PORTFOLIO REFLECTION

Master's Program in Information and Communication Technology

ICT 4370 1 Python Programming

Surafeal G Meshesha

University of Denver University College

Date, June 5, 2022

Faculty: Abeer Khaleq, PhD

Director: Molly Smith, PhD

Dean: Michael J. McGuire, MLS

Reflection and Area of Improvements

In general, going into this course with a little coding background, I had some doubts about how it would be a challenging experience. The portfolio assignment was both challenging and eye-opening for me as it made me look for creative methods to enhance the code that has been building up since week two. In this final portfolio project, I have improved the code to be more efficient with handling changes in data (files) and used a loop to plot the values and dates in the graph increasing code efficiency. I have also used the Tkinter toolkit to create an interface for selecting files and paths to read/write. With the added feature we can choose the file we want regardless of the name as long as the data is accurate, thus reducing the "File not Found error".

Additional Useful Functionality

One additional functionality that I am still looking into and would want to include in my future version is to track daily stock progress or prices using the Yahoo Finance API. The Yahoo Finance API can provide us with more information on the stocks and track data as we go (as the date changes).

New Functionalities

In order to implement the new functionality, I needed to learn how to use the Tkinter function. I found lots of resources and tutorials to learn from on the internet, thus making it less challenging to overcome the challenge I faced.

Screenshots of New Functionalities and Improvements

The new functionality adds the feature of a GUI interface to select the file and the file path. The feature is shown below in Figure 1- Selecting a file path.

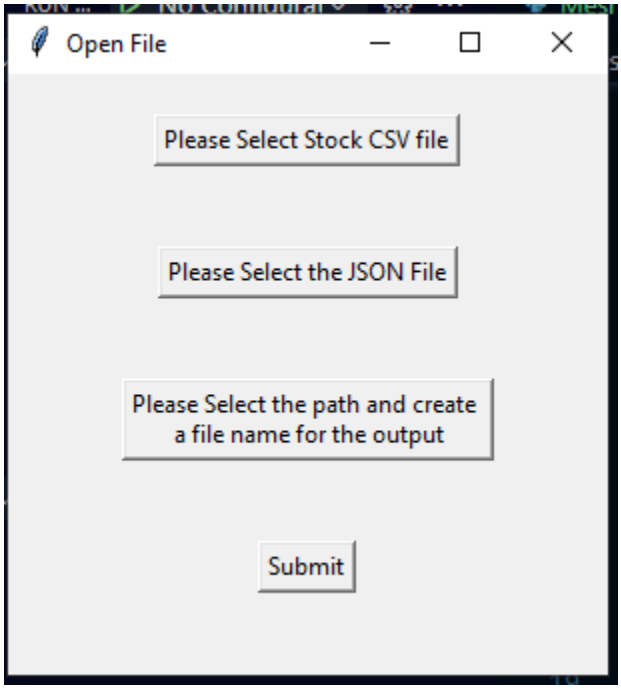

Figure 1-Selecting a file path

In Figure 1-Selecting a file path, we can see three buttons, the first one is used to select the Stock file with the format 'CSV'. While clicking the button we also added the feature of hiding all file that for not has the format of 'CSV'. On the second button where we select the 'JSON' file containing all stock data. Same as the first button, while searching for the JSON file all other types or formats of files will not be displayed. Lastly, the third button is created for the feature of creating the name for the output file.

The code behind the above functionality (Figure 1-Selecting the file path) is displayed below in Figure 2-Tkinter Code.

```
try:
    # Init FileControl object to store paths
    file_control = FileControl()

    # Open tkinter GUI for user
    window = tk.Tk()
    # Default window size/title
    window.geometry('300x300')
    window.title('Open File')

    # Create and pack CSV selection button
    profile_button = tk.Button(window, text = "Please Select Stock CSV file",
                               command=file_control.set_portfolio_path)
    profile_button.pack(pady=20)

    # Create and pack JSON selection button
    information_button = tk.Button(window, text = "Please Select the JSON File",
                                   command=file_control.set_information_path)
    information_button.pack(pady=20)

    # Create and pack PNG output selection button
    output_button = tk.Button(window, text = "Please Select the path and create \n a file name for the output",
                              command=file_control.set_output_path)
    output_button.pack(pady=20)

    # Create and pack Submit button, which closes tkinter GUI
    exit_button = tk.Button(window, text="Submit", command=window.destroy)
    exit_button.pack(pady=20)

    # Keeps tkinter window open until exit_button is clicked
    window.mainloop()
```

Figure 2-Tkinter Code

In Figure 2- Tkinter Code, we can see how the code is structured to be displayed in the GUI. Furthermore, exception handling is also added as a new feature to the code previously submitted. The exception handling in this code will try to use the file selected as an input if the file is in the format required and the data format in the file is accurate even if the values change. After the code tries to execute using the file it will either run or will print the exception error message," An error occurred, please check the file selected" as an output.

```
# Define X axis as dates and remove from dict
x = data['dates']
data.pop('dates')
# Iterate each stock list and create a plot line
for key, value in data.items():

    plt.plot(x, value, label=key)

# Format X axis dates
plt.gcf().autofmt_xdate()
# Add legend
plt.legend(loc='upper left')
# Save plot to disk as PNG
plt.savefig(file_control.get_output_path())
```

Figure 3-Plotting Using Loop

Even though plotting is not an added functionality, I increased code efficiency this week compared to the past weeks by using loops while plotting instead of using multiple statements to plot each stock. Please refer to Figure 3-Plotting Using Loop for better understanding.

```python
class FileControl():
    #Creating the file class to hold and maintain file paths
    def __init__(self):
        self.stock_portfolio_path = None
        self.stock_information_path = None
        self.output_path = None
        self.own_path = os.path.realpath(
            os.path.join(os.getcwd(), os.path.dirname(__file__)))

    def paths_exist(self):
        #Function to determine if all required data was entered

        if (self.stock_information_path != None and self.stock_portfolio_path != None
                and self.output_path != None):
            return True
        return False

    def set_portfolio_path(self):
        #Set path to Stock portfolio CSV file

        # Prompt the dialogbox to only show CSV files
        path = tkinter.filedialog.askopenfilename(initialdir=self.own_path, filetypes=[('CSV Files', '*.csv')])
        self.stock_portfolio_path = path

    def get_portfolio_path(self):
        return self.stock_portfolio_path

    def set_information_path(self):
        #Set path to portfolio CSV

        # Prompt filename dialog in working dir, only show JSON files
        path = tkinter.filedialog.askopenfilename(initialdir=self.own_path, filetypes=[('JSON Files', '*.json')]
```

Figure 4-Class File Control

Moving on from the main code, I have added multiple classes such as 'FileControl', 'Investor', and 'Stock_Timestamp' to my module to handle the path and modify the files which are imported. One of the main classes is the FileControl class. As shown in Figure 4- Class File Control, this class has multiple functions that are used in selecting the path for the input files. This class also has the responsible function to limit and hide other filetypes while selecting both the CSV as well as the JSON files.

If you want to learn more about my code and program, please visit my GITHUB.