Course: INFO 7374 – Prompt Engineering & Generative AI
Student: Kaavya Loganathan
Institution: Northeastern University
Submission Type: Final Project Report

PaperPulse RAG

Prompt Engineering and Retrieval-Augmented Generation System

1. Introduction

Generative AI systems are increasingly used for question answering, summarization, and content generation. However, traditional large language models often suffer from hallucination, where responses may appear fluent but are not grounded in factual evidence. This poses a significant challenge for real-world applications, particularly in academic, research, and technical domains.

This project presents **PaperPulse RAG**, a generative AI system that combines **Prompt Engineering** with **Retrieval-Augmented Generation (RAG)** to produce grounded, citation-aware responses. Instead of relying solely on a model's internal knowledge, the system retrieves relevant document chunks from a knowledge base and constrains generation to this retrieved context.

The goal of this project is to demonstrate a practical, modular, and reliable generative AI pipeline that addresses real-world needs for trustworthy information synthesis.

2. Project Objectives

The primary objectives of this project are:

1. To design a generative AI system using **Prompt Engineering and RAG**
2. To ensure responses are grounded in retrieved documents
3. To reduce hallucinations through context restriction and citations
4. To provide measurable retrieval metrics
5. To deliver an interactive and user-friendly interface
6. To demonstrate ethical and responsible AI usage

3. System Architecture

The system follows a modular architecture separating retrieval, generation, and user interaction.

Architecture Flow (Textual Description)

1. The user interacts with a **Streamlit web interface**
2. Requests are sent to a **FastAPI backend**
3. Documents are ingested and split into overlapping chunks
4. Chunks are embedded and stored in a **FAISS vector database**
5. User queries trigger similarity-based retrieval
6. Retrieved chunks are passed into structured prompts
7. The system generates a grounded response with citations and metrics
8. The response is displayed back to the user

Key Components

- Streamlit UI (Frontend)
- FastAPI Backend
- Document Chunking Module
- FAISS Vector Store
- Prompt-Engineered Response Generator

4. Implementation Details

4.1 Prompt Engineering

Prompt engineering plays a critical role in controlling the behavior of the generative system. The prompts are designed to enforce strict grounding and prevent hallucination.

Key strategies include:

- Separate prompts for **Question Answering** and **Summarization**
- Explicit instructions to use only retrieved context
- Mandatory citation inclusion
- Safe fallback responses when insufficient evidence is available

Two user modes are supported:

- **Ask Mode:** Answers specific questions
- **Summarize Mode:** Generates structured summaries

4.2 Retrieval-Augmented Generation (RAG)

The RAG pipeline ensures responses are based on retrieved evidence rather than model memorization.

Knowledge Base

- Local text documents stored in data/kb_samples/

Chunking Strategy

- Recursive chunking with overlap to preserve semantic continuity

Vector Storage

- FAISS vector database for efficient similarity search
- Persistent storage in data/vectorstore/

Retrieval Mechanism

- Top-k similarity search
- Similarity threshold filtering to remove weak matches

4.3 Backend and API

The backend is implemented using **FastAPI**, exposing the following endpoints:

- /health – System health check
- /ingest – Document ingestion and vectorstore creation
- /query – Query execution with retrieval and generation

Pydantic schemas ensure input validation and structured responses.

4.4 User Interface

The user interface is built using **Streamlit** and provides:

- Knowledge base ingestion button
- Query input area
- Mode selection (Ask / Summarize)

- Display of answers, citations, and metrics

This interface enables non-technical users to interact with the system intuitively.

5. Performance Metrics

The system reports relevant metrics to evaluate retrieval behavior:

- **Retrieved Chunks:** Number of document chunks used per response

These metrics help analyze:

- Retrieval effectiveness
- Context sufficiency
- Transparency of the generation process

Example outputs are stored in:

examples/example_outputs/output1.json

examples/example_outputs/output2.json

6. Results and Discussion

The system successfully demonstrates:

- Grounded responses derived from retrieved documents
- Explicit citations linking answers to source documents
- Stable and consistent retrieval behavior
- Clear separation of retrieval and generation logic

The conservative generation strategy ensures correctness over fluency, which is appropriate for research-oriented applications.

7. Challenges and Solutions

Challenge 1: Avoiding Hallucination

Solution: Strict prompt constraints and context-restricted generation

Challenge 2: Small Knowledge Base

Solution: Similarity threshold filtering and fallback responses

Challenge 3: User Interpretability

Solution: Explicit citations and retrieval metrics

8. Ethical Considerations

Ethical AI principles were incorporated throughout the system:

- No external or personal data collection
- Use of only user-provided documents
- Explicit citation to prevent misleading outputs
- Clear communication of system limitations
- Responsibility placed on users to ensure lawful document usage

9. Future Improvements

Potential enhancements include:

- Integration of large language models for richer generation
- Advanced reranking methods to improve retrieval accuracy
- Support for additional document formats (PDF, HTML)
- Expanded evaluation metrics such as retrieval precision

10. Conclusion

PaperPulse RAG demonstrates a robust and responsible approach to generative AI using Prompt Engineering and Retrieval-Augmented Generation. By grounding responses in retrieved evidence and enforcing citation constraints, the system addresses key limitations of traditional generative models. This project highlights the importance of architectural design and ethical considerations in deploying real-world AI systems.