AI Powered Incident Root Cause Analysis (RCA) - Architecture

This document explains the architecture, flow, and artifacts of the AI Powered Incident Root Cause Analysis (RCA) system built using Python, Streamlit, FAISS, Sentence Transformers, and OpenAI. The system helps banking production support teams identify probable root causes from logs and incident documents.

# 1. High-Level Architecture

The system follows a Retrieval-Augmented Generation (RAG) pipeline. It ingests logs and documents, extracts text, embeds them using Sentence Transformers, indexes them with FAISS, and leverages an LLM (OpenAI or local) to answer questions and generate summaries.

## Flowchart

User Uploads Logs/Docs → Text Extraction → Chunking → Embedding (Sentence Transformers) → Vector Store (FAISS) → Retrieval (Top-K Chunks) → LLM (OpenAI/Local) → Answers & RCA Summary

# 2. System Components

## 2.1 Streamlit Frontend

- Provides UI for uploading logs/documents.  
- Allows pasting raw logs.  
- Options to ingest and index data.  
- Query interface for asking RCA questions.  
- Displays answers, evidence, and summaries.

## 2.2 Ingestion Layer

- Handles file parsing for TXT, PDF, DOCX.  
- Extracts plain text from uploaded documents.  
- Splits large documents into manageable chunks.  
- Ensures overlap for semantic continuity.

## 2.3 Embedding & Vector Store

- Uses Sentence Transformers (all-MiniLM-L6-v2) to embed text.  
- Normalizes embeddings for cosine similarity.  
- Stores vectors in FAISS index.  
- Maintains metadata for retrieved chunks.

## 2.4 Retrieval + AI Agent

- Retrieves top-k relevant chunks from FAISS.  
- Builds a context prompt with evidence.  
- Sends query to LLM (OpenAI GPT or fallback GPT-2).  
- Generates RCA answers with reasoning and confidence.

## 2.5 Incident Summary Generator

- Aggregates context from multiple chunks.  
- Asks LLM to produce structured JSON output:  
 • Summary  
 • Likely Root Causes  
 • Recommended Next Steps

# 3. Key Artifacts

- \*\*app.py\*\*: Main Streamlit UI  
- \*\*rca/ingest.py\*\*: Text extraction & chunking  
- \*\*rca/vectorstore.py\*\*: FAISS index management  
- \*\*rca/agent.py\*\*: LLM-based RCA agent  
- \*\*sample\_data/logs.txt\*\*: Example input logs  
- \*\*requirements.txt\*\*: Dependencies list  
- \*\*README.md\*\*: Usage instructions

# 4. Example Usage Flow

1. User uploads log files (e.g., database errors).  
2. System extracts and embeds text, storing in FAISS.  
3. User queries: 'What is the root cause?'  
4. Top-K chunks are retrieved (showing DB connection failures).  
5. LLM analyzes context and responds with: 'High confidence root cause is DB connectivity outage'.  
6. System generates a structured summary with root cause and next steps.