Stock Market GUI

Varun Guwal, SY I.T., Division – A, Roll no. – 009, SAP ID - 45207220001

Abstract/Synopsis:

I have developed a program to display the stock market information of five companies from information given on Yahoo Finance, using web scraping techniques. The information is displayed in a GUI, using the CustomTkinter module, with it being updated everytime I execute the program.

Keywords:

Stock Market, Amazon.com Inc., Advanced Micro Devices, Inc., Intel Corporation, NVIDIA Corporation, Apple Inc., CustomTkinter, Web Scraping, Python

Objective and Scope:

The main objective of this program is to display the current stock market information of five companies. The scope is limited to these five companies and displays their prices, change, change percent, volume as well as market cap.

Detailed working of the project:

First, the requests, BeautifulSoup4, locale and customtkinter modules will need to be imported.
The syntax to import them is:
pip install requests beautifulsoup4 customtkinter
The locale module is preinstalled. The requests module is required to send HTTP requests to a website, which in turn returns a Response Object containing the websites content, tags, formatting, etc.
The BeautifulSoup4 module is needed to scrape or to extract information from web pages, or HTML files.
The locale module is used for internalization or localization purposes. In this program, it will be utilized to set the currency of the stock prices, volume and market cap to the USD, which is used as the global currency for international trading.
The customtkinter module, developed by Tom Schimansky has been used to create the GUI.
The url of the most active stocks web page on Yahoo Finance has been mentioned.
Using the get function of the requests module, all of the HTML syntax has been

fetched into the requests object named "fetch". The html content in fetch, is parsed using BeautifulSoup's "html.parser", and the parsed data is stored into the variable soup.

prices, change, change_per, volume and market_cap are dictionaries for storing the eponymous data. The tickers (stock listings of the companies) would be acting as the keys and the corresponding numerical data would be the values.

For scraping the market price-

Using locale's setlocale() function, the localization has been set to USD. The parameter locale.LC_ALL is used to set the settings to the user's default settings. BeautifulSoup4's find function is used to find the first instance of the tag having the specified IDs. This gives us all of the data stored in the tag, and hence we require the attribute ['value'] to scrape the value stored in the tag. However, the returned value is a string, and hence it is typecasted to float.

The currency function groups and localizes the returned value in variable 'amount', and sets it to the specified localization. Lastly, the value has been inserted into the dictionary.

For scraping the market change-

The same method is used as for market price, however, since only a positive or negative number, indicating the growth or decay of the stock will be shown, there is no need to localize it. Lastly for each company, we will update the value as per the corresponding ticker.

For scraping the market change percent-

The value from the specified tag and IDs is scraped, typecasted into float and formatted into percentage with 2 decimal points and updated into the dictionary using the corresponding ticker.

For scraping the market volume-

The value from the tag is attained using the fin-streamer tag and the specified IDs, typecasted into float and localized into the USD currency and updated into the dictionary using the corresponding ticker.

For scraping the market cap-

The value is scraped from the specified tag and IDs, typecasted into float, localized into the USD currency and updated into the dictionary using the corresponding ticker.

The root window of the GUI is created using CTk(). The title of the window is specified using the title function.

The main header of the home tab is specified using the CTkLabel. All widgets are placed using the grid method.

The GUI will be having multiple tabs, which will be created using the CTkTabview function. This is similar to tkinter's notebook widget.

Using the CTkTabview function multiple tabs corresponding to the 5 companies – Amazon.com, AMD Inc. Intel Corp.,

NVIDIA Corp. and Apple Inc. The opening tab can be set using the set function.
Within each tab, the market price, change, change percent, market volume and market cap has been displayed using labels and managed using the grid function.
Lastly, the GUI main window is displayed using mainloop(), which displays the window infinitely, till it is closed by the user.

Use and Purpose:

The program can be used to display current stock market data of the five companies.

Merits and Demerits:

A major merit of this program is that the data can be parsed through quickly in a simple, and easy to read GUI.
However, a major demerit is that the changes in the data are not displayed in real-time, for which the program will need to be closed and executed n number of times.

Future Enhancements:

Getting the data to be updated live in the GUI would be a major area to improve. Additionally, implementing graphs, displaying the historical performance of the companies, within the GUI would also be something I look forward to do.

References:

Think Python: An Introduction to Software Design by Allen Downey

Web References:

https://www.upwork.com/resources/web-scraping-python

https://www.geeksforgeeks.org/difference-between-find-and-find_all-in-beautifulsoup-python/

https://medium.com/@mghasemi5/unleashing-the-data-magic-mastering-web-scraping-with-python-and-beautiful-soup-e2820939c54e

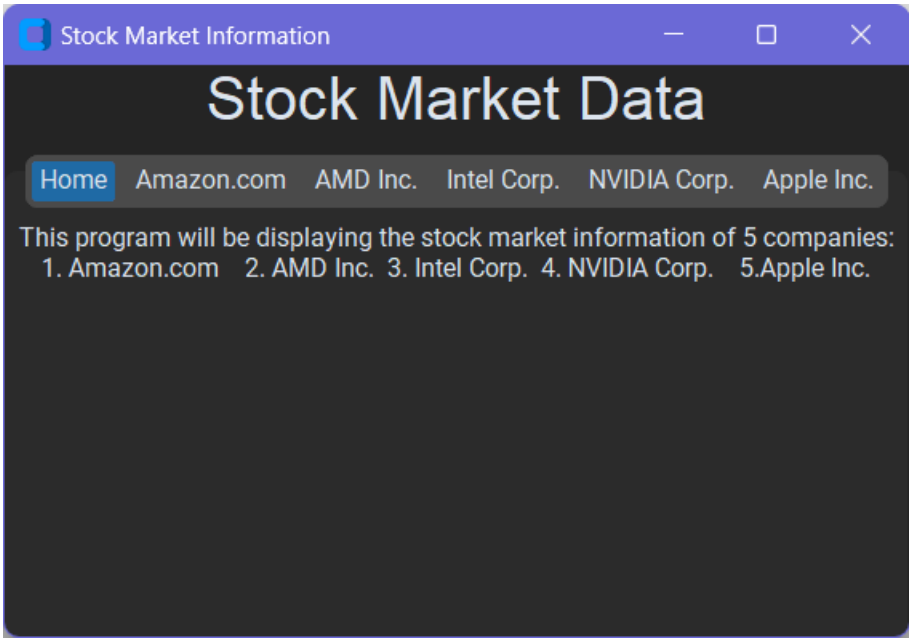
https://customtkinter.tomschimansky.com/documentation/

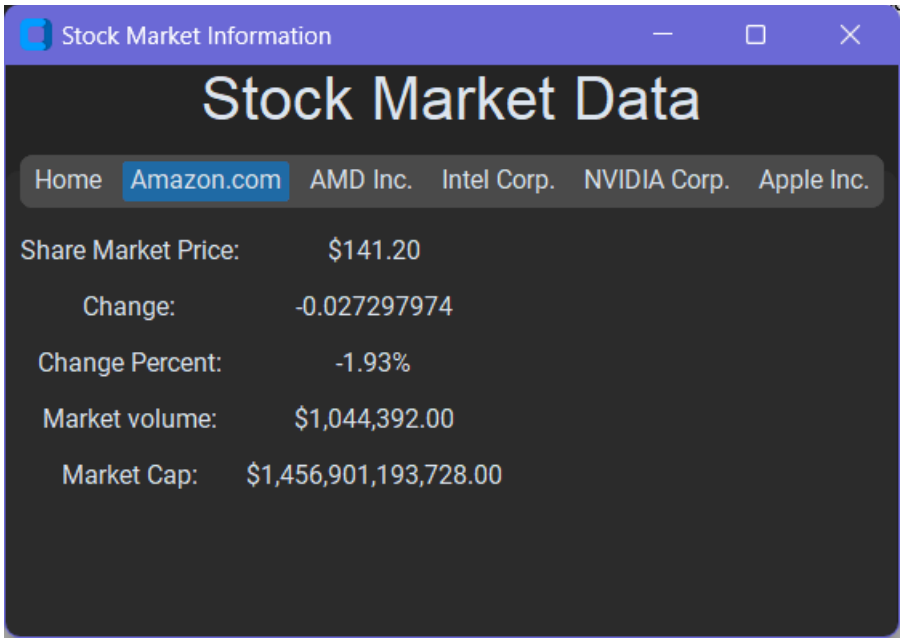
https://tkdocs.com/tutorial/grid.html

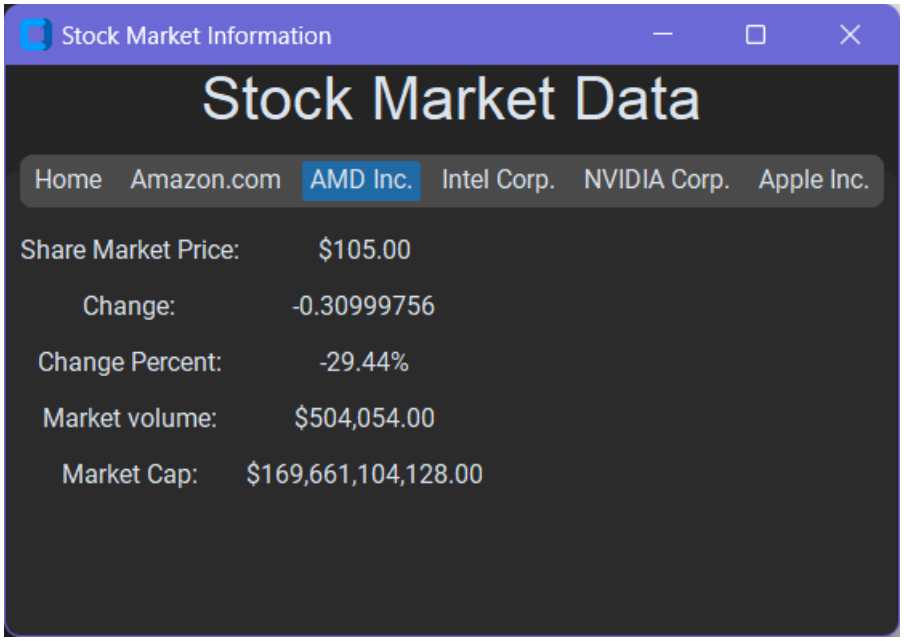
https://www.tutorialstonight.com/python-number-format?expand_article=1

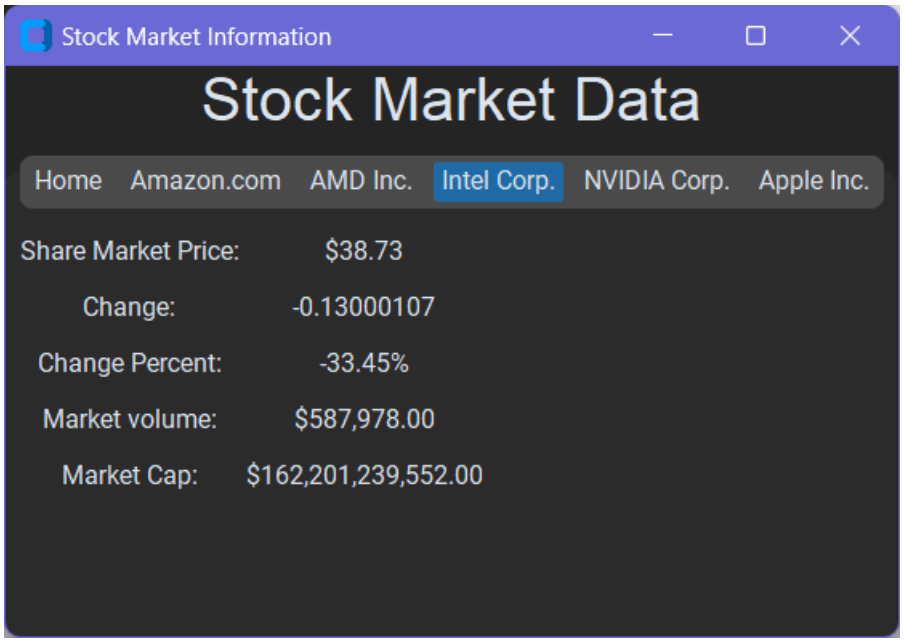
https://phrase.com/blog/posts/beginners-guide-to-locale-in-python/

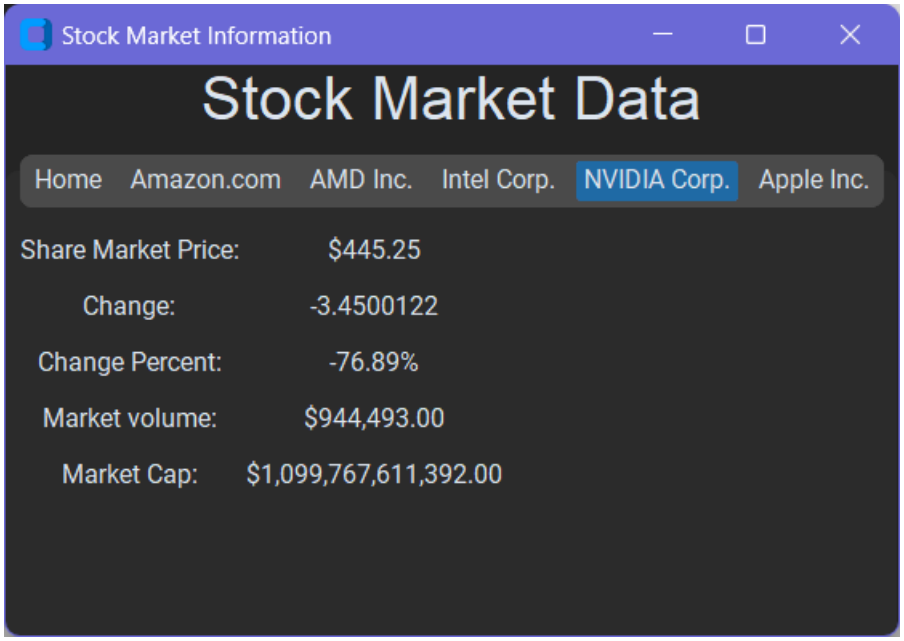
Screenshots:

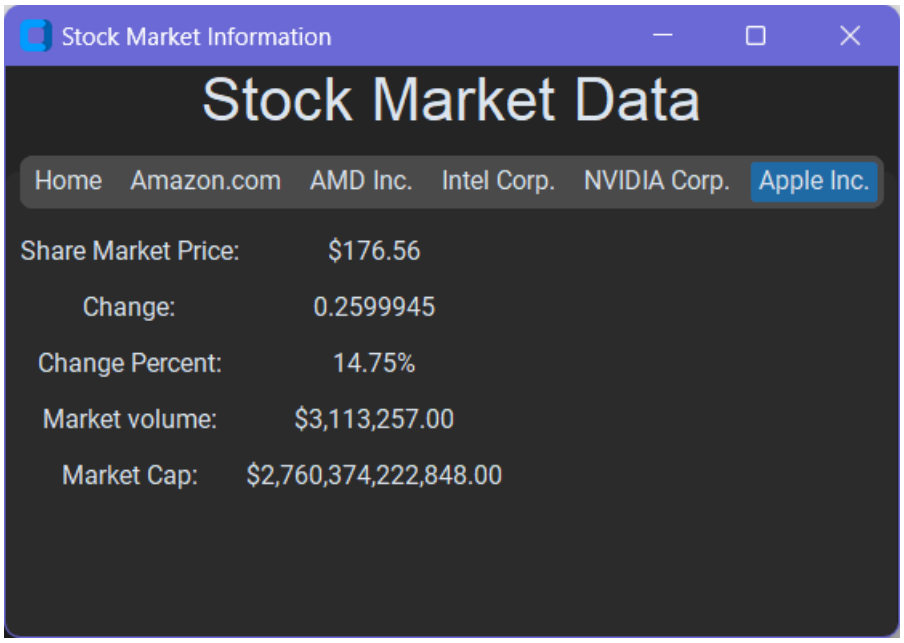





Stock Market Information
Stock Market Data
Home
Amazon.com
AMD Inc.
Intel Corp.
NVIDIA Corp.
Apple Inc.
Share Market Price: $105.00
Change: -0.30999756
Change Percent: -29.44%
Market volume: $504,054.00
Market Cap: $169,661,104,128.00


Stock Market Information
Stock Market Data
Home
Amazon.com
AMD Inc.
Intel Corp.
NVIDIA Corp.
Apple Inc.
Share Market Price: $38.73
Change: -0.13000107
Change Percent: -33.45%
Market volume: $587,978.00
Market Cap: $162,201,239,552.00


Stock Market Information
Stock Market Data
Home
Amazon.com
AMD Inc.
Intel Corp.
NVIDIA Corp.
Apple Inc.
Share Market Price: $445.25
Change: -3.4500122
Change Percent: -76.89%
Market volume: $944,493.00
Market Cap: $1,099,767,611,392.00


Stock Market Information
Stock Market Data
Home
Amazon.com
AMD Inc.
Intel Corp.
NVIDIA Corp.
Apple Inc.
Share Market Price: $176.56
Change: 0.2599945
Change Percent: 14.75%
Market volume: $3,113,257.00
Market Cap: $2,760,374,222,848.00

Source Code:

```
import requests

from bs4 import BeautifulSoup

import locale

from customtkinter import *


url="https://finance.yahoo.com/most-active"


fetch=requests.get(url)


#print(fetch.text)


soup=BeautifulSoup(fetch.content,"html.parser")


#comp=["Amazon.com, Inc.", "Advanced Micro Devices, Inc.", "Intel Corporation",
"NVIDIA Corporation", "Apple Inc."]

prices={}

change={}

change_per={}

volume={}

market_cap={}

locale.setlocale(locale.LC_ALL, 'en_US.UTF-8')


p=soup.find('fin-streamer',{'data-symbol': 'AMZN', 'data-field': 'regularMarketPrice'})
```

```
amount=float(p['value'])
q=locale.currency(amount, grouping=True)
prices.update({"AMZN":q})

p=soup.find('fin-streamer',{'data-symbol': 'AMD', 'data-field': 'regularMarketPrice'})
amount=float(p['value'])
q=locale.currency(amount, grouping=True)
prices.update({"AMD":q})


p=soup.find('fin-streamer',{'data-symbol': 'INTC', 'data-field': 'regularMarketPrice'})
amount=float(p['value'])
q=locale.currency(amount, grouping=True)
prices.update({"INTC":q})


p=soup.find('fin-streamer',{'data-symbol': 'NVDA', 'data-field': 'regularMarketPrice'})
amount=float(p['value'])
q=locale.currency(amount, grouping=True)
prices.update({"NVDA":q})


p=soup.find('fin-streamer',{'data-symbol': 'AAPL', 'data-field': 'regularMarketPrice'})
amount=float(p['value'])
```

```
q=locale.currency(amount, grouping=True)
prices.update({"AAPL":q})


p=soup.find('fin-streamer',{'data-symbol': 'AMZN', 'data-field': 'regularMarketChange'})
p=p['value']
change.update({"AMZN":p})


p=soup.find('fin-streamer',{'data-symbol': 'AMD', 'data-field': 'regularMarketChange'})
p=p['value']
change.update({"AMD":p})


p=soup.find('fin-streamer',{'data-symbol': 'INTC', 'data-field': 'regularMarketChange'})
p=p['value']
change.update({"INTC":p})


p=soup.find('fin-streamer',{'data-symbol': 'NVDA', 'data-field': 'regularMarketChange'})
p=p['value']
change.update({"NVDA":p})
```

```
p=soup.find('fin-streamer',{'data-symbol': 'AAPL', 'data-field': 'regularMarketChange'})

p=p['value']

change.update({"AAPL":p})
```

```
p=soup.find('fin-streamer',{'data-symbol': 'AMZN', 'data-field':
'regularMarketChangePercent'})

p=float(p['value'])

q="{:.2%}".format(p)

change_per.update({"AMZN":q})
```

```
p=soup.find('fin-streamer',{'data-symbol': 'AMD', 'data-field':
'regularMarketChangePercent'})

p=float(p['value'])

q="{:.2%}".format(p)

change_per.update({"AMD":q})
```

```
p=soup.find('fin-streamer',{'data-symbol': 'INTC', 'data-field':
'regularMarketChangePercent'})

p=float(p['value'])

q="{:.2%}".format(p)

change_per.update({"INTC":q})
```

```
p=soup.find('fin-streamer',{'data-symbol': 'NVDA', 'data-field':
'regularMarketChangePercent'})

p=float(p['value'])

q="{:.2%}".format(p)

change_per.update({"NVDA":q})



p=soup.find('fin-streamer',{'data-symbol': 'AAPL', 'data-field':
'regularMarketChangePercent'})

p=float(p['value'])

q="{:.2%}".format(p)

change_per.update({"AAPL":q})



p=soup.find('fin-streamer',{'data-symbol': 'AMZN', 'data-field': 'regularMarketVolume'})

amount=float(p['value'])

q=locale.currency(amount, grouping=True)

volume.update({"AMZN":q})



p=soup.find('fin-streamer',{'data-symbol': 'AMD', 'data-field': 'regularMarketVolume'})

amount=float(p['value'])

q=locale.currency(amount, grouping=True)

volume.update({"AMD":q})
```

```
p=soup.find('fin-streamer',{'data-symbol': 'INTC', 'data-field': 'regularMarketVolume'})
amount=float(p['value'])
q=locale.currency(amount, grouping=True)
volume.update({"INTC":q})


p=soup.find('fin-streamer',{'data-symbol': 'NVDA', 'data-field': 'regularMarketVolume'})
amount=float(p['value'])
q=locale.currency(amount, grouping=True)
volume.update({"NVDA":q})


p=soup.find('fin-streamer',{'data-symbol': 'AAPL', 'data-field': 'regularMarketVolume'})
amount=float(p['value'])
q=locale.currency(amount, grouping=True)
volume.update({"AAPL":q})


p=soup.find('fin-streamer',{'data-symbol': 'AMZN', 'data-field': 'marketCap'})
amount=float(p['value'])
q=locale.currency(amount, grouping=True)
market_cap.update({"AMZN":q})
```

```
p=soup.find('fin-streamer',{'data-symbol': 'AMD', 'data-field': 'marketCap'})
amount=float(p['value'])
q=locale.currency(amount, grouping=True)
market_cap.update({"AMD":q})


p=soup.find('fin-streamer',{'data-symbol': 'INTC', 'data-field': 'marketCap'})
amount=float(p['value'])
q=locale.currency(amount, grouping=True)
market_cap.update({"INTC":q})


p=soup.find('fin-streamer',{'data-symbol': 'NVDA', 'data-field': 'marketCap'})
amount=float(p['value'])
q=locale.currency(amount, grouping=True)
market_cap.update({"NVDA":q})

p=soup.find('fin-streamer',{'data-symbol': 'AAPL', 'data-field': 'marketCap'})
amount=float(p['value'])
q=locale.currency(amount, grouping=True)
market_cap.update({"AAPL":q})
```

```
'''
print("Prices-")
print(prices)
print("Change-")
print(change)
print("Change %-")
print(change_per)
print("Volume")
print(volume)
print("Market Cap")
print(market_cap)
'''

stock=CTk()
stock.title("Stock Market Information")

mainlabel=CTkLabel(stock, text="Stock Market Data", font=(',30))
mainlabel.grid(row=0, column=0)

tabview=CTkTabview(master=stock)
tabview.grid()

tabview.add("Home")
tabview.add("Amazon.com")
```

```
tabview.add("AMD Inc.")

tabview.add("Intel Corp.")

tabview.add("NVIDIA Corp.")

tabview.add("Apple Inc.")

tabview.set("Home")


general="This program will be displaying the stock market information of 5 companies:\n1. Amazon.com    2. AMD Inc.  3. Intel Corp.  4. NVIDIA Corp.    5. Apple Inc."


generalinfo=CTkLabel(master=tabview.tab("Home"), text=general)

generalinfo.grid(row=0, column=0)


AMZN_market_price_label=CTkLabel(master=tabview.tab("Amazon.com"), text="Share Market Price: ", anchor=W)

AMZN_market_price_label.grid(row=0, column=0)


AMZN_market_price_value=CTkLabel(master=tabview.tab("Amazon.com"), text=prices["AMZN"])

AMZN_market_price_value.grid(row=0, column=1)


AMZN_market_change_label=CTkLabel(master=tabview.tab("Amazon.com"), text="Change: ", anchor=W)

AMZN_market_change_label.grid(row=1, column=0)


AMZN_market_change_value=CTkLabel(master=tabview.tab("Amazon.com"), text=change["AMZN"])
```

```
AMZN_market_change_value.grid(row=1, column=1)

AMZN_market_change_per_label=CTkLabel(master=tabview.tab("Amazon.com"),
text="Change Percent: ", anchor=W)

AMZN_market_change_per_label.grid(row=2, column=0)

AMZN_market_change_per_value=CTkLabel(master=tabview.tab("Amazon.com"),
text=change_per["AMZN"])

AMZN_market_change_per_value.grid(row=2, column=1)

AMZN_market_volume_label=CTkLabel(master=tabview.tab("Amazon.com"),
text="Market volume: ", anchor=W)

AMZN_market_volume_label.grid(row=3, column=0)

AMZN_market_volume_value=CTkLabel(master=tabview.tab("Amazon.com"),
text=volume["AMZN"])

AMZN_market_volume_value.grid(row=3, column=1)

AMZN_market_cap_label=CTkLabel(master=tabview.tab("Amazon.com"), text="Market
Cap: ", anchor=W)

AMZN_market_cap_label.grid(row=4, column=0)

AMZN_market_cap_value=CTkLabel(master=tabview.tab("Amazon.com"),
text=market_cap["AMZN"])

AMZN_market_cap_value.grid(row=4, column=1)
```

```
AMD_market_price_label=CTkLabel(master=tabview.tab("AMD Inc."), text="Share Market Price: ", anchor=W)

AMD_market_price_label.grid(row=0, column=0)


AMD_market_price_value=CTkLabel(master=tabview.tab("AMD Inc."), text=prices["AMD"])

AMD_market_price_value.grid(row=0, column=1)


AMD_market_change_label=CTkLabel(master=tabview.tab("AMD Inc."), text="Change: ", anchor=W)

AMD_market_change_label.grid(row=1, column=0)


AMD_market_change_value=CTkLabel(master=tabview.tab("AMD Inc."), text=change["AMD"])

AMD_market_change_value.grid(row=1, column=1)


AMD_market_change_per_label=CTkLabel(master=tabview.tab("AMD Inc."), text="Change Percent: ", anchor=W)

AMD_market_change_per_label.grid(row=2, column=0)


AMD_market_change_per_value=CTkLabel(master=tabview.tab("AMD Inc."), text=change_per["AMD"])

AMD_market_change_per_value.grid(row=2, column=1)


AMD_market_volume_label=CTkLabel(master=tabview.tab("AMD Inc."), text="Market volume: ", anchor=W)
```

```
AMD_market_volume_label.grid(row=3, column=0)

AMD_market_volume_value=CTkLabel(master=tabview.tab("AMD Inc."),
text=volume["AMD"])

AMD_market_volume_value.grid(row=3, column=1)

AMD_market_cap_label=CTkLabel(master=tabview.tab("AMD Inc."), text="Market Cap: ",
anchor=W)

AMD_market_cap_label.grid(row=4, column=0)

AMD_market_cap_value=CTkLabel(master=tabview.tab("AMD Inc."),
text=market_cap["AMD"])

AMD_market_cap_value.grid(row=4, column=1)

INTC_market_price_label=CTkLabel(master=tabview.tab("Intel Corp."), text="Share Market
Price: ", anchor=W)

INTC_market_price_label.grid(row=0, column=0)

INTC_market_price_value=CTkLabel(master=tabview.tab("Intel Corp."),
text=prices["INTC"])

INTC_market_price_value.grid(row=0, column=1)

INTC_market_change_label=CTkLabel(master=tabview.tab("Intel Corp."), text="Change: ",
anchor=W)

INTC_market_change_label.grid(row=1, column=0)
```

```
INTC_market_change_value=CTkLabel(master=tabview.tab("Intel Corp."),
text=change["INTC"])

INTC_market_change_value.grid(row=1, column=1)


INTC_market_change_per_label=CTkLabel(master=tabview.tab("Intel Corp."),
text="Change Percent: ", anchor=W)

INTC_market_change_per_label.grid(row=2, column=0)


INTC_market_change_per_value=CTkLabel(master=tabview.tab("Intel Corp."),
text=change_per["INTC"])

INTC_market_change_per_value.grid(row=2, column=1)


INTC_market_volume_label=CTkLabel(master=tabview.tab("Intel Corp."), text="Market
volume: ", anchor=W)

INTC_market_volume_label.grid(row=3, column=0)


INTC_market_volume_value=CTkLabel(master=tabview.tab("Intel Corp."),
text=volume["INTC"])

INTC_market_volume_value.grid(row=3, column=1)


INTC_market_cap_label=CTkLabel(master=tabview.tab("Intel Corp."), text="Market Cap: ",
anchor=W)

INTC_market_cap_label.grid(row=4, column=0)


INTC_market_cap_value=CTkLabel(master=tabview.tab("Intel Corp."),
text=market_cap["INTC"])
```

```
INTC_market_cap_value.grid(row=4, column=1)


NVDA_market_price_label=CTkLabel(master=tabview.tab("NVIDIA Corp."), text="Share Market Price: ", anchor=W)

NVDA_market_price_label.grid(row=0, column=0)


NVDA_market_price_value=CTkLabel(master=tabview.tab("NVIDIA Corp."), text=prices["NVDA"])

NVDA_market_price_value.grid(row=0, column=1)


NVDA_market_change_label=CTkLabel(master=tabview.tab("NVIDIA Corp."), text="Change: ", anchor=W)

NVDA_market_change_label.grid(row=1, column=0)


NVDA_market_change_value=CTkLabel(master=tabview.tab("NVIDIA Corp."), text=change["NVDA"])

NVDA_market_change_value.grid(row=1, column=1)


NVDA_market_change_per_label=CTkLabel(master=tabview.tab("NVIDIA Corp."), text="Change Percent: ", anchor=W)

NVDA_market_change_per_label.grid(row=2, column=0)


NVDA_market_change_per_value=CTkLabel(master=tabview.tab("NVIDIA Corp."), text=change_per["NVDA"])

NVDA_market_change_per_value.grid(row=2, column=1)
```

```
NVDA_market_volume_label=CTkLabel(master=tabview.tab("NVIDIA Corp."),
text="Market volume: ", anchor=W)

NVDA_market_volume_label.grid(row=3, column=0)


NVDA_market_volume_value=CTkLabel(master=tabview.tab("NVIDIA Corp."),
text=volume["NVDA"])

NVDA_market_volume_value.grid(row=3, column=1)


NVDA_market_cap_label=CTkLabel(master=tabview.tab("NVIDIA Corp."), text="Market
Cap: ", anchor=W)

NVDA_market_cap_label.grid(row=4, column=0)


NVDA_market_cap_value=CTkLabel(master=tabview.tab("NVIDIA Corp."),
text=market_cap["NVDA"])

NVDA_market_cap_value.grid(row=4, column=1)


AAPL_market_price_label=CTkLabel(master=tabview.tab("Apple Inc."), text="Share
Market Price: ", anchor=W)

AAPL_market_price_label.grid(row=0, column=0)


AAPL_market_price_value=CTkLabel(master=tabview.tab("Apple Inc."),
text=prices["AAPL"])

AAPL_market_price_value.grid(row=0, column=1)


AAPL_market_change_label=CTkLabel(master=tabview.tab("Apple Inc."), text="Change: ",
anchor=W)
```

```
AAPL_market_change_label.grid(row=1, column=0)

AAPL_market_change_value=CTkLabel(master=tabview.tab("Apple Inc."),
text=change["AAPL"])

AAPL_market_change_value.grid(row=1, column=1)

AAPL_market_change_per_label=CTkLabel(master=tabview.tab("Apple Inc."),
text="Change Percent: ", anchor=W)

AAPL_market_change_per_label.grid(row=2, column=0)

AAPL_market_change_per_value=CTkLabel(master=tabview.tab("Apple Inc."),
text=change_per["AAPL"])

AAPL_market_change_per_value.grid(row=2, column=1)

AAPL_market_volume_label=CTkLabel(master=tabview.tab("Apple Inc."), text="Market
volume: ", anchor=W)

AAPL_market_volume_label.grid(row=3, column=0)

AAPL_market_volume_value=CTkLabel(master=tabview.tab("Apple Inc."),
text=volume["AAPL"])

AAPL_market_volume_value.grid(row=3, column=1)

AAPL_market_cap_label=CTkLabel(master=tabview.tab("Apple Inc."), text="Market Cap: ",
anchor=W)

AAPL_market_cap_label.grid(row=4, column=0)
```

```
AAPL_market_cap_value=CTkLabel(master=tabview.tab("Apple Inc."),
text=market_cap["AAPL"])

AAPL_market_cap_value.grid(row=4, column=1)


stock.mainloop()
```