PROVISIONAL PATENT APPLICATION (UPDATED)

COVER SHEET

Title of Invention: UltraLLMOrchestrator System (UltrAI)

Inventor(s): Joshua Field 1625 SW Alder St. #510 Portland, OR 97205 Citizenship: United States of America

Correspondence Address: Joshua Field 1625 SW Alder St. #510 Portland, OR 97205

Application Filed By: Inventor

SPECIFICATION

Title: UltraLLMOrchestrator System (UltrAI)

Cross-Reference to Related Applications

[Not Applicable]

Field of the Invention

This invention relates to advanced artificial intelligence systems, specifically to the orchestration and integration of multiple Large Language Models (LLMs) in a unified, extensible, and dynamically configurable analysis platform. The invention addresses the need for robust, multi-model reasoning, dynamic model management, and developer-friendly extensibility in AI-driven analytical workflows.

Background of the Invention

Large Language Models (LLMs) have revolutionized natural language processing, but their deployment in real-world analytical tasks is often limited by the constraints of single-model architectures. Individual LLMs, while powerful, are subject to inherent biases, knowledge gaps, and inconsistent reasoning. Current solutions typically rely on static, single-model pipelines or ad hoc multi-model integrations that lack structure, extensibility, and systematic quality control.

Existing LLM orchestration approaches are often rigid, requiring significant codebase changes to add new models or modify analytical workflows. They lack a unified framework for structured, multi-stage analysis, and do not provide mechanisms for runtime extensibility, automated quality evaluation, or developer onboarding. As a result, organizations and developers face high barriers to integrating new models, experimenting with orchestration strategies, and ensuring the reliability and depth of AI-generated insights.

There is a critical need for a system that:

- Seamlessly orchestrates multiple LLMs through configurable, pattern-based analysis pipelines.
- Supports runtime registration and deregistration of models and orchestration logic.
- Provides automated, multi-dimensional quality evaluation of model outputs.
- Enables rapid developer onboarding and integration of new models, patterns, and business logic.
- Delivers robust, reliable, and context-aware analytical outputs that surpass the capabilities of any single LLM.

Summary of the Invention

The present invention, UltraLLMOrchestrator System (UltrAI), is a next-generation AI analysis platform that orchestrates multiple LLMs through a dynamically configurable, multi-stage pipeline. UltrAI enables users and developers to:

- Register, configure, and orchestrate any number of LLMs at runtime, without codebase modification or downtime.
- Define and extend analysis patterns—structured, multi-stage workflows that guide LLMs through critique, fact-checking, perspective-taking, synthesis, and more.
- Leverage a plugin-like architecture for both models and orchestration logic, supporting rapid experimentation and integration of new AI capabilities.
- Automatically evaluate the quality of each model's output across multiple dimensions (coherence, technical depth, strategic value, uniqueness), with metrics tracked for continuous improvement.
- Onboard new developers and models quickly, supported by comprehensive documentation, onboarding guides, and a robust test suite.

Key innovations and advantages include:

- **Dynamic model registry:** Models can be added, removed, or reconfigured at runtime via a unified API, enabling rapid adaptation to new LLMs and providers.
- **Configurable orchestration pipeline:** The sequence, logic, and participants of each analysis stage are defined by extensible patterns, supporting arbitrary workflows and model combinations.
- **Extensible plugin system:** Both models and orchestration strategies are implemented as plugins, allowing for easy extension, replacement, or customization without core code changes.
- **Automated, multi-dimensional quality evaluation:** Each model's output is scored and tracked, enabling data-driven model selection, performance tuning, and transparent reporting.
- **Developer-centric design:** The system is built for rapid onboarding, with clear extension points, thorough documentation, and a comprehensive test suite to ensure reliability and ease of integration.
- **Seamless document/context integration:** Uploaded documents are processed, chunked, and incorporated as context in analysis tasks, maximizing the relevance and depth of LLM outputs.
- **Integrated business logic and pricing:** The platform supports flexible pricing models, usage tracking, and business logic, making it suitable for commercial deployment.

UltrAI's architecture is illustrated in Figure 1 (DFD) and Figure 2 (Codebase Structure), which highlight the system's modularity, extensibility, and dynamic orchestration capabilities.

Brief Description of the Drawings

Figure 1: Data Flow Diagram (DFD) illustrating the overall system architecture and information flow of the UltraLLMOrchestrator System (UltrAI), including the dynamic model registry and configurable orchestration pipeline.

Figure 2: Codebase structure diagram showing the organization of the application's key components, modules, and their relationships, with emphasis on extensibility and modularity.

Detailed Description of the Invention

1. System Overview

UltrAI is a comprehensive AI analysis platform that orchestrates multiple LLMs through configurable, pattern-based pipelines. The system is designed for:

- Enhanced analytical quality through multi-model synthesis and critique.
- Rapid integration of new models and orchestration strategies.
- Automated, transparent quality evaluation and reporting.
- Developer-friendly extensibility and onboarding.

2. System Architecture

The UltrAI system employs a modular, client-server web architecture (see Figure 1 and Figure 2):

2.1 Frontend

- Built as a modern React SPA (TypeScript, Vite, Tailwind CSS).
- Provides intuitive interfaces for prompt input, document upload, model/pattern selection, and result visualization.
- Supports responsive design and rich user experience features (history, sharing, real-time feedback).

2.2 Backend

- Python API using FastAPI, serving as the central orchestrator.
- Modularized with routers for analysis, documents, models, pricing, users, and health.
- Manages application state, business logic, and external integrations.

2.3 Core Orchestration Engine (MultiLLMOrchestrator)

- Receives analysis requests and orchestrates LLMs through configurable, multi-stage pipelines.
- Supports runtime registration/deregistration of LLMs and orchestration patterns.
- Enables both parallel and sequential execution of LLM calls.
- Tracks token usage, quality metrics, and intermediate outputs for each stage.
- Extensible via subclassing or plugin registration, allowing custom orchestration logic and new analysis patterns.

2.4 Document Processing Module (UltraDocumentsOptimized)

- Handles file uploads, text extraction, and intelligent chunking.
- Integrates document context into analysis workflows, maximizing LLM relevance and depth.

2.5 Pricing & Business Logic Module

- Calculates costs based on token usage, model selection, and feature tiers.
- Implements markups, discounts, and token efficiency factors for flexible business models.
- Manages user authorization, usage tracking, and billing.

2.6 Database

- Stores user accounts, document metadata, analysis history, pricing configs, and cached results.
- Supports collaborative features via shared analysis links.

2.7 External LLM Services Integration

- Securely connects to third-party LLM providers (OpenAI, Anthropic, Google, Mistral, etc.).
- Abstracts API differences and manages credentials.

2.8 Caching Layer

- In-memory TTL cache (cachetools) for fast retrieval of repeated analysis requests.
- Reduces latency and operational costs.

2.9 Error Monitoring

- Integrates with real-time error tracking (e.g., Sentry) for robust monitoring and rapid issue resolution.

3. Detailed Features

3.1 Core Analysis & Orchestration

3.1.1 Multi-LLM Orchestration

- Executes analysis tasks across user-selected LLMs, leveraging their complementary strengths.
- Supports dynamic, runtime model registration and deregistration.
- Orchestration pipelines are fully configurable, supporting arbitrary stages (e.g., initial, meta, synthesis) and model sets.
- Enables both parallel and sequential LLM execution, with intermediate outputs tracked and organized.
- Token usage and quality metrics are collected for each model and stage, supporting transparent reporting and optimization.

3.1.2 Pattern-Based Analysis

- Guides LLM interaction using extensible, structured patterns (e.g., critique, fact-check, perspective, confidence analysis).
- Each pattern defines a unique, multi-stage workflow with custom prompt templates and instructions.
- Patterns can be added, modified, or replaced without core code changes, supporting rapid experimentation.

3.1.3 “Ultra” Model Synthesis

- Designates a user-selected “Ultra” LLM to synthesize intermediate results into a final, enhanced output.
- Synthesis logic is pattern-driven and extensible, supporting new synthesis strategies as plugins.

3.1.4 Model Registration & Extensibility

- New LLMs can be registered at runtime via a simple API, with no need for codebase modification or redeployment.
- Orchestration strategies and analysis patterns can be extended or replaced via subclassing or plugin registration.
- Comprehensive documentation and a robust test suite support rapid developer onboarding and integration.

3.2 Document Handling

- Uploaded documents are processed, chunked, and stored for use as context in analysis tasks.
- Intelligent context integration maximizes the relevance and depth of LLM outputs.

3.3 Pricing, Billing & Business Model

- Flexible pricing models based on token usage, model selection, and feature tiers.
- Supports markups, discounts, token efficiency factors, and business model simulation.
- Usage tracking, reporting, and billing integration for commercial deployment.

3.4 User Interface & Experience

- Intuitive, responsive UI for prompt input, document management, model/pattern selection, and result visualization.
- Features include history management, sharing, real-time feedback, and dynamic cost estimation.

3.5 System Operations & Development

- Modular, testable codebase with clear extension points for models, patterns, and business logic.

- Comprehensive test suite, health/metrics endpoints, and performance monitoring.
- Supports configuration via environment variables/files and scalable deployment (Docker, Vercel, etc.).

4. Novel Technical Aspects

The UltrAI system introduces several novel technical aspects, including:

- **Dynamic, runtime model registration and deregistration** via a unified API, enabling rapid integration of new LLMs and providers.
- **Configurable, pattern-based orchestration pipelines** supporting arbitrary analysis stages, model sets, and custom logic.
- **Automated, multi-dimensional quality evaluation** for each model and stage, with metrics tracked for performance analysis and optimization.
- **Plugin-like extensibility** for both models and orchestration logic, supporting rapid experimentation and adaptation.
- **Support for both parallel and sequential model execution** within the same analysis pipeline.
- **Developer onboarding and testability as first-class features**, with robust documentation, onboarding guides, and a comprehensive test suite.
- **Seamless document/context integration** for context-aware analysis.
- **Integrated business logic and pricing** for commercial deployment.

These innovations are illustrated in Figures 1 and 2, which show the system’s dynamic, extensible architecture and modular codebase organization.

5. Figures/Diagrams

Figure 1: Data Flow Diagram (DFD) of the UltraLLMOrchestrator System (Updated)

Figure 1: Data Flow Diagram of the UltraLLMOrchestrator System

Figure 1: Data Flow Diagram of the UltraLLMOrchestrator System

Figure 1. The UltraLLMOrchestrator DFD illustrates the flow of information and control between the user interface, backend API, dynamic model registry, configurable orchestration pipeline, document processing, LLM execution, quality evaluation, synthesis, and result display. The architecture supports runtime extensibility and robust, multi-stage orchestration.

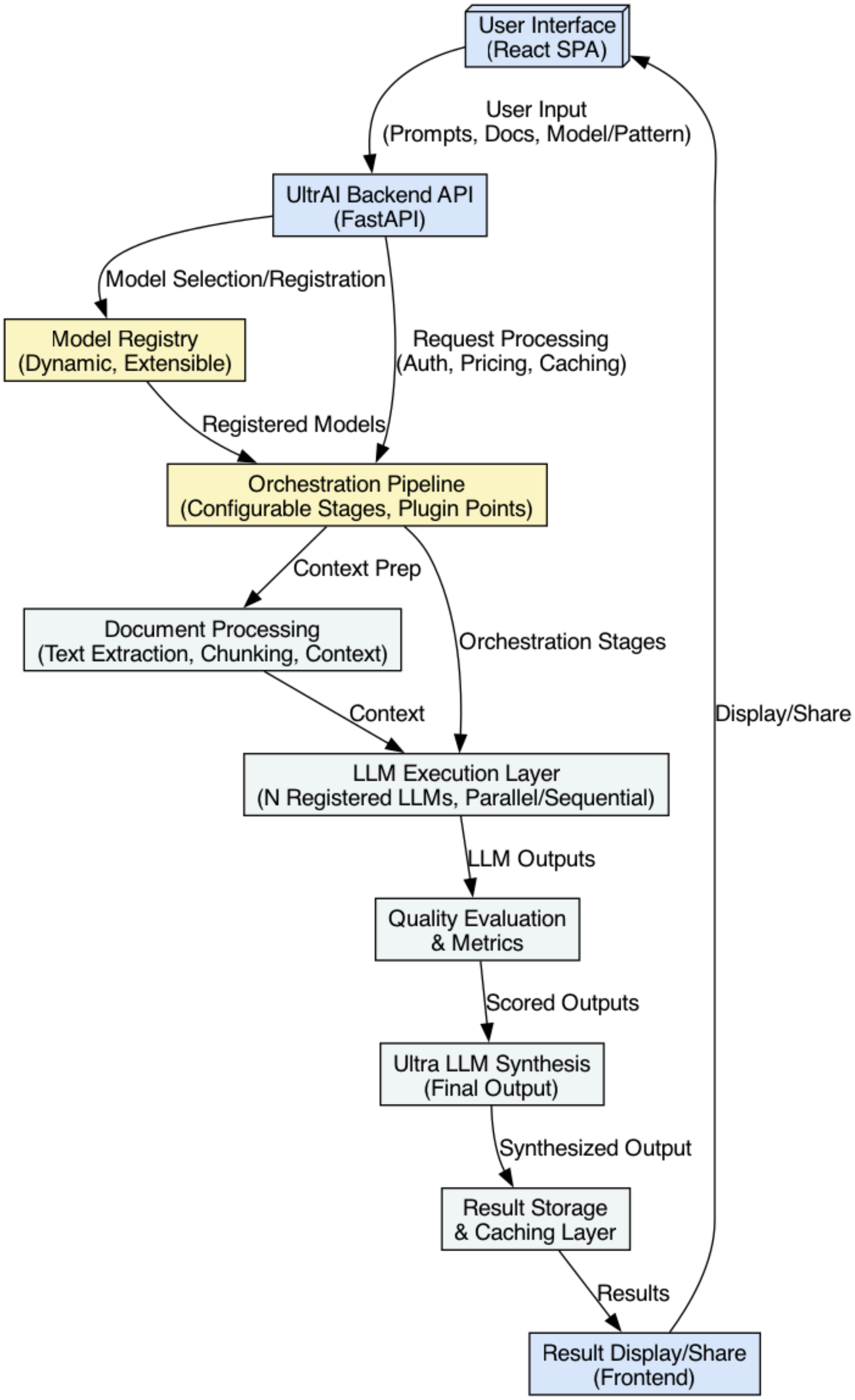
Figure 2: Codebase Structure Diagram (Updated)

Figure 2: Codebase Structure Diagram of the UltraLLMOrchestrator System

Figure 2: Codebase Structure Diagram of the UltraLLMOrchestrator System

Figure 2. The codebase structure diagram highlights the modular, extensible organization of the UltraLLMOrchestrator system, including plugin points for models and orchestration logic,

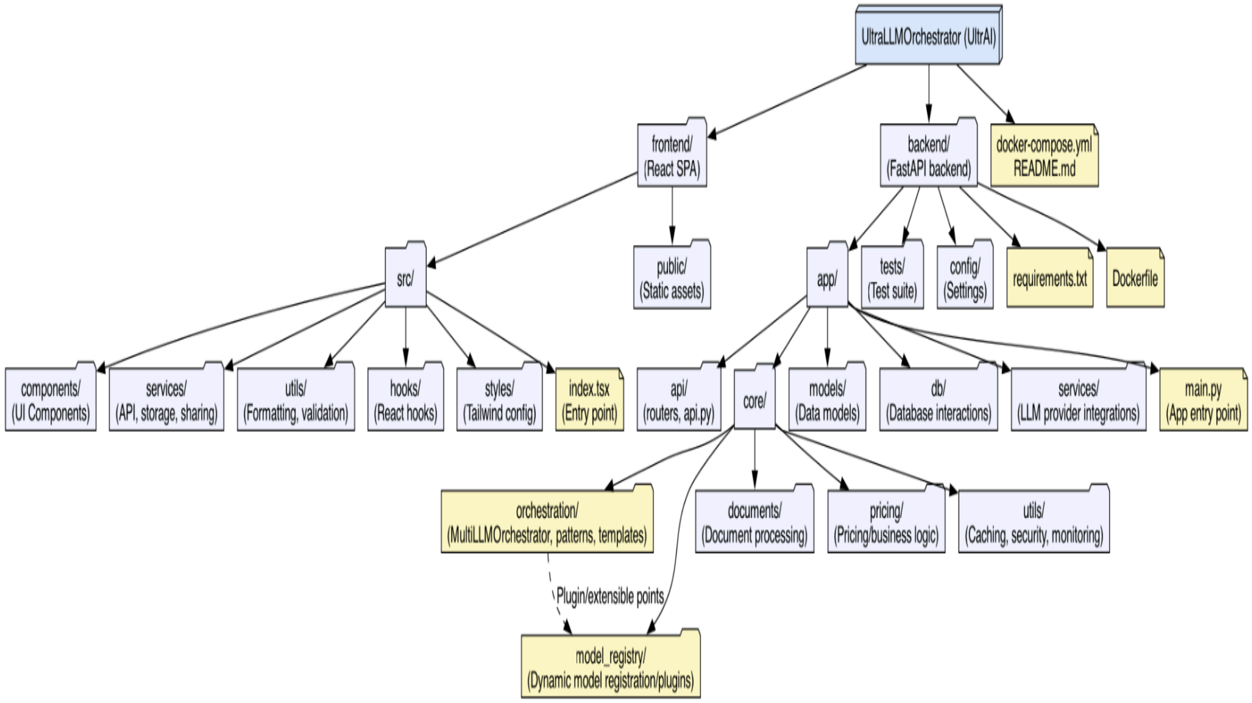
and clear separation of frontend, backend, and core extensibility modules.



file:///Users/joshuafield/Documents/Ultra/legal/figures/ultrai_dfd.png

Abstract

The UltraLLMOrchestrator System (UltrAI) is a robust, extensible AI analysis platform that orchestrates multiple Large Language Models (LLMs) through dynamically configurable, pattern-based pipelines. UltrAI enables runtime registration and orchestration of any number of LLMs, supports extensible analysis patterns, and provides automated, multi-dimensional quality evaluation. The system’s modular architecture, illustrated in the included diagrams, supports rapid integration of new models, patterns, and business logic, making it suitable for both research and commercial deployment. UltrAI delivers enhanced analytical quality, developer-friendly extensibility, and seamless adaptation to advances in AI, setting a new standard for multi-model orchestration and AI-driven analysis.


UltraLLMOrchestrator (UltrAI)
frontend/ (React SPA)
backend/ (FastAPI backend)
docker-compose.yml README.md
src/
public/ (Static assets)
app/
tests/ (Test suite)
config/ (Settings)
requirements.txt
Dockerfile
components/ (UI Components)
services/ (API, storage, sharing)
utils/ (Formatting, validation)
hooks/ (React hooks)
styles/ (Tailwind config)
index.tsx (Entry point)
api/ (routers, api.py)
core/
models/ (Data models)
db/ (Database interactions)
services/ (LLM provider integrations)
main.py (App entry point)
orchestration/ (MultiLLMOrchestrator, patterns, templates)
documents/ (Document processing)
pricing/ (Pricing/business logic)
utils/ (Caching, security, monitoring)
Plugin/extensible points
model_registry/ (Dynamic model registration/plugins)