

Project Title: Document Summarization using Retrieval-Augmented Generation (RAG)

Objective:

To develop a summarization system that combines retrieval-based context selection with large language model (LLM) generation. The system should accept a long document and generate a concise, coherent summary using semantic chunking and RAG.

Project Tasks:

1. Document Ingestion

- Accept documents in PDF, TXT, or Markdown format.
- Split into semantically meaningful chunks using sliding windows or semantic segmenters.

2. Embedding & Retrieval

- Convert chunks to vector embeddings using SentenceTransformers or OpenAI API.
- Store in FAISS or Chroma vector DB.
- Perform semantic retrieval for a general summary query (e.g., "Summarize this document").

3. Summary Generation

- Use top-k retrieved chunks and pass them into a pre-trained LLM (e.g., GPT, LLaMA, Mistral).
- Generate a final summary that is coherent, fluent, and accurate.

4. Output Presentation

- Display the retrieved context and the generated summary.
- Optionally show token usage, latency, and similarity scores.

Dataset Suggestions:

Dataset	Description	Link
ArXiv Abstracts	Scientific article summaries	https://www.kaggle.com/datasets/Cornell-University/arxiv
CNN/DailyMail	News article + summary pairs	https://huggingface.co/datasets/cnn_dailymail

Custom PDFs Local/public documents Use your own documents or publicly available
for summarization datasets

■ Tutorials & Guides

- HuggingFace RAG Tutorial: <https://huggingface.co/blog/rag>
- LangChain Quickstart: https://docs.langchain.com/docs/get_started/introduction
- FAISS Vector Store Tutorial: <https://github.com/facebookresearch/faiss/wiki/Getting-started>

📄 Research Papers & Articles


- Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks (Lewis et al., 2020)
- Don't Pay Attention: Pay Enough! RAG revisited (Borgeaud et al., 2022)
- ColBERT: Efficient and Effective Passage Retrieval


🔧 Tools & Libraries

- LangChain: <https://github.com/hwchase17/langchain>
- FAISS: <https://github.com/facebookresearch/faiss>
- ChromaDB: <https://www.trychroma.com/>
- SentenceTransformers: <https://www.sbert.net/>

Note to Interns:

This is a simplified project focusing on the fundamentals of summarization using pre-trained models and vector search. You are encouraged to be creative and modular in design. We will evaluate your effort and understanding, not just the final output.

 **Submission:** ZIP file containing code, a PDF report, and sample results

 **Grading Rubric (100 Points):**

Category	Max Points	Evaluation Criteria
Document Parsing	15	Clean loading, chunking, and formatting
Embedding & Storage	15	Efficient use of vector DB and embeddings
Retrieval Quality	20	Relevance of selected content to the document's core idea
Summary Generation	20	Fluency, coverage, and accuracy of the summary
Pipeline Design	10	Clear, modular, and reproducible code
Output Presentation	10	Display of retrieved content and generated results
Documentation	10	ReadMe clarity, report explanation, and visual aids

 **Submission Requirements:**

- Python code with requirements.txt or environment.yml
- ReadMe with setup and usage guide
- Sample summarization runs for at least 3 different documents
- PDF report (2 pages max)