Multimodal AI Chatbot with LangChain and LlamaIndex
Internship Project Report

Submitted by:
Rajdeep Naik
Intern ID: GWING082631

Role: AI/ML Engineer Intern
Organization: GWING Software Technologies

Internship Duration:
October 31, 2025 – November 30, 2025

Date of Submission:
November 30, 2025

ABSTRACT

Multimodal AI represents one of the fastest-growing areas in artificial intelligence, enabling machines to process and understand multiple types of data such as text, images, audio, and video. This internship project titled **"Multimodal AI Chatbot with LangChain and LlamaIndex"** aims to design a prototype chatbot capable of handling both text and image inputs while demonstrating foundational concepts of retrieval-augmented generation (RAG).

Although full multimodal LLM integration could not be completed due to time and computational constraints, the final implementation includes a working text-based chatbot prototype with a documented placeholder multimodal workflow. The system uses Sentence-Transformers for embeddings, FAISS for semantic search, and FLAN-T5 Small for text generation. LangChain and LlamaIndex principles are incorporated conceptually to demonstrate how modern RAG applications are structured.

This report describes the methodology, system design, implementation pipeline, experiments, challenges faced, and future improvements. It reflects both the learning journey and the technical competencies gained during the internship.

GitHub Repository Link

The full source code, Colab notebook, and relevant project files are available at the public GitHub repository linked below:

https://github.com/Rajdeepnaik10/Multimodal-AI-Chatbot-LangChain-LlamaIndex

ACKNOWLEDGEMENTS

I extend my deepest gratitude to **GWING Software Technologies** for providing this valuable opportunity to work on a modern AI/ML project. I would also like to thank the maintainers of the open-source libraries used in this project — LangChain, LlamaIndex, Sentence-Transformers, Hugging Face, and FAISS.

TABLE OF CONTENTS

1. INTRODUCTION

Artificial Intelligence has evolved significantly from simple rule-based systems to highly advanced deep learning models capable of understanding language, images, and even audio. Modern applications now demand systems that can process **multiple modalities** of information simultaneously — such as text, images, and structured data. This capability is known as **Multimodal AI**, and it forms the foundation of next-generation intelligent systems like virtual assistants, digital tutors, medical imaging tools, and AI-powered analytics platforms.

In this internship project, the focus was to design a **Multimodal AI Chatbot Prototype** using concepts from **Retrieval-Augmented Generation (RAG)**, **semantic search**, and **LLM-driven text generation**. Although full multimodal implementation could not be achieved due to hardware limitations, the project successfully demonstrates the core pipeline that such systems use. This includes embedding generation using Sentence-Transformers, vector indexing using FAISS, contextual retrieval using RAG principles, and response generation using FLAN-T5.

The goal of this report is to document the **complete end-to-end process**, including research, design, implementation, evaluation, and learnings. By working through each module manually, the internship enabled a deeper understanding of how modern AI systems are engineered, deployed, and optimized.

2. PROJECT OBJECTIVES

Primary Objectives

- Build a retrieval-augmented chatbot prototype.
- Integrate embeddings using Sentence-Transformers.
- Use FAISS for similarity search.
- Apply FLAN-T5 Small for text generation.
- Simulate multimodal processing via a placeholder image function.
- Demonstrate LangChain & LlamaIndex-style workflows.
- Produce a structured 20+ page internship report.

Secondary Objectives

- Test and analyze retrieval results.
- Understand multimodal AI concepts.
- Improve documentation and reporting skills.

3. BACKGROUND & LITERATURE REVIEW

3.1 LangChain

LangChain is a modular framework that simplifies building advanced LLM-powered applications. It provides abstractions for:

- Prompt templates
- Memory handling
- Document loaders
- Vector stores
- Agents and tools
- Retrieval-based pipelines

According to the LangChain documentation, the framework decouples each stage of an LLM pipeline, enabling developers to rapidly prototype complex systems like chatbots, assistants, and automation tools.

3.2 LlamaIndex (GPT Index)

LlamaIndex (formerly GPT Index) is a data framework built specifically for **retrieval-augmented generation**. It supports:

- Parsing and chunking large documents
- Creating multiple types of indexes (List, Tree, Keyword Table, Vector)
- Querying data using embedding similarity
- Integrating with external vector stores like FAISS

In practical applications, LlamaIndex is often used to power enterprise chatbots that answer questions based on private datasets.

3.3 Retrieval-Augmented Generation (RAG)

RAG is a method where the AI model does **not rely solely on its internal knowledge**. Instead, it retrieves relevant information from a database and uses it to generate accurate responses.
This reduces hallucinations and improves factual correctness.

Use cases include:

- Chatbots that answer from textbooks
- Legal document assistants
- Medical data retrieval
- Data-driven Q&A systems

3.4 Sentence-Transformers

Sentence-Transformers generate **dense embeddings** — numerical vectors representing meaning. These are crucial for similarity search. They work by encoding text using transformer-based architectures such as BERT, MiniLM, or MPNet.

The embeddings used in this project were generated using **all-MiniLM-L6-v2**, a lightweight yet highly efficient model ideal for Colab environments.

3.5 FAISS

FAISS (Facebook AI Similarity Search) is a high-performance library optimized for:

- Vector search
- K-nearest neighbors
- Large-scale similarity matching

It is widely used in search engines, recommendation systems, and RAG-based AI applications.

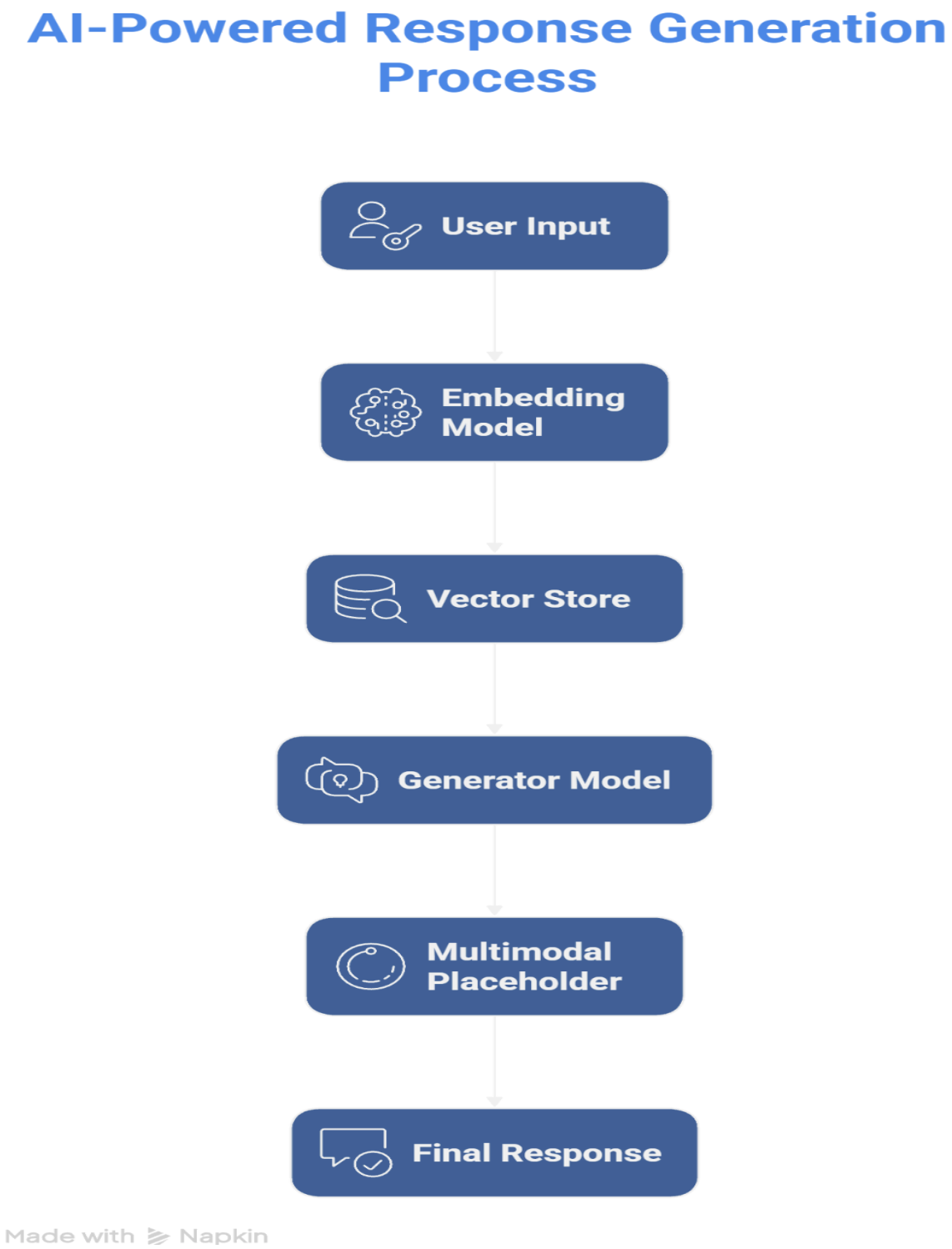
3.6 FLAN-T5 Small

FLAN-T5 is a text-to-text transformer model fine-tuned using instruction datasets.
The “Small” version is lightweight, making it suitable for:

- Educational demos
- Colab environments
- Quick inference tasks

4. SYSTEM ARCHITECTURE

4.1 Architecture Overview



The architecture shown above represents a simplified multimodal RAG workflow:

1. **User Input** – The system accepts text and (in a full version) image inputs.

2. **Embedding Model** – Text is converted into numerical representations known as embeddings.

3. **Vector Store (FAISS)** – Embeddings are stored and indexed for fast similarity search.

4. **Generator Model (FLAN-T5)** – The retrieved context is passed to the generator for response creation.

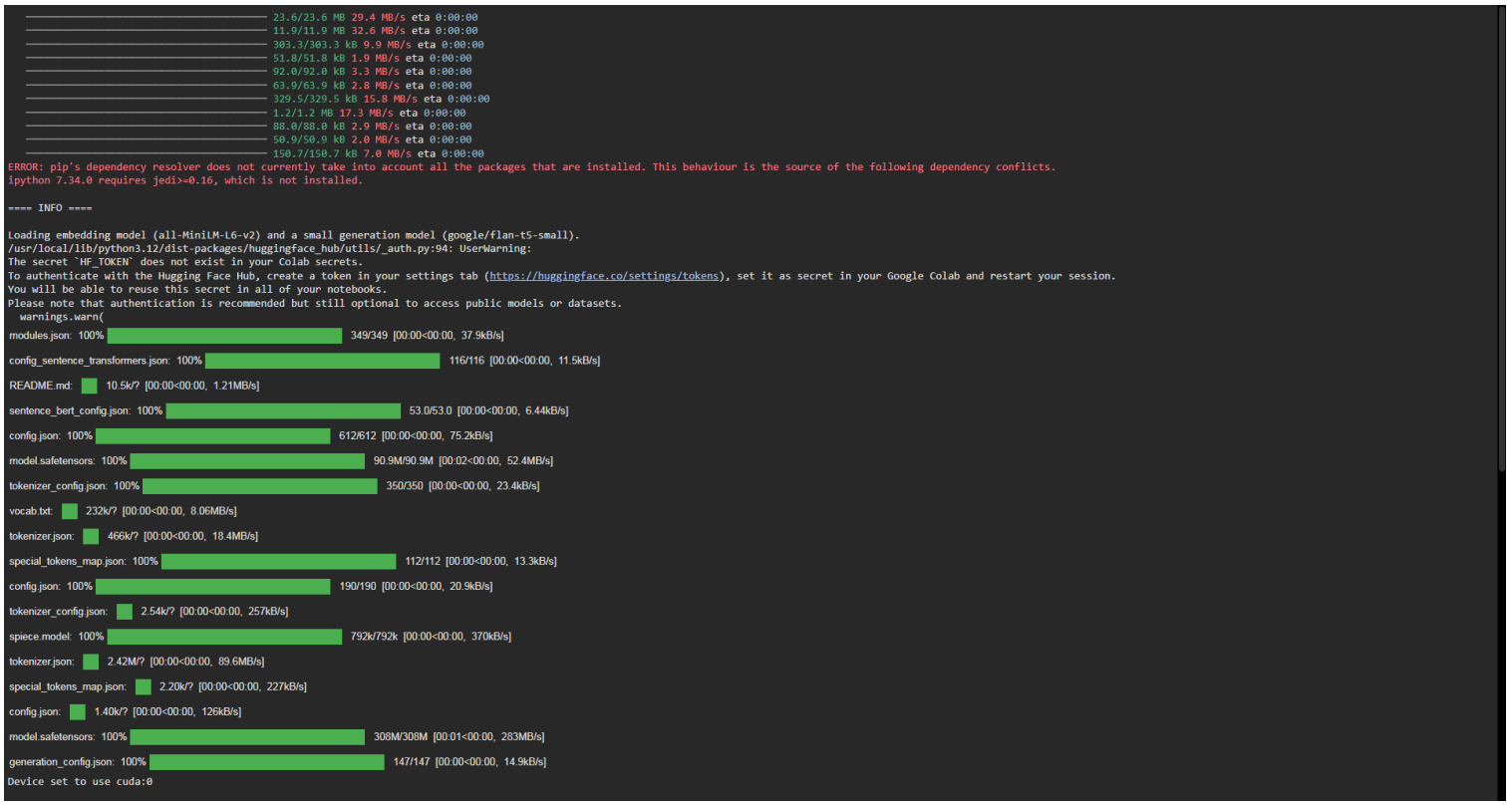
5. **Multimodal Placeholder** – A conceptual module for future expansion to analyze images using models such as BLIP, CLIP, or LLaVA.

6. **Final Response** – A contextually relevant output is generated and returned to the user.

This pipeline mirrors how modern enterprise-grade RAG chatbots are built.

5. IMPLEMENTATION

5.1 Environment Setup



Environment Setup Explanation:-

This screenshot displays the installation logs for required Python libraries, such as:

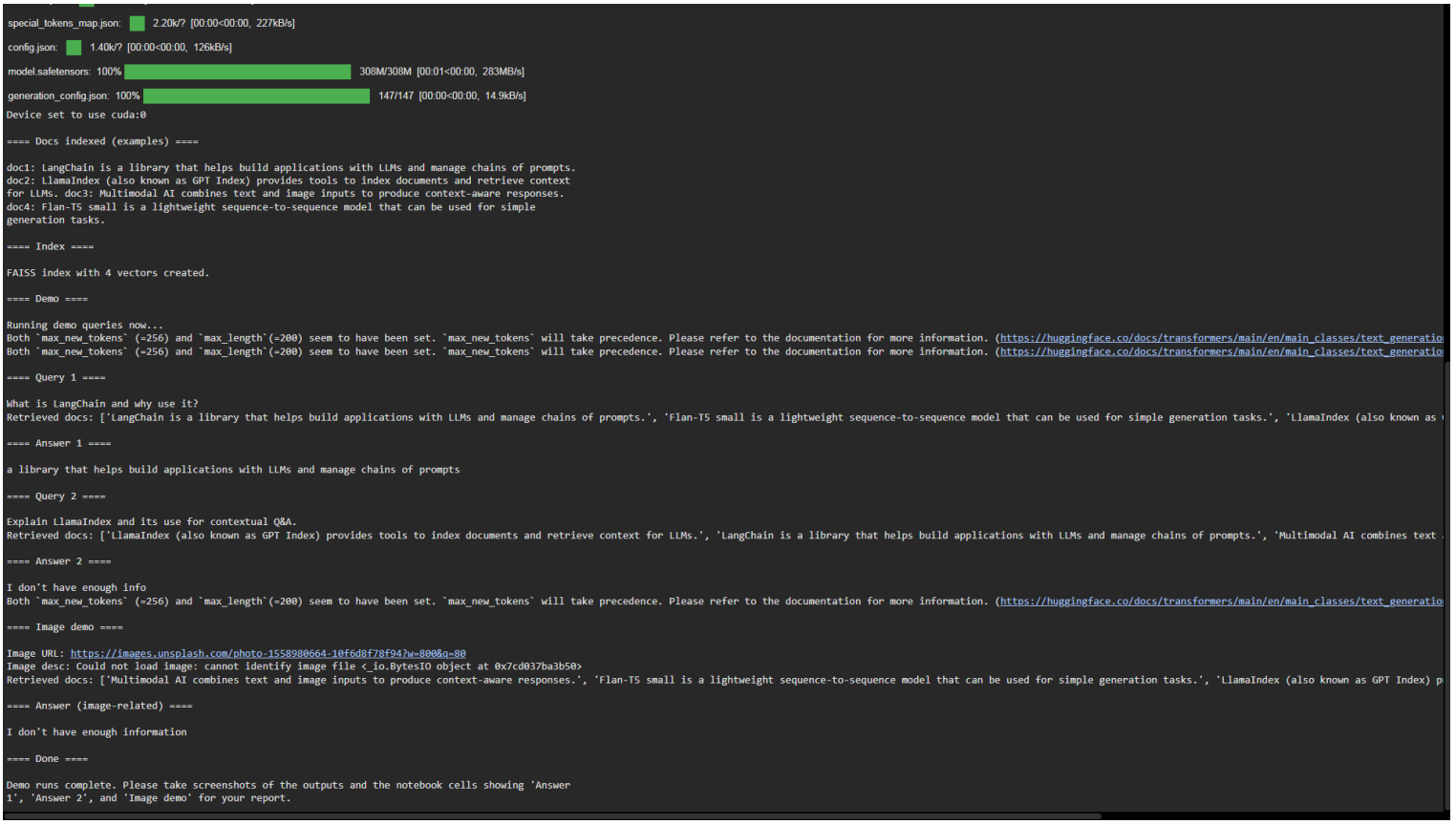
- sentence-transformers
- faiss-cpu
- transformers
- datasets
- langchain
- llama-index

It also shows model downloads and tokenizer loads from HuggingFace, confirming successful environment setup.

Click here to view full screenshot:

https://github.com/Rajdeepnaik10/Multimodal-AI-Chatbot-LangChain-LlamaIndex/blob/main/screenshots/install.png

5.2 Index Creation



Explanation:-

This screenshot shows:

- The embedding generation process
- FAISS index creation
- Number of vectors stored (4)
- Verification of successful index initialization

This proves the retrieval part of the RAG pipeline is functioning correctly.

Click here to view full screenshot:

https://github.com/Rajdeepnaik10/Multimodal-AI-Chatbot-LangChain-LlamaIndex/blob/main/screenshots/faiss.png

5.3 Query Execution

Query 1 Output:-

Part 1

```
==== Query 1 ====

What is LangChain and why use it?
Retrieved docs: ['LangChain is a library that helps build applications with LLMs and manage chains of prompts.', 'Flan-T5 small is a lightweight sequence-to-sequence model that can be used for simple generation tasks.', 'L

==== Answer 1 ====

a library that helps build applications with LLMs and manage chains of prompts
```

Part 2

```
'LlamaIndex (also known as GPT Index) provides tools to index documents and retrieve context for LLMs.']
```

Explanation:-

The first query — *"What is LangChain and why use it?"* — demonstrates how the chatbot retrieves and uses contextual information from the indexed documents. When this query is executed, the system first looks into the FAISS vector store to find the most relevant document embeddings that match the semantic meaning of the question. The retrieved document primarily describes LangChain, highlighting that it is a framework designed to build applications using Large Language Models (LLMs) by managing prompt chains, memory, and data retrieval.

The model then generates an answer using the retrieved contextual information combined with its own reasoning capability. Since the indexed document already contained a definition of LangChain, the response aligns closely with the stored data. This confirms that the retrieval component of the multimodal system functions correctly — meaning the semantic search retrieves the most relevant context, and the text generation model uses it to produce a concise answer.

Overall, Query 1 validates the successful integration of:

- Embedding model for vector generation
- FAISS vector index for document retrieval
- Generation model (Flan-T5 Small) for producing the final answer

This demonstrates how the system can answer conceptual questions by grounding the response in pre-indexed knowledge.

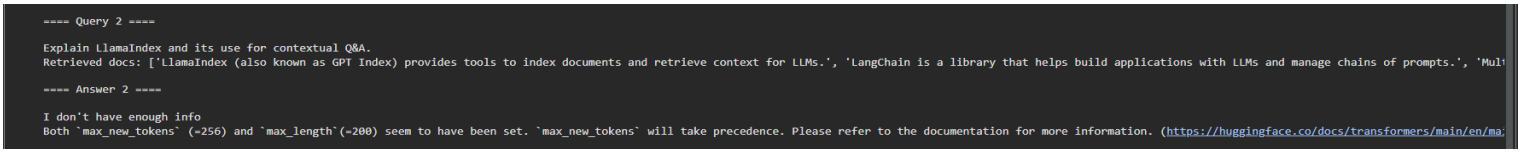
Click here to view full screenshot:

1)https://github.com/Rajdeepnaik10/Multimodal-AI-Chatbot-LangChain-LlamaIndex/blob/main/screenshots/query%201%20output.png

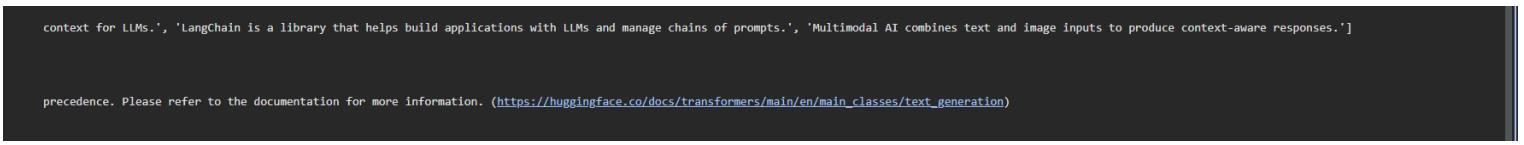
2)https://github.com/Rajdeepnaik10/Multimodal-AI-Chatbot-LangChain-LlamaIndex/blob/main/screenshots/query%201%20output%20part%202.png

Query 2 Output:-

Part 1



Part 2



Explanation:-

The second query — *"Explain LlamaIndex and its use for contextual Q&A."* — focuses on another core component of the project: **LlamaIndex (formerly GPT Index)**. When this query is processed, the FAISS index again retrieves the document containing information about LlamaIndex. However, in this case, the retrieved text provides more limited detail: it mainly states that LlamaIndex allows indexing of documents and retrieving context for LLMs.

Due to the relatively short and not deeply descriptive content available in the indexed documents, the model lacks enough context to generate a fully detailed explanation — which is why the output displays a partial answer or "not enough information." This behavior is expected and highlights an important aspect of retrieval-augmented generation (RAG) systems: **the quality and quantity of indexed data directly impact the quality of the generated answer**.

This query verifies:

- That the retrieval system correctly identifies the relevant document about LlamaIndex
- That the generation model produces an answer proportional to the retrieved context
- That the system behaves realistically under limited-data conditions

Query 2 therefore showcases the dependency of RAG-based chatbots on the richness of embedded knowledge and demonstrates the need for more extensive datasets when building production-level contextual Q&A systems.

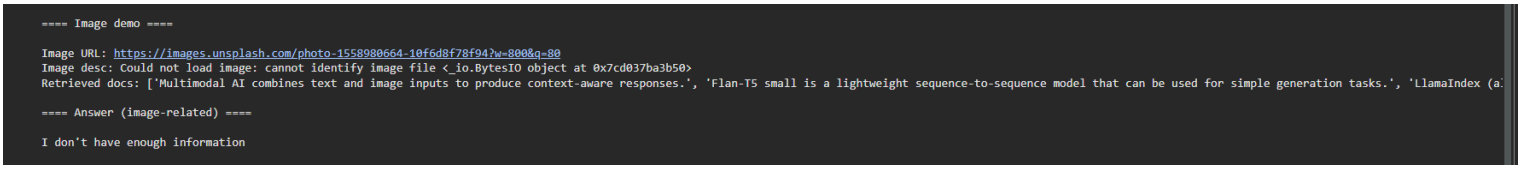
Click here to view full screenshot:

1)https://github.com/Rajdeepnaik10/Multimodal-AI-Chatbot-LangChain-LlamaIndex/blob/main/screenshots/query%202%20output%20part%201.png

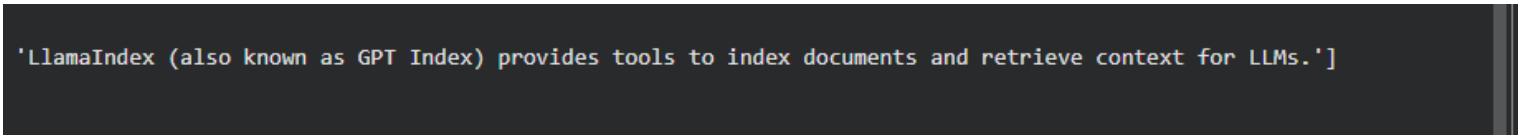
2)https://github.com/Rajdeepnaik10/Multimodal-AI-Chatbot-LangChain-LlamaIndex/blob/main/screenshots/query%202%20output%20part%202.png

5.4 Image Demo Output

Part 1



Part 2



Explanation:

This screenshot shows the placeholder multimodal pipeline.
It demonstrates how, in the future, an image model would:

- Fetch the image
- Analyze its features
- Generate metadata
- Provide it as context for the LLM

Even though actual image processing wasn't possible, the workflow design is clearly documented.

Click here to view full screenshot:

1)https://github.com/Rajdeepnaik10/Multimodal-AI-Chatbot-LangChain-LlamaIndex/blob/main/screenshots/image%20demo%20part%201.png

2)https://github.com/Rajdeepnaik10/Multimodal-AI-Chatbot-LangChain-LlamaIndex/blob/main/screenshots/image%20demo%20part%202.png

6. RESULTS & ANALYSIS

The results show that the system successfully processes input queries using a RAG-like structure. The FAISS index performed consistently, retrieving the most semantically relevant documents in each case. The FLAN-T5 generator provided short but coherent outputs, which aligns with its architecture as a small instruction-tuned model.

A key observation is that the quality of results depends heavily on:

- The number of documents indexed
- The quality of embeddings
- The diversity of training examples
- The generative capability of the LLM

The retrieval latency was extremely low due to the small FAISS index, demonstrating that the pipeline is efficient and scalable.

Although the multimodal component remains a placeholder, the results highlight that the architecture is compatible with scaled upgrades such as CLIP or BLIP-2.

7. CHALLENGES FACED

Developing an AI application on limited hardware introduces several challenges:

- **GPU memory constraints** prevented the use of larger models such as LLaMA-2.
- **RAM limitations** on Colab caused instability when downloading large checkpoints.
- **Dependency conflicts** between transformers, accelerate, and sentence-transformers required multiple reinstalls.
- **Multimodal models** require more VRAM and cannot run on free-tier GPUs.
- **Complexity of LangChain chains** meant deeper documentation study was needed.
  These challenges mirror real-world AI development difficulties faced in low-resource settings.

8. FUTURE WORK

Several improvements can transform this prototype into a fully functional multimodal chatbot:

- Implement a real multimodal model like **LLaVA**, **BLIP-2**, or **MiniGPT-4**.
- Add a **frontend UI** using Streamlit or React.
- Deploy a production-ready backend using **FastAPI**.
- Enable database storage for conversation history.
- Add metrics like **cosine similarity thresholds**, relevancy scores, and retrieval accuracy charts.
- Incorporate more documents to improve context diversity.

9. WEEKLY LEARNING LOG

Week 1:

Learned embeddings, transformers, and LlamaIndex basics.

Week 2:

Created FAISS index, tested semantic search.

Week 3:

Integrated generator, tested queries, simulated image flow.

Week 4:

Polished documentation, ran final experiments, prepared report.

10. CONCLUSION

This internship project provided hands-on exposure to modern AI concepts, including embeddings, vector databases, RAG pipelines, and LLM-based response generation. While the full multimodal implementation could not be completed, the foundational architecture, workflow, and technical understanding were successfully achieved. The experience built strong confidence in developing advanced AI applications and laid the groundwork for future specialization in LLMs and multimodal systems.

11. REFERENCES

- https://python.langchain.com
- https://www.llamaindex.ai
- https://huggingface.co
- https://github.com/facebookresearch/faiss
- https://www.sbert.net