# Academic Policy Chatbot Using RAG & LLMs

## Project Title

Academic Policy Chatbot – Retrieval-Augmented Generation (RAG)

## Developed By

Muhammad Huzaifa Fahim
Department of Computer Science

## Objective

To create an intelligent chatbot that answers questions about the university's academic policy by leveraging Large Language Models (LLMs) and a Retrieval-Augmented Generation (RAG) pipeline.

## Tools & Technologies

| Tool/Library | Description |
|---|---|
| Python | Programming language |
| Streamlit | Web UI framework for deployment |
| Groq API + LLaMA 3 | Large Language Model provider (via Groq) |
| LangChain | RAG pipeline (retrieval + generation) |
| FAISS | Vector database for semantic similarity |
| PyPDF2 | To read and process PDF policy documents |
| Git & GitHub | Version control & cloud repository |
| Streamlit Cloud | Deployment platform |

## Features

- Chatbot UI built with Streamlit
- Upload and index academic policy PDF using vector embeddings

- Retrieve the most relevant chunks based on user queries
- Generate natural responses using LLaMA 3 (via Groq API)
- Deployed publicly on Streamlit Cloud
- Secure key management via Streamlit Secrets

## 🗁 Project Structure

```
academic_policy_chatbot/
├── app.py                # Main Streamlit app
├── academic_policy.pdf     # Source knowledge base
├── requirements.txt        # Dependencies
├── .streamlit/
│   └── secrets.toml        # API key (ignored on Streamlit Cloud)
├── README.md
```

## ⚙ How It Works

1. The policy PDF is split into chunks.
2. Each chunk is embedded using a text embedding model (e.g., HuggingFace or OpenAI).
3. FAISS creates an index from the embeddings.
4. When a user asks a question, the most relevant chunks are retrieved.
5. A prompt combining the user query + context chunks is sent to the Groq-hosted LLaMA 3 model.
6. The model responds with a concise answer based on retrieved context.

## 🚀 Deployment

1. Code pushed to GitHub:
   → https://github.com/huzaifaFahim/academic_policy_chatbot

2. App deployed on Streamlit Cloud:
   → https://academic-policy-chatbot.streamlit.app (or your app URL)

3. Secret Key (GROQ_API_KEY) securely added in Streamlit Secrets Manager

## API Key Management

While `.streamlit/secrets.toml` is part of local development, on Streamlit Cloud the API key is securely managed in the Secrets section under "Manage App".

## Sample Queries

- What is the minimum GPA required for graduation?
- How are course withdrawals handled?
- What is the university's plagiarism policy?

## Screenshots

[Insert 2–3 screenshots of the running app here]

## Conclusion

This chatbot demonstrates how cutting-edge AI can simplify information retrieval from institutional documents.
By combining vector search with LLMs via RAG, users get fast, accurate, and natural answers based on real policy documents.