# **Project Charter**

**Title:** Agentic AI for Investment Market Research (MCP Client–Server Framework with Yahoo Finance + LLM)

## **1. Purpose**

The project aims to design and implement an autonomous investment research agent that leverages **agentic AI workflows**, **Yahoo Finance data**, and **LLM-driven reasoning**. The solution will adopt a **modular MCP client–server architecture** to ensure extensibility, transparency, and responsible AI practices.

## **2. Scope**

### **In-Scope**

* **MCP Server Development**
  + Stock data retrieval via Yahoo Finance API.
  + News ingestion and sentiment analysis tools.
* **MCP Client Development (Autonomous Research Agent)**
  + Planning of research steps.
  + Dynamic routing of tasks (stock vs news).
  + Prompt chaining for multi-step workflows (news preprocessing, summarization).
  + Evaluator–optimizer reflection for iterative self-improvement.
  + Persistent memory for cross-run learning.
* **LLM Integration**
  + Planning, summarization, reflection, and reasoning prompts.
  + Workflow orchestration using prompt chaining.
* **Memory Management**
  + Persistent memory store to record insights.
  + Recall of past analysis to improve accuracy and efficiency over time.

## **3. Key Deliverables**

1. **MCP Server (mcp\_server.py):** Provides financial analysis tools.
2. **MCP Client (mcp\_client.py):** Implements autonomous research agent with planning, routing, and reflection.
3. **LLM Integration Module (llm\_utils.py):** Encapsulates prompt templates and LLM interactions.
4. **Persistent Memory Store (agent\_memory.json):** Maintains cross-run insights and context.
5. **Documentation:**
   * Setup Guide
   * Usage Guide
   * Demo run with example outputs

## **4. Project Objectives**

1. **Build an Autonomous Research Agent**
   * Enable planning, routing, multi-step reasoning, reflection, and memory.
2. **Implement Workflow Patterns**
   * **Prompt Chaining:** Ingest → Preprocess → Classify → Extract → Summarize.
   * **Routing:** Intelligent selection of stock or news analysis tools.
   * **Evaluator–Optimizer:** Generate → Evaluate → Refine for high-quality summaries.
3. **Integrate LLM + Memory**
   * Use LLMs for reasoning, analysis, and self-reflection.
   * Store and recall insights for continual improvement.
4. **Adopt MCP Architecture**
   * Modular server–client separation.
   * Extensible design for future financial datasets/APIs.

# Team Members: Roles and Responsibilities

| **Team Member** | **Primary Focus** | **Key Responsibilities** | **Deliverables** |
| --- | --- | --- | --- |
| **Gangadhar (Team Leader)** | MCP Client (Agent + Orchestration) | - Lead design & integration- Implement mcp\_client.py with planning, routing, and reflection- Develop prompt chaining workflow- Ensure client-server communication- Manage GitHub repo & project timeline | - mcp\_client.py (agent logic + routing)- Integrated client-server system- Final consolidated demo |
| **Akshobya (Member)** | MCP Server (Tools + Data Sources) | - Implement mcp\_server.py with Yahoo Finance stock retrieval & news ingestion- Add sentiment analysis tool- Expose MCP server endpoints- Write unit tests for server- Document server setup | - mcp\_server.py (financial tools)- Test scripts for server- Server-side setup documentation |
| **Nagarajan (Member)** | LLM Integration + Memory + Documentation | - Implement llm\_utils.py with LLM prompts & wrappers- Build persistent memory (agent\_memory.json)- Develop evaluator–optimizer reflection prompts- Write setup guide, usage guide, demo run docs- Support final presentation | - llm\_utils.py (LLM utilities)- agent\_memory.json (memory store)- Documentation (setup, usage, demo) |

.

# **Project Timeline (6 Weeks)**

| **Week** | **Gangadhar (Team Leader)** | **Akshobya (Member)** | **Nagarajan (Member)** | **Milestones** |
| --- | --- | --- | --- | --- |
| **Week 1** | Finalize architecture design, set up GitHub repo, create project skeleton (mcp\_client.py) | Set up mcp\_server.py skeleton with placeholder endpoints | Draft llm\_utils.py structure & memory design plan | Project repo ready, architecture defined |
| **Week 2** | Implement planning & routing logic in client | Implement Yahoo Finance stock retrieval tool | Implement LLM wrappers (basic prompts) | Client can call stock data via server |
| **Week 3** | Add prompt chaining workflow (ingest → preprocess → summarize) | Add news ingestion + sentiment analysis tool | Implement persistent memory (agent\_memory.json) | Server provides both stock + news tools |
| **Week 4** | Integrate evaluator–optimizer reflection in client | Write unit tests for server endpoints | Connect memory + LLM with client | Full pipeline: Client ↔ Server ↔ LLM ↔ Memory |
| **Week 5** | Test integrated system, debug orchestration | Optimize server performance, add logging | Prepare setup & usage documentation | End-to-end workflow runs successfully |
| **Week 6** | Lead demo run prep, finalize mcp\_client.py | Finalize mcp\_server.py, contribute to demo | Finalize documentation, polish presentation | Final demo & project submission |

**Balance Check:**

* Everyone codes in **their own module first**, then works together on **integration & testing**.
* **Gangadhar**: orchestration + leadership.
* **Akshobya**: server + testing.
* **Nagarajan**: LLM + memory + documentation.

