Task Title:

Develop CPU-Based RAG LLM Inference Pipeline

Objective:

Design and implement a Retrieval-Augmented Generation (RAG) pipeline optimized for inference on CPU-only environments. The solution should enable efficient document retrieval and response generation using a lightweight LLM and embedding model.

Responsibilities:

1. **Design the Architecture:**
    - Define a modular pipeline integrating document retrieval and generative response components.
    - Ensure all components are optimized for CPU usage with minimal memory and latency footprint.
2. **Embed and Store Documents:**
    - Use a CPU-compatible embedding model (e.g., `sentence-transformers`, `MiniLM`,llamafile etc.).
    - Store embeddings in a vector database (e.g., FAISS with CPU backend, ChromaDB).
3. **Implement Retrieval:**
    - Perform efficient similarity search using CPU-based vector index.
    - Implement ranking logic if necessary to refine the top-k results.
4. **Integrate LLM Inference:**
    - Use a lightweight CPU-optimized LLM (`llama.cpp`, `ggml`,llamafile,vllm).
    - Implement prompt construction by combining query and retrieved documents.
5. **Pipeline Integration:**
    - Build an inference API or script that accepts a user query, retrieves relevant documents, and returns a generated response.

    - Ensure the pipeline runs within acceptable latency bounds on CPU.

6. **Testing and Benchmarking:**
    - Evaluate response accuracy, retrieval quality, and latency.
    - Compare performance across different CPU models if needed.
7. **Documentation:**
    - Provide clear usage instructions and installation requirements.
    - Document the pipeline structure, model choices, and optimization techniques.

Deliverables:

- End-to-end CPU-based RAG inference pipeline (codebase).
- Sample script/API or streamlit for interactive use.
- Deployment guide and benchmark report.
- Documentation for extending or modifying the pipeline.