TASK-4: Architecture & Evaluation Report

➔ About the Project

This project implements an **Agentic Document Question Answering (QA) System** that enables users to ask natural language questions over uploaded documents and receive accurate, context-grounded responses. The system is designed using an agent-based Retrieval-Augmented Generation (RAG) architecture and emphasizes **local LLM inference**, **modular design**, and **containerized deployment**.

The project was developed as part of an **Internship Technical Task** and demonstrates practical implementation of modern AI workflows using LangGraph, vector databases, and FastAPI.

➔ Project Description

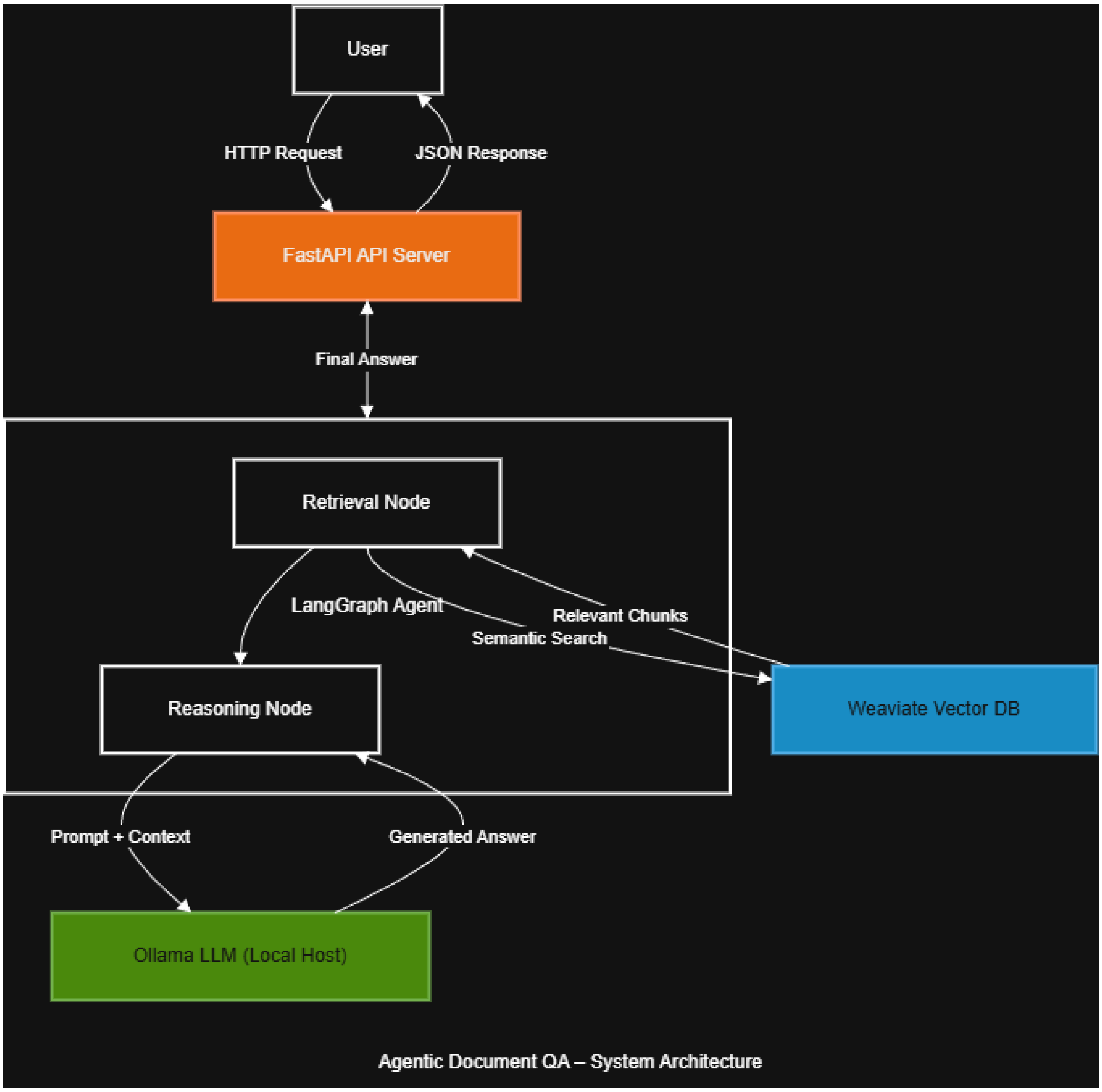
The system ingests PDF documents, splits them into semantically meaningful chunks, and stores vector embeddings in a Weaviate vector database. When a user submits a query, a LangGraph-based agent orchestrates the workflow by retrieving relevant document chunks and passing them to a local Large Language Model (LLM) running via Ollama. The model generates responses strictly grounded in the retrieved content, ensuring accuracy and minimizing hallucinations.

The application exposes its functionality through a RESTful API built with FastAPI and is fully containerized using Docker and Docker Compose for reproducible deployment.

➔ Technologies Used

Component	Technology
Agent Framework	LangGraph
LLM Framework	LangChain
Vector Database	Weaviate
Embedding Model	Sentence-Transformers
LLM Inference	Ollama (Local)
API Framework	FastAPI
Containerization	Docker, Docker Compose
Programming Language	Python

➔ Architecture Diagram



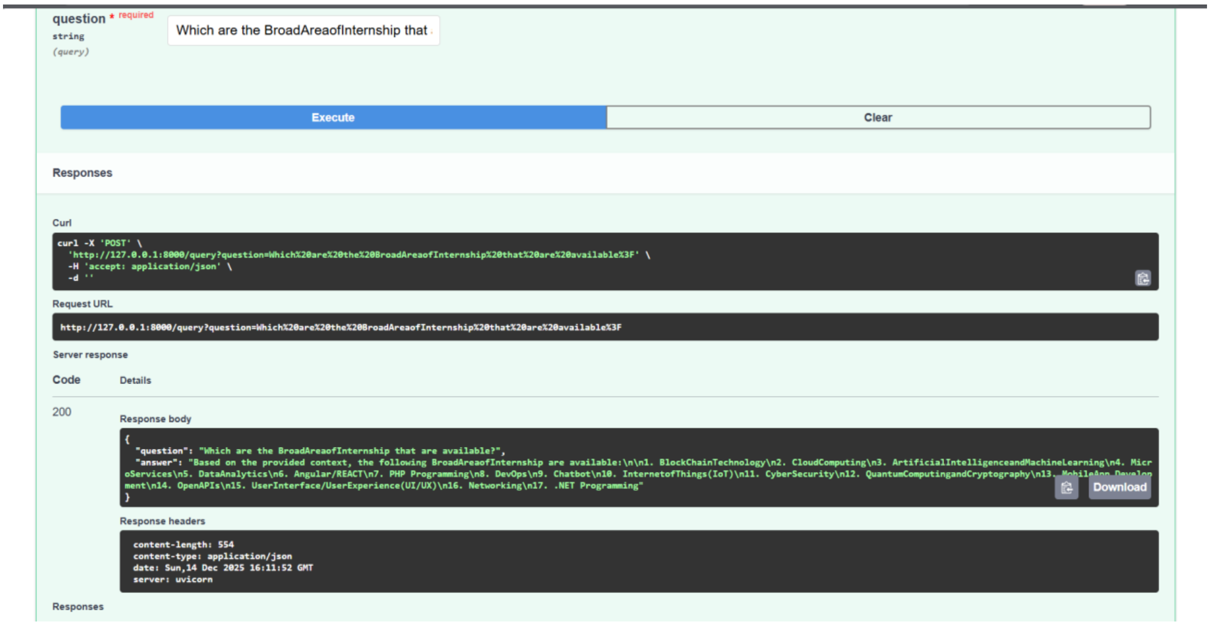
➔ Data Flow Summary

The system receives user queries through a FastAPI REST endpoint. The query is forwarded to a LangGraph-based agent, where a retrieval node performs semantic similarity search on document embeddings stored in a Weaviate vector database. The retrieved document chunks are then passed to a reasoning node, which invokes a locally running Ollama large language model to generate a response strictly grounded in the retrieved context. The final answer is returned to the user in structured JSON format via the API.

➔ Evaluation Observations

The system successfully retrieves relevant document segments and produces accurate, context-aware responses for factual queries related to the ingested documents. The agentic workflow enforces strict grounding using retrieved context, effectively reducing hallucinations. Local inference using Ollama demonstrated low latency and stable performance for lightweight models. Semantic retrieval via Weaviate was efficient for document-level search. The Dockerized deployment ensured reproducibility and simplified system evaluation.

Screenshots



question: Which are the BroadAreaofInternship that

Curl

```
curl -X 'POST' \
  'http://127.0.0.1:8000/query?question=Which%20are%20the%20BroadAreaofInternship%20that%20are%20available%3F' \
  -H 'accept: application/json' \
  -d ''
```

Request URL

```
http://127.0.0.1:8000/query?question=Which%20are%20the%20BroadAreaofInternship%20that%20are%20available%3F
```

Server response

Code Details

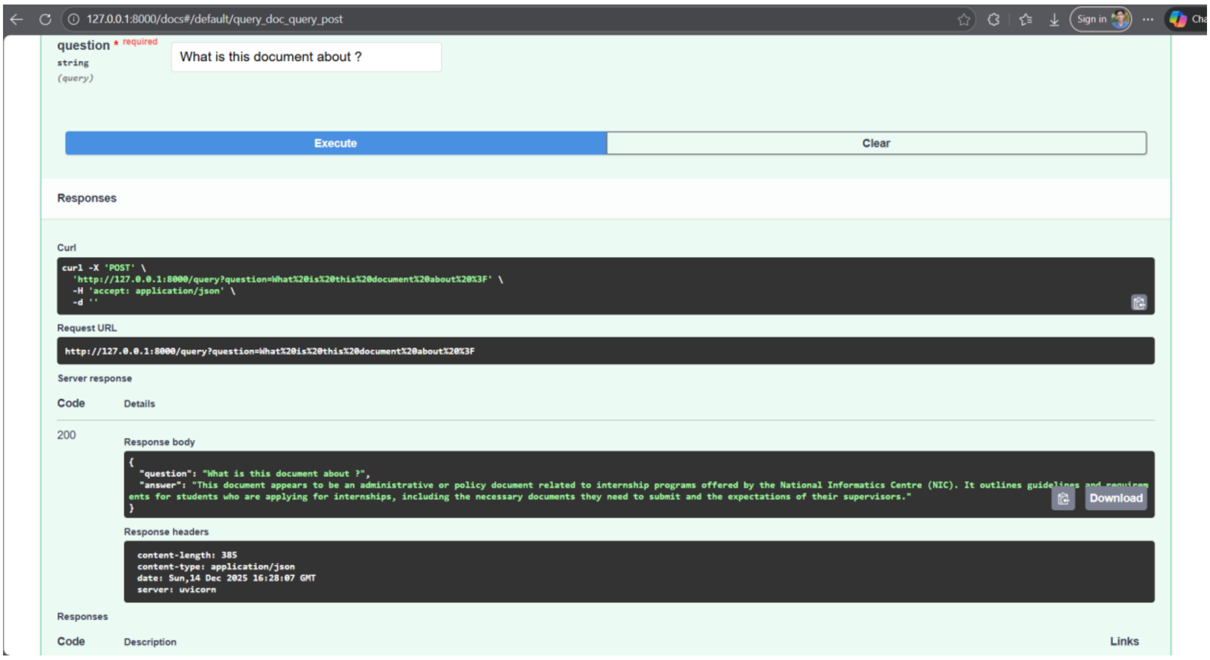
200

Response body

```
{
  "question": "Which are the BroadAreaofInternship that are available?",
  "answer": "Based on the provided context, the following BroadAreaofInternship are available:\n\n1. BlockChainTechnology\n2. CloudComputing\n3. ArtificialIntelligenceandMachineLearning\n4. Micr
oServices\n5. DataAnalytics\n6. Angular/REACT\n7. PHP Programming\n8. DevOps\n9. Chatbot\n10. InternetofThings(IoT)\n11. CyberSecurity\n12. QuantumComputingandCryptography\n13. MobileApp Develop
ment\n14. OpenAPIs\n15. UserInterface/UserExperience(UI/UX)\n16. Networking\n17. .NET Programming"
}
```

Response headers

```
content-length: 554
content-type: application/json
date: Sun,14 Dec 2025 16:11:52 GMT
server: uvicorn
```



question * required
string
(query)

What is this document about ?

Responses

Curl

```
curl -X 'POST' \
  'http://127.0.0.1:8000/query?question=What%20is%20this%20document%20about%20%3F' \
  -H 'accept: application/json' \
  -d ''
```

Request URL

```
http://127.0.0.1:8000/query?question=What%20is%20this%20document%20about%20%3F
```

Server response

Code Details

200

Response body

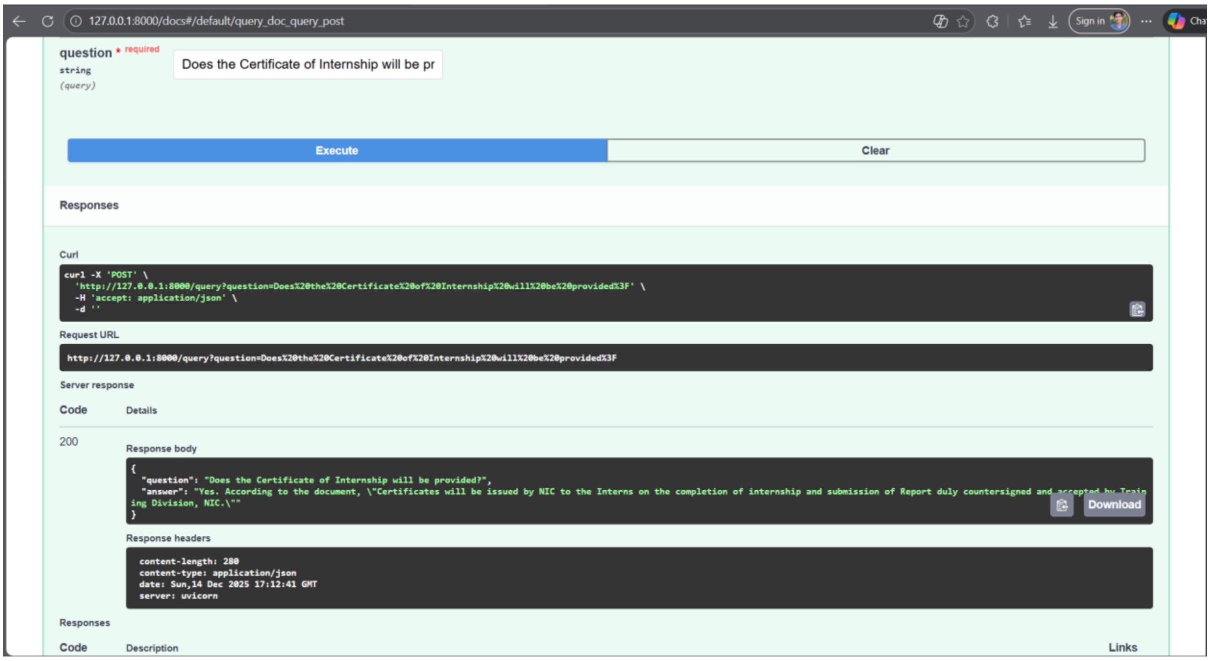
```
{
  "question": "What is this document about ?",
  "answer": "This document appears to be an administrative or policy document related to internship programs offered by the National Informatics Centre (NIC). It outlines guidelines and requirem
ents for students who are applying for internships, including the necessary documents they need to submit and the expectations of their supervisors."
}
```

Response headers

```
 content-length: 385
 content-type: application/json
 date: Sun,14 Dec 2025 16:28:07 GMT
 server: uvicorn
```

Responses

Code Description Links



question * required
string
(query)

Does the Certificate of Internship will be pr

Responses

Curl

```
curl -X 'POST' \
  'http://127.0.0.1:8000/query?question=Does%20the%20Certificate%20of%20Internship%20will%20be%20provided%3F' \
  -H 'accept: application/json' \
  -d ''
```

Request URL

```
http://127.0.0.1:8000/query?question=Does%20the%20Certificate%20of%20Internship%20will%20be%20provided%3F
```

Server response

Code Details

200

Response body

```
{
  "question": "Does the Certificate of Internship will be provided?",
  "answer": "Yes. According to the document, \"Certificates will be issued by NIC to the Interns on the completion of internship and submission of Report duly countersigned and accepted by Train
ing Division, NIC.\""
}
```

Response headers

```
 content-length: 280
 content-type: application/json
 date: Sun,14 Dec 2025 17:12:41 GMT
 server: uvicorn
```

Responses

Code Description Links

➔ Strengths

- Fully local inference (privacy-preserving)
- Clear separation of retrieval and reasoning
- Agentic design improves response grounding

- Production-ready API with FastAPI
- Scalable vector store architecture

➔ Limitations & Future Improvements

- Embedding generation depends on network availability during first load
- Evaluation performed on a single document corpus
- Can be extended with multi-document ingestion
- Can support streaming responses in future versions

➔ Conclusion

This project successfully demonstrates the design and implementation of an agentic document question answering system using modern AI frameworks. By combining semantic retrieval, agent-based reasoning, and local LLM inference, the system delivers accurate and privacy-preserving responses. The modular architecture and containerized deployment make the application extensible, reproducible, and suitable for real-world usage.