Agentic AI Orchestration System - Documentation

Agentic AI Orchestration System

Objective:
Design and implement an Agentic AI system that orchestrates prompts across multiple LLMs, selects the most accurate output, supports model-based prompt routing, and allows seamless hot-swapping of LLMs without downtime.

Background:
Using multiple LLMs (e.g., Gemini, PaLM, Ollama) helps compare outputs, provide fallback, and improve quality. Frequent upgrades demand modularity and version tracking.

Architecture Overview:
- User -> Pub/Sub -> Cloud Function -> Prompt Router
- Prompt Router -> LLMs (Groq, Together, Ollama)
- Responses -> Evaluator -> BigQuery Logging
- (Optional) Human Feedback via Local Interface

LLM Version Tracking & Hot-Swapping:
- Versions stored in config.py for traceability.
- Models hot-swapped by toggling 'enabled' field, no redeployment required.

Routing Logic:
- Based on type, latency, and availability.
- Router intelligently skips or prioritizes LLMs accordingly.

Monitoring & Logging:
- All responses logged in BigQuery: prompt, model, version, selected answer.
- Helps detect regressions and failures.

Human-in-the-loop:
- Local runner allows manual evaluation of model outputs for feedback.

API & Services:
- Pub/Sub: prompt trigger
- Cloud Functions: serverless orchestration
- BigQuery: logging and tracking
- Groq, Together, Ollama: model endpoints

Conclusion:
Cloud-native, scalable, and flexible system for orchestrating LLMs, enabling the best response selection and seamless updates.