Website Q&A; Chatbot (RAG-Based)

Project Overview

This project is a Retrieval-Augmented Generation (RAG) based Website Question Answering Chatbot built using Streamlit, LangChain, Groq LLM, FAISS, and HuggingFace embeddings. Users can enter a website URL, index its content, and ask questions strictly based on that website. If the answer is not found, the chatbot replies: "The answer is not available on the provided website."

Architecture

1. User inputs website URL
2. Website content loaded via UnstructuredURLLoader
3. HTML cleaned using BeautifulSoup
4. Text split into chunks
5. HuggingFace embeddings generated
6. Stored in FAISS vector database
7. User questions retrieve relevant chunks
8. Groq LLM generates answers from retrieved context only

Frameworks Used

Streamlit (UI)
LangChain (RAG orchestration)
Groq (LLM inference)
FAISS (Vector DB)
HuggingFace (Embeddings)
BeautifulSoup + Unstructured (Web scraping)

LLM Model

Default: llama-3.1-8b-instant
Alternative: qwen/qwen3-32b
Chosen for fast inference, strong reasoning, and free-tier friendliness via Groq.

Vector Database

FAISS is used because it is lightweight, fast, runs locally, and requires no external services.

Embedding Strategy

Model: sentence-transformers/all-MiniLM-L6-v2
Chosen for speed, quality semantic embeddings, and production readiness.

Setup Instructions

1. Create virtual environment
2. Install requirements.txt
3. Create .env file with GROQ_API_KEY
4. Run: streamlit run app.py

Assumptions

- Website is publicly accessible
- HTML content only
- Single URL indexing
- Session-based memory

Limitations

- No recursive crawling
- No PDF ingestion
- Local FAISS only
- Basic HTML cleaning

Future Improvements

- Multi-page crawling

- PDF ingestion
- Streaming responses
- Cloud vector databases
- Docker deployment
- UI enhancements

Author
Built as an AI/ML engineering assignment demonstrating practical RAG implementation.