inStock Advanced

Python Stock Tracking Project

By Miguel Mastache

10/28/22

Introduction

inStock is a simple stock tracking application. It is capable of importing historical stock information, from either Yahoo! Finance, via CSV or web scraping. This app is able to save your stock purchase history, and produce reports based on your purchase history.

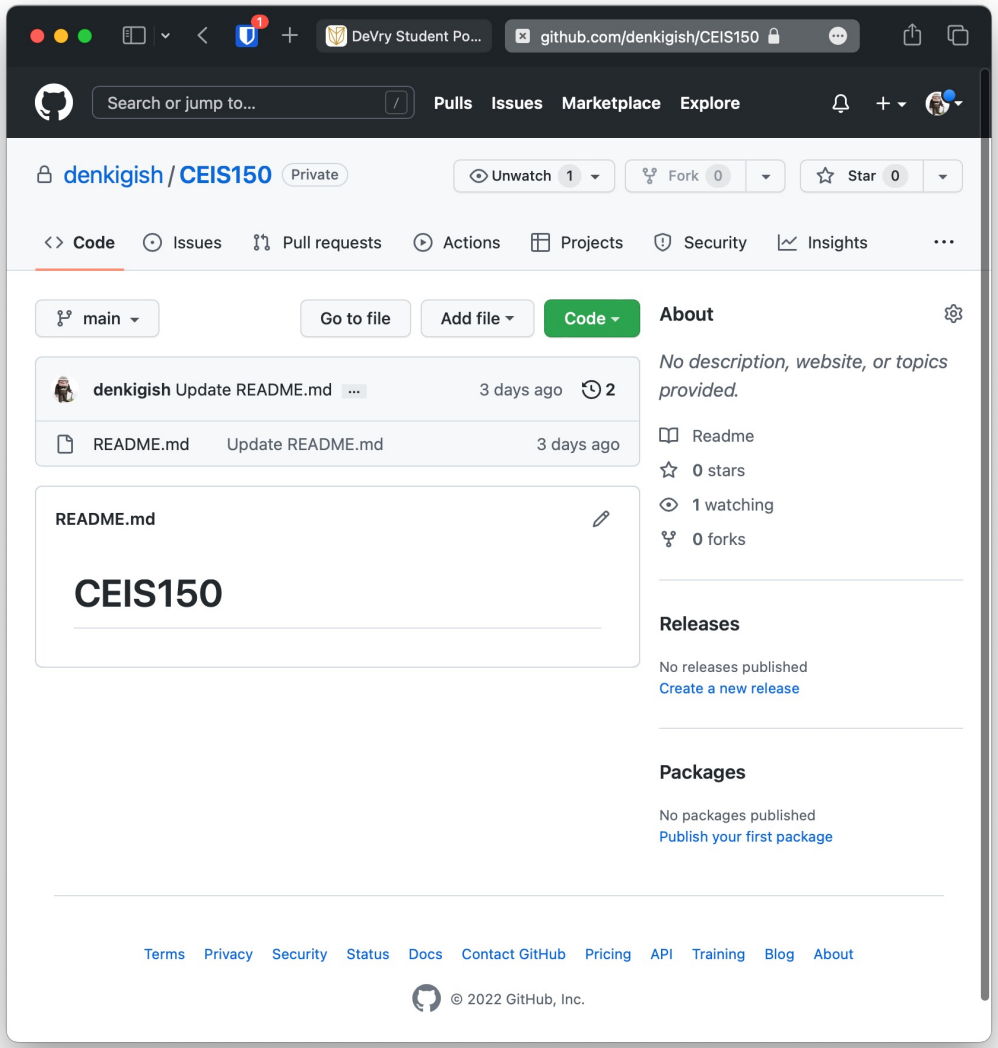
Environment

This part of the project focus' on setting up the environment used to develop the stock tracking application.

GitHub

- Screen shot of GitHub project repository.

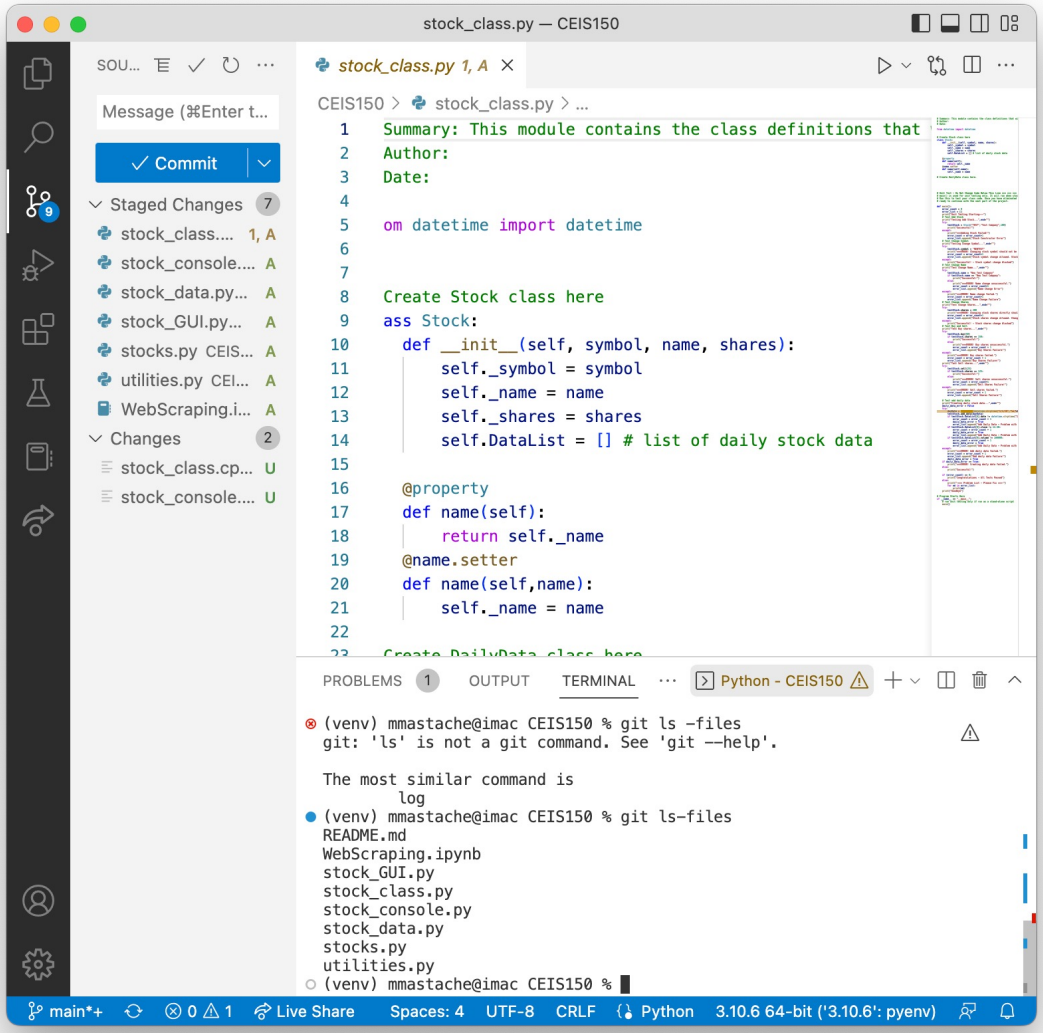
- For this class, I setup a GitHub repository. It can be found at:
- https://github.com/denkigish



IDE & Starter Files

- Screen shot of your IDE (Spyder or VS Code) with the project Starter Files loaded.

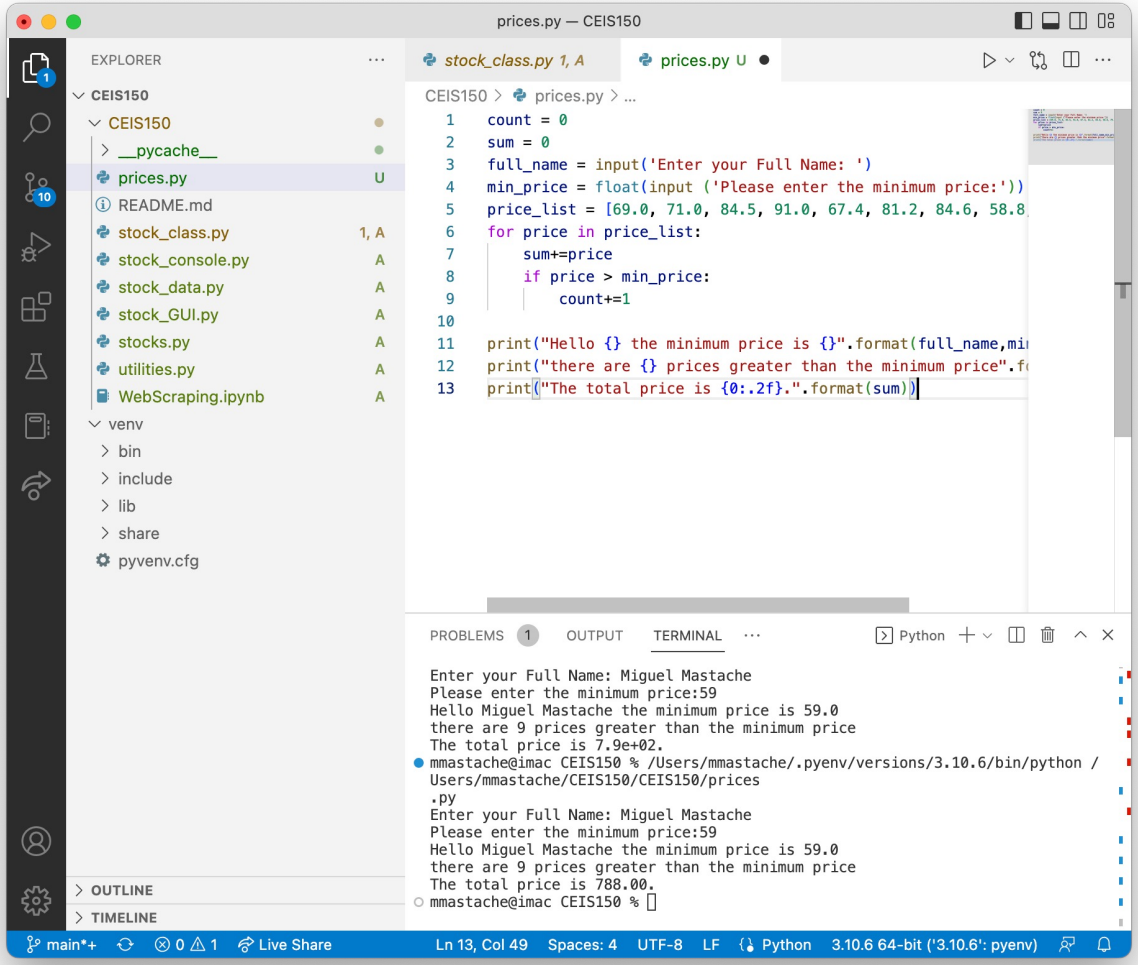
- Also chose to use the VS Code IDE for development of the project. Shown is VS Code with the starter files loaded.



Program

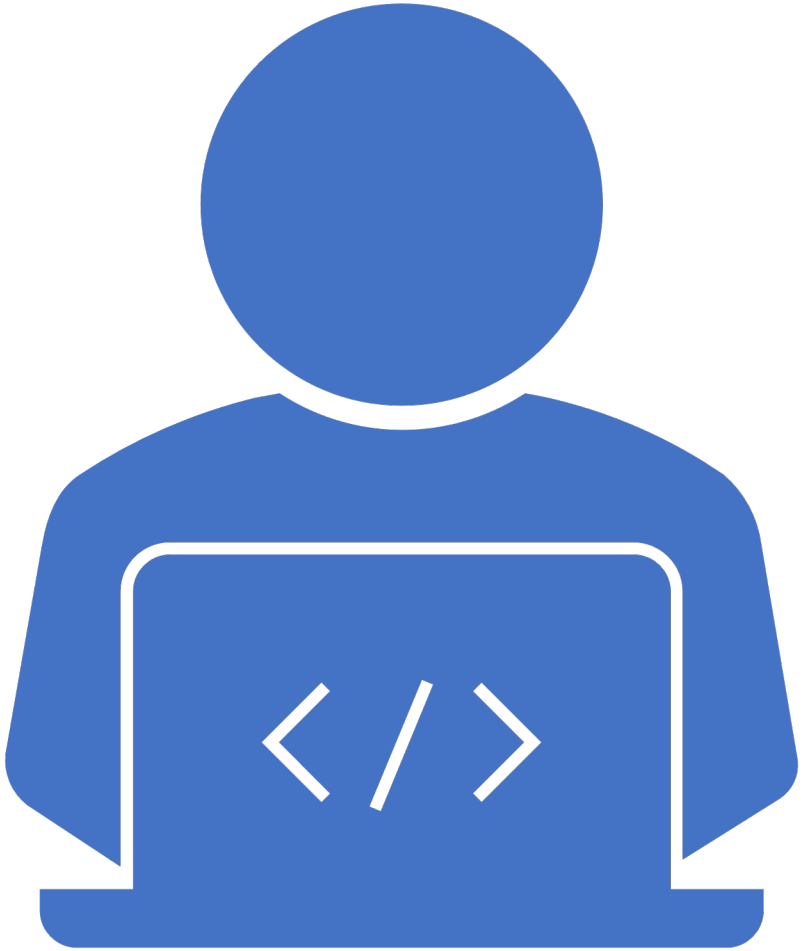
- Screen shot of Python program running successfully.

- prices.py working with price list, summing, and counting stocks above minimum price.



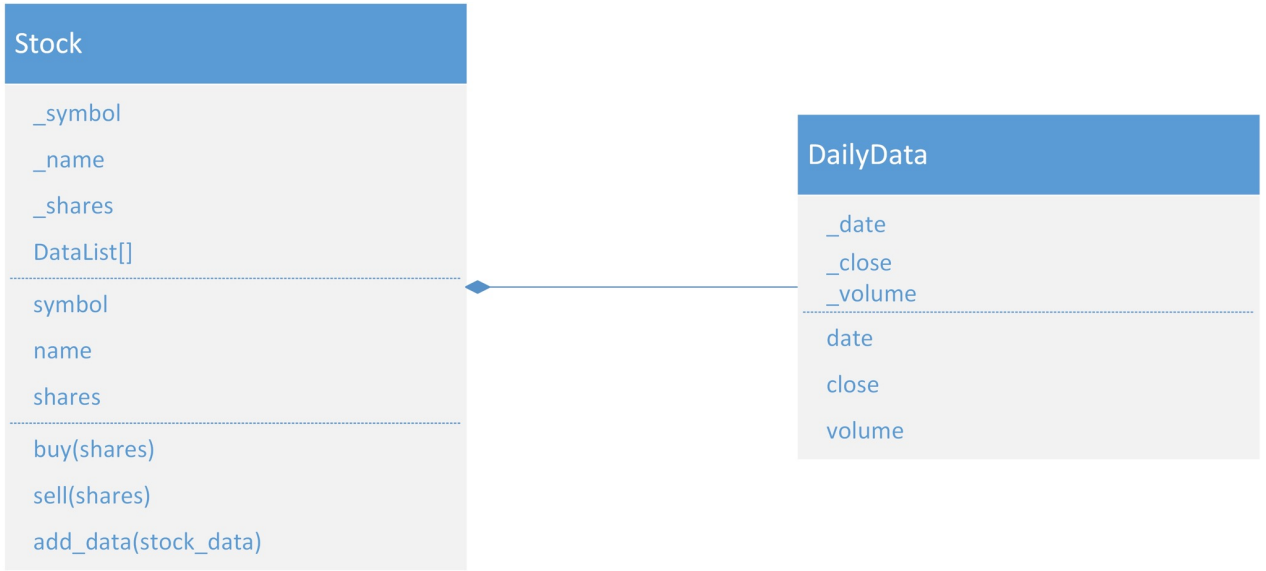
UML and Class Definition

This section covers the development of the classes needed to work with our stock and history data. These classes are the foundation of my program.



Class Diagram

- Paste your Visio Class Diagram



Class Code

- Screen Shot of your stock_class.py file.

stock_class.py — inStock

stock_class.py M

stock_class.py > ...

```python
summary: This module contains the class definitions that will be used in the stock anal
uthor: Miguel Mastache
ate: 11.05.2022

m datetime import datetime


tock Class creates is for creating stock objects that track,
he name and symbol of tock and how many shares are owned of said stock
it uses property object for setting accessor functions.
tock keeps track of the daily data.
ss Stock:
  def __init__(self, symbol, name, shares):
      self._symbol = symbol
      self._name = name
      self._shares = shares
      self.DataList = [] # list of daily stock data

  @property
  def symbol(self):
      return self._symbol
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

```
hon /Users/mmastache/Python/inStock/stock_class.py
Unit Testing Starting---
Testing Add Stock...Successful!
Testing Change Symbol...Successful! - Stock symbol change blocked
Test Change Name...Successful!
Test Change Shares...Successful! - Stock shares change blocked
Test Buy shares...Successful!
Test Sell shares...Successful!
Creating daily stock data...Successful!
Congratulations - All Tests Passed
Goodbye
(venv) mmastache@mac inStock %
```

Python
Python

main* 0↓ 1↑ ⊗ 0 ⚠ 0 Live Share UTF-8 CRLF Python 3.10.7 ('venv': venv)

Unit Test

- Screen Shot of your successful unit test.

stock_class.py — inStock

stock_class.py M

stock_class.py > Stock

```python
# DailyData class, creates the daily data object, which is date, closing price, and vo
# this is used by Stock to keep track of the daily data.
class DailyData:
    def __init__(self,date, close, volume):
        self._date = date
        self._close = close
        self._volume = volume

    @property
    def date(self):
        return self._date

    @date.setter
    def date(self, date):
        self._date = date

    @property
    def close(self):
        return self._close
    @close.setter
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

```
hon /Users/mmastache/Python/inStock/stock_class.py
Unit Testing Starting---
Testing Add Stock...Successful!
Testing Change Symbol...Successful! - Stock symbol change blocked
Test Change Name...Successful!
Test Change Shares...Successful! - Stock shares change blocked
Test Buy shares...Successful!
Test Sell shares...Successful!
Creating daily stock data...Successful!
Congratulations - All Tests Passed
Goodbye
(venv) mmastache@mac inStock %
```

Python
Python
zsh

main* 0↓ 1↑ ⊗ 0 ⚠ 0 Live Share UTF-8 CRLF Python 3.10.7 ('venv': venv)

Console Interface

This section covers the creation of console-based interface for working with stocks and the stock price history. It develops the code for adding stock, and its daily data.

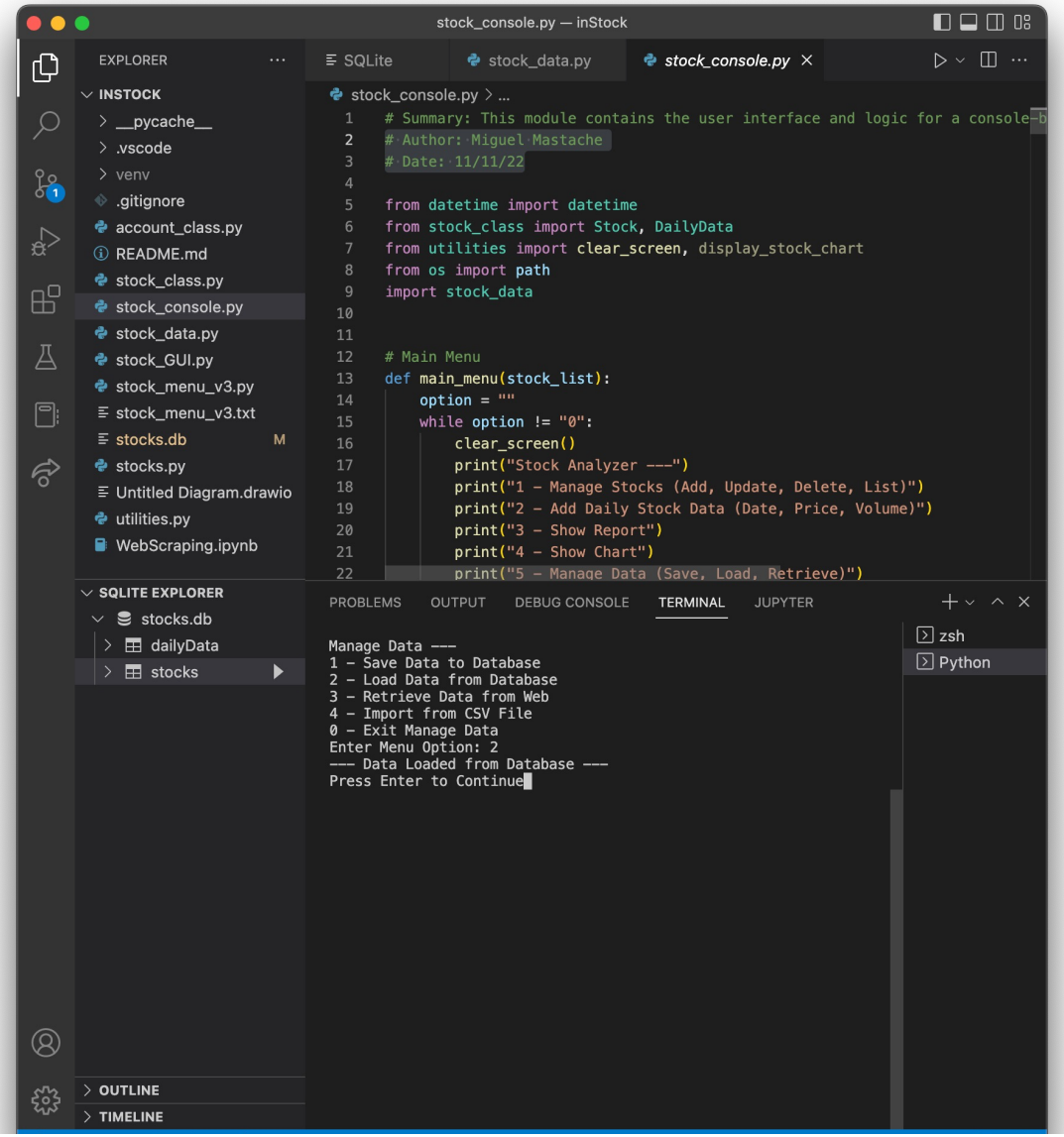
Report

- Paste a screen shot of your working Stock Report.

```
python
0 - Exit Program
Enter Menu Option: 3
Stock Report ---
Report for:  F Ford
Shares:  100.0
11/11/22 50.0 50.0
11/12/22 60.0 60.0
Summary --- 11/11/22 to 11/12/22
Low Price: $50.00
High Price: $60.00
Average Price: $55.00
Low Volume: 50.0
High Volume: 60.0
Average Volume: 55.0
Starting Price: $50.00
Ending Price: $60.00
Change in Price: $10.00
Profit/Loss:$1,000.00
Report for:  GM General Motors
Shares:  100.0
11/11/22 70.0 70.0
11/12/22 70.7 80.0
Summary --- 11/11/22 to 11/12/22
Low Price: $70.00
High Price: $70.70
Average Price: $70.35
Low Volume: 70.0
High Volume: 80.0
Average Volume: 75.0
Starting Price: $70.00
Ending Price: $70.70
Change in Price: $0.70
Profit/Loss:$70.00
Report for:  TSLA Tesla
Shares:  100.0
11/11/22 99.99 100.0
11/12/22 700.0 1.0
Summary --- 11/11/22 to 11/12/22
Low Price: $99.99
High Price: $700.00
Average Price: $400.00
Low Volume: 1.0
High Volume: 100.0
Average Volume: 50.5
Starting Price: $99.99
Ending Price: $700.00
Change in Price: $600.01
Profit/Loss:$60,001.00
Report Complete
*** Press Enter to Continue ***
```

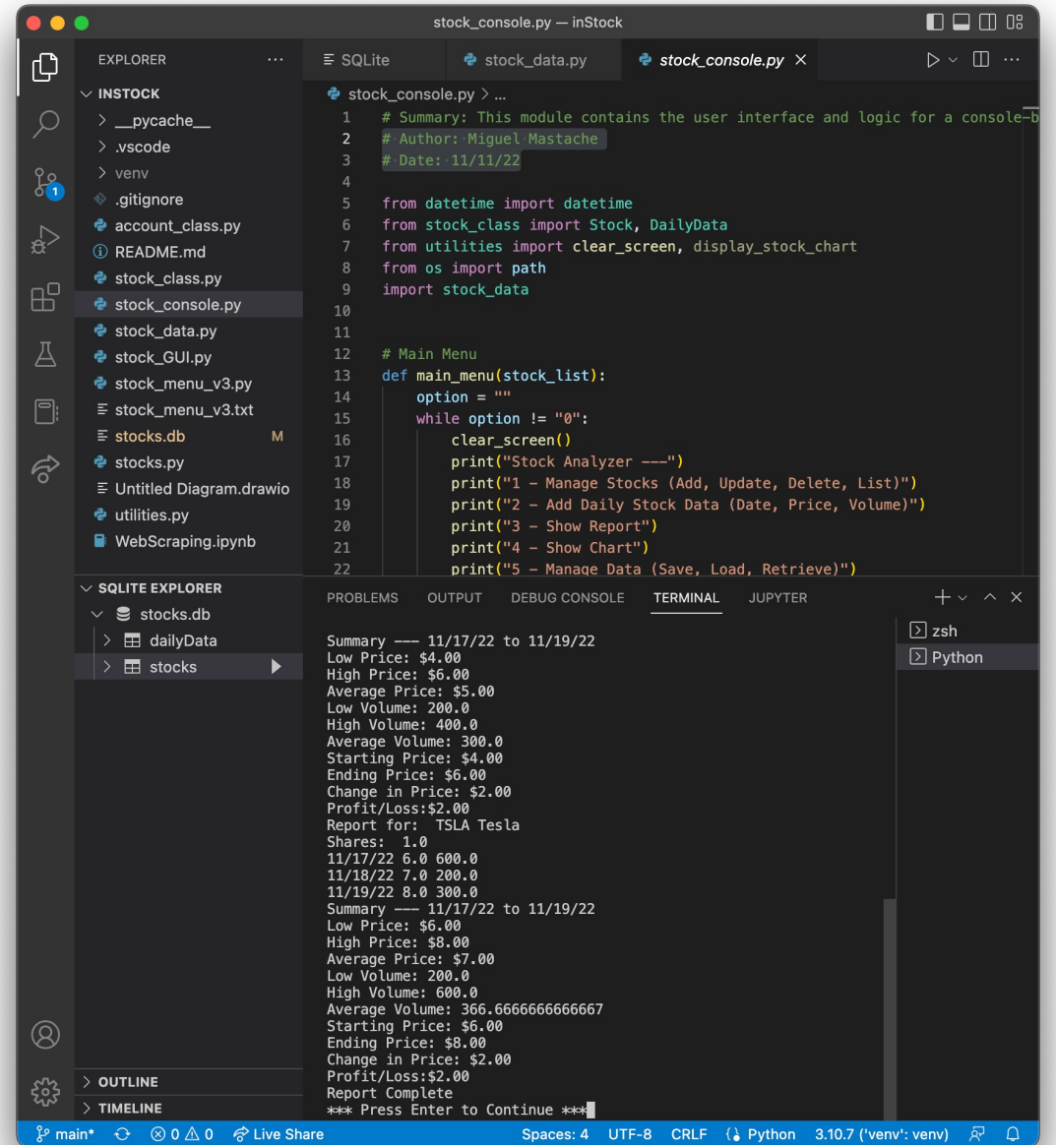
Load Database

- Paste a screen shot of your Data Loaded from Database message.



Report

- Paste a screen shot of your working Stock Report.

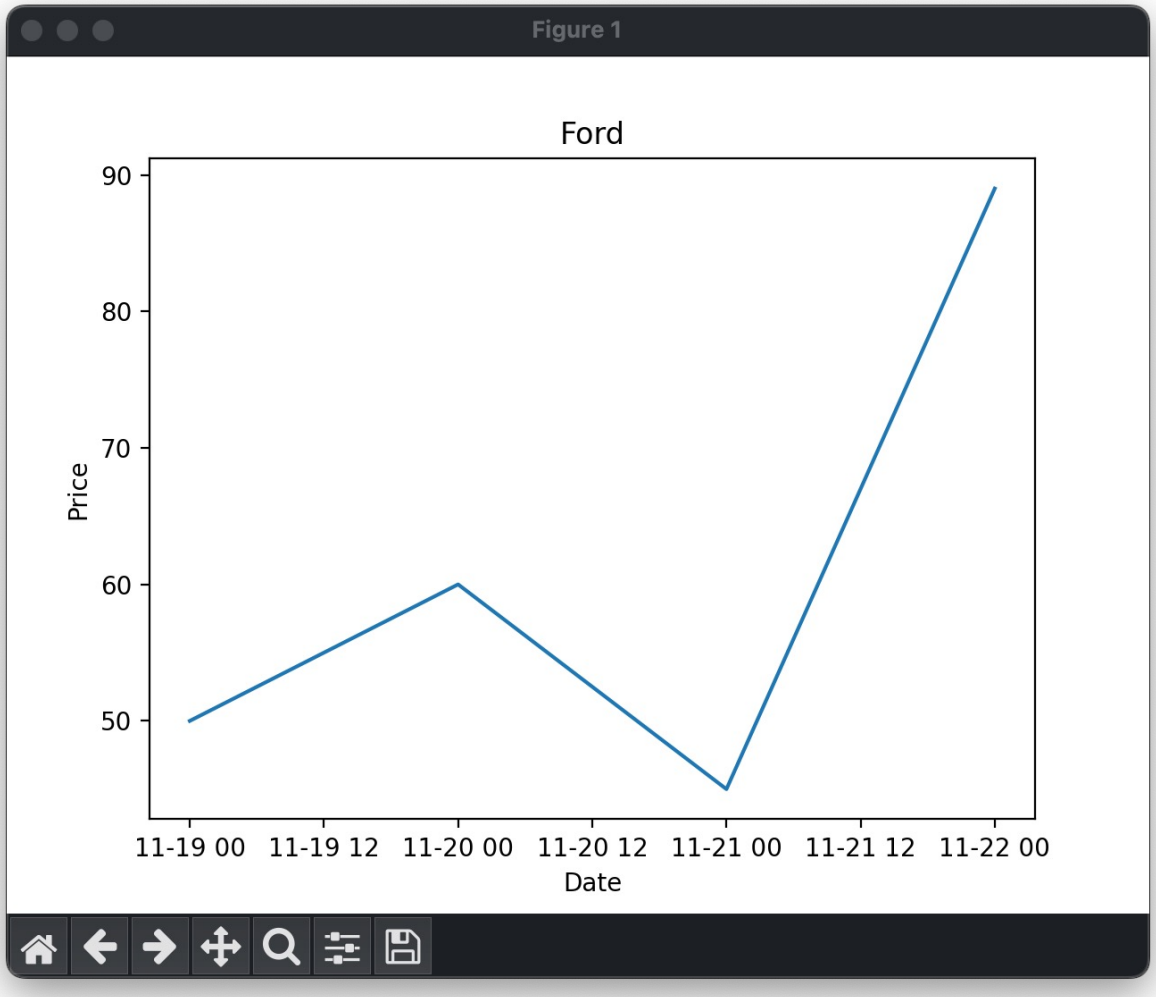


Charts

In this section, we use the matplotlib library to display charts for the stock data.

Optional Chart

- Paste a screen shot of your stock chart.



File Processing

This section covers file processing, by developing file saving code, and importing of saved data in csv format.

Code

stock_menu_v3.py — inStock

EXPLORER

INSTOCK

- __pycache__
- .vscode
- venv
- .gitignore
- account_class.py
- AFRM.csv U
- README.md
- stock_class.py
- stock_console.py
- stock_data.py
- stock_GUI.py
- stock_menu_v3.py M
- stock_menu_v3.txt
- stocks.db
- stocks.py
- Untitled Diagram.drawio
- utilities.py
- WebScraping.ipynb

OUTLINE

TIMELINE

stock_menu_v3.py M | stock_console.py

stock_menu_v3.py > add_stock

```python
@author: D99003734
"""
# Author: Miguel Mastache
# Date: 12/3/22
from datetime import datetime
from stock_class import Stock, DailyData
from account_class import  Traditional, Robo
import matplotlib.pyplot as plt
import csv


def add_stock(stock_list):
    option = ""
    while option != "0":
        print("Add stock ---")
        symbol = input("Enter Ticker Symbol: ").upper()
        name = input("Enter Company Name: ")
        shares = float(input("Enter Number of Shares: "))
        new_stock = Stock(symbol, name, shares)
        stock_list.append(new_stock)
        option = input("Stock Added - Enter to Add Another Stock or 0 to Stop: ")
```

PROBLEMS | OUTPUT | DEBUG CONSOLE | TERMINAL | JUPYTER

zsh | Python

```
Profit/Loss:$115.87
2022-12-01 14.41 14360500.0
Low Price: $12.10
High Price: $127.97
Average Price: $37.77
Low Volume:  999999999999
Highe Volume:  86940400.0
Average Volume: 14323483.266932271
Change in Price: $115.87
Profit/Loss:$115.87
2022-12-02 14.62 10800800.0
Low Price: $12.10
High Price: $127.97
Average Price: $37.68
Low Volume:  999999999999
Highe Volume:  86940400.0
Average Volume: 14309504.365079366
Change in Price: $115.87
Profit/Loss:$115.87
**** No daily history



Report for:  TSLA TESLA
Shares:  1.0
**** No daily history



---Report complete ----
Press Enter to Continue ***
```

main* | 0 0 | Live Share | Ln 18, Col 25 | Spaces: 4 | UTF-8 | CRLF | Python | 3.10.7 ('venv': venv)

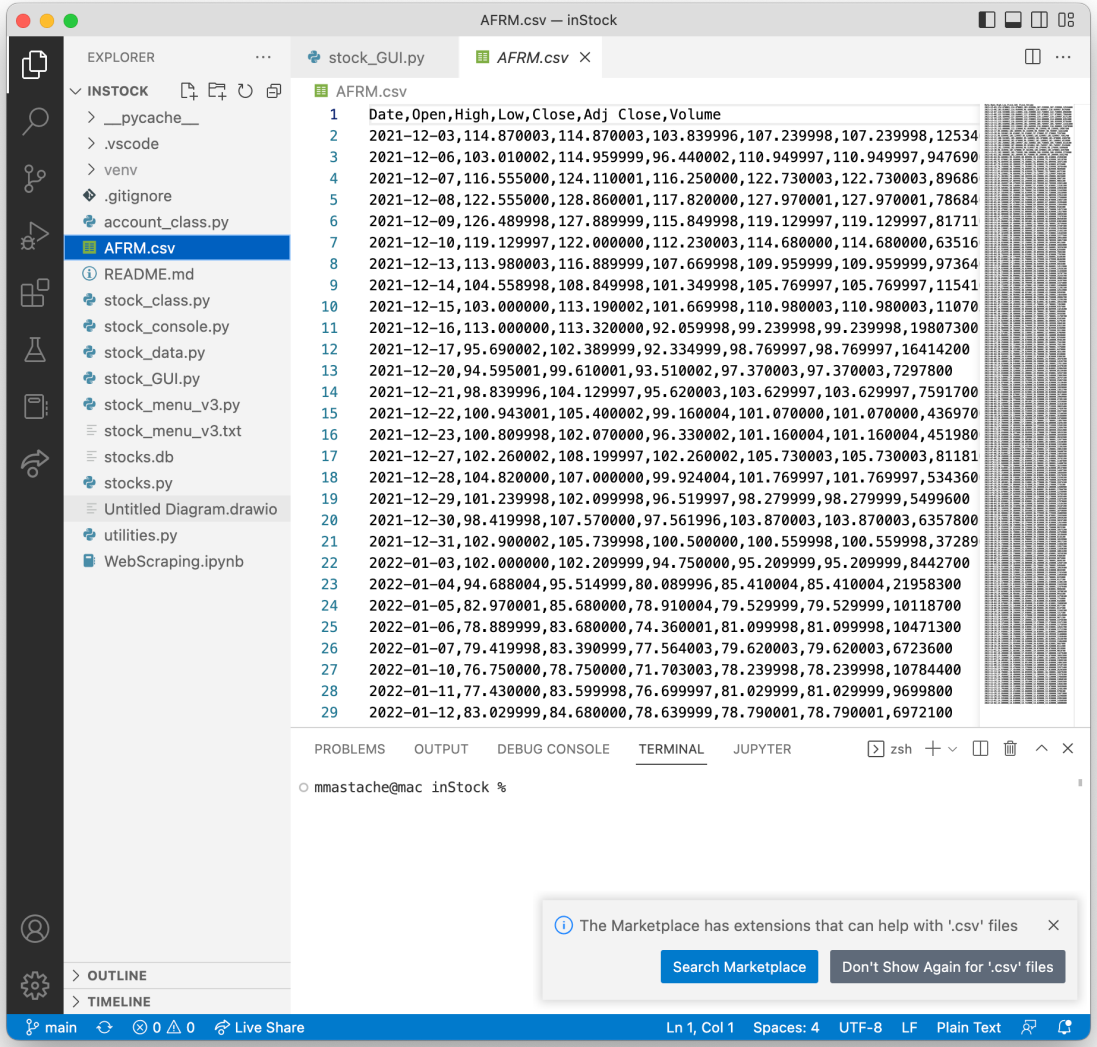
File

- Paste a screen shot of the file downloaded from Yahoo finance

```
-zsh
-rw-r--r--@  1 mmastache  staff   9568 Dec  3 18:55 stock_menu_v3.py
-rw-r--r--@  1 mmastache  staff   2570 Nov  9 20:36 stock_menu_v3.txt
-rw-r--r--   1 mmastache  staff  20480 Nov 22 13:16 stocks.db
-rw-r--r--   1 mmastache  staff    422 Nov  5 17:34 stocks.py
-rw-r--r--   1 mmastache  staff   1162 Nov 22 13:05 utilities.py
drwxr-xr-x   7 mmastache  staff    224 Oct 29 09:50 venv
mmastache@mac inStock % ls -l
total 248
-rw-r--r--@ 1 mmastache  staff  17683 Dec  3 08:08 AFRM.csv
-rw-r--r--  1 mmastache  staff     10 Oct 29 09:07 README.md
-rw-r--r--  1 mmastache  staff   2339 Nov  5 17:34 Untitled Diagram.drawio
-rw-r--r--  1 mmastache  staff   5910 Nov  5 17:34 WebScraping.ipynb
drwxr-xr-x  6 mmastache  staff    192 Nov 22 13:13 __pycache__
-rw-r--r--  1 mmastache  staff   4740 Nov 15 20:21 account_class.py
-rw-r--r--  1 mmastache  staff   7746 Nov  5 17:34 stock_GUI.py
-rw-r--r--  1 mmastache  staff   6806 Nov 22 11:43 stock_class.py
-rw-r--r--  1 mmastache  staff  13183 Nov 22 13:18 stock_console.py
-rw-r--r--  1 mmastache  staff   4029 Nov 19 19:17 stock_data.py
-rw-r--r--@ 1 mmastache  staff   9568 Dec  3 18:55 stock_menu_v3.py
-rw-r--r--@ 1 mmastache  staff   2570 Nov  9 20:36 stock_menu_v3.txt
-rw-r--r--  1 mmastache  staff  20480 Nov 22 13:16 stocks.db
-rw-r--r--  1 mmastache  staff    422 Nov  5 17:34 stocks.py
-rw-r--r--  1 mmastache  staff   1162 Nov 22 13:05 utilities.py
drwxr-xr-x  7 mmastache  staff    224 Oct 29 09:50 venv
mmastache@mac inStock %
```

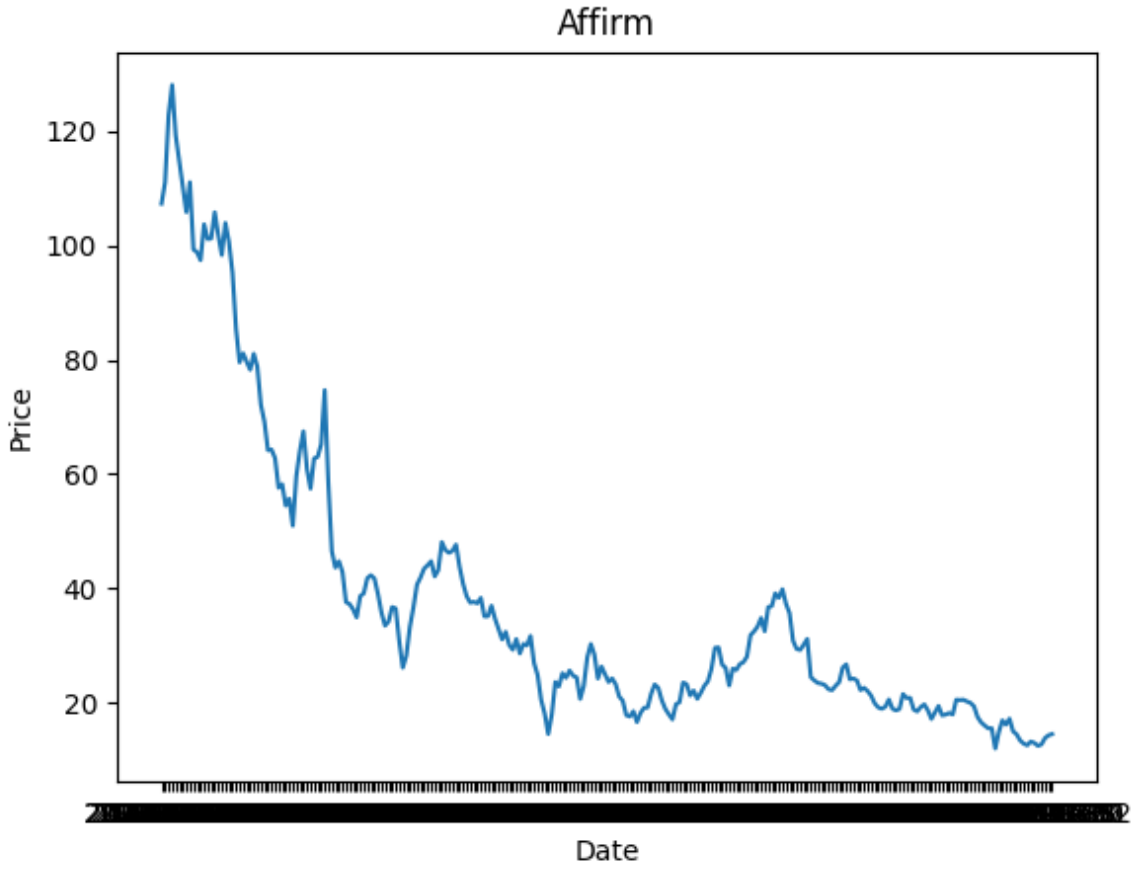
File

- Paste a screen shot of the file downloaded from Yahoo finance



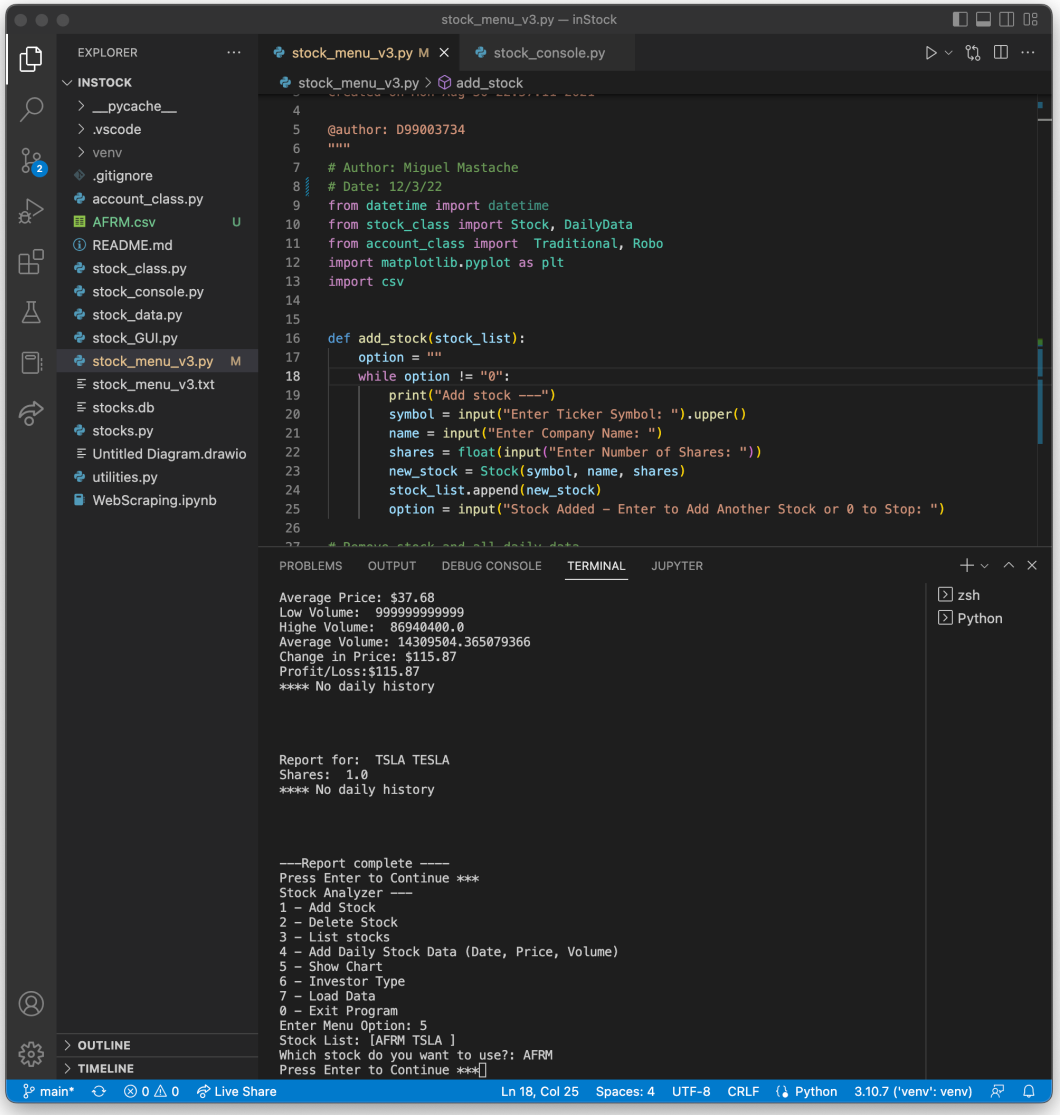
Importing data

- Screenshot of the historical data import



Importing data

- Screenshot of the historical data import

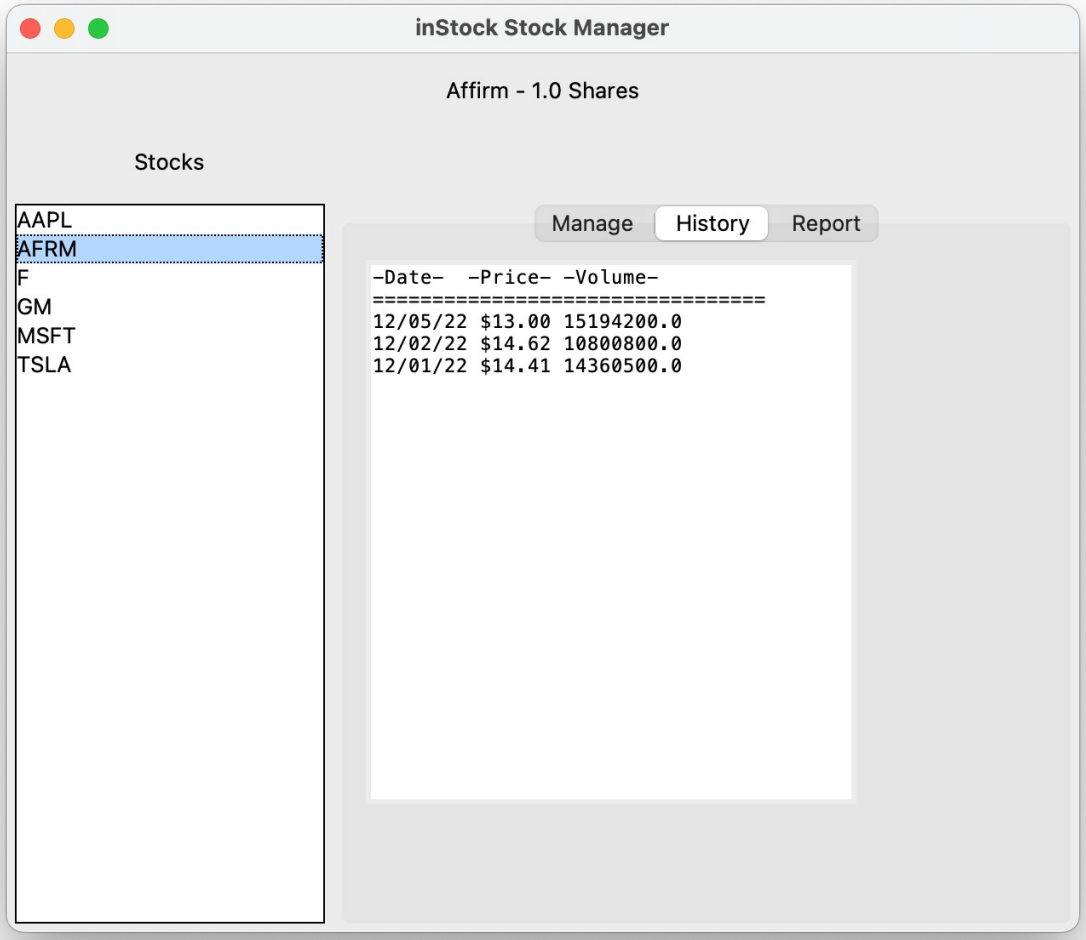


GUI

This section covers the development of the graphical user interface for the stock tracking application.

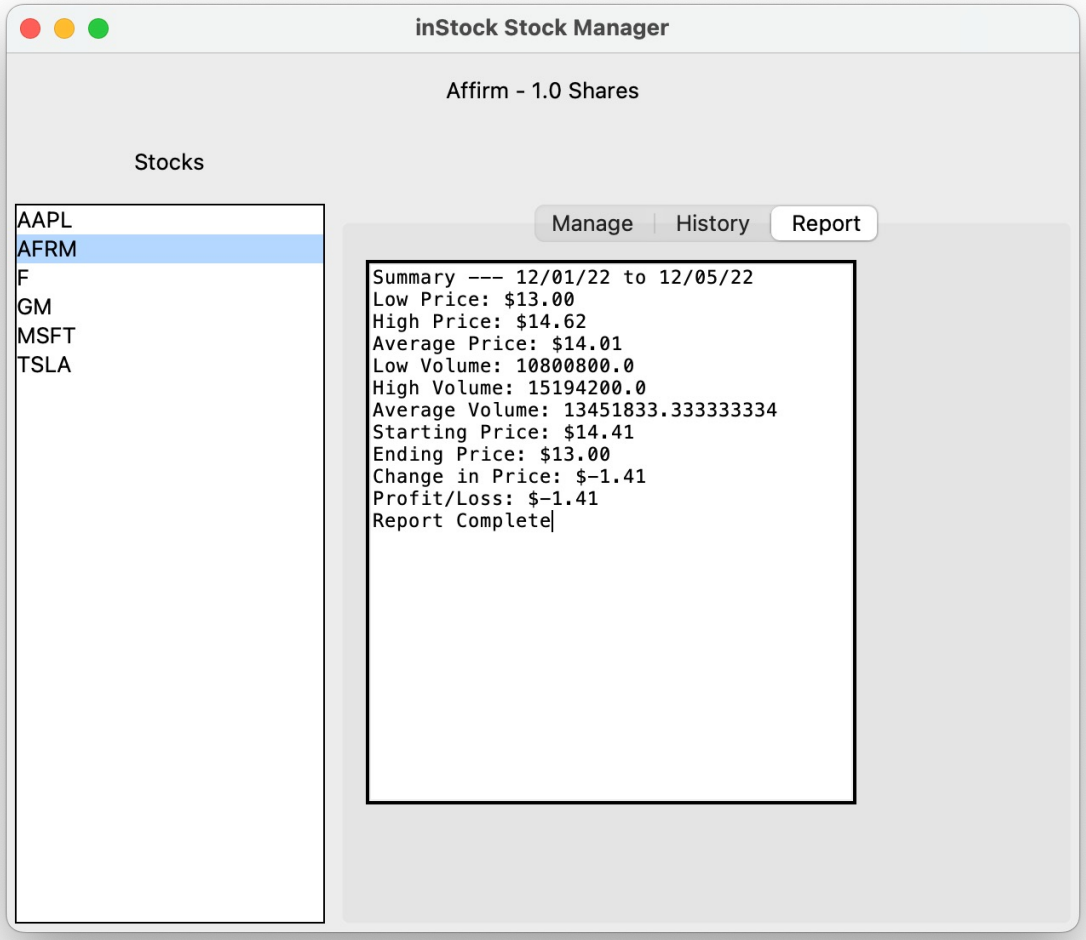
History Tab

- Paste a screen shot of your history tab.



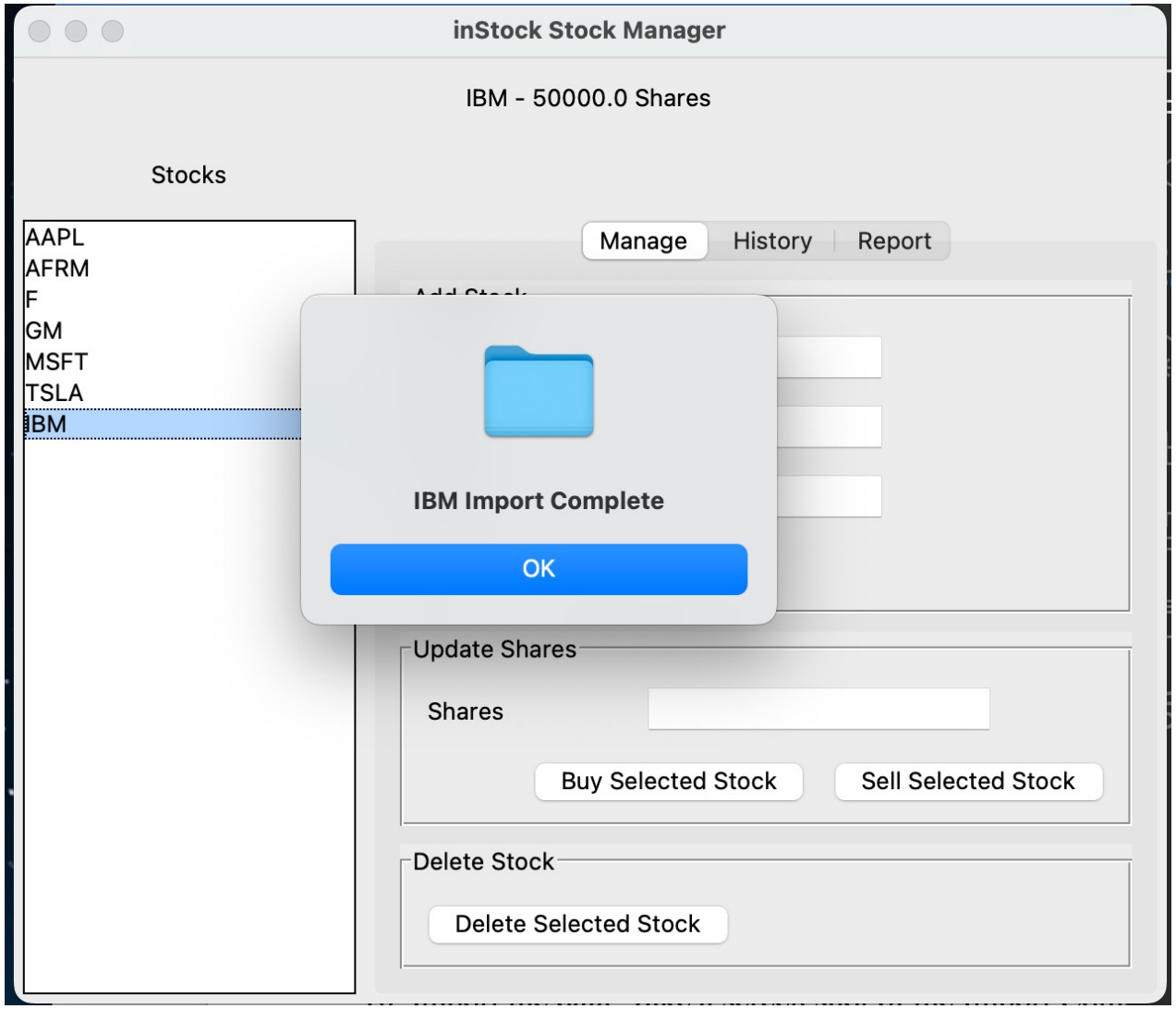
Report Tab

- Paste a screen shot of your report tab.



Import Complete

- Paste a screen shot of your Import Complete confirmation message.



Challenges

A few challenges were faced in the development of this project.
First and foremost, I am not a fan of the limitations imposed by Anaconda. I prefer to use a clean version of python without the conda package. Therefore, I had to learn how to use Visual Studio Code, venv, and pip. Additionally, keeping track of all the installation packages proved to be challenging, but using virtual environments helped address this.

Career skills developed

I became familiar with the use of pyenv on macOS.

Used virtual environments using venv.

Used pip to manage python packages.

Became familiar with Visual Studio Code.

Used object oriented programming.

Became familiar with gui development using tkinter.

Conclusion

Development of this app, allowed me to see how different pieces of software come together and work together to develop a modern gui based application. Additionally, introduced me to further areas of pursuit, in particular web scraping, database processing, and gui development.