RAG System Requirements for Threat Analysis Database

1. System Purpose

Build a retrieval-augmented generation system that systematically analyzes and catalogs threats made during international crises, following a structured codebook for consistent threat documentation.

2. Core Requirements

A. Data Processing Requirements

- Input: Mixed source documents with crisis identifiers. I have already assigned each document to the relevant international crisis.
- Output: Structured threat analysis entries
- Must maintain source traceability
- Must support multiple document types (news, government docs, etc.) I can also categorize that in the pre-processing to include a score for source reputation. For example, Government documents are more trustworthy as a source than blog entries.

B. Query Processing Requirements

1. **Query Expansion System**
    - Transform user queries into multiple structured sub-queries
    - Support parallel query processing
    - Enable comprehensive threat assessment
2. **Information Retrieval**
    - Multi-stage retrieval process
    - Support for metadata filtering by crisis ID

- Cross-document information synthesis
- Source verification and weighting

C. Advanced Retrieval System

- Multi-stage retrieval pipeline with coarse-to-fine approach
- Late chunking implementation for preserving document context (Use "jina-embeddings-v3"). See this page.
- Support for handling large academic documents (~1GB dataset is the total size of my text data)
- Support for multiple retrieval rounds based on information gaps.
  - Multi-stage retrieval process
  - Coarse-grained initial search
  - Fine-grained focused retrieval
  - Support for iterative search refinement
- Custom metadata extraction and indexing for:
  - Dates
  - Locations
  - Key figures
  - Unique identifiers
  - Event classifications
  - I currently have crisis identifier. So we already know which crisis is related to which document.
- Enable metadata-based filtering. For example, I currently have crisis identifier in the text data. So we already know which crisis is related to which document.
- I am open to your other ideas here.

C. Response Generation Requirements

1. **Structured Output**

- Include all required threat characteristics that is asked in the prompt.
- Maintain consistent scoring scales
- Provide source citations. So responses should be traceable to chunk identifier, which are traceable to document identifier.

2. **Quality Control**
   - Generate trustworthiness scores (Use "cleanlab_tlm"). See this page.
   - Validate against criteria given in the code book (the economic costs of the crisis was high, somewhat high, low etc and I would give these criteria for classification) and generate a confidence score (low, medium, high)
   - If the trustworthiness score or confidence score is or trustworthiness score is less than 0.8, it should expand queries until these confidence scores get higher.

5. Quality Requirements

A. Response Quality

- Trustworthiness score > 0.8
- Must include evidence for claims. It should be traceable to chunk identifier, hence, to the original document.
- Must maintain scoring consistency

B. System Performance

- Handle 1GB+ dataset
- Support multiple parallel queries
- Maintain response time < 30 seconds
- Scale with growing document base.

6. Integration Points

A. Input Integration

- Document processing pipeline
- Crisis ID management
- Source tracking system
- Metadata management

B. Output Integration

- Structured threat database
- Quality assessment metrics
- Source citation system
- Trustworthiness scoring

7. Success Criteria

1. Accurate threat identification and classification
2. Consistent codebook adherence
3. Reliable source attribution
4. High trustworthiness scores
5. Comprehensive crisis coverage
6. Efficient information retrieval
7. Structured response generation

3. Technical Specifications

A. Required Tools

```
{
    "embedding": "jina-embeddings-v3",
    "vector_store": "pinecone",
    "orchestration": "langchain",
    "verification": "cleanlab_tlm"
```

```
    "LLM": "OpenAI", "Anthropic"
}
```

B. Processing Pipeline

```
flowchart TD
    I[Input Documents] --> P[Processing]
    P --> V[Vector Storage]
    Q[Query] --> E[Expansion]
    E --> R[Retrieval]
    R --> G[Generation]
    G --> V2[Verification]
```