Requirements Specifications

P04:TradeUp

<team member names & ids>

Student ID	Name
26100355	Muhammad Umar Zubair
26100204	Muhammad Shahmir Sher Qazi
26100216	Muhammad Rayyan Khan
25100137	Mohammed Raiyaan Junaid Hamid
26100200	Muhammad Ahmad

Table of Contents

1. Introduction

The project is a **stock trading simulation platform** that enables hands-on learning for people who want to gain trading experience without the risk of losing money. The platform is intended to provide experiential learning value to users, to supplement their stock trading learning journeys.

Our platform's objectives are to provide a safe and risk-free trading environment that pulls real-time stock market data to simulate the real thing – meaning our environment will capture and simulate the market's movement as it moves in real-time. Furthermore, we aim to provide valuable learning experiences to users with our platform that engage and empower them to improve their stock trading abilities and confidence.

Proposed Feature Set

- Buying/Selling Stocks
- Gamified Leaderboard and Topic-Specific Channels
- AI Insights and Chatbot
- AI News Sentiment Analysis
- Candlestick Charts View
- MarketWatch: to track favorited stocks
- Educational content
- Personalised Notification Systems

2. System Actors

Actor Name	Description
Stock Trading User	End user who will be using the platform to learn and practice trading.
System Admin	Admin user with admin privileges to enforce moderation through an admin dashboard

3. Use Cases

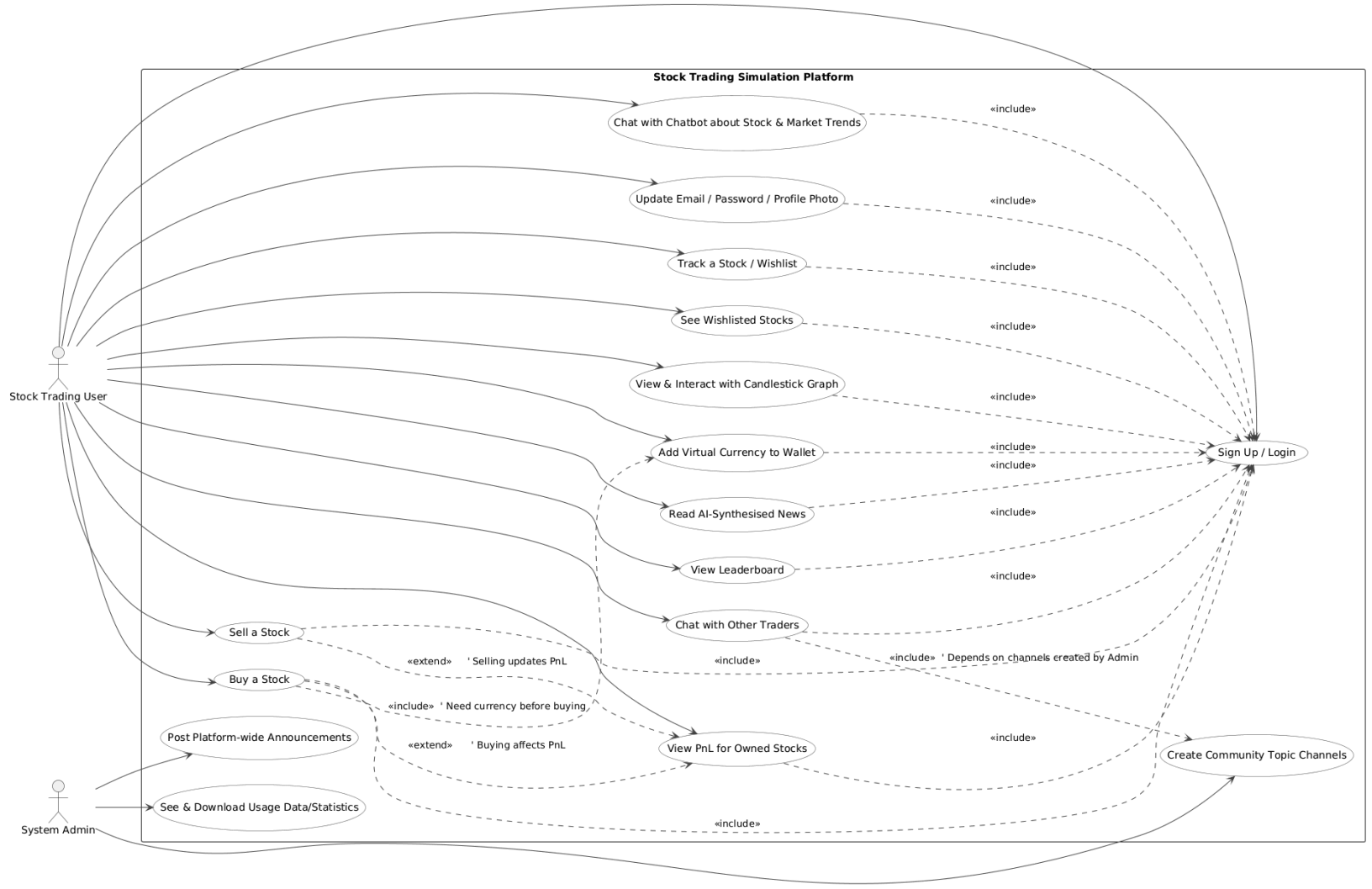
Stock Trading User:

- Sign up / login
- Update email / password / profile photo
- Track a stock (wishlist)
- View PnL information for owned stocks
- Buy a stock
- Sell a stock
- See wishlisted stocks
- View and interact with candlestick graph
- Add virtual currency to wallet
- Chat with chatbot about stock and market trends
- Read AI synthesised news
- Chat with other traders on the platform
- View leaderboard

System Admin:

- Create community topic channels
- Post platform-wide announcements
- See and download platform-wide usage data and statistics

3.1 Use Case Diagram



3.2 Description of Use Cases

3.2.1 Update Email / Password / Profile Photo

Identifier	UC-001
Purpose	The user can update their account information (email, password, or profile photo).
Pre-conditions	The user is logged into their account and has completed authentication.
Post-conditions	The updated account information is saved successfully in the system.
Step #	**Typical Course of Action**
1	The user navigates to the profile settings page
2	The system displays editable fields for email, password, and profile photo.
3	The user selects which information to update (email/password/photo)
4	The user inputs the new information or uploads a new photo
5	The system validates the input (e.g., strong password, valid email format, correct image format).
6	The system prompts the user to confirm changes.
7	The user confirms the update.
8	The system updates the user's account information.
Step #	**Alternate Courses of Action**
	At step 5: If validation fails, the user is prompted to re-enter valid input before proceeding. At step 7: The user cancels instead of confirming → return to profile settings.
Step #	**Exception Paths**
	If the system cannot connect to the server at step 8, an error is shown and the update is not saved.

3.2.2 Track a Stock (Wishlist)

Identifier	UC-002
Purpose	The user can add a stock to their watchlist for monitoring.
Pre-conditions	The user is logged in and has access to the stock search feature.
Post-conditions	The selected stock is added to the user's watchlist.
Step #	**Typical Course of Action**
1	The user searches for a stock using the search bar.
2	The system retrieves stock information in real-time.
3	The user clicks "Add to Watchlist."
4	The system confirms the action and adds the stock to the watchlist.
Step #	**Alternate Courses of Action**
	At step 2: If no matching stock is found, the system displays "Stock not found."
Step #	**Exception Paths**
	If the server is unavailable during step 2, an error message is displayed and the process ends.

3.2.3 View PnL Information for Owned Stocks

Identifier	UC-003
Purpose	The user can view profit and loss (PnL) information for their owned stocks.
Pre-conditions	The user is logged in and has at least one stock in their portfolio.
Post-conditions	The system displays PnL calculations (realized/unrealized) for owned stocks.
Step #	**Typical Course of Action**
1	The user navigates to the "Portfolio" section.

2	The system retrieves current market data for all owned stocks.
3	The system calculates PnL values for each stock.
4	The system displays a summary table with stock names, current prices, buy prices, and PnL.
Step #	**Alternate Courses of Action**
	If the user has no owned stocks, the system displays a message: “You currently own no stocks.”
Step #	**Exception Paths**
	If market data cannot be retrieved at step 2, the system shows cached data with a warning: “Live data unavailable.”

3.2.4 Buying a specific stock

Identifier	UC-004
Purpose	The user can buy his desired stock
Pre-conditions	He has logged in his account and has done user authentication and has enough tokens to buy the stock.
Post-conditions	The stock is added in his portfolio and the information
Step #	**Typical Course of Action**
1	The user searches for the desired stock
2	The system retrieves and displays stock details (price, trend, charts).
3	The user clicks “Buy.”
4	The system prompts for quantity and confirms total cost
5	The user enters quantity.
6	The system verifies sufficient funds in wallet.
7	The system asks for confirmation of purchase
8	The user confirms.
9	The system deducts the amount from wallet and adds the stock to the portfolio

10	The use case ends
Step #	**Alternate Courses of Action**
	At step 5: The user cancels instead of entering a quantity → return to stock page. At step 7: The user cancels confirmation → return to stock page.
Step #	**Exception Paths**
	At step 6: If insufficient funds, the system shows an error and suggests adding money to wallet.

3.2.5 Stock selling

Identifier	UC-005
Purpose	The user can sell an owned stock to free up wallet balance
Pre-conditions	The user is logged in and owns the stock they wish to sell.
Post-conditions	The stock is removed (partially or fully) from the portfolio, and wallet balance is updated.
Step #	**Typical Course of Action**
1.	The user navigates to the “Portfolio” section.
2.	The system displays list of owned stocks.
3.	The user selects a stock and clicks “Sell.”
4.	The system prompts for quantity and shows current market price.
5.	The user enters quantity.
6.	The system confirms the sale details (quantity × price = total).
7.	The user confirms sale.
8.	The system updates wallet balance and removes or reduces the stock from the portfolio
Step #	**Alternate Courses of Action**
	At step 5: The user cancels → return to portfolio.

	At step 7: The user cancels confirmation → return to portfolio.
Step #	**Exception Paths**
	If user enters a quantity greater than owned, the system displays an error and blocks the action. If market data is temporarily unavailable at step 4, cached data is shown with a warning

3.2.6 See Wishlisted Stocks

Identifier	UC-006
Purpose	The user can view all stocks they have added to their watchlist.
Pre-conditions	The user is logged in and has previously added at least one stock to watchlist.
Post-conditions	The system displays real-time information for all wishlisted stocks.
Step #	**Typical Course of Action**
1.	The user navigates to “Watchlist.”
2.	The system retrieves list of wishlisted stocks.
3.	The system fetches real-time market data for each stock.
4.	The system displays stock list with current price, daily change, and trend indicators.
Step #	**Alternate Courses of Action**
	If the watchlist is empty, the system displays “No stocks in your watchlist” and suggests adding some.
Step #	**Exception Paths**
	If real-time data retrieval fails at step 3, cached data is displayed with a warning.

3.2.7 View and Interact with Candlestick Graph

Identifier	UC-007

Purpose	The user can view and interact with candlestick charts of stocks to analyze market trends.
Pre-conditions	The user is logged in and selects a stock that supports candlestick chart data.
Post-conditions	The candlestick chart is displayed with interactive options (zoom, timeframe change, hover tooltips).

Step #	Typical Course of Action
1.	The user selects a stock from portfolio, watchlist, or search results.
2.	The system retrieves stock's historical price data.
3.	The candlestick chart is generated and displayed.
4.	The user hovers over a candlestick to view open, close, high, and low values.
5.	The user can zoom in/out or switch timeframes (1D, 1W, 1M, etc.).
6.	The chart refreshes dynamically to reflect the chosen settings.

Step #	Alternate Courses of Action
	At step 5: If the user selects an unsupported timeframe, system displays "Timeframe not available."
Step #	**Exception Paths**
	At step 2: If historical data cannot be retrieved, system shows error "Chart data unavailable" and exits.

3.2.8 Add Money to Wallet

Identifier	UC-008
Purpose	The user can top up their simulation wallet with virtual funds for trading
Pre-conditions	The user is logged in and has access to wallet features
Post-conditions	The wallet balance is updated with the new funds.

Step #	Typical Course of Action
1.	The user navigates to the “Wallet” section.
2.	The system displays current wallet balance.
3.	The user clicks “Add Funds.”
4.	The system prompts for the top-up amount.
5.	The user enters the amount.
6.	The system confirms the new balance.
7.	The user confirms.
8.	The wallet is updated with the additional funds.
Step #	**Alternate Courses of Action**
	At step 5: If user cancels, return to wallet dashboard. At step 7: If user cancels, no funds are added.
Step #	**Exception Paths**
	If user enters a negative or invalid amount at step 5, an error message is displayed and input must be corrected.

3.2.9 Chat with Chatbot about Stock and Market Trends

Identifier	UC-009
Purpose	The user can interact with an AI chatbot to ask about stocks and market trends.
Pre-conditions	The user is logged in and chatbot service is active.
Post-conditions	The chatbot provides responses based on market data, insights, or educational content.
Step #	**Typical Course of Action**
1.	The user opens the “Chatbot” feature.
2.	The system displays chat interface.
3.	The user types a question (e.g., “What’s the trend for Tesla stock?”).
4.	The chatbot retrieves relevant stock data and AI insights.

5.	The chatbot generates a natural language response.
6.	The system displays the response in chat.
7.	The user can continue chatting or exit.
Step #	**Alternate Courses of Action**
	At step 3: If user asks a general trading question, chatbot responds with educational or help content instead of stock-specific data.
Step #	**Exception Paths**
	At step 4: If chatbot service is unavailable, the system displays “Chatbot currently offline.”

3.2.10 Read AI-Synthesised News

Identifier	UC-0010
Purpose	The user can read AI-generated news summaries and sentiment analysis on market events.
Pre-conditions	The user is logged in and the AI news feed service is active.
Post-conditions	AI-synthesised news articles and insights are displayed to the user
Step #	**Typical Course of Action**
1.	The user navigates to the “News” section
2.	The system fetches real-time financial and market news.
3.	The AI processes and summarizes key articles.
4.	The AI performs sentiment analysis (positive, neutral, negative).
5.	The system displays news summaries with sentiment indicators.
6.	The user clicks on a summary to view details.
7.	The system shows extended insights and related educational notes.
Step #	**Alternate Courses of Action**
	If user filters for a specific sector/stock, the system shows only relevant AI-synthesised news.

Step #	Exception Paths
	If external news sources are unavailable at step 2, the system displays cached news or “No updates available.”

3.2.11 Chat with Other Traders on the Platform

Identifier	UC-0011
Purpose	The user can chat with other traders in topic-specific channels or private chat.
Pre-conditions	The user is logged in and has access to chat/community features.
Post-conditions	The user can exchange messages with others in real time.
Step #	**Typical Course of Action**
1.	The user navigates to the “Community” or “Channels” section.
2.	The system displays available topic-specific channels (e.g., Tech Stocks, Forex, Beginners).
3.	The user selects a channel or private chat.
4.	The system loads recent messages.
5.	The user types and sends a message.
6.	The system delivers the message to all channel participants.
7.	The system updates the chat in real-time with responses.
Step #	**Alternate Courses of Action**
	At step 3: The user creates a new channel instead of joining an existing one. At step 5: The user can attach a stock link, chart, or news summary.
Step #	**Exception Paths**
	If the chat service is down, system displays “Chat currently unavailable.”

	If a user sends inappropriate content, moderation system flags or blocks it.

3.2.12 View Leaderboard

Identifier	UC-0012
Purpose	The user can view the gamified leaderboard ranking traders based on performance.
Pre-conditions	The user is logged in and leaderboard functionality is active.
Post-conditions	Leaderboard rankings are displayed based on metrics (PnL, growth %, activity).

Step #	**Typical Course of Action**
1.	The user navigates to the “Leaderboard” section.
2.	The system retrieves updated rankings of all traders.
3.	The leaderboard is displayed with usernames, rank, and performance metrics.
4.	The user can filter by timeframe (daily, weekly, monthly).
5.	The user can compare their rank against peers.

Step #	**Alternate Courses of Action**
	At step 4: If user selects unsupported timeframe, system defaults to “weekly.”

Step #	**Exception Paths**
	If leaderboard data cannot be retrieved, system displays “Leaderboard unavailable at this time.”

3.2.13 Create Community Topic Channels

Identifier	UC-0013

Purpose	The system admin can create topic-specific community channels for user discussions.
Pre-conditions	The admin is logged in with administrative privileges.
Post-conditions	A new topic channel is created and available for users to join and interact.

Step #	Typical Course of Action
1.	The admin navigates to the “Community Management” dashboard.
2.	The system displays existing channels and the option to create a new one.
3.	The admin clicks “Create channel”
4.	The system prompts for channel name, description, and category
5.	The admin enters the details.
6.	The system asks for confirmation.
7.	The admin confirms.
8	The system creates the new channel and updates the list of available channels

Step #	Alternate Courses of Action
	At step 5: If the admin leaves details incomplete, the system prevents progression until required fields are filled. At step 6: If the admin cancels, no channel is created.
Step #	**Exception Paths**
	If there is a server error at step 8, the system displays “Unable to create channel, try again later.”

3.2.14 Post Platform-Wide Announcements

Identifier	UC-0014
Purpose	The system admin can post announcements visible to all users.
Pre-conditions	The admin is logged in with announcement privileges.

Post-conditions	The announcement is published and displayed to all users on the platform.

Step #	Typical Course of Action
1.	The admin navigates to the “Announcements” section.
2.	The system displays past announcements and an option to create a new one.
3.	The admin clicks “New Announcement.”
4.	The system prompts for title, message content, and optional attachments/links.
5.	The admin fills in the details.
6.	The system previews the announcement.
7.	The admin confirms publishing.
8	The system posts the announcement platform-wide (dashboard banners, notifications, or emails).
Step #	**Alternate Courses of Action**
	At step 5: If the admin saves as a draft, the announcement is stored but not published. At step 7: If the admin cancels, no announcement is posted.
Step #	**Exception Paths**
	If system fails at step 8, a “Publishing error” is displayed, and the announcement is not sent.

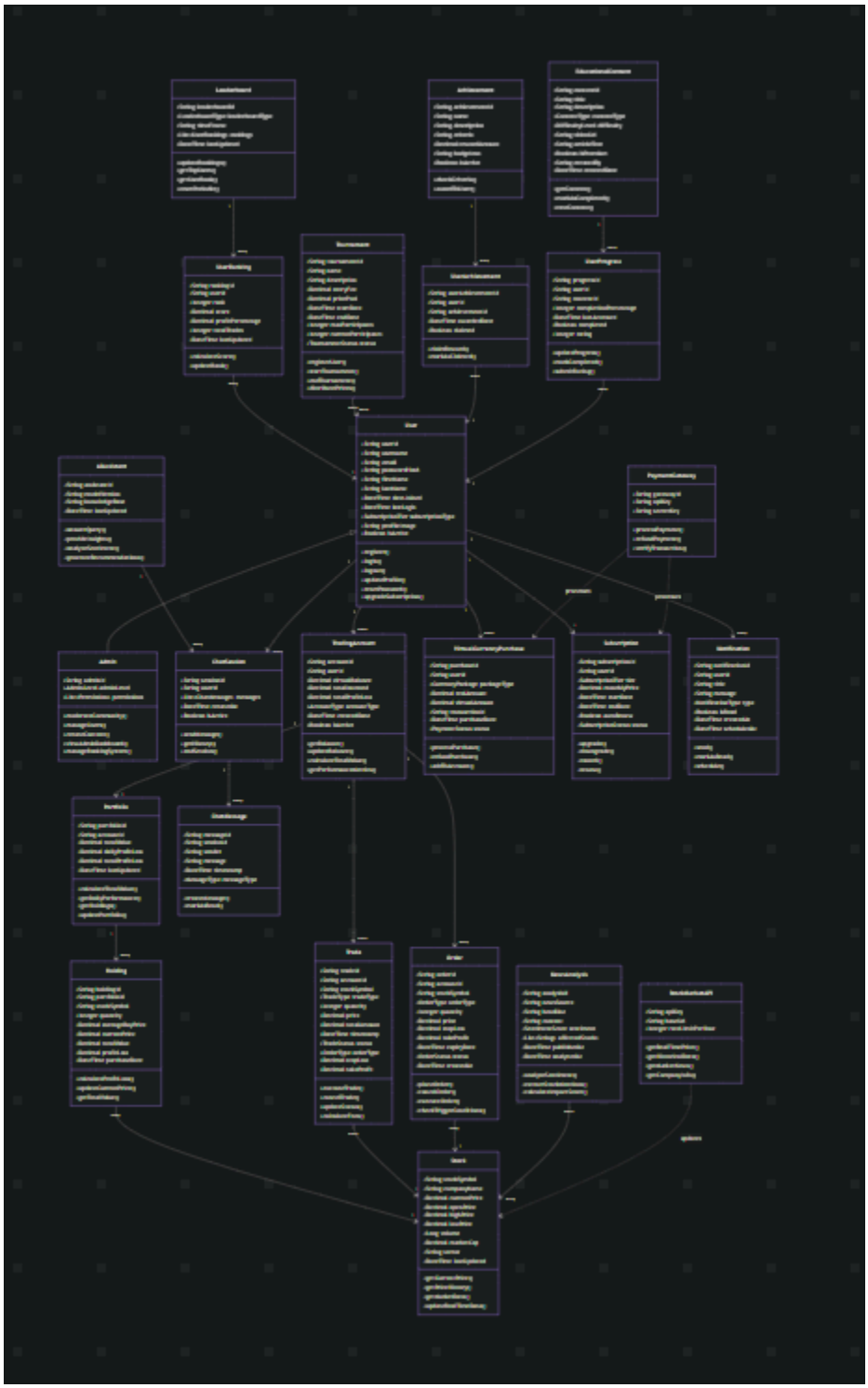
3.2.15 See and Download Platform-Wide Usage Data and Statistics

Identifier	UC-0015
Purpose	The system admin can view and export usage data/statistics to monitor platform performance.
Pre-conditions	The admin is logged in with reporting privileges.
Post-conditions	Usage data is retrieved, displayed, and optionally downloaded in a report format (CSV, PDF, Excel).

Step #	Typical Course of Action
1.	The admin navigates to the “Analytics/Reports” section.
2.	The system displays available metrics (active users, trades executed, wallet balances, engagement levels, etc.).
3.	The admin applies filters (date range, region, user type, etc.).
4.	The system generates visualizations and statistics based on filters.
5.	The admin clicks “Download Report.”
6.	The system prompts for file format (CSV, Excel, PDF).
7.	The admin selects format.
8.	The system generates and downloads the report.
Step #	**Alternate Courses of Action**
	At step 3: If the admin chooses no filters, system generates a full platform-wide report.
Step #	**Exception Paths**
	If data cannot be retrieved at step 2 (server error), system displays “Analytics temporarily unavailable.” If export fails at step 8, system displays “Download failed. Please try again.”

4. Class Diagram

4.1 Diagram



Link to Mermaid Diagram for Class Diagram: Link

4.2 Description

Core User Management

- **User** → Represents a platform user with personal info, login credentials, subscription type, and activity tracking.
- **Admin** → A specialized user with elevated privileges to manage users, content, and system functions.

Trading & Portfolio Management

- **TradingAccount** → A user's virtual trading account that holds balance, investment stats, and trade history.
- **Portfolio** → A collection of holdings within a trading account, showing overall value and performance.
- **Holding** → A single stock position within a portfolio, including quantity, buy price, and profit/loss.
- **Stock** → Market data about a stock, such as price, volume, sector, and company details.
- **Trade** → A completed buy or sell transaction, with details like price, type, quantity, and status.
- **Order** → A pending instruction to buy or sell a stock under certain conditions (market, limit, stop-loss).

AI & Analytics System

- **AIAssistant** → An AI-driven assistant that answers queries, provides insights, and recommends trades.
- **ChatSession** → A conversation between a user and the AI assistant (or other entities).
- **ChatMessage** → A single message within a chat session, from either the user, AI, or system.
- **NewsAnalysis** → Analyzed financial news, showing sentiment, stock mentions, and potential impact.

Gamification & Social Features

- **Leaderboard** → Tracks and displays top users based on trading performance or achievements.
- **UserRanking** → A single user's rank, score, and performance stats on a leaderboard.
- **Tournament** → Competitive trading events where users pay entry fees, trade, and compete for prizes.
- **Achievement** → A milestone or badge users can earn by meeting specific criteria.
- **UserAchievement** → A record of which achievements a user has earned and whether rewards were claimed.

Educational System

- **Educational Content** → Learning materials like videos, articles, or quizzes about trading.
- **UserProgress** → Tracks a user's progress, completion, and ratings for educational content.

Monetization System

- **VirtualCurrencyPurchase** → A record of when a user buys in-game currency using real money.
- **Subscription** → Represents a user's subscription plan, renewal settings, and billing cycle.

Communication System

- **Notification** → Alerts sent to users about trades, portfolio updates, achievements, or news.

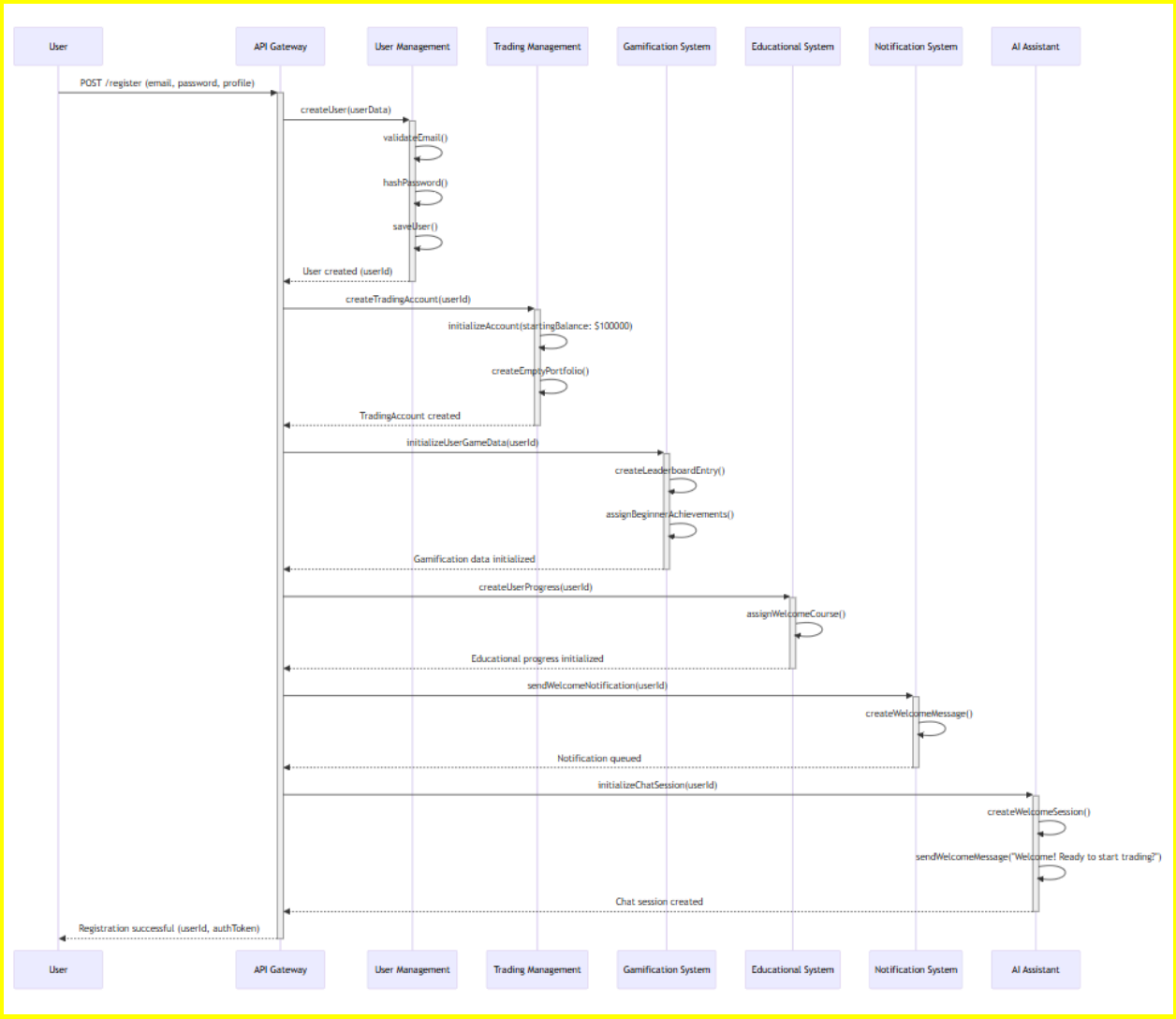
External API Integration

- **StockMarketAPI** → Connects to external stock market data providers for real-time and historical data.

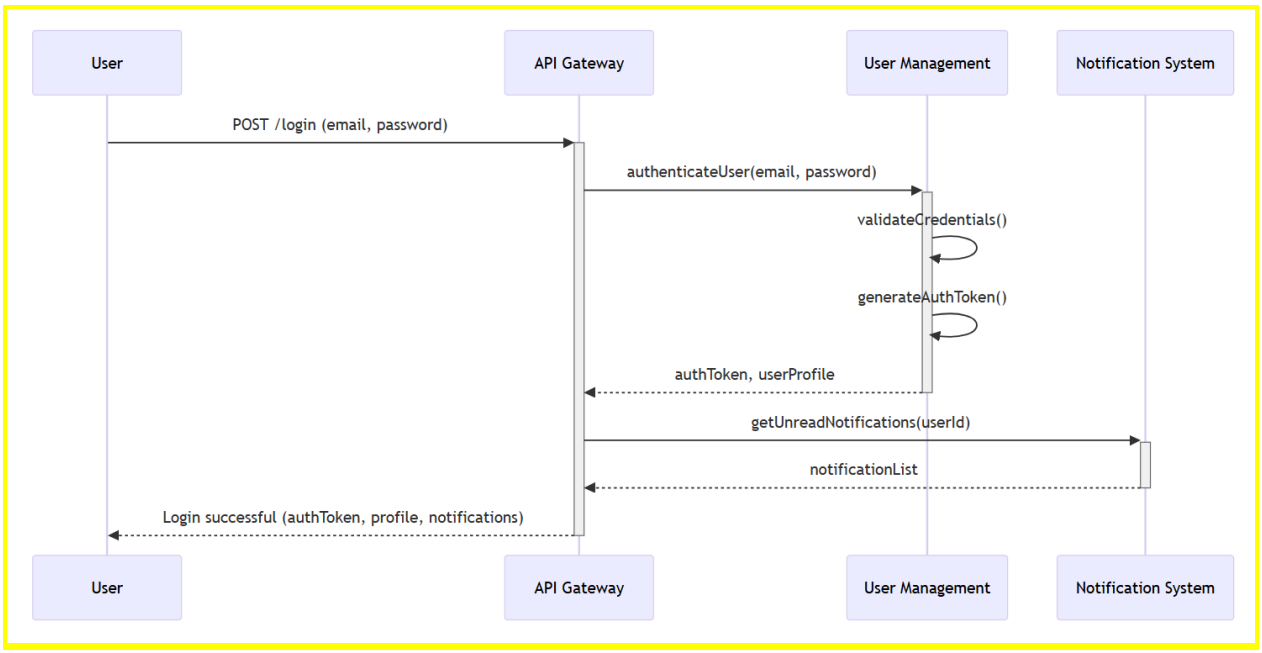
- **PaymentGateway** → Handles payment processing, refunds, and transaction verification.

-5. Sequence Diagrams

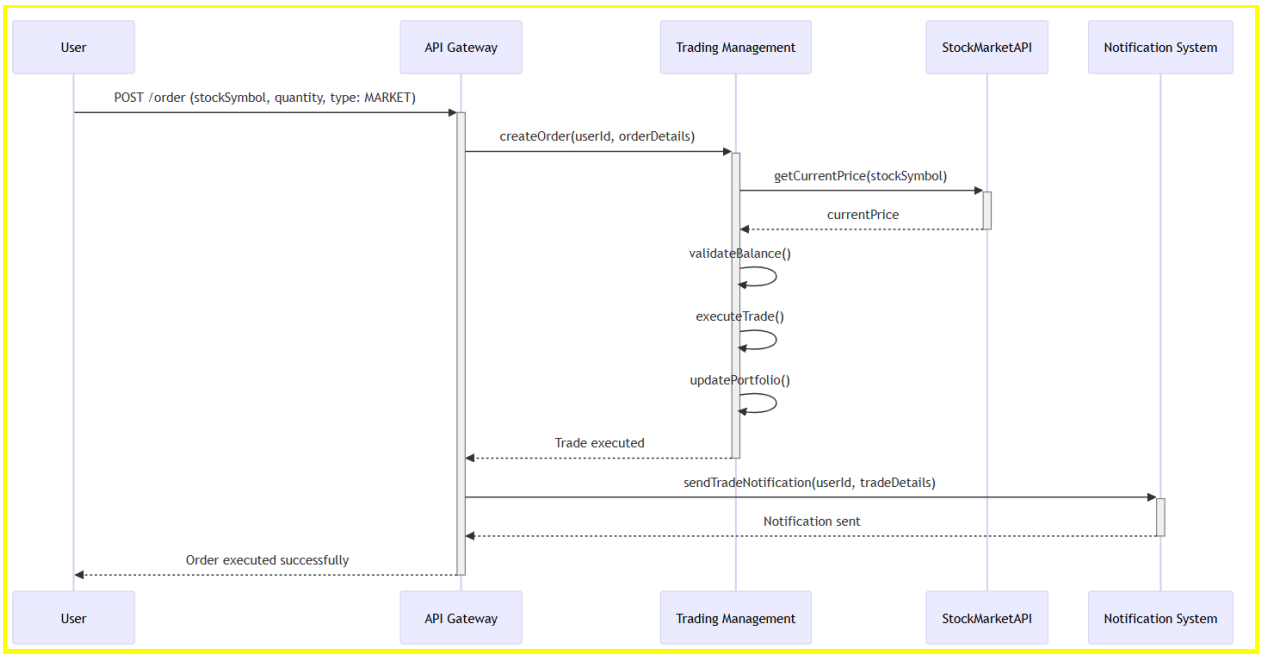
User Registration



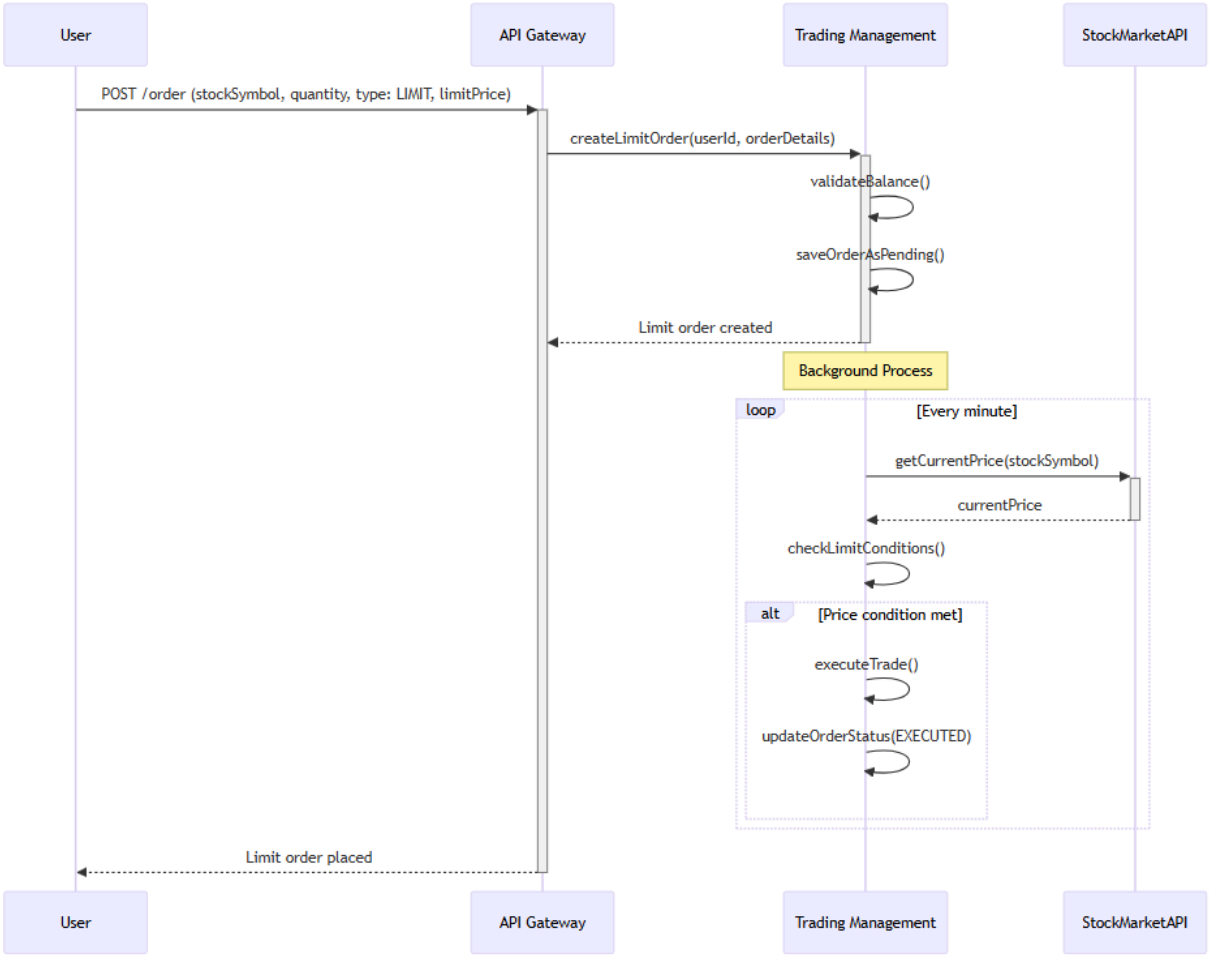
User login



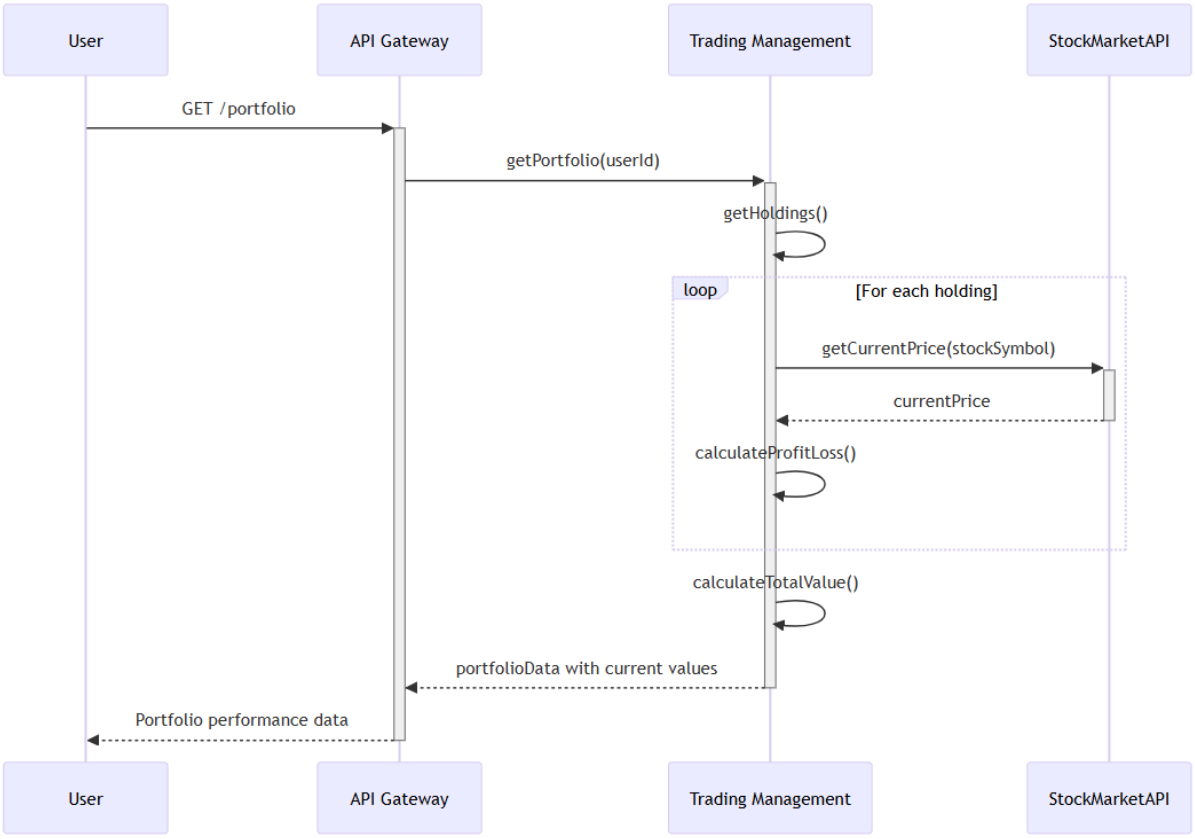
Place Market Order



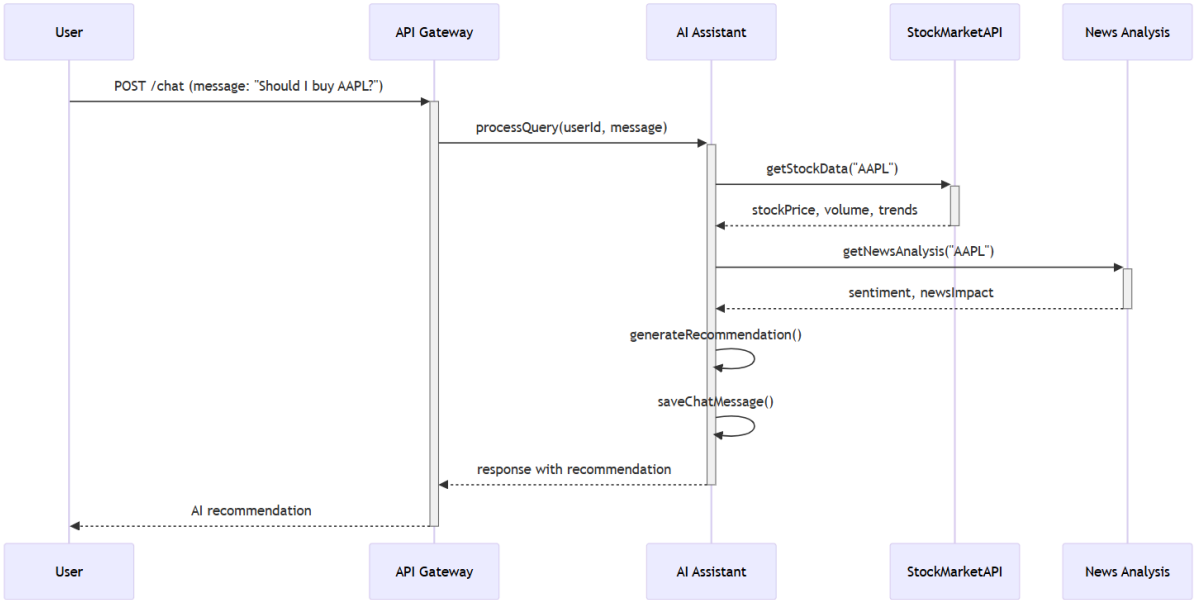
Place Limit Order



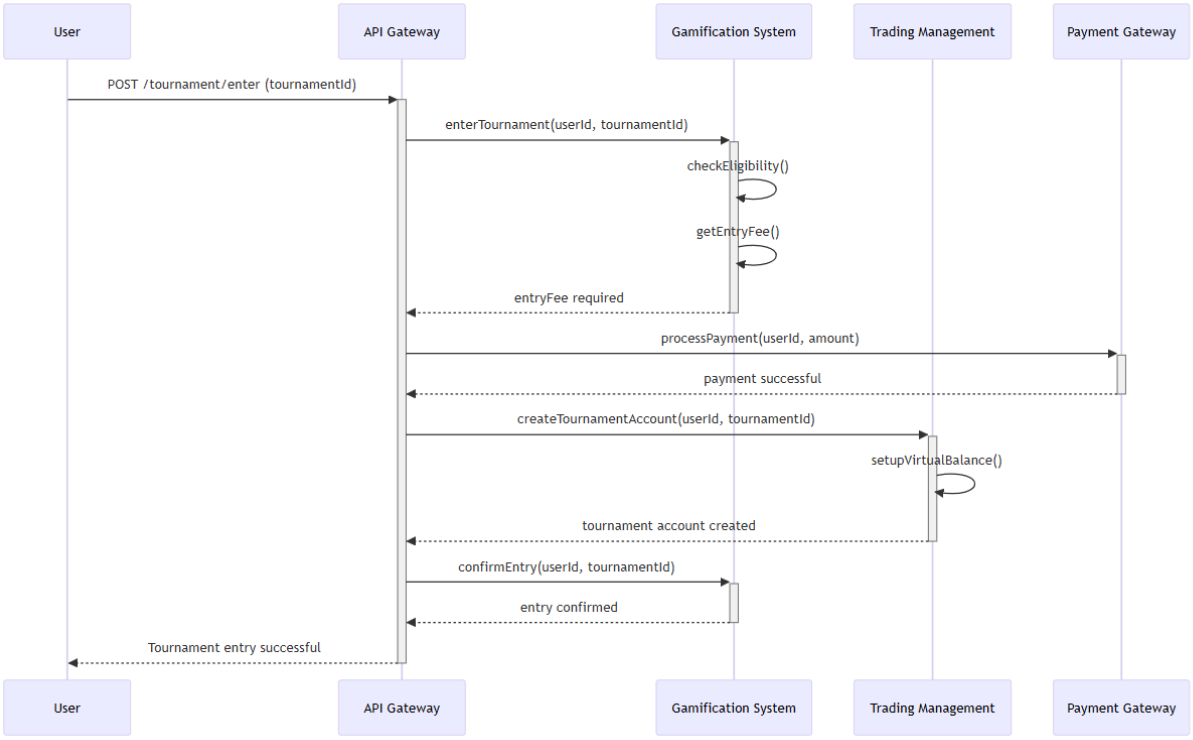
View Portfolio



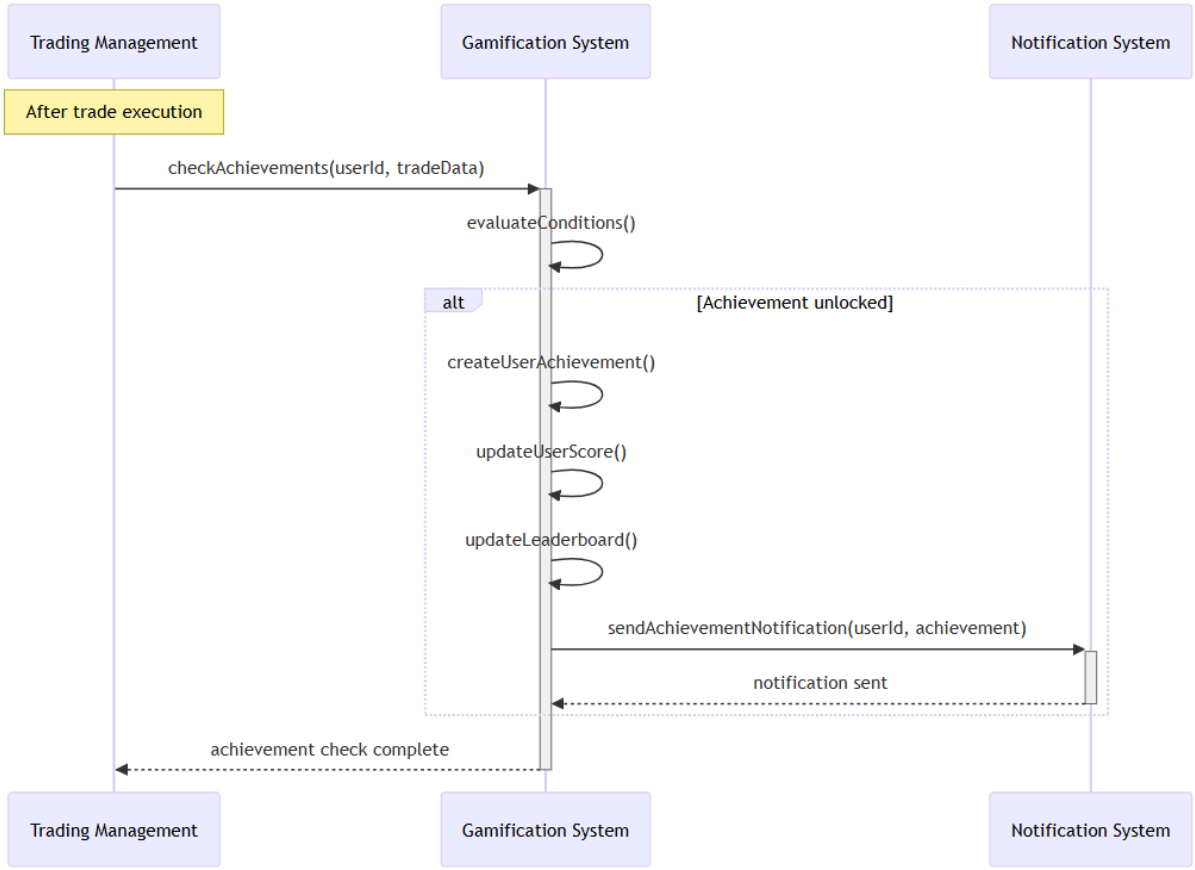
AI Assistant



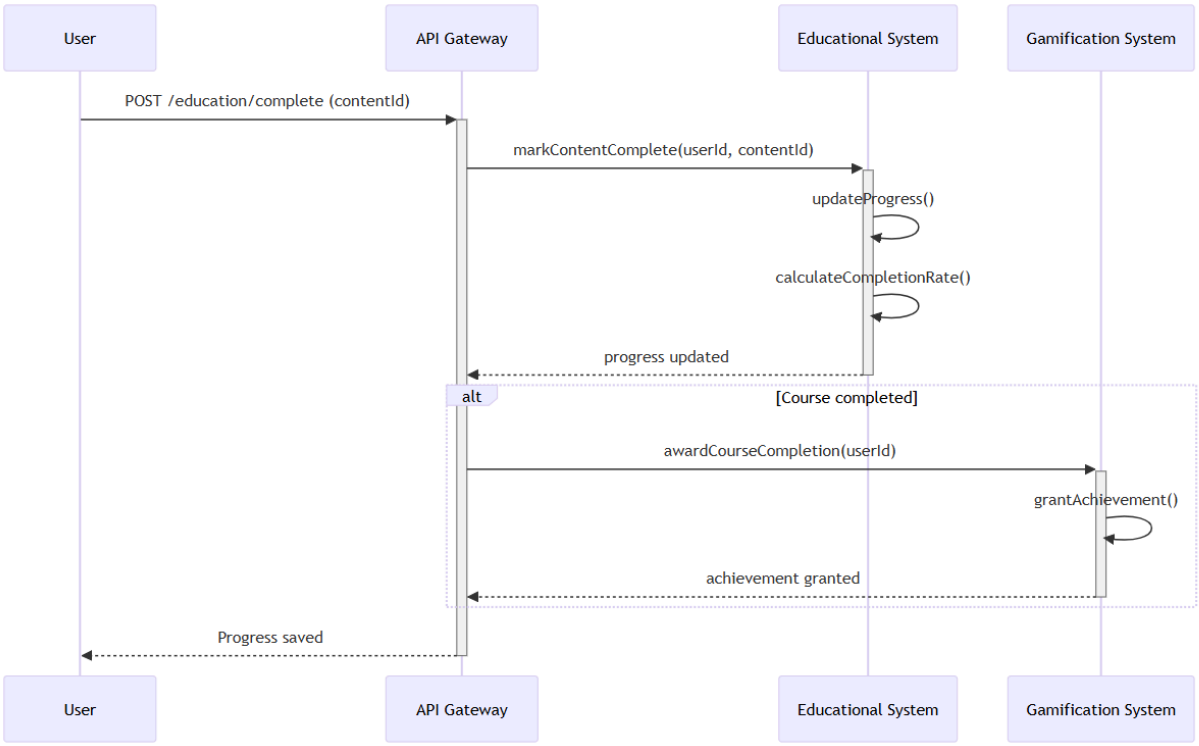
Game/Tournament



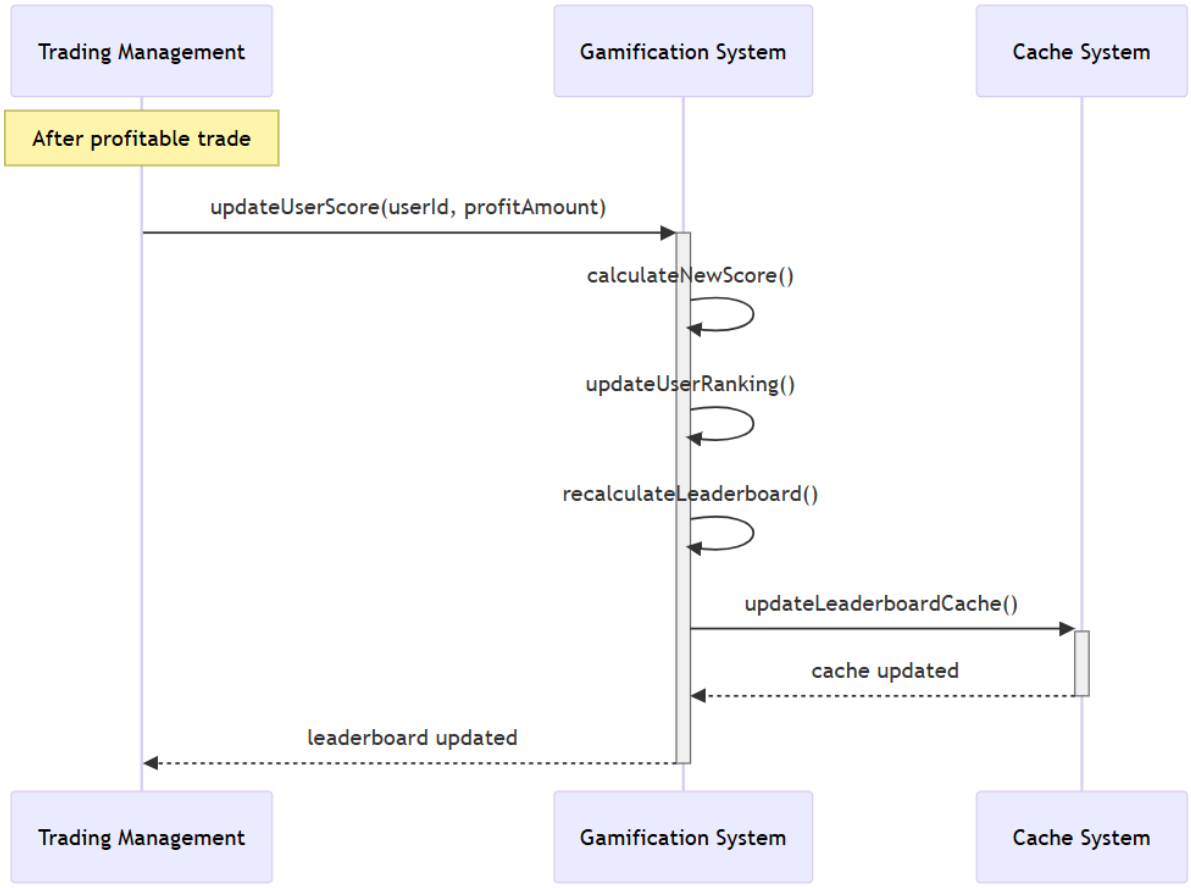
Achievement



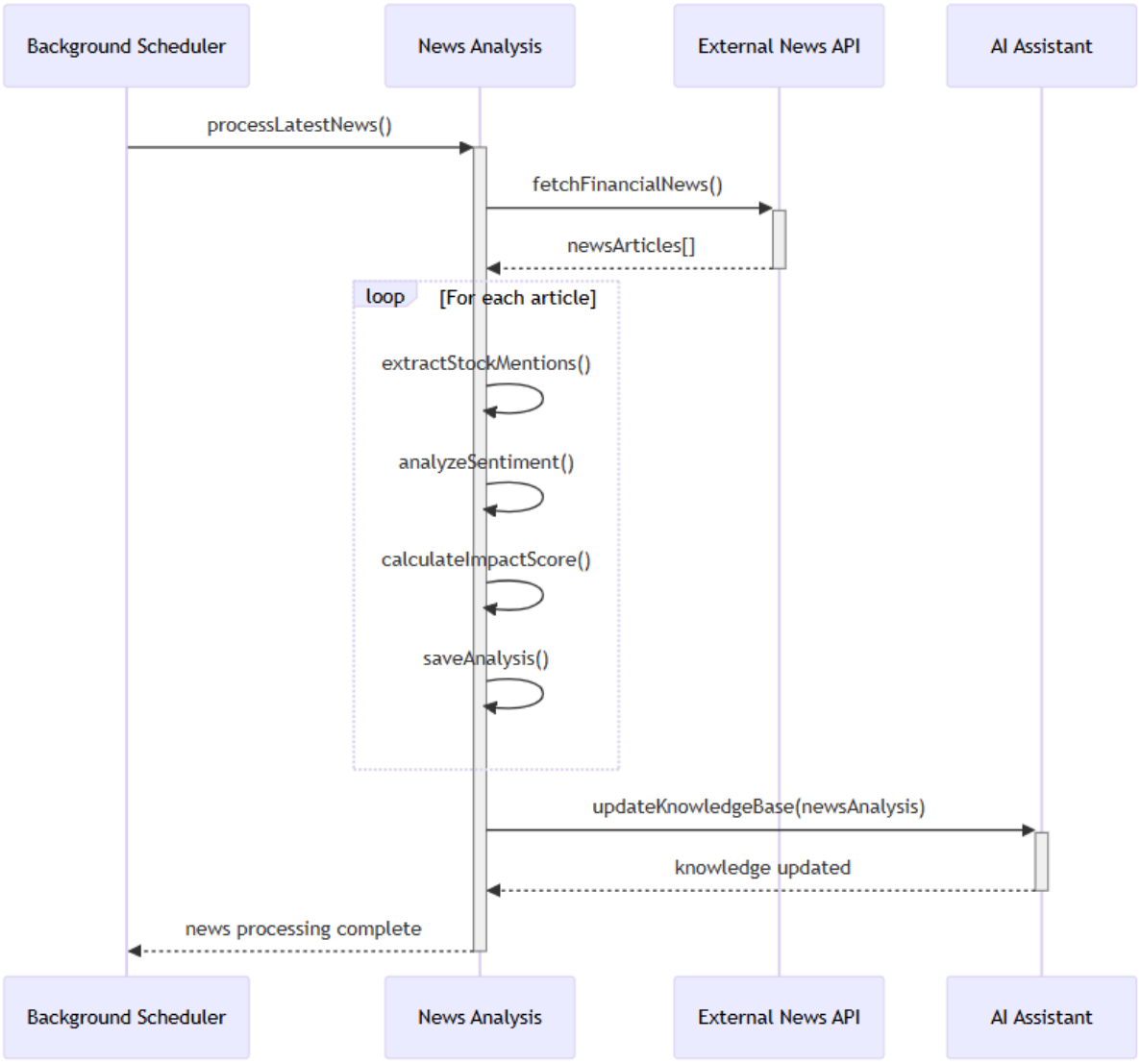
Educational content



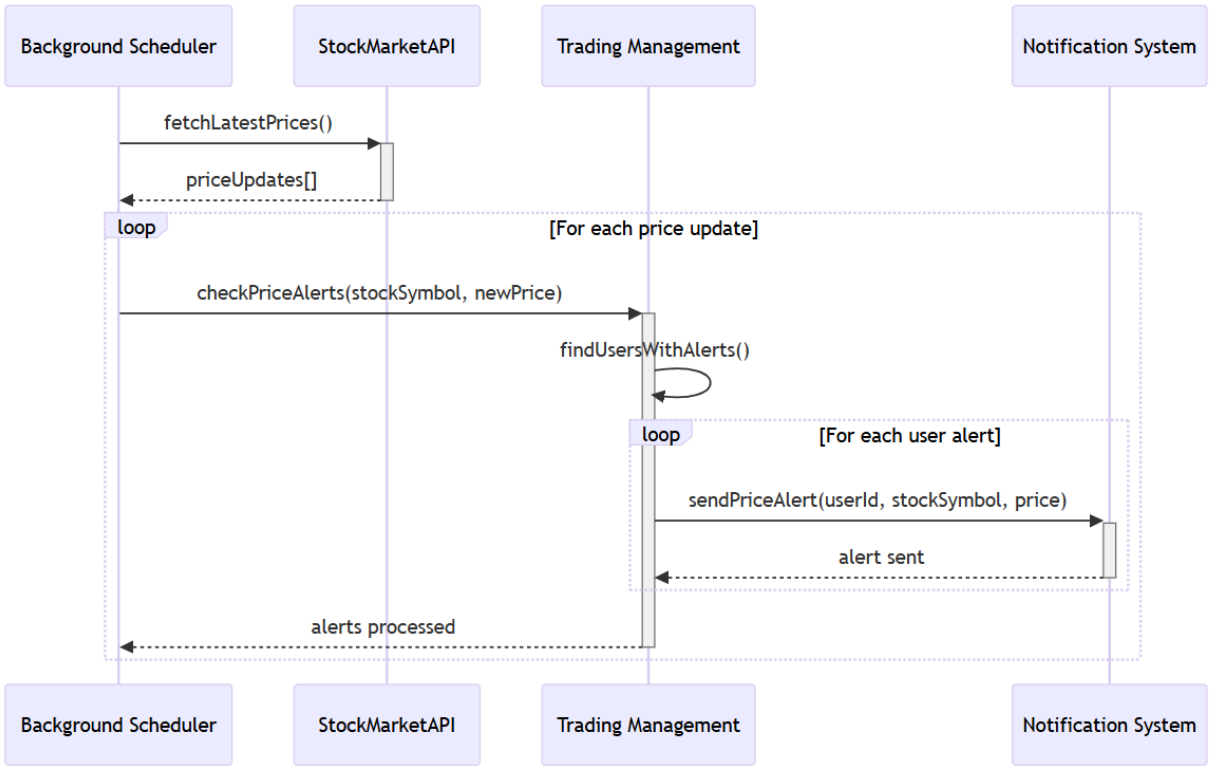
Leaderboard Update



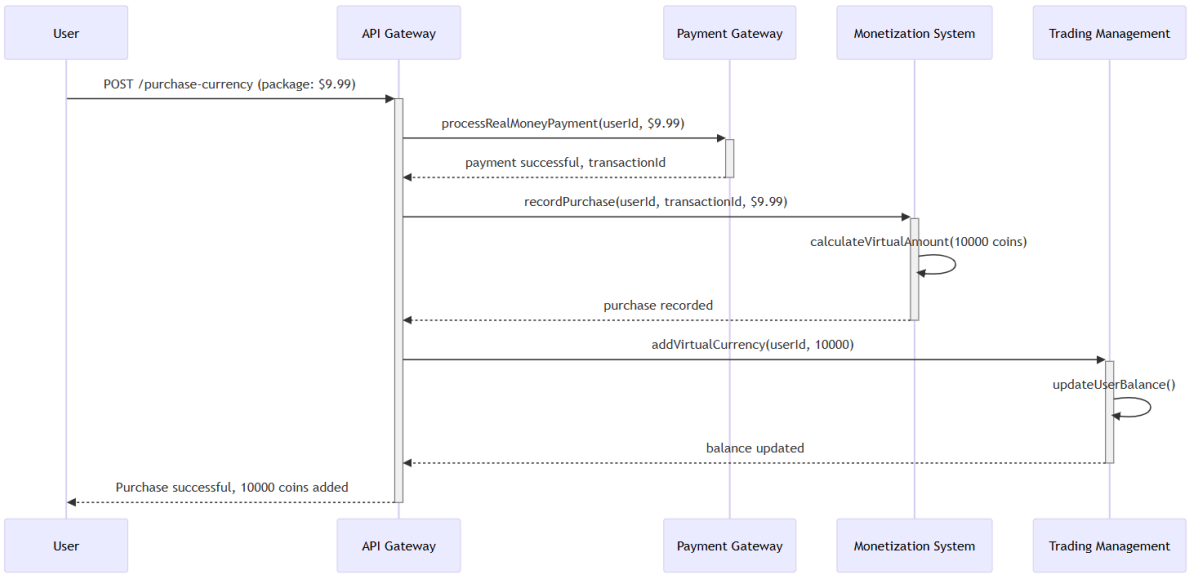
News Analysis



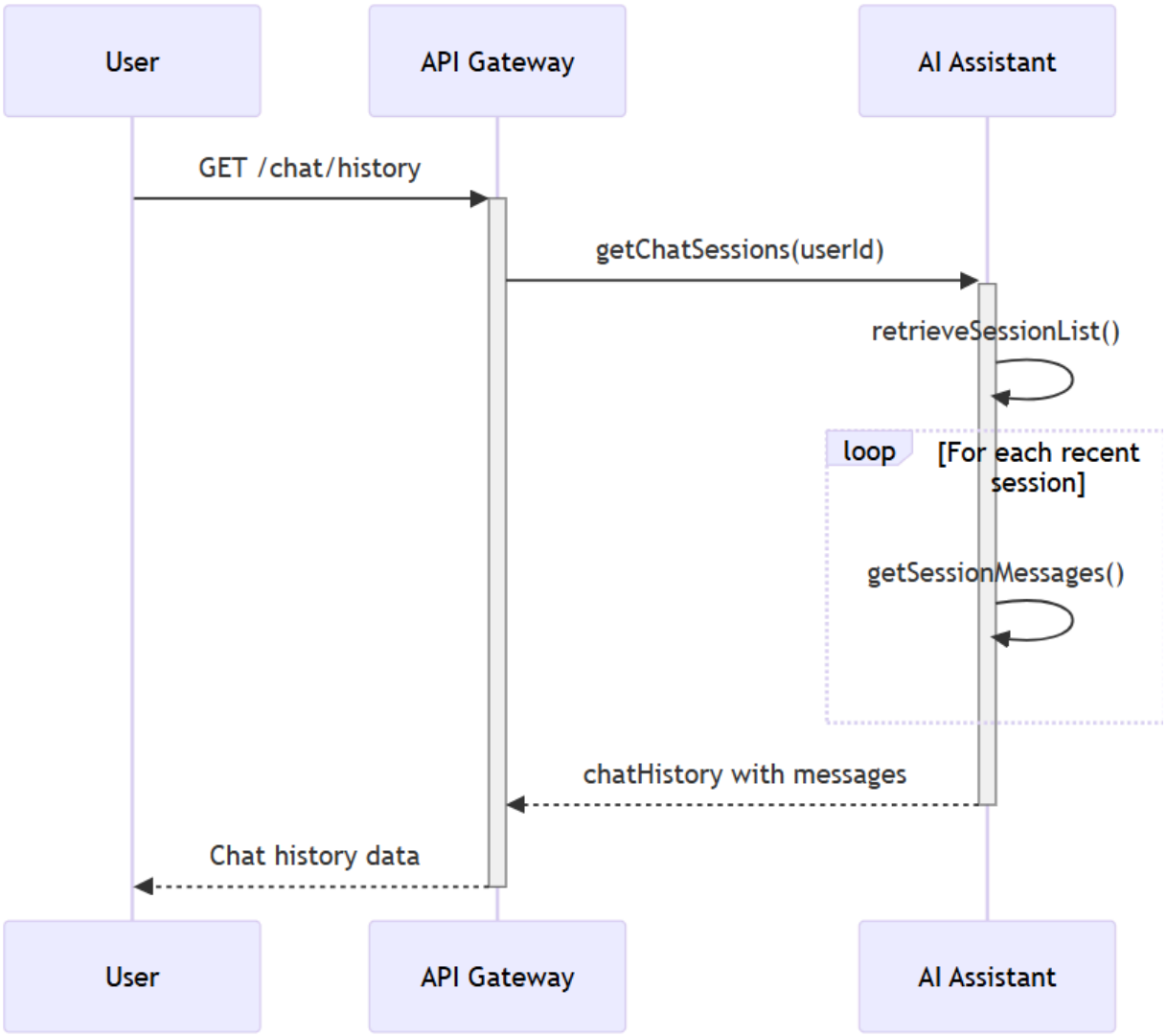
Stock Price Alert



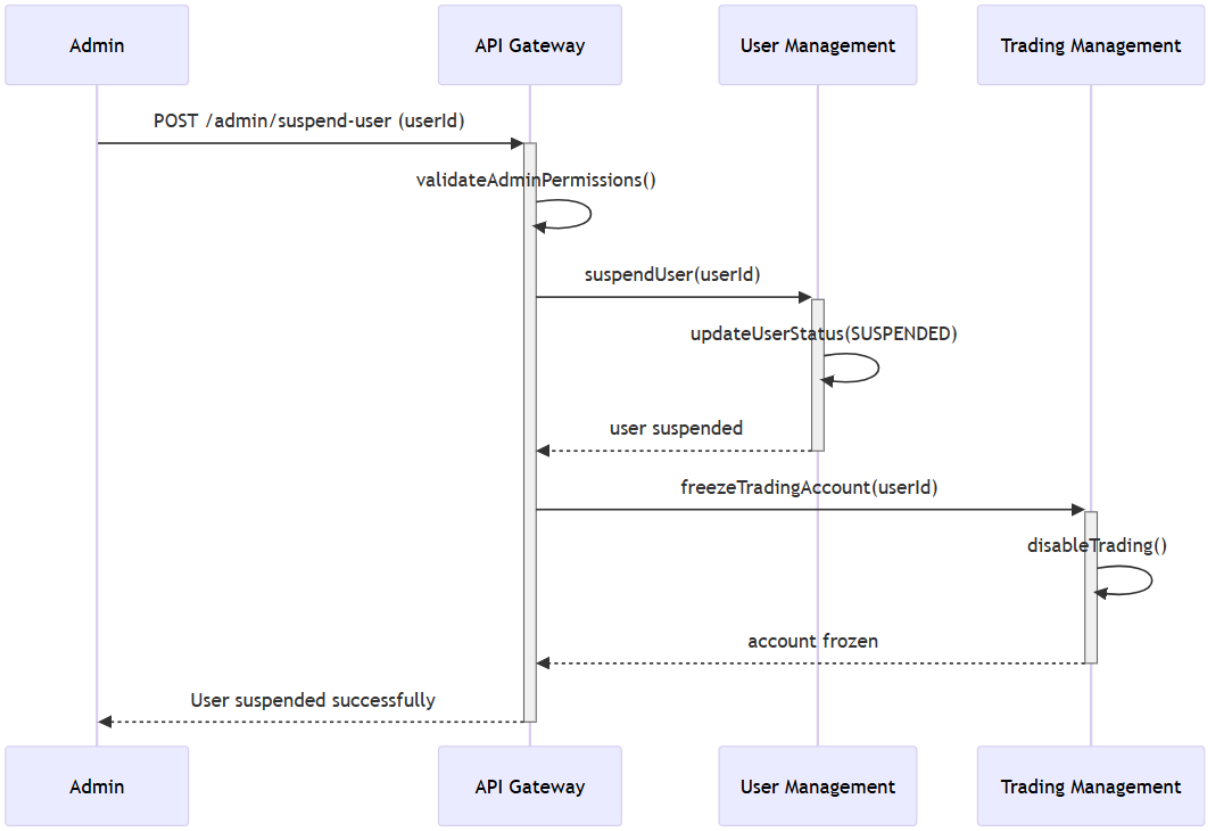
Virtual Currency purchase



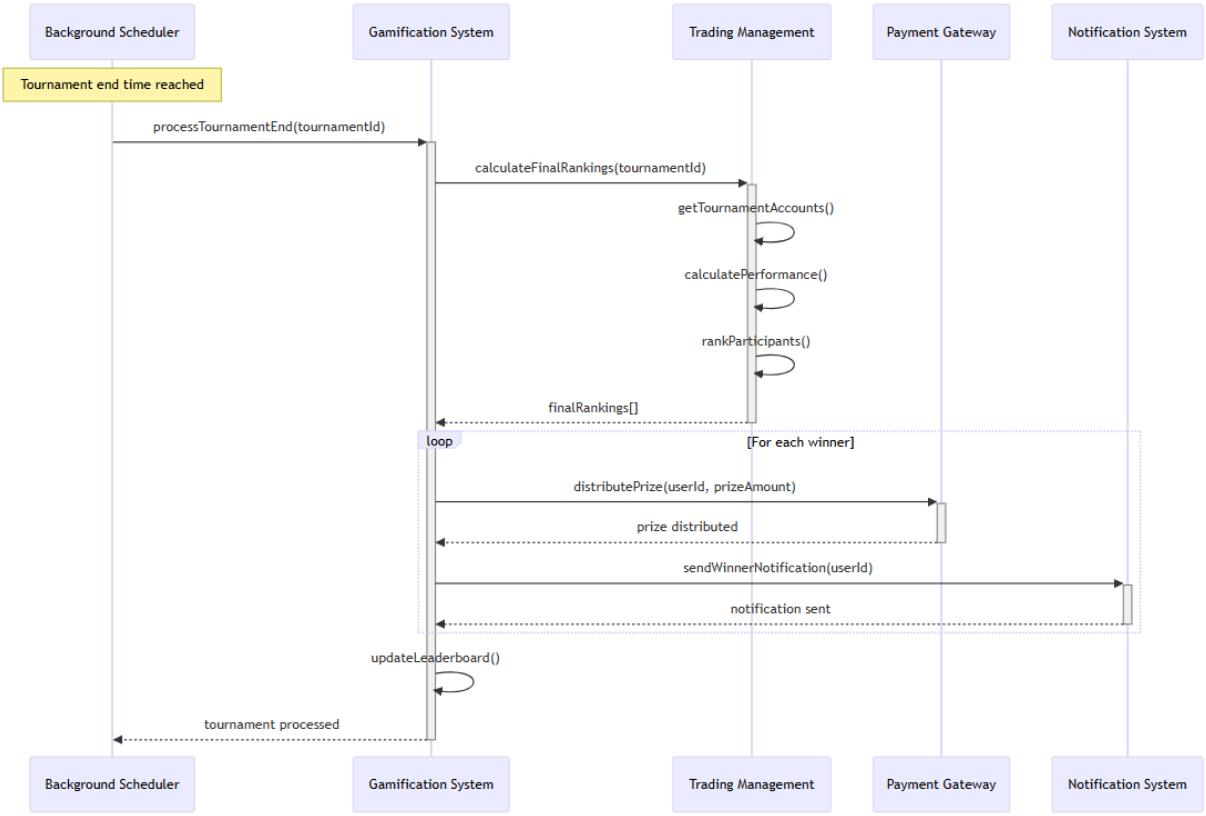
Chat


User
API Gateway
AI Assistant
GET /chat/history
getChatSessions(userId)
retrieveSessionList()
loop
[For each recent session]
getSessionMessages()
chatHistory with messages
Chat history data
User
API Gateway
AI Assistant

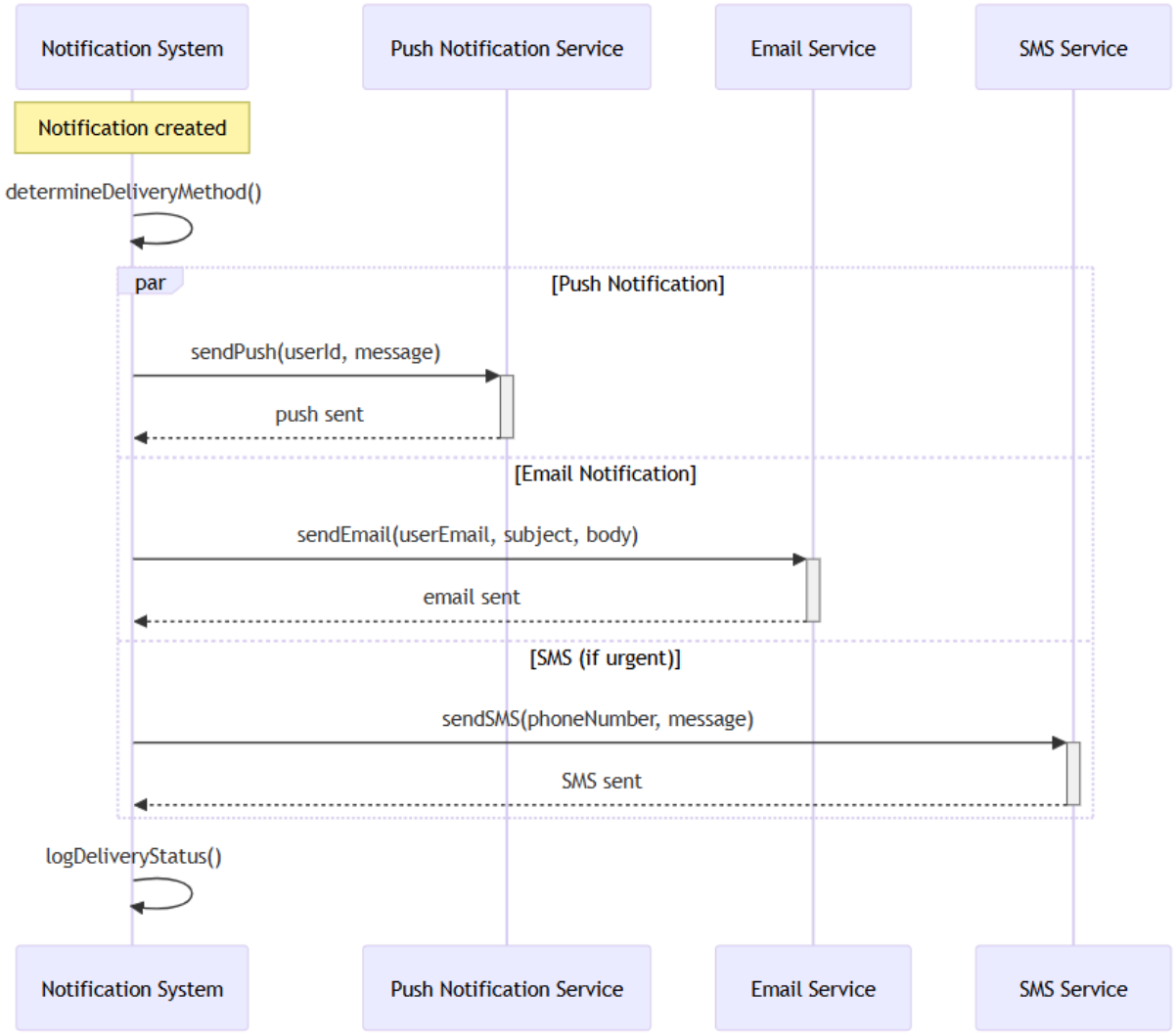
Admin User Management



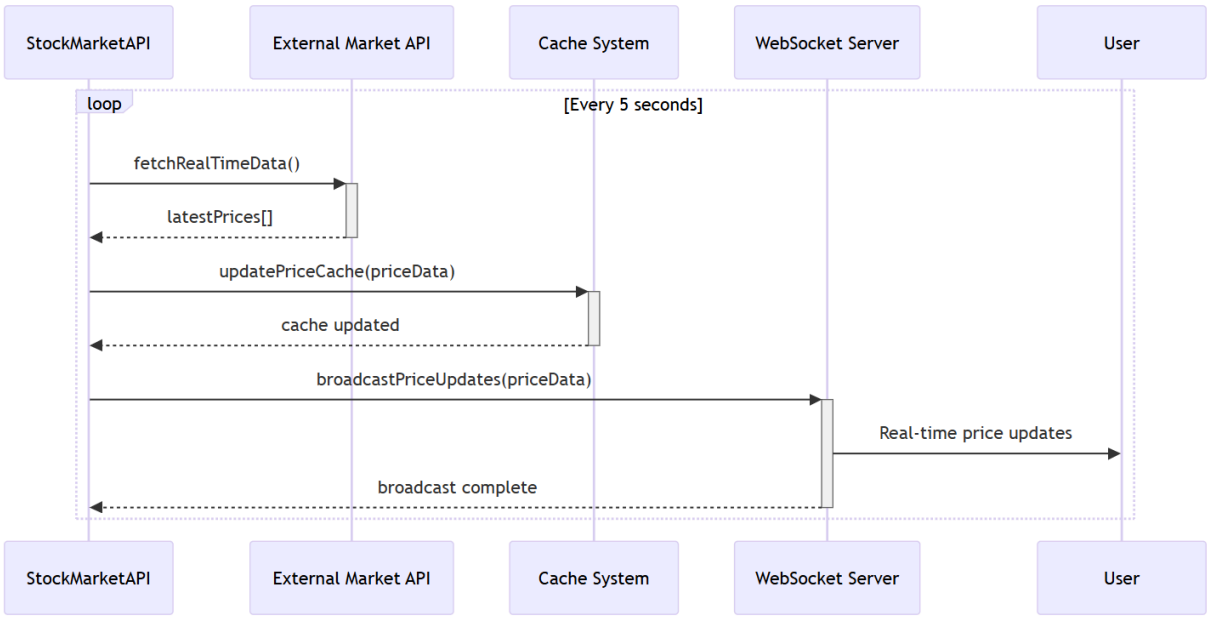
Tournament Results



Notification Delivery



Real time stock data sync



6. State Diagrams

6.1 Diagram details

Object: Trade Order
Description: This diagram illustrates the different states an order can be in, from the moment it's created by a user until it is executed, cancelled, or expires.
States and Messages:

- **Pending:** The order has been created in the system but not yet submitted to the market.
- **Open:** The order has been submitted and is active, waiting for the market conditions (e.g., price match) to be met.
- **Partially Filled:** A portion of the order has been executed, but the full quantity has not yet been traded.
- **Executed:** The order has been successfully completed in full.
- **Cancelled:** The user has manually cancelled the order before it could be fully executed.

- **Expired:** The order was not filled by the end of the trading day or its set time limit, and it has automatically been removed.

Object: User Session

Description: This state diagram models the life cycle of a user's session with the trading simulation platform. It illustrates how the session progresses through various states, from the moment a user first interacts with the site until the session is terminated.

States and Messages:

- **Logged Out**: The initial state when a user first accesses the platform or has not yet authenticated or when closes the app.
- **Awaiting Login**: A temporary state where the system is processing the user's login credentials.
- **Logged In**: The primary active state where the user has full access to the platform's features.
- **Inactive**: An intermediate state triggered by a period of user idleness. The system awaits renewed user activity or session expiration.

Object: AI Assistant

Description: This diagram illustrates the states of the AI Assistant's service. It models the AI's availability and how it handles user requests, processing, and generating responses.

States and Messages:

- **Idle:** The AI Assistant is not currently processing any user queries.
- **Receiving Query:** The AI Assistant has received a request from a user.
- **Processing:** The AI is analyzing the user's query and generating a response.
- **Responding:** The AI is sending the generated response back to the user.
- **Unavailable:** The AI Assistant service is offline or experiencing an issue and cannot process requests.

Object: Educational Module

Description: Represents how a learning module progresses for a user.

States and Messages:

- **Locked**: Content not yet accessible.
- **Available**: Content unlocked and ready to access.
- **In Progress**: User is actively viewing the module.
- **Completed**: User finishes the module.
- **Reviewed**: User revisits or rates the module.
- **Expired**: Content is no longer available due to subscription being expired.

Object: Leaderboard
Description: Shows how the leaderboard updates as user performance changes.

States and Messages:

- **Idle**: Leaderboard exists but no updates happening.
- **Updating**: System is recalculating rankings after trades or achievements.
- **Published**: Updated leaderboard is available for users to view.
- **Error**: Leaderboard update failed due to data retrieval problem.

Object: Portfolio

Description: This diagram represents the states of a user's portfolio on the trading simulator platform. It shows how the portfolio evolves from being empty, to updated when trades occur, to refreshed when new market data is pulled, and how errors are handled (e.g., data unavailability).

States and Messages:

- **Empty:** Initial state when the user has no stocks.
- **Populated:** Portfolio contains one or more stocks.
- **Updated:** A stock is added, sold, or modified, which changes holdings.
- **Refreshing:** The system fetches live market data for the portfolio stocks.
- **Error:** Market data retrieval fails; cached or error message shown.

Object: Notification

Description: This diagram models how notifications move through states from creation to user acknowledgement.

States and Messages:

- **Idle:** No active notifications.
- **Created:** Notification generated (e.g., trade executed, leaderboard update).
- **Delivered:** Notification pushed to the user's interface.
- **Unread:** Notification is visible but not yet opened.
- **Read:** User opens the notification.
- **Dismissed:** User dismisses without opening.
- **Expired:** Notification auto-expires after a set time.

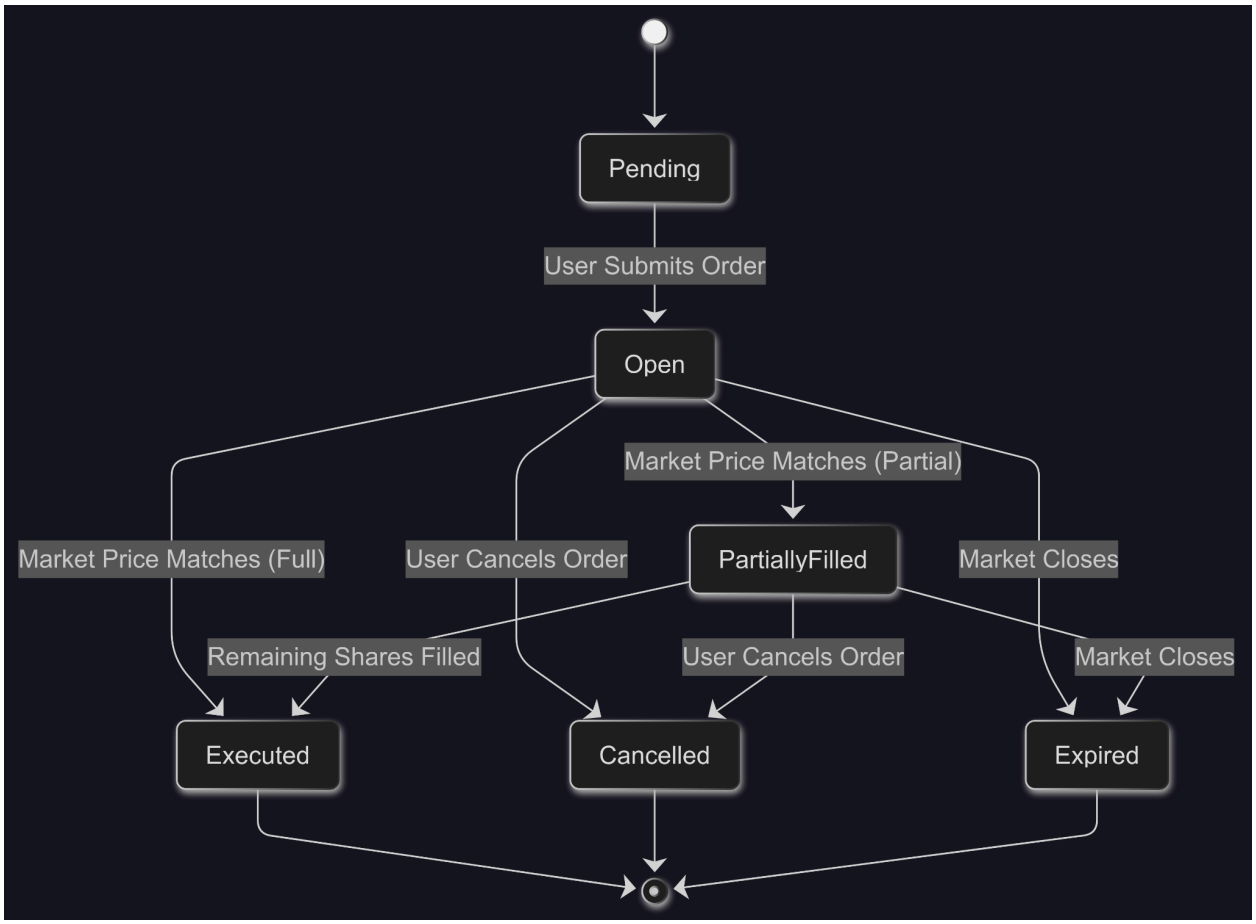
Object: Wallet

Description: This diagram represents the states involved when a user adds funds to their virtual wallet in the simulation platform. It captures successful top-ups, cancellations, and invalid input errors.

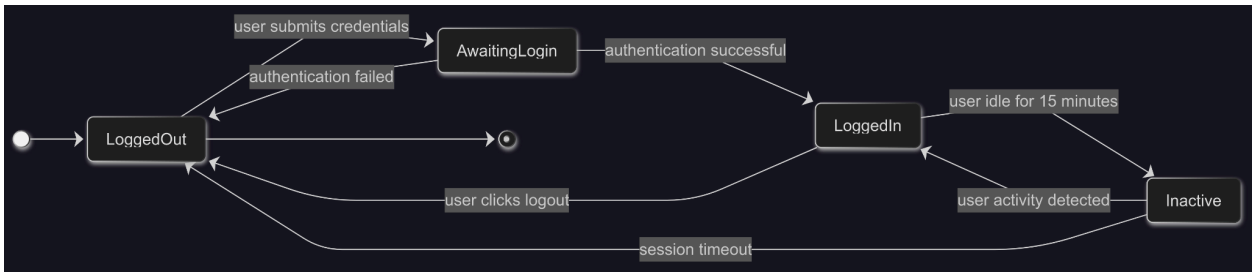
States and Messages:

- **Idle:** The wallet is in its default state, showing the current balance.
- **Awaiting Input:** User has chosen "Add Funds" and must enter an amount.
- **Validation:** The system checks if the input is valid (positive amount).
- **Updated:** Wallet is credited with new funds.
- **Error:** Input was invalid (e.g., negative value).
- **Cancelled:** The user cancelled the top-up process.

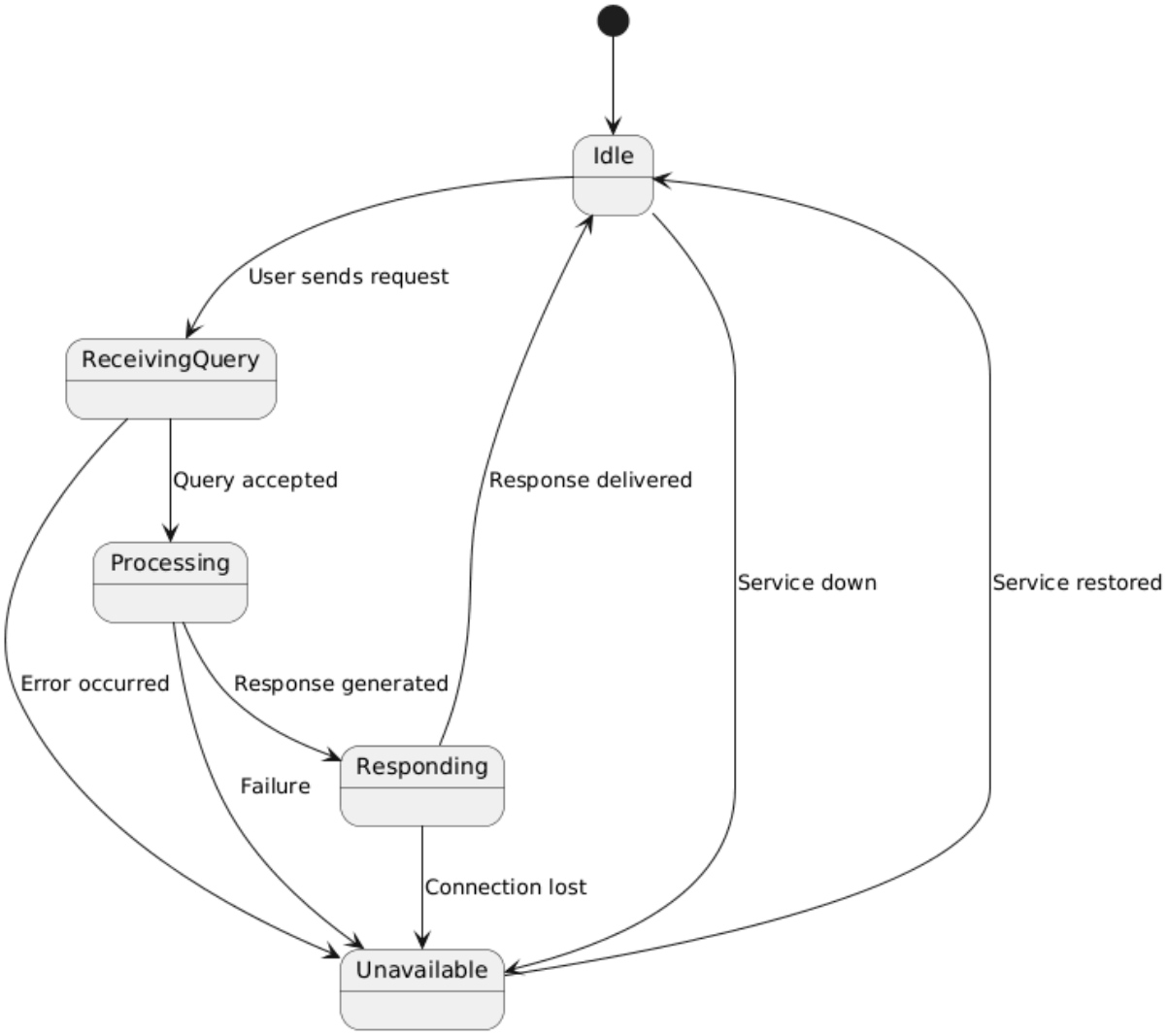
6.2 Diagram



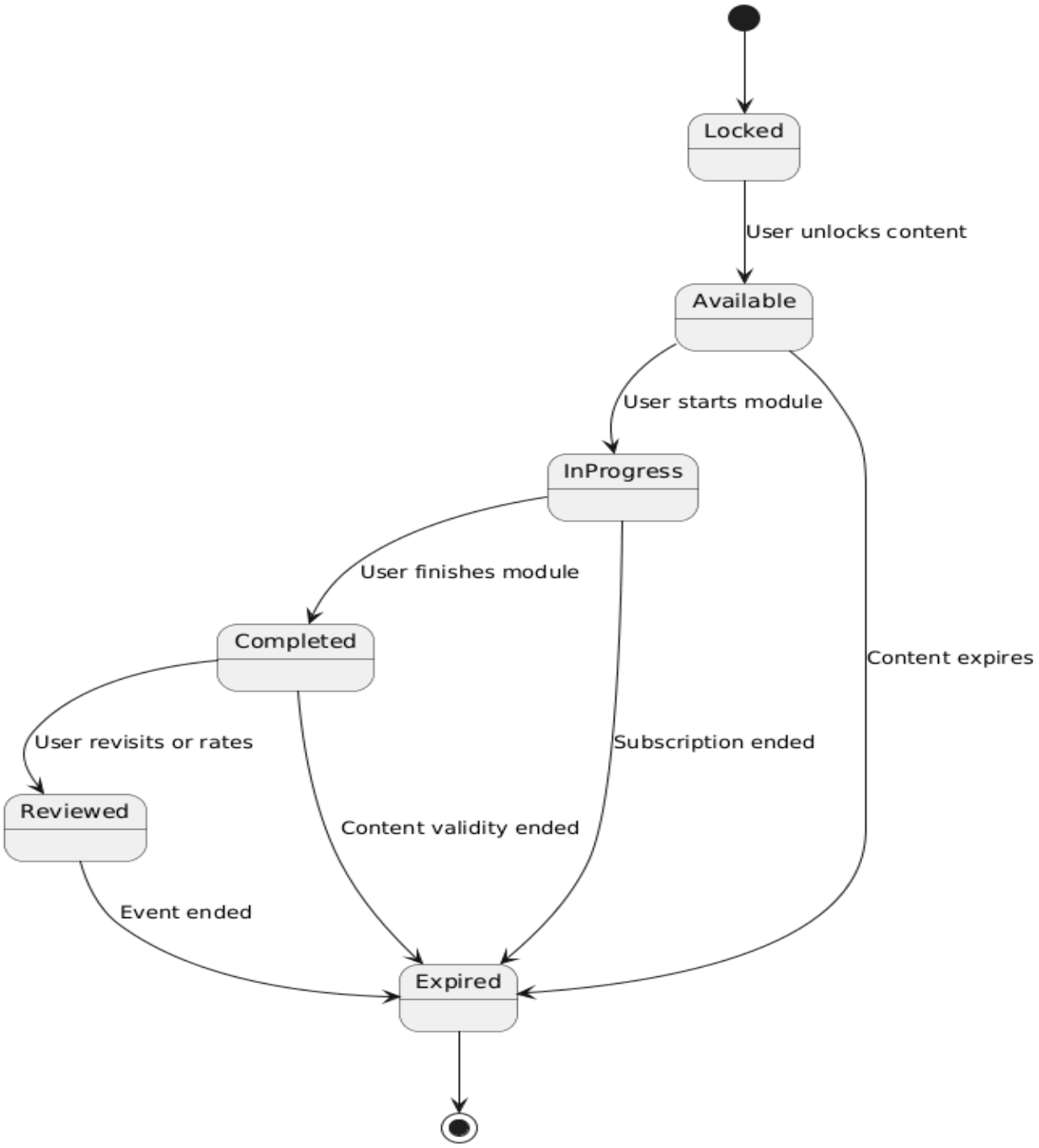
Trade Order



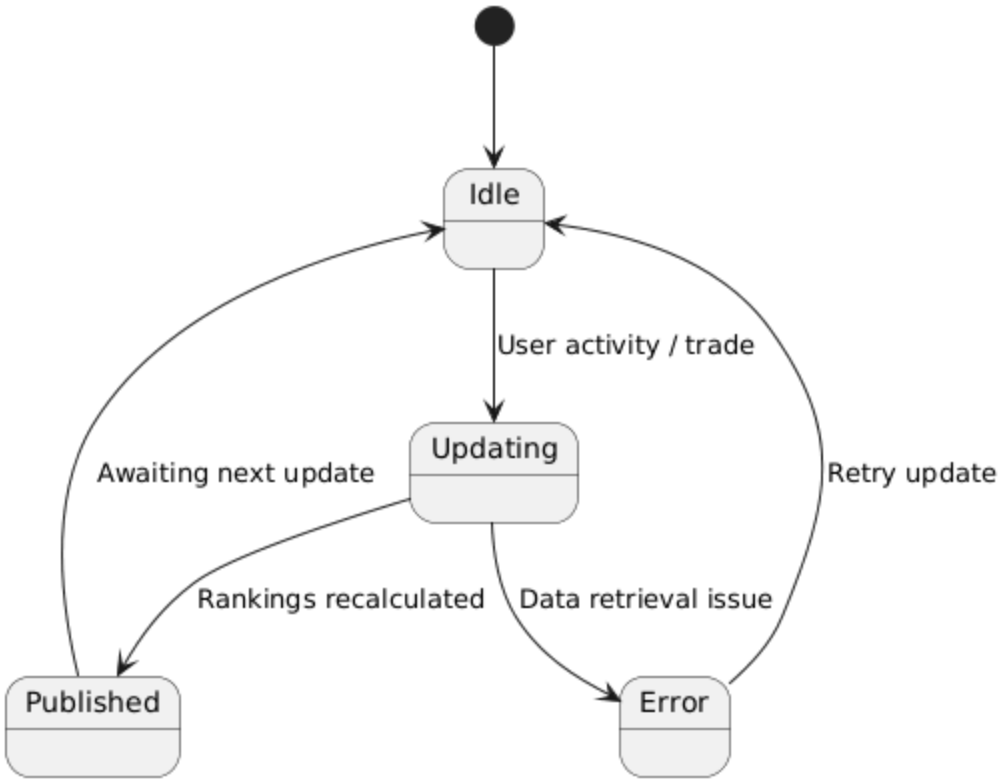
User Session



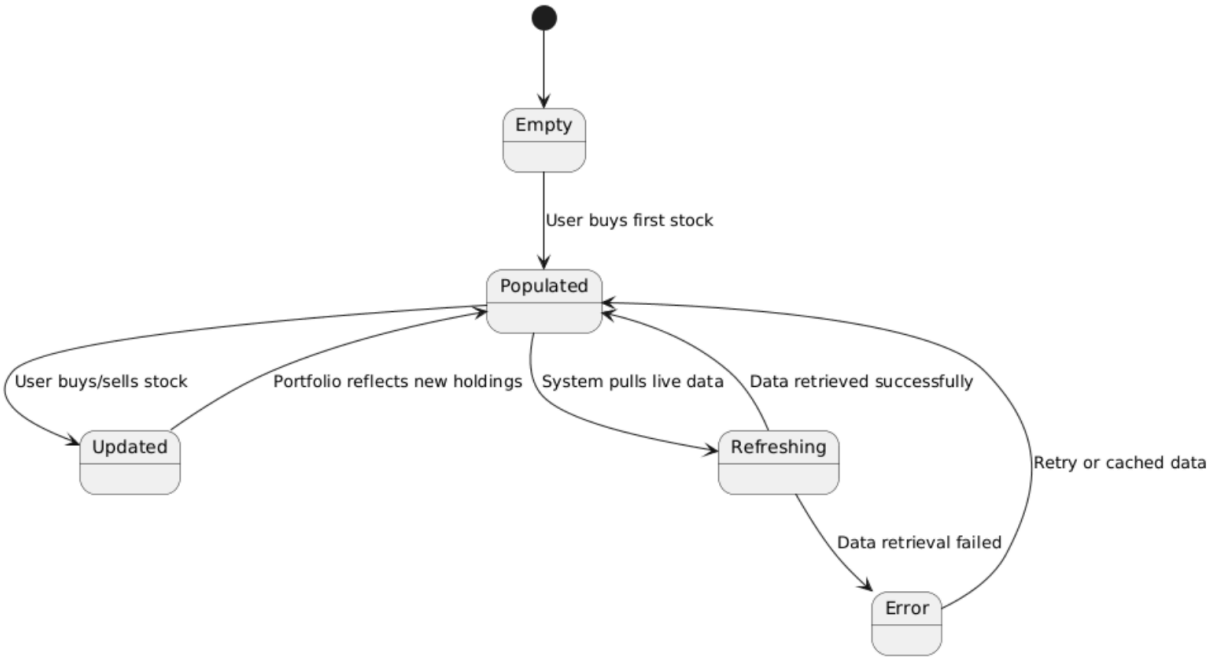
Ai Assistant



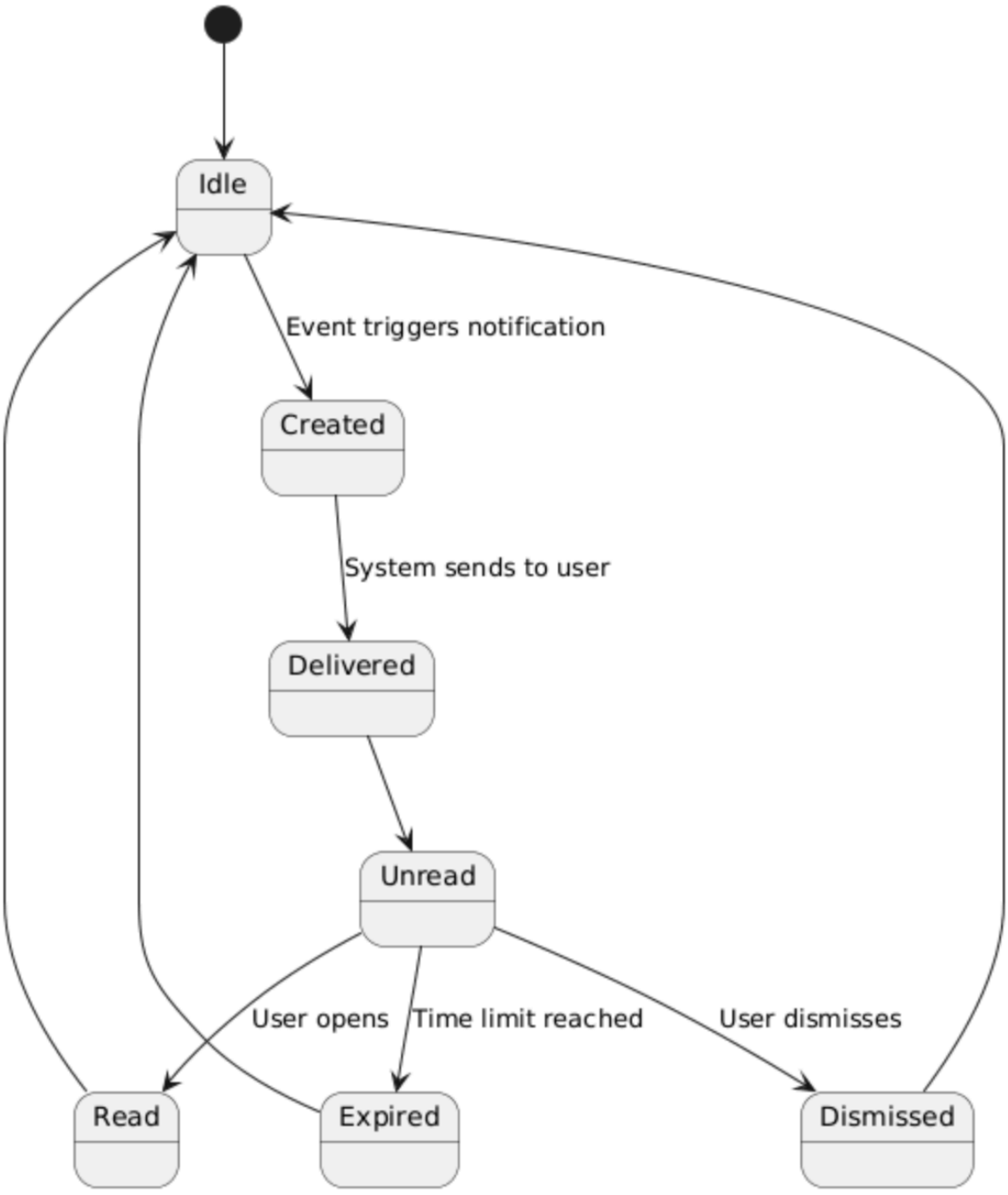
Educational Progress



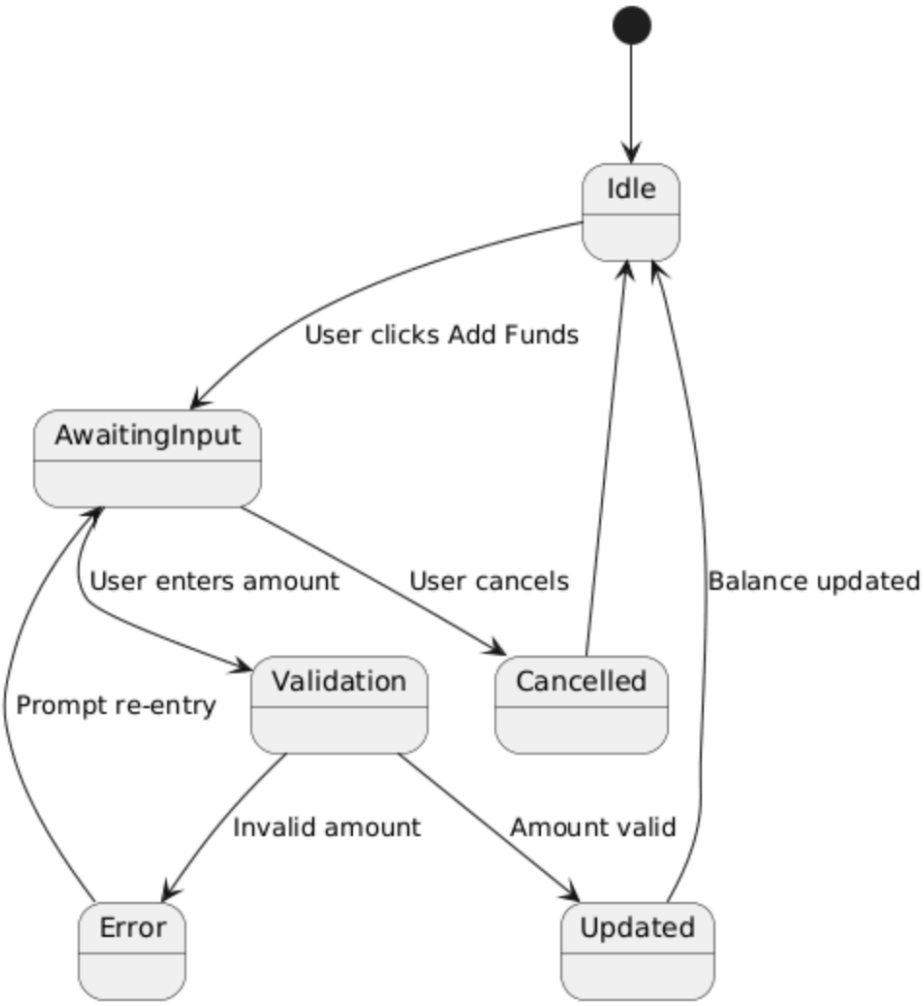
Leaderboard



Portfolio management



Notification



Wallet top up

7. Data Requirements

Data Sources	- Live Stock Market Data Source (PSXTerminal): https://github.com/mumtazkahn/psx-terminal/blob/main/API.md - Historical Stock Market Data Source (for model training): https://www.kaggle.com/datasets/hilalaziz32/pakistan-stock-exchange-data-2021-2024

Data Requirements	**1) How much data do we need?**
	- **Stocks covered:** ~200-500
	- **Minimum size (works for a starter model):**
	  - After turning days into training examples (60-day lookback → predict next day), aim for **≥ 250,000 labeled examples**.
	- **Nice to have (better):**
	  - With ~200–500 PSX listings over ~6000 trading days, we can reach **~1–3M daily rows** raw and **≥ 1M training examples** after windowing/cleaning.
	- **Per-stock history rule:** only train on symbols with **≥ 3 years** of fairly continuous data.
	2) Target label
	Option A — Next-day return (regression)
	"What % will this stock move tomorrow?"
	- Formula:
	return$_{+1}$ = (Close$_{+1}$ − Close$_{}$) / Close$_{}$
	- Example: Close today = 100, tomorrow = 103 → return = **+3%**.

	Option B — Up/Down (classification) "Will the stock be up or down tomorrow?" - Label: **1** if return$_{+1}$ > 0, else **0**. Simple, great for beginners; we can output **probability of going up**. **3) Data Fairness** - **Sectors (types of business):** Don't let one sector dominate. **No sector > 25%**, and any real sector should be **at least 5%** of your data. - **Company size (market cap):** Include **small, medium, and large** companies. Aim for **about one-third each** (roughly 33% ±10%). - **Liquidity (how much a stock is traded):** Split stocks into **5 groups** from least-traded to most-traded. Make sure **each group has at least 15%** of your samples. - **Company age on the exchange (since IPO):** Cover **new (≤2 years), mid-age (2–5 years), and old (>5 years)**. Try for **at least 10%** from each group. - **Market moods (volatility):** Include calm periods **and** crazy ones. Ensure **at least 10%** of your data comes from the **wildest 20%** of days/stocks. - **Delisted/renamed stocks:** Keep them in. Don't train only on survivors.
Model Requirements	**1) Training strategy (how the model will learn)**

	- **Primary target (regression): next-day return** $$y = \frac{\text{Close}_{t+1} - \text{Close}_t}{\text{Close}_t}$$ - **Secondary target (classification): Up/Down** $$y = 1 \text{ if next-day return} > 0, \text{ else } 0.$$ - **Feature window:** last **60 trading days** of features to predict **tomorrow** (t+1). - **Time splits (no leakage):** 1. Train: **2001–2015** 2. Validate: **2016–2019** 3. Test-1 (stress): **2020–2022** 4. Test-2 (recent): **2023–2025** - **Model stack (simple to strong):** - Linear/Logistic (baseline) - Gradient-Boosted Trees (LightGBM/XGBoost) (primary) - **Best on tabular data:** Consistently outperform deep nets without massive data. - **Robust to noisy finance data:** Capture weak, nonlinear signals with strong regularization. - **Recommended**.

8. Non-functional Requirements / Quality Attributes

Sr#	Requirements
1	The system should not utilize more than 4 GB of memory at any time during its execution.
2	System should be up 99% of the time. The system should not fail more than 2 times every 24 hours. In case of a failure, the system should restore to normal operations within 15 minutes of a failure.
3	The system should follow best UI practices as per Jakob Nielsen's 10 usability heuristics and Ben Schneiderman's 8 golden rules.
4	Deploy minor bug fixes or functional modifications within 48 hours, and major bug fixes within 72.
5	The system should be easy to maintain and update with modularity in code and components, and following clean code best principles to ensure a neat code base.
6	Stock market data should be incoming in real-time to simulate the real stock market effectively
7	Leaderboard should be updated everyday after market closes
8	The AI model must maintain 99% uptime and acceptable no more than 5s of response delay time.

9. Security Requirements

Sr#	Security Risks	Potential Losses	Controls
1	A01:2021 – Broken Access Control	Data loss. Trust loss. Business loss.	Deny by default protocol (unless deliberate access granted)
2	A02:2021 – Cryptographic Failures	Data loss. Trust loss. Business secrets loss. Business loss.	Ensure sensitive data is encrypted in database. Ensure data is encrypted when transferring over a transfer protocol.

3	A03:2021 – Injection	Data loss. Database corruption. Trust loss. Database functionality loss. Business loss.	Always “sanitize” or clean up what users type, so harmful stuff is never used directly. Use special “parameterized” queries that keep input and commands separate. Never build command strings by gluing together user input and code.
4	A07:2021 – Identification and Authentication Failures	User account penetration and sensitive information compromise. Business loss.	Implement multi-factor auth. Implement weak password checks. Don’t expose session identifier in URL.
5	A09:2021 – Security Logging and Monitoring Failures	Data loss. Business loss.	Ensure contextual logs are logged for all points of entry in the system.
6	ML02:2023 Data Poisoning Attack	Bad training of the AI model. Incorrect predictions.	Use verifiable and well reputed data. Ensure limited access to DB for AI model.
7	A06:2021-Vulnerable and Outdated Components	Hackable. Constant Bugs	Use updated frameworks and libraries

10. Security Engineer

Name of the Security Engineer	Umar Zubair

11. Use of Generative AI

It was used as help in writing code for the uml diagrams and in research for data requirements.

12. Who Did What?

Name of the Team Member	Tasks done
Muhammad Rayyan Khan	Worked on state diagrams (first 5)
Umar Zubair	Worked on Class and Sequence diagrams
Muhammad Shahmir	Worked on Use cases tables
Mohammed Raiyaan Junaid Hamid	Worked on Use Case Diagram and Data Requirements table
Muhammad Ahmad	Worked on state diagrams

13. Review checklist

Before submission of this deliverable, the team must perform an internal review. Each team member will review one or more sections of the deliverable.

Section Title	Reviewer Name(s)
Data Requirements, use cases	Muhammad Rayyan Khan
Data Requirements,State Diagrams	M Umar Zubair
State Diagrams, Sequence Diagrams, Class Diagram	Mohammed Raiyaan Junaid Hamid

Use cases, Use case tables	Muhammad Shahmir Sher Qazi
Use Cases	Muhammad Ahmad