Multimodal RAG System with Generative AI

Upload a File (PDF or Image)

Upload a PDF or an image file (e.g., PNG, JPG)

Drag and drop file here

Limit 200MB per file • PDF, PNG, JPG, JPEG

Browse files

task2.pdf 19.0KB

Ask a Question

Enter your question:

which model they talk in this file

PDF uploaded successfully: task2.pdf

Text extracted from the PDF.

Document processed successfully with 3 chunks.

Generated Response:

The file mentions the following language models (LLMs):

- **OpenAI GPT**
- **Claude**
- **Mistral**
- **LLaMA**

Relevant Contexts:

[
0 :
" language model (OpenAI GPT,
Claude, etc.) to generate answers.
3. Create an Agentic Flow (Plan → Act → Observe → Reflect)
• Simulate goal-driven behavior (e.g., “help me revise for Topic 2” or “guide me through
key concepts before the exam”).
• Use frameworks like LangChain, Agno (Phidata), or a custom implementation in Python.
4. (Bonus) Multimodal Reasoning
• Optionally include image files or video transcripts.
• Show how the model can reference or reason over non-textual inputs (e.g., diagrams,
slide content, etc.).
Recommended Tech Stack
• Backend: Python, FastAPI
• AI Frameworks: LangChain, Phidata (Agno), or custom
• Vector Database: FAISS, pgvector
• LLMs: OpenAI GPT, Claude, or open-source models (Mistral, LLaMA, etc.)
• Frontend (Optional): Streamlit for a simple chat interface
Deliverables
Please submit either a GitHub repository or a zipped folder containing:
• Source code
• README.md with setup and usage instructions
• Sample course data used
• API documentation
(Optional:"

```
1 :
"Assignment: Agentic AI Integration for LMS
Objective
Demonstrate your ability to design and implement agentic AI capabilities within
a Learning
Management System (LMS) context. This task focuses on developing a course-
specific chat
interaction system using Retrieval-Augmented Generation (RAG)—a core feature of
modern smart LMS platforms.
Assignment Brief
You are tasked with building a prototype system that enables users to ask
questions related to
specific course content (PDFs, articles, or video transcripts). The system
should return
context-aware responses by retrieving relevant course data using RAG and
generating
answers using a language model.
Goals
1. Ingest Sample Course Data
• Use at least 2-3 sample course materials (e.g., PDFs, web content, or video
transcripts).
• Convert content to embeddings and store in a vector database like FAISS or
pgvector.
2. Build a RAG-based Chat API
• Use FastAPI to create a RESTful endpoint.
• On query, retrieve relevant chunks and pass them to a"
2 : " Include a short Loom/YouTube walkthrough, ~2-3 minutes)
   Evaluation Criteria
   • Understanding of RAG and agentic workflows
   • Code quality, clarity, and modularity
   • Appropriate use of AI and vector store tools
   • Ability to provide accurate course-specific responses
   • Bonus: Multimodal reasoning, analytics, smart response patterns"
]
```