# Software Requirements Specification
for

SE HACKATHON

Prepared by

Om Bhutkar 202201040111

Vishal Kesharwani 202201040121

Suraj Didwagh 202201040126

Namita Dewang 202201040151

Atharva Solanke 202201040163

Nihar Bhatt 202201040171

MIT Academy of Engineering

MARKET SENTIMENT ANALYZER FROM TWITTER/NEWS

Index

1. Introduction

1.1 Purpose

The purpose of this project is to analyze public sentiment about financial assets (e.g., stocks, cryptocurrencies) using recent tweets and news headlines. The system uses sentiment analysis techniques to classify public sentiment as **positive**, **negative**, or **neutral** and correlates it with actual asset price movements.

Key Objectives of the System:

- To provide **real-time or near real-time sentiment analysis** from Twitter and news platforms.
- To **visualize sentiment trends over time** for selected financial assets.
- To **overlay sentiment scores with historical price data** to find meaningful correlations.
- To offer a **user-friendly dashboard** with filtering and comparison tools.
- To support **data-driven decision-making** for investors, traders, and financial analysts.

This document outlines the **functional and non-functional requirements**, **system architecture**, and **technical references** for the project.

1.2 Document Conventions

This document follows standard conventions and formatting practices, ensuring clarity and consistency. Below are the key conventions followed:

- **Technical terms and keywords** are highlighted in **bold**.
- **UML diagrams** such as **Use Case Diagrams, Class Diagrams, and Sequence Diagrams** are used for system modeling.
- The software follows **modular architecture**, ensuring ease of scalability and integration.
- **Artificial Intelligence (AI) models** are employed for detecting user attributes.
- **Security protocols and encryption methods** are used to safeguard sensitive data.
- All APIs and modules follow **RESTful design principles**.
- **References are cited** in IEEE format to maintain standard academic integrity.

1.3 Intended Audience and Reading Suggestions

This document is intended for different types of stakeholders involved in the project. The content is structured to be relevant for:

- **Software Developers:**
  - To implement APIs, design the dashboard, and integrate sentiment models.
  - To build pipelines for real-time tweet/news fetching and data preprocessing.

- **Project Managers:**
    - To track development progress and ensure milestones and features are delivered on time.
    - To evaluate business value and potential future expansion of the system.
- **Financial Analysts & Traders:**
    - To use sentiment data to enhance decision-making.
    - To identify potential market reactions and patterns.
- **Testers & Quality Assurance (QA) Engineers:**
    - To test sentiment accuracy, API responses, and UI behavior under different conditions.
    - To ensure system performance and availability under high loads.

Suggested Reading Flow:

1. **Introduction:** Understand the problem statement and system goals.
2. **Product Scope:** Explore functional boundaries and expected features.
3. **Requirement Specification:** Detailed list of functional and non-functional requirements.
4. **System Diagrams:** Structural and behavioral design representation.
5. **References:** Review the resources and research backing the technology choices.

1.4 Product Scope

The Market Sentiment Analyzer is a web-based system designed to:

- **Collect tweets and news headlines** related to financial assets.
- **Analyze sentiment** using AI/ML-based NLP models.
- **Overlay sentiment results** with real-time or historical price charts.
- **Provide filtering tools** (date range, asset name, sentiment category).

Key Features of the System:

- **Data Collection Layer:**
    - Connects to **Twitter API** and **News API** for fetching recent headlines and posts.
    - Uses **scheduled jobs or triggers** for periodic updates.
- **Sentiment Analysis Engine:**
    - NLP model (e.g., VADER, BERT) classifies each input as **positive, negative, or neutral**.
    - Scores are aggregated and timestamped for time-series trend generation.
- **Price Data Integration:**
    - Uses sources like **Yahoo Finance API** to fetch historical price movements.

    - Enables visual correlation between sentiment shifts and price changes.
- **Interactive Dashboard:**
    - Provides graphical views (line charts, bar graphs) for sentiment and price.
    - Includes **filters** (date, asset, sentiment type) for deep analysis.

Business Goals and Benefits:

- Helps investors assess **market mood and volatility**.
- Provides **early indicators** of price movement based on public sentiment.
- Improves **investment decision-making** with data-driven insights.
- Can be extended to support **custom alerts**, **portfolio analysis**, and **risk prediction**

1.5 References

The system design and implementation leverage current advances in NLP, sentiment analysis, and data visualization. The following references were used in planning and development:

1. B. Liu, "Sentiment Analysis and Opinion Mining," Synthesis Lectures on Human Language Technologies, 2012.
   • A foundational resource on sentiment classification techniques.
2. M. Hutto and E. Gilbert, "VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text," Eighth International AAAI Conference on Weblogs and Social Media, 2014.
   • Introduces VADER sentiment model used for social media sentiment classification.
3. J. Devlin et al., "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," arXiv preprint arXiv:1810.04805, 2018.
   • Describes the BERT transformer model used for contextual sentiment analysis.
4. Twitter Developer Documentation – https://developer.twitter.com/en/docs
   • Official documentation for accessing Twitter data via API.
5. NewsAPI Documentation – https://newsapi.org/docs
   • News headline and article data access documentation.
6. Yahoo Finance API (unofficial) – https://www.yahoofinanceapi.com/
   • Price data source for financial assets.

---

2. Overall Description

2.1 Product Perspective

The **Market Sentiment Analyzer from Twitter/News** is a standalone web-based application that leverages natural language processing (NLP) and sentiment analysis to assess public opinion regarding financial assets such as stocks and cryptocurrencies. It collects real-time or recent data from social media platforms (e.g., Twitter) and news APIs, classifies sentiment into positive, negative, or neutral categories, and visualizes sentiment trends alongside price movement charts.

Key System Components:

- **Frontend Dashboard (UI):** Provides interactive data visualization and filtering tools for end users.
- **Data Fetching Module:** Integrates with APIs like Twitter API, NewsAPI, or RSS feeds to gather content.
- **Sentiment Analysis Engine:** Applies NLP models to analyze and classify sentiment in textual data.
- **Asset Price Tracker:** Fetches historical and real-time price data for selected financial assets.
- **Database:** Stores tweets, headlines, sentiment scores, and asset price data for analysis and review.
- **Visualization Engine:** Renders sentiment trends and overlays them with price movement graphs.

System Architecture Overview (Simplified Diagram)

```
[User] → [Dashboard UI]
              ↓
     [Filters & Controls]
              ↓
   [API Data Fetcher] → [Sentiment Engine] → [Database]
                                ↓
                  [Visualization + Price Overlay]
```

• **User**: Initiates sentiment analysis by selecting a financial asset, date range, and data source (Twitter or News).
• **Dashboard UI**: Displays input fields, filters, and visual outputs for user interaction.
• **Filters & Controls**: Applies user-defined filters (asset, date, source) to structure the data query.
• **API Data Fetcher**: Retrieves relevant tweets or news headlines based on filters using external APIs (e.g., Twitter API, News API).
• **Sentiment Engine**: Analyzes the fetched text using NLP and classifies each entry as positive, negative, or neutral.
• **Database**: Stores fetched data, sentiment scores, and metadata for analytics and future reference.
• **Visualization + Price Overlay**: Displays sentiment trends over time and overlays them with actual price movements of the selected asset to identify correlations.

2.2 Product Functions

The core functionalities of the system include:

1. **Real-Time Data Collection:**
   - Fetches recent tweets and news articles using Twitter API / News API.
   - Supports keyword-based and asset-based search filters (e.g., "Tesla", "Bitcoin").
2. **Sentiment Classification:**
   - Uses pretrained or fine-tuned NLP models (e.g., BERT, VADER) to categorize sentiment.
   - Labels each post or headline as *Positive*, *Negative*, or *Neutral*.
3. **Trend Visualization:**
   - Displays sentiment trends over time using interactive line charts or bar graphs.
   - Allows users to explore patterns in public opinion for specific assets.
4. **Asset Price Overlay:**
   - Fetches asset price data from financial APIs (e.g., Yahoo Finance, Alpha Vantage).
   - Overlays price data on sentiment graphs to analyze correlation.
5. **User Filters and Customization:**
   - Filters by date range, sentiment type, asset, or source (Twitter/News).
   - Allows export or download of insights for further analysis.

2.3 User Classes and Characteristics

The system is designed for different **user classes**, each with specific roles and requirements:

- **Retail Investors:**
    - Purpose: Use sentiment trends to guide personal investment decisions.
    - Technical Skill: Basic; requires intuitive and visual UI.
- **Financial Analysts:**
    - Purpose: Conduct deeper sentiment-asset correlation studies.
    - Technical Skill: Moderate; interested in both trends and raw data.
- **Developers and Researchers:**
    - Purpose: Extend, train, or integrate custom models.
    - Technical Skill: High; may access backend or API-level functionality.
- **Business Stakeholders / Decision Makers:**
    - Purpose: Monitor public sentiment to make branding or strategic decisions.
    - Technical Skill: Low to Moderate; prefer summarized dashboards.

2.4 Operating Environment

The software is designed to run in the following environment:

Hardware Requirements:

- Client Device: Any modern device with a browser (PC, laptop, tablet).
- Server: Minimum 8GB RAM, Quad-core CPU; GPU recommended for large NLP model inference.

Software Requirements:

- OS: Linux (Ubuntu), Windows Server, or macOS (for local development).
- Frontend: HTML, CSS, JS framework.
- Backend: Python (Flask/FastAPI).
- Database: MongoDB for data storage (Articles).

- ML Libraries: Pands , Numpy , Flask or PyTorch.

APIs & Services:

- Twitter API v2 / NewsAPI for content.
- Financial APIs (e.g., Alpha Vantage, Yahoo Finance) for asset prices.

Network Requirements:

- Stable internet connection for live API queries.
- Encrypted communication using HTTPS and secure API keys.

2.5 Design and Implementation Constraints

Several constraints and limitations may affect the system design and implementation of the Market Sentiment Analyzer:

1. **Regulatory & Compliance Constraints**:
   • Must comply with data privacy laws (e.g., GDPR, CCPA) when collecting or processing user-generated content (tweets or news).
   • API usage must adhere to terms and conditions of platforms like Twitter and news providers.
   • User-identifiable data (if any) must be anonymized or excluded from storage and analysis.
2. **Hardware Limitations**:
   • Real-time processing and visualization may require moderate-to-high server specifications, especially during peak usage.
   • For larger-scale deployments, additional resources like cloud instances or load balancers may be necessary.
3. **Software & Integration Constraints**:
   • API rate limits (e.g., Twitter API) may restrict the frequency or volume of data that can be fetched.
   • Sentiment analysis model accuracy may be affected by sarcasm, slang, or ambiguous text.
   • Integration with live price feeds (e.g., from stock or crypto APIs) must account for latency and data availability issues.
4. **Performance Constraints**:
   • The system must process and classify sentiment data within a few seconds of retrieval to maintain near real-time responsiveness.
   • High-frequency data fetching or updates may require asynchronous processing or distributed

systems.

• Heavy visualization (e.g., multi-line sentiment vs. price graphs) must be optimized for performance on web dashboards.

2.6 User Documentation

The following documentation will be provided with the software:

1. **User Manual:**
    - Instructions for asset selection, filtering, and trend analysis.
    - Visual aids to explain sentiment interpretation.
2. **API Documentation:**
    - Endpoints for fetching analyzed sentiment and asset data.
    - Authentication, rate limits, and request structure.
3. **Developer Guide:**
    - Architecture, ML model implementation, and deployment steps.
    - Setup for local development and production deployment.
4. **Technical Documentation:**
    - Common issues with API access, dashboard loading, and interpretation.

2.7 Assumptions and Dependencies

Several assumptions and dependencies exist that could impact system performance:

Assumptions:

- Users will have internet access to fetch data from live sources.
- Financial asset prices and tweets are publicly available through APIs.
- Sentiment classification models are pre-trained and reasonably accurate.

Dependencies:

- Twitter/News APIs for data input.
- Financial APIs (e.g., Alpha Vantage) for price overlays.
- Hosting servers or cloud services for model inference and data storage.
- Python/JavaScript runtime environments for backend/frontend processing.

---

3. External Interface Requirements

3.1 User Interfaces

The Market Sentiment Analyzer system features a clean and interactive web-based dashboard to provide users with real-time insights into market sentiment from Twitter and news sources.

Key UI Components:

1. **Login & User Access Panel**
    - Authenticated access to the system with user roles (Admin/User).
    - Supports session-based login or OAuth with third-party accounts (e.g., Google, Twitter).
2. **Dashboard UI**
    - Central hub where users select assets (e.g., stock symbols, cryptocurrencies) and data sources (Twitter, news).
    - Includes sentiment filters (positive/neutral/negative) and time range selectors.
    - Displays real-time visualizations (line charts, bar graphs, pie charts).
    - Shows sentiment scores, trend lines, and asset price overlays.
3. **Visualization Panels**
    - **Sentiment Trend Chart**: Shows sentiment variation over time.
    - **Word Cloud**: Displays frequently used words in tweets/news.
    - **Price Overlay Graph**: Correlates market sentiment with historical price movements.
4. **Admin Console (For Analysts/Developers)**
    - View API status, model confidence scores, and monitor fetch/logging activity.
    - Allows for adjusting model thresholds, keywords, or blacklist sources.

UI Guidelines:

- **Responsive Design** for desktop and mobile.
- **Accessibility Support** (keyboard navigation, screen reader compatibility).
- **User Feedback** (loading spinners, error alerts, success toasts).
- **Standard Controls:**
    - “Apply Filters” – refresh dashboard based on selected parameters.
    - “Export Data” – download logs and analysis reports as CSV or JSON.
    - “Reset” – clears all filters and re-fetches default data.

3.2 Hardware Interfaces

The system is largely software-based and cloud-deployable but may interface with local hardware for performance acceleration.

Relevant Hardware Interfaces:

- **User Devices:**
  - Standard computers or laptops with modern browsers (Chrome, Firefox, Safari).
  - Optional: GPU-enabled servers for local AI model inference.
- **Processing Hardware (if on-prem):**
  - **Recommended CPU**: Intel i5/Ryzen 5 or higher.
  - **Recommended RAM**: 8 GB or more.
  - **GPU Acceleration**: NVIDIA CUDA-enabled GPUs for local sentiment analysis (e.g., RTX 2060+).
  - **Storage**: Minimum 100 GB for storing logs, model checkpoints, and cached tweets/news.

Data Flow:

1. User selects filters →
2. System fetches data via APIs →
3. AI model processes text locally/cloud →
4. Results sent to frontend for visualization.

3.3 Software Interfaces

The system integrates with several external services and uses multiple frameworks/libraries for data fetching, analysis, and storage.

Operating System Compatibility:

- **Cloud-based**: OS-agnostic (Docker-deployed).
- **Local setup**: Windows 10/11, Linux (Ubuntu 20.04+), macOS Monterey+.

Programming Languages & Frameworks:

- **Frontend**: HTML / CSS / Javascript
- **Backend**: Python (FastAPI/Flask)
- **AI & NLP**: Python with Hugging Face Transformers, Vader, or BERT-based models
- **Visualization**: Chart.js / D3.js / Recharts

API & Software Components:

Component	**Purpose**
Twitter API / News API	Fetch tweets and headlines
Hugging Face / NLTK / Vader	Perform sentiment analysis using NLP
RESTful API Endpoints	Interface between frontend and backend
Chart.js / Recharts	Plot sentiment and price trends
Auth0 / Firebase Auth	Secure login and user management

Data Flow & Communication Between Software Components:

1. **User registers** → Software triggers backend request to fetch sentiment data..
2. **Backend server** → Collected text data is sent to the sentiment analysis model.
3. **AI model processes input** → Predicted sentiment scores (positive/neutral/negative) .
4. **(positive/neutral/negative)→** Frontend displays data visualizations and overlays on asset price trends..

3.4 Communications Interfaces

The system uses secure, real-time communication protocols to fetch external data and update visualizations.

Network Protocols Used:

The system uses secure, real-time communication protocols to fetch external data and update visualizations.

Network Protocols Used:

- **HTTPS (TLS 1.2/1.3)**: Secure data transmission.
- **REST APIs**: Fetch data from Twitter/News sources and internal NLP endpoints.
- **WebSockets (Optional)**: For pushing real-time updates to dashboard without reloads.

Data Formats:

- **JSON**: Primary format for data exchange between frontend and backend.
- **CSV Export**: For downloading logs or report generation.
- **XML (Optional)**: Only if integrating with legacy external systems.

Authentication & Security:

- **OAuth 2.0 / Bearer Token**: For secure access to external APIs (e.g., Twitter API).
- **JWT (JSON Web Tokens)**: Used for session-based user authentication.
- **AES-256 Encryption**: Secures sensitive logs or data in transit.
- **API Rate-Limiting & Throttling**: Prevent abuse and maintain system performance.

---

4. Other Nonfunctional Requirements

4.1 Performance Requirements

The system is designed for fast and reliable sentiment analysis of real-time social media and news data related to financial assets.

Response Time:

- The system must fetch and display tweets/news headlines within **2–4 seconds** of request.
- Sentiment classification per item should complete within **1 second**.
- Graph rendering and updates should occur within **2 seconds** of data analysis.

Accuracy & Reliability:

- Sentiment analysis accuracy should exceed **85%** (based on benchmark datasets or fine-tuned models).
- Correlation analysis with asset prices should show consistent **trend alignment** in over **75%** of tested scenarios.
- The system should maintain performance for up to **200 concurrent users**.

Load Handling & Scalability:

- Must support **100+ sentiment evaluations per hour** during high-traffic periods.
- Backend services should auto-scale using containerized deployment (e.g., Kubernetes) to maintain performance.
- Database must handle at least **20 read/write operations per second** for real-time trend tracking.

System Availability:

- The system must maintain **99.9% uptime**.
- Should implement **auto-recovery and failover mechanisms** for all critical services (API, database, UI).

4.2 Safety Requirements

To ensure data integrity and reduce risk of misinformation or errors in financial analysis:

Data Privacy & Protection:

- No user-identifiable data is stored.
- Tweets and news are stored temporarily and encrypted in transit and at rest.

Failure Handling & Error Recovery:

- If data fetch fails (due to API rate limits or network), the system should retry automatically or provide a fallback.
- If sentiment classification fails, a "Neutral" fallback with warning indicator is shown.
- If the asset price API fails, cached historical data is shown with a notification.

Compliance:

- The system must comply with **GDPR** for any user interaction.
- Financial data used must be from **authorized APIs** (e.g., Yahoo Finance, Alpha Vantage, etc.).

4.3 Security Requirements

As a data-driven financial tool, secure operations and API use are critical.

Authentication & Authorization:

- OAuth 2.0 / JWT for secure user logins.
- Admin-only access to configuration settings (e.g., API keys, model tuning).

Data Encryption & Protection:

- All API communications use **TLS 1.3 encryption**.
- Sensitive credentials and API keys are stored using **env secrets** or **vaults**.

Intrusion Prevention & Logging:

- Brute-force protection and DDoS rate-limiting enabled.
- All login attempts and admin actions are logged with timestamps.

API Security:

- API endpoints must require API keys with rate limiting.
- Use **CORS policies** to restrict cross-origin requests.

4.4 Software Quality Attributes

The software must adhere to high-quality standards to ensure efficiency, security, and reliability.

Attribute	Requirement
Availability	99.9% uptime.
Scalability	100+ tweets/headlines processed hourly.
Accuracy	>85% sentiment classification accuracy.
Interoperability	Supports Twitter API, News API, and Financial APIs.
Security	Encrypted data, role-based access, secure APIs.
Reliability	Failover-ready, auto-retry mechanisms.

Usability	Intuitive dashboard with real-time trend graphs.
Portability	Compatible with all major browsers; deployable on Linux servers or cloud (e.g., AWS, GCP).

4.5 Business Rules

- **Sentiment Classification Rule:**
  - Each tweet/news item must be classified as **positive**, **negative**, or **neutral**.
  - Confidence scores below a threshold (e.g., 0.6) are marked **inconclusive**.
- **Data Retention Rule:**
  - Tweet/news data is cached for **12–24 hours** only.
  - Logs and trend data are anonymized and stored for analysis.
- **User Access Rule:**
  - Admin users can configure asset symbols, update models, and download reports.
  - General users can view sentiment trends and correlations but cannot access raw data.
- **Rate Limiting Rule:**
  - API usage per user session is rate-limited (e.g., 60 requests/minute).
  - If exceeded, the system provides a graceful fallback and notifies the user.
- **Price Correlation Rule:**
  - Sentiment graphs must overlay price trends for the same time window.
  - If price data is missing, sentiment graph is displayed with a **data unavailable** warning.

---

5. Other Functional Requirements

5.1 Performance Requirements

Response Time Requirements:

- Dashboard loading time shall not exceed 3 seconds under normal load
- Sentiment analysis processing shall complete within 2 seconds per text item
- API data retrieval shall timeout after 30 seconds maximum
- Database queries shall return results within 500 milliseconds
- Real-time updates shall appear within 5 seconds of data availability

Throughput Requirements:

- System shall handle analysis of up to 1000 tweets per hour
- Database shall support concurrent access by up to 50 users
- API request processing shall manage 10 concurrent external calls
- Data storage shall accommodate 100MB of new data per day

Scalability Requirements:

- System architecture shall support horizontal scaling for increased load
- Database design shall handle growth to 1 million records without performance degradation
- API integration shall adapt to rate limit changes automatically

5.2 Safety Requirements

Data Protection:

- System shall implement backup procedures for all collected data
- Database failures shall not result in complete data loss
- API service disruptions shall not crash the application
- User inputs shall be validated to prevent injection attacks

Error Handling:

- System shall gracefully handle external API service outages
- Invalid user inputs shall generate appropriate error messages
- System failures shall log detailed error information for debugging
- Data corruption scenarios shall trigger automatic recovery procedures

5.3 Security Requirements

Authentication and Authorization:

- API keys shall be stored securely using environment variables
- Database access shall require authentication credentials
- User sessions shall timeout after 4 hours of inactivity
- Administrative functions shall require elevated privileges

Data Security:

- All client-server communication shall use HTTPS encryption
- API credentials shall be encrypted at rest
- User data shall comply with privacy protection standards
- System logs shall not contain sensitive information in plain text

Input Validation:

- All user inputs shall be sanitized to prevent XSS attacks
- SQL injection prevention through parameterized queries
- File upload restrictions to prevent malicious content
- Rate limiting to prevent API abuse and DDoS attacks

5.4 Software Quality Attributes

Reliability:

- System uptime shall exceed 99% during business hours
- Mean time between failures (MTBF) shall exceed 720 hours
- Data accuracy shall maintain 95% consistency with source APIs
- Error recovery procedures shall restore service within 15 minutes

Usability:

- User interface shall be intuitive requiring minimal training
- Dashboard navigation shall follow standard web conventions
- Error messages shall provide clear guidance for resolution
- Help documentation shall be accessible within 2 clicks

Maintainability:

- Code shall follow PEP 8 standards for Python development
- System components shall be modularly designed for independent updates

- API integrations shall be abstracted for easy provider changes
- Database schema shall support version migration procedures

Portability:

- System shall operate on Linux and Windows server environments
- Database layer shall support migration between SQLite and PostgreSQL
- Frontend shall function across Chrome, Firefox, Safari, and Edge browsers
- Deployment configuration shall be containerized using Docker

5.5 Business Rules

Data Usage Rules:

- Twitter data usage shall comply with Twitter API Terms of Service
- News content shall respect copyright and fair use limitations
- Financial data shall be used for analysis purposes only, not redistribution
- User-generated configurations shall be stored for maximum 90 days

Access Control Rules:

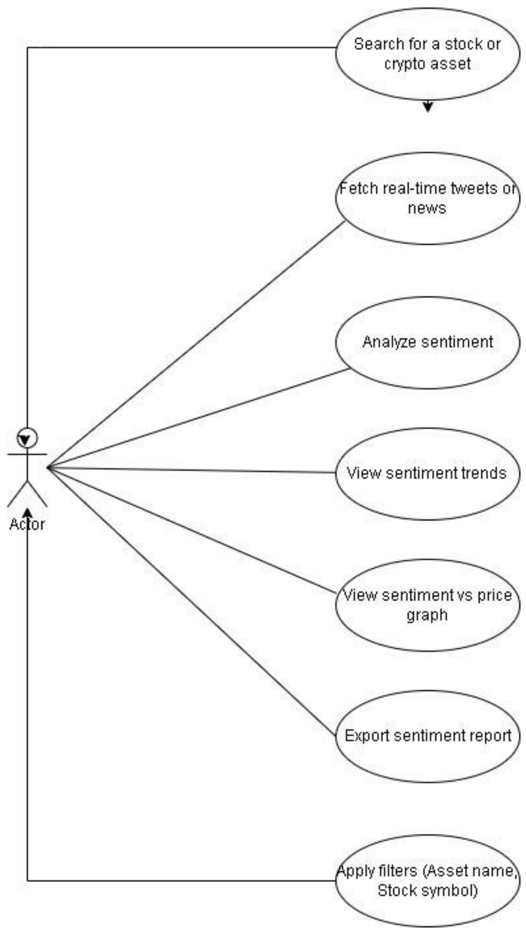
- System administrators shall have full access to all features and data
- Regular users shall have read-only access to analysis results
- API usage quotas shall be monitored and enforced automatically

Data export functionality shall be limited to prevent API quota exhaustion

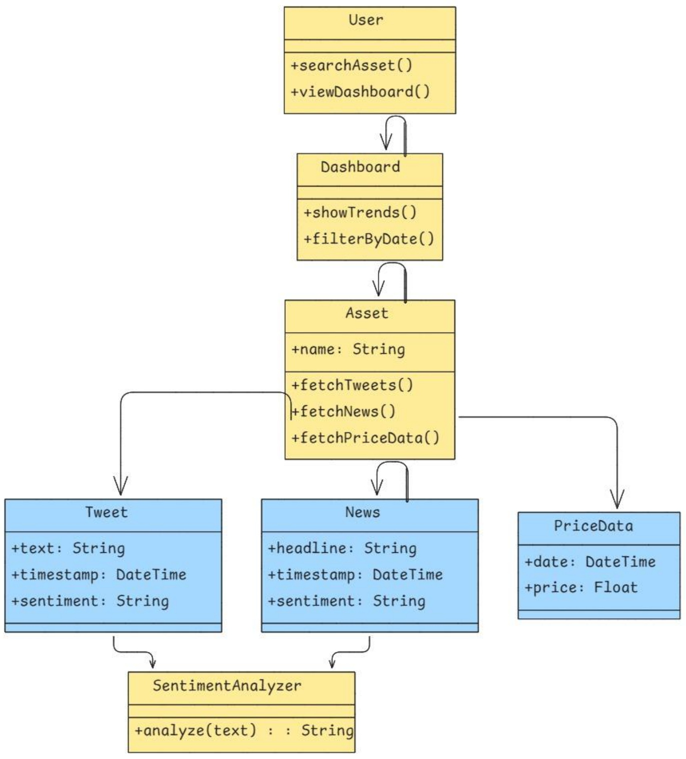
---

6.UML Diagrams

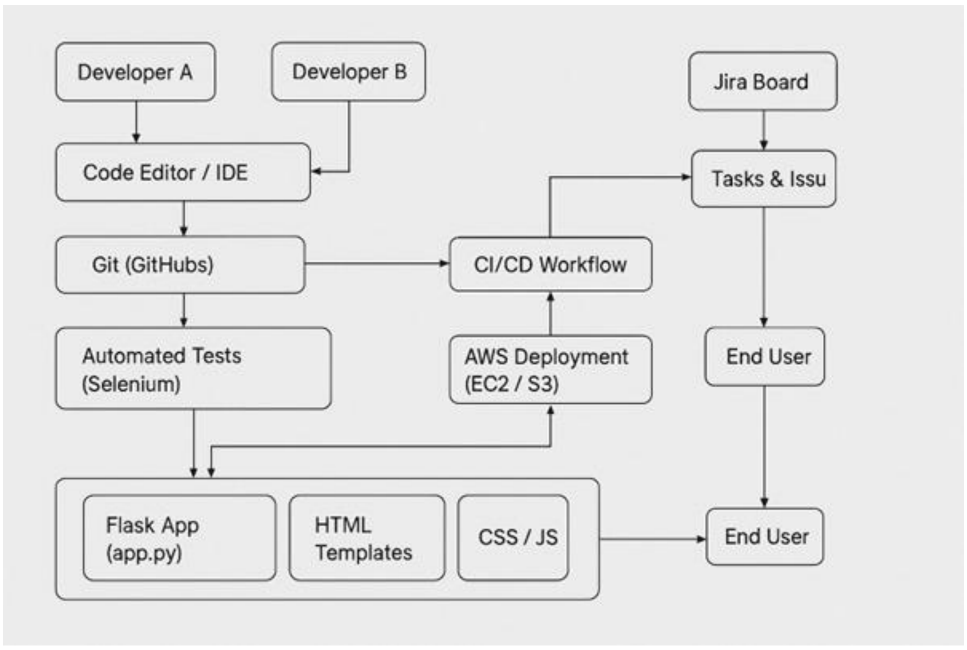
6.1 Use Cse Diagram


Search for a stock or crypto asset
Fetch real-time tweets or news
Analyze sentiment
View sentiment trends
View sentiment vs price graph
Export sentiment report
Apply filters (Asset name, Stock symbol)
Actor

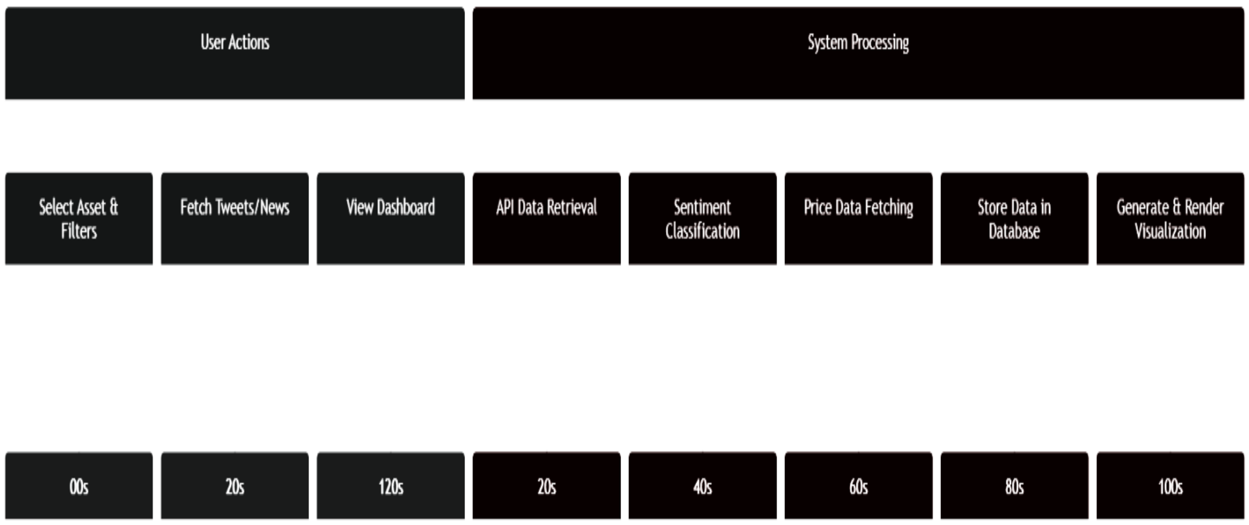
6.2 Object Diagram



6.3 Deployment Diagram



6.4 Time Diagram


User Actions
System Processing
Select Asset & Filters
Fetch Tweets/News
View Dashboard
API Data Retrieval
Sentiment Classification
Price Data Fetching
Store Data in Database
Generate & Render Visualization
00s
20s
120s
20s
40s
60s
80s
100s