**Stock Market Monitoring Dashboard Documentation**

# Table of Contents

1. [Overview](https://grok.com/chat/6a6bb5bf-07f9-4faf-8ce9-6e2dbf746b0b#overview)
2. [Key Features](https://grok.com/chat/6a6bb5bf-07f9-4faf-8ce9-6e2dbf746b0b#key-features)
3. [Technologies Used](https://grok.com/chat/6a6bb5bf-07f9-4faf-8ce9-6e2dbf746b0b#technologies-used)
4. [Installation and Setup](https://grok.com/chat/6a6bb5bf-07f9-4faf-8ce9-6e2dbf746b0b#installation-and-setup)
   * [Install Python and Required Libraries](https://grok.com/chat/6a6bb5bf-07f9-4faf-8ce9-6e2dbf746b0b#install-python-and-required-libraries)
   * [Running the Application](https://grok.com/chat/6a6bb5bf-07f9-4faf-8ce9-6e2dbf746b0b#running-the-application)
5. [Usage Instructions](https://grok.com/chat/6a6bb5bf-07f9-4faf-8ce9-6e2dbf746b0b#usage-instructions)
   * [Starting the Monitor](https://grok.com/chat/6a6bb5bf-07f9-4faf-8ce9-6e2dbf746b0b#starting-the-monitor)
   * [Viewing Operation History](https://grok.com/chat/6a6bb5bf-07f9-4faf-8ce9-6e2dbf746b0b#viewing-operation-history)
   * [Exporting Data to CSV](https://grok.com/chat/6a6bb5bf-07f9-4faf-8ce9-6e2dbf746b0b#exporting-data-to-csv)
6. [File Structure](https://grok.com/chat/6a6bb5bf-07f9-4faf-8ce9-6e2dbf746b0b#file-structure)
7. [Error Handling](https://grok.com/chat/6a6bb5bf-07f9-4faf-8ce9-6e2dbf746b0b#error-handling)
8. [Export Functionality](https://grok.com/chat/6a6bb5bf-07f9-4faf-8ce9-6e2dbf746b0b#export-functionality)
9. [Troubleshooting](https://grok.com/chat/6a6bb5bf-07f9-4faf-8ce9-6e2dbf746b0b#troubleshooting)
10. [License](https://grok.com/chat/6a6bb5bf-07f9-4faf-8ce9-6e2dbf746b0b#license)
11. [Author](https://grok.com/chat/6a6bb5bf-07f9-4faf-8ce9-6e2dbf746b0b#author)

## Overview

The Stock Market Monitoring Dashboard is a Python-based application designed to monitor real-time stock market data using Yahoo Finance as the data source. Built with Tkinter for the graphical user interface (GUI), SQLite for storing operation history, and Matplotlib for visualizing stock price trends, the application allows users to track stock prices, receive alerts for significant price changes, view historical operations, and export stock data to CSV files. The dashboard provides a user-friendly interface for real-time stock monitoring and data analysis.

## Key Features

* **Real-Time Stock Monitoring**: Fetches and displays intraday stock prices for a specified symbol at user-selected intervals (1, 5, or 15 minutes).
* **Price Alerts**: Notifies users with a pop-up alert when the stock price changes by 5% or more from the previous price.
* **Graphical Visualization**: Displays a real-time plot of stock price trends embedded in the GUI.
* **Operation History**: Logs all operations (data fetches, errors, exports) in an SQLite database, viewable via the GUI.
* **CSV Export**: Allows users to export the current stock data (dates and closing prices) to a CSV file for external analysis.
* **User-Friendly Interface**: A clean Tkinter-based GUI with intuitive controls for monitoring, viewing history, and exporting data.

## Technologies Used

* **Python 3.x**: The programming language used to develop the application.
* **Tkinter**: For building the graphical user interface.
* **yfinance**: For fetching real-time stock data from Yahoo Finance.
* **pandas**: For data manipulation and CSV export.
* **Matplotlib**: For plotting stock price trends with the Tkinter backend (backend\_tkagg).
* **SQLite3**: For storing the history of operations.
* **CSV**: For exporting stock data in a readable format.

## Installation and Setup

### Install Python and Required Libraries

Ensure Python 3.x is installed on your system. You can download it from [python.org](https://www.python.org/). Tkinter and SQLite3 are included in the Python standard library, so no separate installation is required for them.

Install the required external libraries using the provided requirements.txt:

1. Navigate to the project folder in a terminal or command prompt.
2. Run the following command:
3. pip install -r requirements.txt

### The requirements.txt file includes:

yfinance>=0.2.40

pandas>=2.2.2

matplotlib>=3.8.4

### Running the Application

1. Open a terminal or command prompt and navigate to the project folder:
2. cd "C:\LOCAL DISK D\Python\_Intern\_Virtunexa\stock\_monitor"
3. Run the main script to launch the application:
4. python main.py

This will open the GUI of the Stock Market Monitoring Dashboard.

## Usage Instructions

### Starting the Monitor

1. Open the application by running main.py.
2. In the main window, enter a stock symbol (e.g., AAPL for Apple Inc.) in the text box labeled “Enter Stock Symbol.”
3. Select a timeframe (1 minute, 5 minutes, or 15 minutes) from the dropdown menu labeled “Select Update Timeframe.”
4. Click the “Start Monitoring” button to begin fetching real-time stock data.
5. The text widget will display the latest price and timestamp, and a plot of the stock price trend will appear below the text widget, updating at the selected interval.
6. If the price changes by 5% or more from the previous price, a pop-up alert will appear with details of the change.

### Viewing Operation History

1. Click the “View History” button to see a list of all recorded operations.
2. A new window will display the history, including:
   * **Timestamp**: The date and time of the operation.
   * **Symbol**: The stock symbol (e.g., AAPL).
   * **Data**: Details of the operation (e.g., data fetch, export, or error).
3. If no history is available, a message will indicate “No history available.”

### Exporting Data to CSV

1. While monitoring a stock, click the “Export to CSV” button to save the current stock data.
2. The application will generate a CSV file named <symbol>\_data\_<timestamp>.csv (e.g., data/AAPL\_data\_20250520\_225430.csv) in the data/ directory.
3. The text widget will confirm the export (e.g., “Data exported to data/AAPL\_data\_20250520\_225430.csv”).
4. Open the CSV file in a spreadsheet application (e.g., Microsoft Excel, Google Sheets) to analyze the data.

## File Structure

stock\_monitor/

│

├── main.py # Main script for the stock monitoring application

├── requirements.txt # Lists required Python libraries

├── utils/

│ ├── stock\_data.py # Functions for fetching and parsing stock data

│ ├── db.py # Database operations for storing history

├── plot.py # Function for plotting stock data

├── data/

│ ├── stock\_history.db # SQLite database for operation history

│ ├── <symbol>\_data\_<timestamp>.csv # Exported CSV files with stock data

### Example of Stock Data in CSV Format

Date,Close

2025-05-20 13:00:00,150.25

2025-05-20 13:01:00,150.50

2025-05-20 13:02:00,151.00

## Error Handling

* **Invalid Symbol**: If an invalid stock symbol is entered (e.g., XYZ), the text widget displays an error (e.g., “Error: No data returned for symbol XYZ. Possibly delisted, invalid symbol, or market closed.”).
* **No Data Available**: If no data is fetched (e.g., due to market closure), an error message is shown in the text widget and logged to the database.
* **Empty Data for Export**: If the “Export to CSV” button is clicked when no data is available, the text widget shows “Error: No data available to export.”
* **Database Issues**: If there’s a problem with the SQLite database (e.g., connection errors), an error message appears in a pop-up, and the issue is logged.
* **Invalid Timeframe**: The application restricts timeframe selection to valid options (1min, 5min, 15min) via a dropdown, preventing invalid inputs.

## Export Functionality

When the user clicks the “Export to CSV” button, the application generates a CSV file containing the current stock data. The file includes:

* **Date**: The timestamp of each data point (e.g., 2025-05-20 13:00:00).
* **Close**: The closing price for that timestamp (e.g., 150.25).  
  The CSV file is saved in the data/ directory with a unique filename based on the stock symbol and current timestamp (e.g., data/AAPL\_data\_20250520\_225430.csv). The export operation is logged in the SQLite database, and a confirmation message appears in the text widget.

## Troubleshooting

* **Application Doesn’t Start**:
  1. Ensure Python 3.x is installed (python --version).
  2. Verify that dependencies are installed:
  3. pip install -r requirements.txt
  4. Check that the data/ directory exists:
  5. mkdir "C:\LOCAL DISK D\Python\_Intern\_Virtunexa\stock\_monitor\data"
  6. If using a virtual environment, ensure it’s activated.
* **No Data Fetched**:
  1. Confirm the stock symbol is valid (e.g., AAPL, GOOGL). Avoid prefixes like $.
  2. Ensure the US market is open (9:30 AM–4:00 PM EST). As of 10:59 PM IST (1:29 PM EST, May 20, 2025), the market is open.
  3. Test yfinance independently:
  4. import yfinance as yf
  5. stock = yf.Ticker("AAPL")
  6. print(stock.history(period="1d", interval="1m"))

If empty, try period="7d" or interval="5m".

* **Plot Not Displaying**:
  1. Ensure matplotlib is installed and using the TkAgg backend:
  2. import matplotlib
  3. print(matplotlib.get\_backend())

If not TkAgg, add matplotlib.use('TkAgg') at the top of main.py.

* 1. Verify the plot appears below the text widget in the Tkinter window.
  2. Check the text widget for data updates. If no data is shown, the issue is with data fetching.
* **CSV Export Fails**:
  1. Ensure the data/ directory exists in the project root.
  2. Check the text widget for errors like “Error exporting to CSV: ...”.
  3. Verify data is available by confirming price updates in the text widget.
* **Application Crashes**:
  1. Run the program from a terminal to capture error logs:
  2. python main.py
  3. Check for specific error messages and report them for further assistance.

## License

This project is open-source and free to use. You are welcome to modify the code for educational purposes or personal projects. If redistributing, please provide appropriate credit to the author.

## Author

Sura Harsh