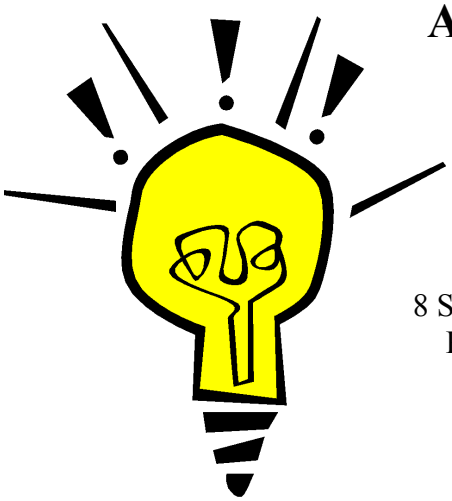
Boston University
Electrical & Computer Engineering
EC464 Capstone Senior Design Project

User's Manual

AI Trading Platform



Submitted to

Self-designed
Osama Alshaykh
8 St. Mary's St Boston, MA 02215,
Photonics Building Room 430
(617) 353-0223
osama@bu.edu

by

Team #6
AI Trading Platform

Team Members

Adrian Pawlowski | ajp12@bu.edu
Gabriel Brown | gabrown@bu.edu
Yagiz Idlman | ykaan@bu.edu
Odilon Quevillon | odilon@bu.edu

Submitted: 4/18/2025

AI Trading Platform User’s Manual

Table of Contents

Executive Summary

This project develops a trading platform that delivers actionable buy or sell outputs for selected stocks. By integrating multiple machine learning techniques, the platform provides a sentiment analysis score, an LSTM-based price prediction, an XGBoost classification, and a reinforcement learning-generated trading signal. Based on the analysis of previous day outputs, the system adopts a simple trading strategy of holding stocks from market open to close the following day. This approach not only simplifies decision-making for users but also serves as a proof-of-concept for combining diverse analytical models to generate coherent trading insights.

1 Introduction

Individual investors today face a rapidly evolving market landscape defined by high volatility, vast data streams, and increasingly sophisticated algorithmic strategies. Our AI Trading Platform directly addresses the core challenges of:

- **Information Overload:** Filtering news, technical indicators, and price histories into concise buy/hold/sell signals.
- **Unpredictable Market Behavior:** Adapting in real time to sudden swings in high‑volatility stocks like TSLA and NVDA.
- **Accessibility of Advanced Tools:** Bringing machine‑learning models, once reserved for institutional traders, into a simple web interface.

Project Purpose & Solution

The platform empowers retail investors with actionable insights across major equities (e.g., AMZN, MSFT, JPM) by combining:

1. **LSTM Forecasting:** A two‑layer LSTM network that analyzes 20‑day lookbacks to predict next‑day price changes with an average directional accuracy of 60%.
2. **XGBoost Classification:** A gradient‑boosted tree ensemble tuned to 200 estimators, achieving a 53% directional accuracy on recent test data.
3. **NLP Sentiment Analysis:** A BERT‑based pipeline converting financial news and social media into rolling sentiment scores that correlate (r = 0.016) with subsequent returns.
4. **Reinforcement Learning Allocation:** A DQN agent that integrates LSTM, XGBoost, and sentiment inputs to generate trading actions, producing an annualized live return of 278% on a 7‑day trial.

By fusing these technologies, our decision layer dynamically weighs each model's signal, then executes trades—historically delivering +$81.74 (LSTM) and +$69.18 (RL) in live profits over one week, while maintaining a robust Sharpe Ratio (> 0.29).

Context & Technology

This project leverages:

- Cloud Compute & Colab for model training;
- Firebase Realtime Database for storing and streaming signals;
- Java Spring Boot API to serve model outputs securely over HTTPS;
- React + TypeScript Front-End with Tailwind CSS for responsive dashboards displaying up‑to‑the‑minute recommendations.

These components ensure that end users—from novice investors to seasoned traders—can access institutional-grade analytics via any modern browser, without installation.

Safety, Security & Limitations

- **Data Privacy:** All API keys and user interactions are secured via TLS 1.3 and Firebase security rules.
- **Model Risk:** Automated signals carry execution risk; users should verify trades against market conditions and be aware of slippage and transaction costs.
- **Continuous Monitoring:** Live trading is enabled but should be overseen to guard against systemic failures or connectivity issues**.**

Document Roadmap

Section 2 details system setup and installation. Section 3 walks through normal and abnormal operation modes. Section 4 presents the technical background of our models. Section 5 lists applicable engineering standards. Section 6 outlines cost considerations, and Section 7 collects appendices with final specifications and team information.

2 System Overview and Installation

2.1 Overview block diagram

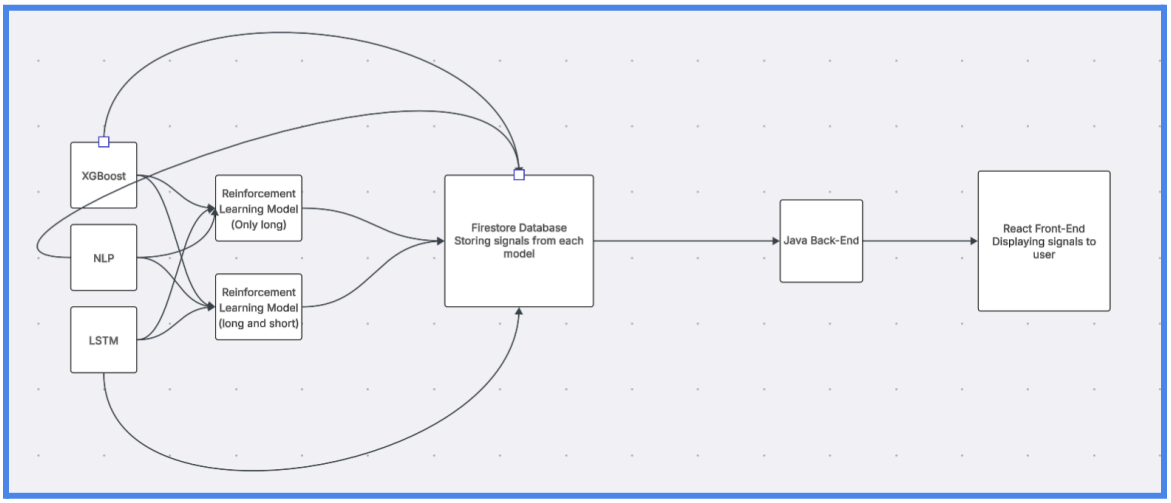
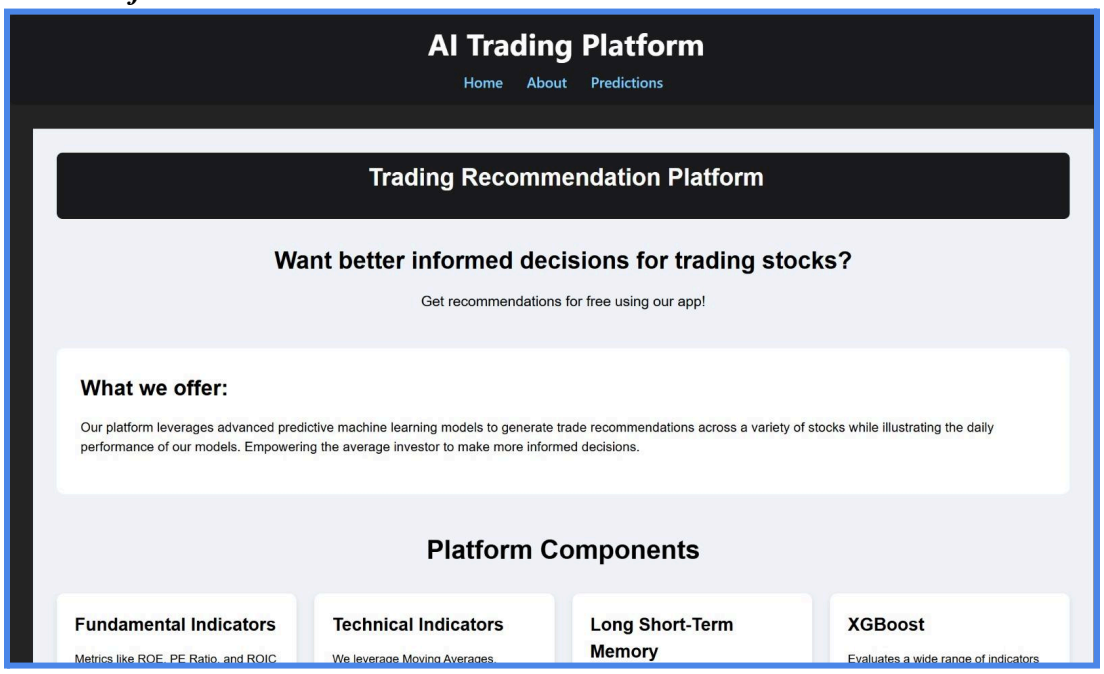
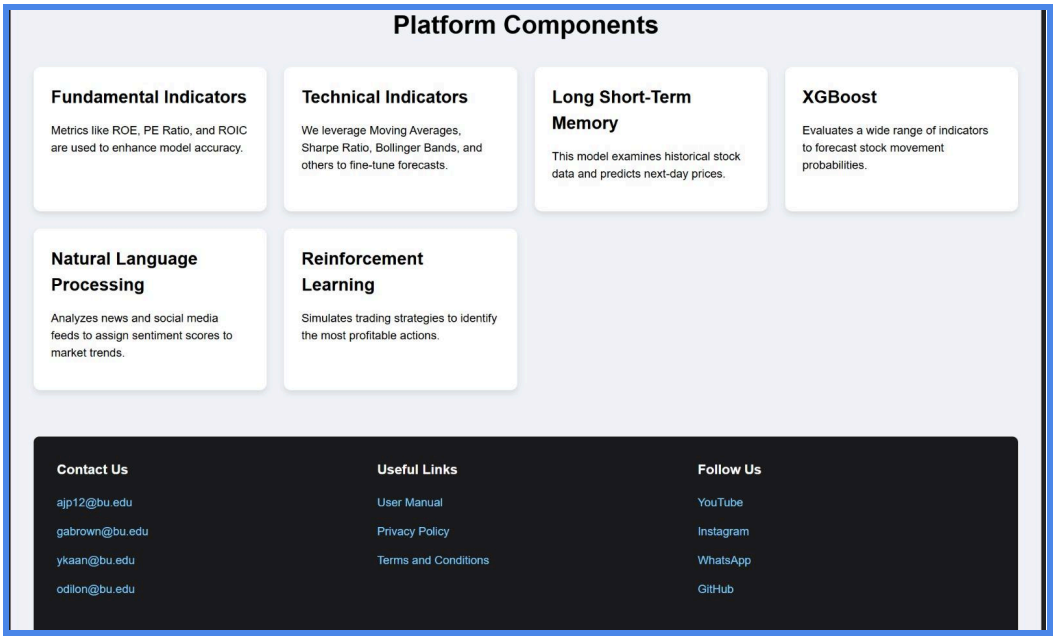
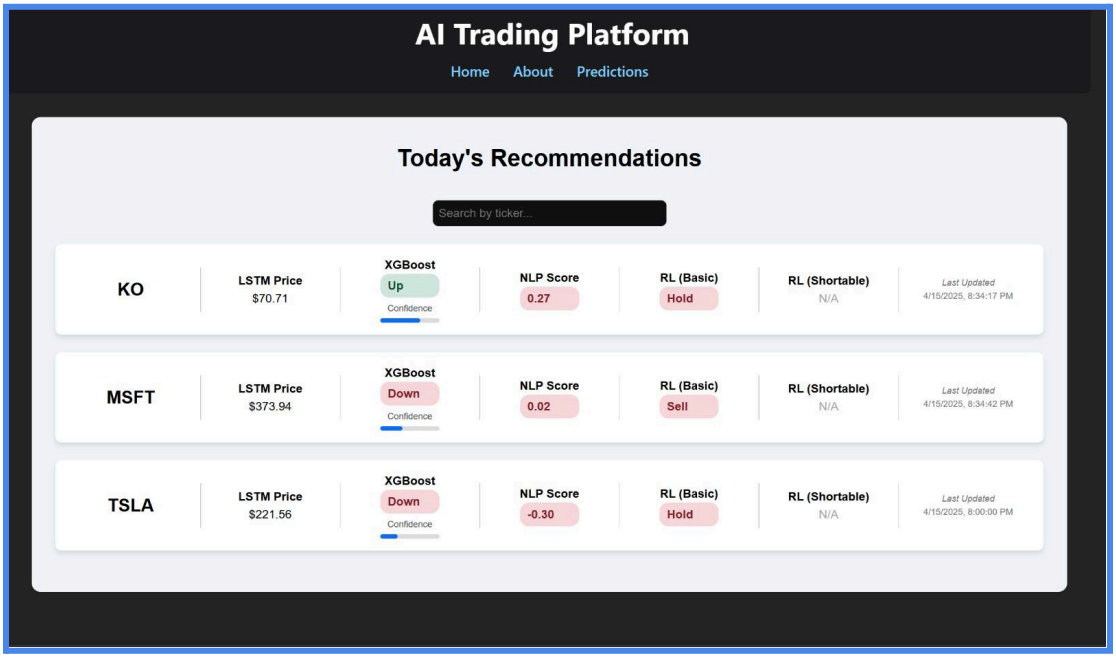


Figure 1: AI Trading Model Architecture

2.2 User interface.




Platform Components
Fundamental Indicators
Metrics like ROE, PE Ratio, and ROIC are used to enhance model accuracy.
Technical Indicators
We leverage Moving Averages, Sharpe Ratio, Bollinger Bands, and others to fine-tune forecasts.
Long Short-Term Memory
This model examines historical stock data and predicts next-day prices.
XGBoost
Evaluates a wide range of indicators to forecast stock movement probabilities.
Natural Language Processing
Analyzes news and social media feeds to assign sentiment scores to market trends.
Reinforcement Learning
Simulates trading strategies to identify the most profitable actions.
Contact Us
ajp12@bu.edu
gabrown@bu.edu
ykaan@bu.edu
odilon@bu.edu
Useful Links
User Manual
Privacy Policy
Terms and Conditions
Follow Us
YouTube
Instagram
WhatsApp
GitHub


AI Trading Platform
Home
About
Predictions
Today's Recommendations
Search by ticker...
KO
LSTM Price
$70.71
XGBoost
Up
Confidence
NLP Score
0.27
RL (Basic)
Hold
RL (Shortable)
N/A
Last Updated
4/15/2025, 8:34:17 PM
MSFT
LSTM Price
$373.94
XGBoost
Down
Confidence
NLP Score
0.02
RL (Basic)
Sell
RL (Shortable)
N/A
Last Updated
4/15/2025, 8:34:42 PM
TSLA
LSTM Price
$221.56
XGBoost
Down
Confidence
NLP Score
-0.30
RL (Basic)
Hold
RL (Shortable)
N/A
Last Updated
4/15/2025, 8:00:00 PM

2.3 Installation, setup, and support

The AI Trading Platform is a fully web-based application. No installation is required to use the platform.

You can access the platform at any time by visiting:

https://stockpredictionplatform.com

We recommend using a modern browser such as:

- Google Chrome
- Mozilla Firefox
- Microsoft Edge
- Safari (latest version)

The site is fully responsive and works on both desktop and mobile devices.

Getting Started

1. Open your browser.
2. Navigate to https://stockpredictionplatform.com
3. Use the navigation bar at the top to explore:
    - Home – Overview of the platform
    - About – Learn about the technology and team
    - Predictions – View today's model-generated trading recommendations

No login, signup, or download is necessary.

3 Operation of the Project

3.1 Operating Mode 1: Normal Operation

Normal operation involves accessing the platform, viewing model-generated trading recommendations, and interpreting them for external use (e.g., in a brokerage account). The platform is hosted on the following website: with the front-end displaying daily recommendations and the back-end API serving model outputs.

Overview of Normal Operation:

In this mode, users:

- Access the Predictions Page to view daily trading recommendations (buy, sell, or hold) for each stock, generated by the LSTM, XGBoost, and Reinforcement Learning models.
- Interpret recommendations, including confidence scores, to inform manual trading decisions outside the platform.
- Stay up to date with recommendations to ensure their trading actions reflect the latest model outputs.

User Interface and Options:

The React-based front-end, accessible via a web browser, includes:

- Predictions Page: A table displaying recommendations with:
- Stock ticker (e.g., TSLA, AMZN).
- Model (LSTM, XGBoost, RL).
- Recommendation (Buy, Sell, Hold).
- Confidence score (0 to 1, indicating model certainty).
- Timestamp of the recommendation.

Steps for Normal Operation:

Step 1: Access the platform website

Verify the Predictions Page loads with recommendations for all 10 stocks.

Step 2: Review Recommendations

Navigate to the Predictions Page.

Prioritize high-confidence recommendations (e.g., confidence >0.7) for decision-making.

Step 3: Interpret Recommendations

Note the recommendation (e.g., “Buy” for TSLA from RL) and confidence score.

Use this information to inform manual trades in an external brokerage account (not handled by the platform).

Step 4: Review Recommendations

Stay attentive to daily recommendations to avoid missing opportunities.

Exiting Normal Operation:

Close the browser.

Note: The platform does not execute trades or store data, so no additional steps are needed to cease operations.

3.2 Operating Mode 2: Abnormal Operations

Overview of Abnormal Operations:

Abnormal states include:

- Model Failure: A model (e.g., LSTM) fails to produce a recommendation due to insufficient data or computational errors.
- Data Issues: Stock price data is missing, outdated, or corrupted, leading to unreliable recommendations.
- API Failure: The back-end API server crashes or becomes unresponsive.

Potential solution: Diagnostic Mode (Not Yet Implemented)

When implemented, diagnostic mode will:

- Access: Be activated via the Settings Panel (“Run Diagnostics” button).
- Functionality:
- Verify API connectivity.
- Check stock data integrity (e.g., validate CSV files).
- Test model outputs with a sample prediction.

Output: Display a diagnostic report on the front-end, listing errors (e.g., “XGBoost model failed: missing AMZN data”) and suggested fixes.

4 Technical Background

1. Overall System Architecture

The platform is organized into three principal layers:

1. **Data Layer**
2. **Modeling Layer**
3. **Interface & Execution Layer**

Each layer communicates through well‑defined APIs and shared configuration files, ensuring modularity and extensibility.

2. Data Layer

2.1 Market Data Ingestion

- **Source**: Historical and real‑time time series (prices, volumes, fundamentals) are fetched from Alpha Vantage via its REST API.
- **Preprocessing**: JSON responses are parsed into pandas DataFrames, resampled to uniform daily intervals, and missing values forward‑filled.
- **Technical Indicators**: A suite of ~65 indicators is computed in data_processing.py, including:
    - **Moving Averages**: SMA, EMA
    - **Oscillators**: RSI, CCI, Stochastic, MACD
    - **Volatility**: Bollinger Bands, ATR
    - **Volume Metrics**: On‑Balance Volume, A/D Oscillator

All computations use vectorized NumPy operations for performance.

2.2 Fundamental & Sentiment Augmentation

- **Fundamental Data**: Key ratios (P/E, ROE, Beta, Profit Margin, EPS) are retrieved alongside price data from Alpha Vantage, normalized, and merged into the feature set.
- NLP Sentiment: A BERT‑based pipeline (HuggingFace bert-base-uncased-finetuned-financial) processes news headlines. Each article yields a sentiment score, which is aggregated over a 3‑day rolling window to create a quantitative sentiment feature.

3. Modeling Layer

3.1 LSTM Price Predictor

- **Principle**: LSTM networks capture temporal dependencies through gated memory cells, mitigating vanishing/exploding gradients.
- **Architecture**:

    - **Input**: 20‑day lookback of normalized adjusted close plus indicators.
    - **Network**: Two stacked LSTM layers (128 units, dropout 0.2) → Dense output predicting next‑day price change.
    - **Training**: MSE loss, Adam optimizer, early stopping on validation loss.
- **Output**: Next‑day price estimate, used standalone and as an input to downstream models.

3.2 XGBoost Signal Classifier

- **Principle**: Gradient boosting builds an ensemble of trees in stage‑wise fashion, optimizing binary logistic loss.
- **Features**: Combines LSTM forecast, raw indicators, and sentiment scores.
- **Tuning**: Bayesian optimization (learning rate=0.05, max_depth=6, n_estimators=200).
- **Output**: Probability of price increase vs. decrease, thresholded at 0.5 for discrete buy/sell signals.

3.3 Reinforcement Learning Allocator

- **Principle**: Formulated as an MDP where the agent maximizes cumulative reward (P/L) under market uncertainty.
- **Algorithm**: Proximal Policy Optimization (PPO) via Stable Baselines3.
- **State**: Concatenation of:
    - Recent normalized price & indicators
    - Latest LSTM prediction
    - XGBoost probability
    - Rolling sentiment score
    - Current portfolio weights
- **Reward**: Realized P/L minus transaction cost penalty (0.1% per trade) to prevent overtrading.
- **Training Data**: Historical period 2008–2021; evaluation out‑of‑sample on Jan 1, 2024–Apr 1, 2025.

4. Backtesting Engine

- **Orchestration**: A Python script sequentially runs: fetch → feature computation → LSTM predict → XGBoost signal → RL allocation → simulate trades.
- **Simulation Rules**:
    - Trades executed at next‑day open.
    - Position sizes from RL output; P/L includes slippage & commission.
- **Period**: Backtests cover Jan 1, 2024 through Apr 1, 2025.
- **Metrics**: Cumulative P/L, annualized return, Sharpe ratio, max drawdown, and classification accuracy.
- **Outputs**: CSV logs of daily portfolio value and a PDF report with equity curves and feature‑importance charts.

5. Interface & Execution Layer

5.1 Java Backend Service

- **Implementation**: A Spring Boot microservice written in Java.
- **Function**: Starts the service, fetches processed data and model outputs from Firebase Realtime Database, and serves them via HTTP endpoints.
- **Endpoints**:
    - predictions

5.2 Frontend Dashboard

- **Framework**: React + TypeScript (Vite).
- **Features**:
    - Live portfolio performance visualization (Recharts).
    - Model signals display (forecast, probability, sentiment gauge).
    - Controls for starting/stopping live trading and adjusting risk parameters.
- **Styling**: Tailwind CSS for responsive layouts.

---

6. Governing Principles & Laws

- **Time‑Series Forecasting**: Relies on autocorrelation structures, captured by LSTM's memory gates.
- **Ensemble Learning**: Reduces variance through gradient boosting's iterative corrections.
- **Reinforcement Learning**: Frames trading as sequential decision‑making under uncertainty, optimizing expected return.

5 Relevant Engineering Standards

Our AI Trading Platform integrates real-time financial data analysis, predictive modeling, and full-stack deployment using React for the frontend and a Java backend connected to Google Firestore. As such, the design and implementation of the system involved multiple engineering standards and best practices across the domains of software development, internet protocols, and data security.

1. Software Design and Coding Standards

On the backend, we followed Java Coding Conventions, ensuring our code was readable, modular, and maintainable. Key practices included the use of meaningful naming conventions and the proper use of object-oriented principles (encapsulation, abstraction, and inheritance).

For the frontend, React development best practices were followed as outlined in community standards. This ensured predictable component behavior, modular CSS-in-JS patterns, reusable UI components, and performance optimization.

2. Data Storage and Communication Protocols

The system uses Google Firestore, a NoSQL document-based cloud database that complies with standards for information security management. Communication between our backend and Firestore is handled over secure HTTPS connections via REST API calls, conforming to TLS 1.2/1.3 protocols for data-in-transit encryption.

All client-server communication adheres to the HTTP/1.1 and HTTP/2 specifications, ensuring reliable request/response cycles and reduced latency for real-time prediction delivery.

3. Internet and Web Application Standards

We followed W3C guidelines for front-end development to ensure cross-browser compatibility, accessibility, and responsiveness. This includes compliance with:

- HTML5 and CSS3 specifications
- WCAG (Web Content Accessibility Guidelines) for accessible UI components

React's virtual DOM and component-based architecture were used in line with modern single-page application (SPA) patterns, enabling real-time interactivity and efficient routing without full page reloads.

4. Cybersecurity and Data Protection

Given that the platform handles financial data and predictive analytics, we adopted security practices. These include:

- HTTPS enforcement for all frontend-backend communication
- Secure handling of API keys using environment variables
- Firebase security rules to restrict unauthorized database access

6 Cost Breakdown

This section outlines the estimated costs for producing the beta version. The beta version follows the alpha prototype (demonstrated in class) and assumes market costs for all components and services, with no donations or reused parts. Costs are aggregated into key categories: Google Colab, API data, domain, and hosting. The platform consists of a React-based front-end, a Java back-end API, and machine learning models running on Google Colab.

Project Costs for Production of Beta Version				
Item	Quantity	Description	Unit Cost	Extended Cost
1	1	Google Colab Pro+ Subscription (1 year)	$600	$600
2	1	Stock Market Data API (1 year)	$300	$300
3	1	Polygon API (1 year)	$300	$300
4	1	AWS Lightsail Cloud Hosting (1 year)	$114	$114
5	1	Domain Registration (1 year)	$15	$15
			Beta Version-Total Cost	$1329

7 Appendices (Everyone)

[Appendices include supplemental information for the User that would distract if included in the regular sections.

7.1 Appendix A - Specifications (Live trading results)

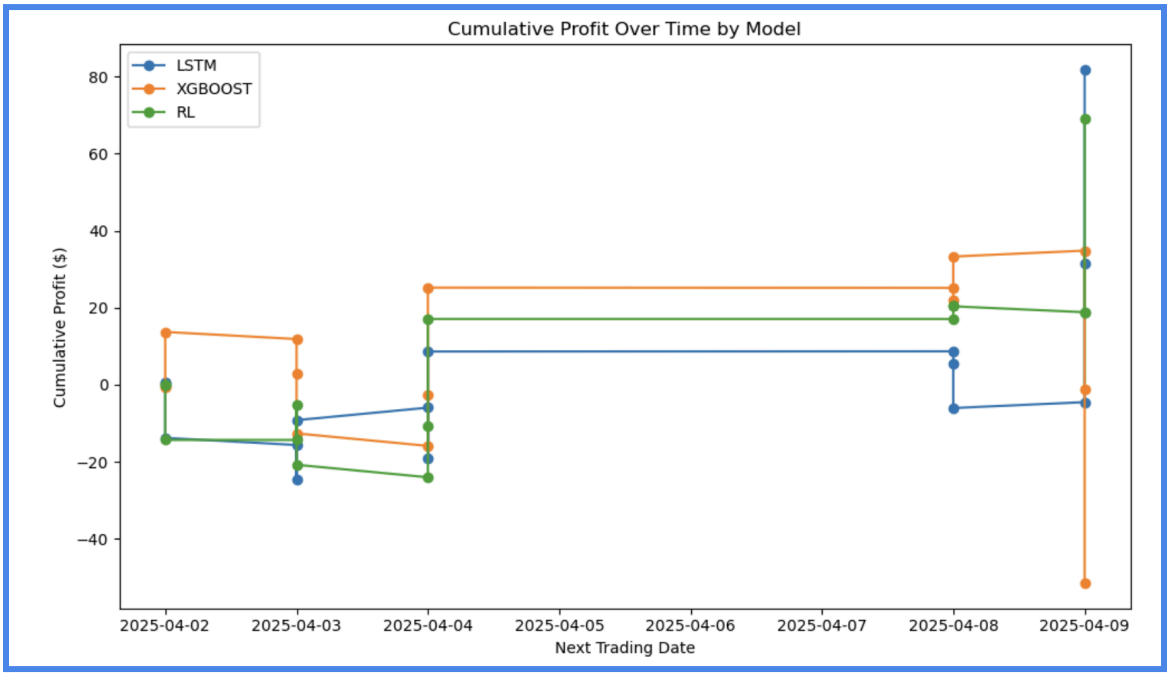


Figure 2 : Cumulative Profit Over Time by Model

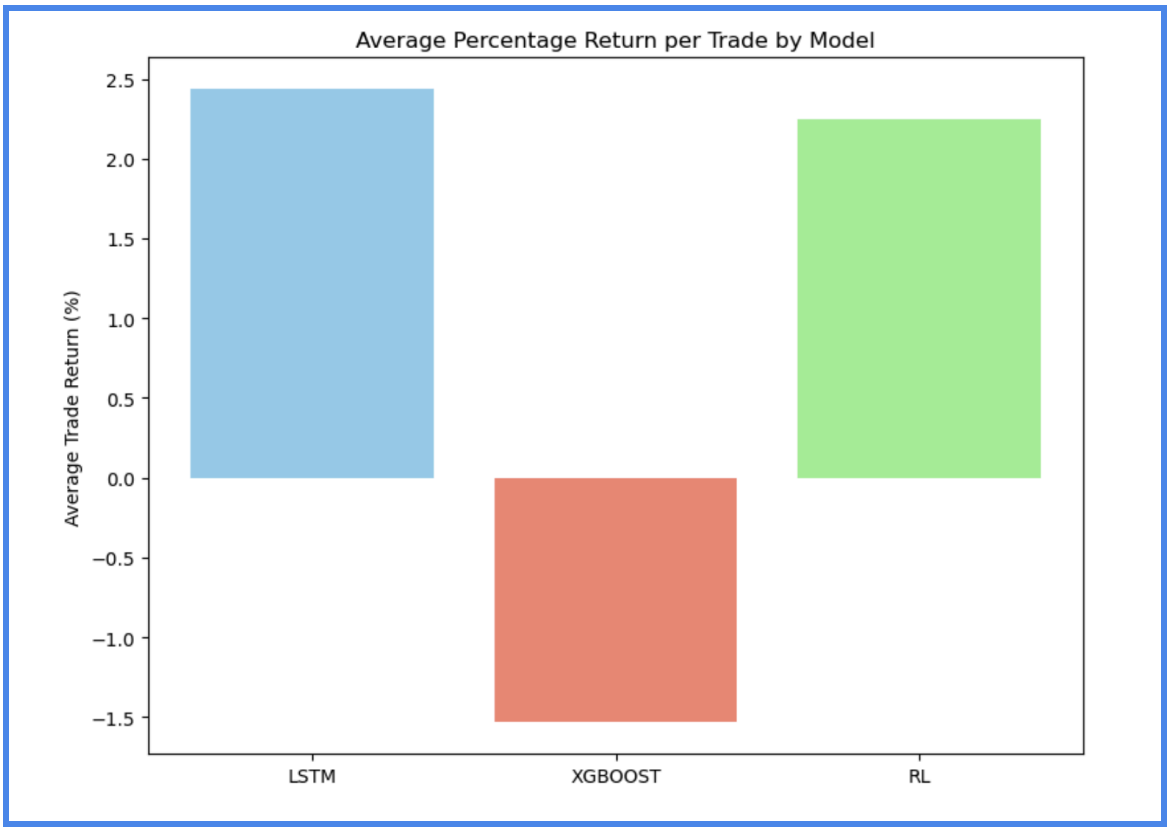


Figure 3 : Average Percentage Return per Trade by Model

Our live trading tests showed **profitable performance overall**, with both the LSTM and RL models generating positive returns. The **LSTM model** earned **$81.74** in profit with an average return of **2.44%** and a **Sharpe Ratio of 0.29**. The Reinforcement Learning model returned **$69.18** with a **2.25%** average return and a **Sharpe Ratio of 0.35**, reflecting strong risk-adjusted gains. While the XGBoost model underperformed with a loss of **-$51.44** and a negative return, the total combined profit across models was **$99.48**, validating our platform's effectiveness in live conditions and confirming system reliability and real-time prediction accuracy.

```
--- Overall Model Performance Metrics (Dollar) ---
LSTM MAE: 13.6935, RMSE: 18.2848, Directional Accuracy: 60.00%
XGBOOST Directional Accuracy: 53.33%
RL Directional Accuracy (excluding Hold): 40.00%
NLP Sentiment / Return Correlation: 0.0157

--- Overall Portfolio Simulation per Model (Dollar Profits) ---
LSTM: {'Total Profit ($)': 81.73999999999994, 'Average Profit ($)': 5.449333333333329, 'Sharpe Ratio': 0.287914717
73189414, 'Number of Trades': 15}
XGBOOST: {'Total Profit ($)': -51.44000000000001, 'Average Profit ($)': -3.429333333333334, 'Sharpe Ratio': -0.176
51584538537946, 'Number of Trades': 15}
RL: {'Total Profit ($)': 69.17999999999992, 'Average Profit ($)': 6.917999999999992, 'Sharpe Ratio': 0.34799911246
42655, 'Number of Trades': 10}
```

Figure 4 : Live Trading Result

```
--- Overall Portfolio Average Trade Percentage Returns ---
LSTM Average % Return: 2.44%
XGBOOST Average % Return: -1.53%
RL Average % Return (excluding Hold): 2.25%

--- Overall Cumulative Percentage Returns ---
LSTM Cumulative % Return: 2.35%
XGBOOST Cumulative % Return: -1.48%
RL Cumulative % Return: 2.58%

--- Overall Annualized Returns ---
LSTM Annualized Return (%): 236.21%
XGBOOST Annualized Return (%): -54.06%
RL Annualized Return (%): 278.54%

--- Overall Combined Profit across Models (Dollar) ---
{'LSTM ($)': 81.73999999999994, 'XGBOOST ($)': -51.44000000000001, 'RL ($)': 69.17999999999992}
```

Figure 5 : Live Trading Result Continued

7.2 Appendix B – Team Information

Team Members:

- Adrian Pawlowski - Computer Engineering, Concentration in Machine Learning
- Gabriel Brown - Computer Engineering
- Odilon Quevillon - Electrical Engineering, Concentration in Energy Tech, Minors in Business Administration and Systems Engineering
- Yagiz Idilman - Electrical Engineering

Figure 6: Team Information