Retrieval-Augmented Generation (RAG) Project

# Final Project Report

Prepared by: Tahir Mustafa

Submission Date: 20th August 2025

# Tools & Libraries

* LLMs: Hugging Face Transformers (Flan-T5), Groq API, OpenAI API, Hugging Face Inference API
* Vector Database: FAISS
* Embeddings: Sentence-Transformers (all-MiniLM-L6-v2)
* Document Handling: pypdf (for PDFs), BeautifulSoup + requests (for scraping)
* UI: Gradio
* Reporting: Pillow, ReportLab, python-docx

# Step-by-Step Process

1. Data Loading: Loaded data from text, PDF, or scraped content.
2. Chunking: Split text into ~220-token chunks with overlap.
3. Embeddings: Generated embeddings using Sentence-Transformers.
4. Vector Database: Stored embeddings in FAISS for similarity search.
5. Retrieval: Queried Top-K most relevant chunks using cosine similarity.
6. Prompt Construction: Built augmented prompt with retrieved chunks.
7. Generation: Passed prompt to chosen LLM backend (Flan-T5, Groq, OpenAI, or HF API).
8. UI: Provided Gradio app for user queries.
9. Report: Generated project report with explanations and screenshots.

# Screenshots

## Database Creation & Insertion

A graph of blue bars

AI-generated content may be incorrect.

## Retrieval Results

A graph of blue bars

AI-generated content may be incorrect.

## Final Answer from LLM

## 