## **Multimodal Assistant**

### **1. Problem Statement**

Users today are presented with both text and image content in the information age. Yet, the majority of AI assistants can only handle one modality—one that is either text or image.

**Challenge**: Create a single assistant that can answer text questions as well as interpret and analyze content from uploaded images (e.g., news headlines, posters, screenshots).

### **2. Proposed Solution**

We are creating a Multimodal Assistant which:

* Accepts text input queries and returns a response via an LLM (LLaMA2).
* Accepts file uploads, applies OCR (Optical Character Recognition), and returns a summarized response via the same LLM.
* Offers a straightforward frontend UI via Streamlit.
* Utilizes Ollama (local LLM) for offline, private model inference.
* Operates without depending on external APIs such as OpenAI.

### **3. Architecture Diagram**

+--------------------+ HTTP POST +--------------------------+

| | -------------------> | |

| Streamlit Frontend| | FastAPI Backend |

| | <------------------- | |

+--------------------+ JSON Response +------------+-------------+

|

|

▼

+--------------------+ +-----------------+

| pytesseract OCR | | Ollama + LLaMA2|

| (Image to Text) | | Summarize/Answer |

+--------------------+ +-----------------+

### **4. Tech Stack**

| **Component** | **Technology** |
| --- | --- |
| Frontend | Streamlit |
| Backend | FastAPI |
| OCR | pytesseract + Tesseract |
| LLM (Local) | Ollama + LLaMA2 |
| Image Handling | Pillow (PIL) |
| HTTP Client | requests |

### **5. Project Structure**

multimodal-assistant/

│

├── backend/

│ ├── main.py

│ ├── routes.py

│ ├── services/

│ │ ├── text\_handler.py

│ │ └── image\_handler.py

│ └── models/

│ └── llm\_model.py

│

├── frontend/

│ └── app.py

│

├── .env

├── requirements.txt # Dependencies

### 

### **6. Key Features**

#### **Text Query**

* User enters question.
* Sent to FastAPI → LLaMA2 through Ollama.
* Response is shown in Streamlit.

#### **Image Query**

* User inputs image.
* FastAPI employs Tesseract to read text.
* Text is summarized through LLaMA2 through Ollama.
* Response presented to user.

### **7. Setup Instructions**

#### **Pre-requisites**

* Python 3.10+
* Ollama installed:<https://ollama.com>
* Tesseract-OCR installed (add to PATH)

LLaMA2 pulled:  
***ollama pull llama2***

**# Clone & setup**

*git clone https://github.com/lekshmi-c/multimodal-assistant*

*cd multimodal-assistant*

*python -m venv .venv*

*.venv\Scripts\activate*

***# Install dependencies***

*pip install -r requirements.txt*

**# Start Ollama model**

*ollama run llama2*

**# Run backend**

*cd backend*

*uvicorn main:app --reload*

**# In another terminal, run frontend**

*cd ../frontend*

*streamlit run app.py*

### **8. Environment Config**

**pytesseract config in image\_handler.py:**

pytesseract.pytesseract.tesseract\_cmd = r"C:\Program Files\Tesseract-OCR\tesseract.exe"

### **9. Sample Use Cases**

#### **Image Use**

* Upload a picture with printed or handwritten text (e.g., news article, signboard, flyer, handwritten note, screenshot).
* Assistant uses OCR to extract the text and LLaMA2 to summarize the content intelligently.

#### **Text Use**

* Ask: "What are the advantages of open-source LLMs?"
* Obtain short, intelligent responses from LLaMA2.

### **10. Future Improvements**

* Insert image captioning with BLIP or CLIP.
* Insert audio query support with whisper.
* Insert user chat history & memory.
* Deploy through Docker or cloud.