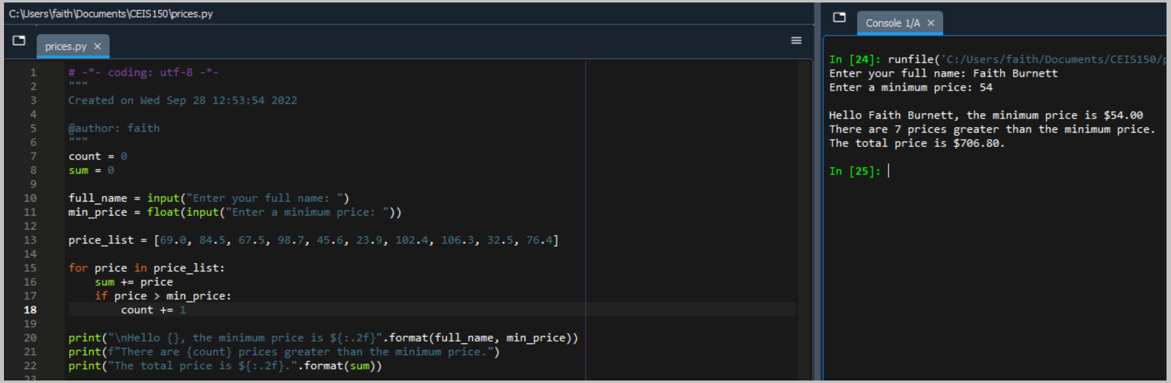
Stock Analyzer

Faith Burnett
CEIS-150
DeVry University

Software Setup

- Install or update IDE
- Review Python by creating a simple program

Python Review Program



```python
# -*- coding: utf-8 -*-
"""
Created on Wed Sep 28 12:53:54 2022

@author: faith
"""
count = 0
sum = 0

full_name = input("Enter your full name: ")
min_price = float(input("Enter a minimum price: "))

price_list = [69.0, 84.5, 67.5, 98.7, 45.6, 23.9, 102.4, 106.3, 32.5, 76.4]

for price in price_list:
    sum += price
    if price > min_price:
        count += 1

print("\nHello {}, the minimum price is ${:.2f}".format(full_name, min_price))
print(f"There are {count} prices greater than the minimum price.")
print("The total price is ${:.2f}.".format(sum))
```

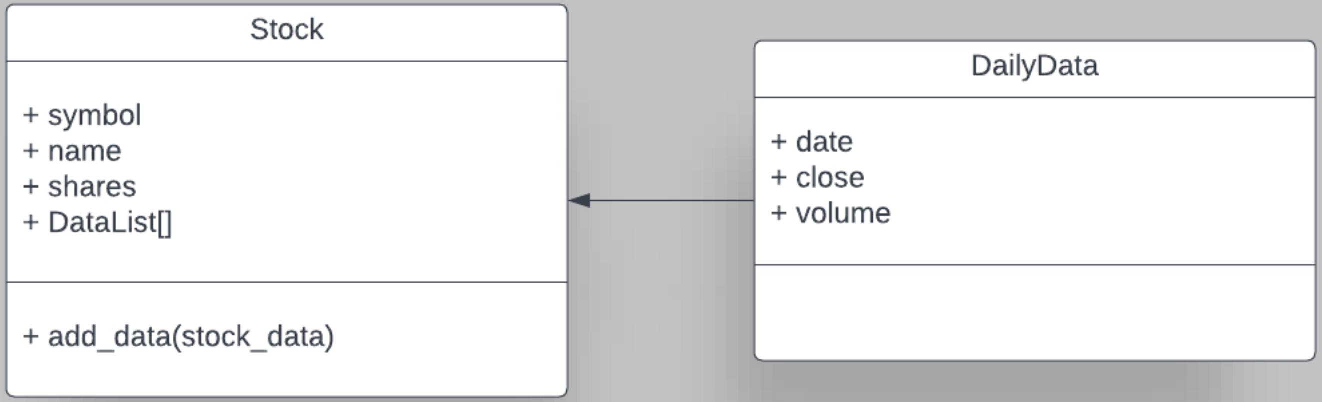
```
In [24]: runfile('C:/Users/faith/Documents/CEIS150/p
Enter your full name: Faith Burnett
Enter a minimum price: 54

Hello Faith Burnett, the minimum price is $54.00
There are 7 prices greater than the minimum price.
The total price is $706.80.

In [25]: 
```

Stock Class Setup

- Class diagram
- Create Stock and DailyData classes
- Run unit test

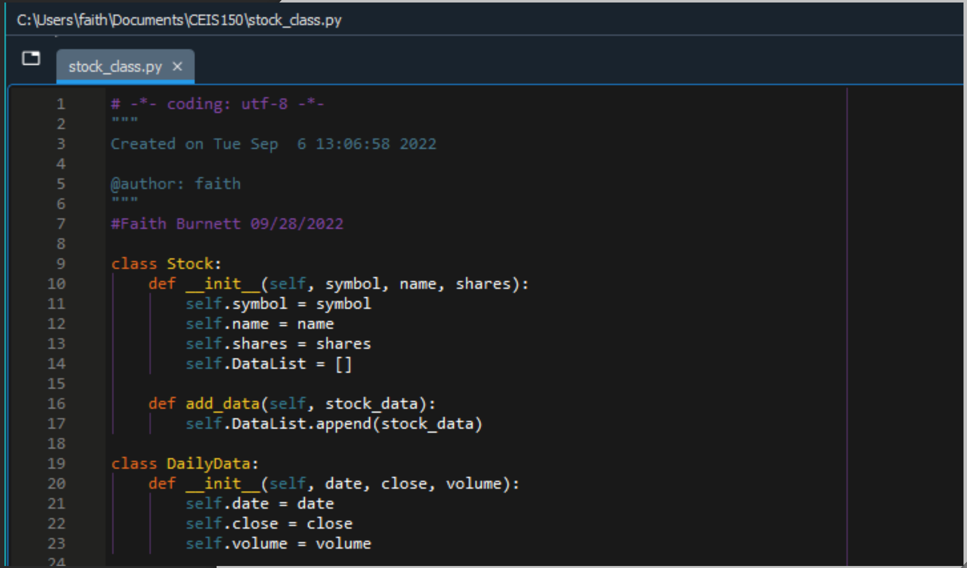
Class Diagram

- Layout of Stock and DailyData classes



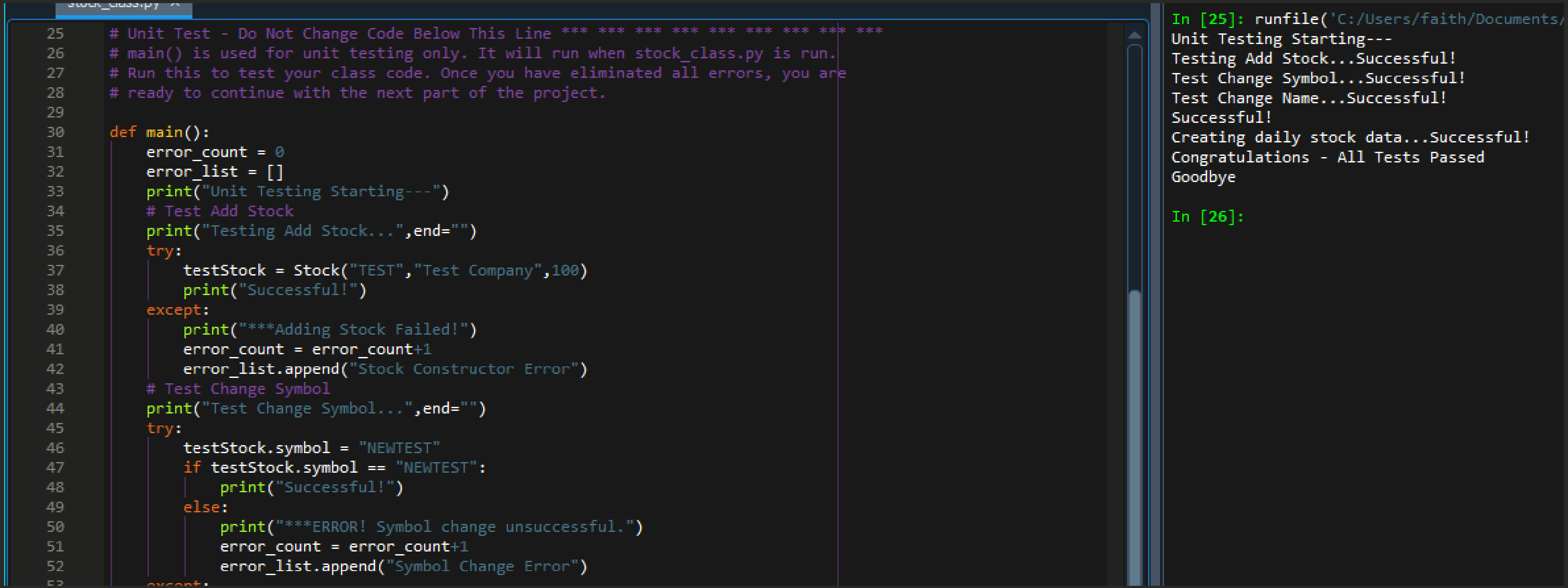
Class Code

- Initialize Stock and DailyData classes
- Declare variables for each class
- Define add_data method for Stock class



```python
# -*- coding: utf-8 -*-
"""
Created on Tue Sep  6 13:06:58 2022

@author: faith
"""
#Faith Burnett 09/28/2022

class Stock:
    def __init__(self, symbol, name, shares):
        self.symbol = symbol
        self.name = name
        self.shares = shares
        self.DataList = []

    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
```

Unit Test



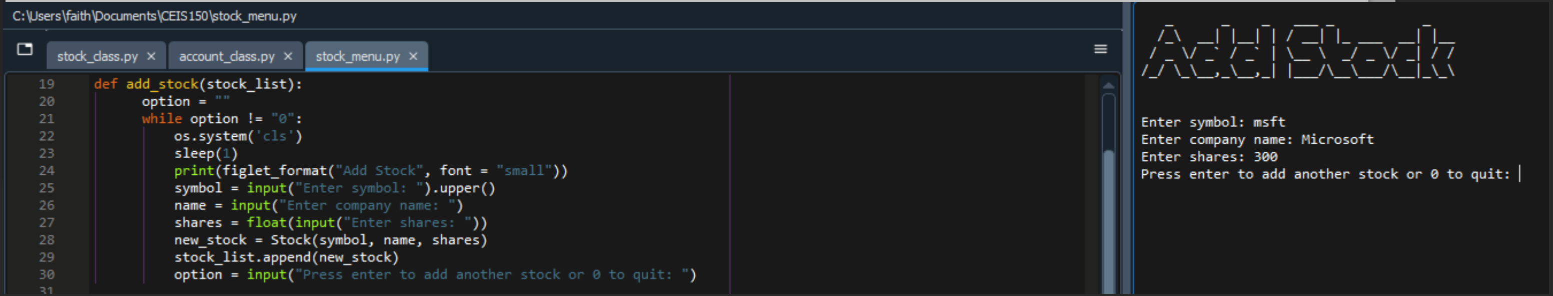
```python
# Unit Test - Do Not Change Code Below This Line *** *** *** *** *** *** *** *** ***
# main() is used for unit testing only. It will run when stock_class.py is run.
# Run this to test your class code. Once you have eliminated all errors, you are
# ready to continue with the next part of the project.

def main():
    error_count = 0
    error_list = []
    print("Unit Testing Starting---")
    # Test Add Stock
    print("Testing Add Stock...",end="")
    try:
        testStock = Stock("TEST","Test Company",100)
        print("Successful!")
    except:
        print("***Adding Stock Failed!")
        error_count = error_count+1
        error_list.append("Stock Constructor Error")
    # Test Change Symbol
    print("Test Change Symbol...",end="")
    try:
        testStock.symbol = "NEWTEST"
        if testStock.symbol == "NEWTEST":
            print("Successful!")
        else:
            print("***ERROR! Symbol change unsuccessful.")
            error_count = error_count+1
            error_list.append("Symbol Change Error")
    except:
```

```
In [25]: runfile('C:/Users/faith/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 [26]:
```

Stock Menu

- Create a menu-driven interface
- Implement add, delete, list and add stock data methods

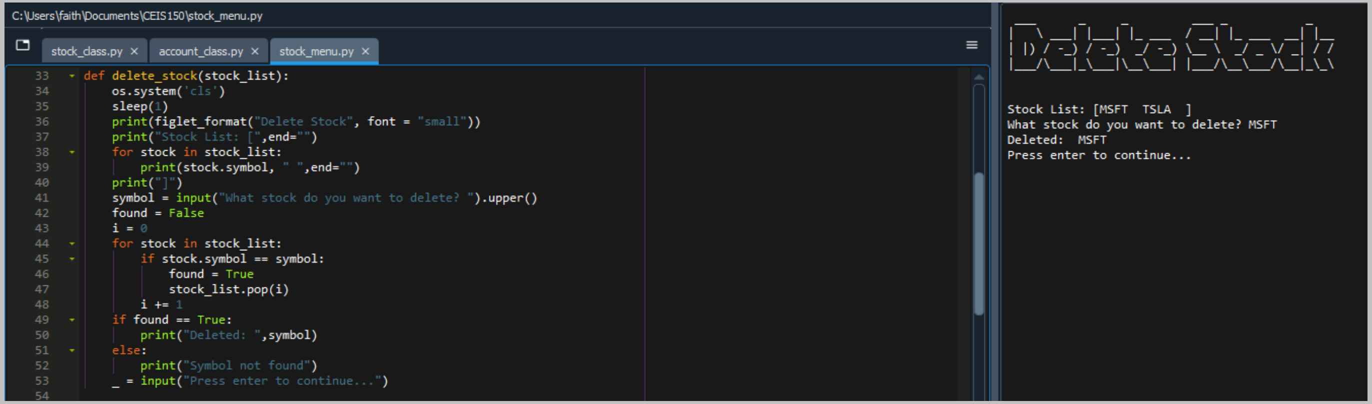
Add Stock



```python
def add_stock(stock_list):
    option = ""
    while option != "0":
        os.system('cls')
        sleep(1)
        print(figlet_format("Add Stock", font = "small"))
        symbol = input("Enter symbol: ").upper()
        name = input("Enter company name: ")
        shares = float(input("Enter shares: "))
        new_stock = Stock(symbol, name, shares)
        stock_list.append(new_stock)
        option = input("Press enter to add another stock or 0 to quit: ")
```

```
   _      _    _   ___ _           _   
  /_\  __| |__| | / __| |_ ___  __| |__
 / _ \/ _` / _` | \__ \  _/ _ \/ _| / /
/_/ \_\__,_\__,_| |___/\__\___/\__|_\_\


Enter symbol: msft
Enter company name: Microsoft
Enter shares: 300
Press enter to add another stock or 0 to quit: 
```

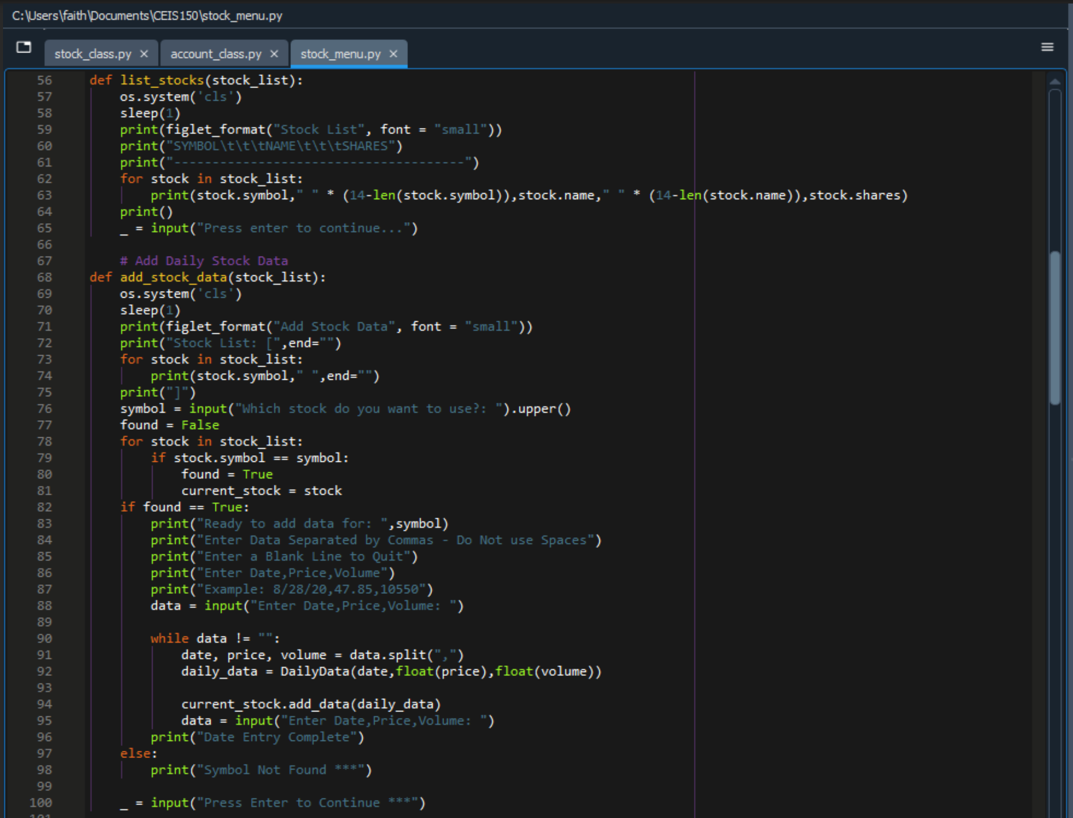
Delete Stock



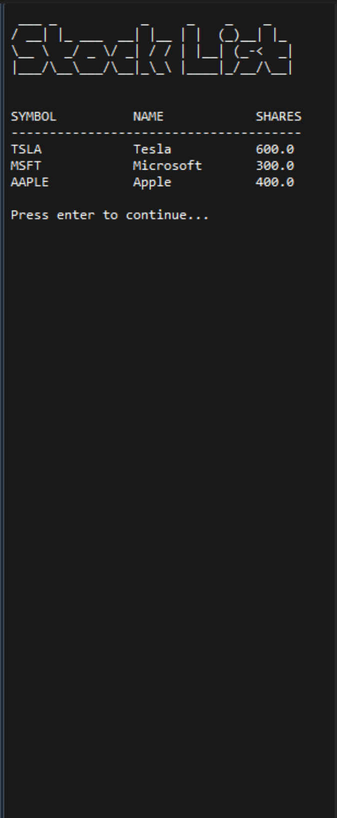
```python
def delete_stock(stock_list):
    os.system('cls')
    sleep(1)
    print(figlet_format("Delete Stock", font = "small"))
    print("Stock List: [",end="")
    for stock in stock_list:
        print(stock.symbol, " ",end="")
    print("]")
    symbol = input("What stock do you want to delete? ").upper()
    found = False
    i = 0
    for stock in stock_list:
        if stock.symbol == symbol:
            found = True
            stock_list.pop(i)
        i += 1
    if found == True:
        print("Deleted: ",symbol)
    else:
        print("Symbol not found")
    _ = input("Press enter to continue...")

```

```
 ___      _      _         ___ _           _   
|   \ ___| |___ | |_ ___  / __| |_ ___  __| |__
| |) / -_) / -_)|  _/ -_) \__ \  _/ _ \/ _| / /
|___/\___|_\___| \__\___| |___/\__\___/\__|_\_\


Stock List: [MSFT  TSLA  ]
What stock do you want to delete? MSFT
Deleted:  MSFT
Press enter to continue...
```

List Stock

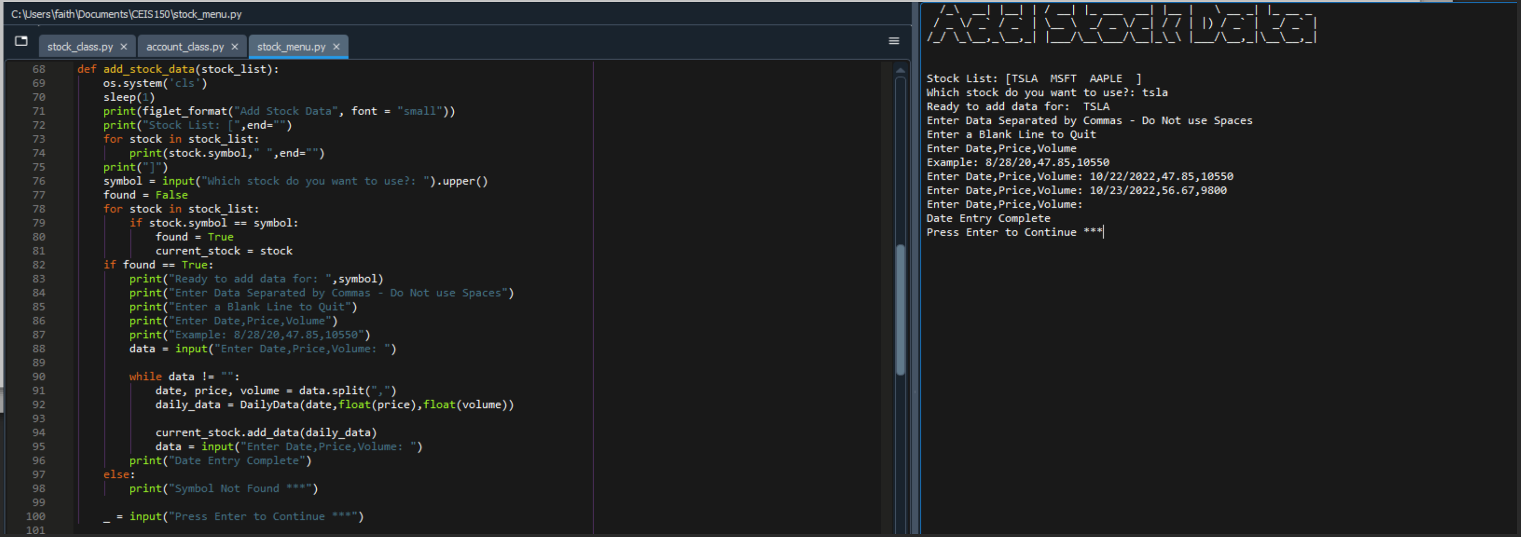


```python
def list_stocks(stock_list):
    os.system('cls')
    sleep(1)
    print(figlet_format("Stock List", font = "small"))
    print("SYMBOL\t\t\tNAME\t\t\tSHARES")
    print("--------------------------------------")
    for stock in stock_list:
        print(stock.symbol," " * (14-len(stock.symbol)),stock.name," " * (14-len(stock.name)),stock.shares)
    print()
    _ = input("Press enter to continue...")

    # Add Daily Stock Data
def add_stock_data(stock_list):
    os.system('cls')
    sleep(1)
    print(figlet_format("Add Stock Data", font = "small"))
    print("Stock List: [",end="")
    for stock in stock_list:
        print(stock.symbol," ",end="")
    print("]")
    symbol = input("Which stock do you want to use?: ").upper()
    found = False
    for stock in stock_list:
        if stock.symbol == symbol:
            found = True
            current_stock = stock
    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 ***")
```



```
  ___ _           _     _    _    _   
 / __| |_ ___  __| |__ | |  (_)__| |_ 
 \__ \  _/ _ \/ _| / / | |__| (_-<  _|
 |___/\__\___/\__|_\_\ |____|_/__/\__|


SYMBOL          NAME            SHARES
--------------------------------------
TSLA            Tesla           600.0
MSFT            Microsoft       300.0
AAPLE           Apple           400.0

Press enter to continue...
```

Add Stock Data



```python
def add_stock_data(stock_list):
    os.system('cls')
    sleep(1)
    print(figlet_format("Add Stock Data", font = "small"))
    print("Stock List: [",end="")
    for stock in stock_list:
        print(stock.symbol," ",end="")
    print("]")
    symbol = input("Which stock do you want to use?: ").upper()
    found = False
    for stock in stock_list:
        if stock.symbol == symbol:
            found = True
            current_stock = stock
    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 ***")

```

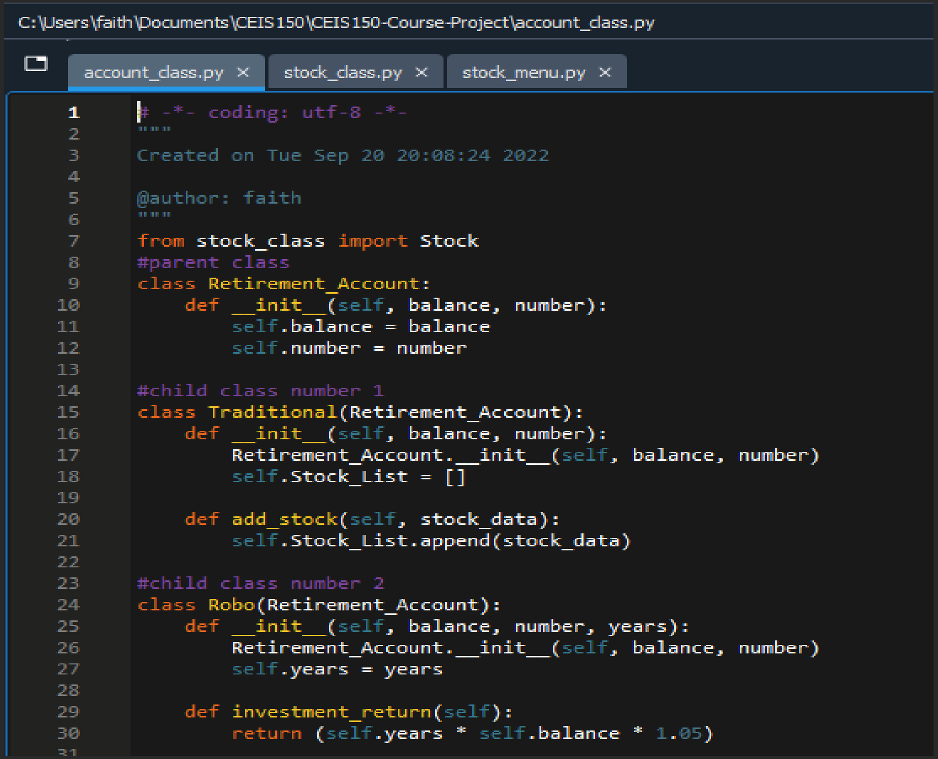
```
Stock List: [TSLA  MSFT  AAPLE  ]
Which stock do you want to use?: tsla
Ready to add data for:  TSLA
Enter Data Separated by Commas - Do Not use Spaces
Enter a Blank Line to Quit
Enter Date,Price,Volume
Example: 8/28/20,47.85,10550
Enter Date,Price,Volume: 10/22/2022,47.85,10550
Enter Date,Price,Volume: 10/23/2022,56.67,9800
Enter Date,Price,Volume:
Date Entry Complete
Press Enter to Continue ***
```

Account Class Setup

- Retirement account class
- Inherited classes Traditional, and Robo
- Unit Test
- Test investment account

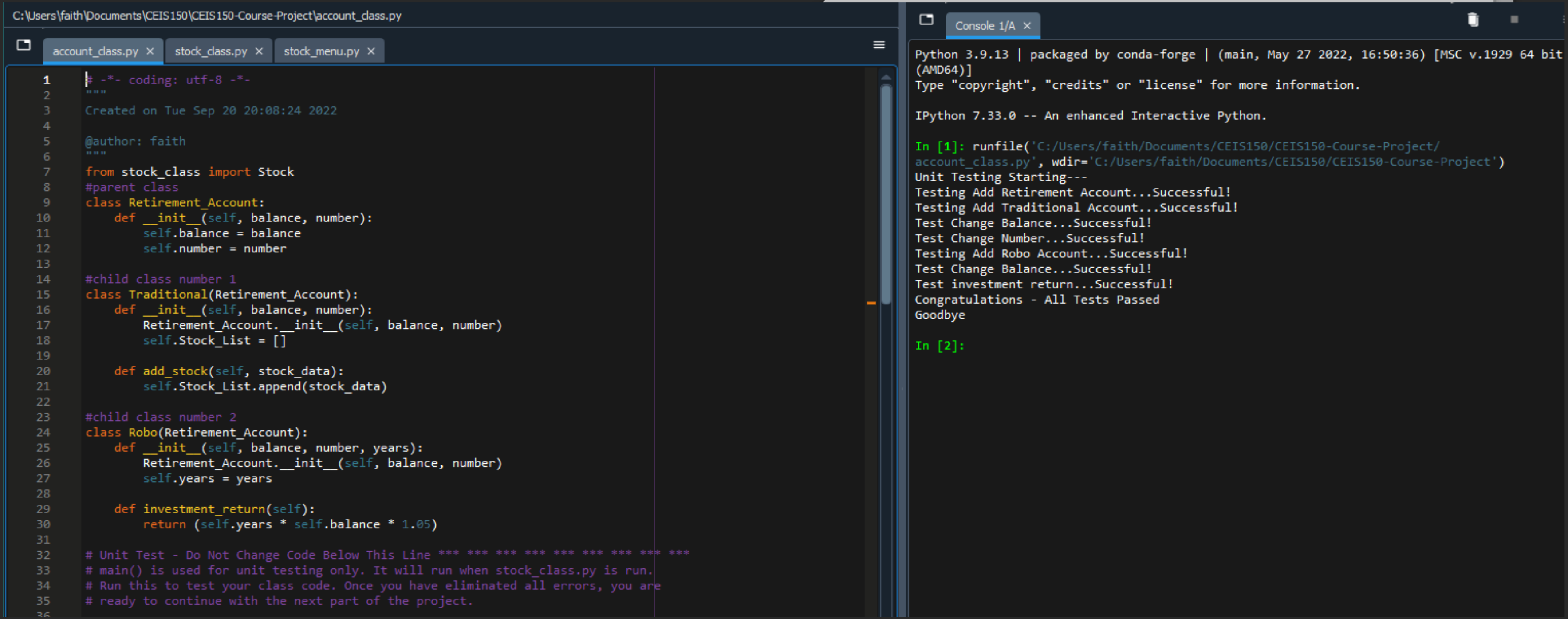
Inherited Classes

- Initialize Retirement_Account class
- Initialize inherited classes Traditional and Robo
- Declare variables for each class
- Define add_stock and investment_return methods


C:\Users\faith\Documents\CEIS150\CEIS150-Course-Project\account_class.py
account_class.py ×
stock_class.py ×
stock_menu.py ×

```python
# -*- coding: utf-8 -*-
"""
Created on Tue Sep 20 20:08:24 2022

@author: faith
"""
from stock_class import Stock
#parent class
class Retirement_Account:
    def __init__(self, balance, number):
        self.balance = balance
        self.number = number

#child class number 1
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)

#child class number 2
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)
```

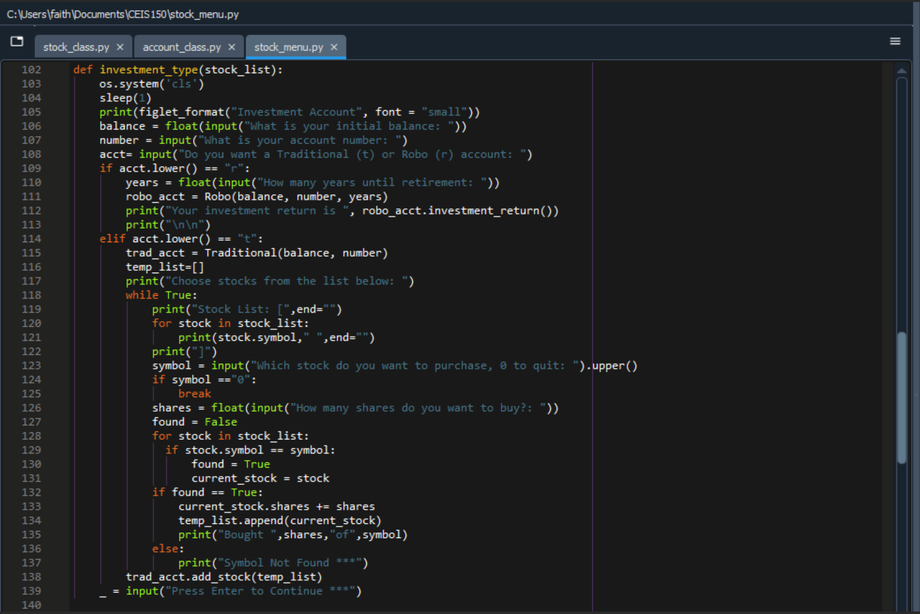
Unit Test


C:\Users\faith\Documents\CEIS150\CEIS150-Course-Project\account_class.py
account_class.py ×
stock_class.py ×
stock_menu.py ×
Console 1/A ×

```python
# -*- coding: utf-8 -*-
"""
Created on Tue Sep 20 20:08:24 2022

@author: faith
"""
from stock_class import Stock
#parent class
class Retirement_Account:
    def __init__(self, balance, number):
        self.balance = balance
        self.number = number

#child class number 1
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)

#child class number 2
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)

# Unit Test - Do Not Change Code Below This Line *** *** *** *** *** *** *** *** ***
# main() is used for unit testing only. It will run when stock_class.py is run.
# Run this to test your class code. Once you have eliminated all errors, you are
# ready to continue with the next part of the project.
```

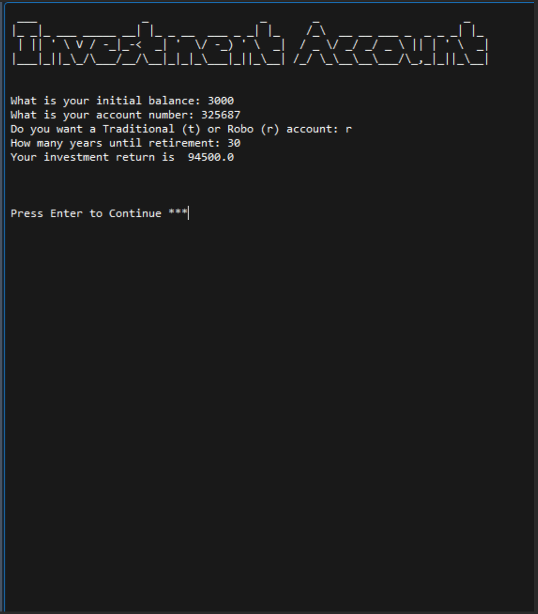
```
Python 3.9.13 | packaged by conda-forge | (main, May 27 2022, 16:50:36) [MSC v.1929 64 bit
(AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.33.0 -- An enhanced Interactive Python.

In [1]: runfile('C:/Users/faith/Documents/CEIS150/CEIS150-Course-Project/
account_class.py', wdir='C:/Users/faith/Documents/CEIS150/CEIS150-Course-Project')
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

In [2]:
```

Investment Account Test



C:\Users\faith\Documents\CEIS150\stock_menu.py

```python
def investment_type(stock_list):
    os.system('cls')
    sleep(1)
    print(figlet_format("Investment Account", font = "small"))
    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.shares += shares
                temp_list.append(current_stock)
                print("Bought ",shares,"of",symbol)
            else:
                print("Symbol Not Found ***")
        trad_acct.add_stock(temp_list)
    _ = input("Press Enter to Continue ***")
```



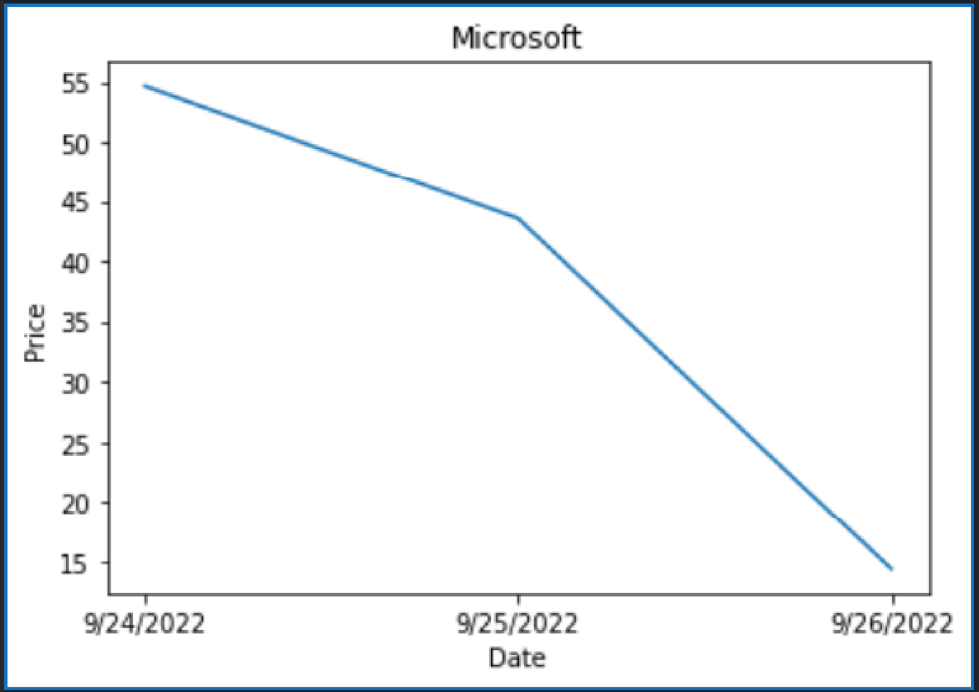
```
 ___                 _                    _      _                          _
|_ _|_ ___ _____ __| |_ _ __  ___ _ _| |_   /_\  __ __ ___ _  _ _ _| |_
 | || ' \ V / -_|_-<  _| '  \/ -_) ' \  _| / _ \/ _/ _/ _ \ || | ' \  _|
|___|_||_\_/\___/__/\__|_|_|_\___|_||_\__|/_/ \_\__\__\___/\_,_|_||_\__|


What is your initial balance: 3000
What is your account number: 325687
Do you want a Traditional (t) or Robo (r) account: r
How many years until retirement: 30
Your investment return is  94500.0



Press Enter to Continue ***
```

- Run stock_menu.py
- Simulate setting up an investment account

Charting Stock Data

- Add display_chart and display_stock_chart methods to stock menu
- Test show chart option

Chart

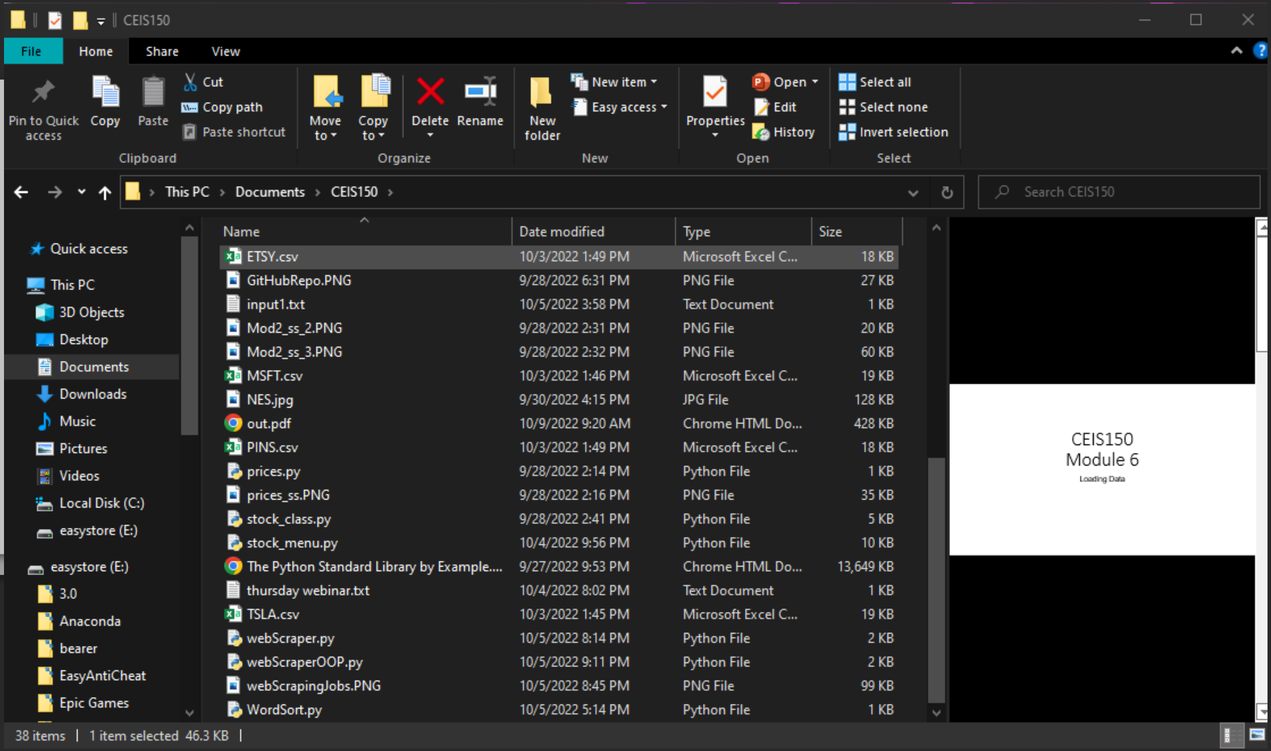


- Run stock_menu.py
- Select add daily data
- Select chart option to test

Importing Data

- Download historical stock data
- Move data to location of stock program
- Add import csv and display report methods
- Test import csv and display report

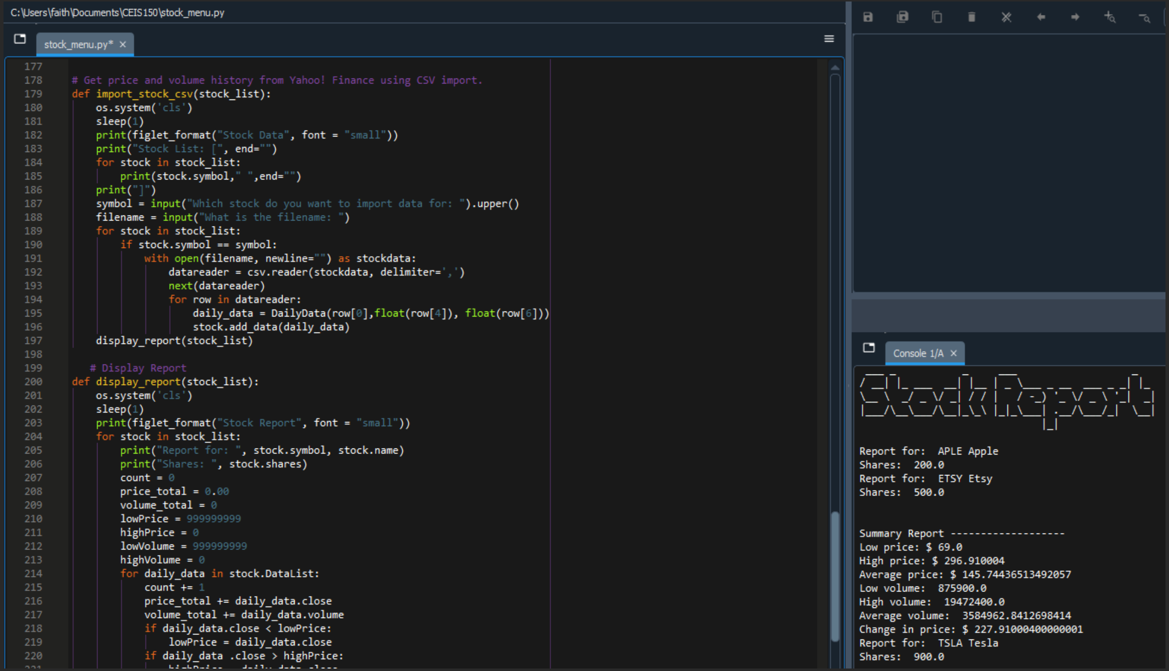
File location



Name	Date modified	Type	Size
ETSY.csv	10/3/2022 1:49 PM	Microsoft Excel C...	18 KB
GitHubRepo.PNG	9/28/2022 6:31 PM	PNG File	27 KB
input1.txt	10/5/2022 3:58 PM	Text Document	1 KB
Mod2_ss_2.PNG	9/28/2022 2:31 PM	PNG File	20 KB
Mod2_ss_3.PNG	9/28/2022 2:32 PM	PNG File	60 KB
MSFT.csv	10/3/2022 1:46 PM	Microsoft Excel C...	19 KB
NES.jpg	9/30/2022 4:15 PM	JPG File	128 KB
out.pdf	10/9/2022 9:20 AM	Chrome HTML Do...	428 KB
PINS.csv	10/3/2022 1:49 PM	Microsoft Excel C...	18 KB
prices.py	9/28/2022 2:14 PM	Python File	1 KB
prices_ss.PNG	9/28/2022 2:16 PM	PNG File	35 KB
stock_class.py	9/28/2022 2:41 PM	Python File	5 KB
stock_menu.py	10/4/2022 9:56 PM	Python File	10 KB
The Python Standard Library by Example....	9/27/2022 9:53 PM	Chrome HTML Do...	13,649 KB
thursday webinar.txt	10/4/2022 8:02 PM	Text Document	1 KB
TSLA.csv	10/3/2022 1:45 PM	Microsoft Excel C...	19 KB
webScraper.py	10/5/2022 8:14 PM	Python File	2 KB
webScraperOOP.py	10/5/2022 9:11 PM	Python File	2 KB
webScrapingJobs.PNG	10/5/2022 8:45 PM	PNG File	99 KB
WordSort.py	10/5/2022 5:14 PM	Python File	1 KB

38 items | 1 item selected 46.3 KB |

Importing Data



```python

# Get price and volume history from Yahoo! Finance using CSV import.
def import_stock_csv(stock_list):
    os.system('cls')
    sleep(1)
    print(figlet_format("Stock Data", font = "small"))
    print("Stock List: [", end="")
    for stock in stock_list:
        print(stock.symbol," ",end="")
    print("]")
    symbol = input("Which stock do you want to import data for: ").upper()
    filename = input("What is the filename: ")
    for stock in stock_list:
        if stock.symbol == symbol:
            with open(filename, newline="") as stockdata:
                datareader = csv.reader(stockdata, delimiter=',')
                next(datareader)
                for row in datareader:
                    daily_data = DailyData(row[0],float(row[4]), float(row[6]))
                    stock.add_data(daily_data)
    display_report(stock_list)

   # Display Report
def display_report(stock_list):
    os.system('cls')
    sleep(1)
    print(figlet_format("Stock Report", font = "small"))
    for stock in stock_list:
        print("Report for: ", stock.symbol, stock.name)
        print("Shares: ", stock.shares)
        count = 0
        price_total = 0.00
        volume_total = 0
        lowPrice = 999999999
        highPrice = 0
        lowVolume = 999999999
        highVolume = 0
        for daily_data in stock.DataList:
            count += 1
            price_total += daily_data.close
            volume_total += daily_data.volume
            if daily_data.close < lowPrice:
                lowPrice = daily_data.close
            if daily_data .close > highPrice:
                highPrice = daily_data.close
```

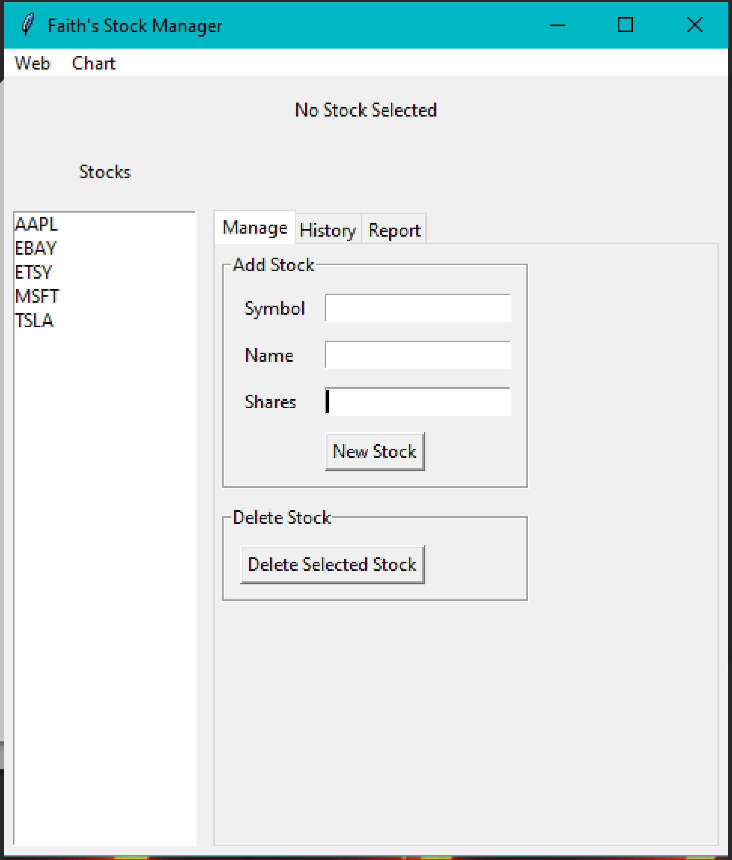
```
 ___ _           _     ___                   _   
/ __| |_ ___  __| |__ | _ \___ _ __  ___ _ _| |_ 
\__ \  _/ _ \/ _| / / |   / -_) '_ \/ _ \ '_|  _|
|___/\__\___/\__|_\_\ |_|_\___| .__/\___/_|  \__|
                              |_|                

Report for:  APLE Apple
Shares:  200.0
Report for:  ETSY Etsy
Shares:  500.0


Summary Report -------------------
Low price: $ 69.0
High price: $ 296.910004
Average price: $ 145.74436513492057
Low volume:  875900.0
High volume:  19472400.0
Average volume:  3584962.8412698414
Change in price: $ 227.91000400000001
Report for:  TSLA Tesla
Shares:  900.0
```

- Define import_stock_csv and display_report methods in stock_menu.py
- Run stock report to test

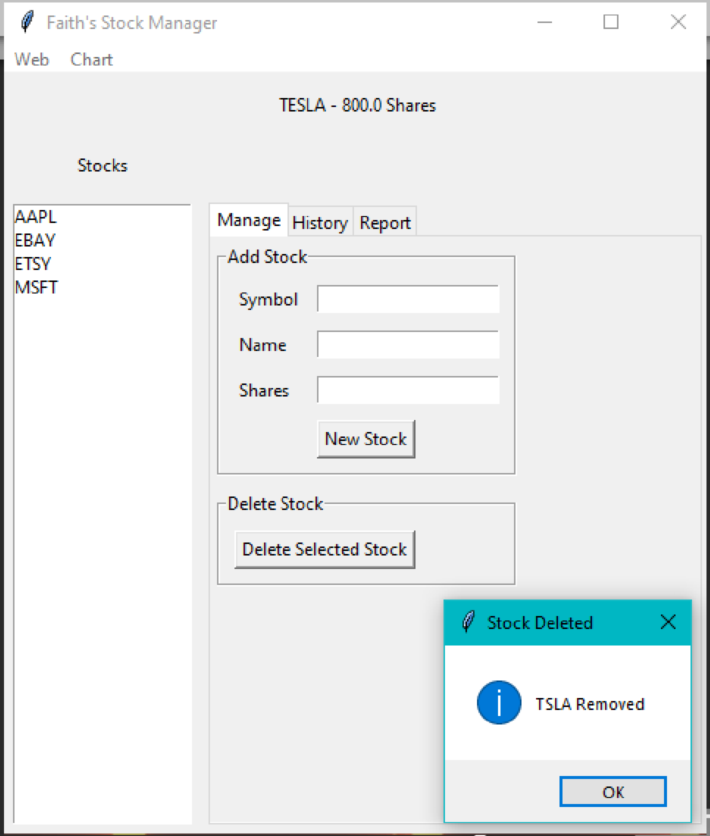
Stock GUI Setup

- Use Tkinter to layout graphical interface in stock_GUI.py
- Add methods from stock_menu.py to stock_GUI.py
- Test program

Add Stock


Faith's Stock Manager
Web
Chart
No Stock Selected
Stocks
AAPL
EBAY
ETSY
MSFT
TSLA
Manage
History
Report
Add Stock
Symbol
Name
Shares
New Stock
Delete Stock
Delete Selected Stock

Delete Stock

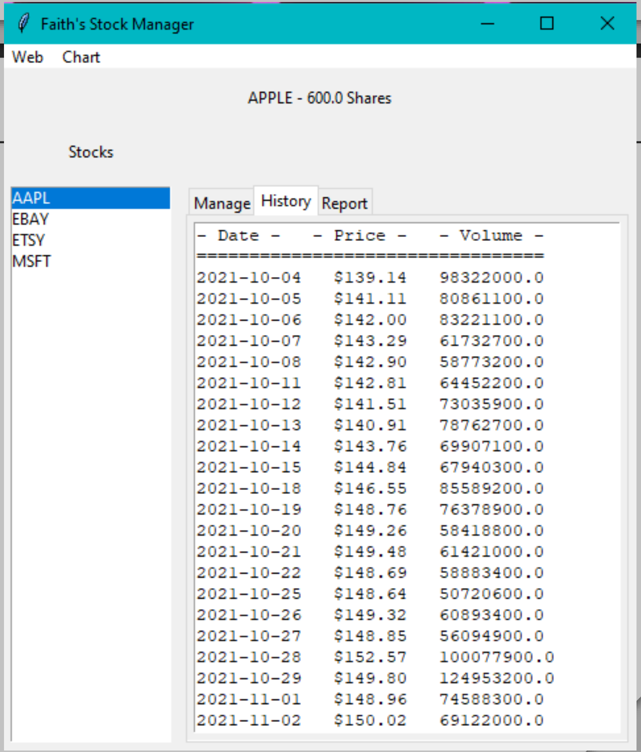


TESLA - 800.0 Shares

AAPL
EBAY
ETSY
MSFT

TSLA Removed

History

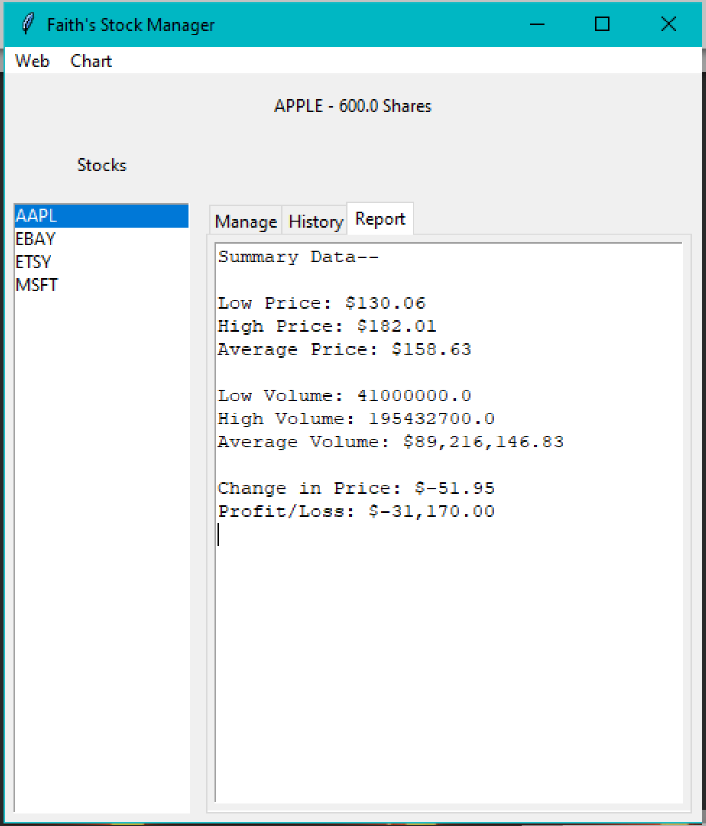


APPLE - 600.0 Shares

Stocks

```
- Date -   - Price -   - Volume -
=================================
2021-10-04   $139.14   98322000.0
2021-10-05   $141.11   80861100.0
2021-10-06   $142.00   83221100.0
2021-10-07   $143.29   61732700.0
2021-10-08   $142.90   58773200.0
2021-10-11   $142.81   64452200.0
2021-10-12   $141.51   73035900.0
2021-10-13   $140.91   78762700.0
2021-10-14   $143.76   69907100.0
2021-10-15   $144.84   67940300.0
2021-10-18   $146.55   85589200.0
2021-10-19   $148.76   76378900.0
2021-10-20   $149.26   58418800.0
2021-10-21   $149.48   61421000.0
2021-10-22   $148.69   58883400.0
2021-10-25   $148.64   50720600.0
2021-10-26   $149.32   60893400.0
2021-10-27   $148.85   56094900.0
2021-10-28   $152.57   100077900.0
2021-10-29   $149.80   124953200.0
2021-11-01   $148.96   74588300.0
2021-11-02   $150.02   69122000.0
```

Report



```
Summary Data--

Low Price: $130.06
High Price: $182.01
Average Price: $158.63

Low Volume: 41000000.0
High Volume: 195432700.0
Average Volume: $89,216,146.83

Change in Price: $-51.95
Profit/Loss: $-31,170.00
```

Challenges

- Trying to clear and pause screen to simulate separate pages for each option
- Combining stock methods with Tkinter properties

Career Skills

- Object-oriented programming
- Building a program using classes.
- Develop a program using inheritance.
- Using Python libraries for data storage, analysis, and data visualization.
- Concepts of functional programming.
- Using iterators and generators.
- Creating Graphical User Interfaces.
- Saving data and web scraping

Thanks!

Do you have any questions?

faithburnett@outlook.com

CREDITS: This presentation template was created by Slidesgo, including icons by Flaticon and infographics & images by Freepik
Please keep this slide for attribution