

DATA - 200
Stock Analysis
Lab 2

Shibin Biji Thomas

018318261

Git Hub Link: https://github.com/Shibin506/Shibin_PYTHON_LAB-2

1. Objective

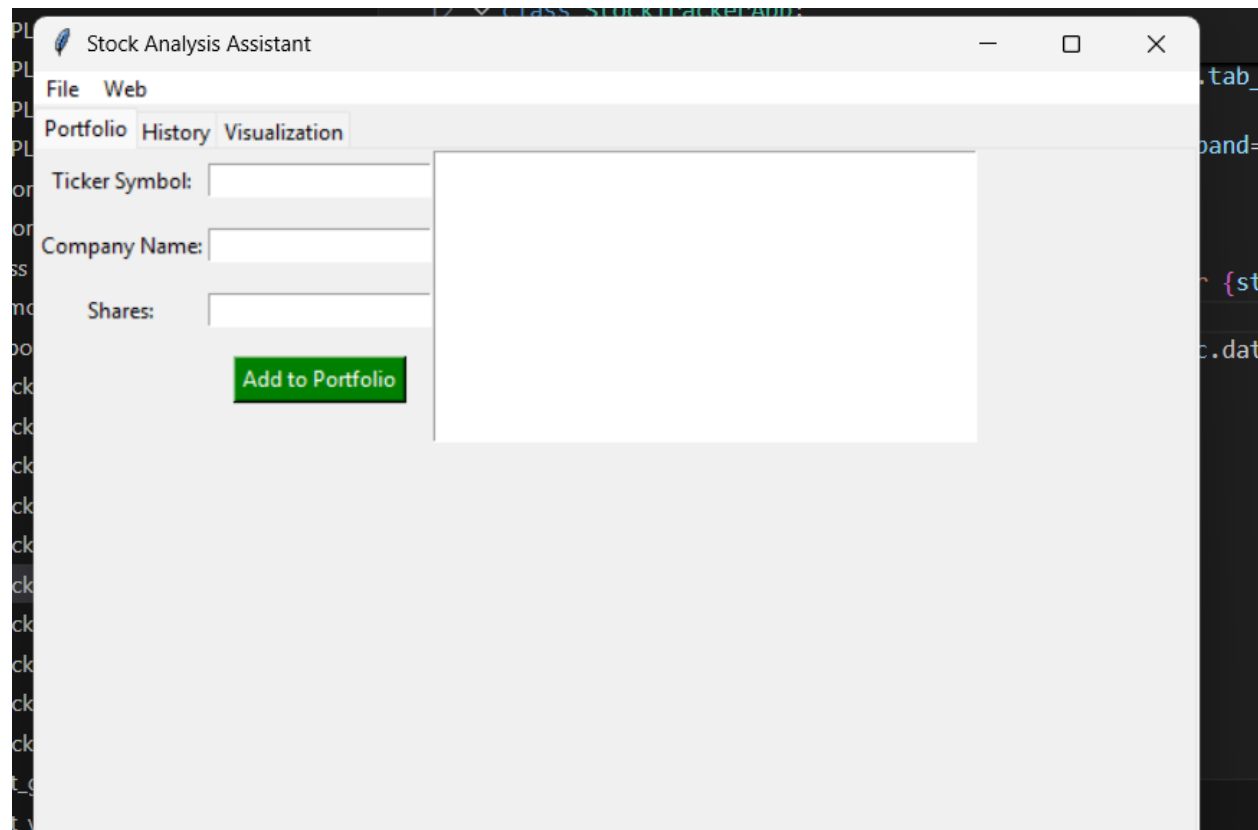
The goal of Lab 2 is to extend our Python-based stock tracking application by incorporating external data sources. The lab introduces web scraping techniques using BeautifulSoup and Selenium, as well as CSV file imports to populate historical stock data into the application. This lab enhances both the GUI and console components of the app for real-time and bulk data ingestion.

2. Tools & Libraries Used

- Python (v3.x)
 - Tkinter – GUI interface
 - BeautifulSoup (bs4) – HTML parsing for scraping
 - Selenium – Browser automation
 - CSV – File reading/writing
 - Matplotlib – Chart visualization
 - SQLite – Embedded database (via stock_db.py)
-

GUI workflow

Stock Analysis Assistant



The screenshot shows a web application window titled "Stock Analysis Assistant". The window has a menu bar with "File" and "Web". Below the menu bar are three tabs: "Portfolio" (selected), "History", and "Visualization". The main content area is divided into two sections. On the left, there are three input fields: "Ticker Symbol:", "Company Name:", and "Shares:". Below these fields is a green button labeled "Add to Portfolio". On the right, there is a large, empty white rectangular area, likely intended for displaying stock data or charts. The background of the window is light gray.

Stock Analysis Assistant

File Web

Portfolio History Visualization

Ticker Symbol:

Company Name:

Shares:

Add to Portfolio

Manually Adding Data

Form

board

File Web

Portfolio History Visualization

Ticker Symbol: AAPL

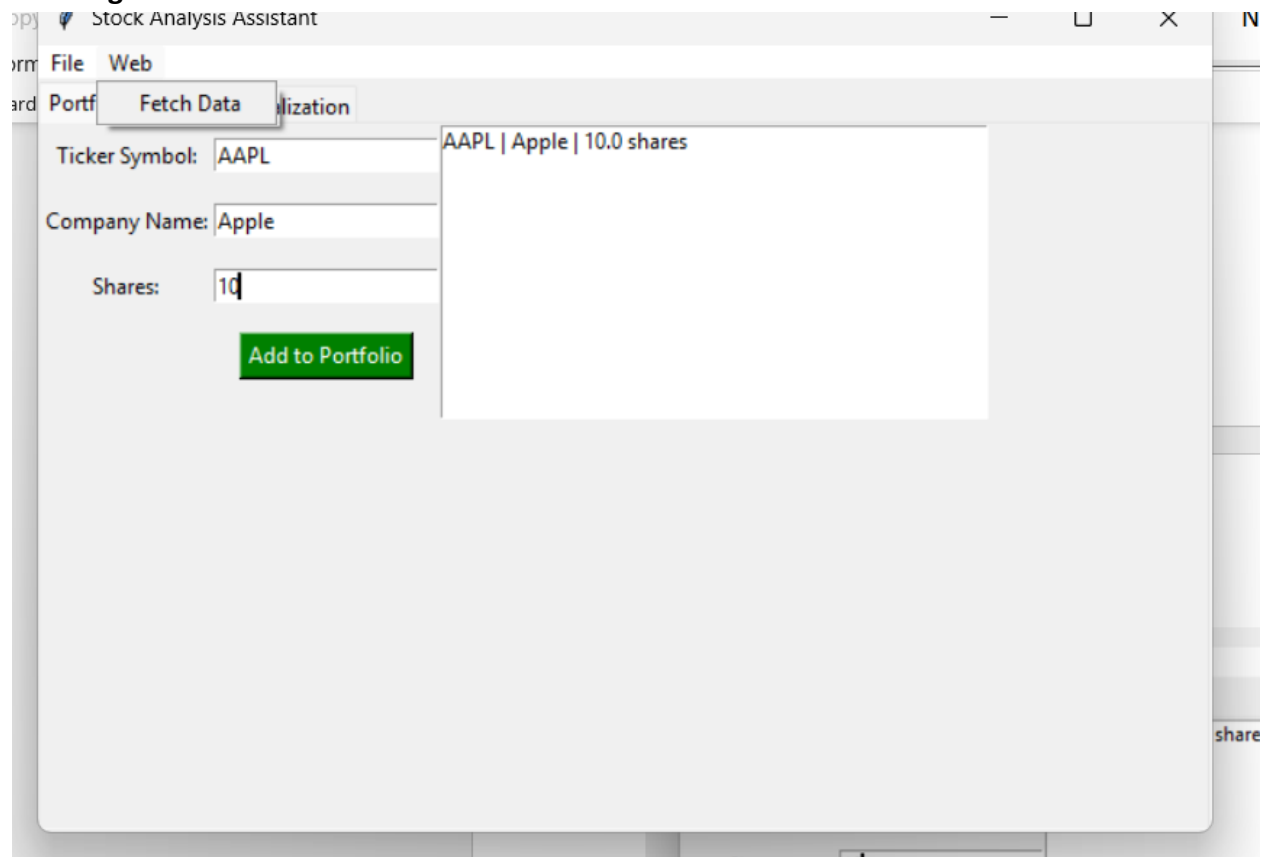
Company Name: Apple

Shares: 10

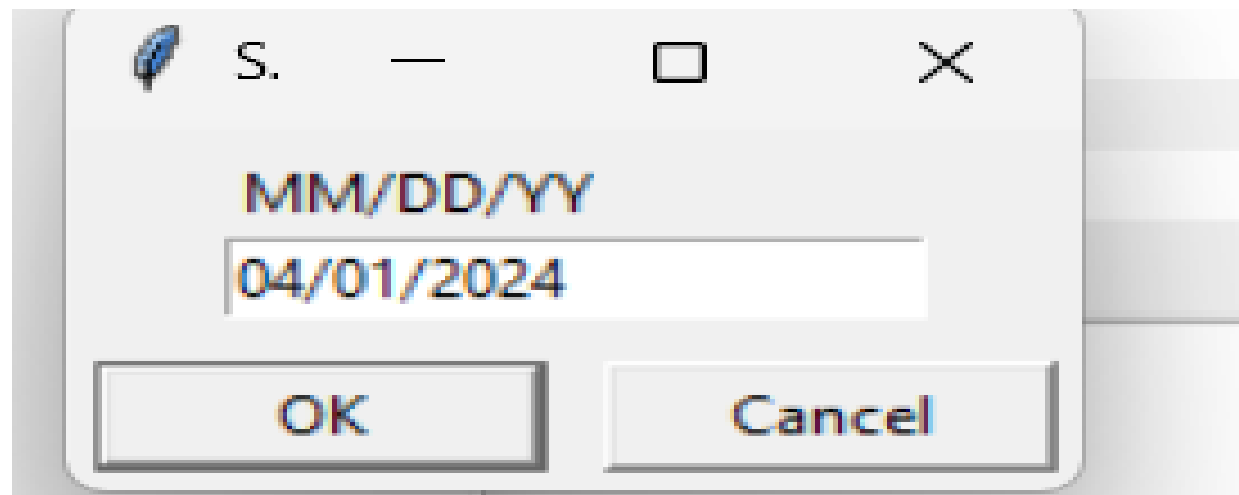
Add to Portfolio

AAPL | Apple | 10.0 shares

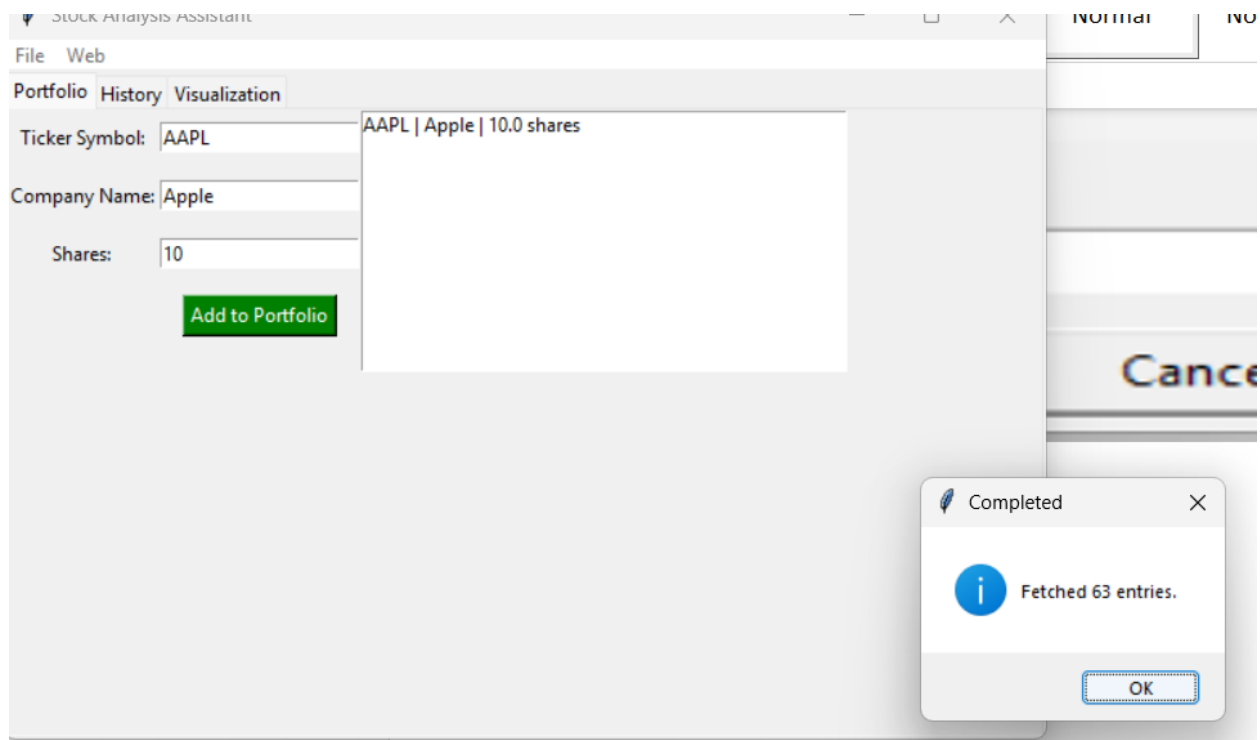
Fetching Data



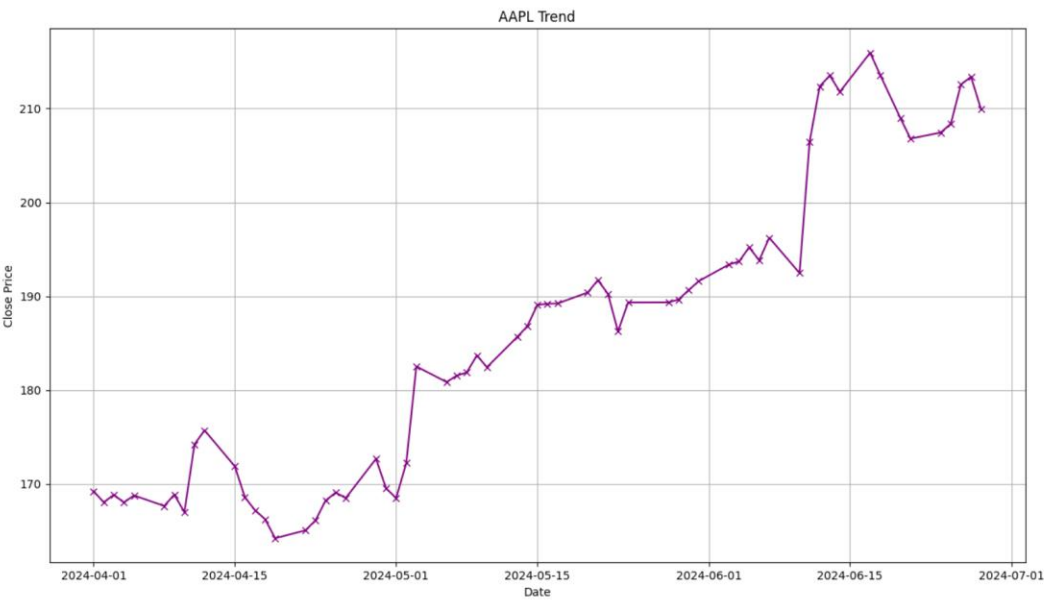
Start Date



Fetches 63 entries



AAPL Trend



Stock_db

```
stock_db.py > save_to_db
1  import sqlite3
2  from stock_models import Stock, DailyData
3
4  DB_NAME = "stock_data.db"
5
6  def save_to_db(portfolio):
7      conn = sqlite3.connect(DB_NAME)
8      cur = conn.cursor()
9
10     # Create tables
11     cur.execute("DROP TABLE IF EXISTS stocks")
12     cur.execute("DROP TABLE IF EXISTS history")
13
14     cur.execute("""
15         CREATE TABLE stocks (
16             symbol TEXT PRIMARY KEY,
17             name TEXT,
18             shares REAL
19         )
20     """)
21
22     cur.execute("""
23         CREATE TABLE history (
24             symbol TEXT,
25             date TEXT,
26             close_price REAL,
27             volume INTEGER,
28             FOREIGN KEY(symbol) REFERENCES stocks(symbol)
29         )
30     """)
31
32     # Insert stock data
33     for stock in portfolio:
34         cur.execute("INSERT INTO stocks VALUES (?, ?, ?)", (stock.symbol, stock.name, stock.shares))
35         for d in stock.history:
36             cur.execute("INSERT INTO history VALUES (?, ?, ?, ?)", (stock.symbol, d.date, d.close price, d.volume))
37
```

Stock Console:

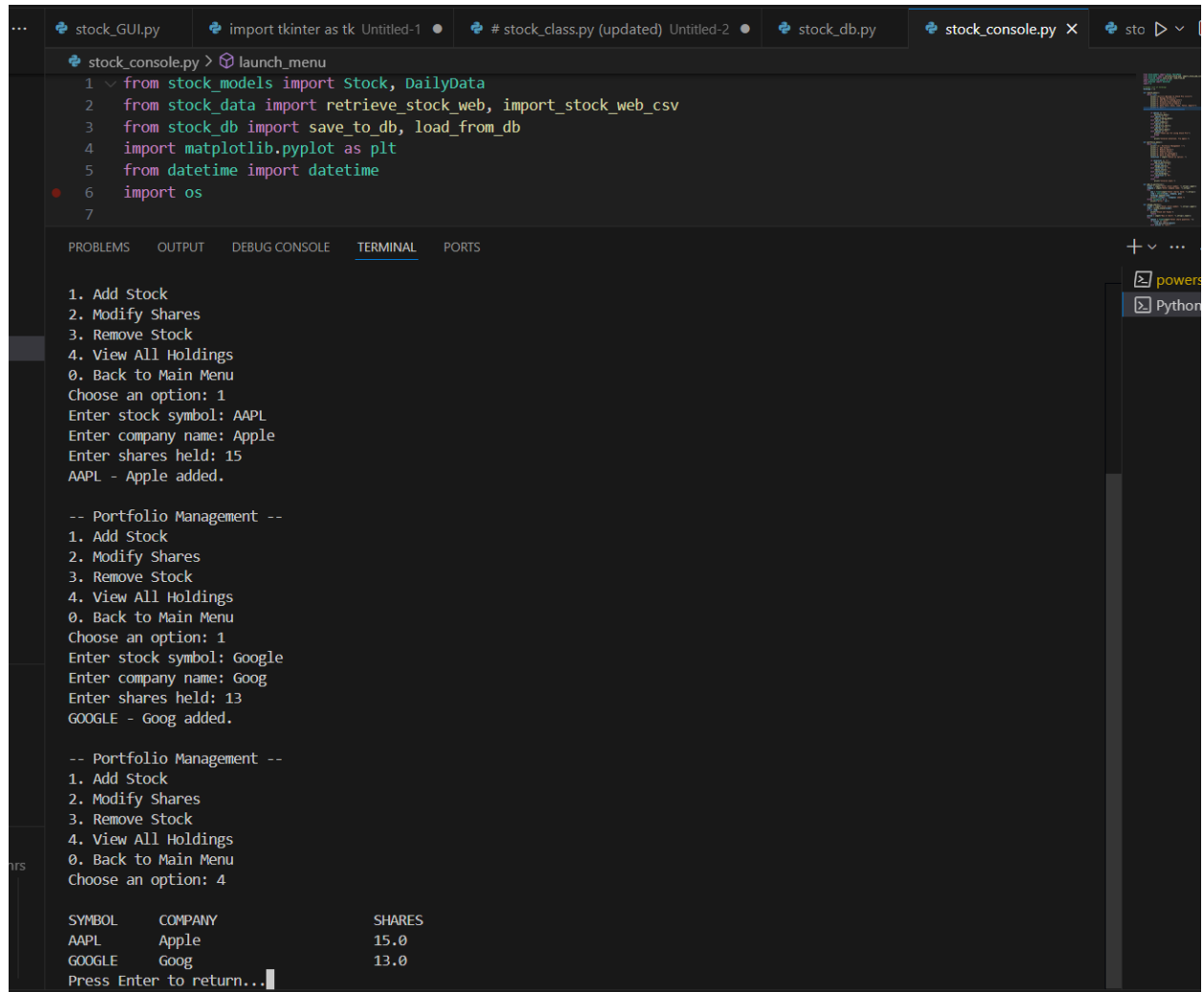
```
stock_GUI.py  import tkinter as tk  Untitled-1  # stock_class.py (updated)  Untitled-2  stock_db.py  stock_console.py  sto >

stock_console.py > ...
1  from stock_models import Stock, DailyData
2  from stock_data import retrieve_stock_web, import_stock_web_csv
3  from stock_db import save_to_db, load_from_db
4  import matplotlib.pyplot as plt
5  from datetime import datetime
6  import os
7
8  # Global list of holdings
9  holdings = []
10
11 def launch_menu():
12     while True:
13         print("\n===== Welcome to Stock Pro =====")
14         print("1. Manage Portfolio")
15         print("2. Add Daily Trading Info")
16         print("3. Display Stock Summary")
17         print("4. Visualize Price Chart")
18         print("5. Data Tools (Save, Load, Fetch, Import)")
19         print("6. Quit")
20         option = input("Enter your choice: ")
21
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\shibi\Downloads\Python Lab 2> & "C:/Users/shibi/Downloads/Python Lab 2/venv/Scripts/python.exe" "c:/Users/shibi/D
ownloads/Python Lab 2/stock_GUI.py"
PS C:\Users\shibi\Downloads\Python Lab 2> & "C:/Users/shibi/Downloads/Python Lab 2/venv/Scripts/python.exe" "c:/Users/shibi/D
ownloads/Python Lab 2/stock_console.py"

===== Welcome to Stock Pro =====
1. Manage Portfolio
2. Add Daily Trading Info
3. Display Stock Summary
4. Visualize Price Chart
5. Data Tools (Save, Load, Fetch, Import)
6. Quit
Enter your choice:
```


Manually adding stock and fetching all the details.



The screenshot shows a code editor with several tabs: `stock_GUI.py`, `import tkinter as tk Untitled-1`, `# stock_class.py (updated) Untitled-2`, `stock_db.py`, and `stock_console.py`. The `stock_console.py` tab is active, displaying the following Python code:

```
1 from stock_models import Stock, DailyData
2 from stock_data import retrieve_stock_web, import_stock_web_csv
3 from stock_db import save_to_db, load_from_db
4 import matplotlib.pyplot as plt
5 from datetime import datetime
6 import os
7
```

Below the code editor is a terminal window with the following output:

```
1. Add Stock
2. Modify Shares
3. Remove Stock
4. View All Holdings
0. Back to Main Menu
Choose an option: 1
Enter stock symbol: AAPL
Enter company name: Apple
Enter shares held: 15
AAPL - Apple added.

-- Portfolio Management --
1. Add Stock
2. Modify Shares
3. Remove Stock
4. View All Holdings
0. Back to Main Menu
Choose an option: 1
Enter stock symbol: Google
Enter company name: Goog
Enter shares held: 13
GOOGLE - Goog added.

-- Portfolio Management --
1. Add Stock
2. Modify Shares
3. Remove Stock
4. View All Holdings
0. Back to Main Menu
Choose an option: 4
```

Below the terminal output is a table showing the current stock holdings:

SYMBOL	COMPANY	SHARES
AAPL	Apple	15.0
GOOGLE	Goog	13.0

Press Enter to return...

Saving Portfolio

```
Press Enter to continue...5

===== Welcome to Stock Pro =====
1. Manage Portfolio
2. Add Daily Trading Info
3. Display Stock Summary
4. Visualize Price Chart
5. Data Tools (Save, Load, Fetch, Import)
6. Quit
Enter your choice: 5

-- Data Utilities --
1. Save Portfolio
2. Load Portfolio
3. Fetch Online Prices
4. Import from CSV
0. Back to Main Menu
Choose option: 1
Saved to database.

-- Data Utilities --
1. Save Portfolio
2. Load Portfolio
3. Fetch Online Prices
4. Import from CSV
0. Back to Main Menu
Choose option: 2
Loaded from database.

-- Data Utilities --
1. Save Portfolio
2. Load Portfolio
3. Fetch Online Prices
4. Import from CSV
0. Back to Main Menu
```

Loading Portfolio

```
-- Data Utilities --
1. Save Portfolio
2. Load Portfolio
3. Fetch Online Prices
4. Import from CSV
0. Back to Main Menu
Choose option: 2
Loaded from database.

-- Data Utilities --
1. Save Portfolio
2. Load Portfolio
3. Fetch Online Prices
4. Import from CSV
0. Back to Main Menu
Choose option: 3
Start Date (m/d/yy): 03/01/2024
End Date (m/d/yy): 04/01/2024
📄 Downloading AAPL from 2024-03-01 to 2024-04-01
YF.download() has changed argument auto_adjust default to True
c:\Users\shibi\Downloads\Python Lab 2\stock_data.py:31: FutureWarning: Calling int on a single element Series is deprecated a
nd will raise a TypeError in the future. Use int(ser.iloc[0]) instead
  volume = int(row['Volume'])
```

Fetching Online Prices

```
Choose option: 2
Loaded from database.

-- Data Utilities --
1. Save Portfolio
2. Load Portfolio
3. Fetch Online Prices
4. Import from CSV
0. Back to Main Menu
Choose option: 3
Start Date (m/d/yy): 03/01/2024
End Date (m/d/yy): 04/01/2024
📄 Downloading AAPL from 2024-03-01 to 2024-04-01
YF.download() has changed argument auto_adjust default to True
c:\Users\shibi\Downloads\Python Lab 2\stock_data.py:31: FutureWarning: Calling int on a single element Series is deprecated and will raise a TypeError in the future. Use int(ser.iloc[0]) instead
    volume = int(row['Volume'])
c:\Users\shibi\Downloads\Python Lab 2\stock_class.py:4: FutureWarning: Calling float on a single element Series is deprecated and will raise a TypeError in the future. Use float(ser.iloc[0]) instead
    self.close = float(close)
✓AAPL - Mar 01, 2024: Close = Ticker
AAPL    178.82
Name: 2024-03-01 00:00:00, dtype: float64, Volume = 73488000
✓AAPL - Mar 04, 2024: Close = Ticker
AAPL    174.28
Name: 2024-03-04 00:00:00, dtype: float64, Volume = 81510100
✓AAPL - Mar 05, 2024: Close = Ticker
AAPL    169.32
```

Importing from CSV

```
stock_GUI.py  import tkinter as tk  Untitled-1  # stock_class.py (updated)  Untitled-2  stock_db.py  stock_console.py > launch_menu

180  # Helper to locate stock object
181  def locate_stock(ticker):
182      return next((s for s in holdings if s.symbol.upper() == ticker.upper()), None)
183

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

📄 Downloading GOOGLE from 2024-03-01 to 2024-04-01

1 Failed download:
['GOOGLE']: HTTPError('HTTP Error 404: ')
⚠️No data for GOOGLE
Web data retrieved.

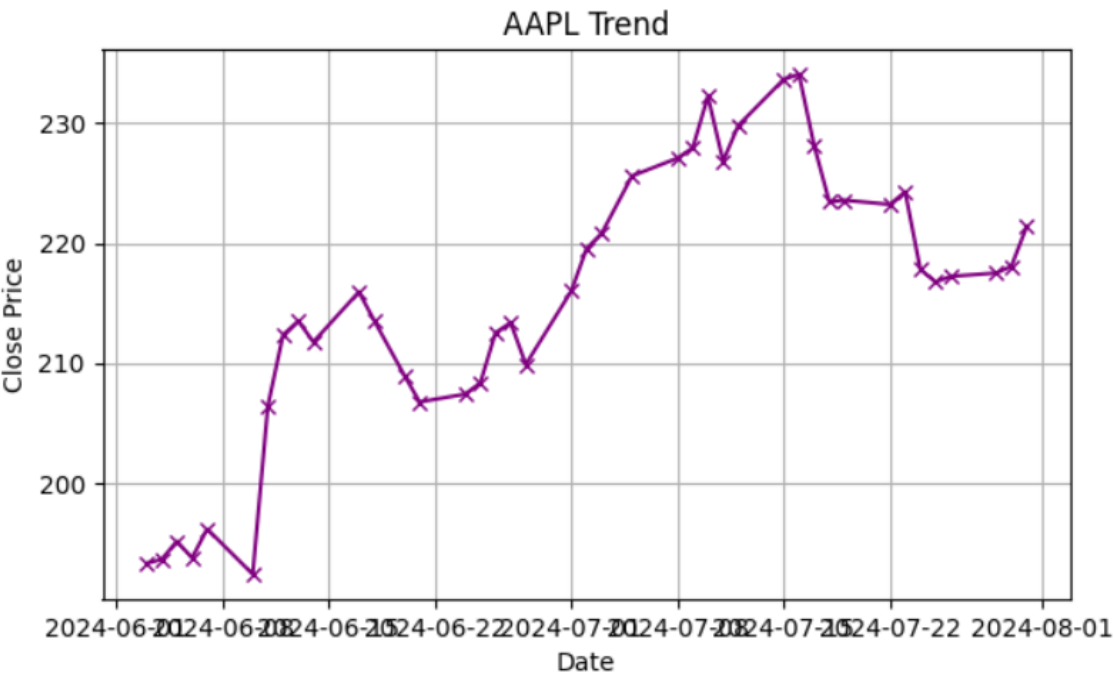
-- Data Utilities --
1. Save Portfolio
2. Load Portfolio
3. Fetch Online Prices
4. Import from CSV
0. Back to Main Menu
Choose option: 4
Enter stock symbol: AAPL
```

Daily Trading Info doing selling and buying of stocks

```
===== Welcome to Stock Pro =====
1. Manage Portfolio
2. Add Daily Trading Info
3. Display Stock Summary
4. Visualize Price Chart
5. Data Tools (Save, Load, Fetch, Import)
6. Quit
Enter your choice: 1

-- Portfolio Management --
1. Add Stock
2. Modify Shares
3. Remove Stock
4. View All Holdings
0. Back to Main Menu
Choose an option: 2
Enter stock symbol: AAPL
Buy or Sell?: Sell
Enter share quantity: 5
```

Visualization



Key Learnings & Insights (Summary)

- **Gained practical experience with web scraping using Selenium and BeautifulSoup.**
- **Learned to import and process CSV files for financial data integration.**
- **Strengthened object-oriented programming skills by working with custom classes.**
- **Built both GUI and console interfaces to interact with stock data.**
- **Applied data visualization using matplotlib to generate stock price charts.**
- **Understood the importance of modular code, error handling, and step-by-step testing.**