Distributed Multi-Agent LLM System: Consensus Detection and Analysis Across Language Models

Manasvi Goyal[1], Sukanya Krishna[1]

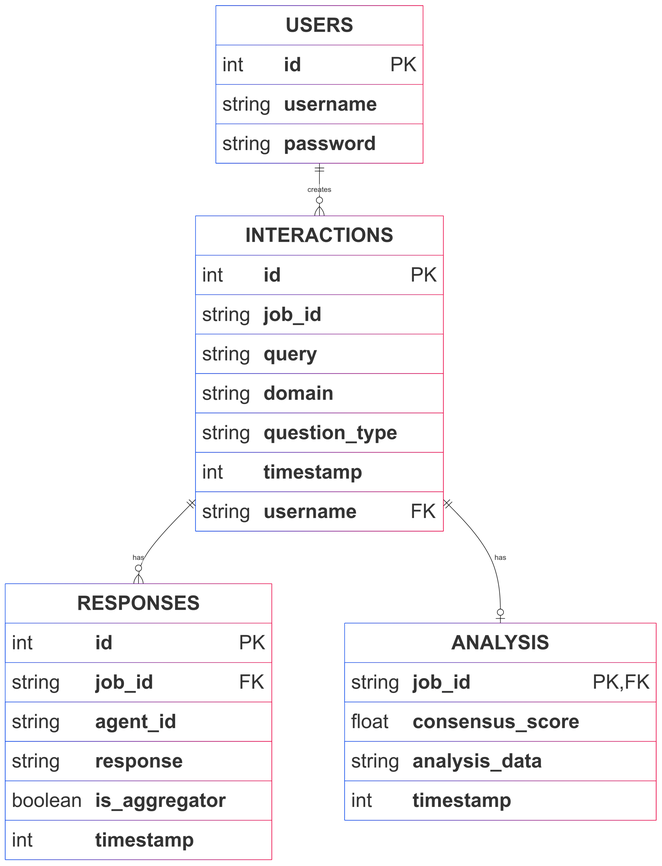
1. Harvard University, School of Engineering and Applied Sciences, Allston, MA, USA

Harvard John A. Paulson School of Engineering and Applied Sciences

Need and Motivation

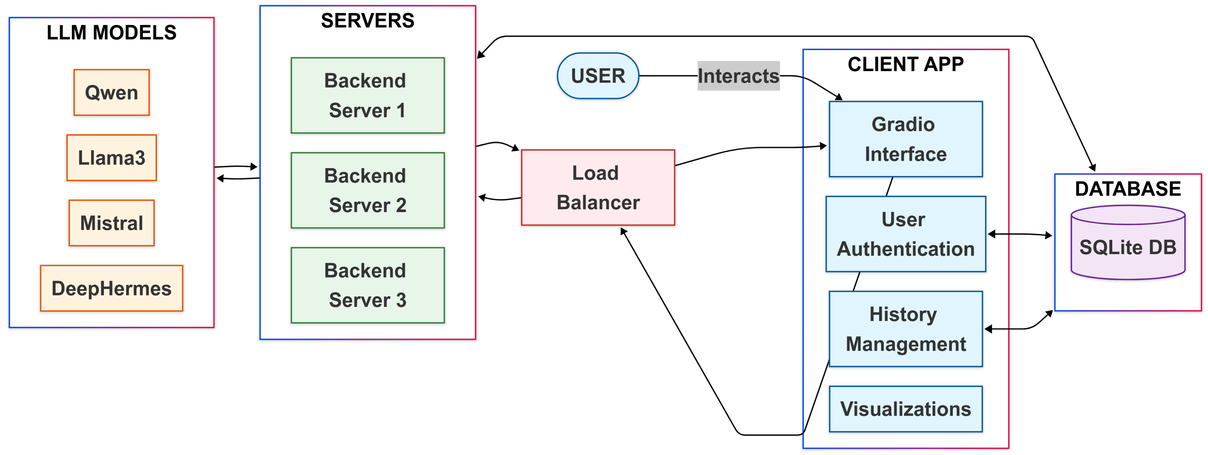
Single LLMs vary in reliability, ethics, and output quality, making it insufficient for effective decision-making. Challenges include inconsistent responses, lack of fault tolerance, high API costs, and limited ethical coverage. We propose a distributed multi-agent system where multiple LLMs reason independently and an aggregator synthesizes a consensus, improving persistence, ethical diversity, and efficiency.

Persistence and Storage Schema



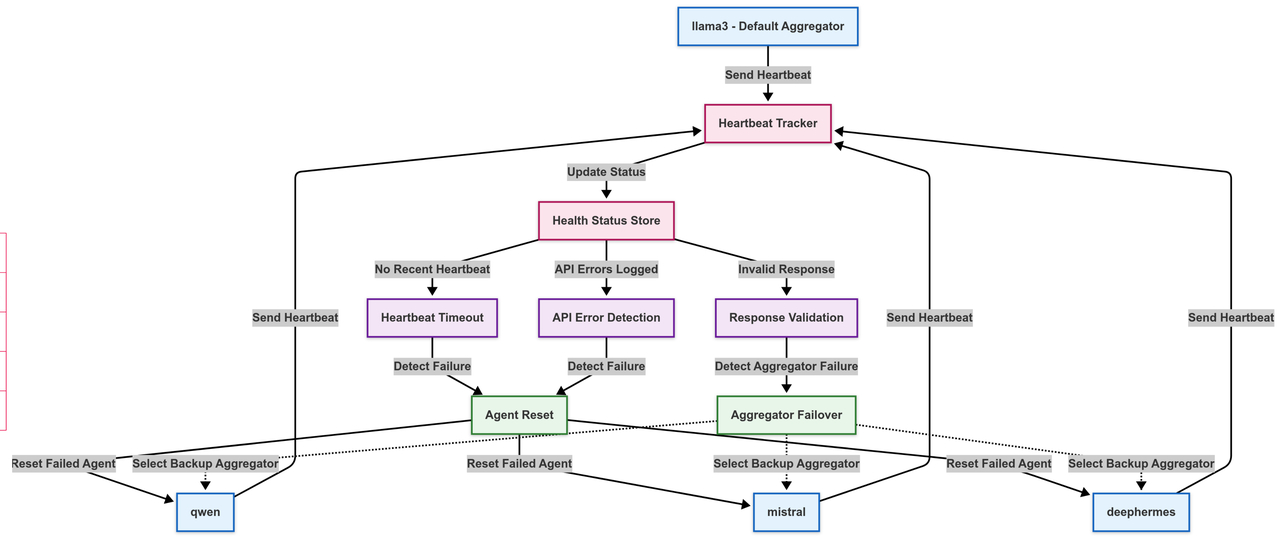
Persistence is critical for ensuring that user queries, model responses, and consensus summaries are reliably stored. This allows users to revisit past interactions, reload previous results, and continue sessions without re-querying external APIs. By caching responses locally, the system reduces redundant API calls, significantly reducing cost and latency.

Architecture of the Distributed Multi-Agent LLM System



User authenticates and submits a query via the client. A load balancer routes it to a backend server, which forwards it in parallel to multiple LLMs. Each model generates a response, and the designated aggregator synthesizes a consensus summary. The server analyzes responses for similarity, sentiment, and tone. All data is saved to the database, and final results with visualizations are returned to the client interface.

Fault Tolerance and Agent Failover Mechanism



Each agent sends a heartbeat to a central tracker every 10 seconds. The tracker updates the agent's health status based on heartbeat timeouts and error detections. If no heartbeat is received or API errors are logged, the agent is marked unhealthy and retried with exponential backoff. If failures persist, the system triggers an agent reset or initiates an aggregator failover, selecting a healthy backup aggregator from available models.

User Interaction and Analysis