inStock

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.

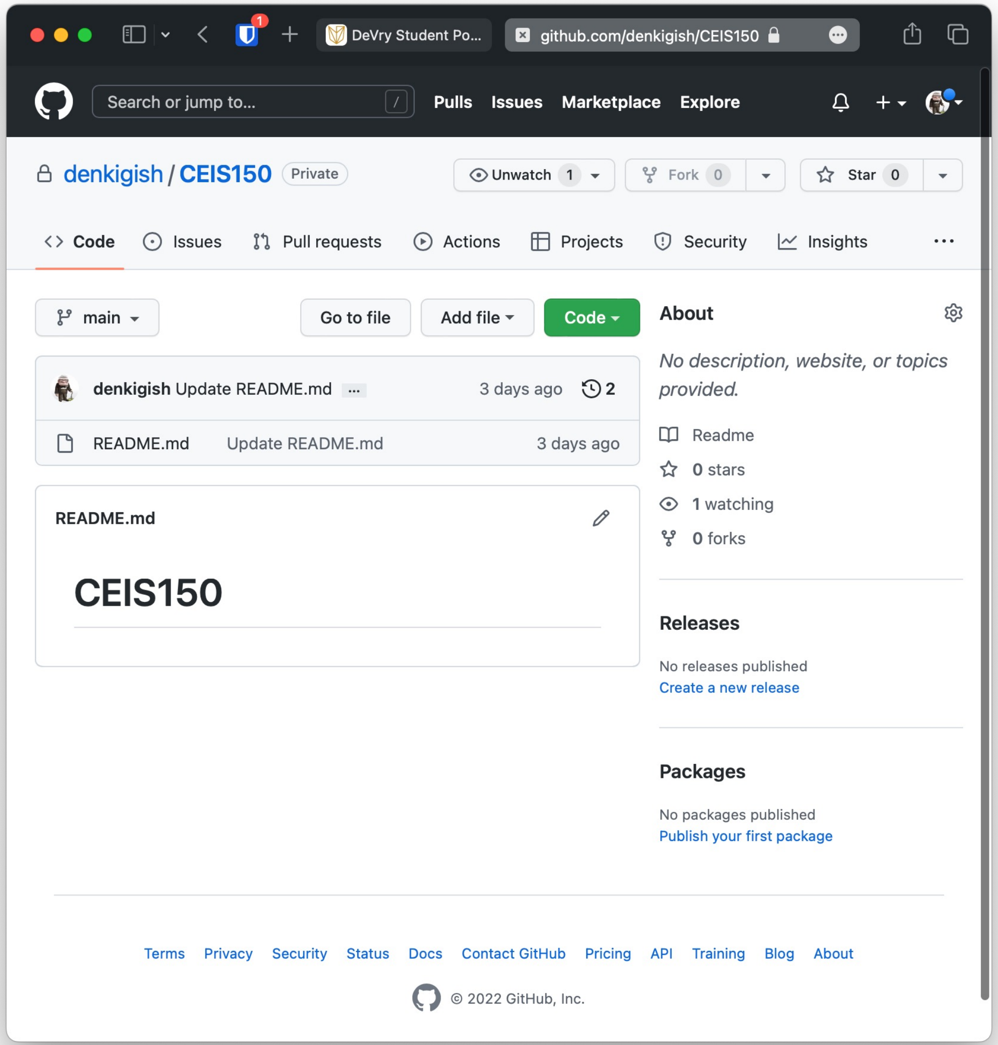
Environment

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



denkigish Update README.md ... 3 days ago 2

README.md Update README.md 3 days ago

README.md

CEIS150

About

No description, website, or topics provided.

Readme
0 stars
1 watching
0 forks

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

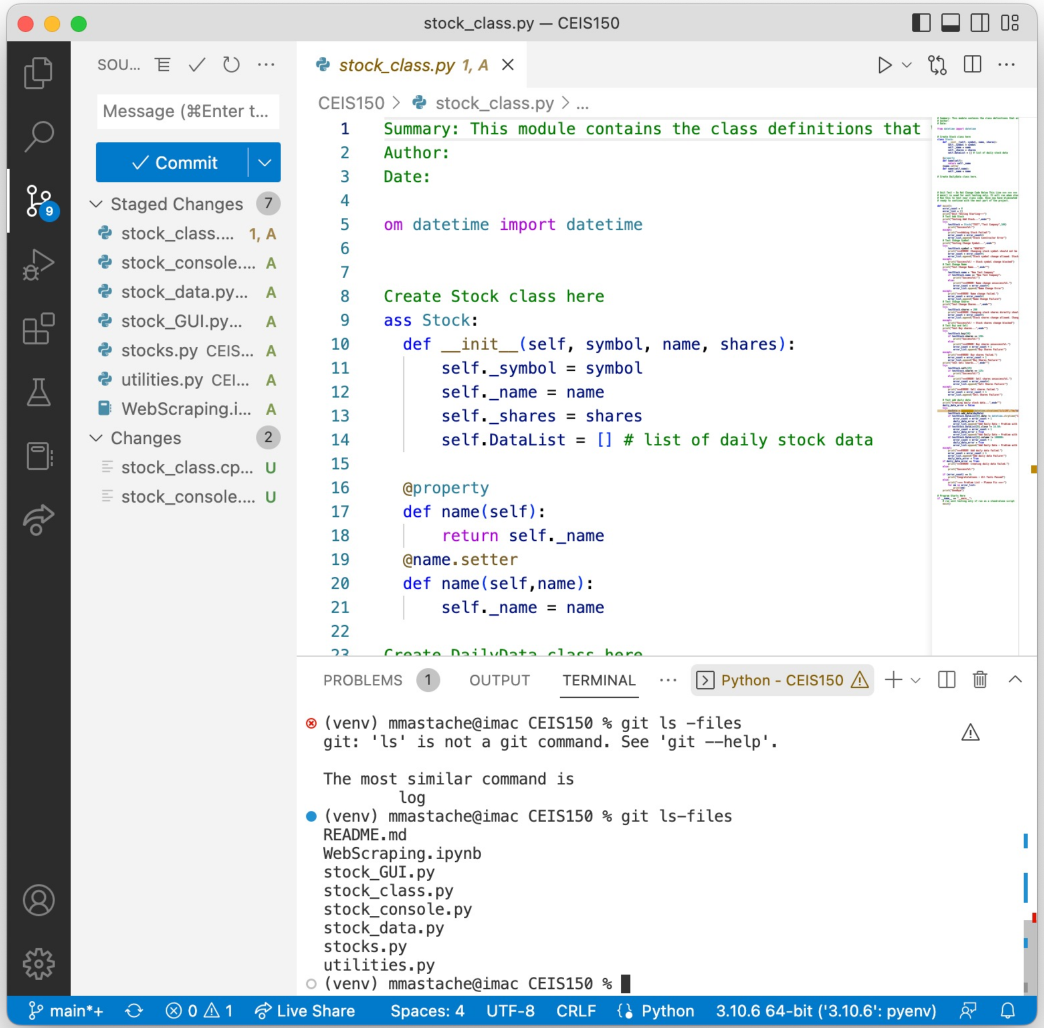
Terms Privacy Security Status Docs Contact GitHub Pricing API Training Blog About

© 2022 GitHub, Inc.

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.

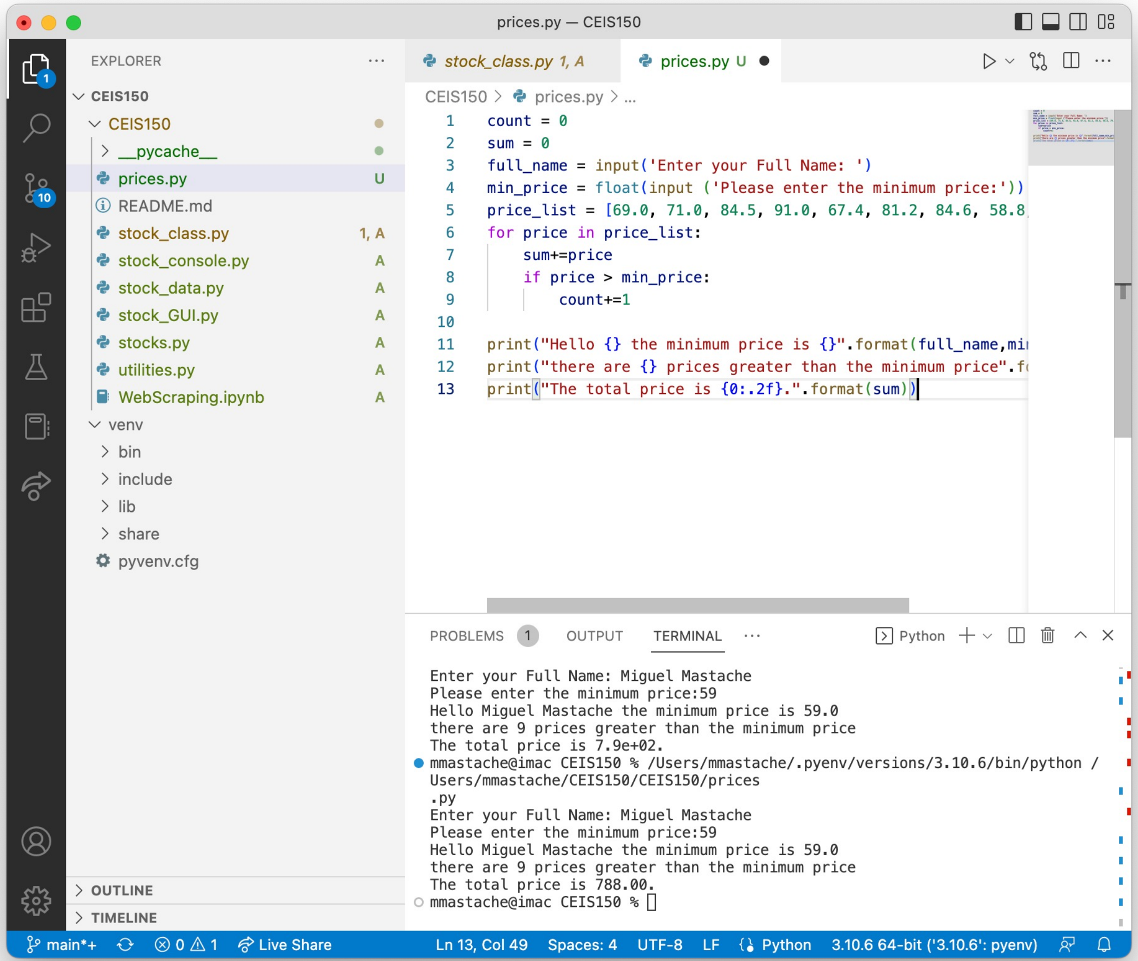


```
Summary: This module contains the class definitions that
Author:
Date:

om datetime import datetime


Create Stock class here
ass 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 name(self):
        return self._name
    @name.setter
    def name(self,name):
        self._name = name

Create DailyData class here
```

```
(venv) mmastache@imac CEIS150 % git ls -files
git: 'ls' is not a git command. See 'git --help'.

The most similar command is
        log
(venv) mmastache@imac CEIS150 % git ls-files
README.md
WebScraping.ipynb
stock_GUI.py
stock_class.py
stock_console.py
stock_data.py
stocks.py
utilities.py
(venv) mmastache@imac CEIS150 %
```

Program

- Screen shot of Python program running successfully.
- prices.py working with price list, summing, and counting stocks above minimum price.



```python
count = 0
sum = 0
full_name = input('Enter your Full Name: ')
min_price = float(input ('Please enter the minimum price:'))
price_list = [69.0, 71.0, 84.5, 91.0, 67.4, 81.2, 84.6, 58.8
for price in price_list:
    sum+=price
    if price > min_price:
        count+=1

print("Hello {} the minimum price is {}".format(full_name,mi
print("there are {} prices greater than the minimum price".f
print("The total price is {0:.2f}.".format(sum))
```

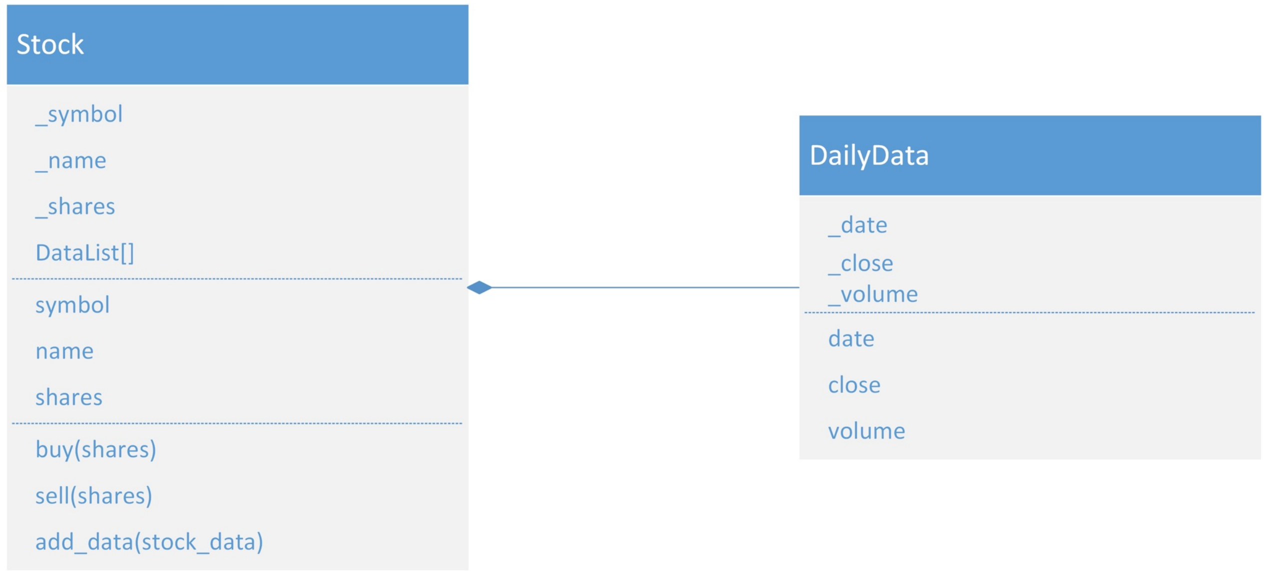
```
Enter your Full Name: Miguel Mastache
Please enter the minimum price:59
Hello Miguel Mastache the minimum price is 59.0
there are 9 prices greater than the minimum price
The total price is 7.9e+02.
mmastache@imac CEIS150 % /Users/mmastache/.pyenv/versions/3.10.6/bin/python /
Users/mmastache/CEIS150/CEIS150/prices
.py
Enter your Full Name: Miguel Mastache
Please enter the minimum price:59
Hello Miguel Mastache the minimum price is 59.0
there are 9 prices greater than the minimum price
The total price is 788.00.
mmastache@imac CEIS150 %
```

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.

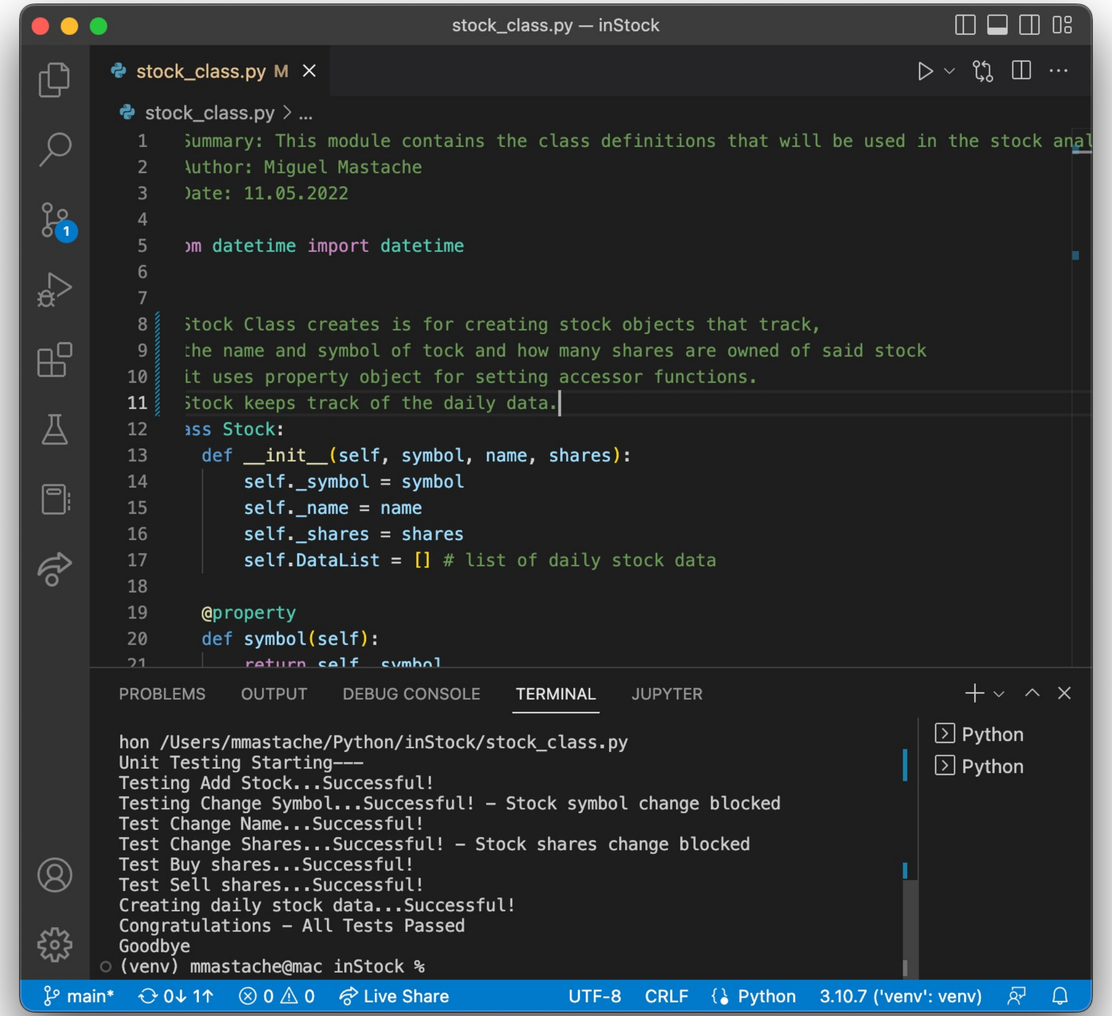
Class Diagram

- Paste your Visio Class Diagram



Class Code

- Screen Shot of your stock_class.py file.

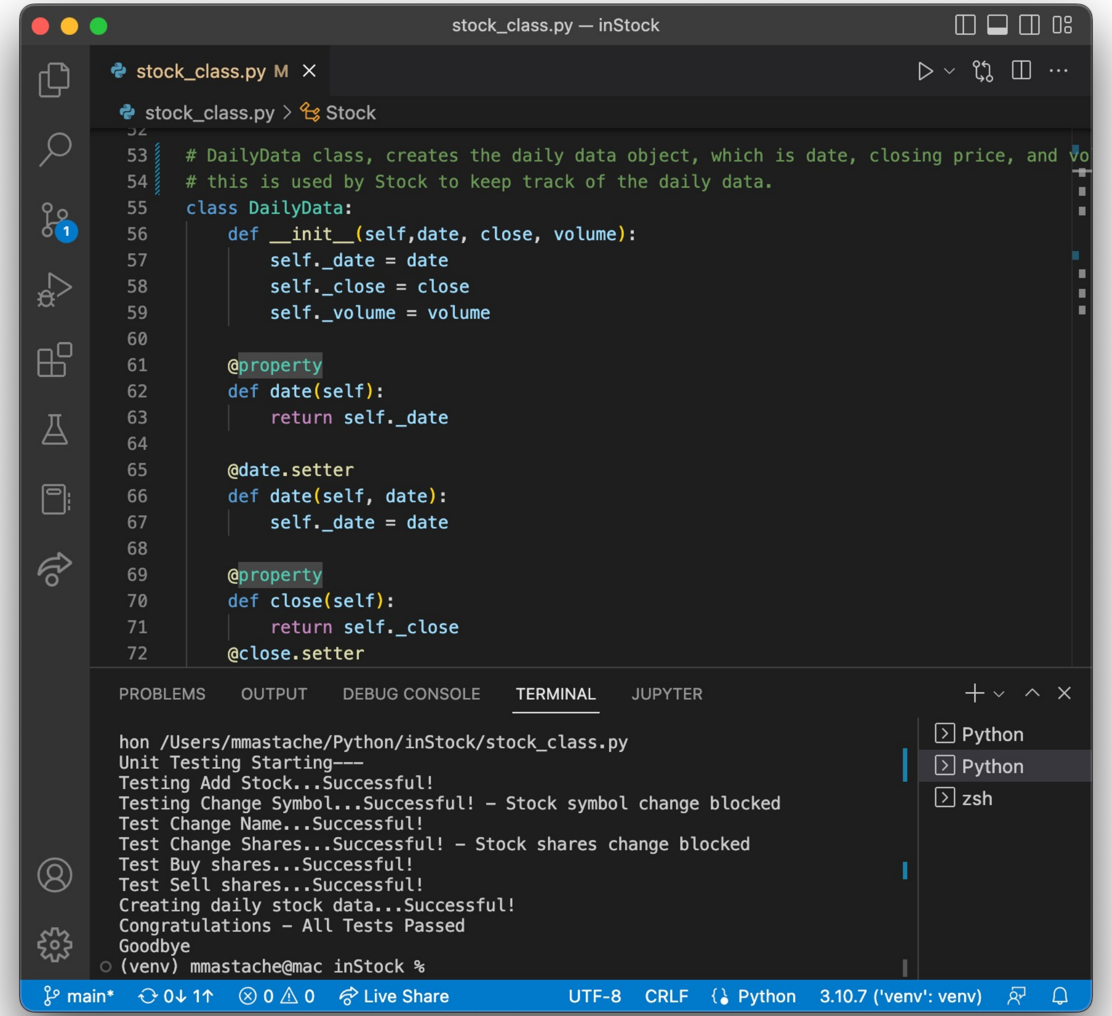


```
ummary: This module contains the class definitions that will be used in the stock anal
uthor: Miguel Mastache
ate: 11.05.2022

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
t 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
```

```
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 %
```

Unit Test

- Screen Shot of your successful unit test.


stock_class.py — inStock
stock_class.py M
stock_class.py > Stock
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER
Python
Python
zsh
main* 0↓ 1↑ 0 0 Live Share UTF-8 CRLF Python 3.10.7 ('venv': venv)

```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
```

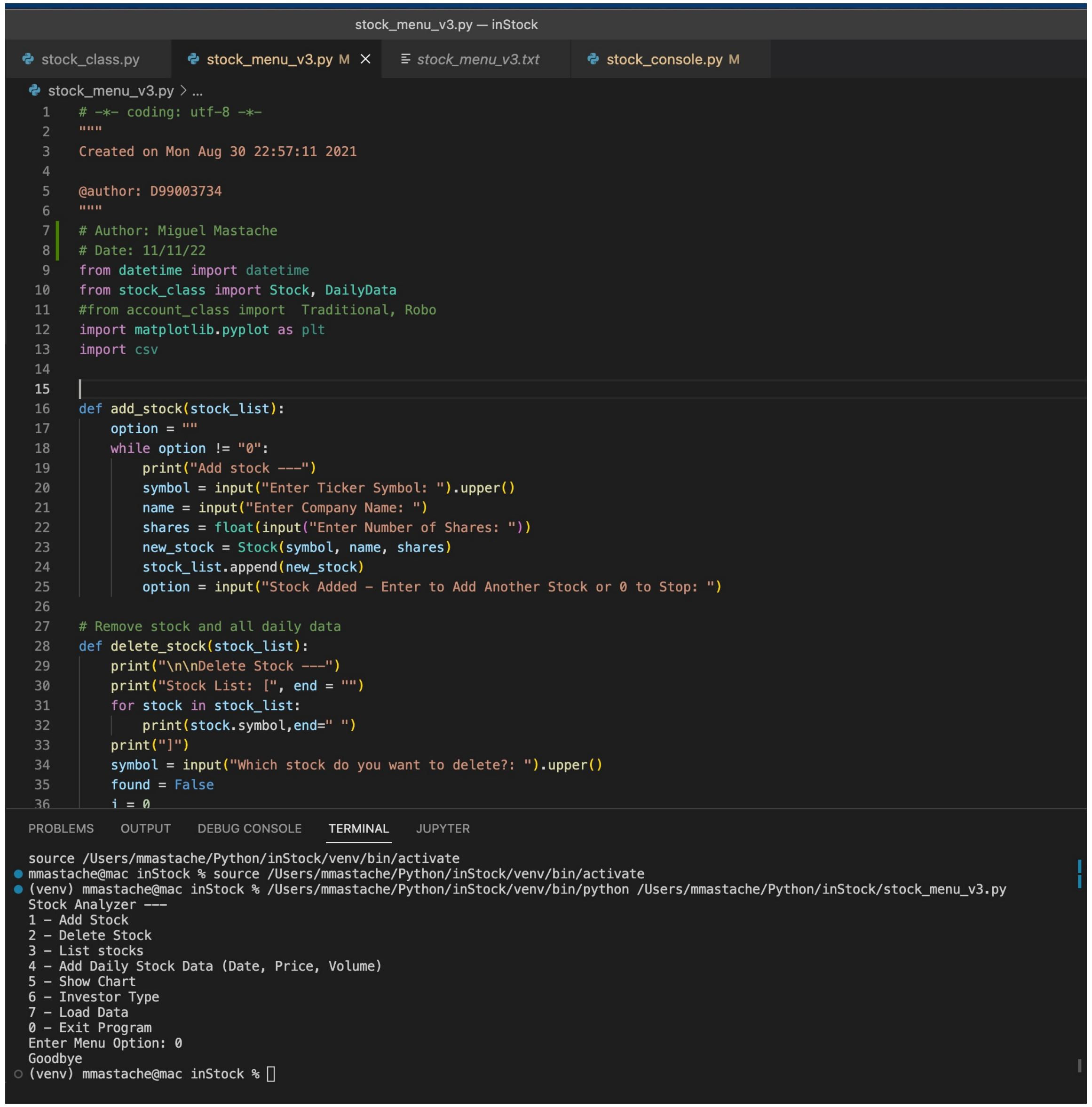
```
hon /Users/mmastache/Python/inStock/stock_class.py
Unit Testing Starting---
Testing Add Stock...Successful!
Testing Change Symbol...Successful! - Stock symbol change blocked
Test Change Name...Successful!
Test Change Shares...Successful! - Stock shares change blocked
Test Buy shares...Successful!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

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.

Code

- Code snippet.

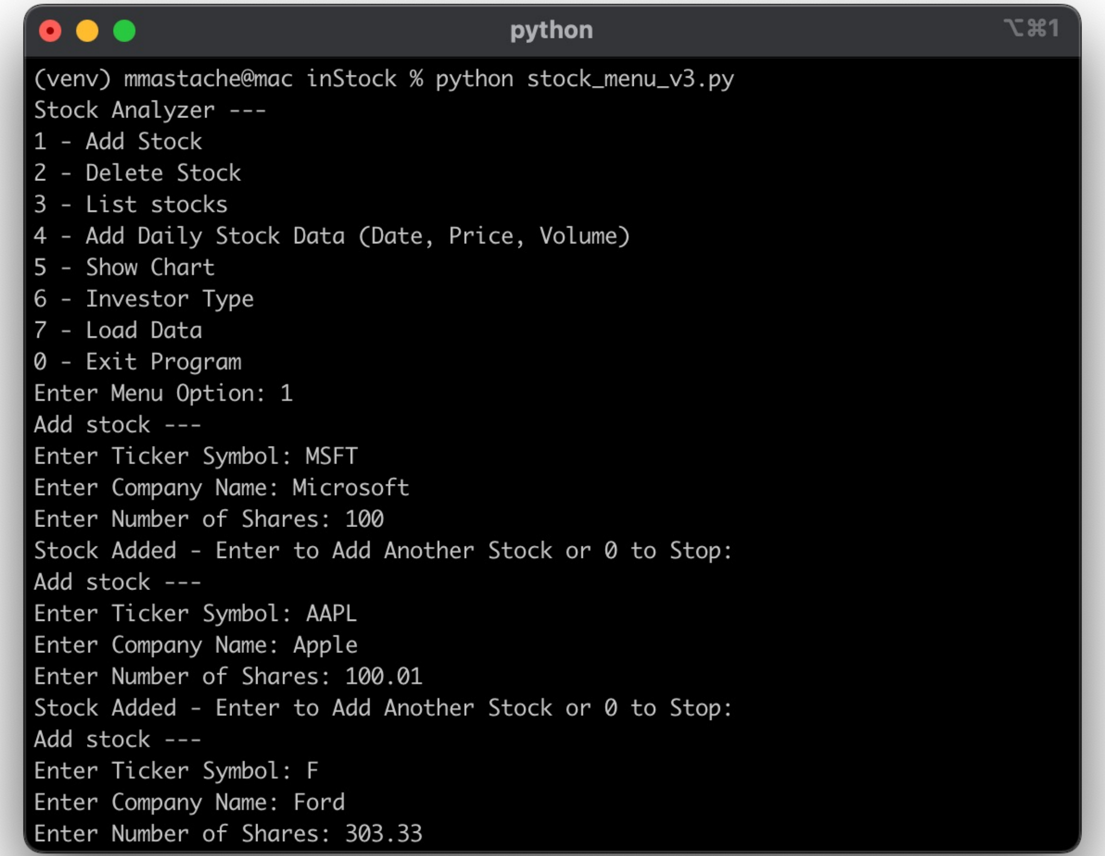


```python
# -*- coding: utf-8 -*-
"""
Created on Mon Aug 30 22:57:11 2021

@author: D99003734
"""
# Author: Miguel Mastache
# Date: 11/11/22
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: ")

# Remove stock and all daily data
def delete_stock(stock_list):
    print("\n\nDelete Stock ---")
    print("Stock List: [", end = "")
    for stock in stock_list:
        print(stock.symbol,end=" ")
    print("]")
    symbol = input("Which stock do you want to delete?: ").upper()
    found = False
    i = 0
```

```
source /Users/mmastache/Python/inStock/venv/bin/activate
mmastache@mac inStock % source /Users/mmastache/Python/inStock/venv/bin/activate
(venv) mmastache@mac inStock % /Users/mmastache/Python/inStock/venv/bin/python /Users/mmastache/Python/inStock/stock_menu_v3.py
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: 0
Goodbye
(venv) mmastache@mac inStock %
```

Adding a Stock

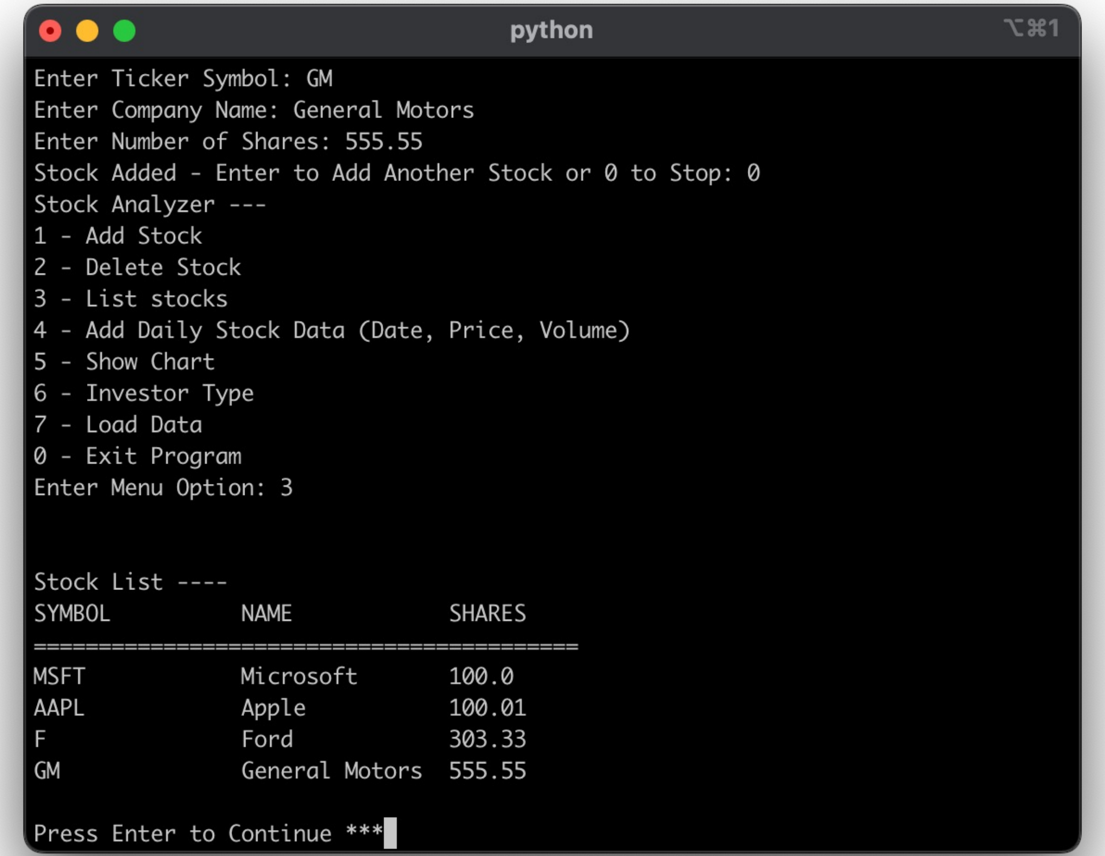
- Paste a screen shot of your working Stock program.



```
(venv) mmastache@mac inStock % python stock_menu_v3.py
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: 1
Add stock ---
Enter Ticker Symbol: MSFT
Enter Company Name: Microsoft
Enter Number of Shares: 100
Stock Added - Enter to Add Another Stock or 0 to Stop:
Add stock ---
Enter Ticker Symbol: AAPL
Enter Company Name: Apple
Enter Number of Shares: 100.01
Stock Added - Enter to Add Another Stock or 0 to Stop:
Add stock ---
Enter Ticker Symbol: F
Enter Company Name: Ford
Enter Number of Shares: 303.33
```

Listing 3 Stocks

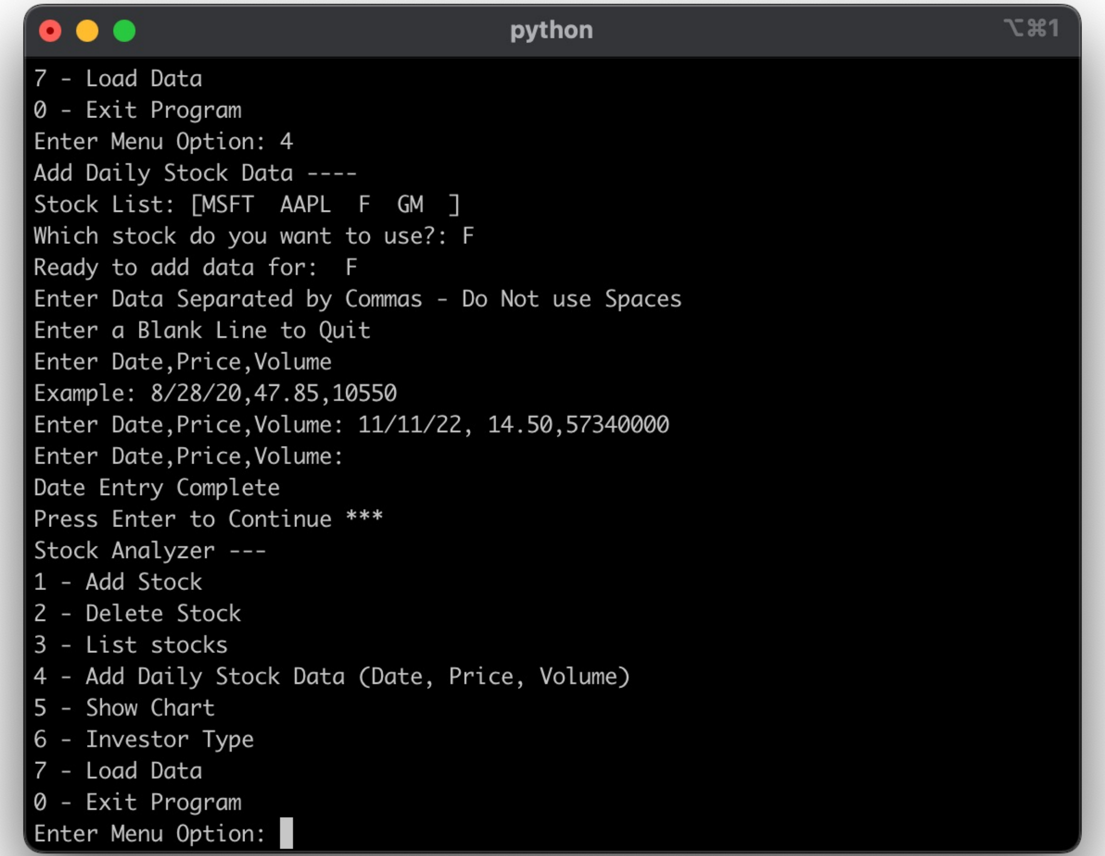
- Paste a screen shot of your working Stock program.



```
Enter Ticker Symbol: GM
Enter Company Name: General Motors
Enter Number of Shares: 555.55
Stock Added - Enter to Add Another Stock or 0 to Stop: 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       100.0
AAPL            Apple           100.01
F               Ford            303.33
GM              General Motors  555.55

Press Enter to Continue ***
```

Daily Data

- Paste a screen shot of your working Stock program.



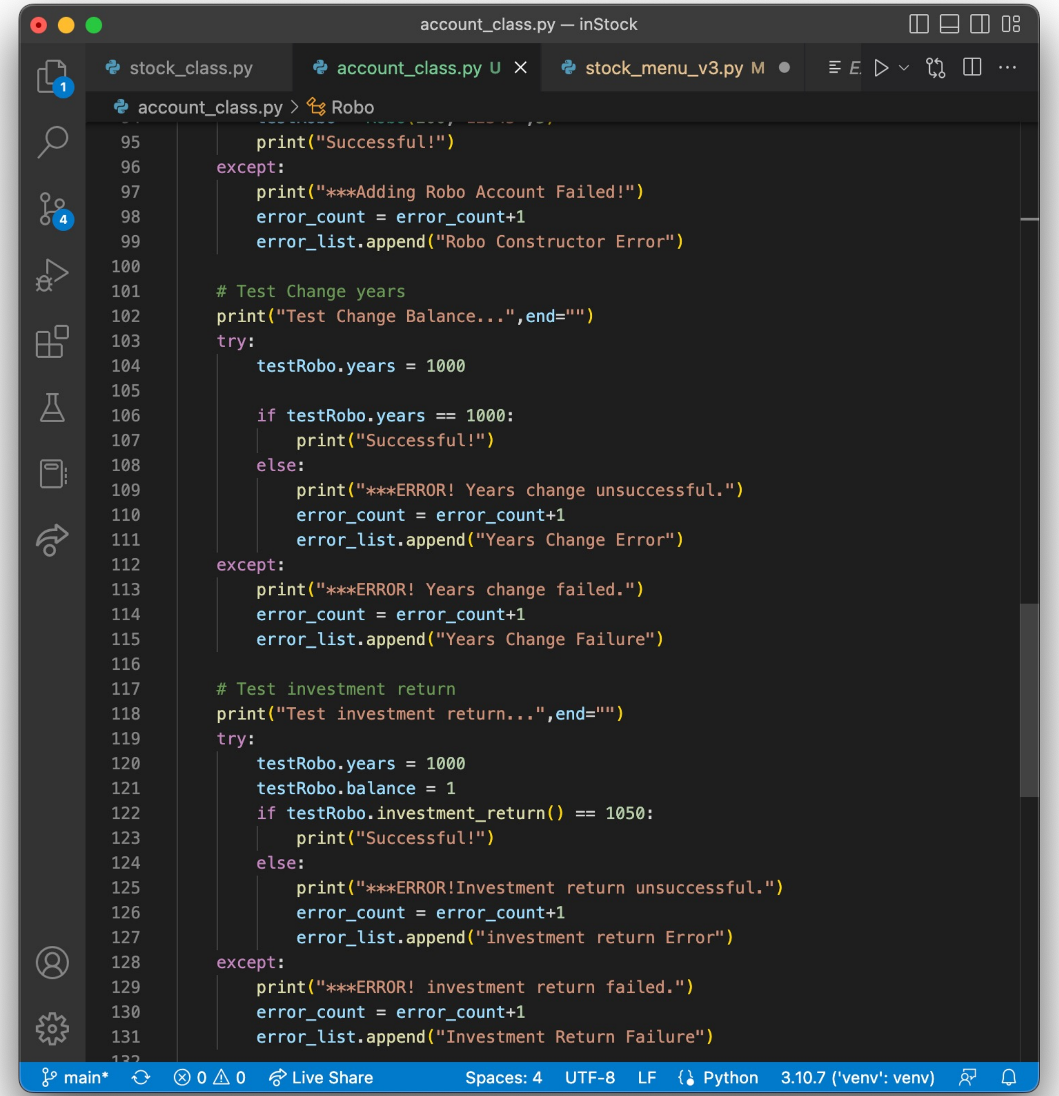
```
7 - Load Data
0 - Exit Program
Enter Menu Option: 4
Add Daily Stock Data ----
Stock List: [MSFT  AAPL  F  GM  ]
Which stock do you want to use?: F
Ready to add data for:  F
Enter Data Separated by Commas - Do Not use Spaces
Enter a Blank Line to Quit
Enter Date,Price,Volume
Example: 8/28/20,47.85,10550
Enter Date,Price,Volume: 11/11/22, 14.50,57340000
Enter Date,Price,Volume:
Date Entry Complete
Press Enter to Continue ***
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:
```

Inherited Classes

This part of the project uses inheritance to develop different account types. And a menu interface for the console.

Inherited classes

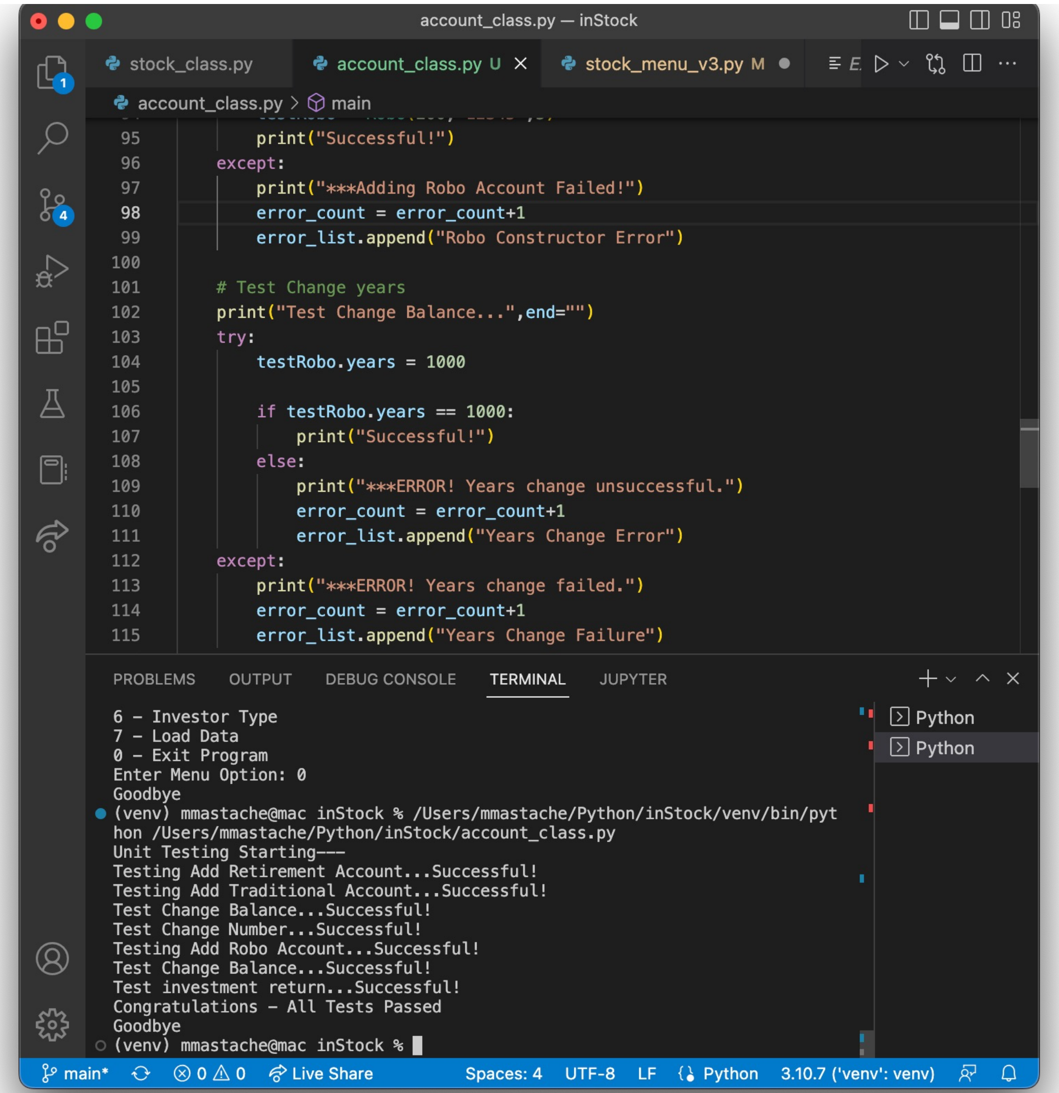
- Paste a screen shot of your classes



```
        print("Successful!")
    except:
        print("***Adding Robo Account Failed!")
        error_count = error_count+1
        error_list.append("Robo Constructor Error")

    # Test Change years
    print("Test Change Balance...",end="")
    try:
        testRobo.years = 1000

        if testRobo.years == 1000:
            print("Successful!")
        else:
            print("***ERROR! Years change unsuccessful.")
            error_count = error_count+1
            error_list.append("Years Change Error")
    except:
        print("***ERROR! Years change failed.")
        error_count = error_count+1
        error_list.append("Years Change Failure")

    # Test investment return
    print("Test investment return...",end="")
    try:
        testRobo.years = 1000
        testRobo.balance = 1
        if testRobo.investment_return() == 1050:
            print("Successful!")
        else:
            print("***ERROR!Investment return unsuccessful.")
            error_count = error_count+1
            error_list.append("investment return Error")
    except:
        print("***ERROR! investment return failed.")
        error_count = error_count+1
        error_list.append("Investment Return Failure")
```

Unit Tests

- Paste a screen shot of your unit tests successfully completed

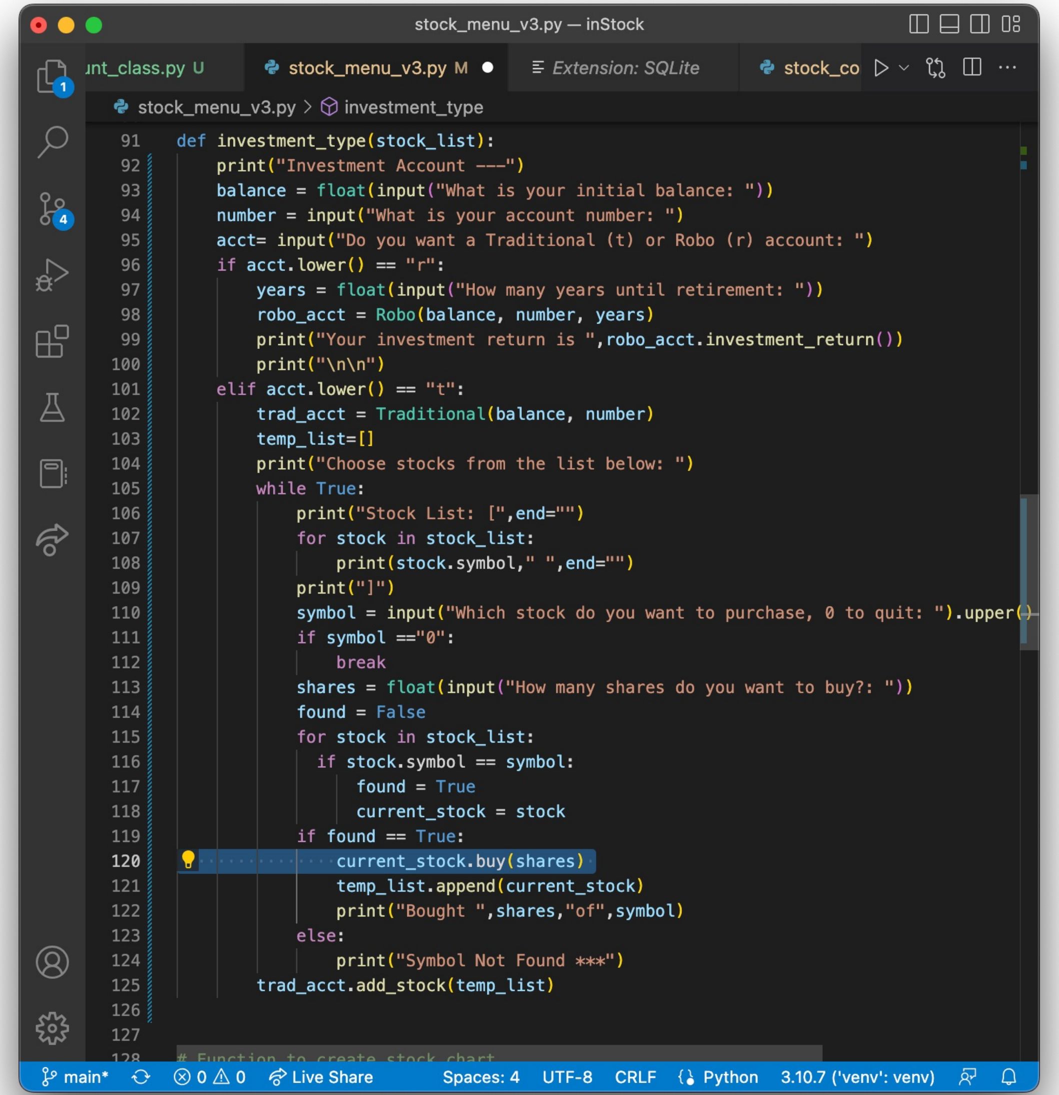


```python
        print("Successful!")
    except:
        print("***Adding Robo Account Failed!")
        error_count = error_count+1
        error_list.append("Robo Constructor Error")

    # Test Change years
    print("Test Change Balance...",end="")
    try:
        testRobo.years = 1000

        if testRobo.years == 1000:
            print("Successful!")
        else:
            print("***ERROR! Years change unsuccessful.")
            error_count = error_count+1
            error_list.append("Years Change Error")
    except:
        print("***ERROR! Years change failed.")
        error_count = error_count+1
        error_list.append("Years Change Failure")
```

```
6 - Investor Type
7 - Load Data
0 - Exit Program
Enter Menu Option: 0
Goodbye
(venv) mmastache@mac inStock % /Users/mmastache/Python/inStock/venv/bin/pyt
hon /Users/mmastache/Python/inStock/account_class.py
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
(venv) mmastache@mac inStock %
```

Stock menu program

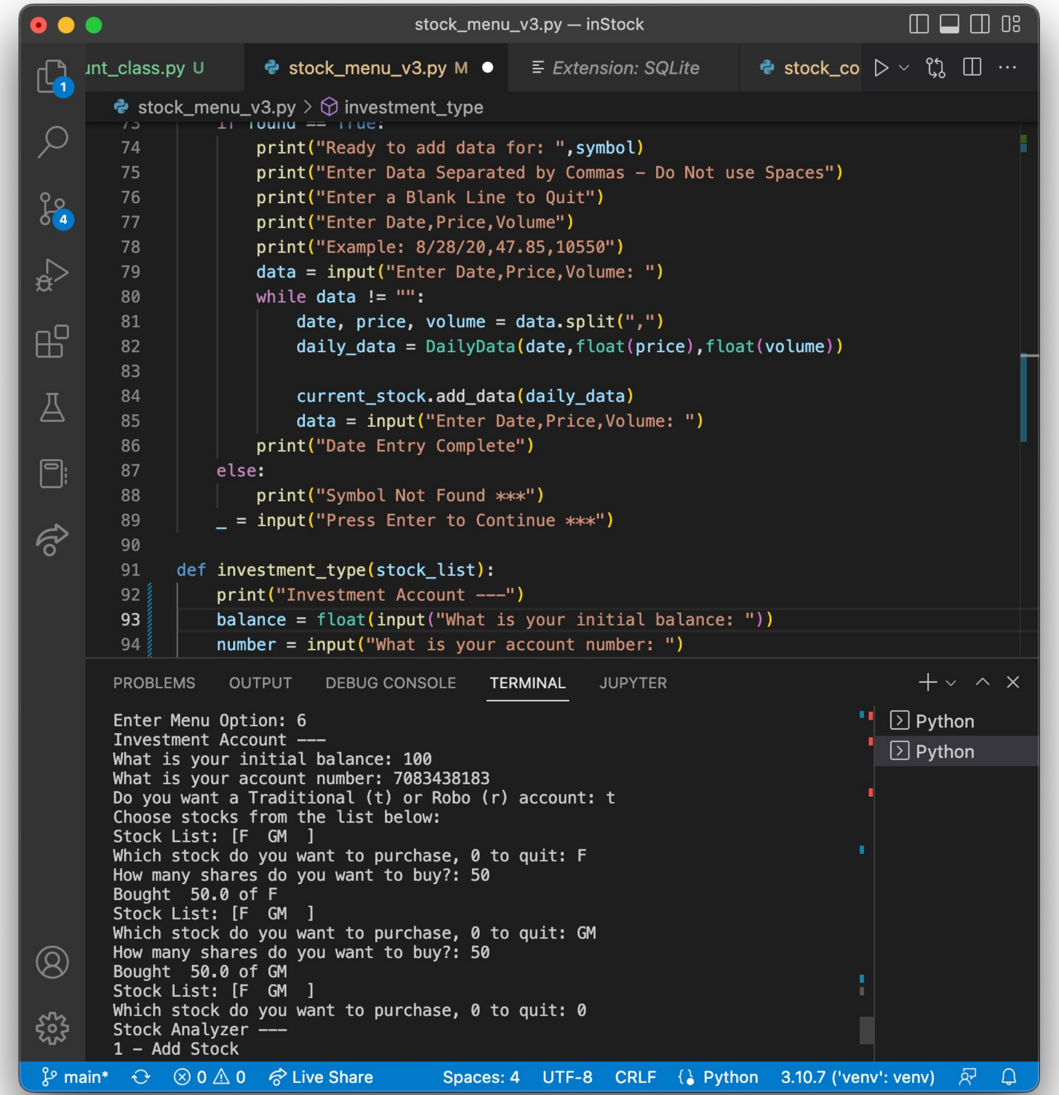
- Paste a screen shot of your classes in the main program



```
def investment_type(stock_list):
    print("Investment Account ---")
    balance = float(input("What is your initial balance: "))
    number = input("What is your account number: ")
    acct= input("Do you want a Traditional (t) or Robo (r) account: ")
    if acct.lower() == "r":
        years = float(input("How many years until retirement: "))
        robo_acct = Robo(balance, number, years)
        print("Your investment return is ",robo_acct.investment_return())
        print("\n\n")
    elif acct.lower() == "t":
        trad_acct = Traditional(balance, number)
        temp_list=[]
        print("Choose stocks from the list below: ")
        while True:
            print("Stock List: [",end="")
            for stock in stock_list:
                print(stock.symbol," ",end="")
            print("]")
            symbol = input("Which stock do you want to purchase, 0 to quit: ").upper(
            if symbol =="0":
                break
            shares = float(input("How many shares do you want to buy?: "))
            found = False
            for stock in stock_list:
              if stock.symbol == symbol:
                  found = True
                  current_stock = stock
            if found == True:
                current_stock.buy(shares)
                temp_list.append(current_stock)
                print("Bought ",shares,"of",symbol)
            else:
                print("Symbol Not Found ***")
        trad_acct.add_stock(temp_list)


# Function to create stock chart
```

Stock menu program

- Traditional Account

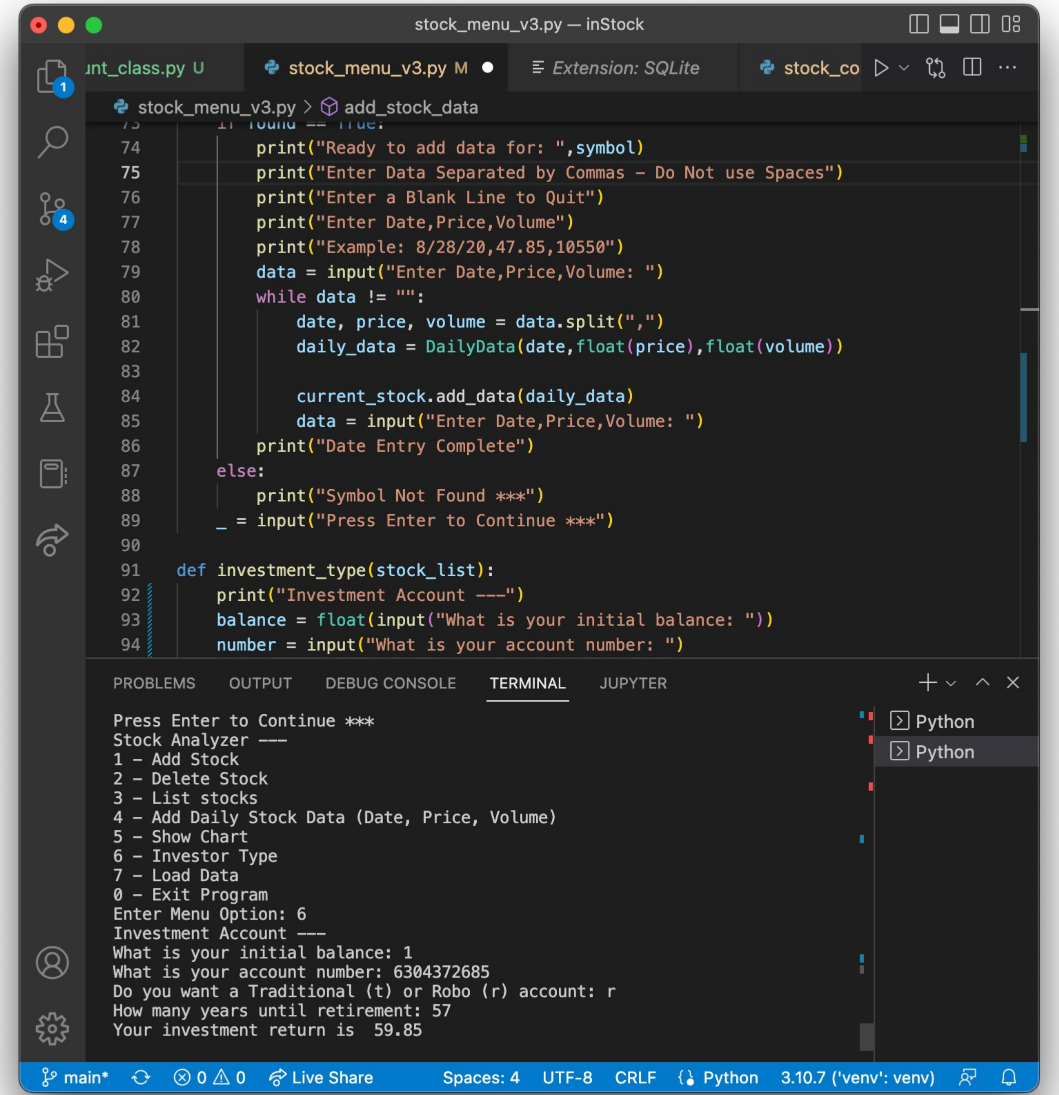


```python
        print("Ready to add data for: ",symbol)
        print("Enter Data Separated by Commas - Do Not use Spaces")
        print("Enter a Blank Line to Quit")
        print("Enter Date,Price,Volume")
        print("Example: 8/28/20,47.85,10550")
        data = input("Enter Date,Price,Volume: ")
        while data != "":
            date, price, volume = data.split(",")
            daily_data = DailyData(date,float(price),float(volume))

            current_stock.add_data(daily_data)
            data = input("Enter Date,Price,Volume: ")
        print("Date Entry Complete")
    else:
        print("Symbol Not Found ***")
    _ = input("Press Enter to Continue ***")

def investment_type(stock_list):
    print("Investment Account ---")
    balance = float(input("What is your initial balance: "))
    number = input("What is your account number: ")
```

```
Enter Menu Option: 6
Investment Account ---
What is your initial balance: 100
What is your account number: 7083438183
Do you want a Traditional (t) or Robo (r) account: t
Choose stocks from the list below:
Stock List: [F  GM  ]
Which stock do you want to purchase, 0 to quit: F
How many shares do you want to buy?: 50
Bought  50.0 of F
Stock List: [F  GM  ]
Which stock do you want to purchase, 0 to quit: GM
How many shares do you want to buy?: 50
Bought  50.0 of GM
Stock List: [F  GM  ]
Which stock do you want to purchase, 0 to quit: 0
Stock Analyzer ---
1 - Add Stock
```

Stock menu program

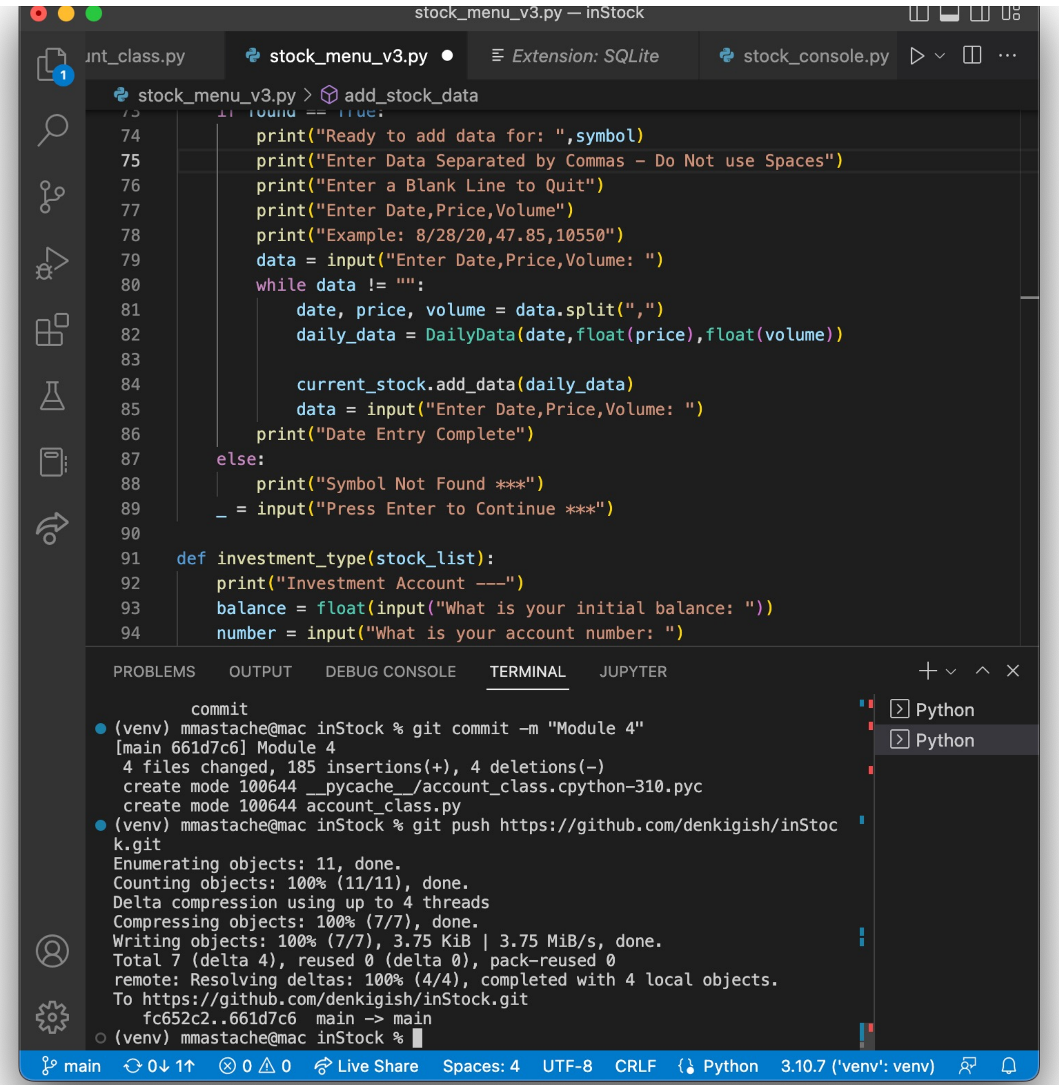
- Robo Account



```python
    if found == True:
        print("Ready to add data for: ",symbol)
        print("Enter Data Separated by Commas - Do Not use Spaces")
        print("Enter a Blank Line to Quit")
        print("Enter Date,Price,Volume")
        print("Example: 8/28/20,47.85,10550")
        data = input("Enter Date,Price,Volume: ")
        while data != "":
            date, price, volume = data.split(",")
            daily_data = DailyData(date,float(price),float(volume))

            current_stock.add_data(daily_data)
            data = input("Enter Date,Price,Volume: ")
        print("Date Entry Complete")
    else:
        print("Symbol Not Found ***")
    _ = input("Press Enter to Continue ***")

def investment_type(stock_list):
    print("Investment Account ---")
    balance = float(input("What is your initial balance: "))
    number = input("What is your account number: ")
```

```
Press Enter to Continue ***
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: 1
What is your account number: 6304372685
Do you want a Traditional (t) or Robo (r) account: r
How many years until retirement: 57
Your investment return is  59.85
```

Github Push



```python
            print("Ready to add data for: ",symbol)
            print("Enter Data Separated by Commas - Do Not use Spaces")
            print("Enter a Blank Line to Quit")
            print("Enter Date,Price,Volume")
            print("Example: 8/28/20,47.85,10550")
            data = input("Enter Date,Price,Volume: ")
            while data != "":
                date, price, volume = data.split(",")
                daily_data = DailyData(date,float(price),float(volume))

                current_stock.add_data(daily_data)
                data = input("Enter Date,Price,Volume: ")
            print("Date Entry Complete")
        else:
            print("Symbol Not Found ***")
        _ = input("Press Enter to Continue ***")

    def investment_type(stock_list):
        print("Investment Account ---")
        balance = float(input("What is your initial balance: "))
        number = input("What is your account number: ")
```

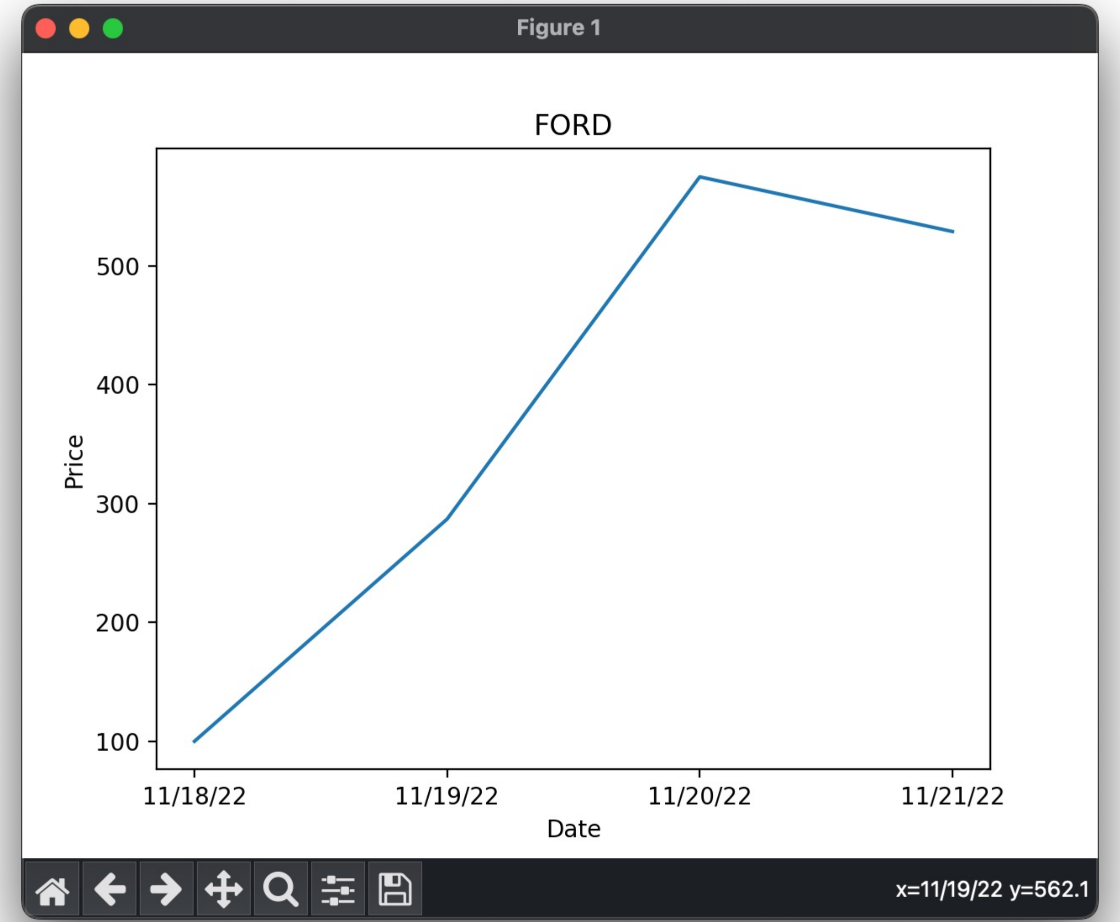
```
    commit
(venv) mmastache@mac inStock % git commit -m "Module 4"
[main 661d7c6] Module 4
 4 files changed, 185 insertions(+), 4 deletions(-)
 create mode 100644 __pycache__/account_class.cpython-310.pyc
 create mode 100644 account_class.py
(venv) mmastache@mac inStock % git push https://github.com/denkigish/inStoc
k.git
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 4 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (7/7), 3.75 KiB | 3.75 MiB/s, done.
Total 7 (delta 4), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (4/4), completed with 4 local objects.
To https://github.com/denkigish/inStock.git
   fc652c2..661d7c6  main -> main
(venv) mmastache@mac inStock %
```

Charts

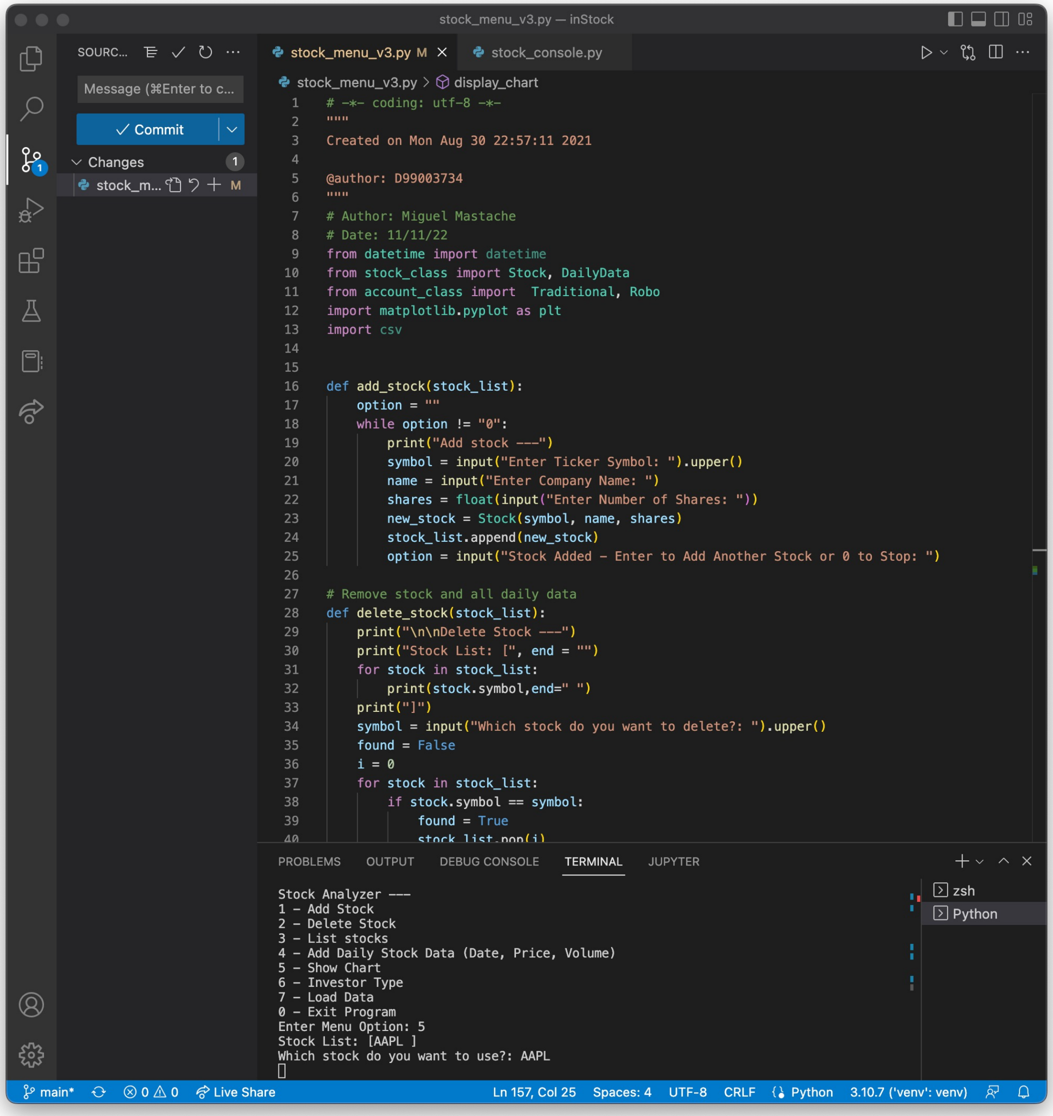
In this section, we use the matplotlib library to display charts for the stock data.

Chart

- Paste a screen shot of your stock chart.



Code

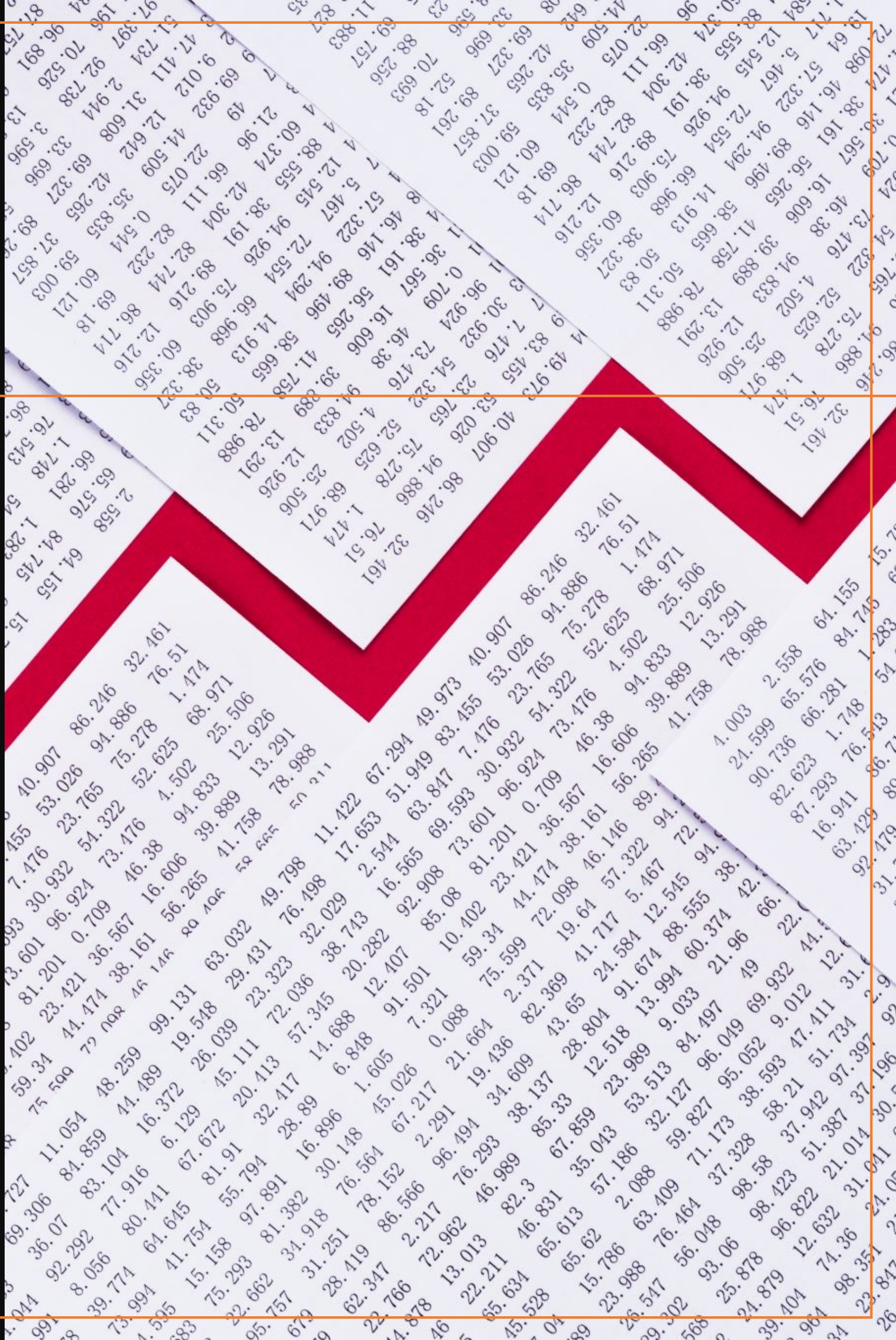


```python
# -*- coding: utf-8 -*-
"""
Created on Mon Aug 30 22:57:11 2021

@author: D99003734
"""
# Author: Miguel Mastache
# Date: 11/11/22
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: ")

# Remove stock and all daily data
def delete_stock(stock_list):
    print("\n\nDelete Stock ---")
    print("Stock List: [", end = "")
    for stock in stock_list:
        print(stock.symbol,end=" ")
    print("]")
    symbol = input("Which stock do you want to delete?: ").upper()
    found = False
    i = 0
    for stock in stock_list:
        if stock.symbol == symbol:
            found = True
            stock_list.pop(i)
```

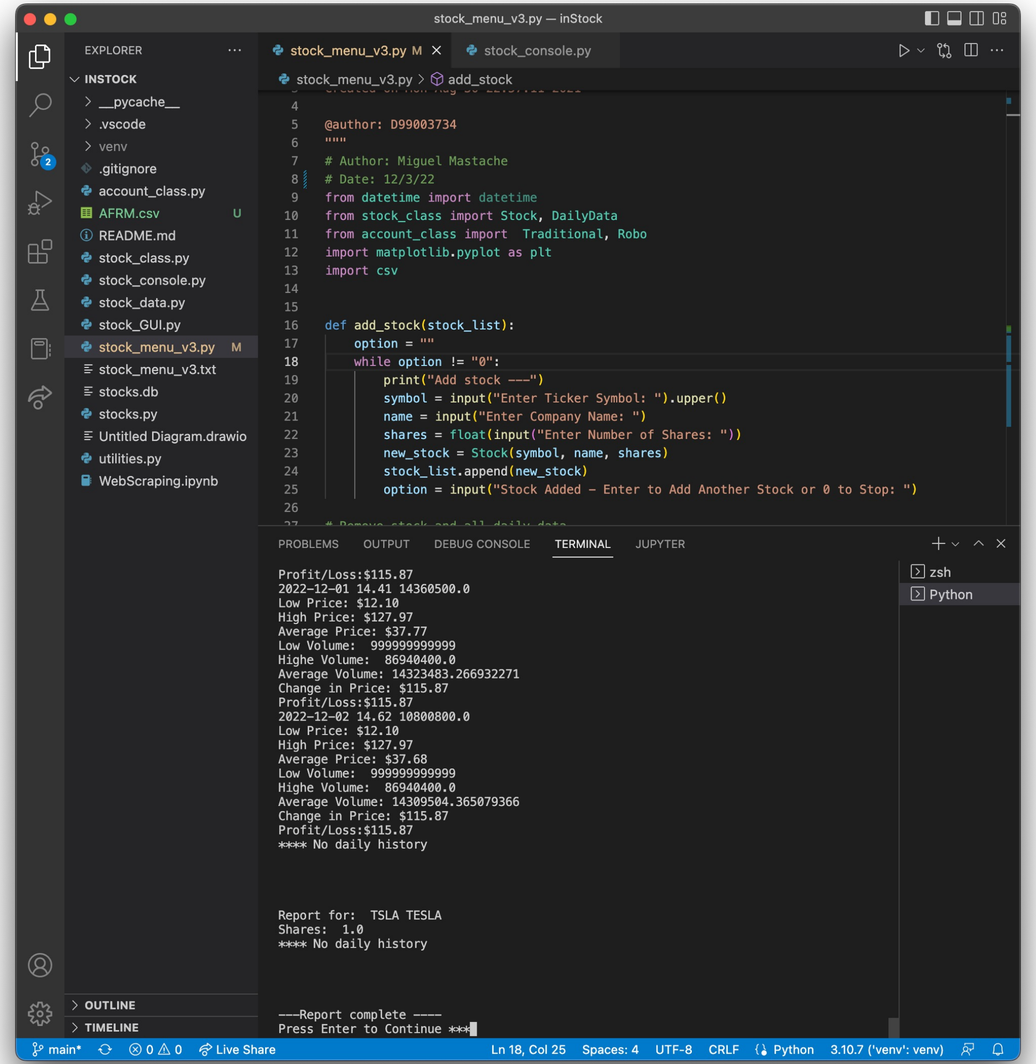
```
Stock Analyzer ---
1 - Add Stock
2 - Delete Stock
3 - List stocks
4 - Add Daily Stock Data (Date, Price, Volume)
5 - Show Chart
6 - Investor Type
7 - Load Data
0 - Exit Program
Enter Menu Option: 5
Stock List: [AAPL ]
Which stock do you want to use?: AAPL
```

File Processing

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



Code

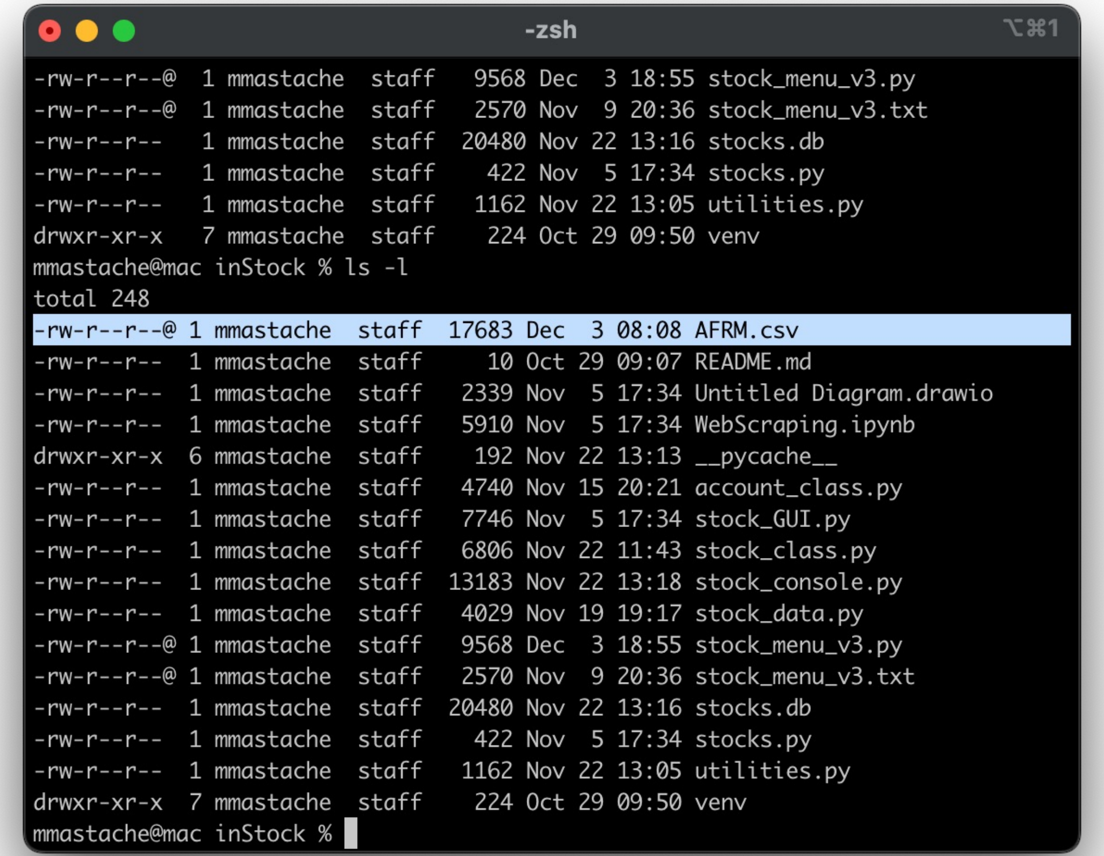


```python
@author: D99003734
"""
# Author: Miguel Mastache
# Date: 12/3/22
from datetime import datetime
from stock_class import Stock, DailyData
from account_class import  Traditional, Robo
import matplotlib.pyplot as plt
import csv


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: ")
```

```
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 ***
```

File

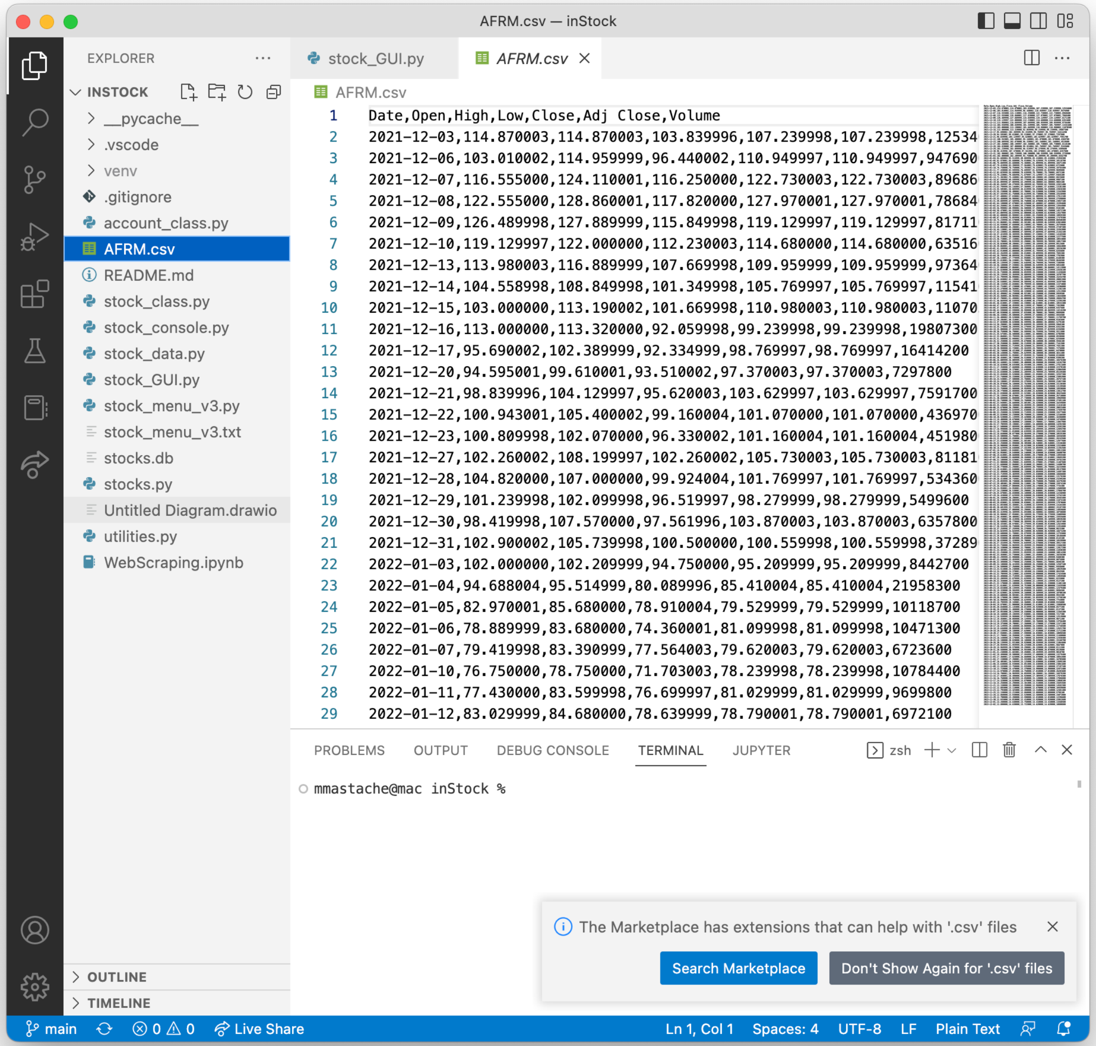
- Paste a screen shot of the file downloaded from Yahoo finance



```
-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



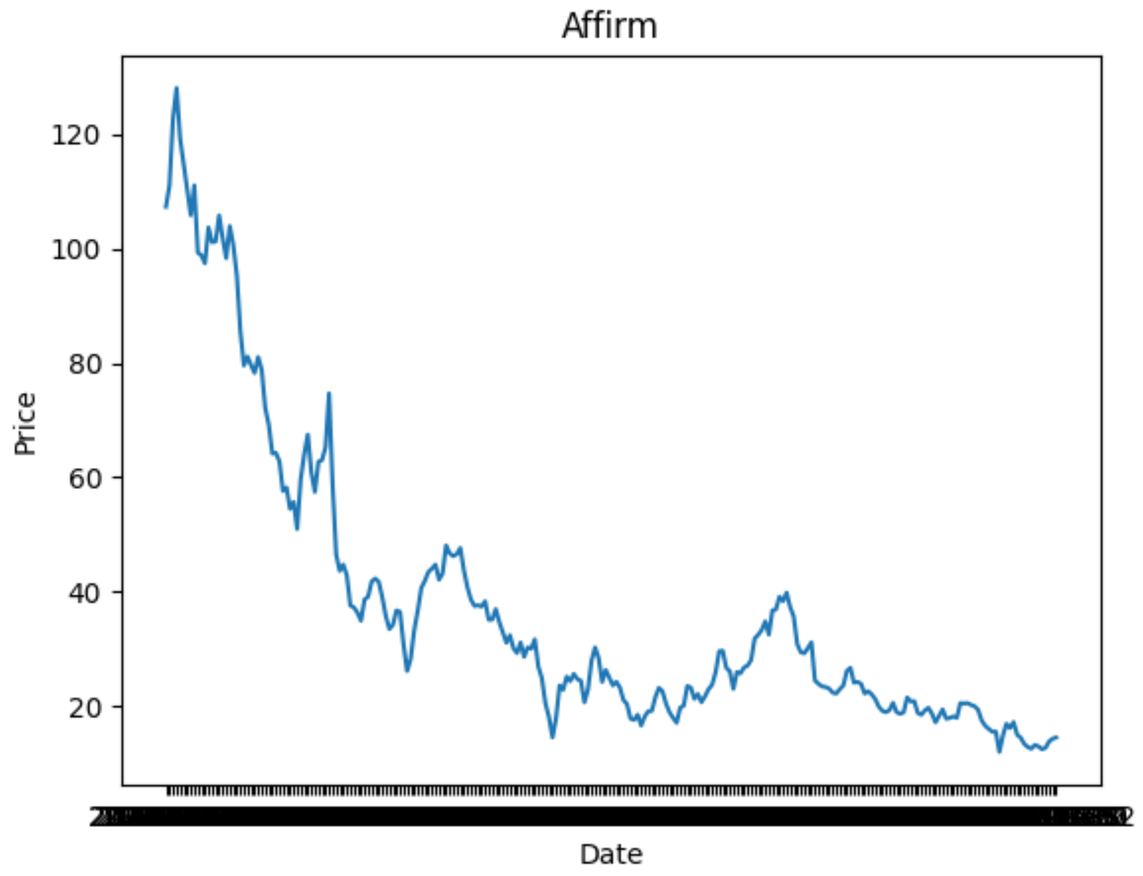
```
Date,Open,High,Low,Close,Adj Close,Volume
2021-12-03,114.870003,114.870003,103.839996,107.239998,107.239998,12534
2021-12-06,103.010002,114.959999,96.440002,110.949997,110.949997,947690
2021-12-07,116.555000,124.110001,116.250000,122.730003,122.730003,89686
2021-12-08,122.555000,128.860001,117.820000,127.970001,127.970001,78684
2021-12-09,126.489998,127.889999,115.849998,119.129997,119.129997,81711
2021-12-10,119.129997,122.000000,112.230003,114.680000,114.680000,63516
2021-12-13,113.980003,116.889999,107.669998,109.959999,109.959999,97364
2021-12-14,104.558998,108.849998,101.349998,105.769997,105.769997,11541
2021-12-15,103.000000,113.190002,101.669998,110.980003,110.980003,11070
2021-12-16,113.000000,113.320000,92.059998,99.239998,99.239998,19807300
2021-12-17,95.690002,102.389999,92.334999,98.769997,98.769997,16414200
2021-12-20,94.595001,99.610001,93.510002,97.370003,97.370003,7297800
2021-12-21,98.839996,104.129997,95.620003,103.629997,103.629997,7591700
2021-12-22,100.943001,105.400002,99.160004,101.070000,101.070000,436970
2021-12-23,100.809998,102.070000,96.330002,101.160004,101.160004,451980
2021-12-27,102.260002,108.199997,102.260002,105.730003,105.730003,81181
2021-12-28,104.820000,107.000000,99.924004,101.769997,101.769997,534360
2021-12-29,101.239998,102.099998,96.519997,98.279999,98.279999,5499600
2021-12-30,98.419998,107.570000,97.561996,103.870003,103.870003,6357800
2021-12-31,102.900002,105.739998,100.500000,100.559998,100.559998,37289
2022-01-03,102.000000,102.209999,94.750000,95.209999,95.209999,8442700
2022-01-04,94.688004,95.514999,80.089996,85.410004,85.410004,21958300
2022-01-05,82.970001,85.680000,78.910004,79.529999,79.529999,10118700
2022-01-06,78.889999,83.680000,74.360001,81.099998,81.099998,10471300
2022-01-07,79.419998,83.390999,77.564003,79.620003,79.620003,6723600
2022-01-10,76.750000,78.750000,71.703003,78.239998,78.239998,10784400
2022-01-11,77.430000,83.599998,76.699997,81.029999,81.029999,9699800
2022-01-12,83.029999,84.680000,78.639999,78.790001,78.790001,6972100
```

```
mmastache@mac inStock %
```

The Marketplace has extensions that can help with '.csv' files

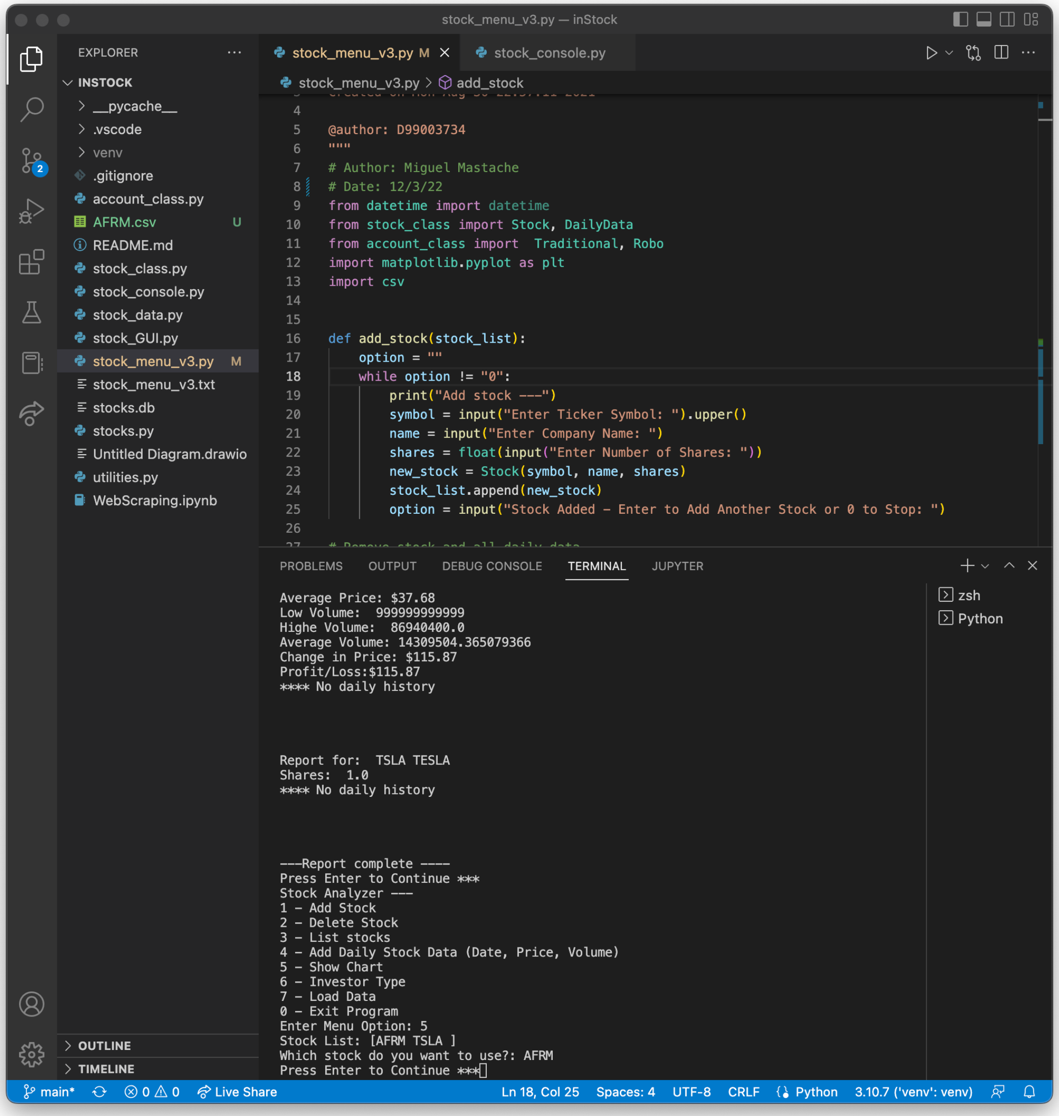
Importing data

- Screenshot of the historical data import



Importing data

- Screenshot of the historical data import



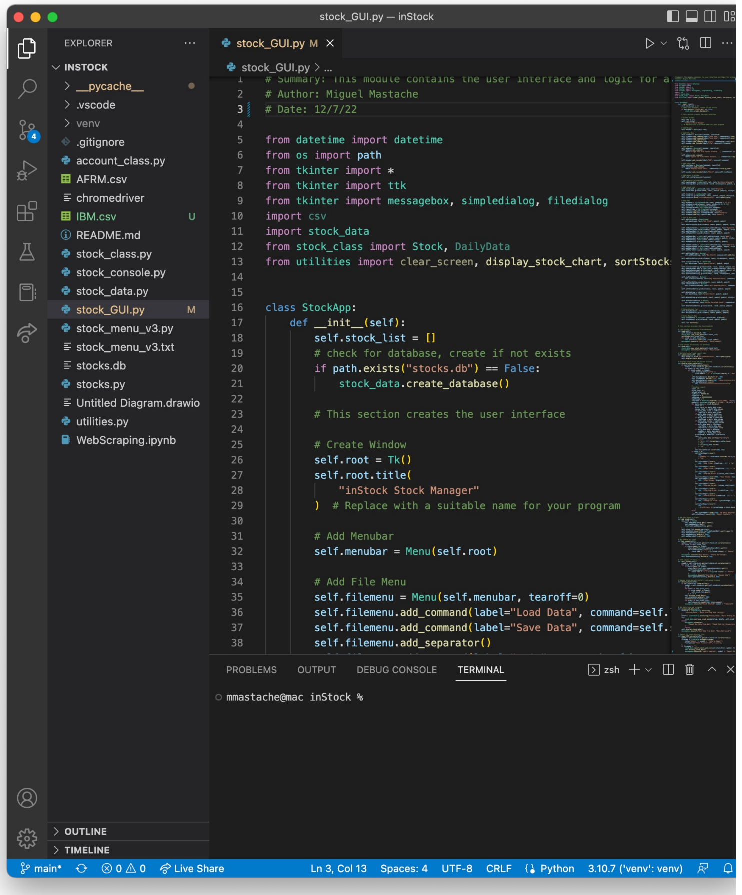
```python
@author: D99003734
"""
# Author: Miguel Mastache
# Date: 12/3/22
from datetime import datetime
from stock_class import Stock, DailyData
from account_class import  Traditional, Robo
import matplotlib.pyplot as plt
import csv


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: ")
```

```
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 ***
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: 5
Stock List: [AFRM TSLA ]
Which stock do you want to use?: AFRM
Press Enter to Continue ***
```

GUI

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

Code

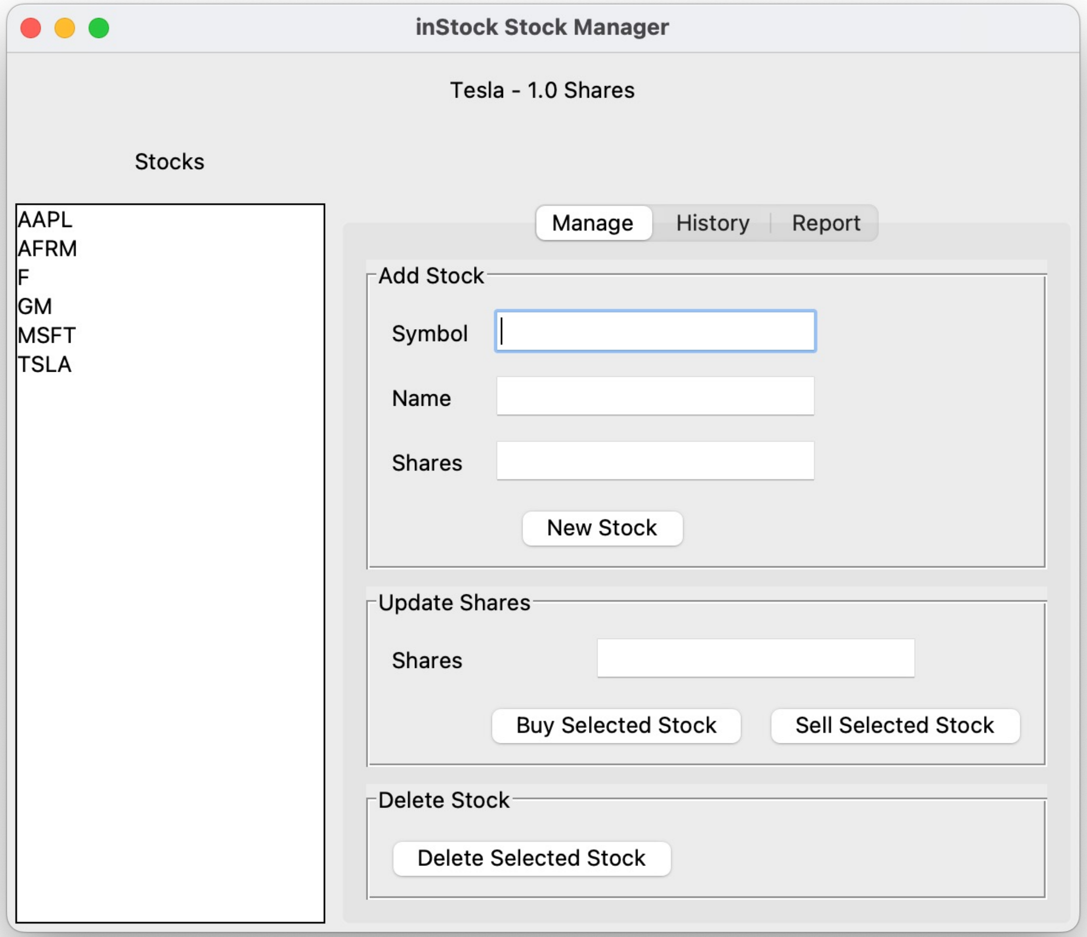


```python
# Summary: This module contains the user interface and logic for a
# Author: Miguel Mastache
# Date: 12/7/22

from datetime import datetime
from os import path
from tkinter import *
from tkinter import ttk
from tkinter import messagebox, simpledialog, filedialog
import csv
import stock_data
from stock_class import Stock, DailyData
from utilities import clear_screen, display_stock_chart, sortStock


class StockApp:
    def __init__(self):
        self.stock_list = []
        # check for database, create if not exists
        if path.exists("stocks.db") == False:
            stock_data.create_database()

        # This section creates the user interface

        # Create Window
        self.root = Tk()
        self.root.title(
            "inStock Stock Manager"
        )  # Replace with a suitable name for your program

        # Add Menubar
        self.menubar = Menu(self.root)

        # Add File Menu
        self.filemenu = Menu(self.menubar, tearoff=0)
        self.filemenu.add_command(label="Load Data", command=self.
        self.filemenu.add_command(label="Save Data", command=self.
        self.filemenu.add_separator()
```

```
mmastache@mac inStock %
```

Stocks in GUI

- Paste a screen shot of your GUI working


inStock Stock Manager

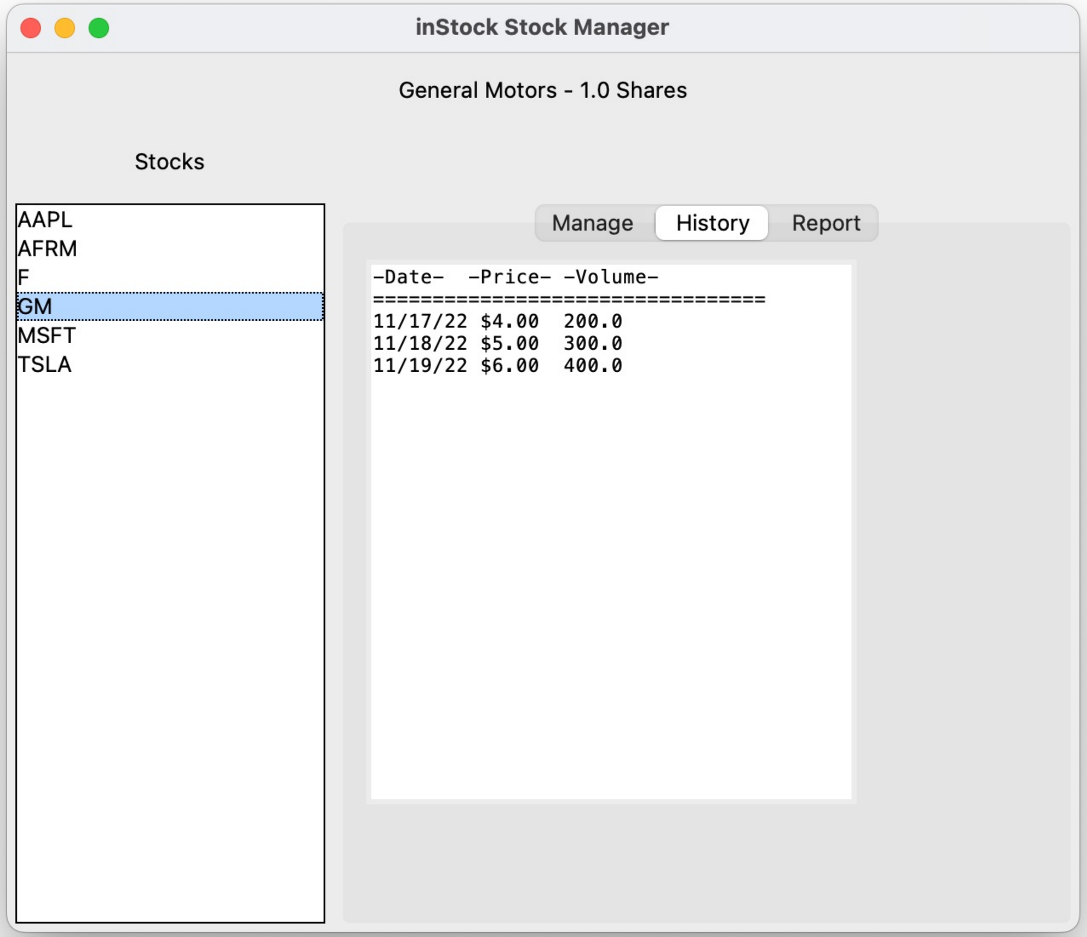
Tesla - 1.0 Shares

Stocks

AAPL
AFRM
F
GM
MSFT
TSLA

History Tab

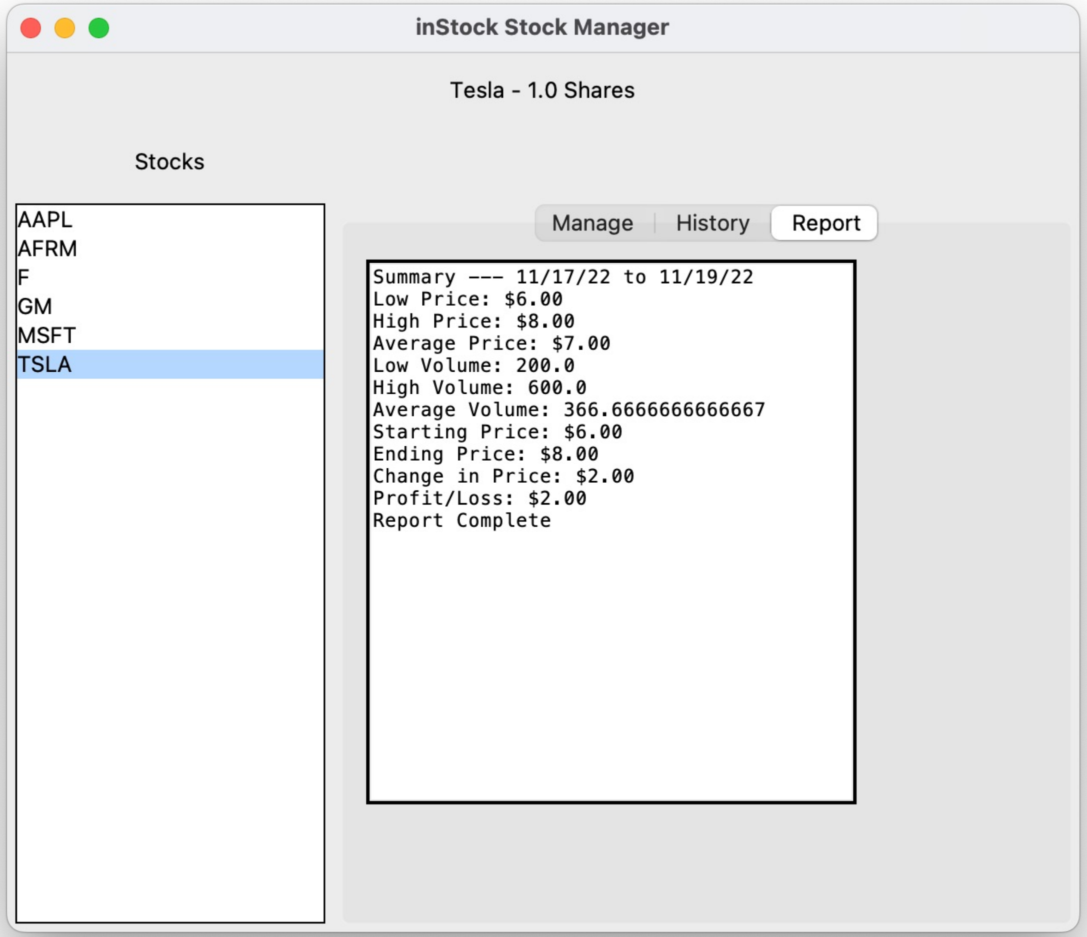
- Paste a screen shot of your History tab with import working.



```
-Date-   -Price- -Volume-
=================================
11/17/22 $4.00   200.0
11/18/22 $5.00   300.0
11/19/22 $6.00   400.0
```

Report Complete

- Paste a screen shot of your Report tab



```
Summary --- 11/17/22 to 11/19/22
Low Price: $6.00
High Price: $8.00
Average Price: $7.00
Low Volume: 200.0
High Volume: 600.0
Average Volume: 366.6666666666667
Starting Price: $6.00
Ending Price: $8.00
Change in Price: $2.00
Profit/Loss: $2.00
Report Complete
```

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.