Scheme Research Application – Project Report

Submitted By: Yata Veera Shiva Reddy  
BTech CSE Student

# Overview

At Haqdarshak, the research team manually extracts details from various government scheme portals. This project aims to automate that process by building a web application that:  
- Loads article content from URLs  
- Extracts structured summaries on key criteria  
- Enables users to ask questions and get accurate answers using an LLM  
  
This tool empowers both researchers and citizens with fast, relevant, and easily accessible information about government schemes.

# Objective

To build a Streamlit-based web application that:  
1. Accepts scheme article URLs.  
2. Extracts and summarizes based on:  
 - Scheme Benefits  
 - Application Process  
 - Eligibility  
 - Documents Required  
3. Creates embeddings using HuggingFace + FAISS for semantic retrieval.  
4. Uses a powerful LLM (LLaMA 3 via Groq) to answer user queries contextually.

# Tools & Technologies Used

|  |  |
| --- | --- |
| Tool | Purpose |
| Streamlit | Web interface |
| LangChain | LLM integration and document processing |
| FAISS | Fast vector similarity search |
| HuggingFace Embeddings | Local embedding generation |
| UnstructuredURLLoader | Article parsing |
| Groq + LLaMA3 | LLM-based Q&A using OpenAI-compatible API |
| Python | Programming language |
| Pickle | Store FAISS index locally |

# Architecture & Workflow

1. Input Handling:  
- User inputs one or more scheme article URLs.  
- Optionally, URLs can be uploaded via a text file.  
  
2. Content Extraction:  
- URLs are processed using UnstructuredURLLoader from LangChain Community.  
- The article text is split into manageable chunks.  
  
3. Vector Embeddings:  
- Embeddings are generated using HuggingFace’s all-MiniLM-L6-v2.  
- FAISS is used to store and index these embeddings.  
  
4. Storage:  
- The FAISS index is stored in a file (faiss\_store\_openai.pkl) for persistent use.  
  
5. User Q&A:  
- Users ask questions via the app.  
- The app retrieves relevant document chunks using similarity search.  
- LLaMA3 (via Groq API) is queried to answer using the retrieved context.  
- Source URLs and a summary are provided with each answer.

# User Interface

Section | Functionality  
--------|--------------  
Sidebar | Enter URLs or upload URL file and start processing  
Main View | Ask questions and get answers + relevant sources  
Expandable Sections | (Optional) Show extracted summaries per article

# Files Submitted

File Name | Description  
----------|------------  
main.py | Main Streamlit web app  
requirements.txt | Python dependencies  
faiss\_store\_openai.pkl | Saved FAISS index file  
.config | API key configuration file (not shared for security)  
demo.mp4 | Screen recording showing app functionality  
project\_report.pdf | This project report

# Testing

- Tested on multiple scheme URLs.  
- Verified summary generation and Q&A outputs.  
- Checked edge cases like empty URLs and invalid articles.

# Limitations

- Some websites may block bot scraping.  
- If the article content is poorly formatted, summaries may be inaccurate.  
- FAISS index is rebuilt every session unless reloaded.

# Future Enhancements

- Add PDF and text document upload support.  
- Automatic classification of scheme categories (e.g., health, education).  
- Multi-language support (e.g., Hindi/Marathi).  
- Add user authentication and save query history.

# Demo Video

A demonstration of the app in action is included as demo.mp4, showcasing:  
- URL input and processing  
- Summary extraction  
- Interactive Q&A  
- Source retrieval and responses

# Conclusion

This project demonstrates how modern NLP tools can significantly reduce manual effort in processing government scheme data. The Scheme Research Application is a powerful prototype to aid researchers, NGOs, and end users in navigating complex policy documents more effectively.