Stock Analyzer

Authors: I Huan C., Cameron C.
Revision: May 26

Introduction:
The Stock Analyzer is an interactive Java application that provides users with an in-depth view of stock market data. The primary goal of this program is to allow users to visualize past stock trends, predict possible future stock prices, and simulate investments. Users can either web scrape stock data live or (optionally) upload a CSV file containing historical stock prices.
The program will also include sentiment analysis by leveraging NLP libraries to interpret current news articles related to selected stocks, helping to predict stock movements based on public perception. A stock market simulation will allow users to "invest" a virtual budget and track their investment performance over time.
This program is designed for anyone interested in understanding stocks better — from beginners wanting a visual representation to more advanced users who want prediction and simulation tools.

Instructions:

Main Menu:

- When the program launches, you'll see a Main Menu with two primary options:
  ```
  1 – Stock Analysis
  2 – Investment Simulation
  3 – Exit
  ```
- You can select an option using your number keys or by clicking the corresponding button.

Stock Analysis Mode:

- If Stock Analysis is chosen, you will see:
  - Stock data graphing box
  - Stock data range buttons
  - Various stock data analysis buttons
  - Future stock prediction button
  - Company Selection Button
- If you press company selection, you will see:
  - Current stock status
  - Text box to enter a ticker symbol
  - Button to choose from CSV Files

Simulation Mode:

- Selecting Simulation Mode starts you with $10,000.
- There are 6 different companies you can invest in
- You will see:
  - Tickers to choose from

    - Graph Box
    - Range Buttons
    - Buy, Sell, Next, and Exit buttons
    - Your current cash amount and stock holdings
- If you click the buy or sell button:
    - You will be able to type in the amount you want to purchase or sell

<u>Features List (THE ONLY SECTION THAT CANNOT CHANGE LATER)</u>:

Must-have Features:

- **Graph stock price data**
    - Display historical stock prices on a line graph using data from a file or live web scraping.
- **Web scraping for stock prices**
    - Use Java libraries (like Jsoup) to pull real-time stock information without manual file uploads.
- **Predict future stock prices**
    - Apply a basic predictive algorithm (e.g., linear regression) to extrapolate future trends based on past data.
    - Simple moving average
- **Investment simulation**
    - Give users a virtual portfolio to simulate buying/selling stocks and track profits/losses.
- **User-friendly GUI**
    - Intuitive interface with clickable menus and clear graph displays (Java Swing or JavaFX).

Want-to-have Features:

- **Sentiment analysis via NLP**
    - Scrape current news headlines and use simple NLP libraries (e.g., Stanford NLP) to determine overall sentiment for a company.
- **Advanced prediction model**
    - Use more complex prediction models (e.g., moving averages, exponential smoothing).
- **Multiple stock graphs**
    - Allow users to overlay multiple companies' stock graphs for comparison.
- **Real-time updates**
    - Refresh stock data periodically if connected to a live source.
- **Leaderboards**
    - Track and display the top-performing virtual investors across multiple users.
- **Linear Regression Predictions**

Stretch Features:

- **Machine learning predictions**
    - Train and apply a machine learning model to better predict stock prices.
- **Interactive graph features**
    - Zoom, pan, and hover for more detailed stock price information.

- **News API integration**
    - Pull more structured news data via an API instead of web scraping.
- **Custom portfolio analysis**
    - Provide recommendations based on the user's investment behavior.
- **Mobile support**
    - Port basic functionality to an Android app using Java.
- **Exponential Smoothing**

Class List:

core
Main
- Description: Entry point of the program; starts the controller.

AppController
- Description: Coordinates user input and controls the program flow and data distribution.

FeatureManager (abstract)
- Description: Superclass for all feature managers; handles activation, naming

data
StockDataFetcher
- Description: Loads stock data from a CSV or online source.

StockData
- Description: Stores a list of price points for a specific stock ticker.

PricePoint
- Description: Represents one historical data point with a date and price.

ShortPoint
- Description: Represents one historical data point with a dateTime and price.

prediction
PredictionManager (extends FeatureManager)
- Description: Manages the prediction logic and selects whether to include sentiment.

Prediction
- Description: Core predictor class with SMA, regression, and smoothing algorithms.

PredictionResult (extends StockData)
- Description: Holds prediction output data and metadata like model type and sentiment use.

sentiment
SentimentManager (extends FeatureManager)
- Description: Coordinates news scraping and sentiment analysis.

NewsScraper
- Description: Retrieves news headlines or articles about a stock.

SentimentAnalyzer
- Description: Converts text data into a sentiment score.

graphing
GraphingManager (extends FeatureManager)
- Description: Manages user requests for graphing and integrates prediction overlays.

GraphPlotter
- Description: Renders graphs using lists of price points.

simulation
SimulationManager (extends FeatureManager)
- Description: Oversees simulation execution using prediction + randomness.

Simulator (abstract)
- Description: Abstract class for simulations with shared state and methods.

MarketSimulator (extends Simulator)
- Description: Simulates stock movement based on predictions and randomness.

InvestmentSimulator (extends Simulator)
- Description: Tracks user's portfolio, executes trades, and evaluates performance.

Holding
- Description: Represents one stock purchase with quantity, price, and date.

ui
Screen (interface)
- Description: Interface used by all screens in the program

DrawingSurface
- Description: Manages the drawing of screens

BuyingScreen (implements Screen)
- Description: Screen where the player buys stocks

SellingScreen (implements Screen)
- Description: Screen where the player sells stocks

DayTradingBuyingScreen (implements Screen)
- Description: Buying screen specific to the day trading simulator

DayTradingSellingScreen (implements Screen)
- Description: Selling screen specific to the day trading simulator

GraphingScreen (implements Screen)
- Description: Screen that displays a graph

MainMenuScreen (implements Screen)
- Description: Screen that serves as the main menu, program starts on this screen

SearchScreen (implements Screen)
- Description: Screen where the player can search for stocks to be simulated

InvestmentSearchScreen (implements Screen)
- Description: Search screen used to edit tickers in the stock simulation

DayTradingSearchScreen (implements Screen)
- Description: Search screen used to edit tickers in the day trading simulation

SimulationScreen (implements Screen)
- Description: Screen that displays the stock simulation

SimulationsScreen (implements Screen)
- Description: Screen where the player can enter the stock simulation or the day trading simulation

DayTradingScreen (implements Screen)
- Description: Screen that displays the day trading simulation

InfoScreen (implements Screen)
- Description: Advanced info on program

Credits:
Cameron Cremin:

- BuyingScreen
- SellingScreen
- ReadMe
- Simulation Package
- Presentation Slides
- Bug Fixing

I Huan Chao:

- Core Package
- Data Package
- Graphing Package
- Prediction Package
- UI Package
- UI Improvements
- Presentation Slides
- Bug Fixing
- ReadMe
- UML Diagram
- Jars & JavaDocs
- Project Organization

Credits:

- ChatGPT: JavaDoc Comments, Color picking, and Error Identification
- StackOverflow: API fetching & other questions
- AlphaVantage, MarketStack, and YahooFinance: Stock Data
- freepik.com: Background Image
- Coding Demos: Screen Template