CEIS 150 PROJECT: PYTHON STOCK TRACKING SHEET

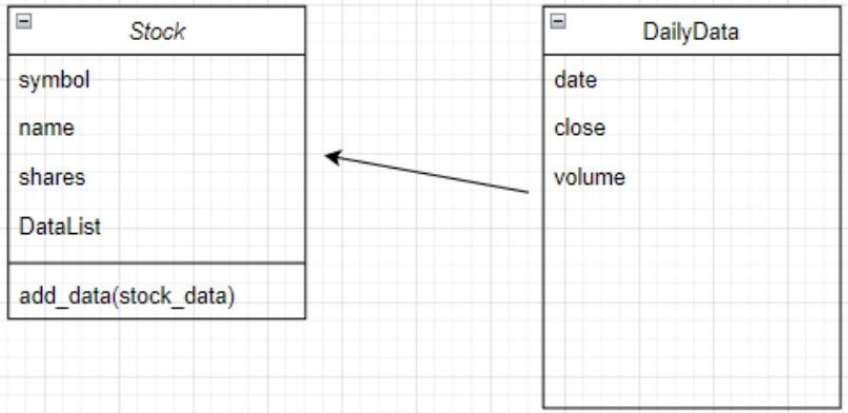
Tyler Brainer

INTRODUCTION

I designed a program in python that had a multifunction menu. You can add, delete, and list stocks as well as add daily data, and show a chart.

CLASS DIAGRAMS, CODING, AND UNIT TESTING

First I Created a Diagram to show what variables need to apply to each stock. I then began coding each Class and definition for parameters. I finished with a Unit test to make sure calculations followed as expected.



brai\Documents\stock_class.py

ock_class.py ×

```python
# -*- coding: utf-8 -*-
"""
Created on Sun Jan 15 21:29:57 2023

@author: tbrai
"""

class Stock:
    def __init__(self, symbol, name, shares):
        self.symbol = symbol
        self.name = name
        self.shares = shares
        self.DataList = []  #list of daily stock data

    def add_data(self, stock_data):
        self.DataList.append(stock_data)

class DailyData:
    def __init__(self, date, close, volume):
        self.date = date
        self.close = close
        self.volume = volume
```

Console 1/A ×

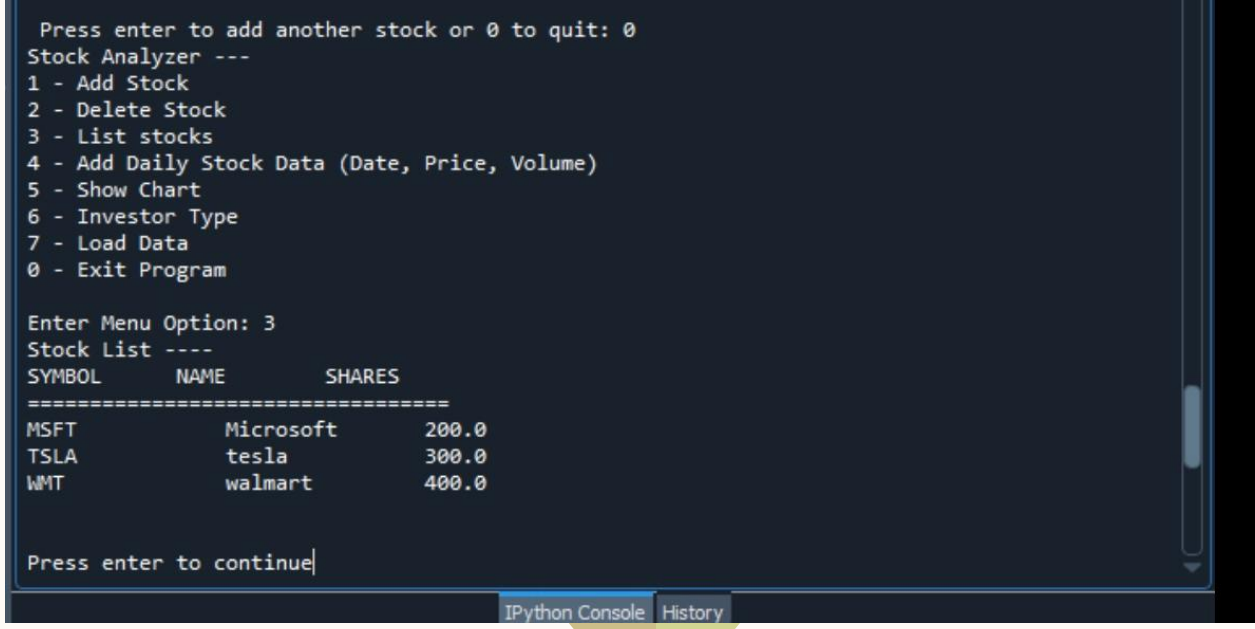
```
In [5]: runfile('C:/Users/tbrai/Documents/stock_class.py', wdir='C:/Users/tbrai/Documents')
Unit Testing Starting---
Testing Add Stock...Successful!
Test Change Symbol...Successful!
Test Change Name...Successful!
Successful!
Creating daily stock data...Successful!
Congratulations - All Tests Passed
Goodbye

In [6]:
```

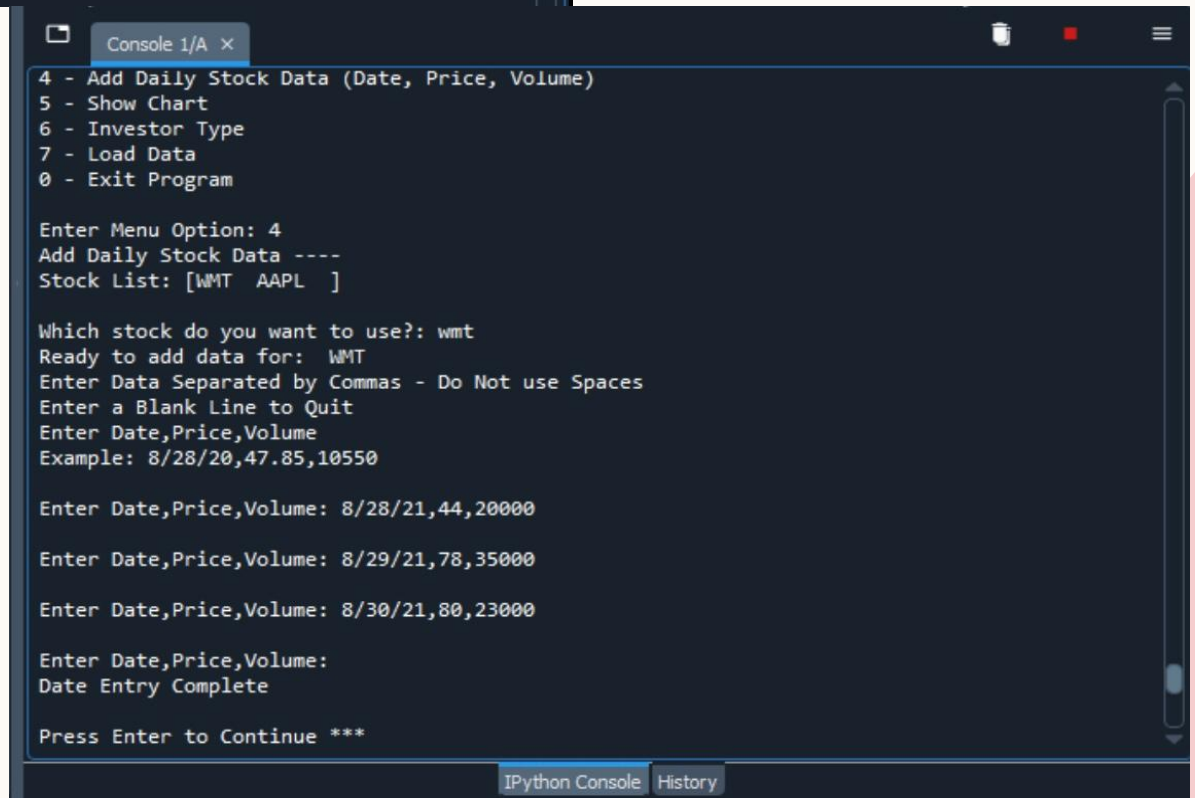
IPython Console History

ADDING STOCKS AND DATA, AND LISTING

I coded the functions for adding a stock and listing a stock. I tested them listing a few stocks and entered daily data for one of them.


Console 1/A
7 - Load Data
0 - Exit Program
Enter Menu Option: 1
Adding a stock
Enter symbol: msft
Enter company name: Microsoft
Enter shares: 200
Press enter to add another stock or 0 to quit:
Adding a stock
Enter symbol: tsla
Enter company name: tesla
Enter shares: 300
Press enter to add another stock or 0 to quit: 0


Press enter to add another stock or 0 to quit: 0
Stock Analyzer ---
1 - Add Stock
2 - Delete Stock
3 - List stocks
4 - Add Daily Stock Data (Date, Price, Volume)
5 - Show Chart
6 - Investor Type
7 - Load Data
0 - Exit Program
Enter Menu Option: 3
Stock List ----
SYMBOL NAME SHARES
MSFT Microsoft 200.0
TSLA tesla 300.0
WMT walmart 400.0
Press enter to continue
IPython Console
History


Console 1/A
4 - Add Daily Stock Data (Date, Price, Volume)
5 - Show Chart
6 - Investor Type
7 - Load Data
0 - Exit Program
Enter Menu Option: 4
Add Daily Stock Data ----
Stock List: [WMT AAPL]
Which stock do you want to use?: wmt
Ready to add data for: WMT
Enter Data Separated by Commas - Do Not use Spaces
Enter a Blank Line to Quit
Enter Date,Price,Volume
Example: 8/28/20,47.85,10550
Enter Date,Price,Volume: 8/28/21,44,20000
Enter Date,Price,Volume: 8/29/21,78,35000
Enter Date,Price,Volume: 8/30/21,80,23000
Enter Date,Price,Volume:
Date Entry Complete
Press Enter to Continue ***
IPython Console
History

INHERITANCE WITH CLASSES

I created several more classes in a separate file. These classes inherit the properties which means I can use the properties from the main class for the other ones. I followed up with a Unit test.

stock_class.py × stock_menu.py × account_class.py ×

```python
# -*- coding: utf-8 -*-
"""
Created on Wed Jan 25 18:04:47 2023

@author: tbrai
"""

from stock_class import Stock
#create parent class
class Retirement_Account:
    def __init__(self, balance, number):
        self.balance = balance
        self.number = number

#create Traditional Account
class Traditional(Retirement_Account):
    def __init__(self, balance, number):
        Retirement_Account.__init__(self, balance, number)
        self.Stock_List =[]

    def add_stock(self, stock_data):
        self.Stock_List.append(stock_data)

#create derived class Robo account

class Robo(Retirement_Account):
    def __init__(self, balance, number, years):
        Retirement_Account.__init__(self, balance, number)
        self.years = years

    def investment_return(self):
        return (self.years*self.balance*1.05)
```

Console 1/A ×

```
Stock Analyzer ---
1 - Add Stock
2 - Delete Stock
3 - List stocks
4 - Add Daily Stock Data (Date, Price, Volume)
5 - Show Chart
6 - Investor Type
7 - Load Data
0 - Exit Program

Enter Menu Option: 6
Investment Account ---

What is your initial balance: 5000

What is your account number: 8765

Do you want a Traditional (t) or Robo (r) account: t
Choose stocks from the list below:
Stock List: [MSFT  AAPL  ]

Which stock do you want to purchase, 0 to quit: msft

How many shares do you want to buy?: 700
Bought  700.0 of MSFT
```

IPython Console History

Console 1/A ×

```
Enter shares: 500

 Press enter to add another stock or 0 to quit: 0
Stock Analyzer ---
1 - Add Stock
2 - Delete Stock
3 - List stocks
4 - Add Daily Stock Data (Date, Price, Volume)
5 - Show Chart
6 - Investor Type
7 - Load Data
0 - Exit Program

Enter Menu Option: 6
Investment Account ---

What is your initial balance: 5000

What is your account number: 8765

Do you want a Traditional (t) or Robo (r) account: r

How many years until retirement: 12
Your investment return is  63000.0
```

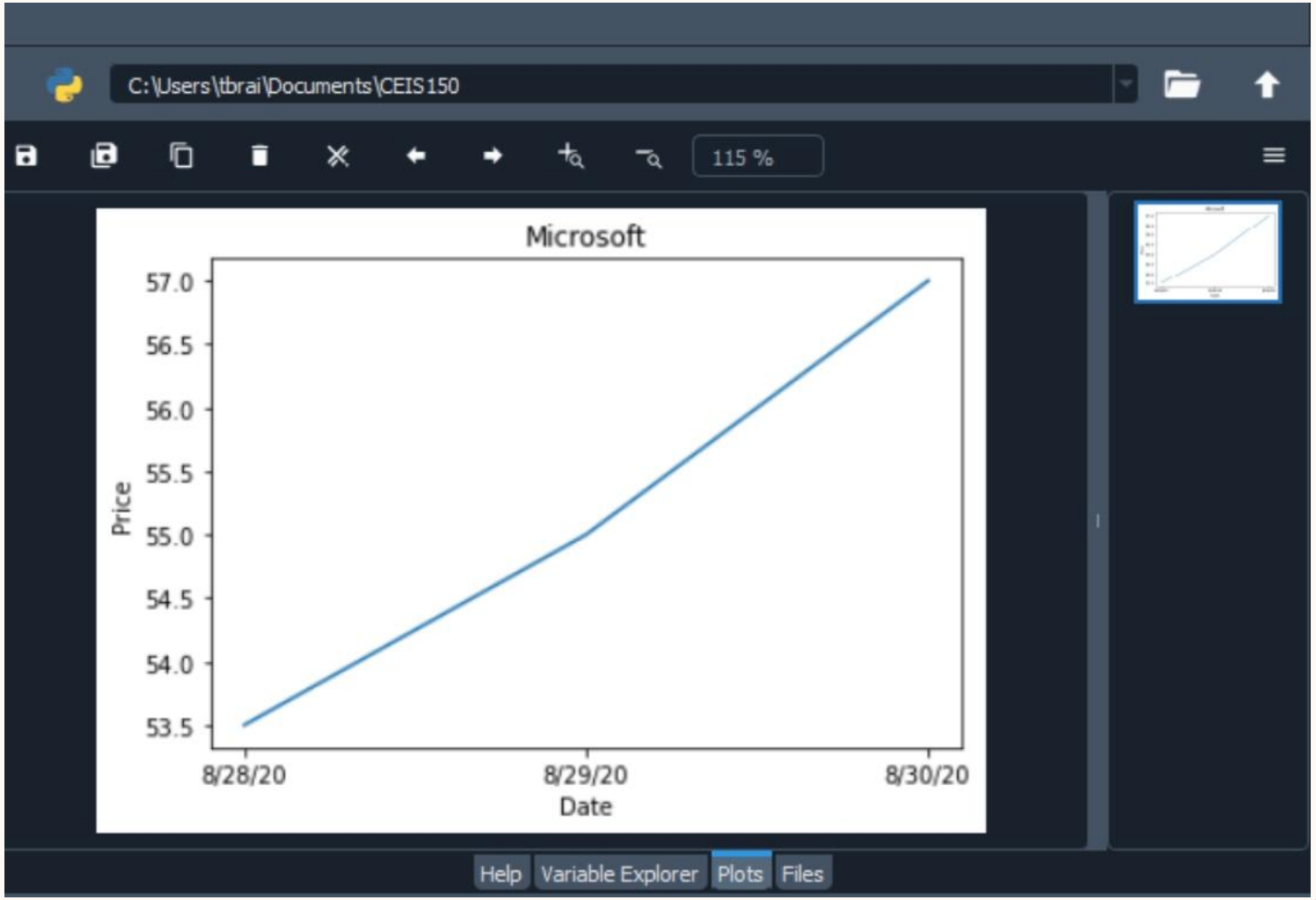
IPython Console History

Console 1/A ×

```
In [2]: runfile('C:/Users/tbrai/Documents/CEIS150/account_class.py', wdir='C:/Users/tbrai/
Documents/CEIS150')
Reloaded modules: stock_class
Unit Testing Starting---
Testing Add Retirement Account...Successful!
Testing Add Traditional Account...Successful!
Test Change Balance...Successful!
Test Change Number...Successful!
Testing Add Robo Account...Successful!
Test Change Balance...Successful!
Test investment return...Successful!
Congratulations - All Tests Passed
Goodbye
```

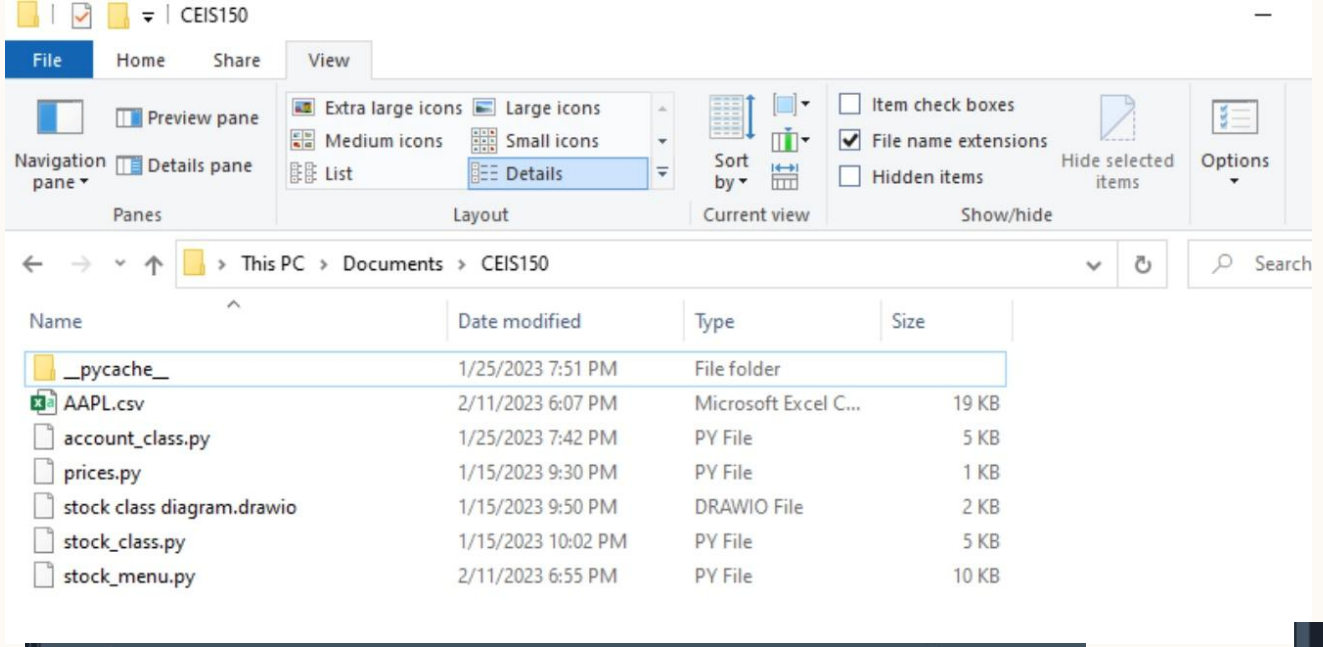
CREATING A CHART

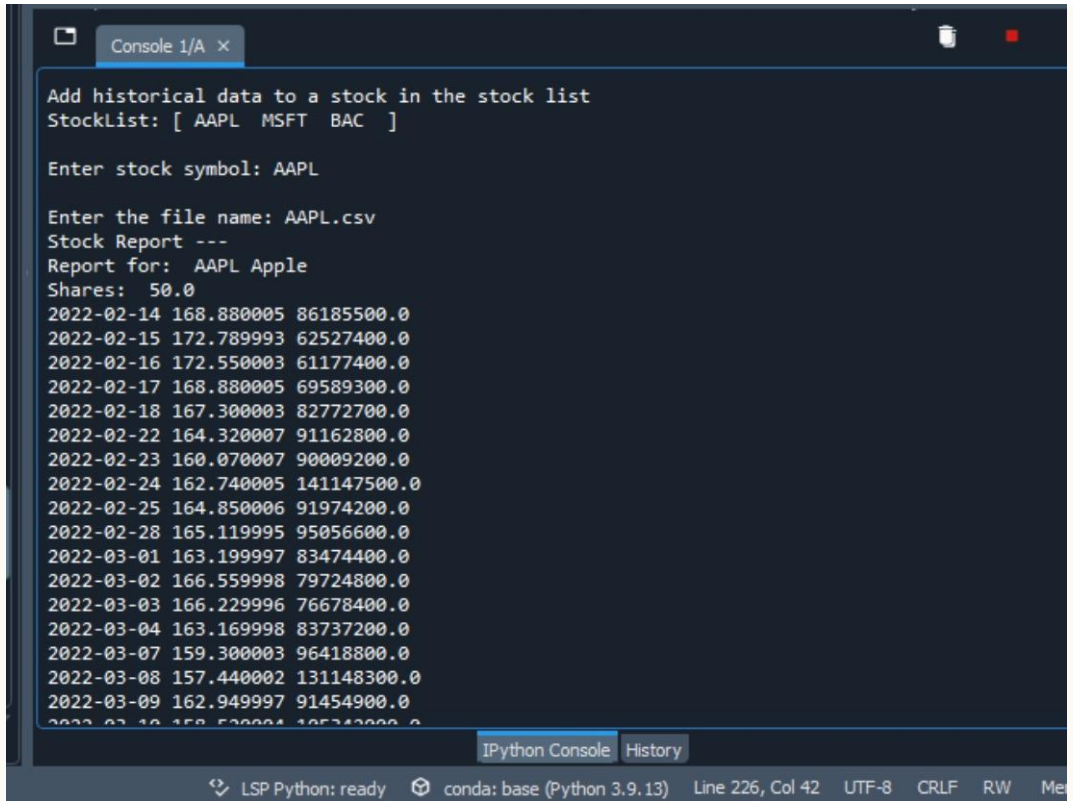
I used a python library for plotting on a graph to take information entered for the daily data and display it in a chart

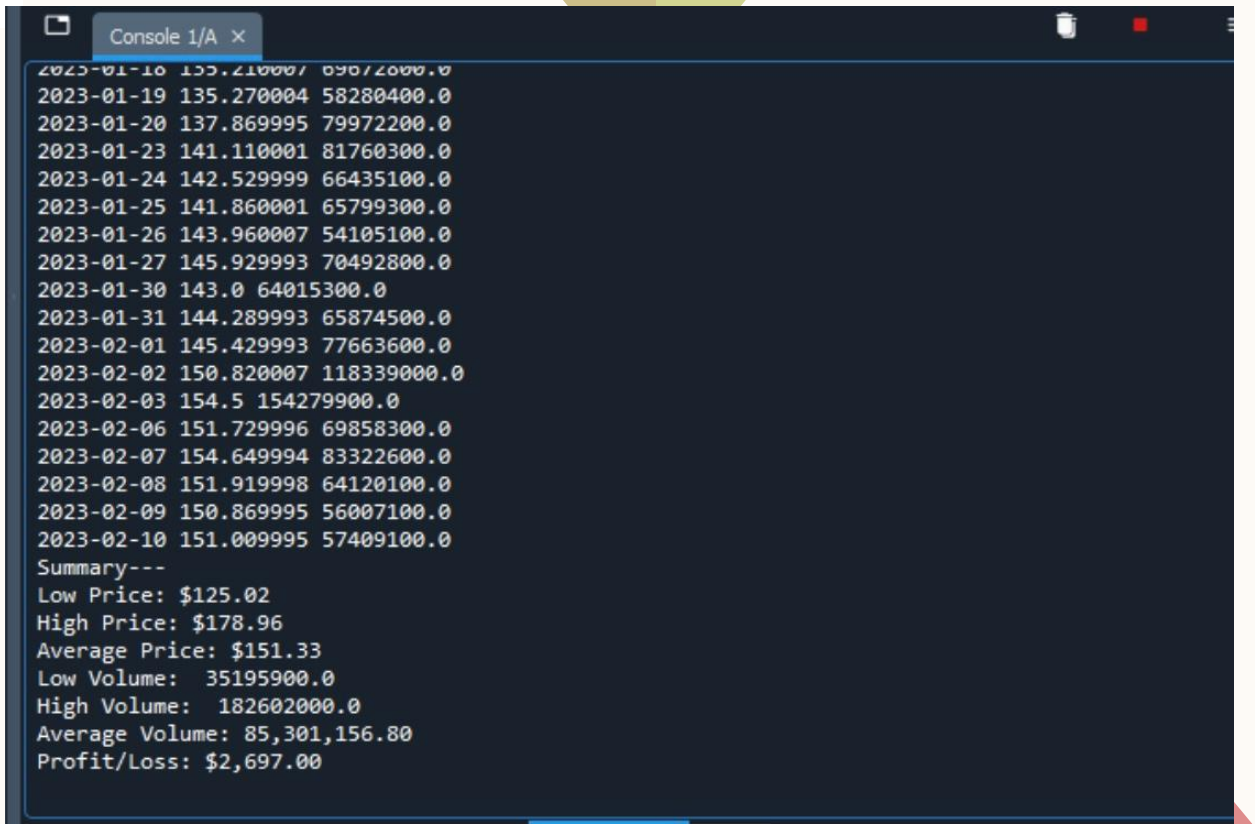

C:\Users\tbrai\Documents\CEIS150
115 %
Microsoft
57.0
56.5
56.0
55.5
55.0
54.5
54.0
53.5
Price
8/28/20
8/29/20
8/30/20
Date
Help
Variable Explorer
Plots
Files

LOADING DATA

I used data from a CSV file to display historical history of a stock to the user.

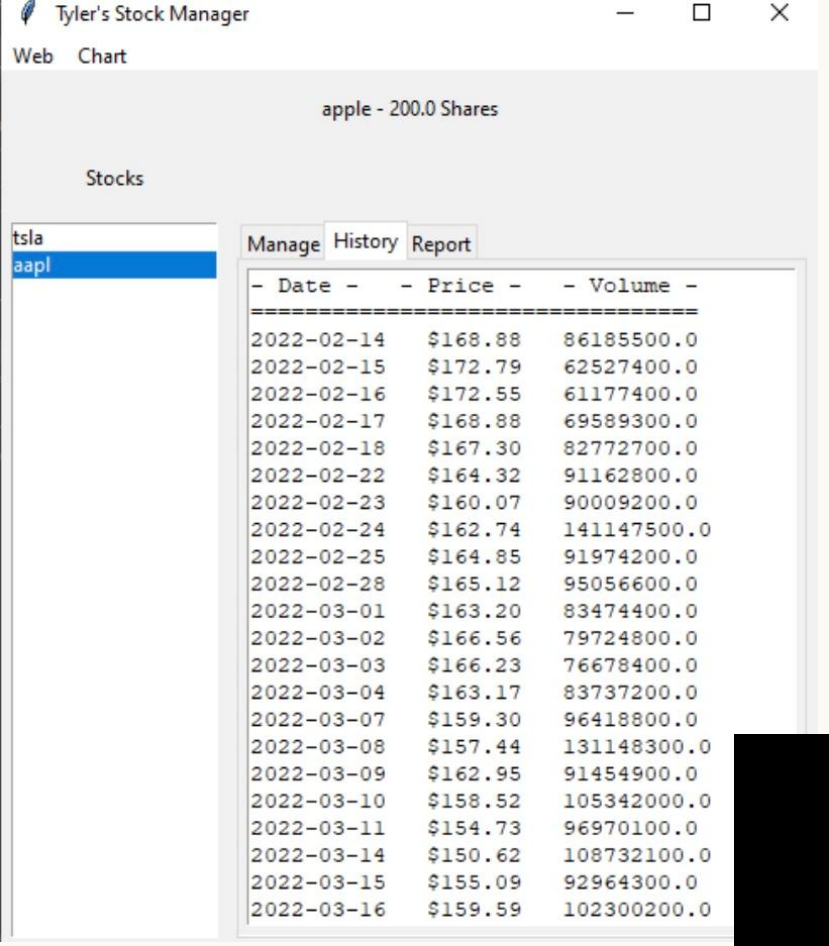

CEIS150
File
Home
Share
View
Navigation pane
Preview pane
Details pane
Panes
Extra large icons
Large icons
Medium icons
Small icons
List
Details
Layout
Sort by
Current view
Item check boxes
File name extensions
Hidden items
Hide selected items
Show/hide
Options
This PC › Documents › CEIS150
Search
Name
Date modified
Type
Size
__pycache__ 1/25/2023 7:51 PM File folder
AAPL.csv 2/11/2023 6:07 PM Microsoft Excel C... 19 KB
account_class.py 1/25/2023 7:42 PM PY File 5 KB
prices.py 1/15/2023 9:30 PM PY File 1 KB
stock class diagram.drawio 1/15/2023 9:50 PM DRAWIO File 2 KB
stock_class.py 1/15/2023 10:02 PM PY File 5 KB
stock_menu.py 2/11/2023 6:55 PM PY File 10 KB

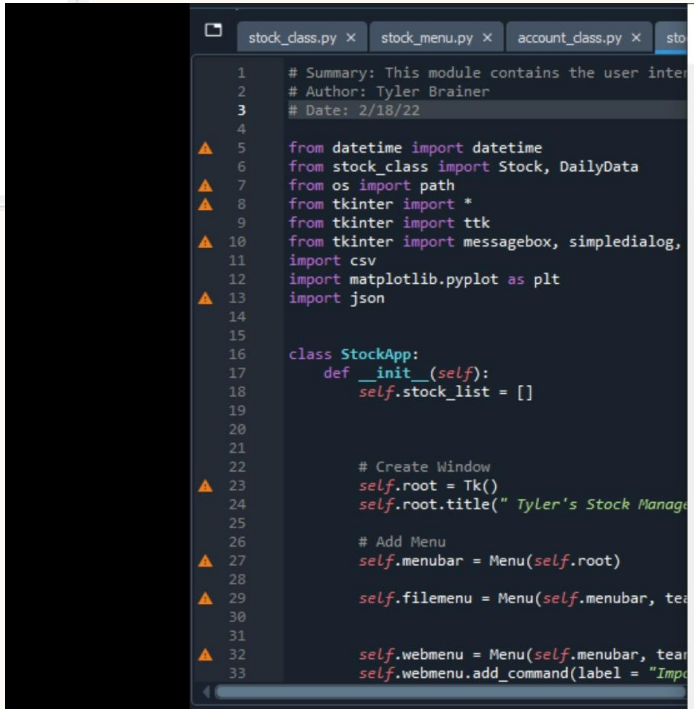

Console 1/A
Add historical data to a stock in the stock list
StockList: [AAPL MSFT BAC]
Enter stock symbol: AAPL
Enter the file name: AAPL.csv
Stock Report ---
Report for: AAPL Apple
Shares: 50.0
2022-02-14 168.880005 86185500.0
2022-02-15 172.789993 62527400.0
2022-02-16 172.550003 61177400.0
2022-02-17 168.880005 69589300.0
2022-02-18 167.300003 82772700.0
2022-02-22 164.320007 91162800.0
2022-02-23 160.070007 90009200.0
2022-02-24 162.740005 141147500.0
2022-02-25 164.850006 91974200.0
2022-02-28 165.119995 95056600.0
2022-03-01 163.199997 83474400.0
2022-03-02 166.559998 79724800.0
2022-03-03 166.229996 76678400.0
2022-03-04 163.169998 83737200.0
2022-03-07 159.300003 96418800.0
2022-03-08 157.440002 131148300.0
2022-03-09 162.949997 91454900.0
IPython Console
History
LSP Python: ready
conda: base (Python 3.9.13)
Line 226, Col 42
UTF-8
CRLF
RW

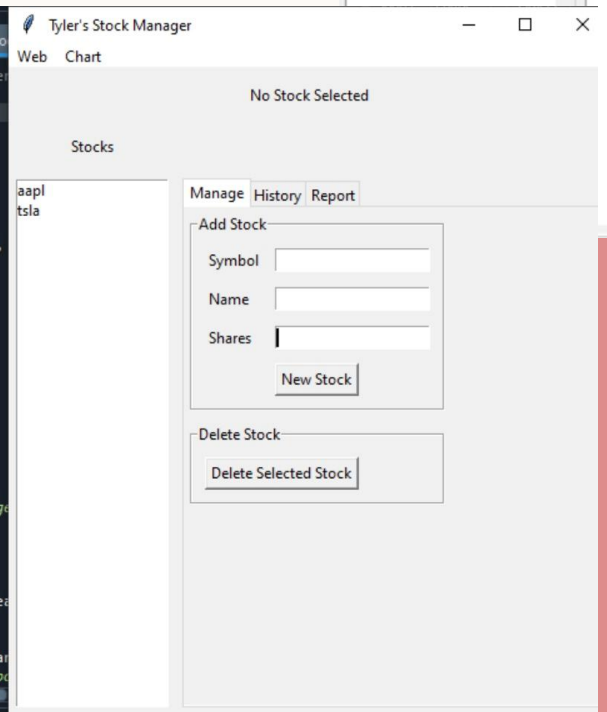

Console 1/A
2023-01-19 135.270004 58280400.0
2023-01-20 137.869995 79972200.0
2023-01-23 141.110001 81760300.0
2023-01-24 142.529999 66435100.0
2023-01-25 141.860001 65799300.0
2023-01-26 143.960007 54105100.0
2023-01-27 145.929993 70492800.0
2023-01-30 143.0 64015300.0
2023-01-31 144.289993 65874500.0
2023-02-01 145.429993 77663600.0
2023-02-02 150.820007 118339000.0
2023-02-03 154.5 154279900.0
2023-02-06 151.729996 69858300.0
2023-02-07 154.649994 83322600.0
2023-02-08 151.919998 64120100.0
2023-02-09 150.869995 56007100.0
2023-02-10 151.009995 57409100.0
Summary---
Low Price: $125.02
High Price: $178.96
Average Price: $151.33
Low Volume: 35195900.0
High Volume: 182602000.0
Average Volume: 85,301,156.80
Profit/Loss: $2,697.00

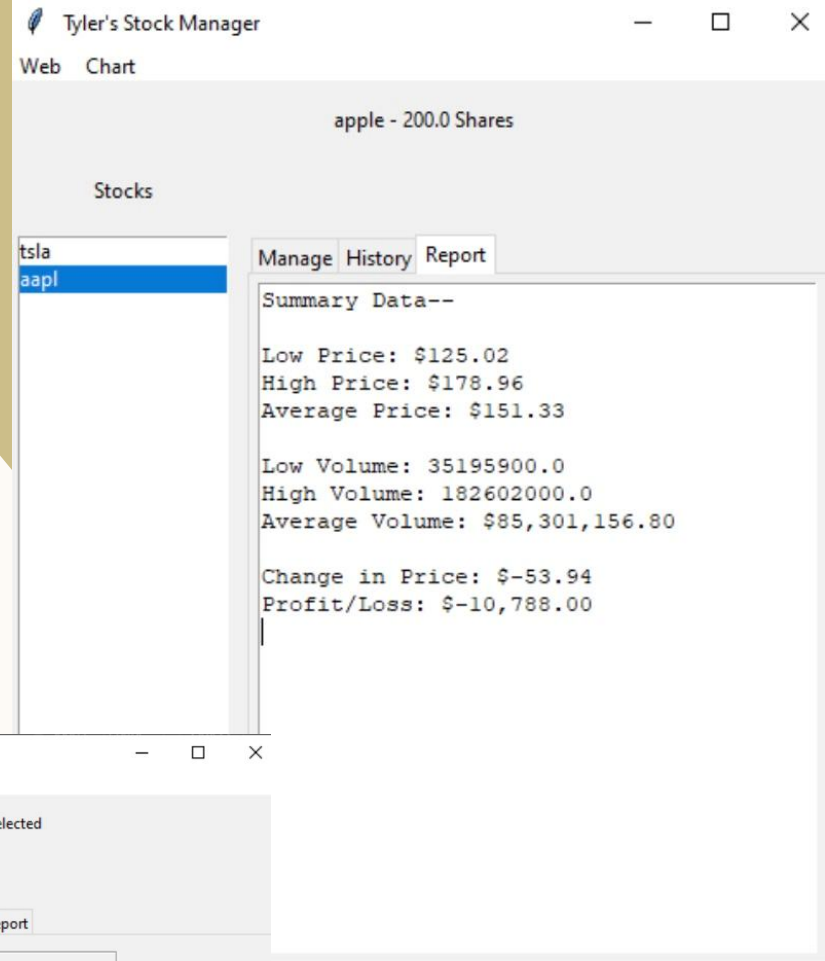
CREATING A GUI

I created a GUI for my program to be better suited for the end user. This included adding and deleting a stock, importing web data (CSV), and history and report tab


Tyler's Stock Manager
Web
Chart
apple - 200.0 Shares
Stocks
tsla
aapl
Manage
History
Report
- Date - - Price - - Volume -
2022-02-14 $168.88 86185500.0
2022-02-15 $172.79 62527400.0
2022-02-16 $172.55 61177400.0
2022-02-17 $168.88 69589300.0
2022-02-18 $167.30 82772700.0
2022-02-22 $164.32 91162800.0
2022-02-23 $160.07 90009200.0
2022-02-24 $162.74 141147500.0
2022-02-25 $164.85 91974200.0
2022-02-28 $165.12 95056600.0
2022-03-01 $163.20 83474400.0
2022-03-02 $166.56 79724800.0
2022-03-03 $166.23 76678400.0
2022-03-04 $163.17 83737200.0
2022-03-07 $159.30 96418800.0
2022-03-08 $157.44 131148300.0
2022-03-09 $162.95 91454900.0
2022-03-10 $158.52 105342000.0
2022-03-11 $154.73 96970100.0
2022-03-14 $150.62 108732100.0
2022-03-15 $155.09 92964300.0
2022-03-16 $159.59 102300200.0


stock_class.py
stock_menu.py
account_class.py
# Summary: This module contains the user inter
# Author: Tyler Brainer
# Date: 2/18/22
from datetime import datetime
from stock_class import Stock, DailyData
from os import path
from tkinter import *
from tkinter import ttk
from tkinter import messagebox, simpledialog,
import csv
import matplotlib.pyplot as plt
import json
class StockApp:
def __init__(self):
self.stock_list = []
# Create Window
self.root = Tk()
self.root.title(" Tyler's Stock Manag
# Add Menu
self.menubar = Menu(self.root)
self.filemenu = Menu(self.menubar, te
self.webmenu = Menu(self.menubar, tea
self.webmenu.add_command(label = "Imp


Tyler's Stock Manager
Web
Chart
No Stock Selected
Stocks
aapl
tsla
Manage
History
Report
Add Stock
Symbol
Name
Shares
New Stock
Delete Stock
Delete Selected Stock


Tyler's Stock Manager
Web
Chart
apple - 200.0 Shares
Stocks
tsla
aapl
Manage
History
Report
Summary Data--
Low Price: $125.02
High Price: $178.96
Average Price: $151.33
Low Volume: 35195900.0
High Volume: 182602000.0
Average Volume: $85,301,156.80
Change in Price: $-53.94
Profit/Loss: $-10,788.00

CAREER SKILLS DEVELOPED

- Coding using python
- Object oriented programming
- Using python libraries
- Web Scrapping in programming

CONCLUSION

In this project I learned how to use object oriented programming to code more efficiently. I have a better understanding of how to use data from the web to implement in the program as well as programming syntax. I enjoyed programming each step and seeing the final project come together.