Agentic AI System: Research Assistant

# 1. Overview

The Agentic AI System is a multi-agent orchestration framework built using CrewAI. It demonstrates multi-agent coordination with a central controller agent and specialized agents that use both built-in and custom tools to accomplish a research-oriented task. This system focuses on automating information gathering, processing, and summarization.

# 2. System Architecture

The architecture includes a Controller Agent that orchestrates task delegation to specialized agents such as the Research Agent and Summarization Agent. It integrates memory management, tools, and a custom domain extractor to demonstrate agentic system capabilities. A diagram describing the system is included in the `docs/` folder.

# 3. Agent Roles and Responsibilities

• Controller Agent: Orchestrates tasks, delegates to specialized agents, handles workflow and memory.

• Research Agent: Retrieves information using a web search tool and performs analysis.

• Summarization Agent: Summarizes the content retrieved into concise insights.

# 4. Tool Integration and Functionality

• Web Search Tool: Searches and scrapes data from the internet.

• Data Processing Tool: Cleans and transforms collected data.

• Output Formatting Tool: Formats the final summaries for readability.

• Custom Domain Extractor: A custom-built tool to extract domain-specific insights.

# 5. Custom Tool Implementation

The custom tool `custom\_domain\_extractor.py` performs named entity recognition and domain-specific filtering. It enhances the overall system by isolating critical domain information and provides clean output for downstream tasks.

# 6. Challenges and Solutions

• Pydantic compatibility required backtracking to CrewAI version 0.35.8 for stable integration.

• OpenAI tools required accurate registration via @tool decorators to work correctly with CrewAI.

• Merge conflicts while committing to GitHub were handled through local preference resolution.

# 7. System Performance and Limitations

• Accurate task execution with readable summaries.

• Limitation: Static tool definitions; could benefit from dynamic agent feedback and correction loops.

# 8. Conclusion

The Agentic Research Assistant System showcases the power of CrewAI in orchestrating AI agents for research workflows. Through modular agent design, seamless tool integration, and a custom tool, it demonstrates a scalable framework for real-world productivity tasks.

# Project Structure

```  
agentic-system  
├── src  
│ ├── controller\_agent.py # Implements the ControllerAgent class for workflow orchestration  
│ ├── specialized\_agents # Contains specialized agents for specific tasks  
│ │ ├── research\_agent.py # Gathers and analyzes information  
│ │ └── summarization\_agent.py # Summarizes content  
│ ├── tools # Various tools for data handling and processing  
│ │ ├── web\_search\_tool.py # Retrieves information from the internet  
│ │ ├── data\_processing\_tool.py # Transforms and cleans data  
│ │ ├── output\_formatting\_tool.py # Formats output data for presentation  
│ │ └── custom\_domain\_extractor.py # Extracts domain-specific information  
│ ├── orchestration # Coordinates agent interactions  
│ │ └── workflow.py # Defines the workflow for task execution  
│ ├── memory # Manages contextual awareness  
│ │ └── memory\_manager.py # Implements memory management functions  
│ └── main.py # Entry point for the application  
├── tests # Contains unit tests for the system  
│ ├── test\_controller\_agent.py # Tests for the controller agent  
│ ├── test\_specialized\_agents.py # Tests for specialized agents  
│ ├── test\_tools.py # Tests for the various tools  
│ └── test\_workflow.py # Tests for the workflow implementation  
├── requirements.txt # Lists project dependencies  
├── README.md # Documentation for the project  
└── docs # Additional documentation  
 ├── architecture\_diagram.png # System architecture diagram  
 └── technical\_report.pdf # Comprehensive system report  
```