Multi-Agentic Workflow for Generating AI/ML Use Cases

Objective

The objective of this project is to design and implement a multi-agent system that automates the generation of AI/ML use cases tailored to specific industries or companies. The system leverages multiple agents to conduct market research, identify industry trends, propose impactful AI/ML solutions, and provide references and resources for implementation. The ultimate goal is to enhance operations, customer satisfaction, and overall efficiency.

Methodology

Architecture Overview

The system is designed with a modular and agent-based architecture. Each agent performs a specific task, contributing to the overall output. These agents are integrated into a Streamlit app for user-friendly interaction. Below is a detailed breakdown of each component:

Agents

1. Industry Research Agent:

    ○ Purpose: Conducts web searches to identify the industry and key focus areas of the given company.
    ○ Implementation: Uses the googlesearch-python library to fetch URLs related to the company’s industry and offerings.

2. Market Trends & Standards Agent:

    ○ Purpose: Analyzes current AI/ML trends and standards within the specified industry.
    ○ Implementation: Performs Google searches for AI/ML trends and gathers insights from the top results.

3. Use Case Generation Agent:

    ○ Purpose: Proposes AI/ML use cases relevant to the industry’s needs and strategic focus areas.
    ○ Implementation: Generates predefined use cases such as supply chain optimization, predictive maintenance, and workforce training.

4. Resource Asset Agent:

    - Purpose: Collects references, datasets, and resources to support the implementation of the proposed use cases.
    - Implementation: Provides links to datasets from platforms like Kaggle and GitHub.

5. Proposal Generation Agent:

    - Purpose: Aggregates all information into a structured and exportable format (CSV and markdown).

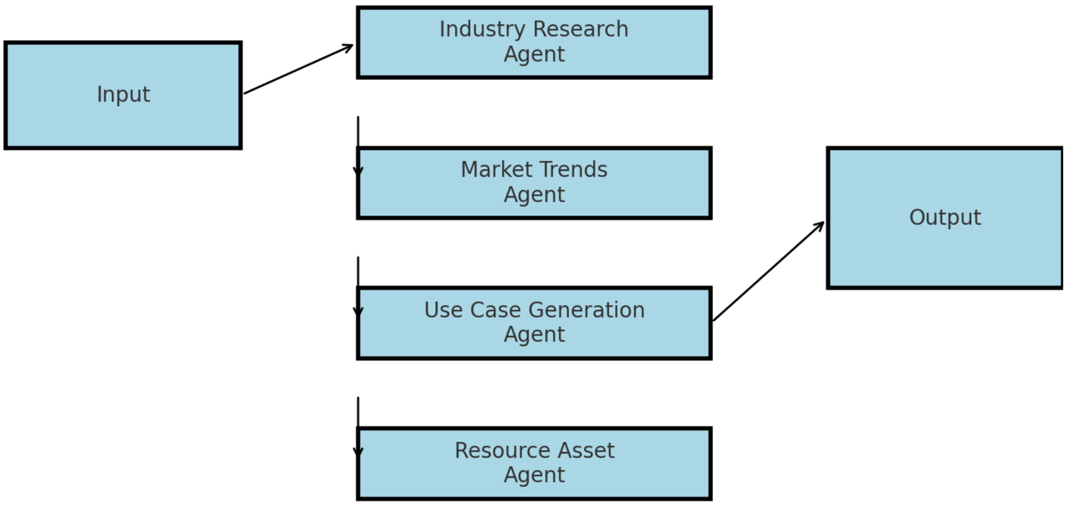
Workflow Integration

The agents are integrated into a Streamlit app that facilitates:

- User input for the company and industry.
- Real-time execution of the agents.
- Display of results in tabular format.
- Downloadable outputs (CSV and markdown).

Architecture

Architecture Flowchart for Multi-Agentic Workflow



The architecture illustrates the following workflow:

1. Input:
    - Company Name
    - Industry Name
    - Focus Areas (e.g., Customer Experience, Operations)
2. Agents:
    - Industry Research Agent
    - Market Trends Agent
    - Use Case Generation Agent
    - Resource Asset Agent
3. Output:
    - DataFrame displayed in the app
    - CSV file with use cases, descriptions, and references

Results

Example Output: Steel Industry Use Cases

Use Case	Description	Reference
Supply Chain Optimization	To enhance demand forecasting, inventory management, and logistics, reducing costs and waste while improving cash flow, product availability, and delivery speed.	Kaggle Link
Predictive Maintenance	To predict equipment failures, reducing unplanned downtime and optimizing maintenance scheduling. It extends equipment lifespan and lowers costs.	GitHub Link
Workforce Training	Provides personalized training and real-time guidance, reducing errors and improving productivity.	Kaggle Link

Key Features of the Output:

1. Comprehensive Use Cases: Clearly defined use cases with actionable descriptions.
2. References: Curated links to datasets and resources to aid implementation.
3. Downloadable Formats: Outputs available as both CSV and markdown files.

Deliverables

1. Source Code: Python script (multi_agent_use_case_generator.py) and requirements.txt.
2. Report: This document, explaining the methodology and results.
3. Demo Video: A walkthrough showcasing the workflow and app functionality.
4. Deployed Application (Optional): Hosted on Streamlit Cloud with a public URL.

Appendix

How to Run the Application

Install dependencies:

```
pip install -r requirements.txt
```

1. Run the Streamlit app:

   ```
   streamlit run multi_agent_use_case_generator.py
   ```

2. Input the company and industry details in the app.

3. Generate and download the results.

References

1. Kaggle: Supply Chain Optimization Datasets
2. GitHub: Predictive Maintenance Resources