**BÁO CÁO KIẾN TRÚC TỔNG THỂ: AI SERVICE**

**Dự án:** ITAPIA - Hệ thống AI Hỗ trợ Đầu tư Cá nhân Thông minh và Minh bạch  
**Phiên bản:** 1.0  
**Ngày:** 05/07/2024

**1. Tổng quan và Triết lý Thiết kế**

AI Service là bộ não trung tâm của hệ thống ITAPIA, chịu trách nhiệm thực hiện tất cả các phân tích, dự báo và tạo lập quyết định thông minh. Kiến trúc của AI Service được xây dựng dựa trên các nguyên lý cốt lõi:

* **Tính Minh bạch và Giải thích được (Explainable AI - XAI):** Mọi quyết định và phân tích đều phải có khả năng được giải thích một cách rõ ràng, dễ hiểu, nhằm xây dựng lòng tin và đồng phát triển kiến thức cùng người dùng.
* **Kiến trúc Ensemble Learning:** Hệ thống không phụ thuộc vào một "siêu mô hình" duy nhất. Thay vào đó, quyết định cuối cùng là sự tổng hợp và phối hợp từ nhiều module chuyên gia (Base Models), tương tự nguyên lý Stacking trong Machine Learning.
* **Kiến trúc Phân cấp (Tiered Architecture):** Để tối ưu hóa trải nghiệm người dùng và chi phí vận hành, AI Service cung cấp hai cấp độ xử lý riêng biệt: **Quick Check** (nhanh, đồng bộ, CPU-based) và **Deep Dive** (sâu, bất đồng bộ, GPU-based).
* **Tính Module hóa và Dễ mở rộng:** Các thành phần được thiết kế theo nguyên tắc đơn trách nhiệm và được tổ chức để dễ dàng bảo trì, nâng cấp và mở rộng trong tương lai.

**2. Quy trình Xử lý Hai Cấp độ**

**2.1. Quick Check Process (Phản hồi Nhanh)**

* **Mục đích:** Cung cấp một đánh giá sơ bộ, gần như tức thời (< 3 giây) cho người dùng.
* **Luồng hoạt động:** Duy trì một API endpoint để nhận request trực tiếp từ API Service và xử lý **đồng bộ (synchronous)**. Toàn bộ quy trình được thiết kế để chạy hiệu quả trên hạ tầng CPU.
* **Đầu ra:** Một bản tóm tắt ngắn gọn, bao gồm các chỉ báo kỹ thuật chính, phân tích xu hướng, các mẫu hình đơn giản và một quyết định sơ bộ dựa trên quy tắc.

**2.2. Deep Dive Process (Phân tích Sâu)**

* **Mục đích:** Cung cấp một báo cáo phân tích toàn diện, sâu sắc và đa chiều.
* **Luồng hoạt động:** Xử lý **bất đồng bộ (asynchronous)** thông qua một hệ thống hàng đợi (Message Queue).
  1. Client yêu cầu "Deep Dive" thông qua API Service, ngay lập tức nhận về một job\_id.
  2. API Service đẩy một "job" chứa thông tin cần thiết vào **Job Queue**.
  3. AI Service (hoạt động như một **Worker** trên hạ tầng có GPU) lấy các job từ hàng đợi để xử lý, có thể theo batch để tối ưu hóa việc tải các mô hình lớn (lazy loading).
  4. Sau khi xử lý (mất khoảng 2-5 phút), Worker đẩy kết quả vào **Result Queue**.
  5. Client định kỳ "hỏi thăm" (poll) API Service với job\_id. API Service sẽ kiểm tra Result Queue và trả về kết quả khi có, hoặc thông báo trạng thái đang xử lý.
* **Đầu ra:** Một báo cáo chuyên sâu, bao gồm các giải thích chi tiết được tạo bởi LLM, phân tích tin tức đa chiều, dự báo và một lập luận quyết định hoàn chỉnh.

**3. Phân loại các Thành phần Chức năng**

Các module và chức năng được phân loại dựa trên tốc độ, độ phức tạp và sự tương thích để phục vụ cho hai quy trình trên.

**3.1. Common Components (Thành phần Chung)**  
Đây là nền tảng được sử dụng bởi cả hai quy trình, chủ yếu thuộc về **Technical Module**.

* **FeatureEngine:** Chịu trách nhiệm tính toán hàng loạt các đặc trưng kỹ thuật từ dữ liệu OHLCV. Bao gồm:
  + **Chỉ báo Xu hướng:** SMA, EMA với nhiều chu kỳ.
  + **Chỉ báo Động lượng:** RSI, Stochastic, CCI, Williams %R.
  + **Chỉ báo Biến động:** Bollinger Bands, ATR.
  + **Chỉ báo Khối lượng:** OBV, MFI.
  + **Mẫu hình nến:** Tạo các cột boolean cho hàng trăm mẫu hình nến (cdl\_\*).
  + **Đặc trưng Trễ (Lag Features):** Tạo các cột trễ cho các chỉ báo quan trọng (RSI\_lag1, close\_lag5...) để cung cấp "trí nhớ" cho các mô hình ML.
* **Trend Analysis:** Phân tích các chỉ báo xu hướng (MA, ADX, DMI) để đưa ra nhận định đa chiều về xu hướng (Trung hạn & Dài hạn) và sức mạnh của chúng.
* **Support/Resistance Identifier (v1):** Xác định các mức Hỗ trợ/Kháng cự dựa trên các phương pháp không phụ thuộc vào nhận dạng đỉnh/đáy phức tạp (MA, Bollinger Bands, Pivot Points, Fibonacci đơn giản).
* **Simple Pattern Recognizer:** Chỉ nhận dạng các mẫu hình nến dựa trên các cột đã được FeatureEngine tính sẵn.
* **Forecasting Model (ML-based):** Sử dụng các mô hình Machine Learning có tính giải thích cao như LightGBM hoặc Random Forest, được huấn luyện trên các đặc trưng kỹ thuật (bao gồm cả Lag Features) để đưa ra dự báo. *(Lưu ý: Module này đang trong giai đoạn lên kế hoạch chi tiết, chưa triển khai).*

**3.2. Quick Check Components**

* **Simple News Analysis (News Module):** Sử dụng các mô hình nhẹ như BERT-tiny hoặc các phương pháp dựa trên từ điển (VADER) để gán nhãn cảm xúc (Positive/Negative/Neutral) cho tin tức một cách nhanh chóng.
* **Rule-based Agent (Advisor Module):** Áp dụng một bộ quy tắc (được định nghĩa trước hoặc một phiên bản đơn giản từ Evo Agent) lên kết quả từ các module phân tích nhanh để đưa ra quyết định sơ bộ.
* **Simple Explain Agent (Advisor Module):** Tạo ra các giải thích đơn giản bằng cách chuyển đổi một quy tắc (ví dụ: rule\_RSI\_oversold\_triggered) thành một câu văn mẫu ("Tín hiệu Mua do chỉ báo RSI đang ở vùng quá bán.").

**3.3. Deep Dive Components**

* **Advanced Pattern Recognizer (Technical Module):** Bổ sung khả năng nhận dạng các mẫu hình biểu đồ phức tạp hơn (Hai Đỉnh/Đáy, Vai-Đầu-Vai) dựa trên thuật toán phát hiện đỉnh/đáy (scipy.find\_peaks).
* **LLM News Analysis (News Module):** Sử dụng các Mô hình Ngôn ngữ lớn (LLM) để thực hiện các tác vụ sâu sắc như tóm tắt thông minh, phân tích tác động, và trích xuất thông tin từ tin tức.
* **LLM Decision Maker & Explain Agent (Advisor Module):**
  + Sử dụng LLM nền tảng (ví dụ: FinGPT) được tối ưu hóa bằng kỹ thuật Lazy Loading Lora Adapter để tổng hợp tất cả các tín hiệu (kỹ thuật, tin tức, rủi ro) và tạo ra một lập luận quyết định toàn diện.
  + Có thể tận dụng các mô hình mạnh hơn như Gemini hoặc GPT-4 thông qua API để chưng cất tri thức, làm mượt và tăng cường chất lượng của các giải thích cuối cùng.

**4. Các Module Hỗ trợ Chuyên biệt**

**4.1. Evolutionary Module (Evo Agent)**  
Đây là một module cốt lõi, hoạt động độc lập và thường xuyên ở chế độ nền (background).

* **Mục tiêu:** Tự động khám phá và tối ưu hóa các chiến lược/quy tắc giao dịch hiệu quả.
* **Kiến trúc:** Triển khai theo mô hình **Đồng tiến hóa (Co-evolution)**:
  + **Quần thể 1 (Cấu trúc Rule):** Sử dụng **Genetic Programming (GP)** để tiến hóa cấu trúc logic của các quy tắc.
  + **Quần thể 2 (Tham số Rule):** Sử dụng các thuật toán tối ưu hóa số thực hiệu quả (như **CMA-ES, DE**) để tinh chỉnh các hằng số/ngưỡng giá trị trong các quy tắc đó.
  + **Cơ chế Tương tác:** Hai quần thể tương hỗ lẫn nhau. Một đề xuất là sử dụng mô hình tối ưu hóa lồng nhau, nơi mỗi cấu trúc rule mới sẽ trải qua một vòng tiến hóa tham số nhỏ để đánh giá tiềm năng tốt nhất của nó.
* **Hàm Fitness:** Một hàm tổng hợp có trọng số thích ứng (adaptive weights), có thể điều chỉnh dựa trên hồ sơ rủi ro của người dùng và điều kiện thị trường, ưu tiên các chỉ số như Sharpe Ratio, Max Drawdown, và Lợi nhuận.
* **Đầu ra:** Các bộ quy tắc được tối ưu hóa cho từng nhóm ngành (ví dụ: 1 bộ quy tắc chung và 12 biến thể đã được tinh chỉnh tham số cho 12 nhóm ngành).

**4.2. Simulation Module**  
Là một module dịch vụ, đóng vai trò là "phòng thí nghiệm" để kiểm thử các chiến lược.

* **Chức năng:** Nhận một bộ quy tắc và dữ liệu lịch sử, thực hiện backtest và trả về một bộ các chỉ số hiệu suất chi tiết. Đây là thành phần không thể thiếu để cung cấp dữ liệu cho Fitness Function của Evo Agent.

**Trạng thái Hiện tại và Hướng phát triển tiếp theo:**

Hệ thống hiện đã hoàn thiện việc thiết kế kiến trúc tổng thể. Các module cốt lõi trong **Technical Module (Common Components)** và luồng xử lý **Quick Check** đang được ưu tiên phát triển. Các thành phần "Deep Dive" và các module chuyên biệt như **Evolutionary** và **Simulation** là các bước tiếp theo trong lộ trình, với **Forecasting Model** hiện mới chỉ dừng lại ở giai đoạn lên kế hoạch chi tiết. Các module khác đang trong quá trình phát triển và sẽ được tích hợp trong các phiên bản tương lai.