# Adaptive RAG Chat-bot for Reliable Information Retrieval

Mohamed Farrag, Rayyan Husain

## Introduction

LAI assistants leveraging Retrieval-Augmented Generation (RAG) systems are increasingly used across various fields. In customer support, they provide precise and contextually relevant answers. In education, they offer personalized tutoring and answer academic queries. In finance, they generate insightful analyses and reports. In the legal field, they assist by retrieving and summarizing pertinent legal documents and case laws. RAG systems enhance accuracy by grounding responses inreal data, reducing the likelihood of incorrect or hallucinated information. They offer more contextually appropriate responses by using specific information from relevant documents and are crucial for applications requiring up-to-date response. However, challenges include delays in real-time performance and the need for advanced indexing and search techniques to ensure effficient and accurate retrieval from large corpora

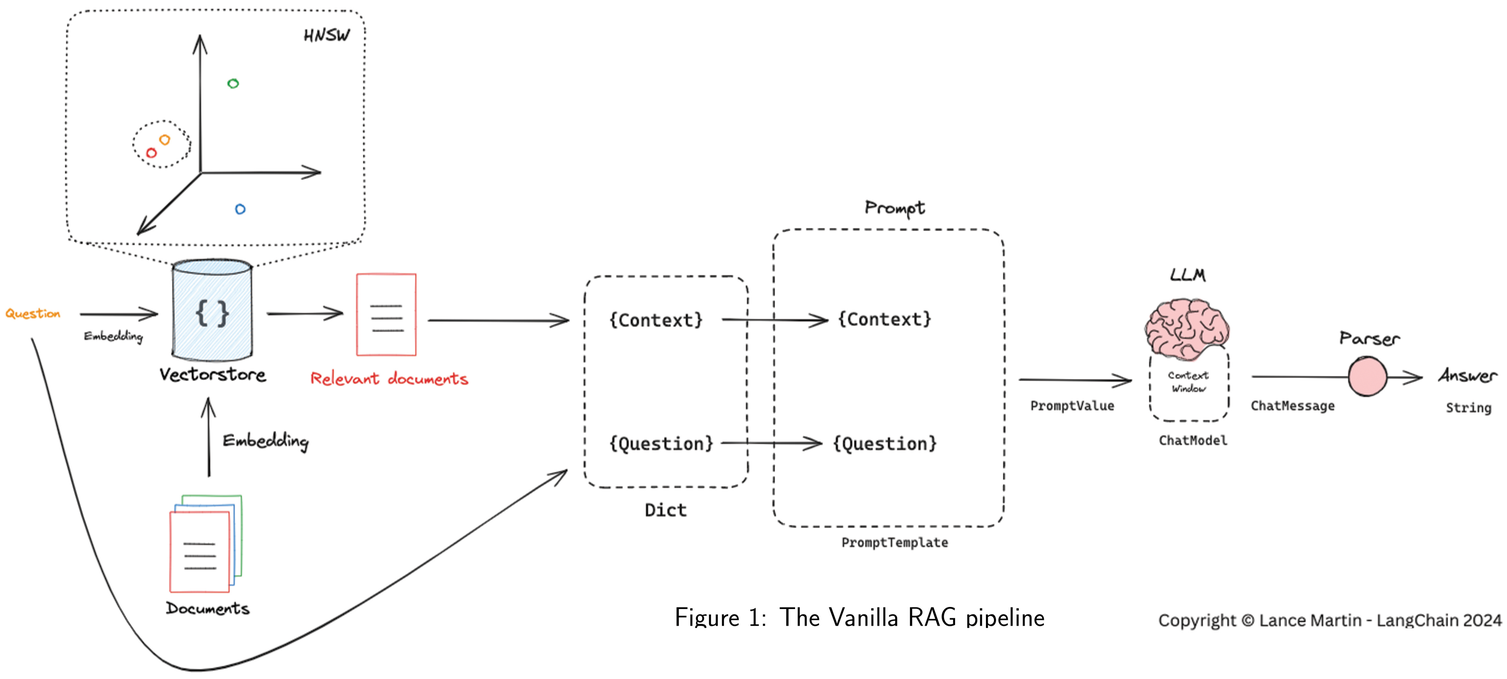


Figure 1: The Vanilla RAG pipeline

The basic pipeline of RAG systems shown in Figure 1 involves several key steps. First, the system receives a question or query from the user. Then, relevant documents are retrieved from a pre-indexed corpus using similarity search techniques. The retrieved documents are used as context to generate a coherent and contextually relevant response. By integrating retrieval mechanisms with generative models, RAG systems enhance the reliability and relevance of AI-generated content and broadening the applicability of AI assistants across diverse domains .

## Adaptive RAG System Architecture

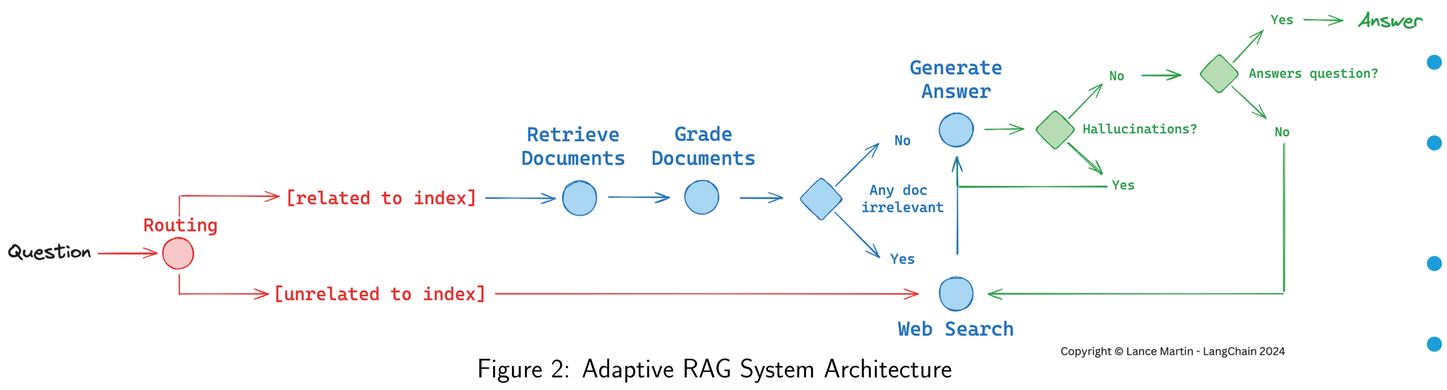


Figure 2: Adaptive RAG System Architecture

The architecture of the Adaptive RAG system illustrated in Figure 2 shows that upon receiving a user's question, the system first determines if it relates to indexed documents. If related, it retrieves documents from the vector store; if not, it initiates a web search. For index-related questions, the system queries the vector store to find the most similar documents based on vector embeddings. These documents are then graded for relevance, discarding any irrelevant ones. Once relevant documents are identified, the system generates an answer using them. This answer is evaluated for hallucinations. If reliable, it moves to the final stage; otherwise, a web search is performed for additional information. In the final evaluation, the system checks if the answer satisfactorily addresses the question. If so, it is provided to the user; if not, the system self-corrects or offers an alternative response. This architecture ensures the RAG system retrieves and processes relevant information efficiently to generate accurate and reliable answers.

### Features of the proposed System:

- **Online Search:** If no relevant document is found in the database, the system searches online for an answer.
- **Answer Grounding:** The system evaluates if the generated answer is grounded in the database documents or is a result of the model's creativity.
- **Hallucination Detection:** The system checks for hallucinations and ensures the answer is consistent.
- **User Interface:** A user-friendly front-end allows users to interact with the system, add documents/URLs, adjust settings, and ask questions.

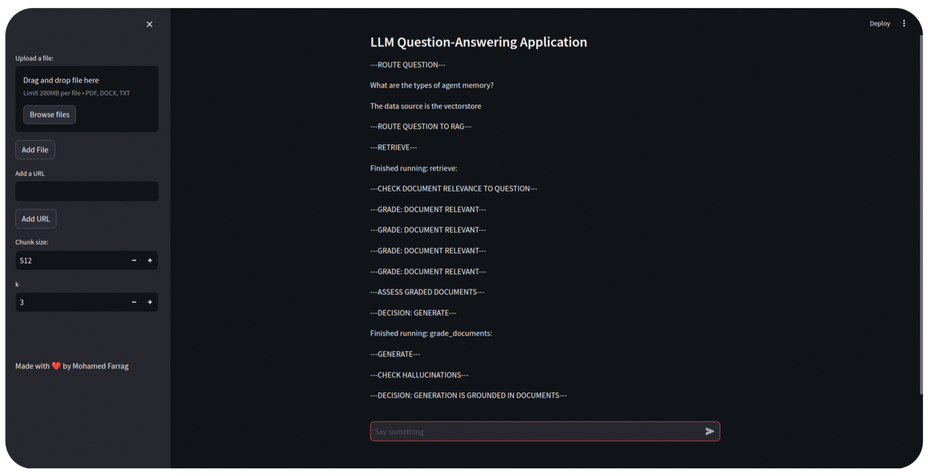


Figure 3: Adaptive RAG Web application UI

## Used Technologies

- **LangChain & Lang-graph**
- **Groq API**
- **Chroma DB**
- **GPT4All & Llama 3**
- **Tavily API**
- **Streamlit**

Chroma · Meta LLAMA 3 · Tavily · LangChain · Streamlit · groq

## Future Work

- **Enhancing PDF Embedding:** PDF is an unstructured data type that contains text, images and diagrams. So, enhancing the embedding of PDFs will enhance information augmentation and retrieval.
- **More file formats to be embedded:** Text can be available with different formats, adapting the RAG app to handle many formats will increase its agility.
- **Enhancing the UX:** The Front-End side of the app will need more work in order to present the reach amount of information in a more informative ways which will enhance the user experience.
- **More customizable parameters:** More flexibility in changing the system parameters specially customizing the system prompts for each node.

## References

[1] Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong C. Park. Adaptive-rag: Learning to adapt retrieval-augmented large language models through question complexity, 2024.

[2] Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection, 2023.

[3] Shi-Qi Yan, Jia-Chen Gu, Yun Zhu, and Zhen-Hua Ling. Corrective retrieval augmented generation, 2024.

[4] LangChain AI. Langgraph: A tool for building and analyzing knowledge graphs with langchain., 2024. Accessed: 2024-06-24.

## Acknowledgement

I extend my sincere gratitude to **Prof. Dr. Jörn Hees** and **Tim Metzler** from H-BRS for their invaluable lectures and lab classes. Special thanks to **Lance Martin** from LangChain for his excellent tutorials and explanations. Their support made this project possible.

## Contact

Mohamed Farrag
Hochschule Bonn-Rhein-Sieg
Email: mfarrag97@gmail.com

b-it Bonn-Aachen International Center for Information Technology

Hochschule Bonn-Rhein-Sieg
University of Applied Sciences