ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

----****----

KỸ THUẬT VÀ CÔNG NGHỆ DỮ LIỆU LỚN CHO TRÍ TUỆ NHÂN TẠO

BÁO CÁO

CHỦ ĐỀ: REAL-TIME CRYPTO ANALYSIS AND PREDICTION

Giảng viên hướng dẫn: TS. Trần Hồng Việt
ThS. Ngô Minh Hương

Sinh viên thực hiện: Bùi Thanh Dân - 23020342
Phạm Tiến Dũng - 23020345
Vũ Nguyên Đan - 23020351

Hà Nội, 2025

NHIỆM VỤ CỦA CÁC THÀNH VIÊN	
Họ và tên	**Công việc**
Bùi Thanh Dân	**- Thiết kế và xây dựng kiến trúc tổng thể của hệ thống (hạ tầng Big Data, pipeline xử lý, triển khai Docker).** **- Xây dựng base code cho dự án**: Kafka ingestion, Spark Streaming + Batch APIs, HDFS/HBase integration. - Thiết kế luồng dữ liệu (dataflow), tối ưu hiệu năng pipeline. - Tích hợp Apache Airflow vào hệ thống, DAG xử lý dữ liệu định kỳ. - QA toàn bộ dự án: kiểm thử batch + stream layer. - Viết báo cáo chương 2, làm slide, trình bày.
Phạm Tiến Dũng	**- Phát triển hệ thống AI Chatbot (Gemini LLM + NewsAPI + web scraping).** - Thiết kế logic NLP: nhận diện intent, phân tích ngôn ngữ, sentiment và trả lời bằng LLM. - Nâng cấp **Real-time Dashboard** (Flask + JavaScript UI), cải thiện UX/UI. - Tối ưu stream layer: xử lý real-time bằng Spark Streaming, tinh chỉnh ghi dữ liệu vào HBase. - Testing, refactor mã nguồn backend + chatbot. - Viết báo cáo chương 3, làm slide, trình bày.
Vũ Nguyên Đan	**- Xây dựng và tối ưu Batch Layer**: xử lý dữ liệu lịch sử, tính toán hơn 40+ chỉ báo kỹ thuật (indicators). **- Huấn luyện, đánh giá và tối ưu mô hình XGBoost** (1h + 1d forecasting cho 10 đồng crypto). - Thiết kế đầu ra mô hình: prediction, confidence, volatility, momentum, technical signals. - **Phát triển Streamlit Dashboard nâng cao**: KPIs, correlation heatmap, portfolio analytics, volatility/risk metrics. - Viết báo cáo chương 1 và phụ lục, viết README, tham gia trình bày.

LỜI MỞ ĐẦU

Sự phát triển mạnh mẽ của công nghệ chuỗi khối đã biến thị trường tiền mã hóa thành một lĩnh vực tài chính năng động, thu hút hàng triệu nhà đầu tư toàn cầu. Thị trường này tạo ra lượng dữ liệu khổng lồ mang đặc trưng của Dữ liệu lớn (Big Data) - Volume, Velocity, Variety, Veracity và Value. Tuy nhiên, tính biến động cao và tốc độ phát sinh dữ liệu nhanh đặt ra thách thức lớn: làm sao khai thác và phân tích nguồn dữ liệu này một cách hiệu quả, kịp thời để hỗ trợ ra quyết định.

Trước bối cảnh đó, đề tài "Real-time crypto analysis and prediction" được thực hiện nhằm thiết kế và triển khai một hệ thống có khả năng xử lý dữ liệu theo cả hai hướng: thời gian thực và theo lô lịch sử. Cụ thể, nhóm tập trung xây dựng pipeline thu thập dữ liệu hiệu suất cao; triển khai Lớp xử lý luồng (Stream Layer) cho dự đoán tức thời và Lớp xử lý hàng loạt (Batch Layer) cho phân tích sâu, huấn luyện mô hình; đồng thời tích hợp trợ lý ảo sử dụng Mô hình Ngôn ngữ Lớn (LLM) để cung cấp giao diện tương tác trực quan, giàu thông tin.

Việc lựa chọn đề tài xuất phát từ nhu cầu thực tiễn khi các phương pháp truyền thống không đáp ứng được tốc độ và quy mô dữ liệu tiền mã hóa. Kiến trúc Lambda, với khả năng kết hợp giữa luồng xử lý nóng (hot-path) và lạnh (cold-path), là giải pháp phù hợp. Cùng với sự phát triển của trí tuệ nhân tạo, đặc biệt là các LLM, người dùng có thể không chỉ quan sát mà còn "trò chuyện" với dữ liệu, đặt câu hỏi và nhận phân tích tổng hợp từ nhiều nguồn, nâng cao hiệu quả ra quyết định.

Để đảm bảo tính khả thi, đồ án tập trung vào dữ liệu giá (OHLCV) và tin tức của một số đồng tiền phổ biến. Hệ thống được xây dựng trên nền tảng mã nguồn mở gồm Apache Kafka, Spark, HDFS, HBase, MongoDB và triển khai bằng Docker. Đối tượng nghiên cứu không chỉ là kết quả phân tích mà còn là hiệu năng và khả năng phối hợp giữa các thành phần hệ thống trong việc giải quyết bài toán thực tế.

Nhóm áp dụng quy trình nghiên cứu có hệ thống: từ tìm hiểu lý thuyết về kiến trúc Lambda và công nghệ liên quan, đến thiết kế chi tiết luồng dữ liệu, triển khai theo từng module bằng Docker Compose để đảm bảo tính nhất quán. Cuối cùng, hệ thống được kiểm thử toàn diện nhằm đánh giá hiệu năng, độ trễ và chất lượng kết quả đầu ra.

Báo cáo gồm ba chương chính: Chương 1 trình bày tổng quan, thách thức và cơ sở lý thuyết; Chương 2 mô tả chi tiết kiến trúc, công nghệ và pipeline xử lý dữ liệu; Chương 3 tổng kết kết quả, đánh giá hạn chế và đề xuất hướng phát triển tương lai.

Mục lục

Chương 1

TỔNG QUAN

1.1 Tổng quan lĩnh vực nghiên cứu

1.1.1 Dữ liệu trong thị trường tiền mã hóa

Thị trường tiền mã hóa (cryptocurrency) là một trong những lĩnh vực có tốc độ biến động mạnh nhất hiện nay, nơi mà mỗi thay đổi về giá, khối lượng hay trạng thái lệnh đều tạo ra một dòng dữ liệu thời gian thực với tần suất rất cao. Mỗi giao dịch được ghi nhận đều góp phần hình thành các chuỗi dữ liệu liên tục, phản ánh tức thời trạng thái của thị trường. Đây là một đặc trưng quan trọng của các hệ thống tài chính phi tập trung và thị trường giao dịch tốc độ cao.
Các nguồn dữ liệu phổ biến trong thị trường tiền mã hóa bao gồm:

- **Dữ liệu giao dịch (trade data):** Bao gồm giá khớp lệnh, khối lượng giao dịch và thời điểm thực hiện giao dịch. Đây là loại dữ liệu được cập nhật theo từng mili-giây, thể hiện trực tiếp hoạt động mua bán của thị trường.
- **Dữ liệu sổ lệnh (order book):** Thể hiện mức giá bid/ask, độ sâu thị trường và thanh khoản. Sổ lệnh là nguồn dữ liệu có tốc độ thay đổi rất nhanh, được sử dụng rộng rãi trong các mô hình giao dịch tần suất cao (HFT).
- **Dữ liệu OHLCV:** Bao gồm giá mở cửa, đóng cửa, cao nhất, thấp nhất và khối lượng giao dịch theo chu kỳ (1 phút, 1 giờ, 1 ngày,...). Đây là dạng dữ liệu được tiêu chuẩn hóa, hỗ trợ tốt cho phân tích kỹ thuật và mô hình dự đoán.
- **Dữ liệu phi cấu trúc:** Tin tức, mạng xã hội, thông báo từ các sàn giao dịch hoặc dự án blockchain. Loại dữ liệu này thường mang đặc trưng phi cấu trúc (unstructured) và được sử dụng để phân tích cảm xúc (sentiment) hoặc phát hiện sự kiện.

Đặc điểm chung của dữ liệu thị trường crypto là **khối lượng lớn**, **tốc độ cao** và **tính biến động liên tục**. Điều này khiến việc thu thập, lưu trữ và phân tích dữ liệu trở thành một thách thức kỹ thuật đáng kể, yêu cầu các hệ thống phải vừa đảm bảo khả năng mở rộng, vừa duy trì độ trễ thấp trong xử lý.

1.1.2 Thách thức trong phân tích thời gian thực

Phân tích thời gian thực trong lĩnh vực tiền mã hóa đối mặt với nhiều thách thức quan trọng, đòi hỏi hệ thống phải đảm bảo tốc độ xử lý cao, khả năng mở rộng và mức độ ổn định trước biến động mạnh của thị trường.

- **Tốc độ dữ liệu cao (high-velocity):** Các sàn giao dịch có thể phát sinh hàng nghìn sự kiện mỗi giây. Điều này yêu cầu hệ thống phải xử lý liên tục với độ trễ (latency) cực thấp để đảm bảo thông tin kịp thời cho các tác vụ phân tích và dự đoán.
- **Tính không ổn định của thị trường:** Giá crypto biến động mạnh trong thời gian rất ngắn. Mô hình dự đoán dễ bị nhiễu nếu dữ liệu không được làm sạch, chuẩn hoá và đồng bộ đúng cách.
- **Đa dạng nguồn dữ liệu:** Việc kết hợp giữa dữ liệu định lượng như *price, volume* và dữ liệu phi cấu trúc như *news, sentiment* gây khó khăn cho quá trình làm giàu dữ liệu và trích xuất đặc trưng (feature extraction).
- **Yêu cầu về tính mở rộng (scalability):** Hệ thống cần khả năng mở rộng theo chiều ngang để đáp ứng lượng dữ liệu liên tục tăng theo thời gian, đặc biệt trong các sự kiện thị trường lớn (ví dụ: tăng giảm đột biến).
- **Giới hạn của mô hình truyền thống:** Các mô hình thống kê cổ điển thường không theo kịp động học phức tạp của thị trường crypto, từ đó đặt ra nhu cầu áp dụng các mô hình học máy và học sâu (ML/DL) để cải thiện độ chính xác.

Những thách thức trên là cơ sở thúc đẩy sự phát triển của các kiến trúc xử lý dữ liệu hiện đại như **Lambda Architecture**, đồng thời khuyến khích ứng dụng các phương pháp ML/DL/AI nhằm nâng cao độ chính xác và tính thích ứng trong phân tích thị trường crypto.

1.2 Cơ sở lý thuyết

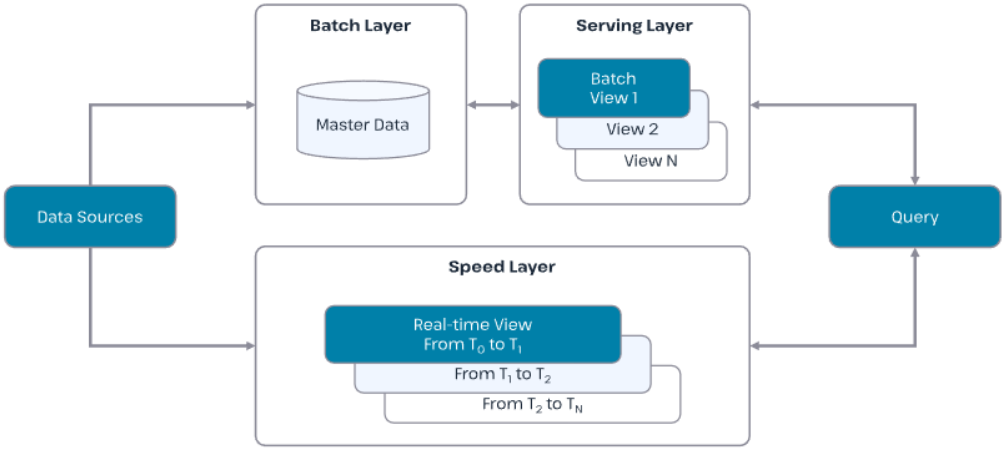
1.2.1 Dữ liệu lớn và kiến trúc Lambda

Dữ liệu lớn (Big Data) thường được mô tả bởi ba đặc trưng cốt lõi: **Volume** (khối lượng dữ liệu rất lớn), **Velocity** (tốc độ sinh dữ liệu cao) và **Variety** (đa dạng loại dữ liệu). Trong bối cảnh thị trường tiền mã hóa, cả ba đặc trưng này đều xuất hiện rõ rệt. Đặc biệt, yếu tố **Velocity** được thể hiện mạnh mẽ khi dữ liệu giá, khối lượng giao dịch và các sự kiện thị trường được cập nhật liên tục theo từng mili-giây.

Để xử lý hiệu quả các luồng dữ liệu có tốc độ lớn như vậy, kiến trúc **Lambda Architecture** được áp dụng. Đây là mô hình xử lý dữ liệu kết hợp ưu điểm của hai hướng tiếp cận khác nhau nhằm đảm bảo vừa có độ chính xác cao, vừa đáp ứng yêu cầu thời gian thực.

- **Batch Layer:** xử lý dữ liệu lịch sử với quy mô lớn. Lớp này có độ trễ cao nhưng mang lại độ tin cậy và độ chính xác cao cho các phép tính tổng hợp hoặc mô hình phân tích.
- **Speed Layer (Streaming Layer):** xử lý dữ liệu ngay khi phát sinh nhằm tạo ra kết quả tức thì, bù đắp độ trễ vốn có của batch layer.
- **Serving Layer:** lưu trữ, tổng hợp và cung cấp dữ liệu đầu ra cho ứng dụng, mô hình hoặc giao diện người dùng.

Nhờ sự kết hợp này, Lambda Architecture cho phép hệ thống đồng thời đảm bảo tính **chính xác** (từ batch layer) và **tính kịp thời** (từ speed layer), rất phù hợp cho các bài toán phân tích và dự đoán trong thị trường tài chính thời gian thực như crypto.



Hình 1.1: Mô tả tổng quát kiến trúc Lambda.

1.2.2 Đặc trưng của Big Data và xử lý song song

Các hệ thống Big Data thường dựa trên phương pháp **xử lý song song phân tán**, cho phép chia nhỏ dữ liệu và phân phối khối lượng tính toán qua nhiều nút (nodes) trong cụm máy chủ. Cách tiếp cận này giúp tối ưu tốc độ xử lý, tăng khả năng chịu lỗi và đảm bảo hệ thống có thể mở rộng linh hoạt theo nhu cầu. Một số đặc trưng quan trọng bao gồm:

- **Parallel Processing:** các framework như Hadoop và Apache Spark cho phép xử lý dữ liệu dưới dạng nhiều tác vụ nhỏ, hoạt động song song nhằm tăng tốc độ xử lý các tập dữ liệu lớn.
- **Fault Tolerance:** hệ thống có khả năng chịu lỗi thông qua cơ chế sao lưu, nhân bản dữ liệu hoặc lập lại tác vụ, giúp toàn bộ pipeline tiếp tục hoạt động ngay cả khi một số node gặp sự cố.
- **Distributed Storage:** dữ liệu được lưu trữ và quản lý theo dạng phân tán, tiêu biểu như HDFS, đảm bảo nâng cao khả năng mở rộng cũng như độ tin cậy trong môi trường dữ liệu lớn.
- **Scalability:** hệ thống dễ dàng mở rộng quy mô bằng cách bổ sung thêm tài nguyên tính toán hoặc lưu trữ mà không cần thay đổi cấu trúc ứng dụng hiện có.

Trong bối cảnh dự án này, Apache Spark giữ vai trò trung tâm trong việc xử lý song song cho cả hai loại dữ liệu: **dữ liệu streaming** đến từ Kafka và **dữ liệu batch** thu thập từ Yahoo Finance. Việc kết hợp hai hướng xử lý này giúp hệ thống đảm bảo vừa phản hồi nhanh theo thời gian thực, vừa phân tích sâu dữ liệu lịch sử với hiệu năng cao.

1.2.3 Ứng dụng AI trong phân tích tài chính

Trí tuệ nhân tạo (AI) và học máy (Machine Learning) đã trở thành những công nghệ chủ chốt trong phân tích tài chính hiện đại. Với khả năng mô hình hóa các quan hệ phi tuyến và khám phá các mẫu ẩn trong dữ liệu nhiễu, các mô hình AI thường đạt hiệu quả dự báo vượt trội so với các phương pháp thống kê truyền thống. Một số hướng nghiên cứu tiêu biểu bao gồm:

- **Dự báo chuỗi thời gian (time-series forecasting):** Các mô hình như mạng nơ-ron hồi tiếp (RNN, LSTM) và các thuật toán ensemble (Random Forest, XGBoost) đã chứng

minh hiệu quả cao trong dự báo giá và biến động thị trường, đặc biệt đối với các chuỗi dữ liệu phi tuyến tính [1, 2].

- **Phân tích cảm xúc (sentiment analysis):** Việc khai thác tin tức, mạng xã hội, diễn đàn và hành vi người dùng giúp đánh giá tâm lý thị trường. Nghiên cứu cho thấy các chỉ số sentiment có mối tương quan đáng kể với sự biến động giá trong nhiều loại tài sản tài chính, bao gồm cả crypto [3].
- **Phát hiện bất thường và rủi ro:** Các mô hình unsupervised như Isolation Forest, Autoencoder hay clustering được sử dụng để phát hiện giao dịch bất thường, các cú sốc thị trường, hoặc dấu hiệu thao túng giá.
- **Tối ưu hóa danh mục đầu tư:** Các thuật toán heuristic và reinforcement learning ngày càng được ứng dụng rộng rãi để hỗ trợ phân bổ tài sản tối ưu và quản trị rủi ro dưới điều kiện thị trường biến động nhanh.

Tổng quan lại, AI đóng vai trò quan trọng trong việc nâng cao hiệu quả dự báo, cải thiện khả năng nhận diện rủi ro và khai thác giá trị từ dữ liệu lớn trong lĩnh vực tài chính hiện đại.

1.2.4 Tích hợp dữ liệu giá và tin tức trong phân tích crypto

Trong thị trường tiền mã hóa, việc kết hợp dữ liệu giá dạng chuỗi thời gian (structured time series) với dữ liệu tin tức và sentiment (unstructured text) được xem là một hướng tiếp cận toàn diện và mang lại hiệu quả cao. Dữ liệu giá phản ánh trạng thái thị trường tại từng thời điểm, trong khi tin tức cung cấp bối cảnh định tính về tâm lý cộng đồng, chính sách, sự kiện và các yếu tố có thể tác động mạnh tới biến động giá.

Một số nghiên cứu nổi bật chỉ ra rằng:

- **Tin tức ảnh hưởng mạnh đến biến động ngắn hạn**, đặc biệt trong thị trường crypto vốn nhạy cảm với tâm lý nhà đầu tư và thay đổi liên tục [4].
- **Mô hình lai (hybrid)** kết hợp giữa dữ liệu giá và dữ liệu sentiment thường cho kết quả dự báo chính xác hơn so với các mô hình chỉ dựa trên dữ liệu định lượng [5].
- **Dữ liệu phi truyền thống (alternative data)**, bao gồm tin tức, mạng xã hội và mức độ tương tác cộng đồng, mang lại tín hiệu bổ trợ quan trọng giúp mô hình hiểu rõ hơn về động lực thị trường [6].

Quy trình tích hợp hai loại dữ liệu thường bao gồm các bước:

1. Thu thập dữ liệu giá từ sàn giao dịch;
2. Thu thập tin tức từ API hoặc thông qua web crawling;
3. Tiền xử lý và trích xuất đặc trưng văn bản bằng các kỹ thuật NLP (BERT, TF-IDF, LSTM embedding);
4. Đồng bộ hai nguồn dữ liệu theo mốc thời gian;
5. Xây dựng mô hình dự báo hoặc mô hình đánh giá rủi ro.

Cách tiếp cận này giúp mô hình không chỉ phản ánh biến động giá thuần túy, mà còn hiểu sâu hơn về nguyên nhân của những biến động đó, từ đó cải thiện độ chính xác và độ tin cậy của dự báo.

Chương 2

HỆ THỐNG VÀ CÔNG NGHỆ TRIỂN KHAI

2.1 Kiến trúc tổng thể hệ thống và luồng dữ liệu

Hệ thống được thiết kế theo **Lambda Architecture**, nhằm tận dụng sức mạnh của cả hai hướng xử lý dữ liệu: *real-time streaming* và *batch processing*. Mô hình này giúp hệ thống vừa phản hồi nhanh với dữ liệu thị trường tức thời, vừa có khả năng tổng hợp và phân tích sâu dữ liệu lịch sử, đồng thời tích hợp thêm phân tích tin tức từ mô hình ngôn ngữ lớn (LLM).
Luồng dữ liệu của hệ thống bao gồm bốn lớp chính:

- **Ingestion Layer:**

  - Thu thập dữ liệu giá và khối lượng từ *Binance WebSocket API* (real-time) và dữ liệu lịch sử từ *Yahoo Finance API*.

  - Dữ liệu được gửi vào *Apache Kafka*, nơi đóng vai trò hàng đợi trung gian (message broker), giúp tách biệt nguồn dữ liệu với các tầng xử lý sau.

- **Stream Layer:**

  - Mô hình *XGBoost* (được huấn luyện trước đó trong batch layer) được nạp vào để tạo ra **dự đoán ngắn hạn** cho từng đồng crypto.

  - Kết quả xử lý được ghi vào *Apache HBase* để phục vụ truy vấn nhanh và hiển thị qua API Flask.

- **Batch Layer:**

  - PySpark định kỳ đọc dữ liệu lịch sử từ *HDFS*, tiến hành làm sạch, tạo đặc trưng (feature engineering), và huấn luyện lại mô hình *XGBoost*.

  - Sau khi huấn luyện, mô hình mới được lưu lại trên *HDFS* và đồng bộ tới Stream Layer để phục vụ dự đoán thời gian thực.

  - Dữ liệu tổng hợp, thống kê và phân tích dài hạn được lưu trong *MongoDB*, phục vụ hiển thị qua Streamlit Dashboard.

- **Serving Layer:**

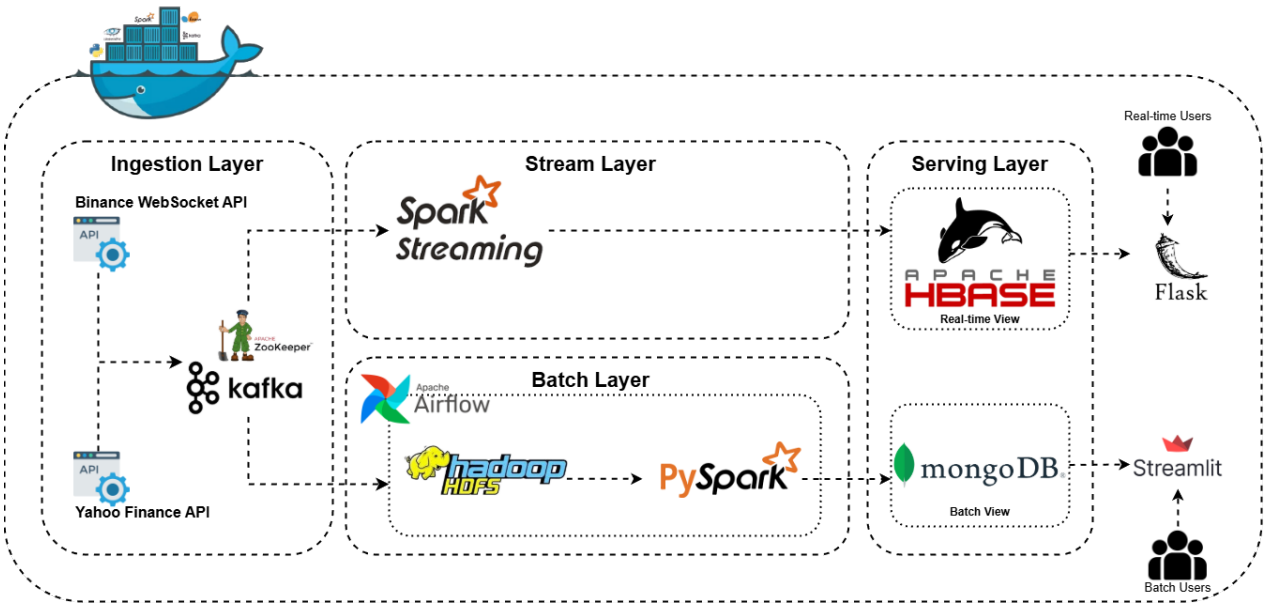
  - *Apache HBase* phục vụ các yêu cầu truy xuất nhanh từ lớp real-time thông qua Flask API.

  - *MongoDB* lưu các kết quả batch và thống kê dài hạn, được hiển thị thông qua Streamlit UI.

- **Addition**

  - **LLM (Large Language Model)** nhận câu hỏi từ người dùng và dữ liệu tin tức từ *NewsAPI* thông qua Flask, thực hiện tóm tắt, phân tích cảm xúc (sentiment) và đánh giá tác động lên thị trường.

  - Kết quả phân tích được kết hợp với dữ liệu giá để hiển thị tại Flask UI, nơi người dùng có thể tương tác trực tiếp qua chatbot AI.



Hình 2.1: Kiến trúc tổng thể hệ thống (Lambda Architecture).

Tóm tắt: Luồng dữ liệu chính đi từ các API đầu vào (Binance WebSocket API, Yahoo Finance API) → Kafka, sau đó được xử lý song song bởi **Stream Layer** và **Batch Layer**. Kết quả được lưu trữ vào các cơ sở dữ liệu tương ứng: Apache HBase cho *stream* và MongoDB cho *batch*. Ngoài ra, Apache Airflow giúp tự động hóa quá trình thu thập dữ liệu lịch sử (Yahoo Finance) và huấn luyện mô hình XGBoost thường xuyên. Dữ liệu sau đó hiển thị tại hai giao diện người dùng. Trong đó, **Flask API** đảm nhiệm cung cấp dữ liệu cho *real-time users* (từ HBase) và tích hợp với LLM cùng NewsAPI. Dữ liệu tổng hợp từ *batch layer* (MongoDB) được cung cấp cho *batch users* thông qua dashboard.

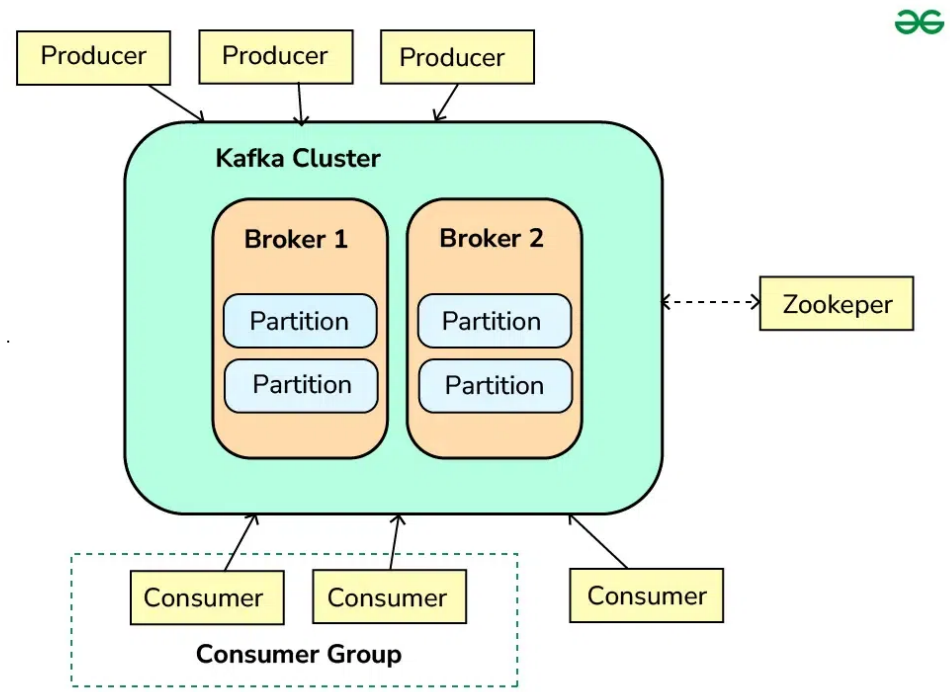
2.2 Công nghệ sử dụng

Việc lựa chọn công nghệ sử dụng là yếu tố quyết định sự thành công và khả năng mở rộng của hệ thống. Phần này tập trung trình bày các nhóm công nghệ cốt lõi, được phân chia thành ba trụ cột chính: đầu tiên là nền tảng dữ liệu lớn dùng để thu thập và lưu trữ, thứ hai là các công cụ phân tích, ứng dụng và giao diện phục vụ người dùng cuối, và cuối cùng là công nghệ triển khai nhằm đảm bảo tính vận hành hiệu quả và ổn định.

2.2.1 Nền tảng dữ liệu lớn

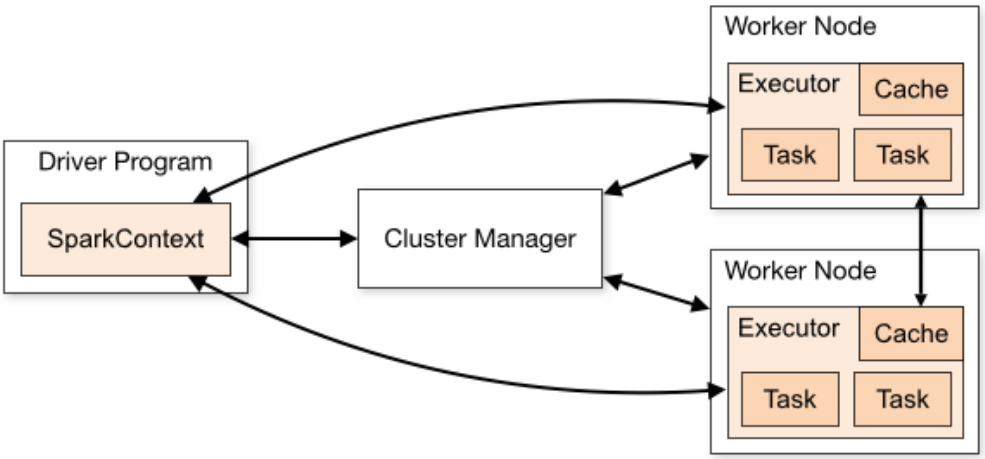
Hệ thống được xây dựng trên nền tảng xử lý dữ liệu lớn (Big Data Platform) nhằm đảm bảo khả năng thu thập, xử lý và lưu trữ khối lượng dữ liệu lớn theo thời gian thực.
Các công nghệ chính bao gồm:

Apache Kafka: là hệ thống hàng đợi thông điệp (message queue) đóng vai trò trung gian truyền dữ liệu giữa các thành phần. Kafka giúp tiếp nhận luồng dữ liệu từ *Binance WebSocket API* và phân phối đến các dịch vụ xử lý (Spark Streaming, Flask API). Việc sử dụng Kafka giúp đảm bảo độ tin cậy, giảm độ trễ và dễ mở rộng trong quy mô hệ thống.



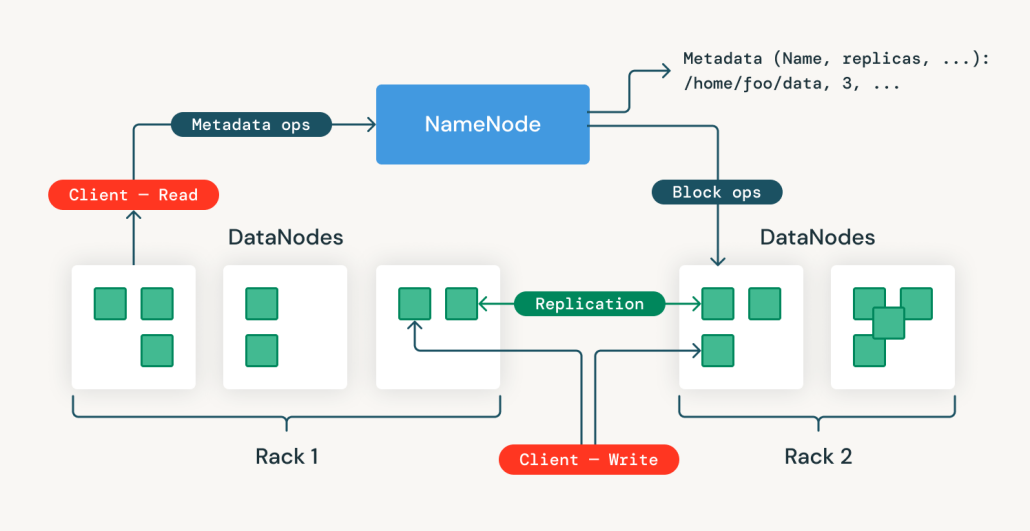
Hình 2.2: Kiến trúc cơ bản của Apache Kafka.

Apache Spark: là công cụ xử lý dữ liệu phân tán, được sử dụng trong hệ thống cho cả *batch layer* và *stream layer*. Spark Streaming xử lý luồng dữ liệu giá crypto thời gian thực, đồng thời Spark batch được dùng để huấn luyện mô hình XGBoost trên dữ liệu lịch sử. Spark giúp tối ưu tốc độ xử lý và dễ dàng tích hợp với HDFS và HBase.



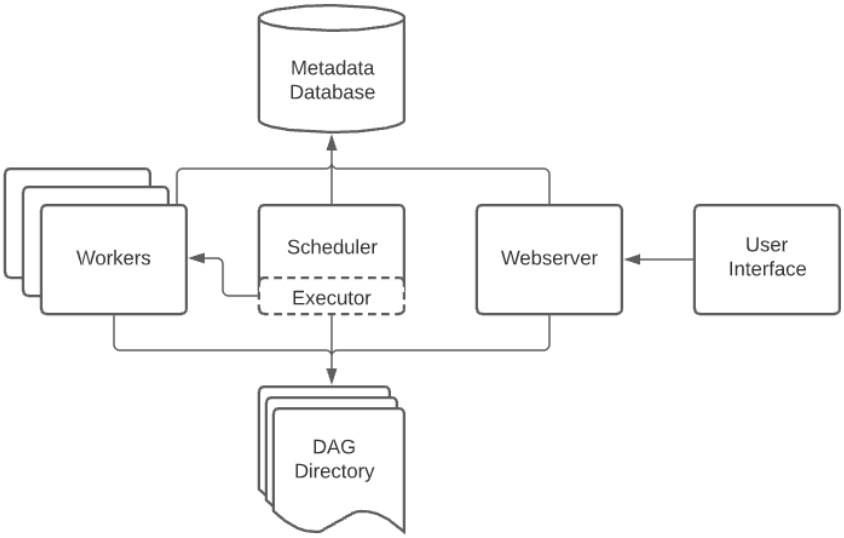
Hình 2.3: Luồng xử lý dữ liệu bằng Apache Spark.

Hadoop Distributed File System (HDFS): được sử dụng để lưu trữ dữ liệu lịch sử crypto thu thập từ Yahoo Finance. HDFS đảm bảo khả năng mở rộng, an toàn dữ liệu và hỗ trợ Spark đọc/ghi dữ liệu hiệu quả cho quá trình huấn luyện mô hình.



Hình 2.4: Kiến trúc cơ bản của HDFS.

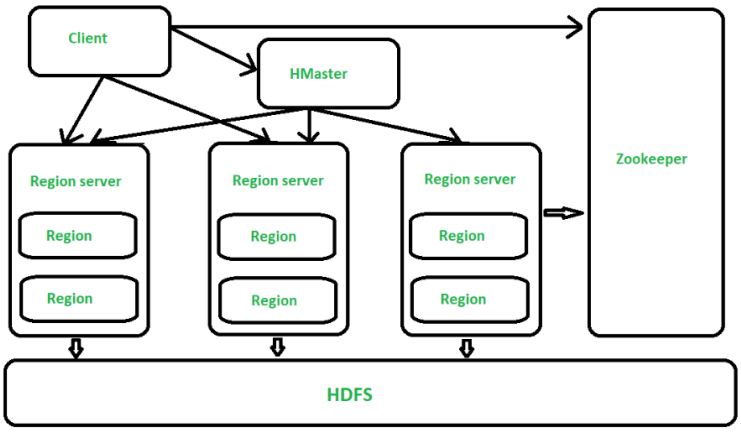
Apache Airflow: là một nền tảng mã nguồn mở dùng để lập lịch, quản lý và giám sát các workflow dưới dạng DAG (Directed Acyclic Graph). Airflow tự động hóa quy trình xử lý dữ liệu và theo dõi trạng thái từng bước qua giao diện web trực quan. Trong dự án này Airflow giúp tự động hóa quy trình lưu trữ dữ liệu lịch sử (HDFS và MongoDB) và huấn luyện mô hình dự đoán.



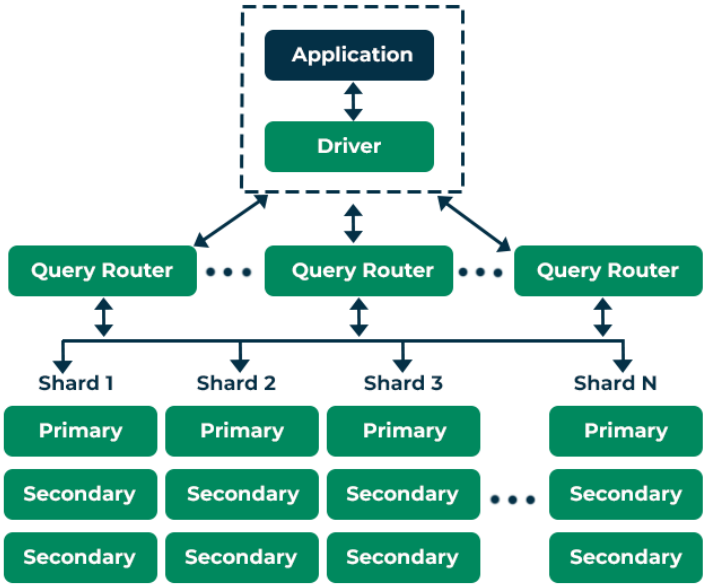
Hình 2.5: Kiến trúc cơ bản của Airflow.

Apache HBase: là cơ sở dữ liệu phi quan hệ (NoSQL) được tối ưu cho truy cập ngẫu nhiên và tốc độ cao. Trong hệ thống, HBase được dùng làm nơi lưu trữ dữ liệu real-time từ Spark Streaming, phục vụ nhanh cho các API truy vấn và hiển thị dashboard.

MongoDB: được dùng làm lớp lưu trữ trong *batch layer*, chứa dữ liệu thống kê, dự đoán dài hạn và các bản ghi đã xử lý. MongoDB giúp linh hoạt trong cấu trúc dữ liệu và hỗ trợ tốt việc hiển thị dữ liệu tổng hợp qua Streamlit.



Hình 2.6: Kiến trúc cơ bản của Apache HBase.



Hình 2.7: Kiến trúc của MongoDB.

2.2.2 Phân tích, ứng dụng và giao diện

1. **LLM (Large Language Model):** được tích hợp vào hệ thống để phân tích tin tức và bài báo liên quan đến thị trường crypto. Mô hình LLM giúp trích xuất *sentiment*, đánh giá xu hướng biến động thị trường từ nguồn tin tức và cung cấp phản hồi tự nhiên cho chatbot tư vấn hỗ trợ. Hệ thống sử dụng API từ *Google AI Studio* để triển khai chức năng này.

2. **NewsAPI:** là nguồn dữ liệu tin tức toàn cầu, được sử dụng để thu thập các bài báo liên quan đến crypto và tài chính. Các tin tức này được xử lý trước khi gửi đến mô hình LLM để phân tích ngữ nghĩa và tóm tắt nội dung.

3. **Flask:** là framework web Python nhẹ, được sử dụng để xây dựng các REST API và giao diện hiển thị real-time cho hệ thống. Flask đảm nhiệm vai trò trung gian giữa các dịch vụ xử lý dữ liệu, giúp nhận yêu cầu từ người dùng hoặc dashboard, truy xuất dữ liệu từ

HBase và phản hồi tức thì các kết quả như giá hiện tại, dự đoán ngắn hạn, hoặc phân tích tin tức và đưa ra những nhận định thị trường trực tiếp từ mô hình *LLM*.

4. **Streamlit:** là nền tảng giao diện phân tích và hiển thị dữ liệu thuộc *batch layer*. Ứng dụng Streamlit được thiết kế để trực quan hóa các dữ liệu thống kê, biểu đồ tổng hợp, kết quả dự đoán dài hạn, và các chỉ số được lưu trong *MongoDB*. Giao diện này phục vụ cho việc đánh giá xu hướng tổng thể của thị trường, đối lập với giao diện Flask - nơi tập trung vào dữ liệu thời gian thực.

2.2.3 Công nghệ triển khai

Hệ thống được triển khai hoàn toàn bằng **Docker Compose**, cho phép mô phỏng và vận hành một môi trường phân tán chỉ với duy nhất một tệp cấu hình. Mỗi thành phần của hệ thống - bao gồm *Kafka*, *Spark*, *Airflow*, *HBase*, *MongoDB*, *Flask*, và *Streamlit* - được đóng gói trong một container độc lập, giúp đảm bảo khả năng tách biệt, linh hoạt và dễ dàng mở rộng.
Docker Compose chịu trách nhiệm tự động hóa toàn bộ quy trình khởi tạo, thiết lập kết nối giữa các dịch vụ, cấu hình mạng nội bộ, cũng như quản lý thứ tự khởi động của các container. Cách triển khai này không chỉ tối ưu tài nguyên hệ thống mà còn cho phép tái sử dụng và triển khai đồng nhất trên nhiều môi trường khác nhau, từ máy cục bộ đến các nền tảng điện toán đám mây (Cloud).

2.3 Quy trình xử lý, triển khai và kết quả

Phần này sẽ trình bày chi tiết về quy trình (pipeline) xử lý dữ liệu, triển khai hệ thống và đánh giá kết quả. Ba bước quan trọng này bao gồm việc thu thập, phân tích dữ liệu để đưa ra dự đoán; quá trình triển khai và tích hợp vào hệ thống hiện có; và cuối cùng là các kết quả đạt được cùng với đánh giá chuyên sâu.

2.3.1 Thu thập, xử lý dữ liệu và phân tích, dự đoán

Hệ thống thu thập dữ liệu đồng thời từ ba nguồn:

- **Binance WebSocket API:** cung cấp dữ liệu giá và khối lượng thời gian thực cho 10 đồng tiền mã hóa.
- **Yahoo Finance API:** cung cấp dữ liệu lịch sử (batch) phục vụ cho quá trình huấn luyện mô hình.
- **NewsAPI:** thu thập tin tức và bài báo liên quan đến thị trường crypto, làm đầu vào cho mô hình ngôn ngữ lớn (LLM). API này chỉ được gọi khi người dùng hỏi chatbot, thông qua Flask và kết quả (tin tức, bài báo) được trả về sẽ làm đầu vào cho chatbot AI.

Luồng dữ liệu sau khi thu thập:

- Dữ liệu real-time sau khi được lưu trữ ở HBase sẽ đi qua mô hình *XGBoost* đã được huấn luyện trước dựa trên dữ liệu lịch sử (interval=1h) để dự đoán ngắn hạn.
- PySpark trong batch layer xử lý dữ liệu lịch sử, huấn luyện mô hình *XGBoost* (ngắn hạn và dài hạn), sau đó đồng bộ mô hình cho stream layer phục vụ việc dự đoán.

Bên cạnh mô hình định lượng XGBoost, hệ thống tích hợp thêm mô hình ngôn ngữ lớn **LLM**, có nhiệm vụ phân tích tin tức (sentiment analysis, topic extraction) và cung cấp ngữ cảnh hỗ trợ dự đoán và phân tích thị trường. Kết quả của hai mô hình được kết hợp để đưa ra nhận định tổng hợp về xu hướng thị trường.

2.3.2 Triển khai và tích hợp hệ thống

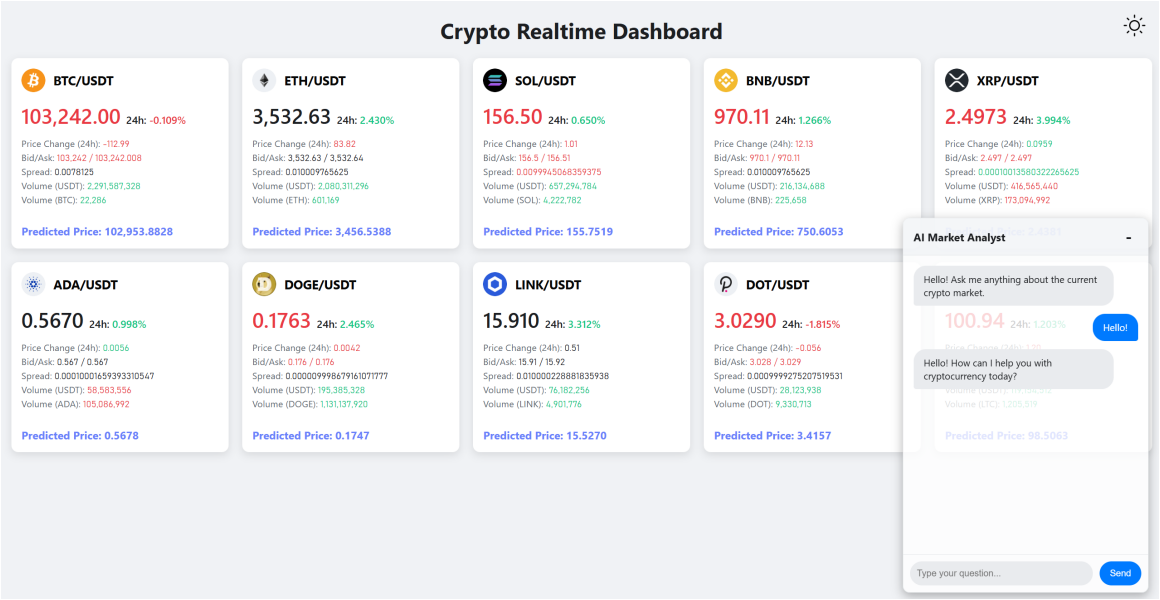
Hệ thống được triển khai hoàn chỉnh bằng **Docker Compose**, đảm bảo mỗi dịch vụ hoạt động độc lập nhưng vẫn kết nối thông qua mạng nội bộ:

- **Kafka, Spark, Airflow, HBase, MongoDB** chạy dưới dạng các container độc lập, đảm nhận các vai trò ingestion, processing và storage.
- **Flask API** cung cấp giao diện REST và websocket real-time, đồng thời tích hợp chatbot AI phục vụ người dùng.
- **Streamlit Dashboard** hiển thị dữ liệu tổng hợp, biểu đồ thống kê, và các chỉ số phân tích batch giúp phân tích xu hướng thị trường dài hạn.

Các container Docker được quản lý tự động, đảm bảo khởi động theo thứ tự phụ thuộc (ví dụ: các dịch vụ hạ tầng như Kafka, Zookeeper, Airflow, → HDFS → các cơ sở dữ liệu (HBase, MongoDB) → các ứng dụng xử lý và API (Flask)). Hệ thống vận hành ổn định, có khả năng mở rộng dễ dàng và tương thích khi triển khai trên môi trường Cloud.

2.3.3 Kết quả và đánh giá

1. Real-time dashboard: Flask UI hiển thị dữ liệu crypto, đồ thị biến động giá theo thời gian thực cùng giá dự đoán ngắn hạn và phần chatbot AI hỗ trợ. Người dùng có thể đặt câu hỏi, chatbot sẽ tự động truy vấn tin tức gần nhất theo ngày, từ đó giúp chatbot hiểu hơn về xu hướng thị trường, và đưa ra những phản hồi chính xác.

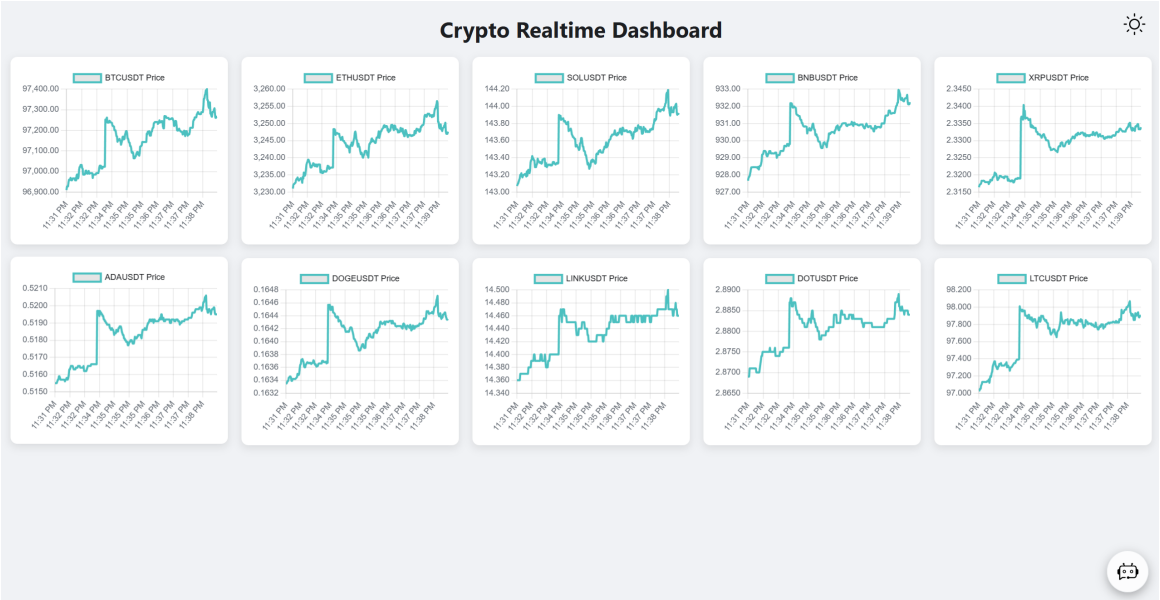


Hình 2.8: Giao diện real-time dashboard 1.

2. Giao diện batch dashboard (Streamlit): Streamlit thể hiện dữ liệu lịch sử và thống kê tổng hợp (giá trung bình, khối lượng giao dịch, xu hướng theo tuần/tháng). Dữ liệu được lấy từ MongoDB - nơi lưu các kết quả xử lý batch. Dữ liệu sẽ được Airflow cập nhật mỗi ngày tùy thuộc vào cài đặt ban đầu của người dùng.

3. Hiệu suất:

- Độ trễ trung bình trong pipeline real-time: $<$ **10 giây** và độ trễ nhỏ từ Binance WebSocket API.



Hình 2.9: Giao diện real-time dashboard 2.

Hình 2.10: Giao diện batch dashboard.

- Tốc độ huấn luyện batch: huấn luyện mô hình *XGBoost* trung bình **2 phút cho 2 năm dữ liệu / mỗi coin**.
- Khả năng mở rộng: có thể thêm cặp coin hoặc dịch vụ mới mà không cần thay đổi cấu trúc lõi.

4. Đánh giá tổng thể: Hệ thống hoạt động ổn định, độ trễ thấp, phản hồi nhanh và cung cấp thông tin toàn diện cho cả người dùng ngắn hạn (trader) và dài hạn (phân tích xu hướng). Kết quả cho thấy việc kết hợp giữa **XGBoost (định lượng)** và **LLM (định tính)** đã giúp nâng cao độ chính xác và tính thuyết phục của phân tích.

Chương 3

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

3.1 Kết luận

3.1.1 Tóm tắt nội dung và kết quả đạt được

Trong khuôn khổ đồ án, hệ thống nền tảng phân tích và dự báo thị trường tiền mã hoá đã được xây dựng hoàn chỉnh theo mô hình end-to-end, tích hợp đầy đủ các bước từ thu thập dữ liệu thời gian thực, xử lý - lưu trữ dữ liệu, huấn luyện mô hình dự báo đến trực quan hóa và cung cấp giao diện tương tác người dùng.
Các kết quả chính đạt được bao gồm:

- **Xây dựng hệ thống thu thập dữ liệu real-time** từ Binance WebSocket, truyền qua Kafka và xử lý bằng Spark Structured Streaming.
- **Thiết kế kho dữ liệu thời gian thực** sử dụng HBase, cho phép truy vấn nhanh và lưu trữ lâu dài dữ liệu giá theo timestamp.
- **Triển khai pipeline batch** sử dụng PySpark + Yahoo Finance:
  - Thu thập dữ liệu 1h/1d trong 2 năm, tính toán trên 40+ chỉ báo kỹ thuật.
  - Huấn luyện mô hình XGBoost cho từng đồng tiền mã hoá.
  - Xuất ra kết quả dự báo giá, confidence score, biến động lũy kế,... lưu vào HDFS và MongoDB.
- **Xây dựng dashboard phân tích** bằng Streamlit:
  - Hiển thị KPI, phân phối rủi ro, dự báo giá, so sánh tài sản, heatmap tương quan, gợi ý phân bổ danh mục.
  - Theo dõi tình trạng hệ thống (model coverage, data quality...).
  - Người dùng có thể tải xuống dữ liệu thống kê dưới dạng file `.csv` từ Streamlit.
- **Phát triển chatbot phân tích thị trường**, tích hợp:
  - Tin tức thời sự qua NewsAPI + trafilatura.
  - Phân tích kỹ thuật real-time dựa trên dữ liệu HBase.

- Xử lý ngôn ngữ tự nhiên với Gemini, nhận diện ý định, cảm xúc và ngôn ngữ của người dùng.
- Sinh câu trả lời theo phong cách chuyên gia phân tích thị trường.

Hệ thống hoàn thiện đã chứng minh khả năng kết hợp Big Data, Machine Learning, Data Visualization và NLP trong một kiến trúc thống nhất, vận hành mượt mà và có khả năng mở rộng.

3.1.2 Đánh giá và hạn chế

Mặc dù đạt được nhiều kết quả quan trọng, hệ thống vẫn tồn tại một số hạn chế nhất định:

(1) Về mô hình dự báo

- XGBoost mạnh nhưng không tối ưu cho chuỗi thời gian phi tuyến, biến động cao như crypto.
- Chưa khai thác kiến trúc Deep Learning chuyên biệt (LSTM, TCN, Temporal Fusion Transformer).
- MAPE và RMSE biến động lớn giữa các đồng do dữ liệu khác biệt về volume/volatility.

(2) Về dữ liệu

- Dữ liệu từ Yahoo Finance có thể gây lệch nhịp so với thị trường thực gây ảnh hưởng huấn luyện.
- Chưa tích hợp dữ liệu on-chain hoặc sentiment, những nguồn quan trọng cho crypto.
- HBase chưa được tối ưu hoá cho truy vấn phức tạp (ví dụ: long-range aggregation).

(3) Về kiến trúc hệ thống

- Hệ thống hiện chỉ được vận hành trên môi trường local thông qua Docker.
- Yêu cầu phần cứng cao cho môi trường local (khuyến nghị RAM 16–32GB).
- Thiếu monitoring đầy đủ (Prometheus, Grafana).
- Một số phần bị phụ thuộc vào cấu hình cục bộ, chưa được container hoá hoàn toàn.

(4) Về giao diện và trải nghiệm

- Dashboard Streamlit chưa hỗ trợ tuỳ biến nâng cao như multi-user hoặc phân quyền.
- Chatbot đôi lúc tạo ra nội dung quá dài hoặc phân tích quá mức khi tin tức thưa thớt.

Những hạn chế này mở ra nhiều hướng cải thiện trong tương lai.

3.2 Hướng phát triển

Trong tương lai, có thể mở rộng và nâng cấp hệ thống theo các hướng sau:

(1) Nâng cấp mô hình học máy

- Áp dụng mô hình deep learning chuyên xử lý chuỗi thời gian [7, 8]:

  - LSTM/BI-LSTM
  - Temporal Convolutional Network
  - Transformer for Time Series
  - NeuralProphet
- Hợp nhất dữ liệu real-time từ HBase với dữ liệu batch để tạo mô hình hybrid.
- Tự động tối ưu siêu tham số bằng Optuna hoặc Hyperopt [9].

(2) Mở rộng dữ liệu

- Tích hợp thêm nguồn dữ liệu [10]:
  - On-chain (Glassnode, Dune Analytics)
  - Sentiment từ mạng xã hội (Twitter, Reddit)
  - Orderbook depth để phân tích thanh khoản
- Lưu trữ dữ liệu realtime dưới dạng phân vùng theo ngày/tháng để tăng tốc độ truy vấn.

(3) Hoàn thiện kiến trúc Big Data

- Triển khai hệ thống lên môi trường cloud (GCP/AWS) hoặc Kubernetes để loại bỏ phụ thuộc vào chạy local bằng Docker.
- Tách các dịch vụ nặng (Spark, Kafka, HBase) sang dạng managed services hoặc cluster cloud để giảm yêu cầu phần cứng cục bộ.
- Tích hợp Prometheus + Grafana để theo dõi [11]:
  - Kafka lag
  - Spark microbatch latency
  - HBase write/read metrics
- Sử dụng MinIO hoặc S3 để thay thế HDFS nhằm hỗ trợ kiến trúc cloud-native.

(4) Nâng cấp dashboard và chatbot

- **Dashboard:**
  - Tối ưu UI/UX, hỗ trợ real-time updates từ WebSocket.
  - Chế độ so sánh nhiều crypto qua timeline.
- **Chatbot:**
  - Bổ sung memory dài hạn:
    - Theo user
    - Theo lịch sử thị trường
  - Cho phép xuất báo cáo PDF tự động theo yêu cầu.
  - Tích hợp voice hoặc Telegram bot.

(5) Định hướng triển khai thực tế

- Đưa hệ thống lên cloud (GCP/AWS/DigitalOcean).
- Mở API public đọc dữ liệu giá real-time + dự báo.
- Tạo module "paper trading" thử nghiệm chiến lược dựa trên dự báo.

Tài liệu tham khảo

[1] G. Zhang, "Time series forecasting using a hybrid ARIMA and neural network model," *Neurocomputing*, vol. 50, pp. 159–175, 2003.

[2] T. Fischer and C. Krauss, "Deep learning with long short-term memory networks for financial market predictions," *European Journal of Operational Research*, vol. 270, no. 2, pp. 654–669, 2018.

[3] J. Bollen, H. Mao, and X. Zeng, "Twitter mood predicts the stock market," *Journal of Computational Science*, vol. 2, no. 1, pp. 1–8, 2011.

[4] D. Shen, W. Zhang, and X. Xiong, "The impact of news on cryptocurrency markets," *Finance Research Letters*, vol. 30, pp. 1–7, 2019.

[5] A. Mittal and A. Goel, "Stock prediction using news sentiment analysis," in *Proc. IEEE/WIC/ACM International Conference on Web Intelligence*, 2012.

[6] J. Liew and T. Budavari, "The use of alternative data in investment management: A practitioner's guide," *The Journal of Financial Data Science*, vol. 1, no. 1, pp. 10–23, 2019.

[7] B. Lim *et al.*, "Temporal Fusion Transformers for interpretable multi-horizon time series forecasting," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[8] A. Vaswani *et al.*, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017.

[9] T. Akiba *et al.*, "Optuna: A next-generation hyperparameter optimization framework," in *Proc. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2019.

[10] F. Victor, "Measuring Ethereum-based assets," in *Financial Cryptography and Data Security*, 2021.

[11] B. Turner, *Monitoring with Prometheus*. O'Reilly Media, 2020.

[12] Documentation: Apache Kafka, Apache Spark, HBase, MongoDB, Streamlit, NewsAPI, OpenAI API.

PHỤ LỤC

A. Mã nguồn, tài liệu bổ sung

Toàn bộ mã nguồn, tài liệu được cung cấp tại đây.
Các video demo hệ thống: batch dashboard, real-time dashboard.

B. Một số commands hữu dụng

1. Docker

```
docker compose up --build -d
docker compose down
docker logs <service>
docker ps
docker compose restart <service>
```

2. Airflow

```
docker exec airflow airflow dags trigger batch_pipeline
docker exec airflow airflow dags list-runs -d batch_pipeline
```

3. HDFS

```
docker exec namenode hadoop fs -ls /crypto/yahoo
docker exec namenode hadoop fs -rm /crypto/yahoo/btcusdt/*
```

4. HBase

```
docker exec hbase hbase shell
list
scan 'crypto_prices', {LIMIT => 5}
```

5. MongoDB

```
docker exec -it mongodb mongosh
use crypto_batch
db.predictions.find().pretty()
```

6. Full System Reset

```
1. Stop all containers:
docker compose down

2. Delete all persistent volumes:
docker volume prune
# Press Y when prompted

# If you want to delete only project-specific volumes:
docker compose down -v

3. Rebuild all images:
docker compose build --no-cache

4. Start everything again:
docker compose up -d
```

C. API Endpoints

- GET /api/crypto/{symbol}
- GET /api/crypto/history/{symbol}
- GET /api/crypto/predictions/{symbol}
- POST /api/chatbot/ask