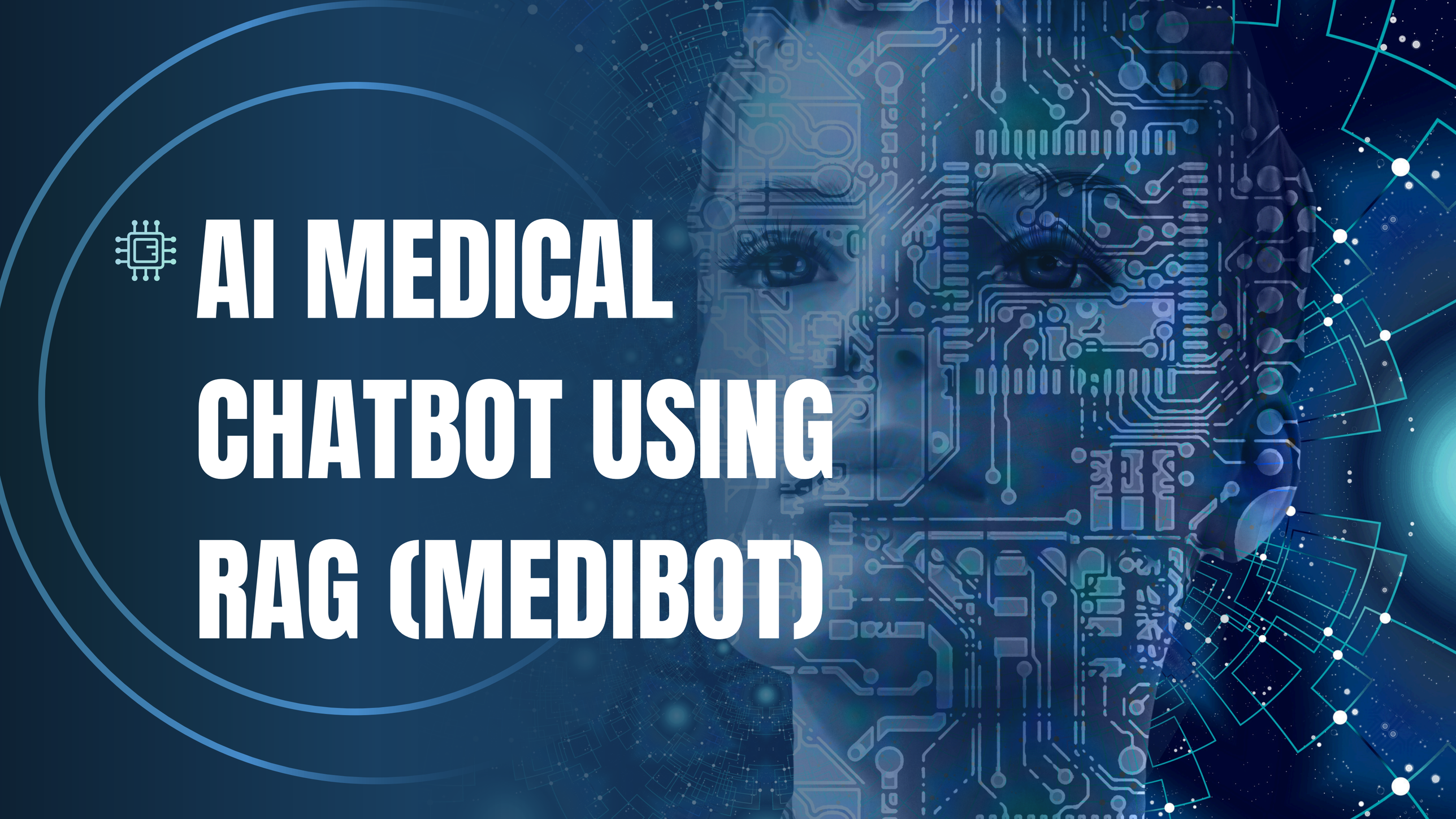
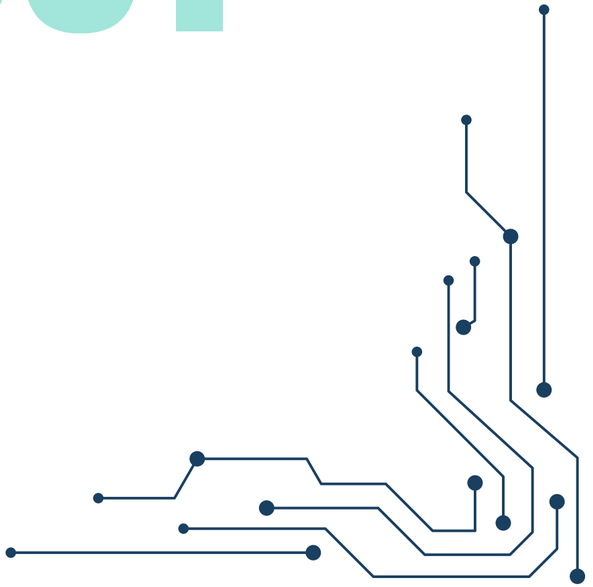

AI MEDICAL
CHATBOT USING
RAG (MEDIBOT)

PROJECT LAYOUT



Phase 1

SETUP MEMORY FOR LLM (VECTOR DATABASE)

- Load raw PDF(s)
- Create Chunks
- Create Vector Embeddings
- Store embeddings in FAISS

Phase 2

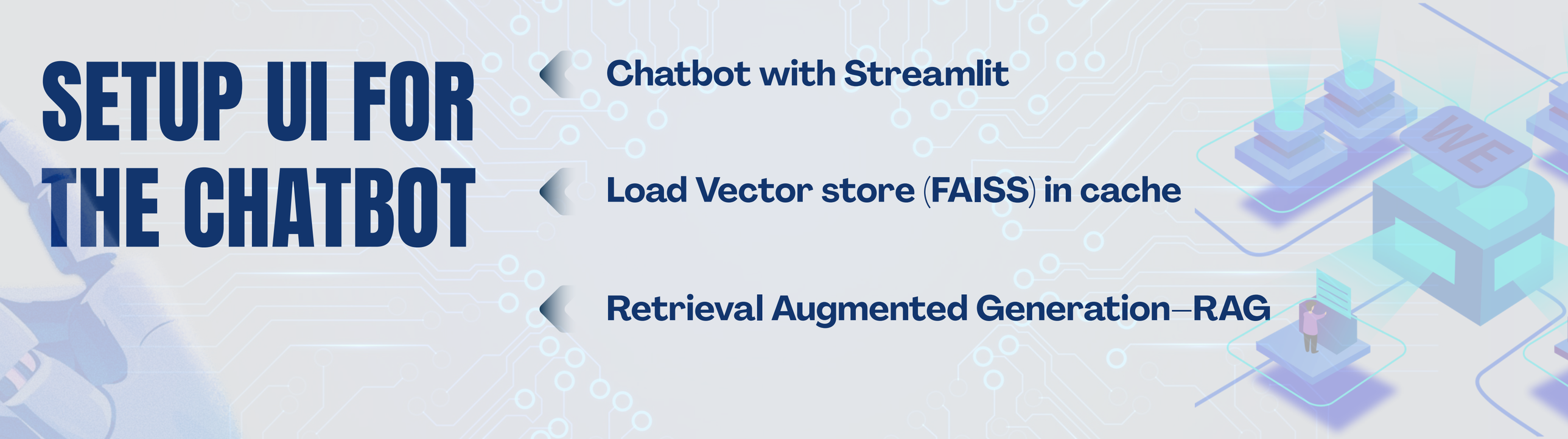
CONNECT MEMORY WITH LLM

- Setup LLM (Mistral with HuggingFace)
- Connect LLM with FAISS
- Create chain

Phase 3

SETUP UI FOR THE CHATBOT

- Chatbot with Streamlit
- Load Vector store (FAISS) in cache
- Retrieval Augmented Generation–RAG



Tools & Technologies

- Langchain (AI Framework for LLM applications)
- FAISS (Vector Database)
- HuggingFace (ML/AI Hub)
- Steamlit (For Chatbot UI)
- Mistral (LLM Model)
- Python (Programming Language)
- HuggingFace (ML/AI Hub)
- Steamlit (For Chatbot UI)

Technical Details

Document Processing

- *PDFsloaded using LangChain document loaders.*
- *Text split into overlapping chunks.*
- *Chunking improves semantic retrieval*

Embeddings

- *HuggingFace embedding model used.*
- *Each chunk converted into dense vector representation.*

Vector Database

- *FAISS (Facebook AI Similarity Search) used.*
- *Efficient similarity search for large embedding collections.*

Technical Details

Language Model

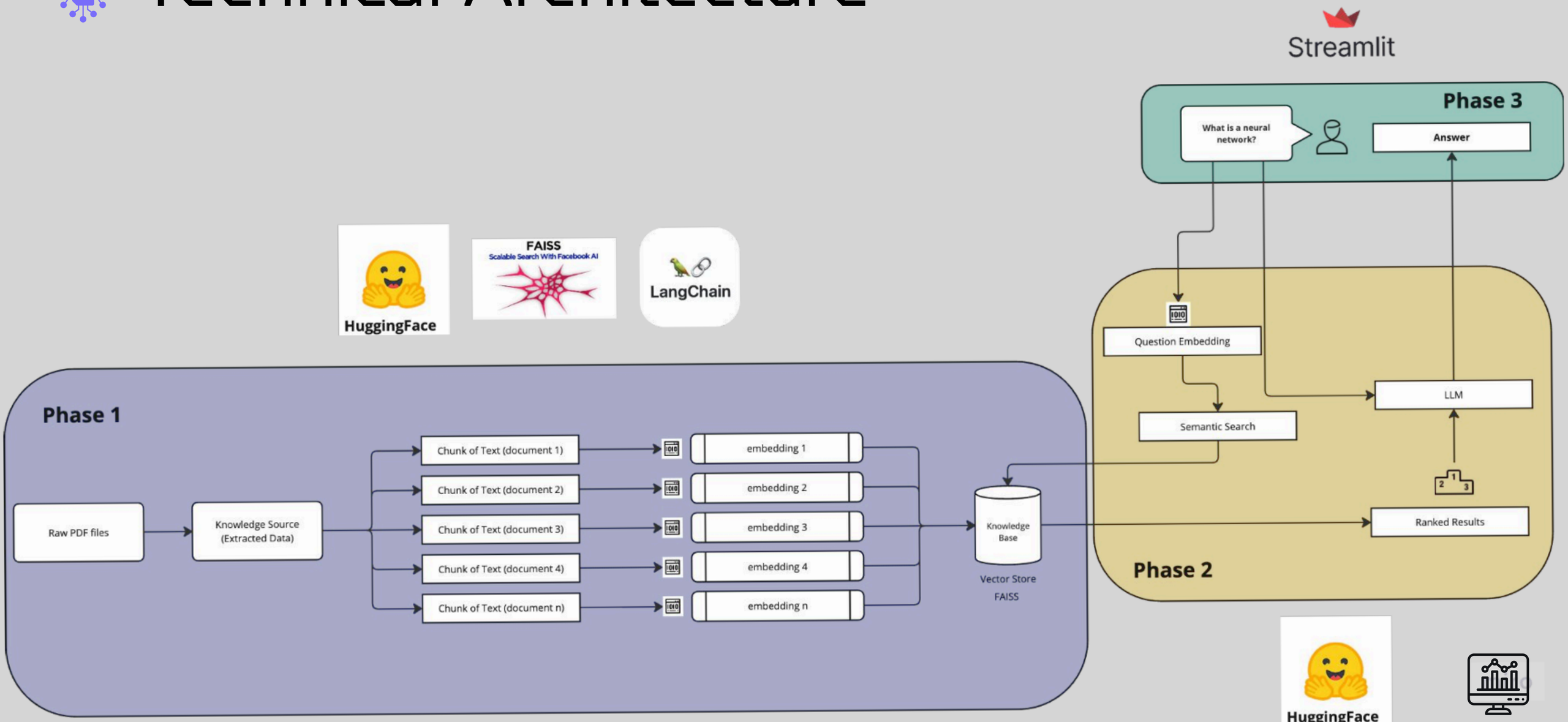
- *Mistral model accessed via HuggingFace.*
- *Integrated using LangChain framework.*

RAG Mechanism

Given a query q:

- *Compute embedding E(q).*
- *Retrieve top-k similar document chunks.*
- *Concatenate retrieved context.*
- *Pass context + query to LLM.*
- *Generate grounded response.*

Technical Architecture



DISCUSSION

THE INTEGRATION OF FAISS WITH LANGCHAIN SIGNIFICANTLY IMPROVED CONTEXTUAL ACCURACY. THE MODULAR 3-PHASE ARCHITECTUREALLOWED CLEAN SEPARATION OF DATA INGESTION, MODEL INFERENCE, AND UI DESIGN.

Trade-offs Observed:

- Large chunk sizes improve context but increase token usage
- Smaller chunks improve retrieval precision but may fragment information.
- Response latency depends on embedding and retrieval time.

Limitations:

- No authentication system in UI.
- Limited to single or predefined documents.
- Evaluation primarily qualitative.
- No automated benchmarking metrics used.

CONCLUSION AND FUTUREWORK

This project successfully implemented an AI Medical Chatbot using a Retrieval-Augmented Generation framework. The system in tegrates document embeddings, vector similarity search, and LLM based response generation into a unified pipeline.

Future Work:

- Adduser authentication in UI
- Enable multiple document uploads
- Implement multi-document embedding
- Addunit testing for RAG pipeline
- Quantitative benchmarking (precision@k, recall@k)
- Deploy on cloud infrastructure

Thank You!