Elijah Senior
Tradie

TRADIE SRS

5.1 Introduction

The Insider Trading Data Mobile App is designed to provide real-time access to insider trading filings, stock market data, and predictive analytics. It aggregates data from OpenInsider through web scraping, processes it using Pandas, and serves it via a FastAPI backend to a React Native mobile app. Additionally, it integrates Gemini and Yahoo Stocks APIs for enhanced stock insights. User authentication and API security are managed via Clerk.

The remainder of this document is structured as follows. Section 5.2 describes the functional requirements of the system. Section 5.3 outlines the performance requirements. Section 5.4 details the development and execution environment requirements.

5.2 Functional Requirements

Functional requirements define the features and capabilities of the system.

5.2.1 Data Scraping and Processing

- 5.2.1.1 The system shall scrape insider trading data from OpenInsider using BeautifulSoup.
- 5.2.1.2 The scraped data shall be processed and stored in a Pandas DataFrame for further analysis.
- 5.2.1.3 The system shall clean and normalize the data before making it available via the API.

5.2.2 API Development and Data Serving

- 5.2.2.1 The system shall use FastAPI to provide an API for accessing insider trading data.
- 5.2.2.2 The API shall support authenticated requests using Clerk.
- 5.2.2.3 The API shall provide endpoints for retrieving stock insights from Gemini and Yahoo Stocks APIs.

5.2.3 Mobile App Features

- 5.2.3.1 The React Native mobile app shall display insider trading data in an intuitive UI.

- 5.2.3.2 The app shall allow users to search for specific stocks and view historical trading patterns.
- 5.2.3.3 The app shall provide real-time stock updates based on API data.
- 5.2.3.4 The app shall include user authentication via Clerk.

5.3 Performance Requirements

5.3.1 API Response Time

- 5.3.1.1 The API shall return responses within 2 seconds for 95% of requests.
- 5.3.1.2 The system shall handle at least 100 concurrent requests without significant performance degradation.

5.3.2 Data Processing Efficiency

- 5.3.2.1 The system shall complete the web scraping process within 30 seconds.
- 5.3.2.2 The processed data shall be updated at least once per hour.

5.3.3 Mobile App Performance

- 5.3.3.1 The app shall load stock details within 3 seconds of user request.
- 5.3.3.2 The app shall support real-time data refresh with a maximum delay of 5 seconds.

5.4 Environment Requirements

5.4.1 Development Environment Requirements

- The system shall be developed using Python, BeautifulSoup, Pandas, and FastAPI.
- The front end shall be built using React Native.
- Clerk shall be used for API authentication.

5.4.2 Execution Environment Requirements

- The backend shall be hosted on a cloud platform supporting FastAPI.
- A database or caching system (e.g., PostgreSQL, Redis, Firebase) shall be used for data storage.
- The mobile app shall be deployable on both iOS and Android platforms.