Agentic File Assistant with LangChain and Local LLaMA

Abstract

In this project, I developed a local agentic file assistant powered by LangChain and a locally hosted LLaMA model via Ollama. The system is capable of extracting, understanding, and answering queries based on both text and image content from user-uploaded documents. This report details the architecture, implementation, and capabilities of the assistant, highlighting its potential as an extensible framework for document reasoning and multi-modal interaction.

Introduction

Agentic AI agents are autonomous systems that interact with tools and environments to achieve user goals through reasoning and planning. This project focuses on building a lightweight, LangChain-powered file assistant that exemplifies key agentic principles-tool use, memory, and multi-modal reasoning-via seamless document question-answering. The assistant accepts uploaded files, extracts both textual and image-based content, and uses a local LLM (LLaMA) for offline, private inference. The goal was to create a usable agent to demonstrate practical knowledge of LangChain framework design, LLM orchestration, and tool integration.

Architecture and Design

The assistant uses a modular LangChain-based agent architecture with the following components:

- Tooling System:
- file_search: Indexes and searches uploaded document content.
- python: Performs mathematical computations and data analysis.

- image understanding tool: Extracts and interprets charts or diagrams embedded in documents.

- Language Model Integration:
- LLaMA model hosted via Ollama was used as the backend model for inference.
- Integration with LangChain's LLMSingleActionAgent for agentic control flow.

- Memory and Agent Flow:
 - Conversational memory to retain context across queries.
 - Structured prompts to maintain consistent, role-aware agent behavior.

Implementation Overview

Environment:
- LangChain framework
- Ollama to host LLaMA model
- Python backend with tool APIs
- Web UI (planned): for frontend hosting and user interaction

Functional Pipeline:
1. Document Upload: User uploads PDF or image-based files.
2. Content Extraction:
- Text is parsed and indexed.
 - Images (e.g., charts) are interpreted using image understanding modules.

3. Query Handling:
- Agent routes questions to relevant tools.
- Tools return intermediate results.
- LLaMA model generates final natural language response based on results and history.

Planned Hosting Additions:
- API key-based LLM access for cloud-hosted demos
- Basic frontend to allow relatives, like the user's uncle, to interact and view outputs

Highlights and Capabilities
- Agentic Reasoning: Uses LangChain's agent class to choose between tools.
- Multi-modal File Parsing: Extracts both text and images from documents.
- Local Inference: Avoids OpenAI API usage by hosting LLaMA locally.
- Scalable Tool Use: Easily extendable to support additional tools or reasoning chains.

Conclusion

This project demonstrates the feasibility of creating a functional agentic file assistant by leveraging LangChain, local LLMs, and multimodal capabilities. It reflects practical understanding of agent construction, tool orchestration, and hybrid input processing, serving both as a prototype and a stepping stone toward advanced agentic AI systems