFERENCES

[1] Patrick Lewis et al., “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks,” in Advances in Neural Information Processing Systems (NeurIPS), 2020.

[2] Y. Gao et al., “Retrieval-Augmented Generation for Large Language Models: A Survey,” arXiv preprint, 2023.

[3] X. Zhang et al., “RAGLAB: A Modular Framework for Retrieval-Augmented Generation,” arXiv preprint, 2024.

[4] J. Jin et al., “FlashRAG: Efficient Toolkit for RAG Research,” arXiv preprint, 2024.

[5] C. Jin et al., “RAGCache: Efficient Knowledge Caching for RAG,” arXiv preprint, 2024.

[6] S. MS, “Evaluation of RAG Pipeline for Document Question Answering,” International Journal for Research in Applied Science & Engineering Technology, 2025.

[7] Y. Xiong et al., “Retrieval-Augmented Generation for NLP: A Survey,” 2024.

[8] F. Ye et al., “R²AG: Improving Retrieval-Augmented Generation,” arXiv preprint, 2024.

[9] Matthijs Douze et al., “The FAISS Library,” Meta AI, 2024.

[10] LangChain Team, “LangChain Framework for LLM Applications,” 2024.

[11] Harrison Chase, “LangChain: Building LLM Applications,” 2022.

[12] NVIDIA, “What is Retrieval-Augmented Generation (RAG)?,” 2023.

[13] Papers with Code, “RAG Method Overview,” 2024.

[14] Frontiers AI, “LLMs for Multi-Document Query Systems,” 2025.

[15] R. Yang et al., “Knowledge Integration in LLM-based QA Systems,” 2023.

[16] D. Varshney et al., “Knowledge-Based Dialogue Systems using AI,” 2023.

[17] H. Wu et al., “Evaluation of RAG Systems in AI Pipelines,” 2024.

[18] Meta AI, “FAISS Vector Search Research,” 2023.