Agentic AI System

Abstract:

The objective of this project was to develop an **Agentic AI System** consisting of multiple agents capable of responding to dynamic user queries. Each agent was designed for a specific purpose: web search, financial data retrieval, image analysis .By leveraging modern libraries such as phi, groq, and APIs like DuckDuckGo and YFinance, the system demonstrates how agent-based architectures can efficiently address domain-specific tasks. This report outlines the implementation, challenges faced, and the results achieved.

Introduction:

The concept of Agentic AI revolves around creating intelligent agents that can autonomously perform tasks in specific domains. These agents interact with users, process queries, and provide actionable insights. With the exponential growth of information, AI agents have become indispensable for automating repetitive tasks, enhancing productivity, and delivering accurate information.

In this project, I have implemented four distinct agents:

1. **Web Search Agent**: Retrieves answers from the web.
2. **Finance Agent**: Provides financial insights, such as stock prices and analyst recommendations.
3. **Image Analysis Agent**: Analyzes images and provides contextual information.

The project demonstrates the flexibility and potential of agentic systems in real-world applications.

Methodology

Technologies Used

- **Programming Language**: Python 3.12
- **Libraries and Frameworks**:
    - phi: For creating AI agents.
    - groq: For leveraging pre-trained models like llama-3.3-70b-versatile.
    - duckduckgo-search: For web search integration.
    - yfinance: For retrieving financial data.

    - dotenv: For managing environment variables.
- **Development Environment**:
    - Python Virtual Environment managed by uv.

System Design

1. **Web Search Agent**:
    - Purpose: Retrieve information from the web using DuckDuckGo.
    - Tools: DuckDuckGo integrated into the phi framework.
2. **Finance Agent**:
    - Purpose: Fetch financial data, including stock prices and analyst recommendations, using the YFinance API.
3. **Image Analysis Agent**:
    - Purpose: Analyze and extract insights from images using Groq's AI model.

Implementation

1. Web Search Agent

- **Functionality**: Retrieves answers from the web.
- **Example Query**: "What is the capital of France?"
- **Code Summary**: The agent uses DuckDuckGo to fetch web results and processes them with Groq's language model for a summarized response.

2. Finance Agent

- **Functionality**: Provides real-time stock prices, analyst recommendations, and company news.
- **Example Query**: "What is the current stock price of Tesla (TSLA)?"
- **Code Summary**: This agent integrates the YFinance API to fetch financial data and uses Groq for formatting the output.

3. Image Analysis Agent

- **Functionality**: Analyzes an image URL and provides detailed insights.
- **Example Query**: "What is the significance of the monument in this image?"

- **Code Summary**: This agent uses Groq's pre-trained image analysis model and DuckDuckGo to provide contextual analysis.

Results

Each agent successfully executed its tasks:

1. **Web Search Agent**:
    - Query: "When did Swami Vivekananda born?"
    - Response: "Swami Vivekananda was born on January 12, 1863."
2. **Finance Agent**:
    - Query: "What is the current stock price of HDFCBANK in NSE?"
    - Response: "HDFCBANK's current stock price is Rs. 1642.00."
3. **Image Analysis Agent**:
    - Image Link: https://upload.wikimedia.org/wikipedia/commons/thumb/1/1d/Taj_Mahal_%28Edited%29.jpeg/375px-Taj_Mahal_%28Edited%29.jpeg
    - Query: "Describe the historical and architectural significance of the monument in this image. "
    - Response: "The image shows the Taj Mahal, a UNESCO World Heritage Site in Agra, India. Built by Emperor Shah Jahan in memory of his wife Mumtaz Mahal, it is a masterpiece of Mughal architecture, featuring white marble, intricate carvings, and symmetrical gardens."

Conclusion

This project demonstrated how Agentic AI systems could streamline tasks in web search, finance, and image analysis. By leveraging modular design, the agents are versatile, reusable, and can handle a wide range of queries. The multi-agent system further highlights the potential of combining specialized agents for complex problem-solving.

References

1. Groq API Documentation: https://www.groq.com
2. YFinance API: https://pypi.org/project/yfinance/
3. Phi Framework: https://phidata.com

Disha Karmakar
22053860
CSE-39