Campaign Performance Optimization Platform

Brief Design Summary

July 5, 2025

1. Overview

This project implements a multi-agent system for marketing budget analysis and reallocation. Users interact via a chat interface; a central **router agent** delegates requests to specialized sub-agents (budget, web search, generic). The system demonstrates:

- **Stateful orchestration** with LangGraph
- **LLM routing** and function-calling via LangChain
- **Modular microservices** (FastAPI + MCP tool hub)
- **Asynchronous, horizontally scalable** design

2. Architecture

1. *Client* (CLI or Streamlit UI) sends user query `/chat`.
2. *FastAPI* server receives, restores session state from Redis.
3. *Router Agent* uses keyword shortcuts & LLM prompt to select branch.
4. *Sub-agent* (e.g. BudgetRecommender) runs via AgentExecutor:
    - Calls MCP tool endpoints (`/mcp/get_budget`, `/mcp/save_proposal`).
    - Returns Markdown table or confirmation.
5. State is updated, saved back to Redis; reply returned to client.

3. Key Components

3.1 Router Agent

- LangGraph state machine: routes to `budget_node`, `search_node`, or `generic_node`.
- Uses simple keywords plus GPT-4O prompt for disambiguation.

3.2 Budget Recommender Agent

- ChatOpenAI with function-calling to pull metrics and commit proposals.
- Shared conversation memory via LangChain's `ConversationBufferMemory`.
- MCP tool hub (FastAPI + Pydantic) exposes database helpers.

3.3 MCP Tool Hub

- Standalone FastAPI service at `/mcp/{tool}/{schema,invoke}`.
- Auto-discovers Pydantic-typed tool modules.
- Stateless, can be load-balanced independently.

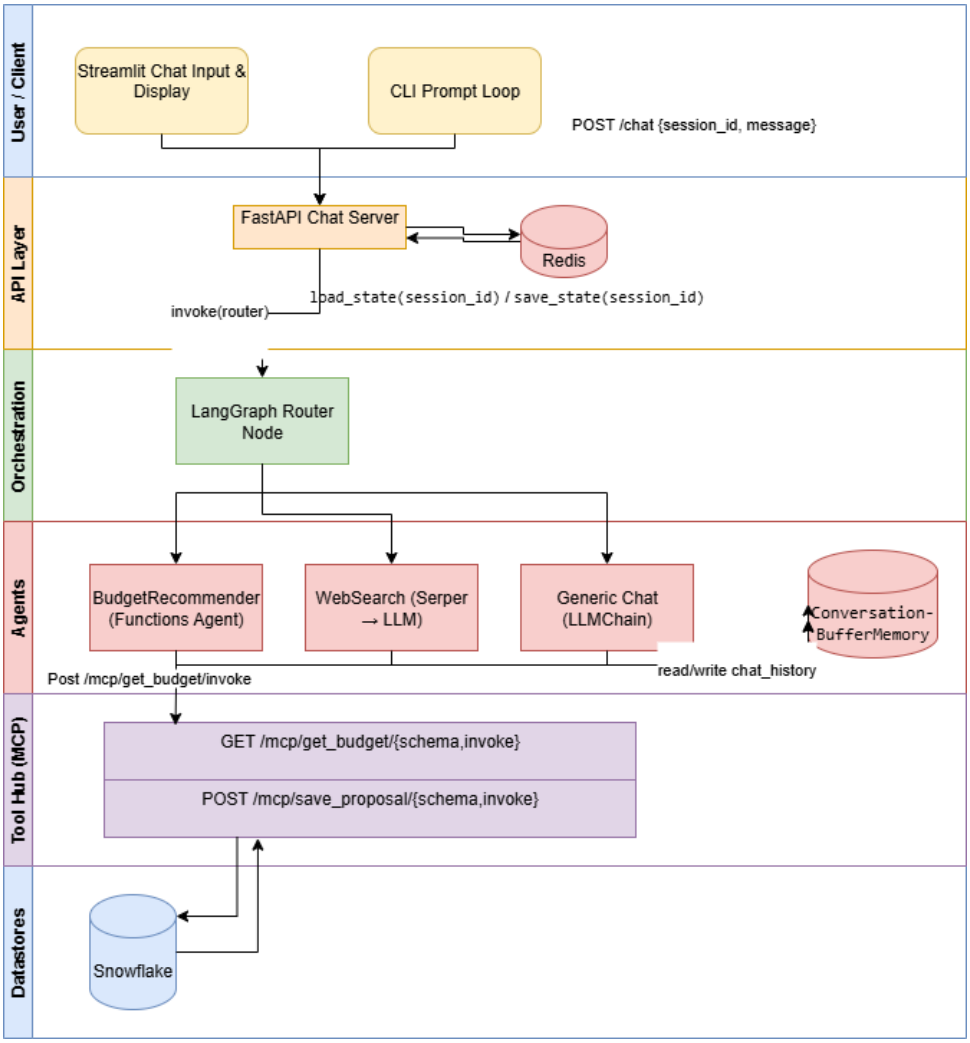

Figure 1: *
System Flow Diagram

4. Scalability & Asynchrony

- **Horizontal scale**:
  - Stateless FastAPI MCP services in Docker/Kubernetes.
  - Shared session store in Redis.
- **Asynchronous I/O**:
  - FastAPI endpoints as `async def`.
  - Tool wrappers use `httpx.AsyncClient`.
  - Blocking DB calls offloaded to threads or async connector.
- **Maintainability**:
  - Clear separation of concerns (router vs. agents vs. tools).
  - Modular "tools/" directory for extensions.

5. Future Enhancements

- Integrate vector embeddings for caching & similarity search.

- Add reinforcement-learning feedback loop for automated budget tuning (currently with human-in-the-loop).
- Extend MCP hub with additional domain tools (weather, CRM APIs).
- Provide multi-modal support (image analysis of creative assets).

6. Conclusion

This solution meets—and in many areas exceeds—the project requirements, showcasing advanced orchestration, stateful context management and hallucination mitigation, tool-calling with MCP, asynchronous design, and a path toward production-grade horizontal scalability.