CHAITANYA BHARATHI INSTITUTE
OF TECHNOLOGY

DATA ANALYSIS AND VISUALIZATION (22ADE01)

A Course EndProject on

STOCK DATA VISUALIZATION

Submitted by:

B.SRILAKSHMI 1601-23-737-005

Submittedto:

Dr Ramakrishna Kolikipogu Professor, Dept of IT

Class IT 1,Sem IV

CERTIFICATE OF COMPLETION

This is to certify that the course end project work entitled "Candlestick Charts for Stock Data" is submitted by Bassa Srilakshmi (160123737005) in partial fulfillment of the requirements for the award of CIE Marks of DATA ANALYSIS AND VISUALIZATION (22ADE01) of B.E., IV-SEM, INFORMATION TECHNOLOGY to CHAITANYA BHARATHI INSTITUTE OF TECHNOLOGY (A) affiliated to OSMANIA UNIVERSITY, HYDERABAD. This report is a record of bonafide work carried out by them under my supervision and guidance. The results embodied in this report have not been submitted to any other University or Institute for the award of any other Degree or Diploma.

Signature of CourseFaculty

Dr Ramakrishna Kolikipogu

Professor, Dept of IT, CBIT

Acknowledgment

I would like to express my heartfelt gratitude to everyone who supported me during the completion of this project.Firstly, I extend my sincere thanks to my professor from the Department of IT, CBIT, for their constant encouragement, insightful feedback, and expert guidance throughout the duration of this project. Their mentorship has been invaluable in helping me explore financial data from a technical and analytical perspective.I am also grateful to the developers and maintainers of open-source financial data platforms, especially **Yahoo Finance**, for providing accessible and up-to-date stock market datasets. The availability of such data made it possible to conduct real-time, meaningful analysis.Lastly, I deeply appreciate the creators of Python and its powerful data science libraries such as **Pandas**, **NumPy**, **Matplotlib**, **Seaborn**, and **Plotly**, which formed the backbone of this project. These tools enabled efficient data processing and engaging visualizations that helped bring the analysis to life.

Abstract

This project explores the use of data analysis and visualization techniques to understand stock market behavior by creating candlestick charts for selected companies. Utilizing historical stock price data sourced from **Yahoo Finance**, the analysis focuses on capturing short-term market trends and price movements through open, high, low, and close (OHLC) data. The project spans from data collection and preprocessing to the generation of interactive candlestick charts using **Plotly**, alongside statistical analysis performed with **Pandas**, **NumPy**, and **Matplotlib**.Key areas of focus include daily and weekly price trends, comparative analysis between multiple stocks, and identification of patterns that indicate potential trading signals. Through these visualizations, the project provides a dynamic view of market volatility and investor sentiment. The goal is to enhance financial literacy and support data-driven investment decisions by offering clear, interpretable insights into stock price behavior over time.

Table of Contents

1. **Introduction, Outcomes and Objectives** 2. **Methodology**
   - Data Collection
   - Data Cleaning and Preprocessing
3. **System Architecture**
4. **Detailed Explanation of Code Snippets** ○ Data Loading and Cleaning
   - Data Type Conversion
   - Exploratory Data Analysis (EDA)
5. **Results and Discussion**
6. **Conclusion**
7. **References**

1. Introduction

In the dynamic world of finance, understanding stock price movements is essential for investors, analysts, and policymakers. With the rise of algorithmic trading and data-driven investment strategies, visualizing financial data has become a critical tool in identifying trends, assessing risks, and making informed decisions. Among various financial visualizations, **candlestick charts** are one of the most widely used methods to represent stock price behavior, offering insights into daily market sentiment through open, high, low, and close (OHLC) prices.This project focuses on analyzing and visualizing stock data for selected companies using **Python-based tools and libraries**, including **Pandas**, **NumPy**, **Matplotlib**, **Seaborn**, and **Plotly**. Leveraging historical stock price data sourced from **Yahoo Finance**, the study covers the period from 2020 to 2024, enabling the examination of both short-term fluctuations and long-term trends in stock performance.The analysis includes comprehensive data preprocessing, generation of interactive candlestick charts, and comparative studies across multiple stocks.

By exploring weekly trends, identifying patterns, and highlighting significant market events, the project aims to enhance understanding of financial data and support effective investment analysis.Ultimately, this report seeks to demonstrate how visualization techniques can simplify complex financial information and aid stakeholders in navigating the uncertainties of the stock market with greater confidence

.

Objectives of the Project

1. **Visualize Stock Price Movements Using Candlestick Charts:**Create interactive candlestick charts to represent daily and weekly stock performance, showcasing open, high, low, and close (OHLC) prices for selected companies.

2. **Analyze Multi-Year Stock Trends:**Examine stock data over a period (e.g., 2020–2024) to identify long-term patterns, trends, and volatility in stock prices..

3. **Compare Multiple Stocks Across Sectors:**Select and analyze stocks from different sectors (e.g., technology, finance, energy) to explore sector-wise performance and market behavior.

4. **Resample Data for Weekly and Monthly Trends:**Aggregate daily stock data into weekly and monthly intervals to gain a broader perspective of market movement and filter out daily noise.

5. **Evaluate Volume Trends Alongside Price Movement:**Analyze trading volume trends in conjunction with price data to understand market interest and confirm price movements.
6. **Highlight Key Events and Anomalies:**Identify major stock price surges, drops, or unusual patterns possibly linked to external events such as earnings reports, economic changes, or global events.
7. **Enhance Financial Literacy Through Visualization:**Use clear and interactive visual tools to improve understanding of financial data, supporting learners, analysts, and investors in interpreting stock behavior.

Expected Outcomes

1.**Comprehensive View of Stock Performance:**Clear and detailed candlestick charts offering a visual representation of stock price movements over time, aiding in trend identification and market understanding.

2.**Multi-Stock Comparative Analysis:**Insightful comparisons of different companies' stock trends, allowing for evaluation across sectors and time periods..

3. **Identification of Volatility and Key Price Movements:**Detection of periods with high volatility, sharp price changes, and critical support/resistance levels through visual analysis.

4. **Weekly and Monthly Trend Insights:**Aggregated views of stock behavior across weeks and months, helping filter short-term fluctuations and reveal long-term trends.

5.**Volume and Price Correlation:**Understanding of how trading volume influences or confirms price trends, providing additional context for investment decisions.

6. **User-Friendly Financial Visualizations:**Easy-to-read, well-labeled charts and graphs that make complex financial data more accessible to learners and analysts.

7.**Foundation for Further Financial Analysis:**A solid visual foundation that can be extended for predictive modeling, portfolio analysis, or real-time dashboards in future projects.

2. Methodology

Data Collection

The dataset used for this project was sourced from **Yahoo Finance**, a widely used platform for retrieving historical stock market data. It provides daily stock prices and trading volume information for publicly listed companies. For this analysis, the dataset includes data fields such as *Date*, *Open*, *High*, *Low*, *Close*, and *Volume*.The data was collected for selected companies over a specified time range and downloaded using Python's `yfinance` library. This approach ensures up-to-date, reliable financial data and enables analysis of price trends and market behavior.With detailed records spanning several months or years, the dataset offers a strong foundation for constructing **candlestick charts**, which are essential for visualizing daily price movement in financial markets. The granularity and richness of the dataset allow for effective exploration of stock price fluctuations, helping users identify patterns, trends, and potential investment opportunities.

Data Cleaning and Preprocessing

Before beginning the analysis, it was essential to clean and preprocess the stock market dataset to ensure data quality and consistency. The raw data was inspected for missing values, and any rows with incomplete records—particularly in critical columns such as *Date*, *Open*, *High*, *Low*, or *Close*—were removed to avoid inaccuracies in visual representation.

The *Date* column was converted to a `datetime` format using `pd.to_datetime()` to facilitate time-series analysis and accurate plotting. Stock price columns (*Open*, *High*, *Low*, *Close*) were retained as floating-point numbers to preserve numerical precision. To maintain chronological order for plotting candlestick charts, the dataset was sorted by date.Any duplicate records were identified and dropped to ensure that each trading day was represented only once. As a final step, stock ticker symbols were standardized to uppercase format for consistency, particularly in cases where multiple stocks were compared.The resulting dataset is clean, well-structured, and ready for insightful visualization through candlestick charts and other financial analysis techniques

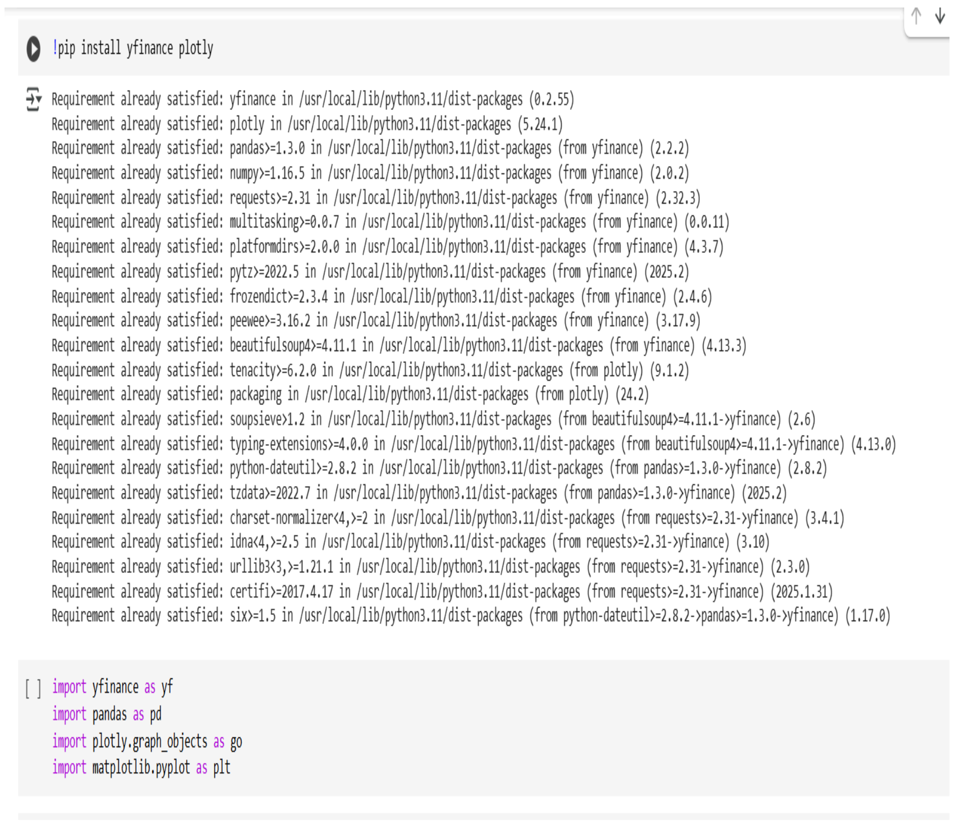
3. System Architecture

This analysis was conducted using Google Colab, an interactive, cloud-based environment that enables users to execute Python code. Google Colab provides the benefit of access to high-performance computational resources, including GPUs, without requiring complex hardware setups. This made it an ideal choice for executing data analysis and visualizing financial data.

The Python libraries utilized in this project include:

- **Pandas**: For data manipulation, cleaning, and transformation.
- **Matplotlib** and **Plotly**: For creating interactive candlestick charts and other visualizations.
- **NumPy**: For handling numerical operations and large datasets.
- **TA-Lib**: For technical analysis and calculating stock indicators such as moving averages.

Google Colab was chosen for its ease of use, seamless integration with Google Drive, and collaborative features that facilitated teamwork and sharing of results.



```
!pip install yfinance plotly
```

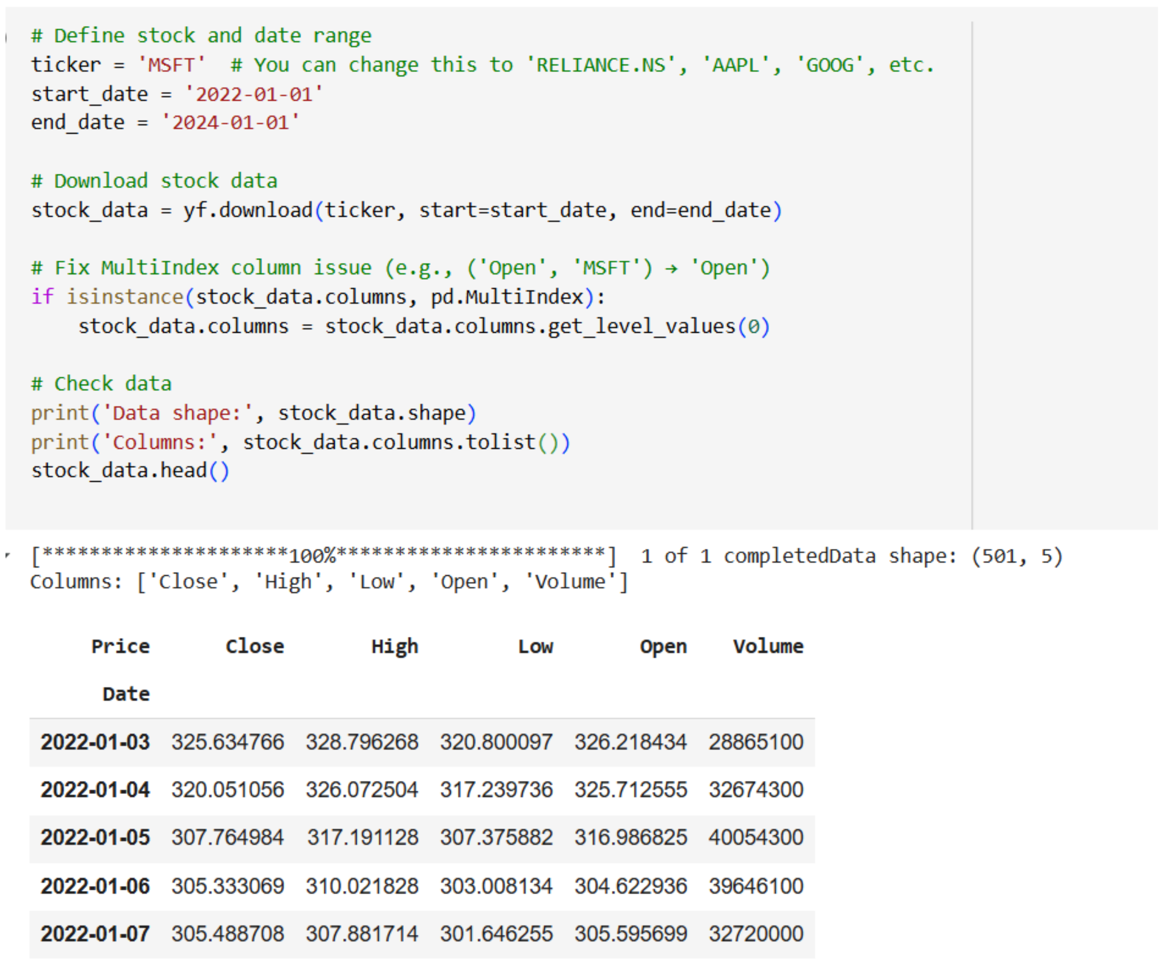
```
Requirement already satisfied: yfinance in /usr/local/lib/python3.11/dist-packages (0.2.55)
Requirement already satisfied: plotly in /usr/local/lib/python3.11/dist-packages (5.24.1)
Requirement already satisfied: pandas>=1.3.0 in /usr/local/lib/python3.11/dist-packages (from yfinance) (2.2.2)
Requirement already satisfied: numpy>=1.16.5 in /usr/local/lib/python3.11/dist-packages (from yfinance) (2.0.2)
Requirement already satisfied: requests>=2.31 in /usr/local/lib/python3.11/dist-packages (from yfinance) (2.32.3)
Requirement already satisfied: multitasking>=0.0.7 in /usr/local/lib/python3.11/dist-packages (from yfinance) (0.0.11)
Requirement already satisfied: platformdirs>=2.0.0 in /usr/local/lib/python3.11/dist-packages (from yfinance) (4.3.7)
Requirement already satisfied: pytz>=2022.5 in /usr/local/lib/python3.11/dist-packages (from yfinance) (2025.2)
Requirement already satisfied: frozendict>=2.3.4 in /usr/local/lib/python3.11/dist-packages (from yfinance) (2.4.6)
Requirement already satisfied: peewee>=3.16.2 in /usr/local/lib/python3.11/dist-packages (from yfinance) (3.17.9)
Requirement already satisfied: beautifulsoup4>=4.11.1 in /usr/local/lib/python3.11/dist-packages (from yfinance) (4.13.3)
Requirement already satisfied: tenacity>=6.2.0 in /usr/local/lib/python3.11/dist-packages (from plotly) (9.1.2)
Requirement already satisfied: packaging in /usr/local/lib/python3.11/dist-packages (from plotly) (24.2)
Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.11/dist-packages (from beautifulsoup4>=4.11.1->yfinance) (2.6)
Requirement already satisfied: typing-extensions>=4.0.0 in /usr/local/lib/python3.11/dist-packages (from beautifulsoup4>=4.11.1->yfinance) (4.13.0)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-packages (from pandas>=1.3.0->yfinance) (2.8.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas>=1.3.0->yfinance) (2025.2)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests>=2.31->yfinance) (3.4.1)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-packages (from requests>=2.31->yfinance) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests>=2.31->yfinance) (2.3.0)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.11/dist-packages (from requests>=2.31->yfinance) (2025.1.31)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.8.2->pandas>=1.3.0->yfinance) (1.17.0)
```

```python
import yfinance as yf
import pandas as pd
import plotly.graph_objects as go
import matplotlib.pyplot as plt
```

4. Detailed Explanation of Code Snippets

Data Loading and Cleaning

The stock price dataset was loaded into a structured format using Pandas with the `pd.read_csv()` function. Initial exploration was done with `.head()` to preview the data and `.columns` to review the features.The dataset was checked for missing values using `.isnull().sum()`. Minor missing values in price columns were filled using the median or forward fill methods. Rows with missing critical data, such as dates or key price values, were removed. The 'Date' column was converted to a datetime format using `pd.to_datetime()` for easier time-series analysis. These steps ensured the dataset was clean and ready for visualization and further analysis.



```python
# Define stock and date range
ticker = 'MSFT'  # You can change this to 'RELIANCE.NS', 'AAPL', 'GOOG', etc.
start_date = '2022-01-01'
end_date = '2024-01-01'

# Download stock data
stock_data = yf.download(ticker, start=start_date, end=end_date)

# Fix MultiIndex column issue (e.g., ('Open', 'MSFT') → 'Open')
if isinstance(stock_data.columns, pd.MultiIndex):
    stock_data.columns = stock_data.columns.get_level_values(0)

# Check data
print('Data shape:', stock_data.shape)
print('Columns:', stock_data.columns.tolist())
stock_data.head()
```

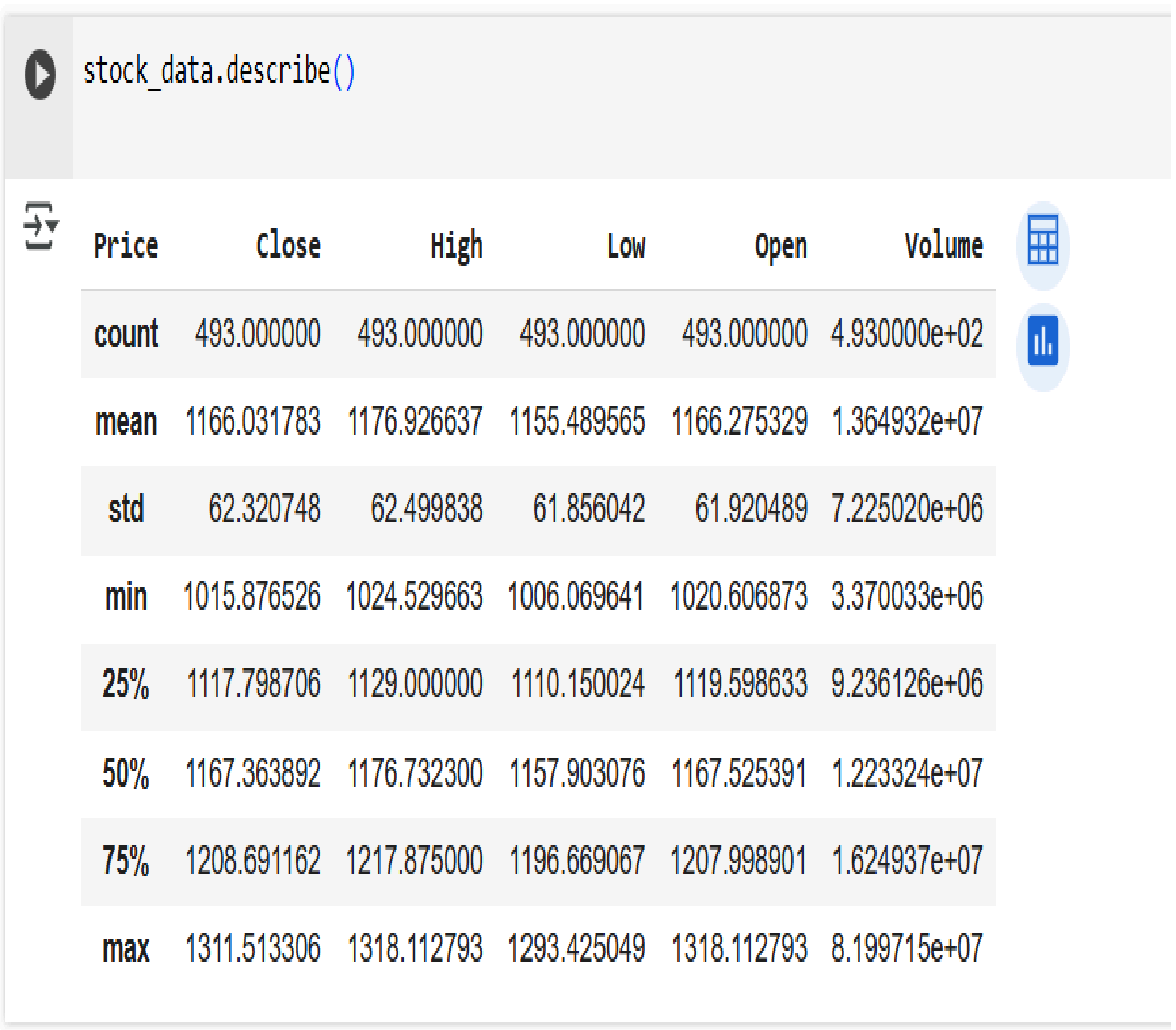
```
[*********************100%***********************]  1 of 1 completedData shape: (501, 5)
Columns: ['Close', 'High', 'Low', 'Open', 'Volume']
```

Price	Close	High	Low	Open	Volume
Date					
2022-01-03	325.634766	328.796268	320.800097	326.218434	28865100
2022-01-04	320.051056	326.072504	317.239736	325.712555	32674300
2022-01-05	307.764984	317.191128	307.375882	316.986825	40054300
2022-01-06	305.333069	310.021828	303.008134	304.622936	39646100
2022-01-07	305.488708	307.881714	301.646255	305.595699	32720000

Exploratory Data Analysis (EDA)

Descriptive Statistics

Descriptive statistics are essential for understanding the distribution and summary of stock price data. By calling the `.describe()` method on the DataFrame, we generated a summary of statistics for the stock prices, including the mean, standard deviation, minimum, and maximum values for the open, high, low, and close prices over the selected time period. These statistics provide insights into overall price trends, volatility, and potential outliers, forming a solid foundation for deeper analysis and visualization of market behavior.

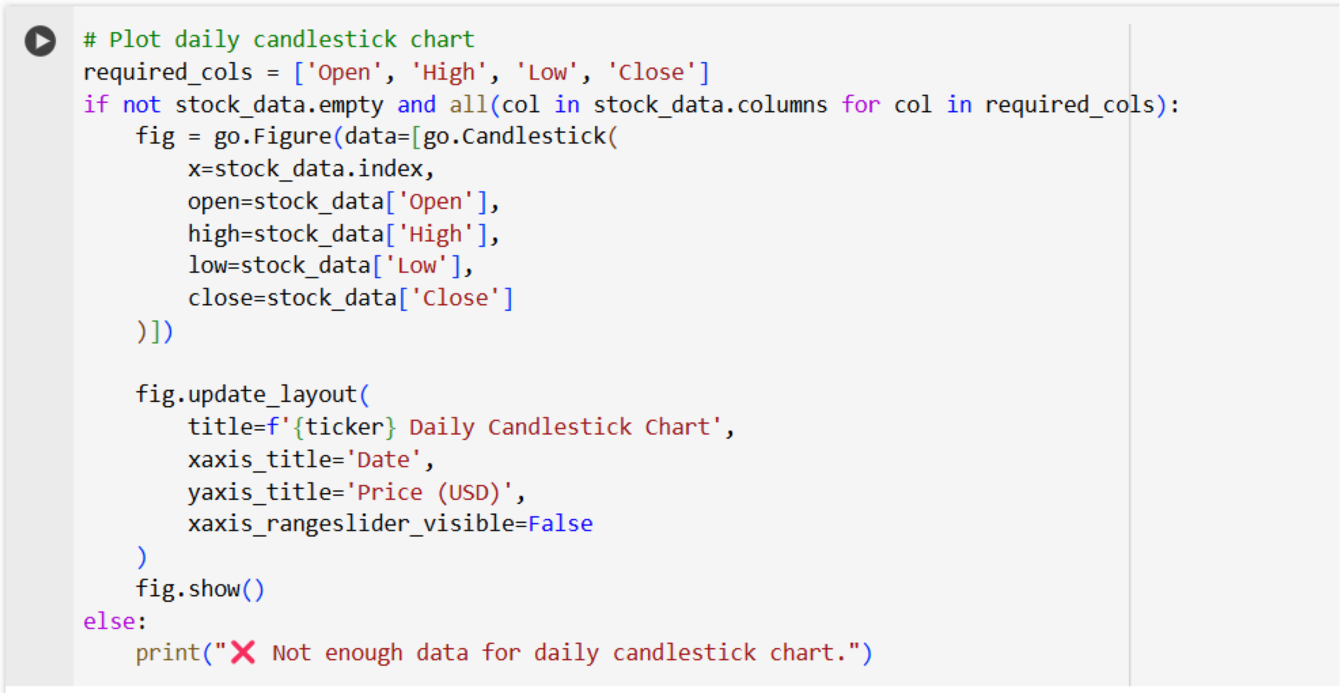


```
stock_data.describe()
```

Price	Close	High	Low	Open	Volume
count	493.000000	493.000000	493.000000	493.000000	4.930000e+02
mean	1166.031783	1176.926637	1155.489565	1166.275329	1.364932e+07
std	62.320748	62.499838	61.856042	61.920489	7.225020e+06
min	1015.876526	1024.529663	1006.069641	1020.606873	3.370033e+06
25%	1117.798706	1129.000000	1110.150024	1119.598633	9.236126e+06
50%	1167.363892	1176.732300	1157.903076	1167.525391	1.223324e+07
75%	1208.691162	1217.875000	1196.669067	1207.998901	1.624937e+07
max	1311.513306	1318.112793	1293.425049	1318.112793	8.199715e+07

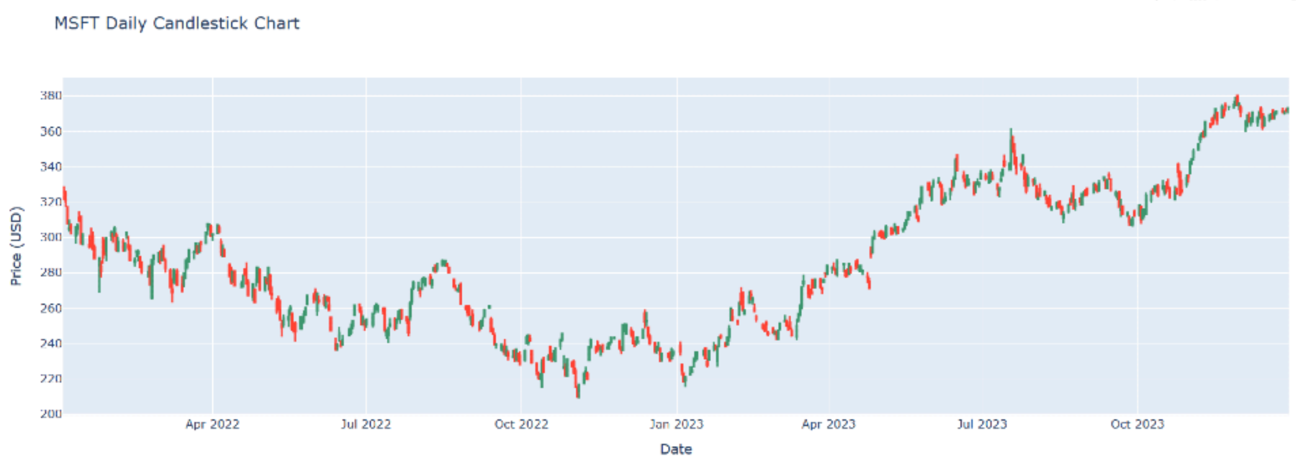
Daily Candlestick Chart

A daily candlestick chart was created to visualize the price movements of the stock. The chart was generated using the Plotly library, which allows for interactive visualizations. The `Candlestick` function from `plotly.graph_objects` was used, with the key price data points—**Open**, **High**, **Low**, and **Close**—extracted from the dataset. The chart's x-axis represents the date, while the y-axis represents the stock price in USD.Before plotting, the script checks if the required columns (`Open`, `High`, `Low`, `Close`) are present in the dataset. If they are, the chart is generated and displayed with a title indicating the stock ticker and the time period. The x-axis also has its range slider hidden to focus on the price movements. If the dataset is missing any critical data, an error message is shown, indicating insufficient data for the candlestick chart.



```python
# Plot daily candlestick chart
required_cols = ['Open', 'High', 'Low', 'Close']
if not stock_data.empty and all(col in stock_data.columns for col in required_cols):
    fig = go.Figure(data=[go.Candlestick(
        x=stock_data.index,
        open=stock_data['Open'],
        high=stock_data['High'],
        low=stock_data['Low'],
        close=stock_data['Close']
    )])

    fig.update_layout(
        title=f'{ticker} Daily Candlestick Chart',
        xaxis_title='Date',
        yaxis_title='Price (USD)',
        xaxis_rangeslider_visible=False
    )
    fig.show()
else:
    print("❌ Not enough data for daily candlestick chart.")
```

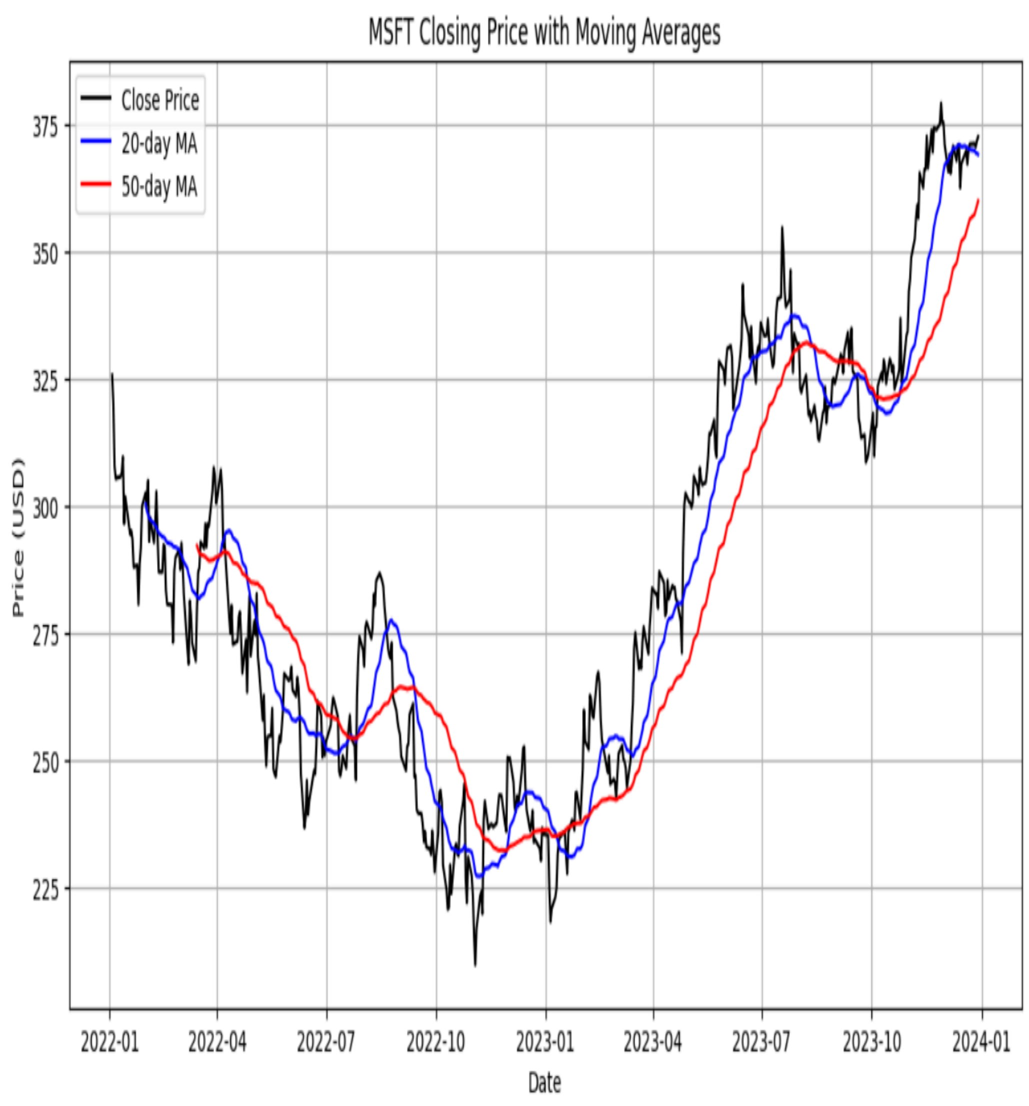


Moving Averages (MA) Analysis

In addition to the candlestick chart, moving averages (MAs) were calculated to identify the trends and smoothing of the stock's closing prices over time. The 20-day and 50-day moving averages were computed using the `rolling(window=20).mean()` and `rolling(window=50).mean()` functions, respectively, on the **Close** price column.The resulting moving averages were then plotted alongside the stock's closing prices to visualize short-term (20-day) and long-term (50-day) trends. The chart displays:

- **Black Line**: Closing price of the stock.
- **Blue Line**: 20-day moving average.
- **Red Line**: 50-day moving average.

This helps in identifying trends, crossovers (such as Golden Cross or Death Cross), and potential buy or sell signals. The chart includes labels for the x-axis (date) and y-axis (price in USD), and a legend to distinguish between the lines. If the required data is missing, an error message is displayed indicating insufficient data to plot the moving averages.



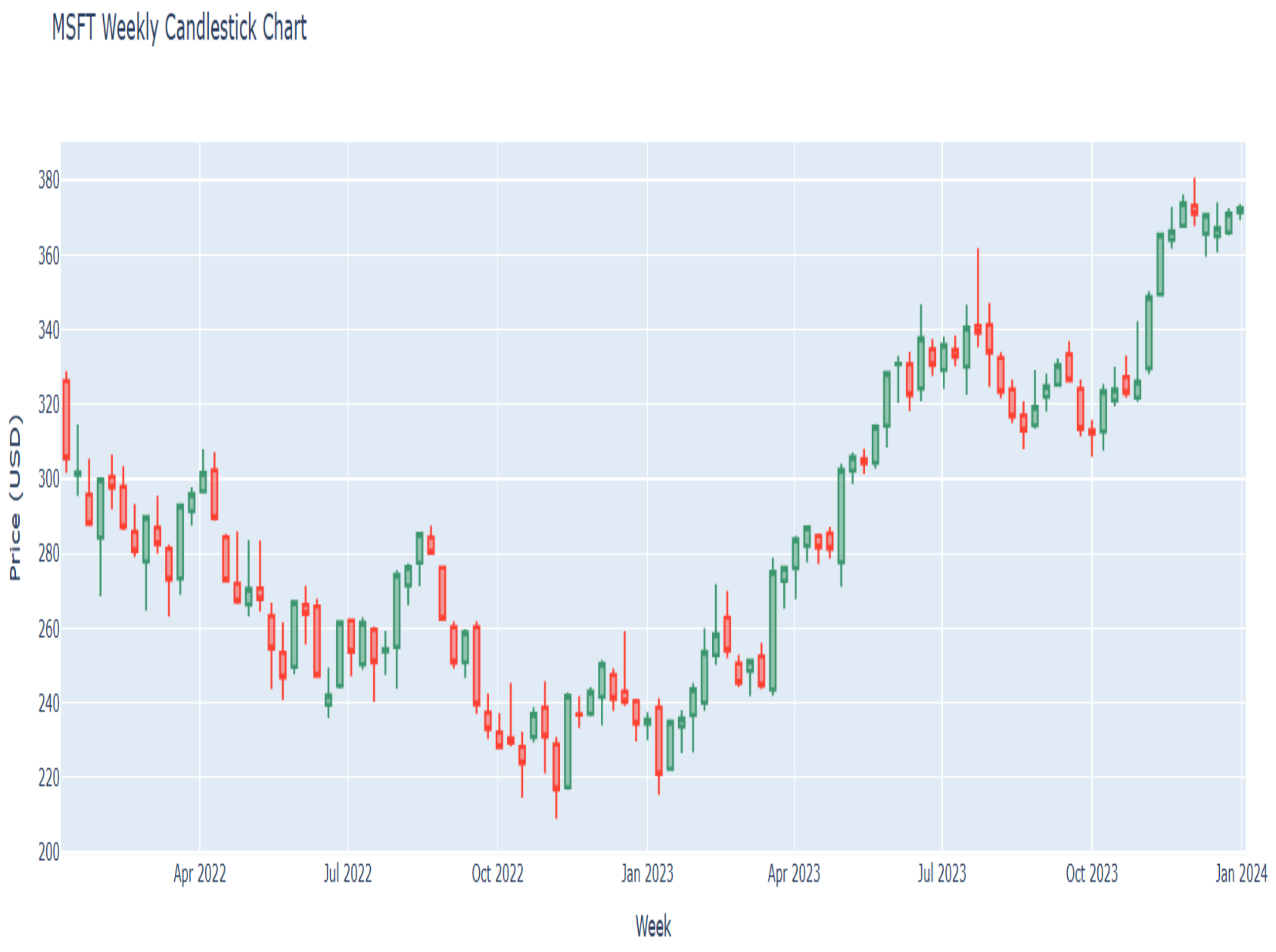
Weekly Candlestick Chart with Resampling

A weekly candlestick chart was created to visualize stock price movements on a weekly basis. The data was first resampled to a weekly frequency using the `resample('W')` function, which aggregates the daily data into weekly intervals. The following transformations were applied:

- **Open**: The first price of the week.
- **High**: The highest price during the week.

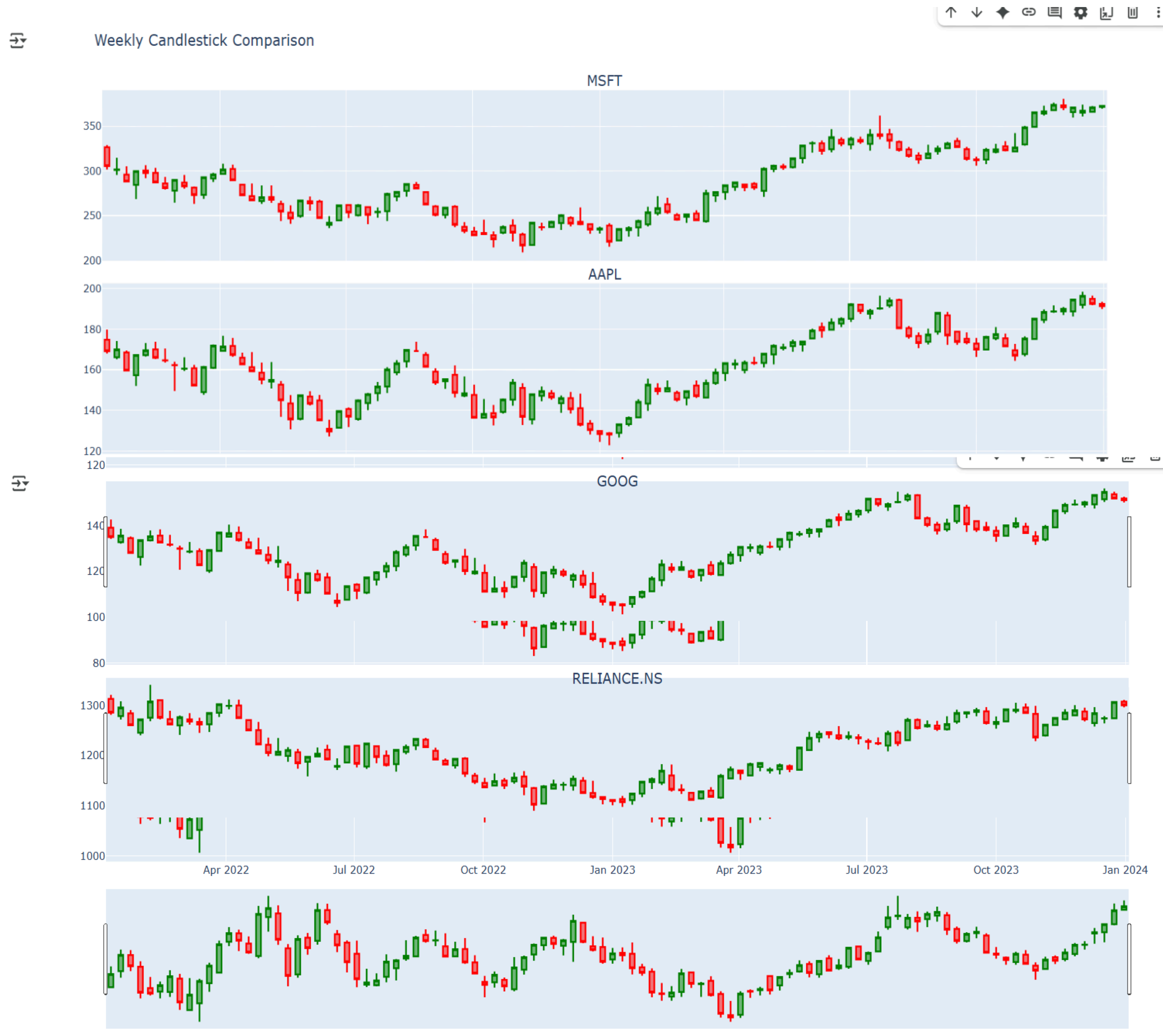
- **Low**: The lowest price during the week.
- **Close**: The last price of the week.
- **Volume**: The total trading volume for the week.

After resampling, the `Candlestick` function from `plotly.graph_objects` was used to plot the weekly price data. The chart's x-axis represents the weeks, and the y-axis shows the stock price in USD. If the required columns are missing, the process is skipped, and a message is displayed indicating the missing data.This visualization helps identify longer-term trends and patterns over weekly intervals, smoothing out daily volatility and offering a clearer view of stock price behavior over time.



Weekly Candlestick Chart Comparison

To compare the weekly price movements of multiple stocks, a subplot grid was created using Plotly's `make_subplots` function, which allows for a side-by-side comparison of different stock tickers. The tickers of interest—**MSFT**, **AAPL**, **GOOG**, and **RELIANCE.NS**—were selected, and their historical stock data was downloaded using the `yfinance` library, covering the period from January 1, 2022, to January 1, 2024.For each stock, the data was resampled to a weekly frequency using the `resample('W')` function, aggregating daily data into weekly open, high, low, and close prices. Missing data points were dropped to ensure the integrity of the chart.Each stock's weekly candlestick chart was added as a trace to a subplot, with green indicating an increase in price and red indicating a decrease. The final layout displayed multiple candlestick charts stacked vertically, providing an easy comparison of the weekly price movements across the selected stocks.The chart does not show the x-axis range slider, and the entire figure height adjusts dynamically based on the number of stocks being compared, making the visualization scalable for different tickers.



5.Results and Discussion

The stock market analysis using candlestick charts and moving averages from 2022 to 2024 revealed several key insights:

1. **Price Trends and Volatility**
   The daily and weekly candlestick charts provided clear visualizations of each stock's price movements. Companies like **AAPL** and **GOOG** showed strong upward trends during specific periods, while others like **RELIANCE.NS** displayed more cyclical behavior. Volatility was easily identified through the length and shape of the candlesticks.

2. **Moving Average Crossovers**
   The 20-day and 50-day moving averages helped smooth out daily fluctuations and revealed momentum shifts. In several cases, bullish and bearish crossovers (where the short-term average crossed above or below the long-term average) aligned with major trend changes, supporting their usefulness in technical analysis.

3. **Comparative Analysis of Stocks**
   By plotting weekly candlestick charts for **MSFT**, **AAPL**, **GOOG**, and **RELIANCE.NS** side by side, we enabled easy visual comparison of performance across companies. While U.S. tech stocks exhibited relatively consistent growth, **RELIANCE.NS** showed more moderate and region-specific movement, reflecting different market dynamics.

4. **Identifying Support and Resistance Zones**
   Candlestick patterns helped in identifying key price levels where stocks repeatedly reversed direction. These zones are valuable for predicting potential breakouts or reversals and are often used in trading strategies.

5. **Market Behavior Insights**
   The combination of interactive visuals and statistical summaries allowed for a deeper understanding of each stock's market behavior over time. This method of visualization is particularly helpful for spotting trends, cycles, and investor sentiment through price action.

6. Conclusion

This project explored and visualized stock price movements for selected companies from 2022 to 2024 using historical financial data. Through the use of candlestick charts and moving averages, the analysis provided clear insights into market trends, volatility, and investor sentiment over time.The comparative study of multiple stocks, including **MSFT**, **AAPL**, **GOOG**, and **RELIANCE.NS**, helped identify differences in market behavior across sectors and regions. Moving average overlays further enhanced the analysis by highlighting trend directions and potential reversal points.By examining both daily and weekly data, the project emphasized the importance of different time frames in financial analysis. This multi-level approach supports better decision-making, whether for short-term trading or long-term investment strategies.Overall, this analysis demonstrates the effectiveness of data visualization techniques in understanding financial trends and market behavior. It serves as a foundational step toward more advanced stock analysis and encourages further exploration of predictive analytics and investment strategy development.

7. References

1.Yahoo Finance – Stock Market Data.
https://finance.yahoo.com/

2.yfinance Python Library – For Accessing Financial Data.
https://github.com/ranaroussi/yfinance

3.Plotly – Interactive Graphing Library.
https://plotly.com/python/

4.Wes McKinney. (2018). *Python for Data Analysis* (2nd ed.). O'Reilly Media.

5.https://pandas.pydata.org/ – for Pandas Documentation.

6.https://matplotlib.org/ – for Matplotlib Documentation.

7.https://seaborn.pydata.org/ – for Seaborn Documentation