Project: Document Intelligence Assistant

Time Limit: 7 Days

Core Assignment

Document Processing Pipeline

- Build a system that can ingest various document types (PDF, TXT, DOCX)
- Extract and chunk text content appropriately
- Generate embeddings using a suitable model
- Store documents in a vector database (Pinecone, Chroma, or local solution)

RAG Implementation

- Implement semantic search over document collection
- Create a question-answering system using retrieved context
- Use any free LLM apis (Gemini, Groq, OpenRouter, etc.)
- Implement proper prompt engineering techniques

Web Interface

- Simple upload interface for documents
- Chat-like Q&A interface
- Display relevant document chunks with answers
- Show confidence scores or source citations

Technical Requirements

- Python-based backend (FastAPI)
- Proper error handling and logging
- Environment variable management
- Input validation and sanitization

Advanced Features (Choose any 3 (including MANDATORY))

- API Integration: Support multiple LLM providers with fallback - **MANDATORY**
- Multi-modal Support: Handle images in documents with vision models
- External tool Integration: Add a web search tool for grounding the responses
- Conversation Memory: Maintain context across multiple questions
- Document Summarization: Generate summaries of uploaded documents
- Query Expansion: Improve search with query rewriting
- Evaluation Metrics: Implement RAG evaluation (faithfulness, relevance)
- Streaming Responses: Real-time response streaming

Evaluation Criteria

- System Design (25%): Architecture, component interaction, scalability considerations
- AI Implementation (25%): RAG quality, prompt engineering, model selection
- Code Quality (20%): Clean, modular, well-documented code
- Functionality (15%): Core features working reliably
- Innovation (10%): Creative problem-solving, additional features
- Documentation (5%): Setup instructions, system explanation

Deliverables

1. GitHub repository with comprehensive README
2. System architecture diagram
3. Demo video showing document upload and Q&A (5-7 minutes)
    - Showcase at least 2 different file types and 1 image
    - Showcase the chosen advanced features
    - Choice of embedding model and why
    - Chunking strategy used
    - Prompt engineering approaches
    - Challenges faced and solutions
4. Brief technical write-up explaining:
    - Choice of embedding model and why
    - Chunking strategy used
    - Prompt engineering approaches
    - Challenges faced and solutions