Case Study – AI Engineer

Multi-Agent Data Insight Discovery System

Goal

Design and implement a prototype multi-agent system that analyses the financial dataset provided to uncover at least two most potentially valuable insights, trends, or anomalies. The system should leverage Large Language Models (LLMs) for tasks like hypothesis generation, data interpretation, and summarisation.

Guidance

This case study is designed to be completed in approximately 4-6 hours. We are most interested in your approach to the problem, your design thinking for the multi-agent system, the quality and explanation of your prompt engineering, your core implementation skills, and your ability to evaluate the outcome. A fully polished, production-ready application is not expected. We also expect you to use AI tools (like GitHub Copilot, Cursor, etc.) to assist you and if you do, please list them in your submission.

Dataset

A sample financial dataset is provided (*Financial Sample.xlsx*). It contains sales and profit data sorted by market segment and country/region.

Task Requirements

1. Design a multi-agent system:
    a. Design a system with at least two distinct agent roles. Define the responsibilities of each agent and how they interact (orchestration).
    b. You are free to use any AI Agent Framework (e.g., LangChain, LangGraph, Microsoft Semantic Kernel, Microsoft Autogen, etc.) or build the agent orchestration logic from scratch.
    c. The system can be either designed in a Q&A format or as a backend process.

2. Implement the system:
    a. Implement your designed system in Python.
    b. The system should interact with LLM(s) via API (e.g., OpenAI, Anthropic, Gemini) for relevant tasks.
    c. Focus on clearly separating agent logic and managing the interaction flow.

3. Evaluate the outputs of the system:
    a. How would you ensure the outputs generated is accurate and reliable?
    b. How would you control the overall system performance and costs?
    c. How to make system transparent for auditing purpose?

Submission

Please submit your solution including:

1. Approach & Architecture Design: Explain your overall approach, the defined agent roles, their responsibilities, and the interaction workflow
2. Source Code & Prompts: all Python scripts and the final prompts used for each LLM interaction. Explain your prompt design choices and rationale. **Please do not include your API keys in the submission.**
3. Setup & Run Instructions:
    a. Provide clear instructions on how to set up the environment and run your code.
    b. Include a requirements.txt file listing all necessary packages.

c. Mention any AI tools used during development.

4. Generated Insights & Evaluation:
    a. Show the key insight(s) generated by your system.
    b. Provide a brief evaluation: Discuss the quality/reliability of the generated insights including a self-critique – what worked well, what were the limitations of your prototype, and what would be the next steps for improvement?
    c. (Optional Suggestion: While extensive unit/integration testing isn't required, briefly discussing your testing approach or including simple test examples for key components, if feasible, would be valued).

Presentation: Selected candidates will be asked to present their solutions, explaining their design choices and findings, during the first round of interviews.