Sanskrit Question Answering System

Retrieval-Augmented Generation (RAG) Based NLP Application

---

1. Introduction

Sanskrit is an ancient and highly structured language with rich literary and philosophical content. However, accessing precise information from Sanskrit documents is challenging due to the lack of search-friendly tools and modern NLP support.

This project presents a **Sanskrit Question Answering System** using **Retrieval-Augmented Generation (RAG)**, which enables users to ask questions in Sanskrit and receive accurate, context-aware answers from Sanskrit texts.

---

2. Objective

- Enable question answering directly in Sanskrit language
- Use RAG to combine document retrieval with generative AI
- Build an easy-to-use web interface using Streamlit
- Demonstrate real-world application of LLMs with domain-specific data

---

3. System Architecture

> **User Question (Sanskrit)** → Text Embedding → Vector Search (FAISS) → Context Retrieval → Large Language Model (Groq LLM) → Final Answer

---

4. Technologies Used

Component	Technology
Programming Language	Python
Frontend	Streamlit
LLM Provider	Groq (LLaMA-based models)
Embeddings	HuggingFace Sentence Transformers
Vector Database	FAISS
Framework	LangChain

---

5. Data Processing

The Sanskrit source document was originally in PDF format and converted into UTF-8 encoded text. The text is split into smaller overlapping chunks using **RecursiveCharacterTextSplitter** to preserve semantic meaning.

Each chunk is converted into vector embeddings and stored in FAISS for efficient similarity search.

---

6. Retrieval-Augmented Generation (RAG)

RAG improves answer accuracy by retrieving the most relevant document chunks before generating a response. Instead of relying only on the LLM's internal knowledge, the system grounds answers in the provided Sanskrit text.

- Prevents hallucinations
- Improves factual correctness
- Supports domain-specific question answering

---

7. Application Workflow

1. User enters a question in Sanskrit
2. Question is embedded and matched against stored vectors
3. Relevant Sanskrit text chunks are retrieved
4. LLM generates an answer using retrieved context
5. Answer is displayed in the Streamlit UI

---

8. Project Structure

```
RAG_Sanskrit_PratikGaigole/
│
├── code/
│   └── app.py
│
├── data/
│   └── sanskrit.txt
│
├── report/
│   └── Sanskrit_RAG_Report.pdf
│
├── requirements.txt
├── README.md
└── .gitignore
```

---

9. Limitations

- Limited size of Sanskrit dataset
- Dependent on external LLM API availability
- Complex Sanskrit grammar may require further fine-tuning

---

10. Future Enhancements

- Support for multiple Sanskrit documents
- Advanced Sanskrit grammar-aware embeddings
- Multilingual translation support
- Improved UI with answer citations

---

11. Conclusion

This project demonstrates the effective use of Retrieval-Augmented Generation for low-resource languages like Sanskrit. By combining vector search and large language models, the system provides accurate and context-aware answers, showcasing the practical application of modern NLP techniques.

Submitted by: Pratik Gaigole
Project: Sanskrit RAG Question Answering System