Extracting and Visualizing Stock Data

Description

Extracting essential data from a dataset and displaying it is a necessary part of data science; therefore individuals can make correct decisions based on the data. In this assignment, I will extract some stock data, I will then display this data in a graph.

Table of Contents

Estimated Time Needed: **30 min**

In [9]:
```
!pip install yfinance==0.2.44
!pip install pandas==2.2.3
!pip install nbformat
```

```
Requirement already satisfied: yfinance==0.2.44 in c:\users\denni\anaconda3\lib\site-packages (0.2.44)
Requirement already satisfied: pandas>=1.3.0 in c:\users\denni\anaconda3\lib\site-packages (from yfinance==0.2.44) (2.2.3)
Requirement already satisfied: numpy>=1.16.5 in c:\users\denni\anaconda3\lib\site-packages (from yfinance==0.2.44) (1.26.4)
Requirement already satisfied: requests>=2.31 in c:\users\denni\anaconda3\lib\site-packages (from yfinance==0.2.44) (2.31.0)
Requirement already satisfied: multitasking>=0.0.7 in c:\users\denni\anaconda3\lib\site-packages (from yfinance==0.2.44) (0.0.11)
Requirement already satisfied: lxml>=4.9.1 in c:\users\denni\anaconda3\lib\site-packages (from yfinance==0.2.44) (4.9.3)
Requirement already satisfied: platformdirs>=2.0.0 in c:\users\denni\anaconda3\lib\site-packages (from yfinance==0.2.44) (3.10.0)
Requirement already satisfied: pytz>=2022.5 in c:\users\denni\anaconda3\lib\site-packages (from yfinance==0.2.44) (2023.3.post1)
Requirement already satisfied: frozendict>=2.3.4 in c:\users\denni\anaconda3\lib\site-packages (from yfinance==0.2.44) (2.4.6)
Requirement already satisfied: peewee>=3.16.2 in c:\users\denni\anaconda3\lib\site-packages (from yfinance==0.2.44) (3.17.6)
Requirement already satisfied: beautifulsoup4>=4.11.1 in c:\users\denni\anaconda3\lib\site-packages (from yfinance==0.2.44) (4.12.2)
Requirement already satisfied: html5lib>=1.1 in c:\users\denni\anaconda3\lib\site-packages (from yfinance==0.2.44) (1.1)
Requirement already satisfied: soupsieve>1.2 in c:\users\denni\anaconda3\lib\site-packages (from beautifulsoup4>=4.11.1->yfinance==0.2.44) (2.5)
Requirement already satisfied: six>=1.9 in c:\users\denni\anaconda3\lib\site-packages (from html5lib>=1.1->yfinance==0.2.44) (1.16.0)
Requirement already satisfied: webencodings in c:\users\denni\anaconda3\lib\site-packages (from html5lib>=1.1->yfinance==0.2.44) (0.5.1)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\denni\anaconda3\lib\site-packages (from pandas>=1.3.0->yfinance==0.2.44) (2.8.2)
Requirement already satisfied: tzdata>=2022.7 in c:\users\denni\anaconda3\lib\site-packages (from pandas>=1.3.0->yfinance==0.2.44) (2023.3)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\denni\anaconda3\lib\site-packages (from requests>=2.31->yfinance==0.2.44) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\denni\anaconda3\lib\site-packages (from requests>=2.31->yfinance==0.2.44) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\denni\anaconda3\lib\site-packages (from requests>=2.31->yfinance==0.2.44) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\denni\anaconda3\lib\site-packages (from requests>=2.31->yfinance==0.2.44) (2024.2.2)
Requirement already satisfied: pandas==2.2.3 in c:\users\denni\anaconda3\lib\site-packages (2.2.3)
Requirement already satisfied: numpy>=1.23.2 in c:\users\denni\anaconda3\lib\site-packages (from pandas==2.2.3) (1.26.4)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\denni\anaconda3\lib\site-packages (from pandas==2.2.3) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\denni\anaconda3\lib\site-packages (from pandas==2.2.3) (2023.3.post1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\denni\anaconda3\lib\site-packages (from pandas==2.2.3) (2023.3)
Requirement already satisfied: six>=1.5 in c:\users\denni\anaconda3\lib\site-packages (from python-dateutil>=2.8.2->pandas==2.2.3) (1.16.0)
Requirement already satisfied: nbformat in c:\users\denni\anaconda3\lib\site-packages (5.9.2)
Requirement already satisfied: fastjsonschema in c:\users\denni\anaconda3\lib\site-packages (from nbformat) (2.16.2)
Requirement already satisfied: jsonschema>=2.6 in c:\users\denni\anaconda3\lib\site-packages (from nbformat) (4.19.2)
Requirement already satisfied: jupyter-core in c:\users\denni\anaconda3\lib\site-packages (from nbformat) (5.5.0)
Requirement already satisfied: traitlets>=5.1 in c:\users\denni\anaconda3\lib\site-packages (from nbformat) (5.7.1)
Requirement already satisfied: attrs>=22.2.0 in c:\users\denni\anaconda3\lib\site-packages (from jsonschema>=2.6->nbformat) (23.1.0)
Requirement already satisfied: jsonschema-specifications>=2023.03.6 in c:\users\denni\anaconda3\lib\site-packages (from jsonschema>=2.6->nbformat) (2023.7.1)
Requirement already satisfied: referencing>=0.28.4 in c:\users\denni\anaconda3\lib\site-packages (from jsonschema>=2.6->nbformat) (0.30.2)
Requirement already satisfied: rpds-py>=0.7.1 in c:\users\denni\anaconda3\lib\site-packages (from jsonschema>=2.6->nbformat) (0.10.6)
Requirement already satisfied: platformdirs>=2.5 in c:\users\denni\anaconda3\lib\site-packages (from jupyter-core->nbformat) (3.10.0)
Requirement already satisfied: pywin32>=300 in c:\users\denni\anaconda3\lib\site-packages (from jupyter-core->nbformat) (305.1)
```

In [13]:
```
!pip install yfinance
!pip install bs4
!pip install nbformat
```

```
Requirement already satisfied: yfinance in c:\users\denni\anaconda3\lib\site-packages (0.2.44)
Requirement already satisfied: pandas>=1.3.0 in c:\users\denni\anaconda3\lib\site-packages (from yfinance) (2.2.3)
Requirement already satisfied: numpy>=1.16.5 in c:\users\denni\anaconda3\lib\site-packages (from yfinance) (1.26.4)
Requirement already satisfied: requests>=2.31 in c:\users\denni\anaconda3\lib\site-packages (from yfinance) (2.31.0)
Requirement already satisfied: multitasking>=0.0.7 in c:\users\denni\anaconda3\lib\site-packages (from yfinance) (0.0.11)
Requirement already satisfied: lxml>=4.9.1 in c:\users\denni\anaconda3\lib\site-packages (from yfinance) (4.9.3)
Requirement already satisfied: platformdirs>=2.0.0 in c:\users\denni\anaconda3\lib\site-packages (from yfinance) (3.10.0)
Requirement already satisfied: pytz>=2022.5 in c:\users\denni\anaconda3\lib\site-packages (from yfinance) (2023.3.post1)
Requirement already satisfied: frozendict>=2.3.4 in c:\users\denni\anaconda3\lib\site-packages (from yfinance) (2.4.6)
Requirement already satisfied: peewee>=3.16.2 in c:\users\denni\anaconda3\lib\site-packages (from yfinance) (3.17.6)
Requirement already satisfied: beautifulsoup4>=4.11.1 in c:\users\denni\anaconda3\lib\site-packages (from yfinance) (4.12.2)
Requirement already satisfied: html5lib>=1.1 in c:\users\denni\anaconda3\lib\site-packages (from yfinance) (1.1)
Requirement already satisfied: soupsieve>1.2 in c:\users\denni\anaconda3\lib\site-packages (from beautifulsoup4>=4.11.1->yfinance) (2.5)
Requirement already satisfied: six>=1.9 in c:\users\denni\anaconda3\lib\site-packages (from html5lib>=1.1->yfinance) (1.16.0)
Requirement already satisfied: webencodings in c:\users\denni\anaconda3\lib\site-packages (from html5lib>=1.1->yfinance) (0.5.1)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\denni\anaconda3\lib\site-packages (from pandas>=1.3.0->yfinance) (2.8.2)
Requirement already satisfied: tzdata>=2022.7 in c:\users\denni\anaconda3\lib\site-packages (from pandas>=1.3.0->yfinance) (2023.3)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\denni\anaconda3\lib\site-packages (from requests>=2.31->yfinance) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\denni\anaconda3\lib\site-packages (from requests>=2.31->yfinance) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\denni\anaconda3\lib\site-packages (from requests>=2.31->yfinance) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\denni\anaconda3\lib\site-packages (from requests>=2.31->yfinance) (2024.2.2)
Requirement already satisfied: bs4 in c:\users\denni\anaconda3\lib\site-packages (0.0.2)
Requirement already satisfied: beautifulsoup4 in c:\users\denni\anaconda3\lib\site-packages (from bs4) (4.12.2)
Requirement already satisfied: soupsieve>1.2 in c:\users\denni\anaconda3\lib\site-packages (from beautifulsoup4->bs4) (2.5)
Requirement already satisfied: nbformat in c:\users\denni\anaconda3\lib\site-packages (5.9.2)
Requirement already satisfied: fastjsonschema in c:\users\denni\anaconda3\lib\site-packages (from nbformat) (2.16.2)
Requirement already satisfied: jsonschema>=2.6 in c:\users\denni\anaconda3\lib\site-packages (from nbformat) (4.19.2)
Requirement already satisfied: jupyter-core in c:\users\denni\anaconda3\lib\site-packages (from nbformat) (5.5.0)
Requirement already satisfied: traitlets>=5.1 in c:\users\denni\anaconda3\lib\site-packages (from nbformat) (5.7.1)
Requirement already satisfied: attrs>=22.2.0 in c:\users\denni\anaconda3\lib\site-packages (from jsonschema>=2.6->nbformat) (23.1.0)
Requirement already satisfied: jsonschema-specifications>=2023.03.6 in c:\users\denni\anaconda3\lib\site-packages (from jsonschema>=2.6->nbformat) (2023.7.1)
Requirement already satisfied: referencing>=0.28.4 in c:\users\denni\anaconda3\lib\site-packages (from jsonschema>=2.6->nbformat) (0.30.2)
Requirement already satisfied: rpds-py>=0.7.1 in c:\users\denni\anaconda3\lib\site-packages (from jsonschema>=2.6->nbformat) (0.10.6)
Requirement already satisfied: platformdirs>=2.5 in c:\users\denni\anaconda3\lib\site-packages (from jupyter-core->nbformat) (3.10.0)
Requirement already satisfied: pywin32>=300 in c:\users\denni\anaconda3\lib\site-packages (from jupyter-core->nbformat) (305.1)
```

In [30]:
```
import yfinance as yf
import pandas as pd
import requests
from bs4 import BeautifulSoup
import plotly.graph_objects as go
from plotly.subplots import make_subplots
```

In Python, we can ignore warnings using the warnings module. we can use the filterwarnings function to filter or ignore specific warning messages or categories.

In [31]:
```
import warnings
# Ignore all warnings
warnings.filterwarnings("ignore", category=FutureWarning)
```

Define Graphing Function

In this section, I define the function `make_graph`. It takes a dataframe with stock data (dataframe must contain Date and Close columns), a dataframe with revenue data (dataframe must contain Date and Revenue columns), and the name of the stock.

In [80]:
```
def make_graph(stock_data, revenue_data, stock):
    fig = make_subplots(rows=2, cols=1, shared_xaxes=True, subplot_titles=("Historical Share Price", "Historical Revenue"), vertical_spacing = .3)
    stock_data_specific = stock_data[stock_data.Date <= '2021--06-14']
```

```
    revenue_data_specific = revenue_data[revenue_data.Date <= '2021-04-30']
    fig.add_trace(go.Scatter(x=pd.to_datetime(stock_data_specific.Date), y=stock_data_specific.Close.astype("float"), name="Share Price"), row=1, col=1)
    fig.add_trace(go.Scatter(x=pd.to_datetime(revenue_data_specific.Date), y=revenue_data_specific.Revenue.astype("float"), name="Revenue"), row=2, col=1)
    fig.update_xaxes(title_text="Date", row=1, col=1)
    fig.update_xaxes(title_text="Date", row=2, col=1)
    fig.update_yaxes(title_text="Price ($US)", row=1, col=1)
    fig.update_yaxes(title_text="Revenue ($US Millions)", row=2, col=1)
    fig.update_layout(showlegend=False,
    height=900,
    title=stock,
    xaxis_rangeslider_visible=True)
    fig.show()
```

Question 1: Use yfinance to Extract Stock Data

Using the `Ticker` function enter the ticker symbol of the stock I want to extract data on to create a ticker object. The stock is Tesla and its ticker symbol is `TSLA`.

In [33]:
```
tesla = yf.Ticker("TSLA")
```

Using the ticker object and the function `history` extract stock information and save it in a dataframe named `tesla_data`. Set the `period` parameter to `"max"` so I get information for the maximum amount of time.

In [35]:
```
tesla_data = tesla.history(period="max")
```

Reset the index using the `reset_index(inplace=True)` function on the tesla_data DataFrame and display the first five rows of the `tesla_data` dataframe using the `head` function.

In [37]:
```
tesla_data.reset_index(inplace=True)
tesla_data.head()
```

Out[37]:

	index	Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
0	0	2010-06-29 00:00:00-04:00	1.266667	1.666667	1.169333	1.592667	281494500	0.0	0.0
1	1	2010-06-30 00:00:00-04:00	1.719333	2.028000	1.553333	1.588667	257806500	0.0	0.0
2	2	2010-07-01 00:00:00-04:00	1.666667	1.728000	1.351333	1.464000	123282000	0.0	0.0
3	3	2010-07-02 00:00:00-04:00	1.533333	1.540000	1.247333	1.280000	77097000	0.0	0.0
4	4	2010-07-06 00:00:00-04:00	1.333333	1.333333	1.055333	1.074000	103003500	0.0	0.0

Question 2: Use Webscraping to Extract Tesla Revenue Data

Use the `requests` library to download the webpage https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/revenue.htm Save the text of the response as a variable named `html_data`.

In [39]:
```
url = "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/revenue.htm"
html_data = requests.get(url).text
```

Parse the html data using `beautiful_soup` using parser i.e `html5lib` or `html.parser`. Make sure to use the `html_data` with the content parameter as follow `html_data.content`.

In [41]:
```
soup = BeautifulSoup(html_data,"html.parser")
soup.find_all('title')
```

Out[41]:
```
[<title>Tesla Revenue 2010-2022 | TSLA | MacroTrends</title>]
```

Using `BeautifulSoup` or the `read_html` function extract the table with `Tesla Revenue` and store it into a dataframe named `tesla_revenue`. The dataframe should have columns `Date` and `Revenue`.

In [42]:
```
tesla_revenue = pd.DataFrame(columns = ['Date', 'Revenue'])
```

Execute the following line to remove the comma and dollar sign from the `Revenue` column.

In [45]:
```
for row in soup.find_all("tbody")[1].find_all("tr"):
    col = row.find_all("td")
    date = col[0].text
    revenue = col[1].text.replace("$", "").replace(",", "")

    new_row = pd.DataFrame({"Date": [date], "Revenue": [revenue]})
    tesla_revenue = pd.concat([tesla_revenue, new_row], ignore_index=True)
```

Execute the following lines to remove an null or empty strings in the Revenue column.

In [46]:
```
tesla_revenue.dropna(inplace=True)

tesla_revenue = tesla_revenue[tesla_revenue['Revenue'] != ""]
```

Display the last 5 row of the `tesla_revenue` dataframe using the `tail` function.

In [47]:
```
tesla_revenue.tail()
```

Out[47]:

	Date	Revenue
156	2010-09-30	31
157	2010-06-30	28
158	2010-03-31	21
160	2009-09-30	46
161	2009-06-30	27

Question 3: Use yfinance to Extract Stock Data

Using the `Ticker` function enter the ticker symbol of the stock we want to extract data on to create a ticker object. The stock is GameStop and its ticker symbol is `GME`.

In [64]:
```
gamestop = yf.Ticker("GME")
```

Using the ticker object and the function `history` extract stock information and save it in a dataframe named `gme_data`. Set the `period` parameter to `"max"` so we get information for the maximum amount of time.

In [65]:
```
gme_data = gamestop.history(period="max")
```

Reset the index using the `reset_index(inplace=True)` function on the gme_data DataFrame and display the first five rows of the `gme_data` dataframe using the `head` function.

In [67]:
```
gme_data.reset_index(inplace=True)
gme_data.head()
```

Out[67]:

	index	Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
0	0	2002-02-13 00:00:00-05:00	1.620129	1.693350	1.603296	1.691667	76216000	0.0	0.0
1	1	2002-02-14 00:00:00-05:00	1.712707	1.716074	1.670626	1.683250	11021600	0.0	0.0
2	2	2002-02-15 00:00:00-05:00	1.683250	1.687458	1.658001	1.674834	8389600	0.0	0.0
3	3	2002-02-19 00:00:00-05:00	1.666418	1.666418	1.578047	1.607504	7410400	0.0	0.0
4	4	2002-02-20 00:00:00-05:00	1.615920	1.662210	1.603296	1.662210	6892800	0.0	0.0

Question 4: Use Webscraping to Extract GME Revenue Data

Use the `requests` library to download the webpage https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/stock.html. Save the text of the response as a variable named `html_data_2`.

In [69]:
```
url1 = "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/stock.html"
html_data_2 = requests.get(url1).text
```

Parse the html data using `beautiful_soup` using parser i.e `html5lib` or `html.parser`.

In [71]:
```
soup = BeautifulSoup(html_data_2, "html.parser")
soup.find_all('title')
```

Out[71]:
```
[<title>GameStop Revenue 2006-2020 | GME | MacroTrends</title>]
```

Using `BeautifulSoup` or the `read_html` function extract the table with `GameStop Revenue` and store it into a dataframe named `gme_revenue`. The dataframe should have columns `Date` and `Revenue`. Make sure the comma and dollar sign is removed from the `Revenue` column.

> **Note: Use the method similar to what you did in question 2.**

In [75]:
```
gme_revenue = pd.DataFrame(columns=['Date', 'Revenue'])

for table in soup.find_all('table'):

    if ('GameStop Quarterly Revenue' in table.find('th').text):
        rows = table.find_all('tr')

        for row in rows:
            col = row.find_all('td')

            if col != []:
                date = col[0].text
                revenue = col[1].text.replace(',','').replace('$','')
```

```
        new_row = pd.DataFrame({"Date": [date], "Revenue": [revenue]})

        gme_revenue = pd.concat([gme_revenue, new_row], ignore_index=True)
```

Display the last five rows of the `gme_revenue` dataframe using the `tail` function.

In [76]:
```
gme_revenue.tail()
```

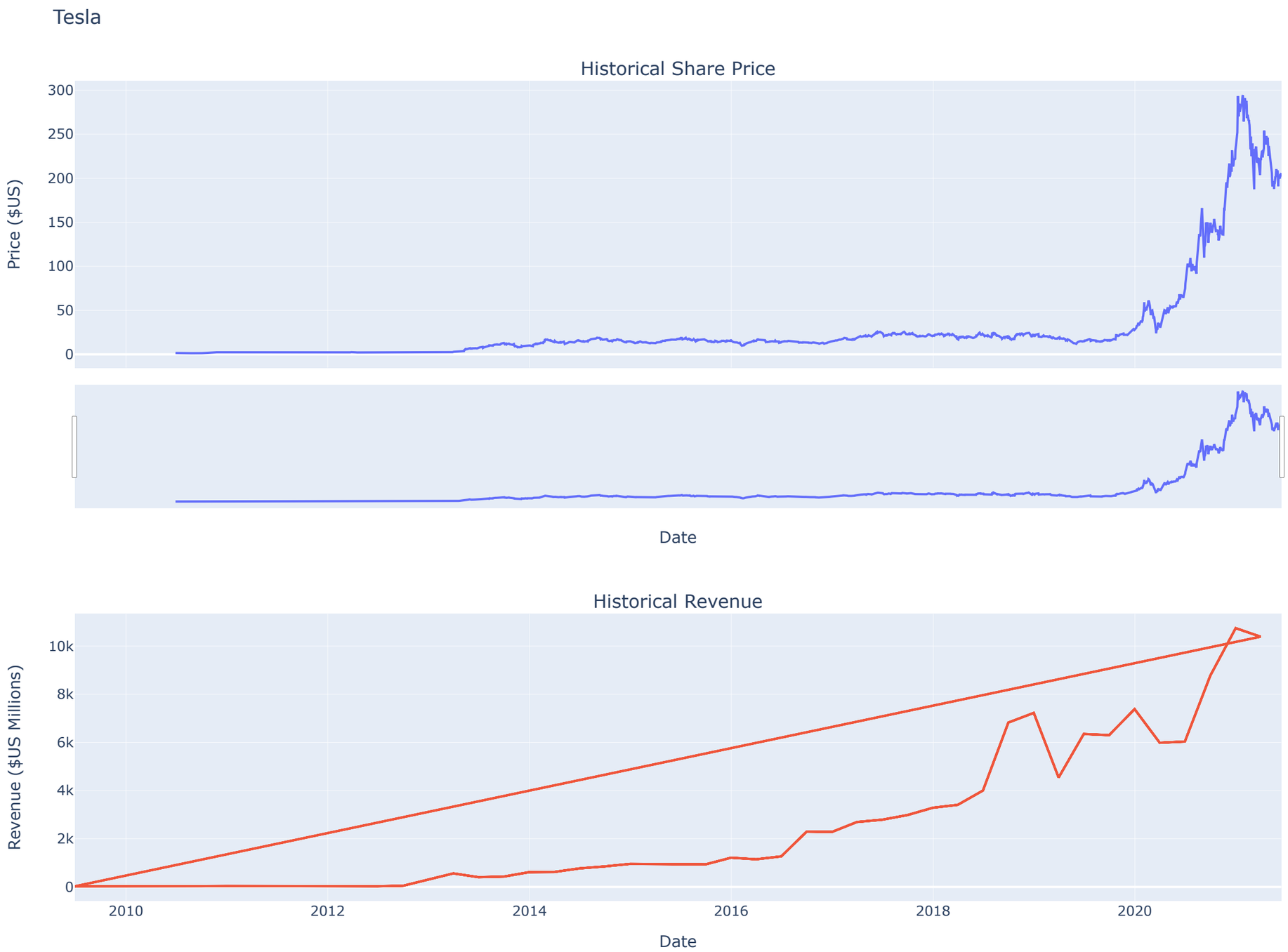
Out[76]:

	Date	Revenue
57	2006-01-31	1667
58	2005-10-31	534
59	2005-07-31	416
60	2005-04-30	475
61	2005-01-31	709

Question 5: Plot Tesla Stock Graph

Use the `make_graph` function to graph the Tesla Stock Data, also provide a title for the graph. Note the graph will only show data upto June 2021.

In [61]:
```
make_graph (tesla_data, tesla_revenue, 'Tesla')
```



Question 6: Plot GameStop Stock Graph

Use the `make_graph` function to graph the GameStop Stock Data, also provide a title for the graph. The structure to call the `make_graph` function is `make_graph(gme_data, gme_revenue, 'GameStop')` . Note the graph will only show data upto June 2021.

In [78]:
```
make_graph (gme_data, gme_revenue, 'GameStop')
```