RAG System Architecture Report

Executive Summary

This document outlines a Retrieval-Augmented Generation (RAG) system that enables intelligent document querying through semantic search and response generation. The system uses a two-stage retrieval approach with vector similarity search and cross-encoder re-ranking for accurate, contextually relevant responses.

System Architecture

Core Components

Vector Database: ChromaDB (`./chroma_db`) - Stores document embeddings for fast similarity search

Embedding Model: mxbai-embed-large (Ollama) - Converts text to vector representations

Language Model: Llama 3.2 3B (Ollama) - Generates responses from retrieved context

Re-ranking: CrossEncoder - Improves relevance scoring of retrieved documents

Process Flow

1. Initialization

System loads ChromaDB, embedding model, and LLM, preparing all components for query processing.

2. Query Processing

- User submits query
- System performs semantic search for top-5 relevant document chunks
- If no results found: Returns "no documents found"
- If results found: Proceeds to re-ranking

3. Re-ranking and Selection

CrossEncoder re-ranks retrieved chunks and selects top-2 for optimal context balance.

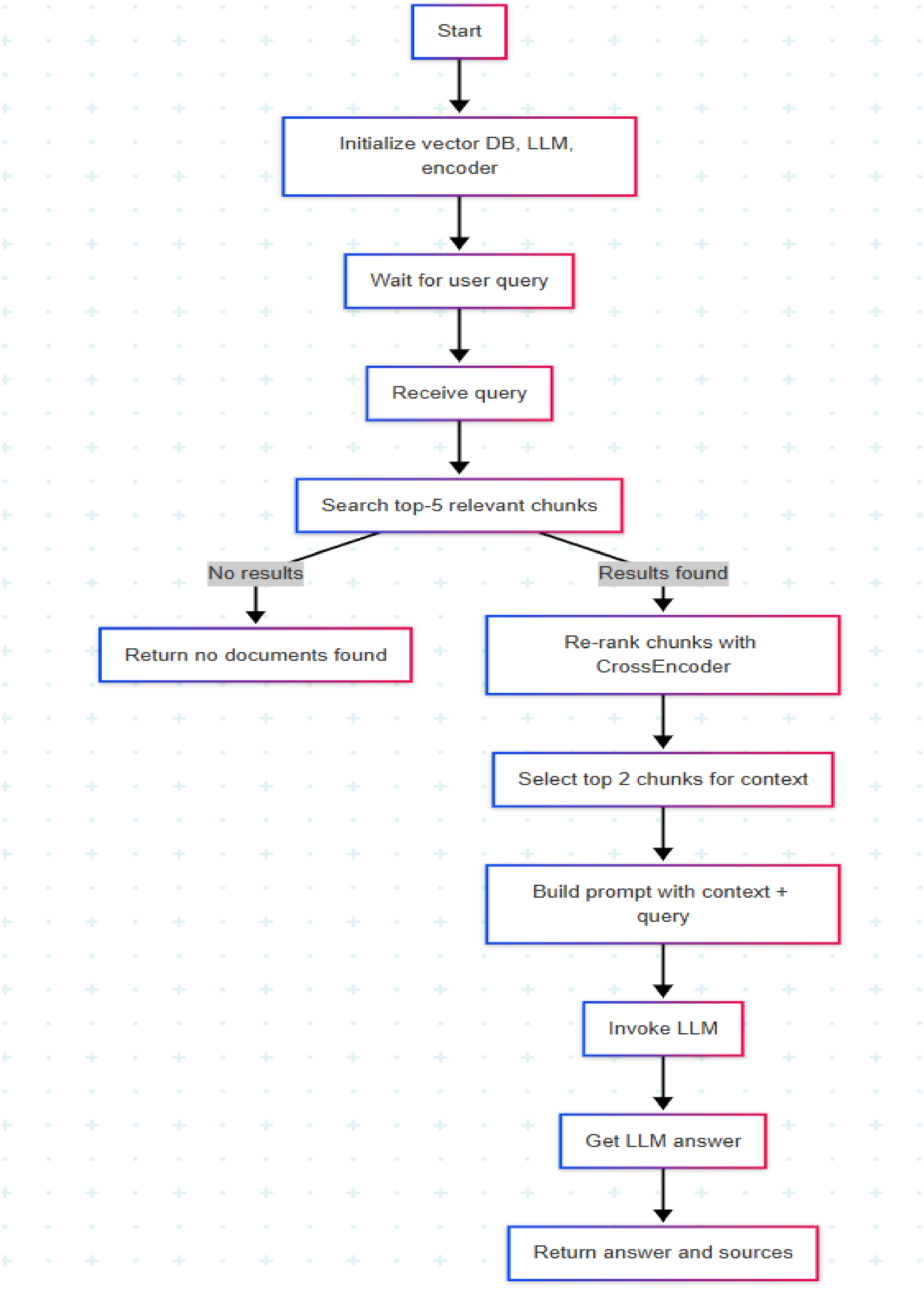
4. Response Generation

Selected chunks are combined with the user query to build an enhanced prompt. Llama 3.2 processes this prompt to generate the final response.

5. Output

System returns generated answer with source references for transparency.

Flow Chart



Technical Specifications

Component	Technology	Configuration
Vector DB	ChromaDB	Local storage
Embeddings	mxbai-embed-large	Via Ollama
LLM	Llama 3.2	3B parameters
Retrieval	Two-stage	Top-5 → Top-2

Key Benefits

Enhanced Accuracy: Two-stage retrieval with re-ranking improves response relevance

Local Deployment: On-premises processing ensures data privacy and eliminates API dependencies

Source Attribution: Provides transparent sourcing for generated responses

Cost Effective: No ongoing cloud API costs with local model deployment

Implementation Requirements

Hardware: GPU-recommended for optimal model performance

Storage: Adequate disk space for vector database and model files

Memory: Sufficient RAM for concurrent model operations

Conclusion

This RAG architecture delivers efficient, accurate document querying through modern vector search and language generation technologies. The two-stage retrieval design balances performance with accuracy, while local deployment ensures privacy and operational independence.