Finflow Engineering

(Final)

2025. 12. 09

Team 3

정현준, 강승욱, 박효빈, 이형경

POSTECH
POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Contents

1. Introduction & Motivation

2. Project Objective

3. Related Work & Technical Background

4. Data Sources

5. Implementation

6. Evaluation

7. Conclusion

8. Reference

Introduction & Motivation


Forbes

MONEY > WEALTH MANAGEMENT

Information Overload: Unraveling The Paradox Of Investment Knowledge

By John Jennings, Contributor. © John Jennings covers investments & issues t... Follow Author

Published Jul 21, 2023, 08:34am EDT

- **Problems faced by investors**



Information Overload: Difficult to distinguish valuable signals from noise in the flood of financial news.



Time Limitation: Individual investors lack sufficient time to track and analyze news in real time.



Lack of Expertise: Most investors do not have the expertise to interpret causal relationships between news and stock price movements.

Introduction & Motivation

- **Limitations of Existing Solutions**
  - **Technical Analysis Tools (e.g., TradingView):**

    Cannot interpret or analyze news content.
  - **Robo-Advisors (e.g., Fint):**

    Cannot proactively respond to real-time market events.
  - **AI Asset Management Firms (e.g., QRAFT):**

    Operate as black boxes, providing little to no transparency in decision-making.
  - **Conclusion**

    There is no platform that transforms real-time financial news into transparent, evidence-based insights.
- **Our project aims to fill this gap by converting real-time news into actionable insights that empower individual investors.**

Category (Function)	Technical Analysis Tools (e.g., TradingView)	Robo-Advisors (e.g., Fint)	AI Asset Management Firms (e.g., QRAFT)	Finflow (Our Project)
Real-time charts/indicators	○	X	X	○
Automated asset allocation	△ (limited)	○	○	○
Real-time news analysis	X	X	△ (indirect/limited)	○
Transparent rationale for insights	X	X	X	○

Project Objective

- **Main Goal**
  - ***Turning real-time financial news into transparent, actionable insights.***

- **Three Core Features**
  - **Real-Time Data Integration**

    Continuously collect and align financial news with market/FX data
  - **LLM-Based Insights**

    - Go beyond sentiment and keywords; summarize news context with LLMs

    - Deliver transparent **BUY / SELL / HOLD** signals
  - **Interactive Dashboard**

    - Visualize complex news–asset relationships in a user-friendly dashboard

    - [Scalability] Expand to crypto, FX, and other asset classes

Related Work & Technical Background

- **Latest Research on LLMs in Finance**
  - **Recent studies*** (e.g., *MDPI, 2024*) show that LLMs are moving beyond simple text analysis toward applications in investment decision-making.
  - LLMs can capture subtle financial contexts, perform sentiment analysis, and generate signals useful for trading decisions.

- **Our Approach**
  - Based on this research trend, our project leverages the latest sentiment analysis and summarization techniques to guide users with clear investment advice (**BUY / SELL / HOLD**).
  - We aim to apply academically validated methods to a real-time data pipeline and deliver practical, user-focused insights.

Recent studies*:
WWW'24 (ACM): *LLMs turn news into explainable stock predictions (SEP).*
WWW'24 (ACM): *Explainable earnings forecasting via financial news (FinReport).*
NeurIPS'24: *Causal discovery reduces noise in news-driven multi-stock prediction (CausalStock).*

Data Source

- **Finance News**
  - Google News API (Global)
  - Bloomberg/Reuters RSS (Real-time financial articles)

- **Market Price Data**
  - Yahoo Finance API (Stocks/Indices/Exchange Rates/Cryptocurrencies)
  - Alpha Vantage (Free, minute-by-minute quotes)

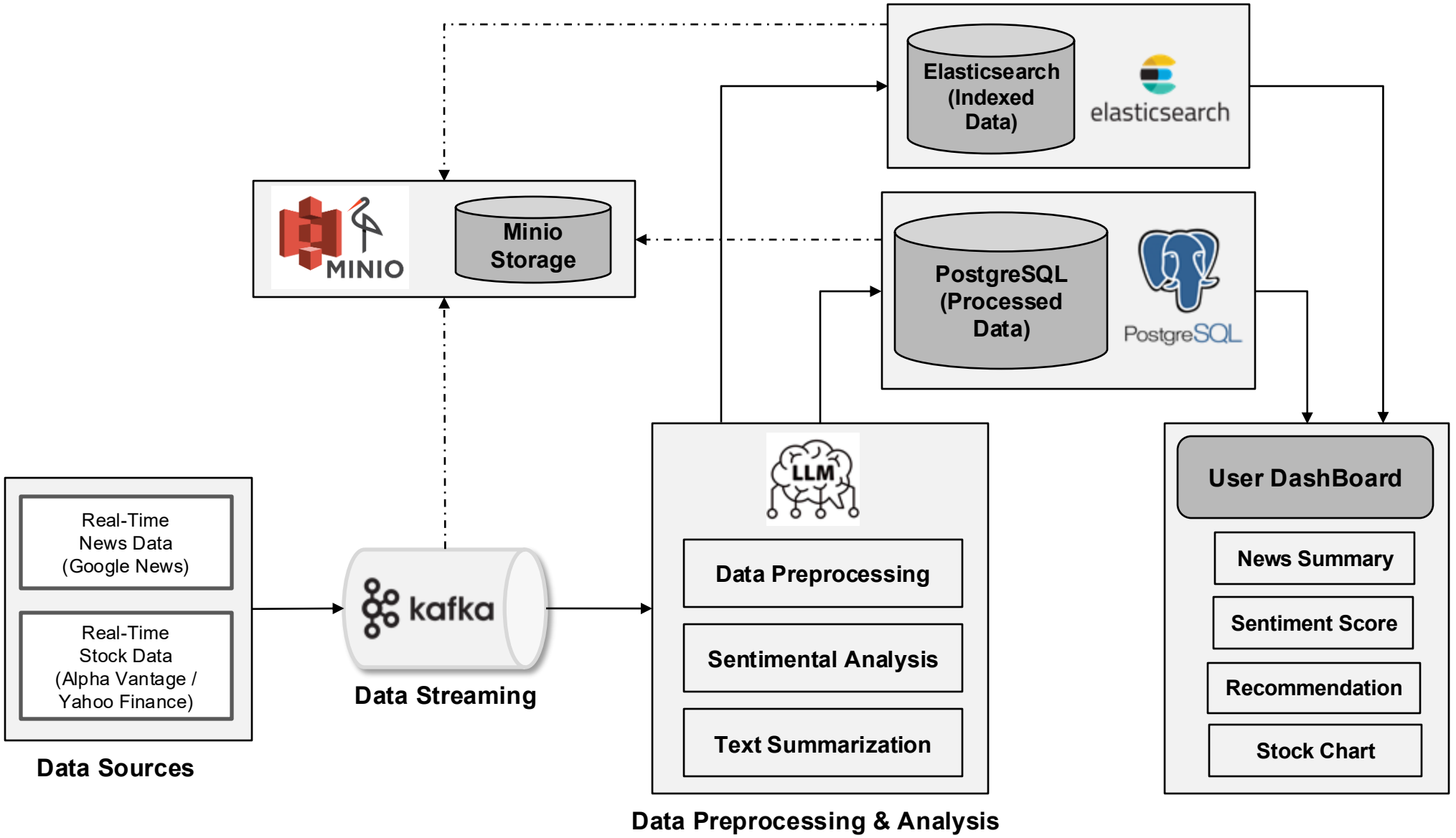


<Google News API>



<Alpha Vantage>

System Architecture



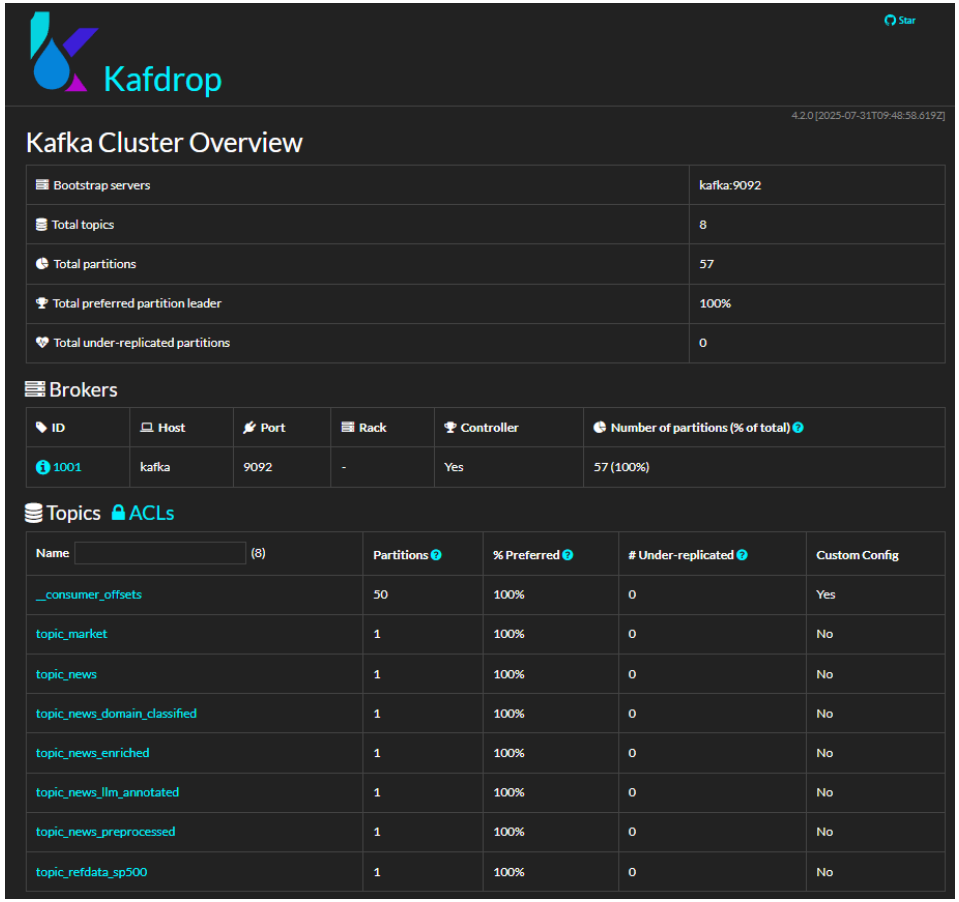
Implementation - Data Collection

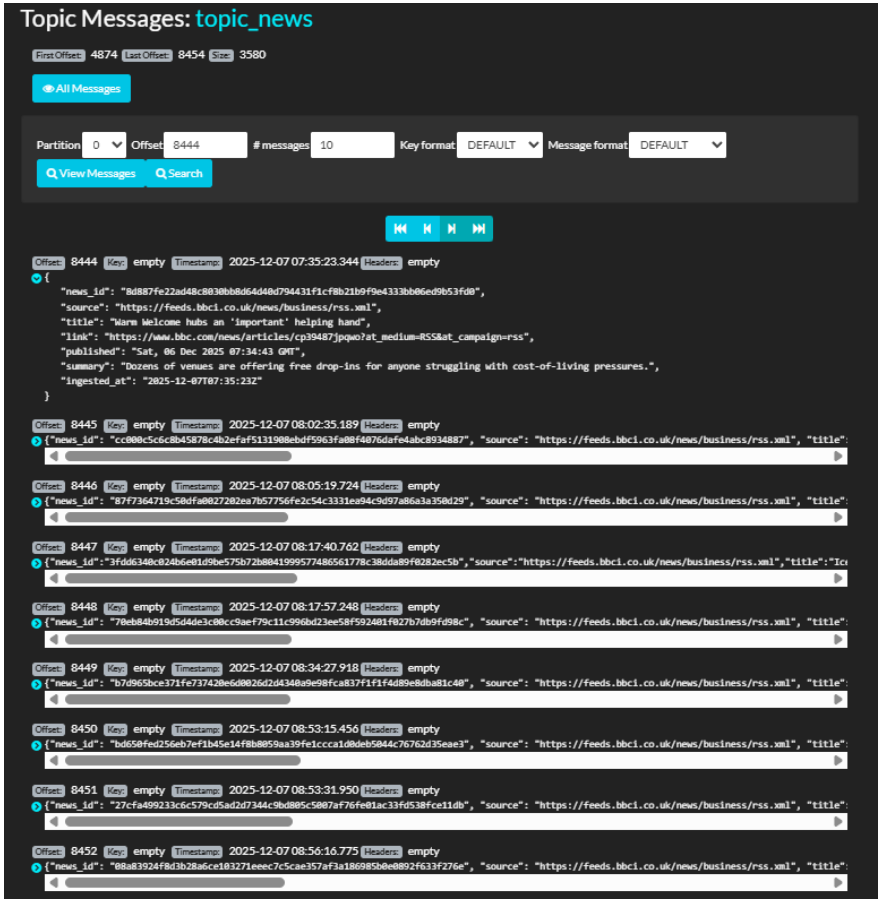
- **Kafka Producer**
  - **News Crawler**
    - Google News API, Bloomberg/Reuters RSS → Extracts HTML content & Convert into JSON schema
  - **Market Data Collector**
    - Yahoo Finance / Alpha Vantage API → Real-time stock price and volume collection
  - **Kafka Producer Characteristics**
    - Asynchronous batch sending for higher throughput
    - Dual-write: Kafka + MinIO to ensure durability and reproducibility
    - Forward collected data to Kafka Producer → Publish to ***topic_news*** and ***topic_market***

- **Kafka Broker**
  - **Multi-Stage Topic Pipeline**
    - topic_news_raw – Raw crawled articles (JSON)
    - topic_news_preprocessed – Cleaned text, language detection, parsed content
    - topic_news_domain_classified – LLM-based domain classification (Tech, Energy, etc.)
    - topic_news_enriched – Sentiment score, importance, event type, ticker mapping
    - topic_market – Real-time stock price and volume data

Implementation - Data Collection

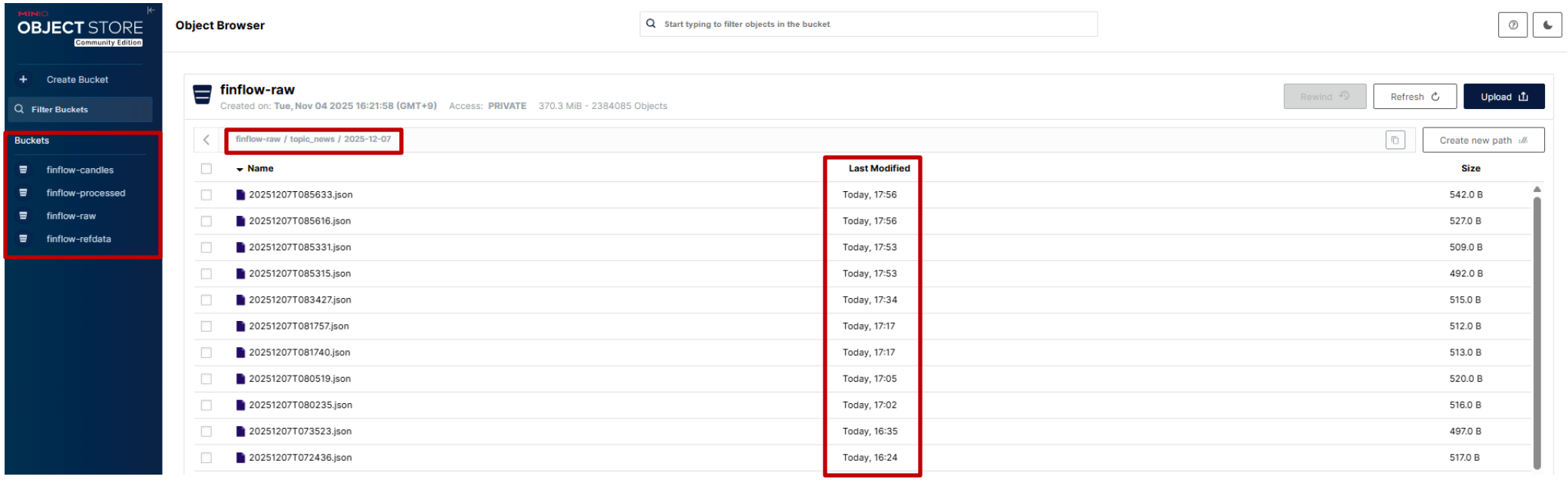
- **Kafka Visualization – Kafdrop**
  - Data Collection Monitoring with Kafdrop
  - Allows real-time inspection of ingested news and market data
  - Provides transparency of the entire streaming pipeline from ingestion → processing → storage





Implementation - Data Collection

- **Raw Data Storage - Minio**
  - Bucket Types
    - Raw Bucket
    - Processed Bucket
    - Reference Bucket
    - Candle Bucket



Implementation - Database Management (Storage Strategy)

- **Raw Data (Data Lake)**
  - **Minio** → Long-term storage of original news/price quotes
    - Original news text (JSON)
    - Original price log (CSV/JSON)

- **Structured DB (RDBMS)**
  - **PostgreSQL** → Preprocessed news, quotes, and insights stored
    - Preprocessed news (title, sentiment score, keywords, summary)
    - Quote data (ticker, price, return, volume)
    - Insight results (BUY/SELL/HOLD, description)

- **Search/Analysis DB**
  - **Elasticsearch** → Quick keyword/stock-based search support
    - News text + keyword index

- **Data flow:**
  - Raw (Minio) → Preprocessing/Processed (PostgreSQL) → Search/Analytics (Elasticsearch)

Implementation – Candles(Price) Data Processing

- **Candlestick Data: Ingestion, Aggregation, and Storage**

1. Unified OHLCV Schema

- Maintain a **single standardized OHLCV schema** across all timeframes
  symbol, ts, start_ts, end_ts, o, h, l, c, v with **[start_ts, end_ts)** bucket semantics.
- Timeframes: 1m, 5m, 15m, 30m, 1h, 1d, 1w, 1mo (optionally 1y)
  stored in ES indices *candles-{tf}-0001*

Batch: Historical Loader

2. Historical Backfill from yahoo! finance

▶ Load the S&P 500 symbol list from MinIO (refdata/sp500/latest.csv)

▶ For each symbol and timeframe:
- Download OHLCV from **Yahoo Finance** with per-TF lookback limits
- Normalize to the standard schema and assign bucket boundaries using
  - fixed intervals for intraday (1m–1h),
  - calendar rules for 1d / 1w (Monday start) / 1mo / 1y

▶ Bulk idempotent upsert to Elasticsearch *(candles-{tf}-0001*, id = *symbol_epoch_second)*

▶ A **daily scheduler** triggers this loader around **UTC 00:00** in "missing_only" mode to keep history fresh with minimal load

Stream: Candles Aggregator

3. Real-time 1m Aggregation from kafka

▶ Consume 1-minute OHLCV messages for each symbol from Kafka topic *topic_market*.

▶ For each message:
- Align timestamp to the **minute floor (UTC)**.
- Use the **NYSE trading calendar** to detect regular session → **continuous regular-session candles**
- Upsert the 1m candle into *candles-1m-0001*

▶ For higher TFs (5m~1mo):
- Re-aggregate from 1m data in ES:
  - open = first 1m
  - close = last 1m
  - high = max
  - low = min
  - volume = sum.
- Special handling for 1w / 1mo buckets (ISO Monday start, first day of month)

Loop: Candles Reconciler

4. Reconciliation & Data Quality

▶ Periodically scan each symbol/TF for **missing buckets** using ES date_histogram ("zero bucket" detection)

▶ For missing ranges:
- Download the corresponding interval from yahoo! finance (per TF interval).
- Map each record to the correct bucket start (*align_start*) and upsert only the empty buckets with *writer = "reconciler_yf"*

▶ This loop **fills gaps caused by downtime or late messages** while keeping existing candles untouched.

Real-time candle data

Candle data from very distant past

Missing data due to issues (server maintenance, API issues)

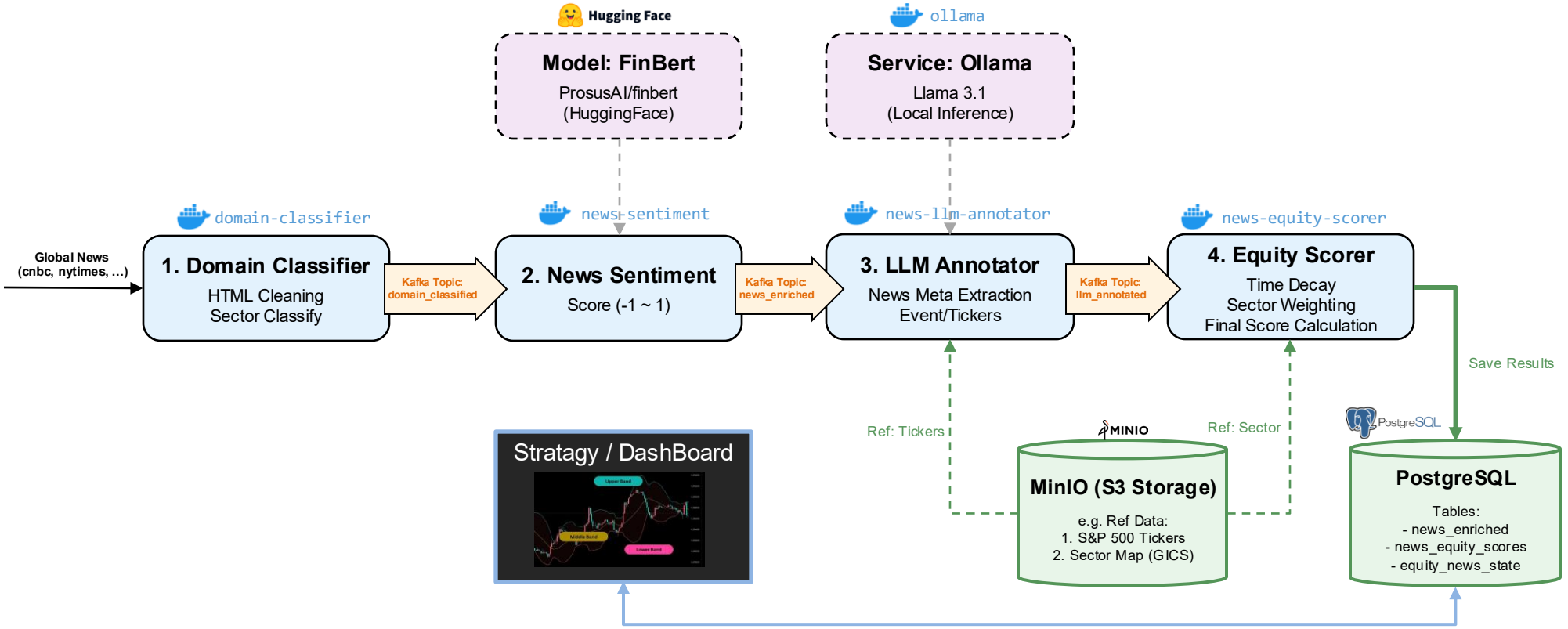
elasticsearch

All pipelines converge to: Elasticsearch Indices (candles-{tf}-0001)

Implementation - News Sentiment & Scoring: Big Picture

News Sentiment & Equity Scoring Module

- **Goal**
  - Transform global news stream into **real-time news scores (−1 ~ 1)** for each S&P 500 stock.
  - Store results in PostgreSQL so strategies and dashboards can directly query per-equity news signals.
- **Pipeline (End-to-End News Scoring)**



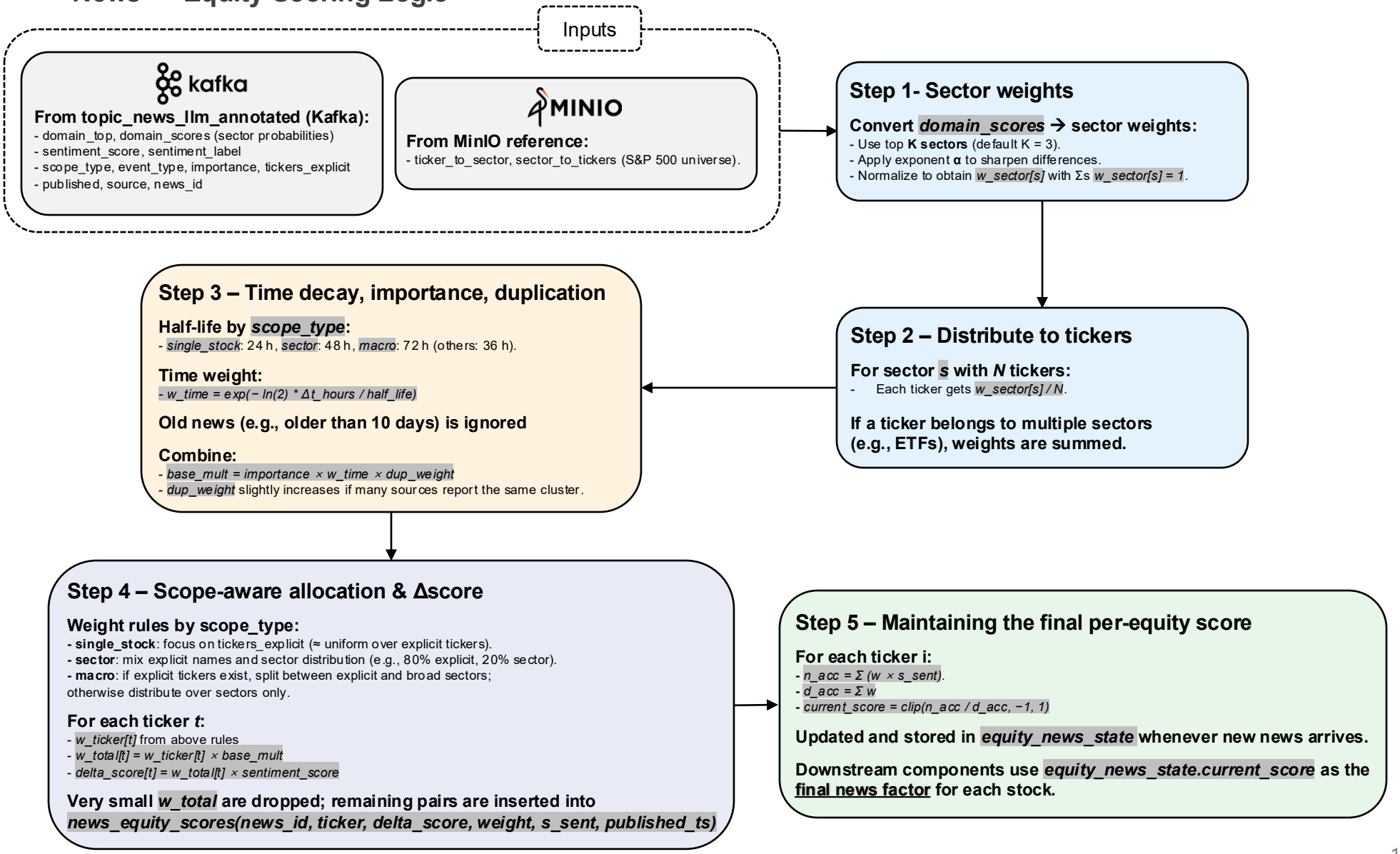
"This module takes sector-classified news and produces a clean, persistent per-stock news factor."

Implementation - Sentiment & LLM: From Text to Structured Signals

- **FinBERT sentiment (news-sentiment)**
  - Model: **ProsusAI/finbert** (finance-specific BERT sentiment model).
  - Input: **title + full_text** of each article.
  - Long articles:
    - Split into 512-token chunks → run FinBERT per chunk.
    - Aggregate via **token-weighted average** of P(neg / neu / pos).
  - Output fields:
    - *sentiment_score = P(pos) − P(neg)* → **continuous [−1, 1]**
    - *sentiment_label ∈ {neg, neu, pos}*
    - *sentiment_model*, *sentiment_at*
  - Writes into Kafka topic *topic_news_enriched*.
- **LLM meta annotation (news-llm-annotator)**
  - Backend: **Llama via Ollama** (local LLM server).
  - Reference: S&P 500 tickers + sectors loaded from MinIO (*refdata/sp500/latest.csv*).
  - For each enriched news, the LLM predicts:
    - *scope_type* – *single_stock / sector / macro*
    - *event_type* – *earnings / macro / policy / company_action / generic*
    - *importance* – continuous **[0, 1]**, strength of impact
    - *tickers_explicit* – directly affected S&P 500 tickers (e.g., *["AAPL","MSFT"]*)
  - Protocol:
    - Prompt = ticker list + news JSON + explicit instruction to output only these 4 fields as JSON.
    - Output topic: *topic_news_llm_annotated.*

Implementation - From One News Article to Per-Stock News Score

- **News → Equity Scoring Logic**



Implementation – Investment Strategy

- **News-Driven Stock Selection + RSI & Bollinger Band Trading**

1. Stock Selection (News-Based Top 10 Filtering)

- Each stock receives a **News Score** based on aggregated sentiment and relevance.
- The **Top 10 stocks** with the highest News Scores are selected for trading.

  → Focuses on stocks with **high market attention, strong catalysts, and short-term volatility.**

2. Buy/Sell Decision: RSI + Bollinger Band Combination

RSI (Relative Strength Index)

- Momentum indicator ranging from 0–100.
- **RSI < 30 → Oversold condition → High probability of rebound**.

Bollinger Bands

- Middle Band: 20-period moving average.
- Upper/Lower Bands: MA ± 2 standard deviations
- Price below the Lower Band → Excessive short-term sell pressure.



[Bollinger Bands]

Implementation – Investment Strategy

- **News-Driven Stock Selection + RSI & Bollinger Band Trading.**

< News-Score Top 10 >

Google

amazon

NVIDIA.

Microsoft

< RSI & Bollinger Band >

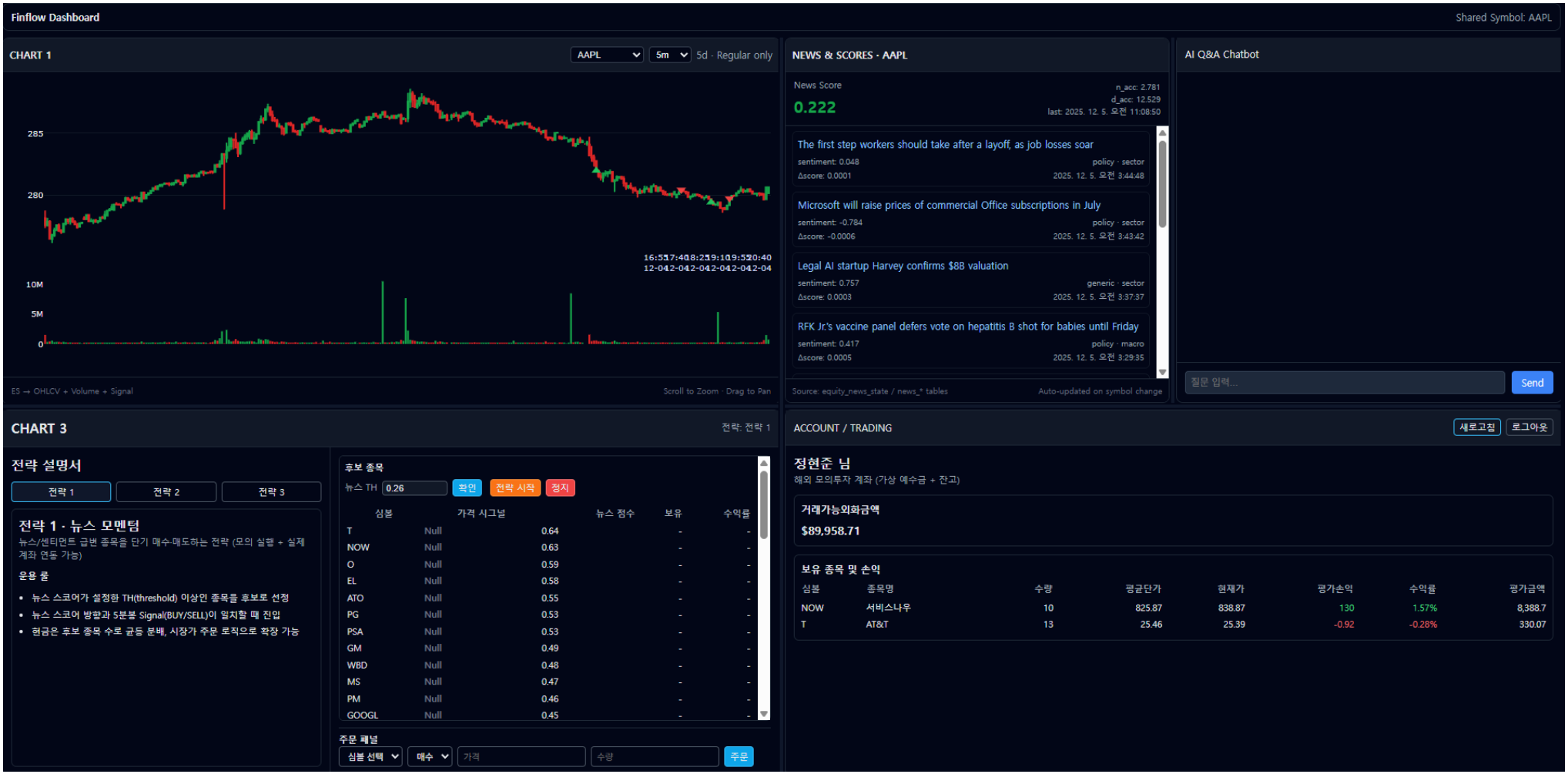
BUY RSI < 30, AND
Close price < Lower Bollinger Band

SELL RSI > 55, OR
Close price ≥ Middle Band (MA20)



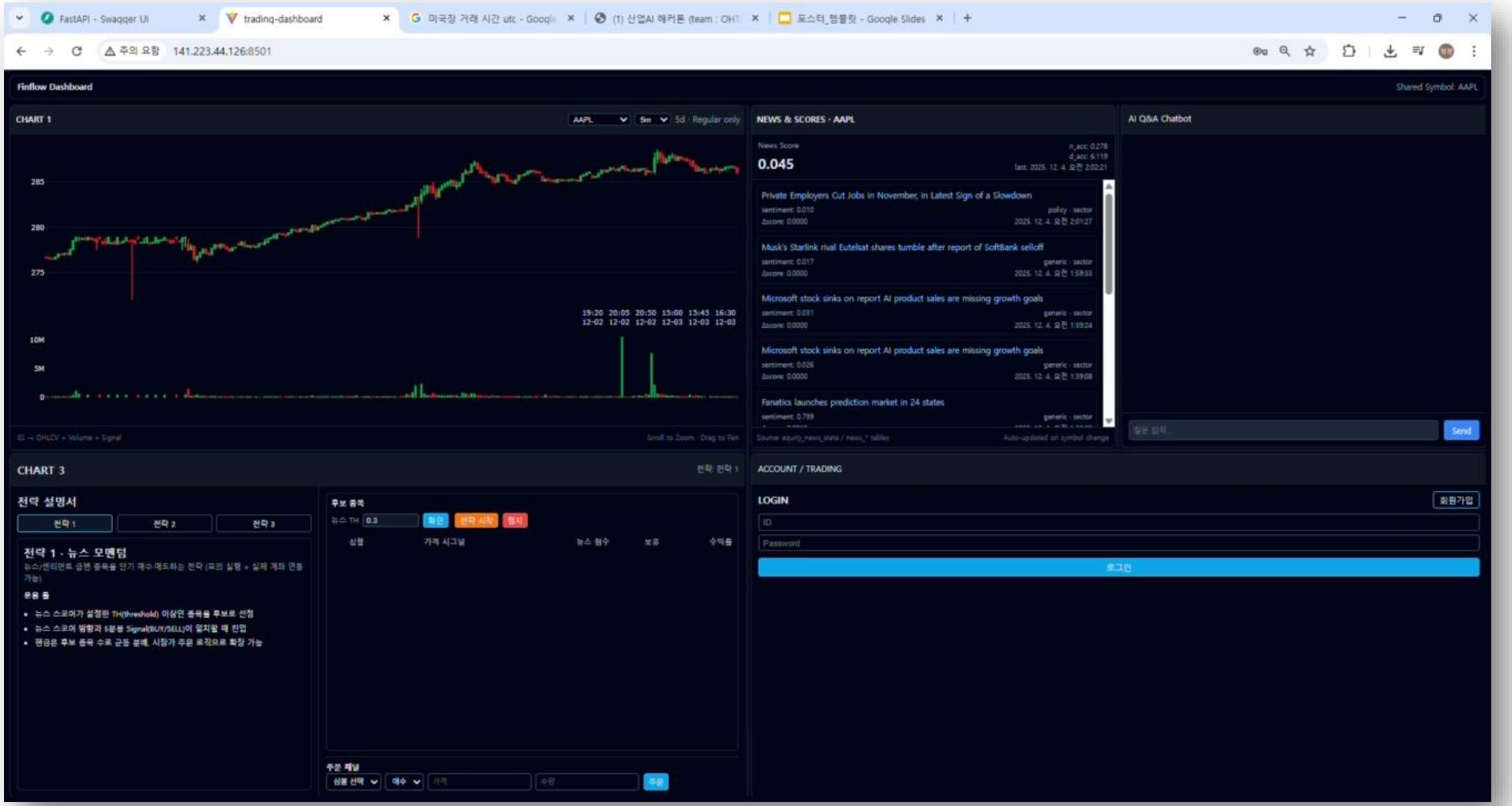
Implementation – Dashboard

- **Finflow Dashboard**
  - http://141.223.44.126:8501/



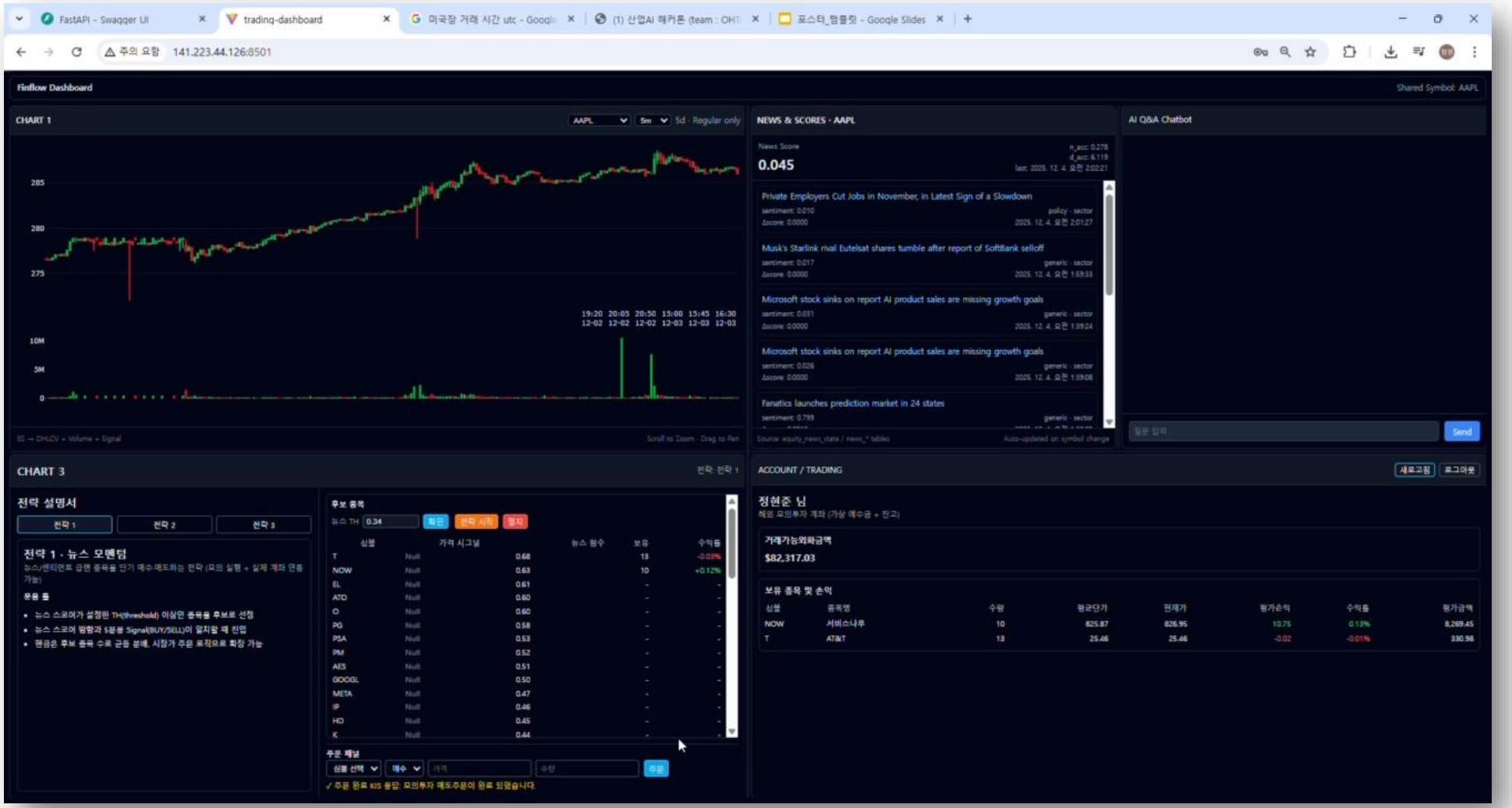
Demo

- **Available buying power on login**



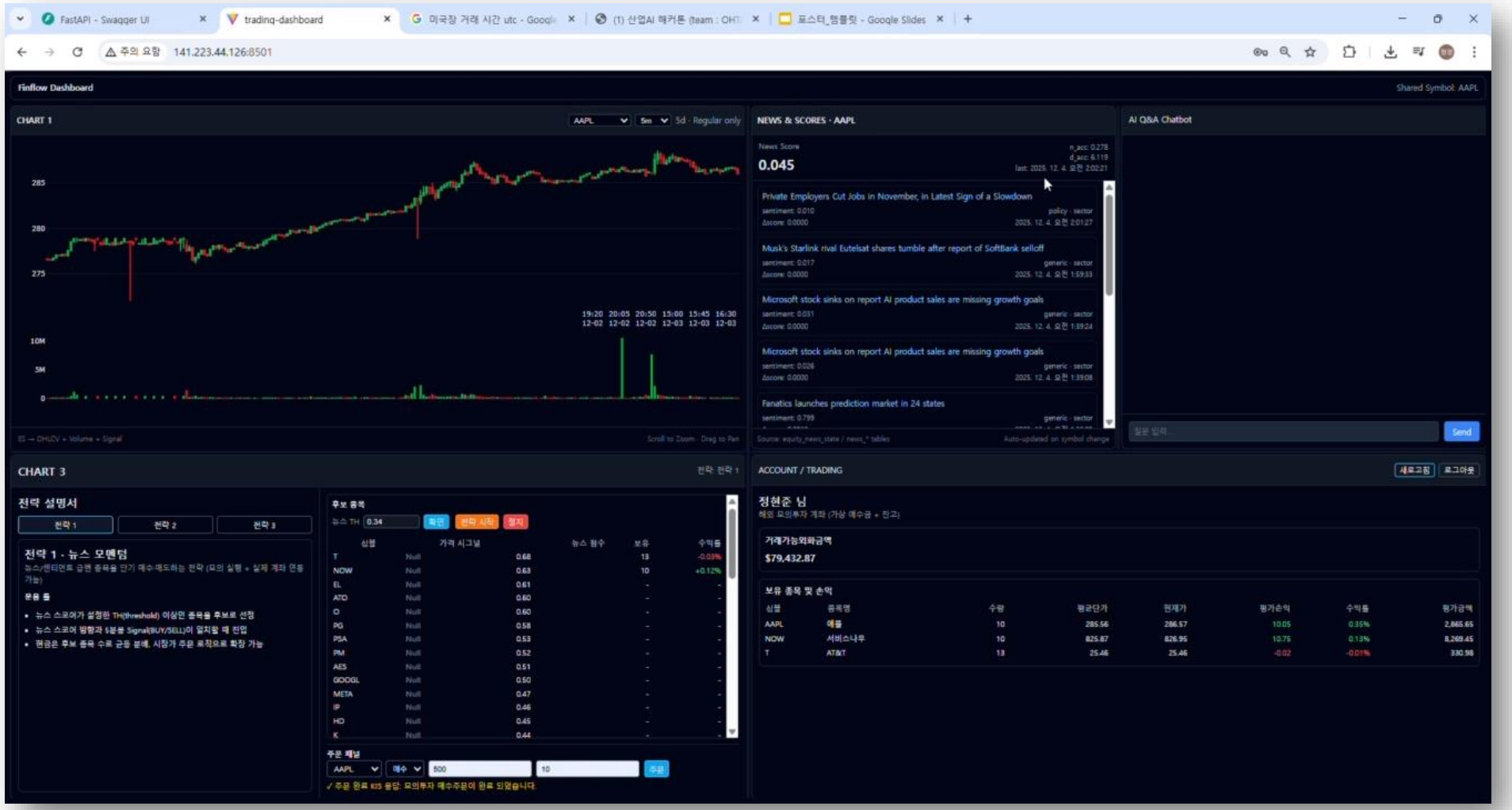
Demo

- **Manual Buy**



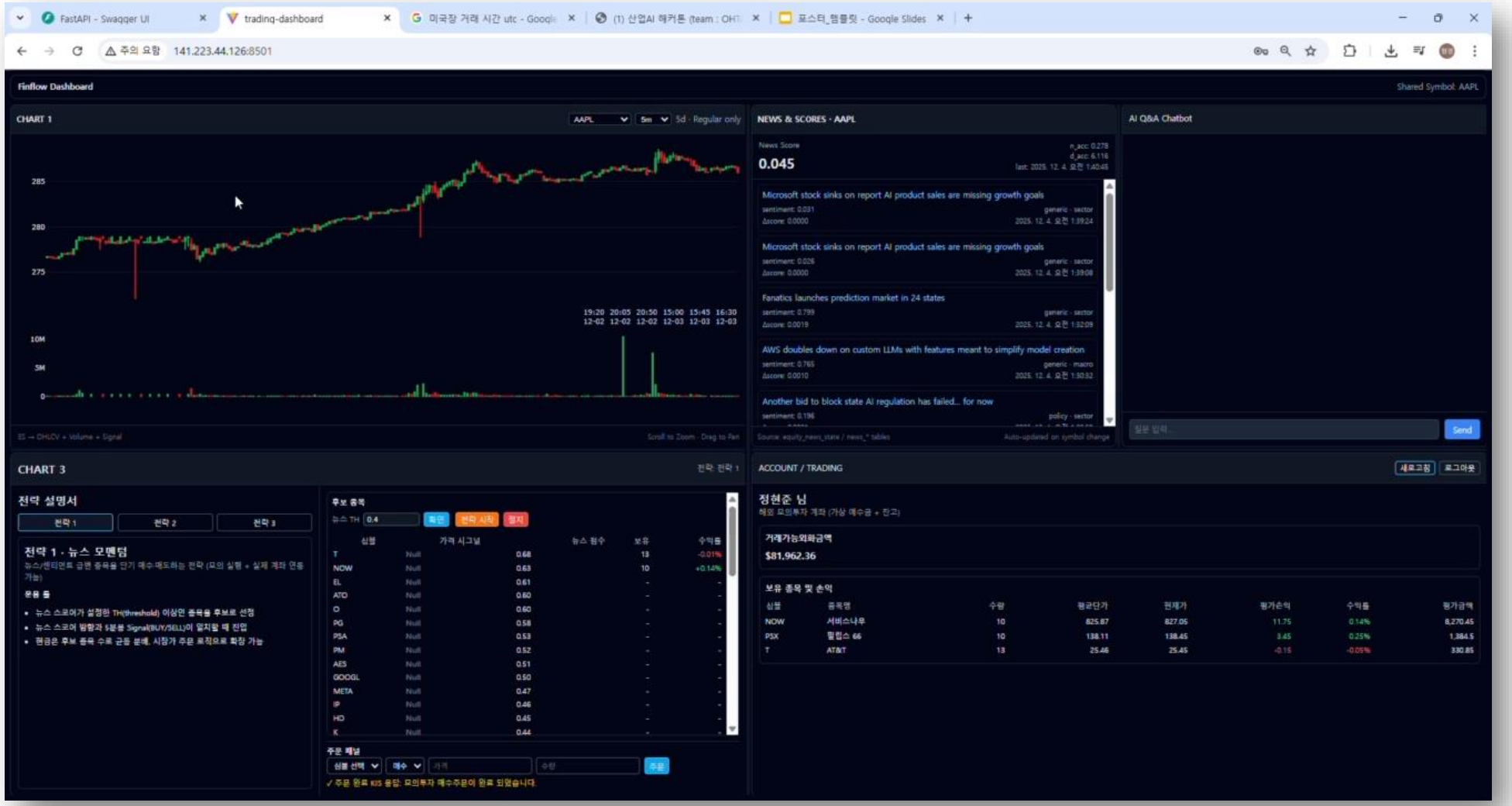
Demo

- **Manual Sell**



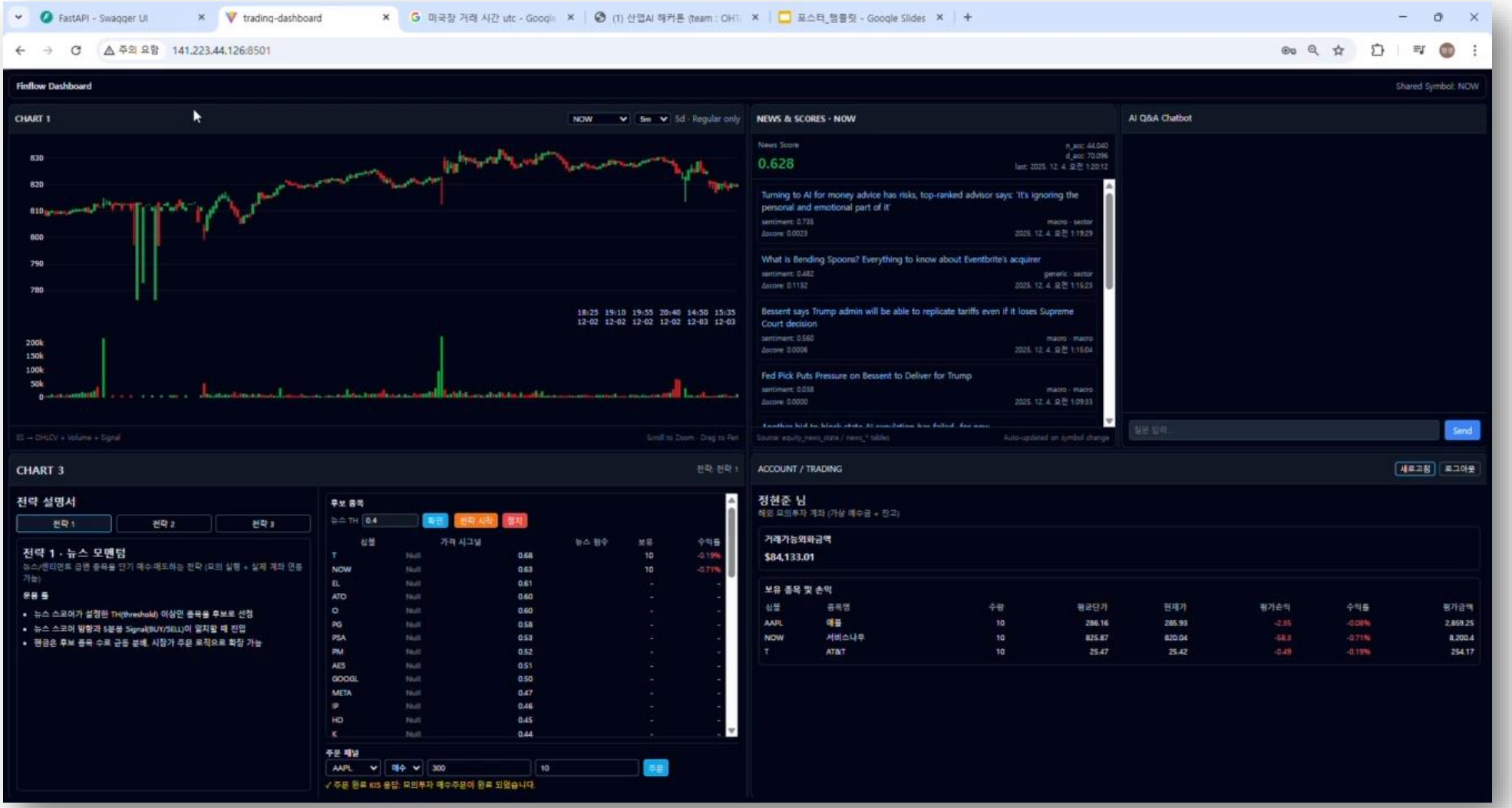
Demo

- **Sell signal detected within candidate list**



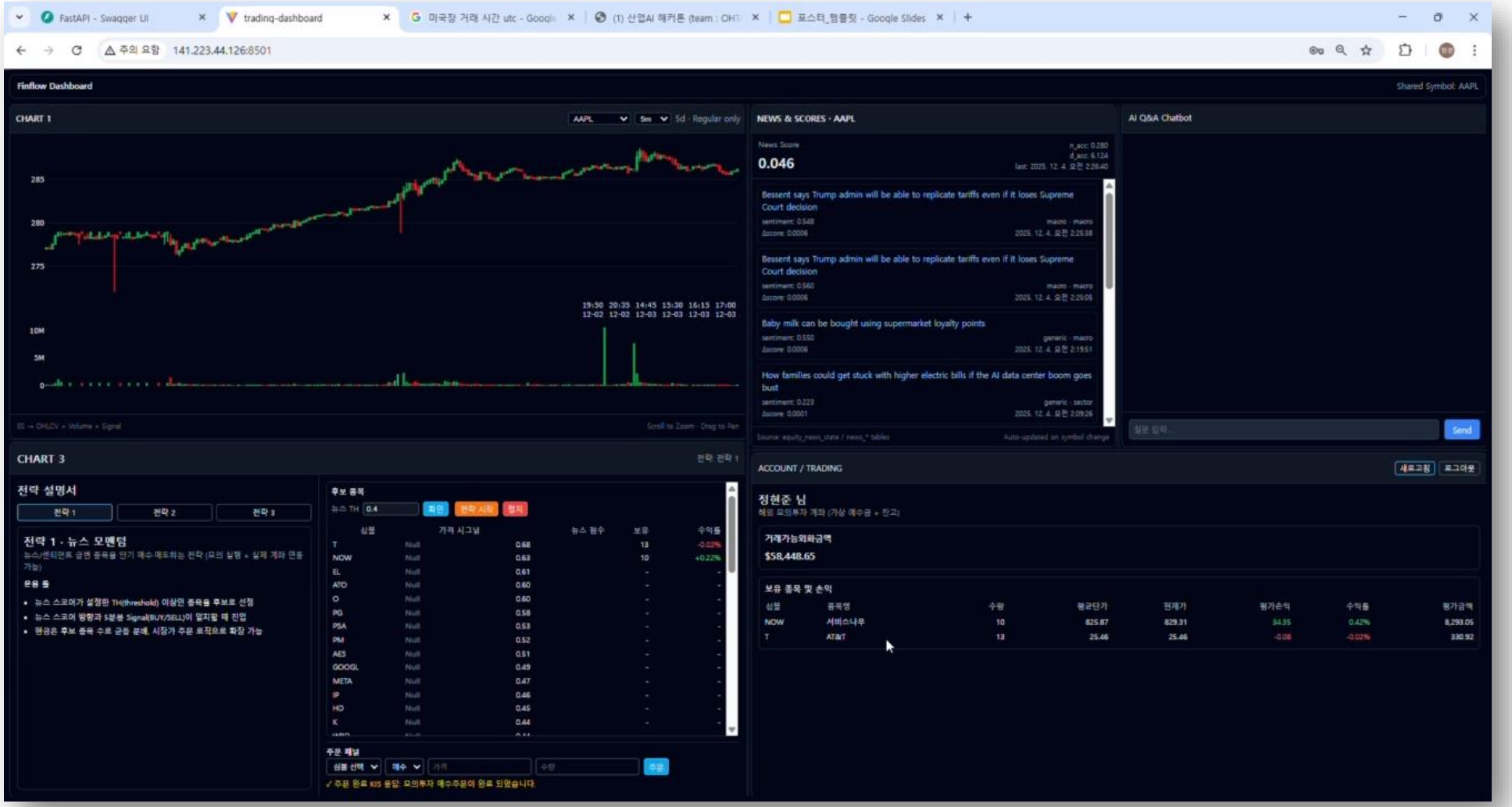
Demo

- **Sell signal for symbols not in the candidate list**



Demo

- **Buy signal detected within candidate list**



Conclusion

- **Project Summary**
  - Finflow implements a fully automated, real-time financial intelligence pipeline that ingests **RSS news** and **live market data** through Kafka, normalizes them into a unified schema, and stores them in MinIO, PostgreSQL, and Elasticsearch
  - Overall, the project demonstrates how streaming architecture, NLP/LLM models, and rule-based strategies can work together to form an end-to-end quantitative trading platform

- **Lessons Learned**
  - Importance of a unified data schema
  - News data is noisy and requires multiple layers of enrichment
  - Real-time systems require careful fault tolerance & idempotency
  - LLM modules provide strong interpretability but must be combined with rules

- **Future Work**
  - Quantitative Evaluation
    - News sentiment → Concordance rate with actual price movements (%)
    - Rule-based vs. LLM-based recommendations → Return comparison

Project Timeline

Task	Chuseok	Oct 2nd Week	Midterm	Oct 4th week	Nov 1st week	Nov 2nd week	Nov 3rd week	Nov 4th week	Dec 1st week
Set up development environment & integrate data source APIs									
Implement basic data collection & streaming pipeline									
Develop core modules integrating FinBERT & LLM APIs									
Integrate PostgreSQL & Elasticsearch for data modeling									
Develop and demo dashboard prototype using Streamlit/Flask									

References

- **Koa, K. J. L., et al. (2024). *Learning to Generate Explainable Stock Predictions using News*. In Proceedings of The Web Conference (WWW '24). ACM.**

- **Wang, X., et al. (2024). *FinReport: Explainable Stock Earnings Forecasting via Financial News*. In Proceedings of The Web Conference (WWW '24). ACM.**

- **Shuqi Li, et al. (2024). *CausalStock: Deep End-to-end Causal Discovery for News-driven Multi-stock Movement Prediction*. In Advances in Neural Information Processing Systems (NeurIPS '24).**