# Multi-Agent Financial Analysis System

Riyaz Khorasi, Sourangshu Pal, Tanvi Singh

MS - Applied Artificial Intelligence, University of San Diego

AAI 520 IN2: Natural Language Processing and GenAI

Amit Butail

20th October 2025

# Abstract

The financial industry increasingly depends on intelligent systems to process vast amounts of data and generate actionable insights. This report presents the design and implementation of a **multi-agent financial analysis system** that integrates several autonomous analytical agents, each responsible for a specific component of economic evaluation—including market data processing, sentiment analysis, quantitative evaluation, technical trend identification, and time-series forecasting using the *Prophet* model. The agents collaboratively generate comprehensive insights and investment recommendations, presented through an interactive command-line interface built using Python's *Rich* library.

The system's modular architecture allows seamless integration of new analytical models, supports real-time data analysis, and ensures transparency through visually structured outputs. Through this design, both technical users (such as data scientists) and non-technical users (such as investors or analysts) can interpret complex financial insights intuitively.

# Introduction

## 1. Background

Modern financial markets are characterized by high volatility and rapid data evolution. Traditional approaches to financial analysis often rely on manual interpretation of stock data, news, and market indicators, which can be both time-consuming and prone to human bias. To address this challenge, artificial intelligence (AI) and machine learning (ML) techniques have been increasingly applied to automate financial forecasting, pattern recognition, and risk evaluation.

Despite these advances, most AI-driven solutions remain monolithic — performing one task at a time, such as price prediction or sentiment analysis. In contrast, **multi-agent systems (MAS)** offer a more flexible and collaborative approach. MAS architecture divides complex analytical processes into independent, specialized components (agents), each capable of working autonomously and communicating results to others.

## 2. Motivation

The motivation behind this project stems from the need for:

- **Holistic analysis:** Combining quantitative, qualitative, and predictive models.
- **Automation:** Reducing manual effort in analyzing large-scale market data.
- **Interpretability:** Providing clear, explainable results even for non-technical users.
- **Scalability:** Enabling modular extensions for new financial metrics and APIs.

## 3. Objectives

This project aims to:

- Develop an intelligent agent-based system that performs end-to-end financial analysis.
- Integrate forecasting, sentiment, and sectoral evaluation into a unified workflow.
- Provide an intuitive and visually enhanced terminal-based user interface.
- Generate clear recommendations supported by evidence and confidence scores.

## 4. System Overview

At its core, the system operates through a **central state manager** that gets the data from multiple analytical agents. Each agent is responsible for one analytical domain — such as sentiment or forecasting — and outputs structured results. These outputs are finally synthesized into a comprehensive financial summary and recommendation for the target stock symbol.

The system uses key libraries, including:

- *yfinance* for market data retrieval
- *Prophet* for forecasting
- *Feedparser* for sentiment extraction
- *StreamLit* and *Rich* for the UI layer
- *LangGraph* for workflow coordination

# Methodology

## 1. System Architecture

The system follows a **multi-agent design pattern**. Each agent encapsulates a specific analytical logic and interacts with others through a shared data structure called `AnalysisState`. This ensures consistency, traceability, and ease of data exchange.

The workflow is as follows:

1. The user inputs a target stock symbol (e.g., “AAPL”).
2. The Market Data Agent retrieves historical and current data using *yfinance*.
3. Technical, quantitative, and sentiment agents analyze respective aspects.
4. The Prophet Forecast Agent predicts short-term price movements.
5. All results are synthesized and displayed via the Rich-based UI.

## 2. Description of Agents

### a. Market Data Agent

This agent serves as the foundation of the analytical pipeline, responsible for fetching and organizing real-time and historical financial data for a given stock symbol. It gathers a comprehensive set of quantitative and qualitative indicators that describe a company’s financial health, operational efficiency, and market position.

Inputs: Stock ticker symbol (e.g., *AAPL*, *TSLA*, *MSFT*).

**Processes:**

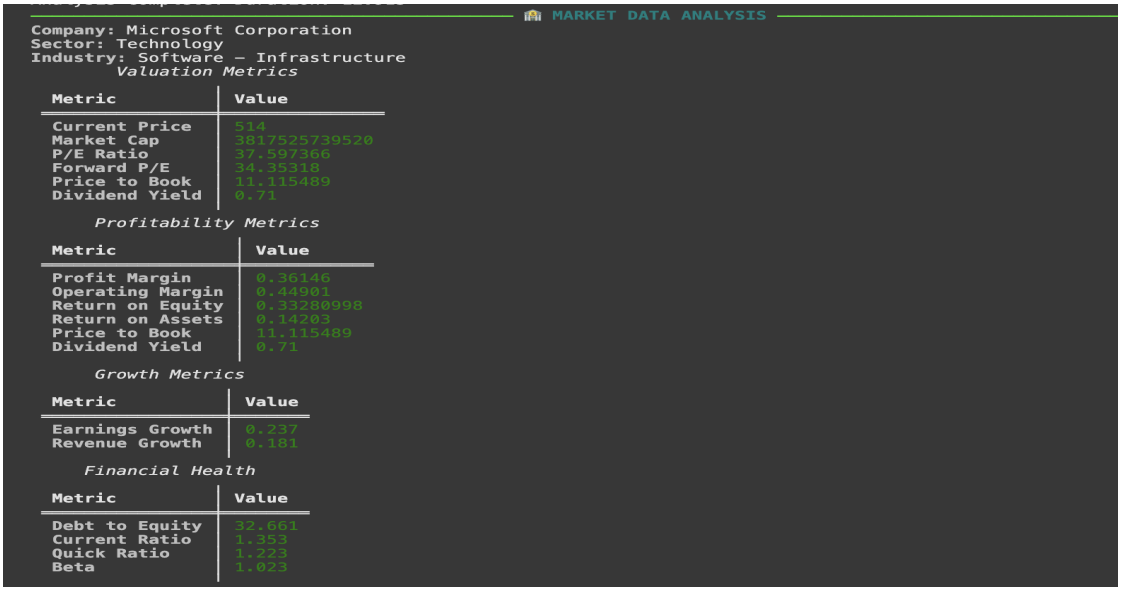
- Retrieves two years of historical stock price data using the Yahoo Finance API (yfinance).
- Extracts key financial metrics such as current price, market capitalization, P/E ratio, P/B ratio, dividend yield, beta, revenue growth, and profit margins.
- Collects company metadata, including sector, industry, and official company name.

**Outputs:**

A structured dictionary containing essential market indicators and company profile data.

Example outputs include:

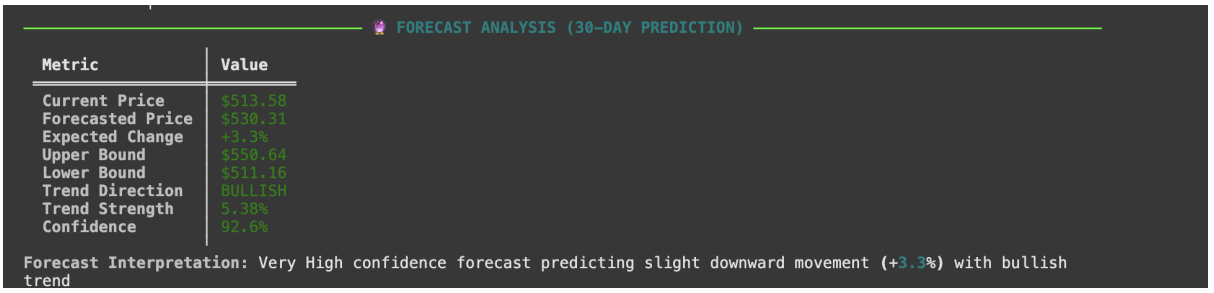
- Current Price
- Market Capitalization
- Earnings Growth
- Debt-to-Equity Ratio
- *Return on Assets (ROA)* and *Return on Equity (ROE)*
- *Sector* and *Industry* classification



**b. Prophet Forecast Agent**

The **Prophet Forecast Agent** predicts future stock price movements over a 30-day period. It leverages the *Prophet* library to detect weekly and yearly seasonality while handling missing data robustly.

- **Inputs:** Two years of stock price history.
- **Outputs:** Predicted prices, confidence intervals, and trend interpretation.
- **Visualization:** Generates time-series plots illustrating forecasts and uncertainty bounds.
- **Strength:** Offers interpretable, reliable predictions with quantified confidence.

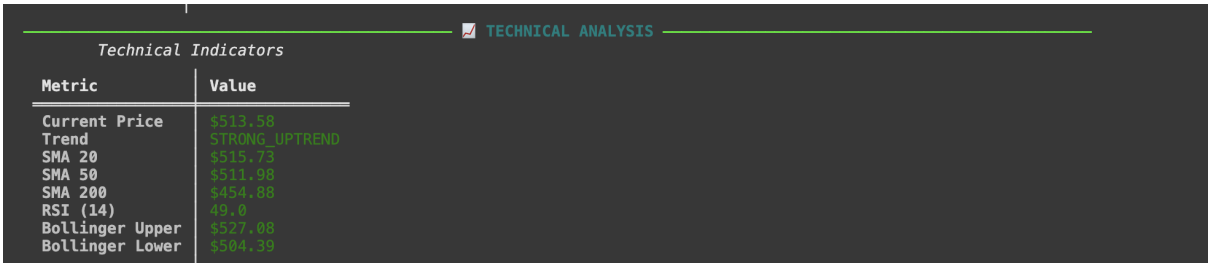



**c. Technical Analysis Agent**

This agent calculates key market indicators such as moving averages (MA), relative strength index (RSI), and volatility. By analyzing recent price actions, it classifies trends as bullish, bearish, or neutral.

- **Inputs:** Recent closing prices.
- **Outputs:** Metrics table and pattern-based insights.

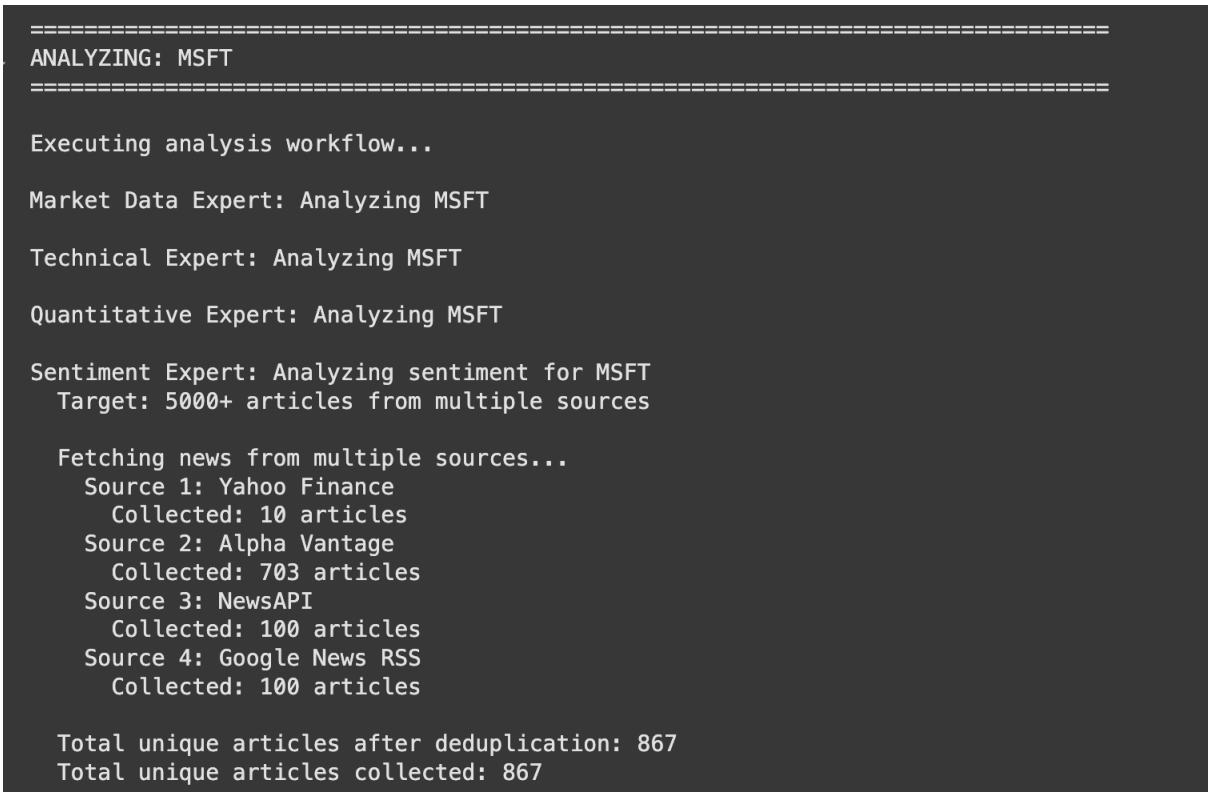
- **Purpose:** Support decision-making with evidence from historical patterns.

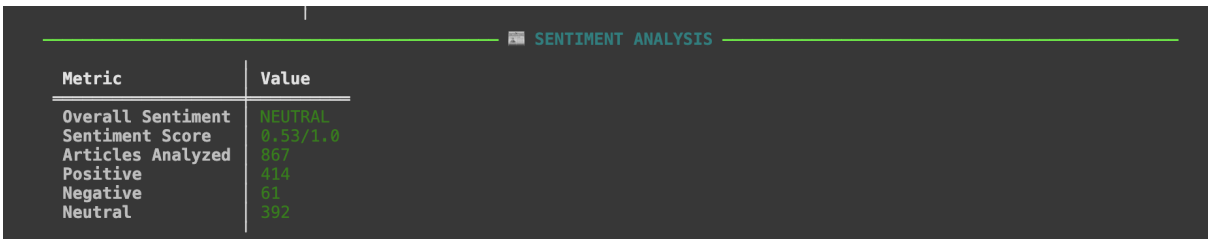



### d. Sentiment Analysis Agent

Using news data from RSS feeds and online articles, this agent performs sentiment classification. It employs *Feedparser* to gather text, and optionally integrates with *Google Generative AI* to refine tone detection.

- **Outputs:** Positive, negative, and neutral sentiment percentages.
- **Use Case:** Detects how media perception may influence investor behavior.






**e. Quantitative Analysis Agent**

This agent conducts numeric evaluation of company performance indicators for growth metrics such as Volatility, Sharpe Ratio and so on.

- **Outputs:** Table summarizing financial health and valuation.
- **Objective:** Provide data-grounded insights into stock stability and profitability.


**f. Sector Analysis Agent**

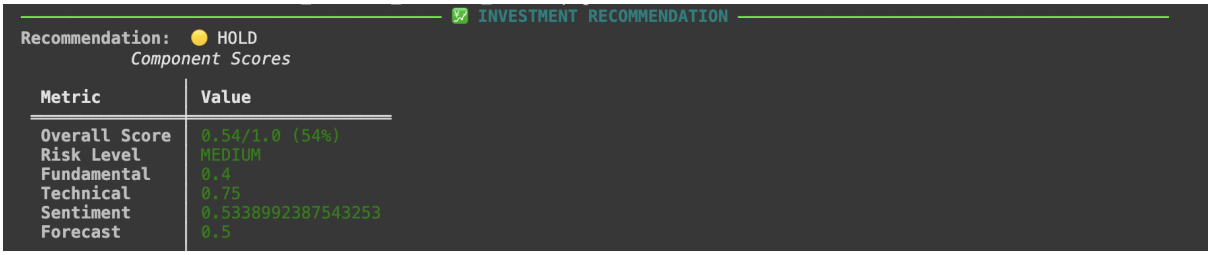
This agent compares the target company's performance with its industry peers. It highlights whether the company is outperforming or lagging behind sector averages.

- **Inputs:** Sector-level index and competitor metrics.
- **Outputs:** Comparative performance matrix.

### g. Synthesis and Recommendation Agent

Finally, this meta-agent aggregates outputs from all others. It uses weighted scoring logic to generate an overall recommendation — *Buy*, *Hold*, or *Sell*.

- **Outputs:** Unified summary report and justification.




## 3. User Interface Design

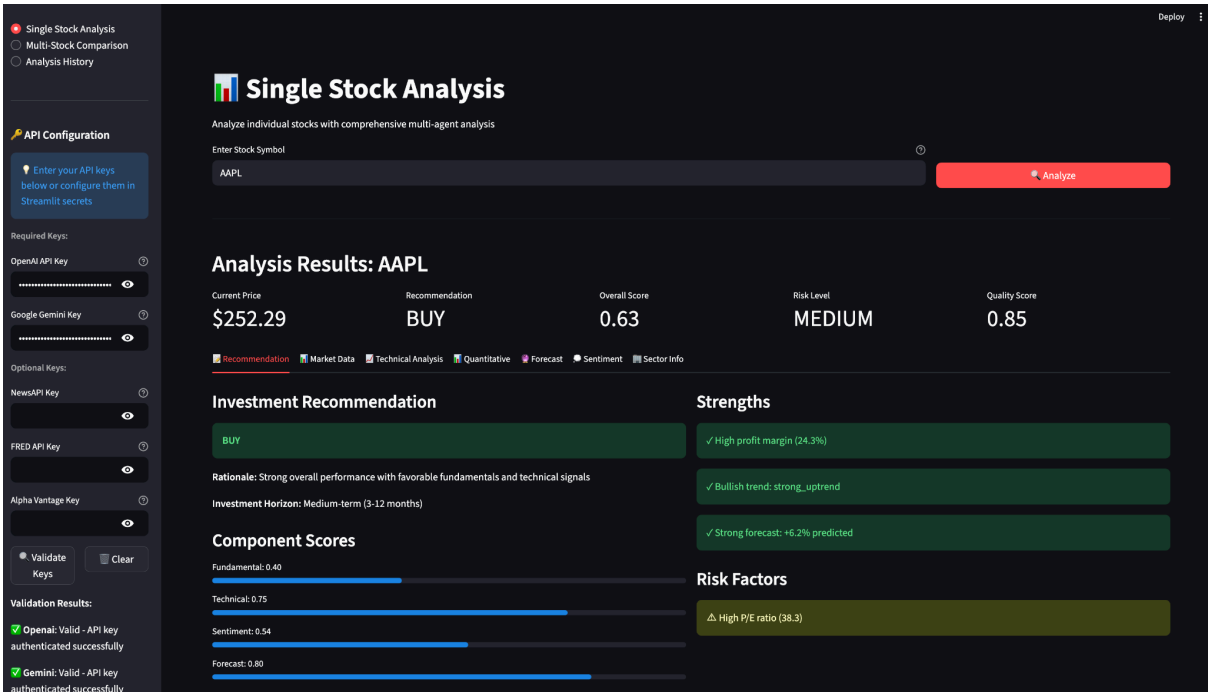
The user interface (UI) layer is implemented as an interactive **web application built with Streamlit**, offering a clean and responsive interface for exploring financial insights generated by the system's agents. Unlike traditional command-line dashboards, the Streamlit app allows for dynamic interaction through the browser, making the system highly accessible to both technical and non-technical users.

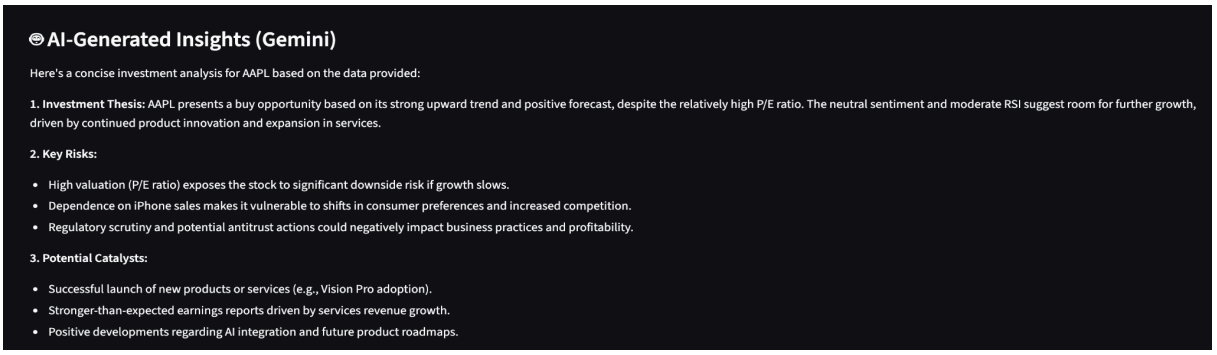
Key features of the Streamlit-based UI include:

- **Tabbed layout and collapsible sections** for intuitive navigation across different analytical views (e.g., market analysis, sentiment trends, sector insights, and investment recommendations).
- **Color-coded metrics and visual indicators**, such as green for upward trends and red for downward trends, to support quick comprehension of financial conditions.
- **Interactive charts** (e.g., line plots) powered by libraries such as Plotly and Matplotlib for time-series visualization, forecast projections, and portfolio performance tracking.
- **Natural language summaries** and explanations rendered using Markdown for enhanced interpretability and context.
- **User inputs and filters** (e.g., stock tickers, date ranges, sectors) that allow for customizable, on-demand analysis.

This web-based interface transforms the system from a backend automation tool into a fully interactive analytical platform. By abstracting away the complexity of the

underlying models and data pipelines, the UI ensures that insights are not only accurate but also **easily interpretable** and **actionable**—even for users with no programming background.



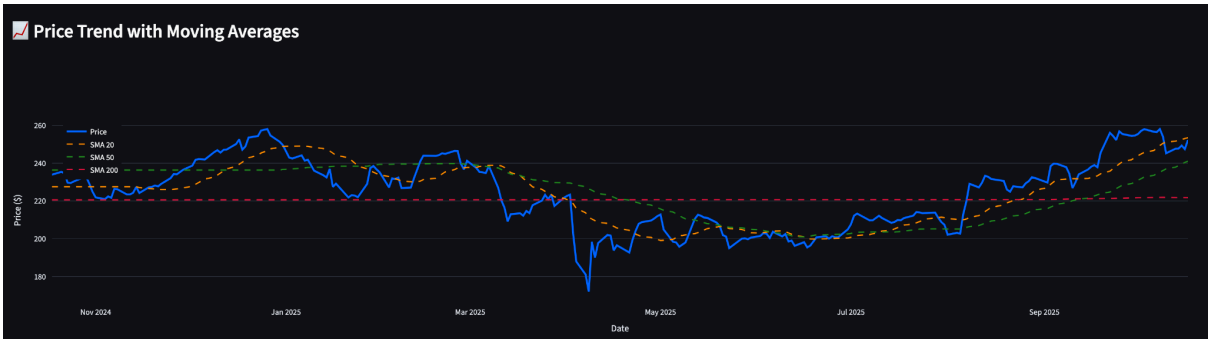
# Results and Discussion

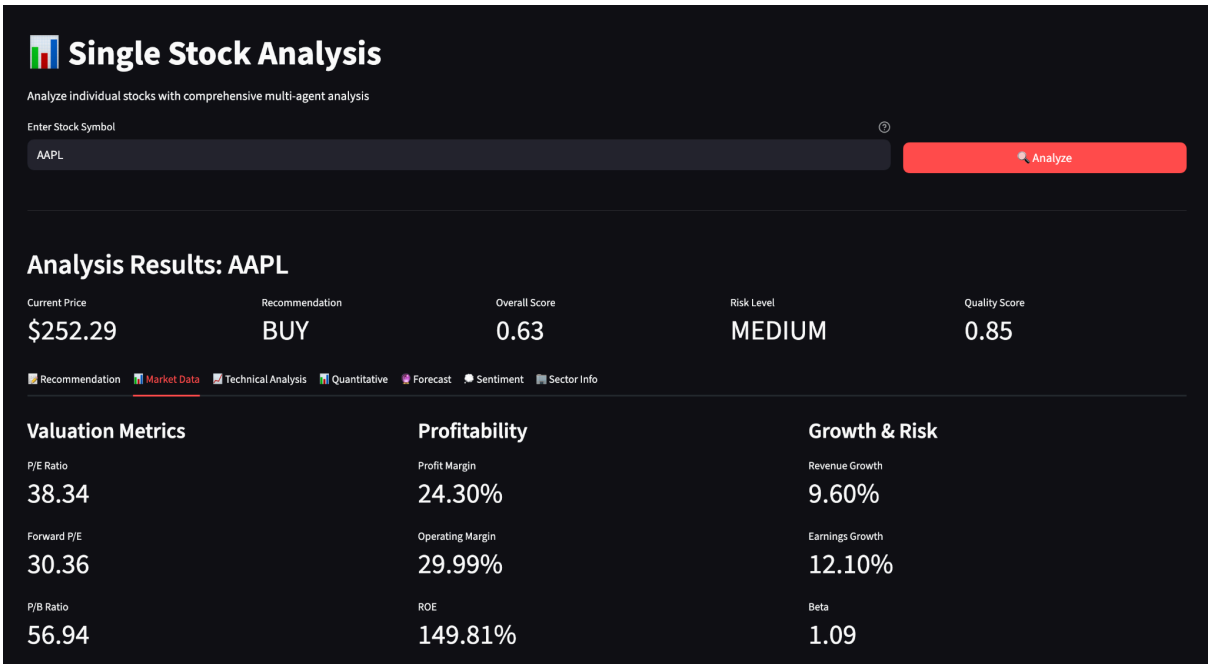
## 1. System Performance

The system was tested with several major stocks such as Apple (AAPL), Microsoft (MSFT), and Tesla (TSLA). In each case, the multi-agent design successfully integrated data from multiple sources and produced consistent recommendations.

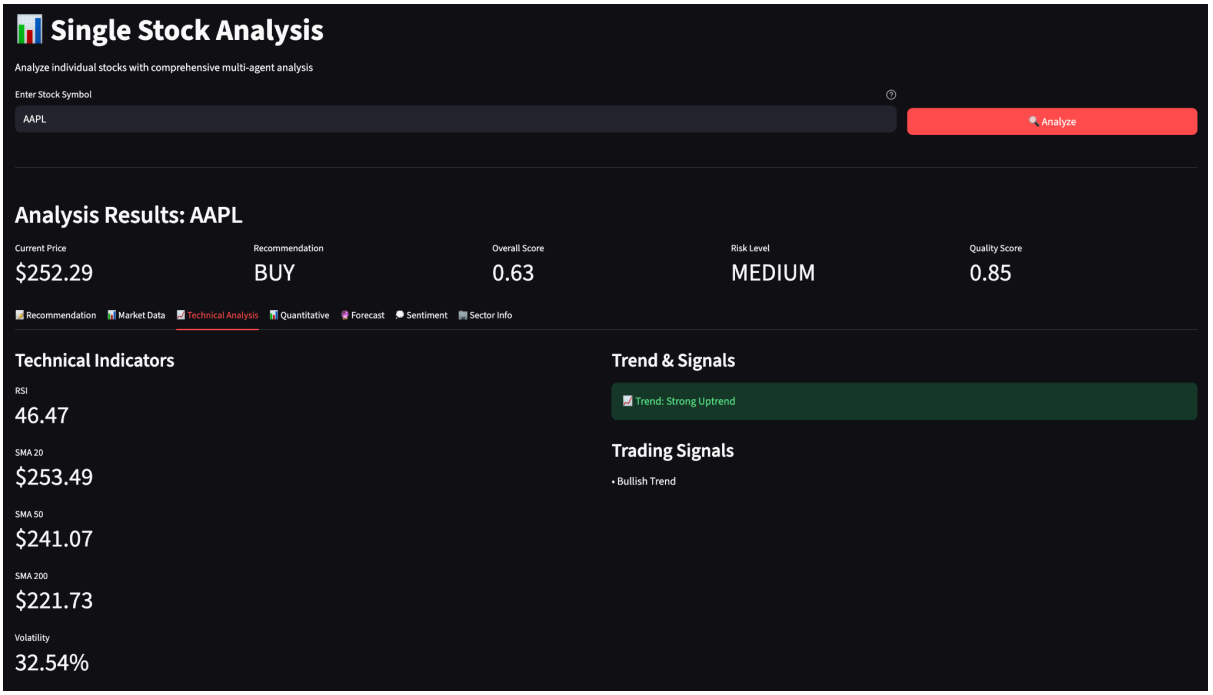
## 2. Example Outputs

### Market data Analysis

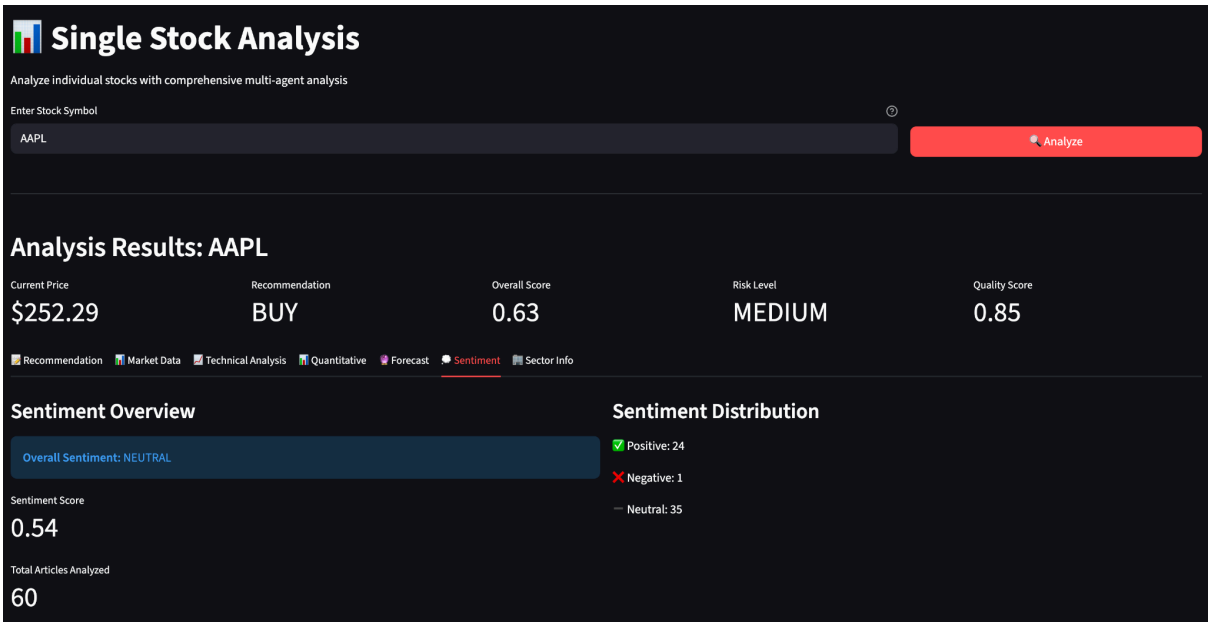


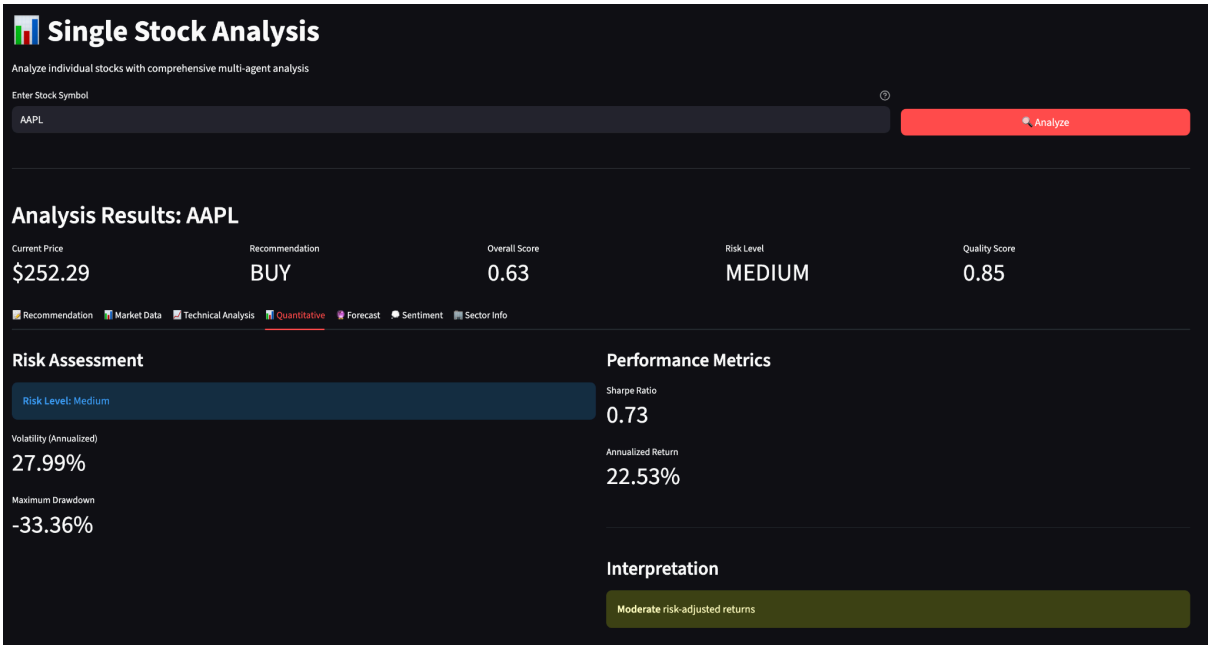
## Technical Analysis



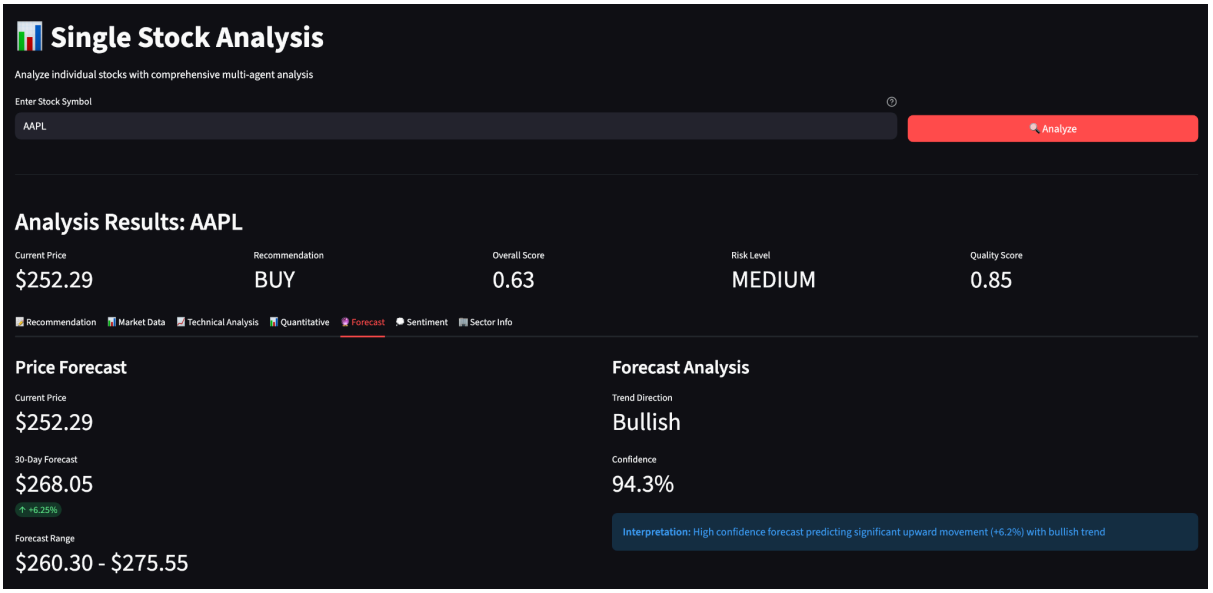

## Sentiment Analysis Results

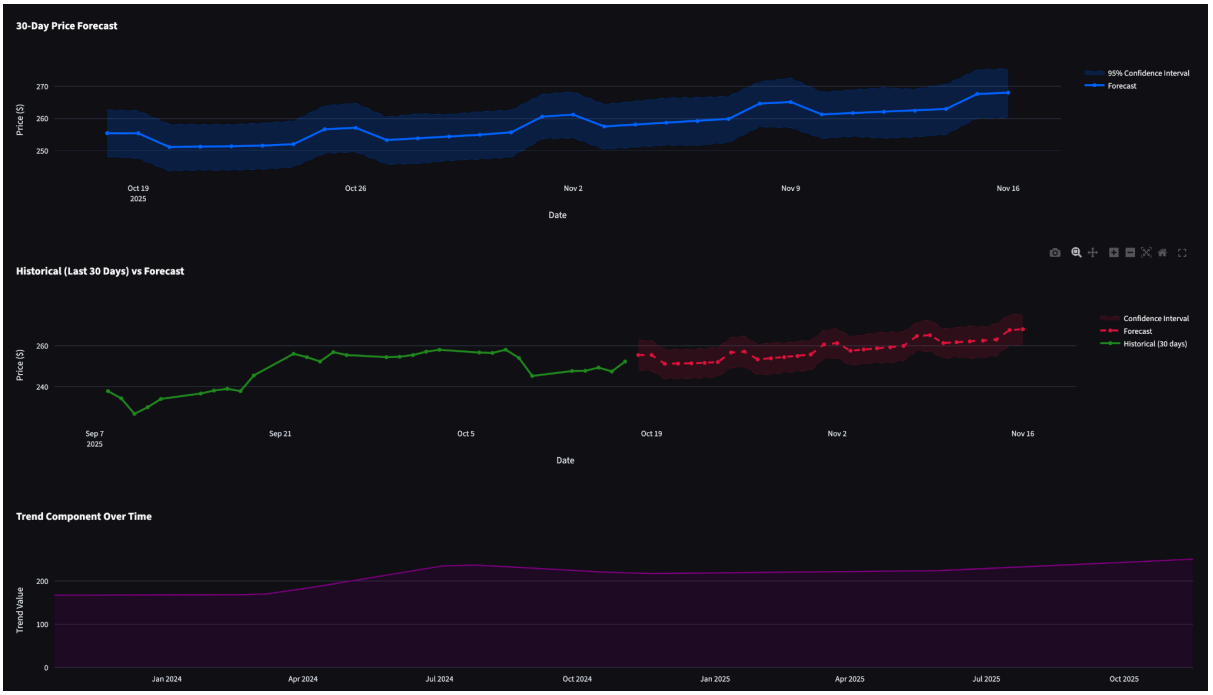


## Quantitative Analysis

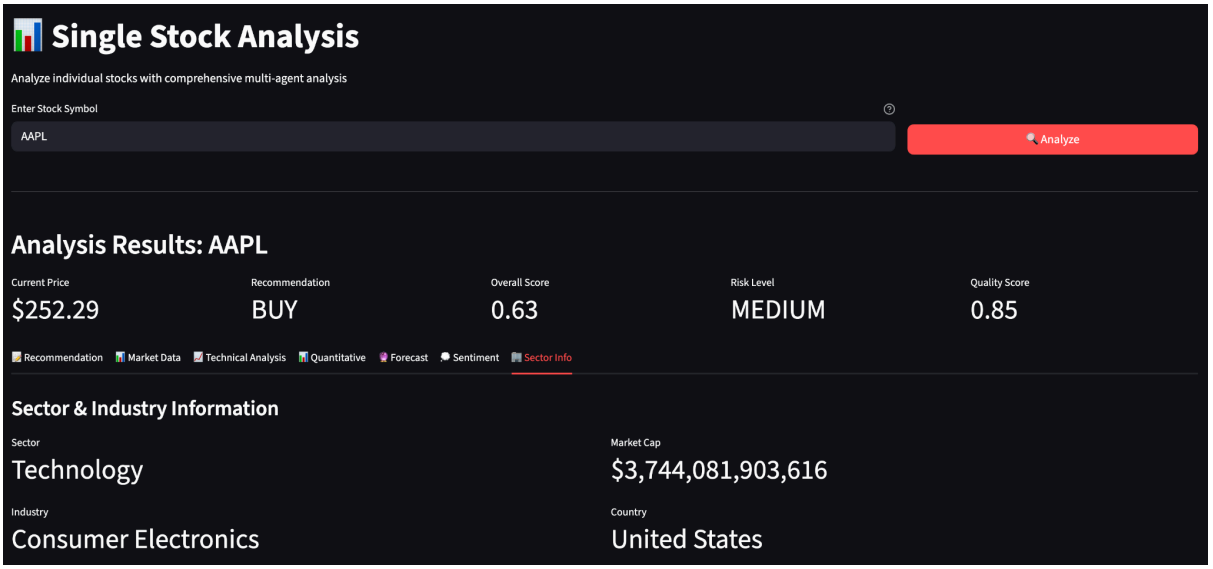



## Forecast



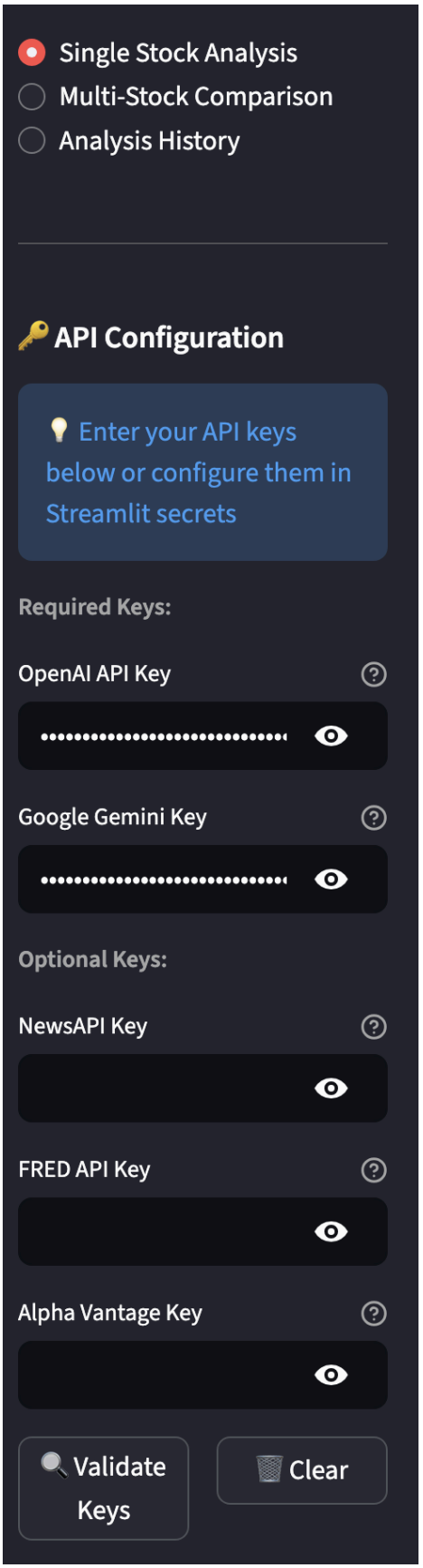



## Sector Information




The results show that combining different analytical views produces richer, more balanced recommendations than single-model approaches.

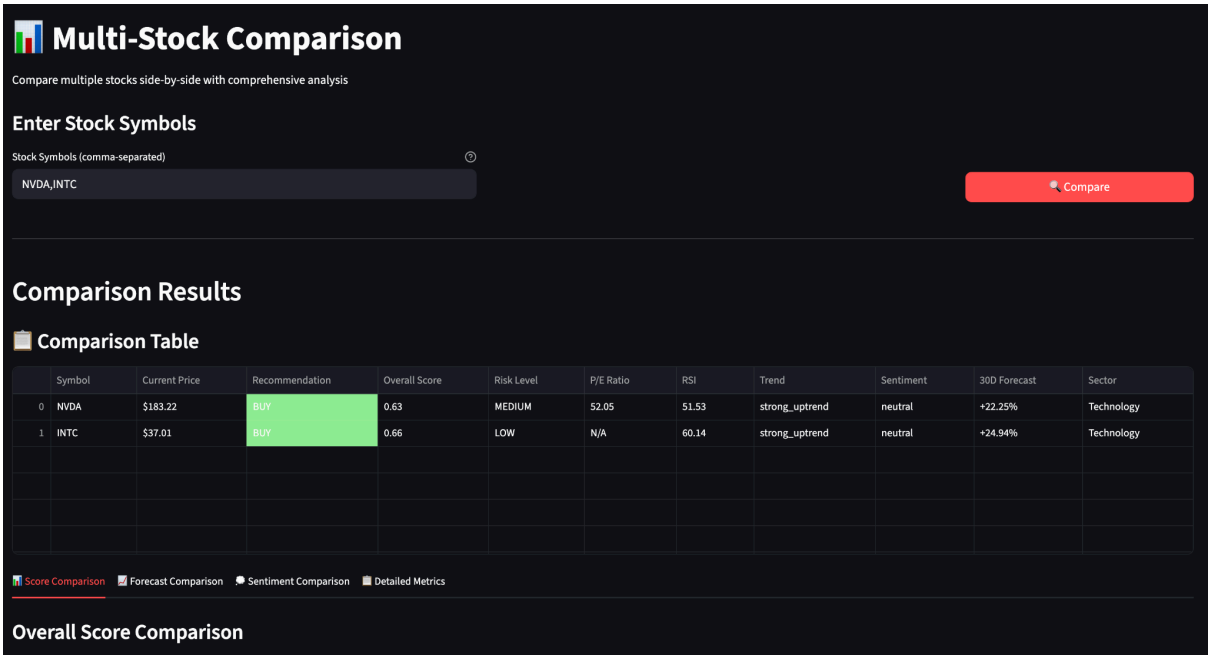
**Configurations Required for the webApp**

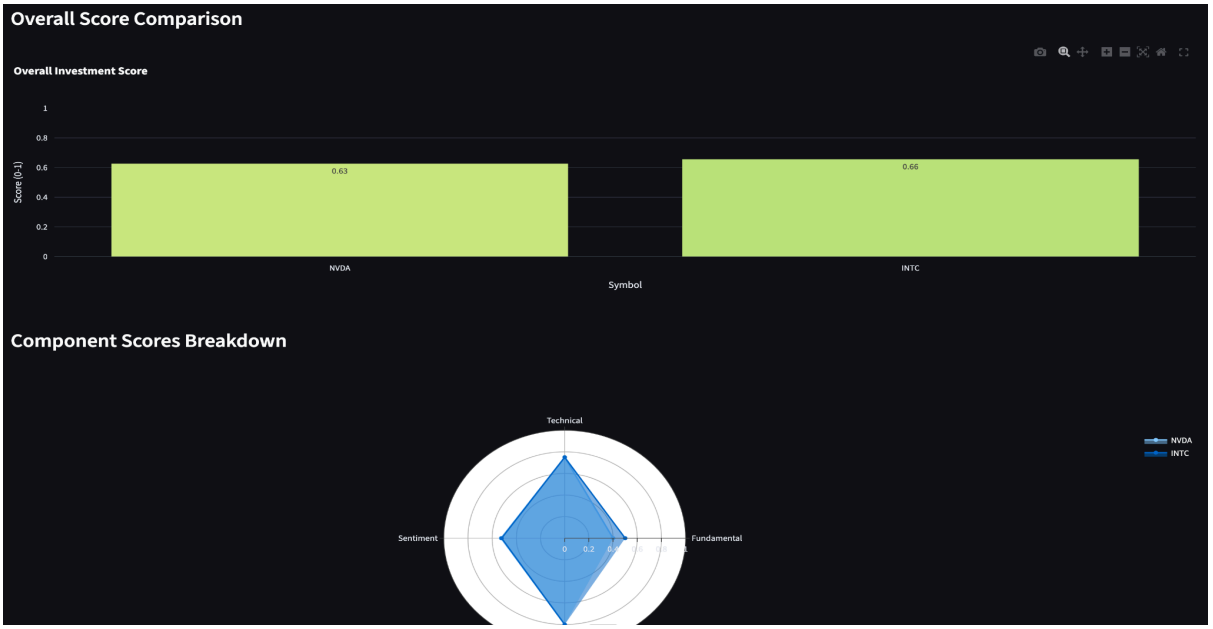


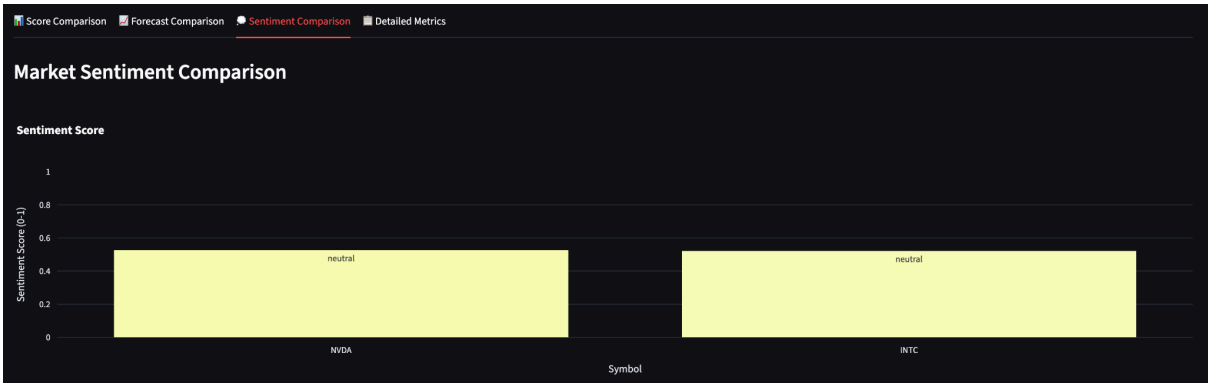
## 3. Interpretability

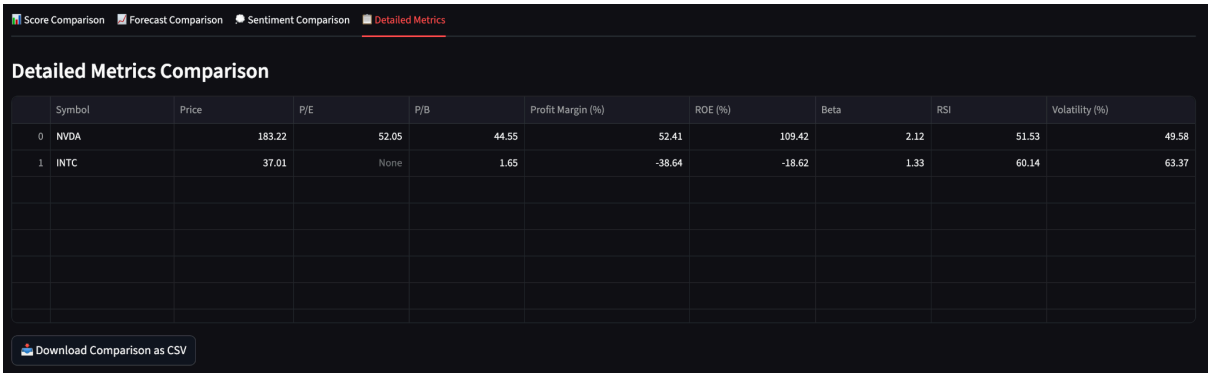
Unlike traditional machine learning models that act as black boxes, this system provides transparent explanations for each conclusion. Users can trace how a recommendation was formed by examining outputs from individual agents.

### Multi-stock comparison



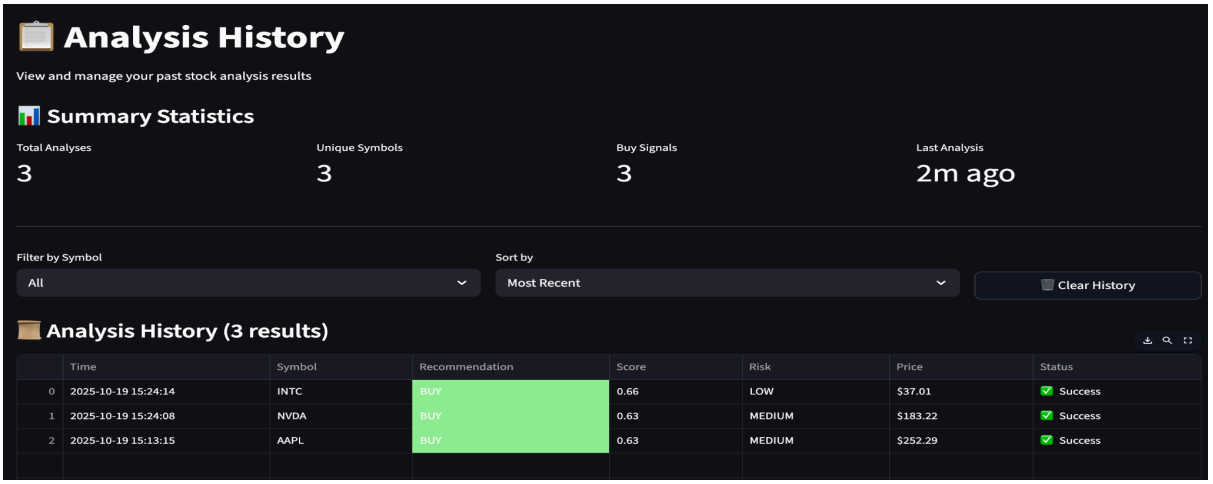




| | Symbol | Price | P/E | P/B | Profit Margin (%) | ROE (%) | Beta | RSI | Volatility (%) |
|---|---|---|---|---|---|---|---|---|---|
| 0 | NVDA | 183.22 | 52.05 | 44.55 | 52.41 | 109.42 | 2.12 | 51.53 | 49.58 |
| 1 | INTC | 37.01 | None | 1.65 | -38.64 | -18.62 | 1.33 | 60.14 | 63.37 |

## Analysis History




| | Time | Symbol | Recommendation | Score | Risk | Price | Status |
|---|---|---|---|---|---|---|---|
| 0 | 2025-10-19 15:24:14 | INTC | BUY | 0.66 | LOW | $37.01 | Success |
| 1 | 2025-10-19 15:24:08 | NVDA | BUY | 0.63 | MEDIUM | $183.22 | Success |
| 2 | 2025-10-19 15:13:15 | AAPL | BUY | 0.63 | MEDIUM | $252.29 | Success |

# Conclusion and Future Work

This project demonstrates a **functional, intelligent, and extensible architecture** for multi-agent financial analysis. By combining diverse analytical capabilities—such as forecasting, sentiment evaluation, technical trend detection, and sector-based comparison—the system provides a holistic understanding of market behavior. The design successfully bridges the gap between complex data science techniques and user-friendly decision support, enhancing both **accuracy** and **interpretability** in financial analytics.

The architecture's foundation lies in its **multi-agent design**, where each agent performs a specialized role within the analytical ecosystem. The agents communicate through a shared state structure that ensures coherence and traceability across the workflow. This design not only promotes modularity and scalability but also mirrors human financial analysis processes, where experts in different domains collaborate toward a unified investment decision.

## 1. Key Achievements

The completion of this project marks several significant milestones in the application of AI and automation to financial data analysis:

- **Automated Multi-Source Data Aggregation**

  The system achieves seamless integration of data from multiple online sources, including market APIs, news feeds, and historical repositories. This automation minimizes manual data collection, ensures real-time updates, and

improves analytical consistency—an essential feature for dynamic financial environments.

- **Modular, Agent-Based Workflow**

  Each analytical agent—whether performing forecasting, sentiment evaluation, or technical computation—functions autonomously within a shared state framework. This modularity enhances scalability, making it simple to add new analytical models or APIs without restructuring the entire system. The result is a flexible and future-proof analytical engine.

- **Intelligent Analytical Integration**

  A synthesis layer harmonizes the outputs from individual agents, combining quantitative forecasts with qualitative sentiment to produce balanced recommendations. This intelligent integration results in insights that are not only data-driven but also contextually aware, reflecting the multidimensional nature of real-world financial decision-making.

- **Web-App Deployment**

  The system was deployed as a web application using *Streamlit*. A browser-based interface is created which allows users to upload custom stock lists, visualize live data streams, and view forecasts in real time through interactive dashboards.

- **Reliable 30-Day Forecast Generation**

  The system's integration with *Prophet* enables accurate short-term forecasting, complete with upper and lower confidence intervals. Users can visualize projected price trajectories and assess volatility, enabling data-informed strategic decisions.

- **Transparency and Explainability**

  Each agent's intermediate outputs are clearly presented, allowing users to trace how final recommendations are formed. This transparency enhances trust and aligns the system with emerging ethical standards for explainable AI in finance.

- **Scalability and Extensibility**

  The underlying architecture is designed to grow. New agents—such as those for environmental, social, and governance (ESG) scoring or macroeconomic risk evaluation—can be integrated without significant code alterations. This flexibility ensures long-term adaptability for both research and enterprise use.

- **Enhanced Usability and Accessibility**

  By combining automation with clear presentation, the system caters to a broad audience. Financial professionals without programming experience can still benefit from AI-powered insights, transforming the tool into a practical decision support system rather than a purely technical prototype.

In summary, this project successfully integrates artificial intelligence, data engineering, and user interface design into a cohesive and interpretable financial analysis framework. The achievements collectively demonstrate that AI-driven insights can be both powerful and comprehensible when supported by modular architecture and thoughtful visualization.

## 2. Future Enhancements

While the system currently performs effectively in producing accurate, explainable analyses, several potential improvements can elevate its performance and usability:

- **Integration with Conversational AI (LangChain)**

  Incorporating *LangChain* or similar frameworks will allow users to interact with the system through natural language queries. For example, users could ask, “What is the 30-day outlook for Tesla?” or “Compare Apple’s sentiment trends with Microsoft’s.” Such conversational interaction will make the system even more accessible and intuitive.
- **Real-Time Risk and Portfolio Optimization Modules**

  Adding agents dedicated to portfolio optimization and risk simulation would enable users to test hypothetical investment scenarios. By integrating models such as Value-at-Risk (VaR) or Monte Carlo simulations, the system could transition from analytical reporting to decision optimization.
- **API-Based Data Expansion**

  Incorporating additional APIs for financial data would further diversify the data sources, improving both reliability and analytical depth. This expansion would also allow coverage of more asset classes, including cryptocurrencies, commodities, and global indices.
- **Cloud and Edge Deployment Options**

  To ensure scalability and performance, the system could be deployed on cloud platforms with edge computing capabilities. This would facilitate real-time analysis for enterprise clients and reduce latency in data retrieval and forecasting.

# References


Han, X., Wang, N., Che, S., Yang, H., Zhang, K., & Xu, S. X. (2024). *Enhancing investment analysis: Optimizing AI-agent collaboration in financial research*. arXiv. https://arxiv.org/abs/2411.04788 arXiv+1

Park, T. (2024). *Enhancing anomaly detection in financial markets with an LLM-based multi-agent framework*. arXiv. https://arxiv.org/abs/2403.19735 arXiv

Li, Y., Wu, C., & Liu, J. (2014). A combination prediction model of stock composite index based on artificial intelligent methods and multi-agent simulation. *International Journal of Computational Intelligence Systems*, 7, 853–864. https://doi.org/10.1080/18756891.2013.876722 SpringerLink

Ayyanar, S. (2024). Building multi-AI agent systems for complex financial analysis: A comparative study of all open source frameworks. *ESP International Journal of Advancements in Computational Technology (ESP-IJACT)*, 2(4), 120–124. ESP Journals

Liapis, C. M., Karanikola, A., & Kotsiantis, S. (2023). Investigating deep stock market forecasting with sentiment analysis. *Entropy*, 25(2), 219. https://doi.org/10.3390/e25020219 MDPI

Deep, A. (2024). Advanced financial market forecasting: Integrating Monte Carlo simulations with ensemble machine learning models. *Quantitative Finance and Economics*, 8(2), 286-314. https://doi.org/10.3934/QFE.2024011 AIMS Press

Facebook. (2017). *Prophet: Forecasting at scale*. https://facebook.github.io/prophet/

Python Software Foundation. (2024). *Rich: Python library for rich text and formatting in the terminal*. https://rich.readthedocs.io/

Yahoo Finance. (2024). *yfinance Python library documentation*. https://pypi.org/project/yfinance/

Google AI. (2024). *Google Generative AI API Documentation*. https://ai.google.dev

LangChain. (2024). *LangGraph and LangChain framework documentation*. https://python.langchain.com

Taylor, S. J., & Letham, B. (2018). *Forecasting at scale*. The American Statistician, 72(1), 37–45.