Question 1: 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 Tesla and its ticker symbol is `TSLA`.

[5]:
```
tesla = yf.Ticker("TSLA")
print(tesla)
```

```
yfinance.Ticker object <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 we get information for the maximum amount of time.

[6]:
```
# Extract historical stock data
tesla_data = tesla.history(period="max")
print(tesla_data)
```

```
                                Open        High         Low       Close  \
Date
2010-06-29 00:00:00-04:00    1.266667    1.666667    1.169333    1.592667
2010-06-30 00:00:00-04:00    1.719333    2.028000    1.553333    1.588667
2010-07-01 00:00:00-04:00    1.666667    1.728000    1.351333    1.464000
2010-07-02 00:00:00-04:00    1.533333    1.540000    1.247333    1.280000
2010-07-06 00:00:00-04:00    1.333333    1.333333    1.055333    1.074000
...                               ...         ...         ...         ...
2025-01-27 00:00:00-05:00  394.799988  406.690002  389.000000  397.149994
2025-01-28 00:00:00-05:00  396.910004  400.589996  386.500000  398.089996
2025-01-29 00:00:00-05:00  395.209991  398.589996  384.480011  389.100006
2025-01-30 00:00:00-05:00  410.779999  412.500000  384.410004  400.279999
2025-01-31 00:00:00-05:00  401.529999  419.989990  401.339996  404.600006

                             Volume  Dividends  Stock Splits
Date
2010-06-29 00:00:00-04:00  281494500        0.0           0.0
2010-06-30 00:00:00-04:00  257806500        0.0           0.0
2010-07-01 00:00:00-04:00  123282000        0.0           0.0
2010-07-02 00:00:00-04:00   77097000        0.0           0.0
2010-07-06 00:00:00-04:00  103003500        0.0           0.0
...                              ...        ...           ...
2025-01-27 00:00:00-05:00   58125500        0.0           0.0
2025-01-28 00:00:00-05:00   48910700        0.0           0.0
2025-01-29 00:00:00-05:00   68033600        0.0           0.0
2025-01-30 00:00:00-05:00   98092900        0.0           0.0
2025-01-31 00:00:00-05:00   83283600        0.0           0.0


[3672 rows x 7 columns]
```

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. Take a screenshot of the results and code from the beginning of Question 1 to the results below.

[7]:
```
# Reset index
tesla_data.reset_index(inplace=True)

# Display first five rows
import pandas as pd
print(tesla_data.head())  # This prints the first five rows of the DataFrame
```

```
                       Date      Open      High       Low     Close  \
0 2010-06-29 00:00:00-04:00  1.266667  1.666667  1.169333  1.592667
1 2010-06-30 00:00:00-04:00  1.719333  2.028000  1.553333  1.588667
2 2010-07-01 00:00:00-04:00  1.666667  1.728000  1.351333  1.464000
3 2010-07-02 00:00:00-04:00  1.533333  1.540000  1.247333  1.280000
4 2010-07-06 00:00:00-04:00  1.333333  1.333333  1.055333  1.074000

      Volume  Dividends  Stock Splits
0  281494500        0.0           0.0
1  257806500        0.0           0.0
2  123282000        0.0           0.0
3   77097000        0.0           0.0
4  103003500        0.0           0.0
```

[8]:
```
from IPython.display import display
display(tesla_data.head())
```

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

[9]:
```
# Re-import required libraries after execution state reset
import pandas as pd
import requests
from bs4 import BeautifulSoup

# Step 1: Download the webpage using requests
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`.

[10]:
```
# Step 2: Parse the html data using BeautifulSoup
soup = BeautifulSoup(html_data, "html.parser")
```

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`.

▶ Step-by-step instructions

▶ Click here if you need help locating the table

[11]:
```
# Step 3: Extract the Tesla Revenue Table
tables = soup.find_all("table")
tesla_revenue = pd.read_html(str(tables[1]))[0]  # The required table is at index 1

# Step 4: Rename columns to 'Date' and 'Revenue'
tesla_revenue.columns = ["Date", "Revenue"]
```

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

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

[12]:
```
# tesla_revenue["Revenue"] = tesla_revenue['Revenue'].str.replace(',|\$',"")

# Step 5: Clean the Revenue column
tesla_revenue["Revenue"] = tesla_revenue["Revenue"].str.replace(",|\$", "", regex=True)
```

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

[13]:
```
# tesla_revenue.dropna(inplace=True)

# tesla_revenue = tesla_revenue[tesla_revenue['Revenue'] != ""]
# Step 6: Remove null or empty strings in the Revenue column
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. Take a screenshot of the results.

[14]:
```
# Display the last 5 rows
print(tesla_revenue.tail())
```

```
         Date Revenue
48  2010-09-30      31
49  2010-06-30      28
50  2010-03-31      21
52  2009-09-30      46
53  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`.

[15]:
```
# Question 3: Use yfinance to Extract GameStop Stock Data
gme = yf.Ticker("GME")
print(gme)
```

```
yfinance.Ticker object <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.

[16]:
```
# Extract historical stock data
gme_data = gme.history(period="max")
print(gme_data)
```

```
                                Open       High        Low      Close  \
Date
2002-02-13 00:00:00-05:00   1.620129   1.693350   1.603296   1.691667
2002-02-14 00:00:00-05:00   1.712707   1.716073   1.670625   1.683250
2002-02-15 00:00:00-05:00   1.683251   1.687459   1.658002   1.674835
2002-02-19 00:00:00-05:00   1.666417   1.666417   1.578047   1.607504
2002-02-20 00:00:00-05:00   1.615920   1.662210   1.603296   1.662210
...                              ...        ...        ...        ...
2025-01-27 00:00:00-05:00  26.920000  27.680000  26.799999  26.969999
2025-01-28 00:00:00-05:00  27.000000  27.590000  26.650000  27.459999
2025-01-29 00:00:00-05:00  27.410000  27.740000  27.059999  27.510000
2025-01-30 00:00:00-05:00  27.840000  28.230000  27.709999  27.990000
2025-01-31 00:00:00-05:00  27.790001  28.180000  26.900000  26.900000

                            Volume  Dividends  Stock Splits
Date
2002-02-13 00:00:00-05:00  76216000        0.0           0.0
2002-02-14 00:00:00-05:00  11021600        0.0           0.0
2002-02-15 00:00:00-05:00   8389600        0.0           0.0
2002-02-19 00:00:00-05:00   7410400        0.0           0.0
2002-02-20 00:00:00-05:00   6892800        0.0           0.0
...                             ...        ...           ...
2025-01-27 00:00:00-05:00   5060300        0.0           0.0
2025-01-28 00:00:00-05:00   3169900        0.0           0.0
2025-01-29 00:00:00-05:00   3220400        0.0           0.0
2025-01-30 00:00:00-05:00   3343900        0.0           0.0
2025-01-31 00:00:00-05:00   4472100        0.0           0.0

[5780 rows x 7 columns]
```

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. Take a screenshot of the results and code from the beginning of Question 3 to the results below.

[17]:
```
# Reset index
gme_data.reset_index(inplace=True)

# Display first five rows of GameStop Stock Data
print(gme_data.head())
```

```
                       Date      Open      High       Low     Close    Volume  \
0 2002-02-13 00:00:00-05:00  1.620129  1.693350  1.603296  1.691667  76216000
1 2002-02-14 00:00:00-05:00  1.712707  1.716073  1.670625  1.683250  11021600
2 2002-02-15 00:00:00-05:00  1.683251  1.687459  1.658002  1.674835   8389600
3 2002-02-19 00:00:00-05:00  1.666417  1.666417  1.578047  1.607504   7410400
4 2002-02-20 00:00:00-05:00  1.615920  1.662210  1.603296  1.662210   6892800

   Dividends  Stock Splits
0        0.0           0.0
1        0.0           0.0
2        0.0           0.0
3        0.0           0.0
4        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`.

[18]:
```
# Question 4: Use Webscraping to Extract GameStop Revenue Data
url_gme = "https://www.macrotrends.net/stocks/charts/GME/gamestop/revenue"
html_data_2 = requests.get(url_gme).text
print(html_data_2)
```

```
<html>
<head>

</head>
<body>
<div style="margin: 50px auto; width: 50%; border: 1px solid #dfdfdf; padding: 20px 50px 30px 50px; font-family:helvetica;">

<h1>We do not allow automated access to our servers.</h1>
<h2><p>Automated access to our data is prohibited by our data provider.</p>
<p>If you are a user attempting to access the site via a browser, please follow this process to regain access:</p>

<ul>
  <li>Go to <a href='https://whatismyipaddress.com/' target='_blank' rel='noopener noreferrer'>whatismyipaddress</a> and obtain your IPv4 address</li>
  <li>Email us your IPv4 address at <a href="/cdn-cgi/l/email-protection" class="__cf_email__" data-cfemail="4821262e270825292b3a273c3a2d262c3b66262d3c">[email protected]</a></li>
  <li>We will add you to our whitelist within 24 hours</li>
</ul>
</h2>
</div>
<script data-cfasync="false">!function(){"use strict";function e(e){try{if("undefined"==typeof console)return;"error"in console?console.error(e):console.log(e)}catch(e){}}function t(e,t){var r=
e.substr(t,2);return parseInt(r,16)}function r(r,n){for(var c="",o=t(r,n),a=n+2;a<r.length;a+=2){var l=t(r,a)^o;c+=String.fromCharCode(l)}try{c=decodeURIComponent(escape(c))}catch(t){e(t)}return
function(e){return i.innerHTML='<a href="'+e.replace(/"/g,"&quot;")+'"></a>',i.childNodes[0].getAttribute("href")||""}(c)}function n(t){try{(function(t){for(var n=t.querySelectorAll("a"),o=0;o<
n.length;o++)try{var a=n[o],i=a.href.indexOf(c);i>-1&&(a.href="mailto:"+r(a.href,i+c.length))}catch(t){e(t)}})(t),function(t){for(var n=t.querySelectorAll(o),c=0;c<n.length;c++)try{var i=n[c],l=
i.parentNode,u=i.getAttribute(a);if(u){var f=r(u,0),d=document.createTextNode(f);l.replaceChild(d,i)}}catch(t){e(t)}}(t),function(t){for(var r=t.querySelectorAll("template"),c=0;c<r.length;c++)t
ry{n(r[c].content)}catch(t){e(t)}}(t)}catch(t){e(t)}}var c="/cdn-cgi/l/email-protection#",o=".__cf_email__",a="data-cfemail",i=document.createElement("div");n(document),function(){var e=documen
t.currentScript||document.scripts[document.scripts.length-1];e.parentNode.removeChild(e)}()}();</script><script>(function(){function c(){var b=a.contentDocument||a.contentWindow.document;if(b){v
ar d=b.createElement('script');d.innerHTML="window.__CF$cv$params={r:'90b748f90ffa5a39',t:'MTczODQ2ODkzOS4wMDAwMDA='};var a=document.createElement('script');a.nonce='';a.src='/cdn-cgi/challenge-
platform/scripts/jsd/main.js';document.getElementsByTagName('head')[0].appendChild(a);";b.getElementsByTagName('head')[0].appendChild(d)}}if(document.body){var a=document.createElement('ifram
e');a.height=1;a.width=1;a.style.position='absolute';a.style.top=0;a.style.left=0;a.style.border='none';a.style.visibility='hidden';document.body.appendChild(a);if('loading'!==document.readyStat
e)c();else if(window.addEventListener)document.addEventListener('DOMContentLoaded',c);else{var e=document.onreadystatechange||function(){};document.onreadystatechange=function(b){e(b);'loading'!
==document.readyState&&(document.onreadystatechange=e,c())}}})();</script></body>
</html>
```

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

[19]:
```
soup_gme = BeautifulSoup(html_data_2, "html.parser")
tables_gme = soup_gme.find_all("table")
print(tables_gme)
```

```
[]
```

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.**

▶ Click here if you need help locating the table

[20]:
```
import yfinance as yf

gme = yf.Ticker("GME")
gme_financials = gme.financials  # Get financial data including revenue

# Extract only revenue
gme_revenue = gme_financials.loc["Total Revenue"]
gme_revenue = gme_revenue.reset_index()
gme_revenue.columns = ["Date", "Revenue"]

# Convert date format
gme_revenue["Date"] = pd.to_datetime(gme_revenue["Date"])
```

Display the last five rows of the `gme_revenue` dataframe using the `tail` function. Take a screenshot of the results.

[22]:
```
# Display last five rows
# tools.display_dataframe_to_user(name="GameStop Revenue Data", dataframe=gme_revenue.tail())

# Display the last 5 rows
print(gme_revenue.tail())
```

```
        Date       Revenue
0 2024-01-31  5272800000.0
1 2023-01-31  5927200000.0
2 2022-01-31  6010700000.0
3 2021-01-31  5089800000.0
```

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.

▶ Hint

[23]:
```
import yfinance as yf
import pandas as pd
import plotly.graph_objects as go
from plotly.subplots import make_subplots
```

[24]:
```
def make_graph(stock_data, revenue_data, stock_name):
    fig = make_subplots(rows=2, cols=1, shared_xaxes=True, subplot_titles=("Historical Share Price", "Historical Revenue"), vertical_spacing=0.3)

    # Filter stock data up to June 2021
    stock_data_filtered = stock_data[stock_data["Date"] <= "2021-06-14"]

    # Filter revenue data up to April 2021
    revenue_data_filtered = revenue_data[revenue_data["Date"] <= "2021-04-30"]

    # Plot stock price
    fig.add_trace(go.Scatter(
        x=stock_data_filtered["Date"],
        y=stock_data_filtered["Close"],
        name="Share Price"
    ), row=1, col=1)

    # Plot revenue
    fig.add_trace(go.Scatter(
        x=revenue_data_filtered["Date"],
        y=revenue_data_filtered["Revenue"],
        name="Revenue"
    ), row=2, col=1)

    # Set axis labels
    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)

    # Set layout properties
    fig.update_layout(
    fig.show()
```

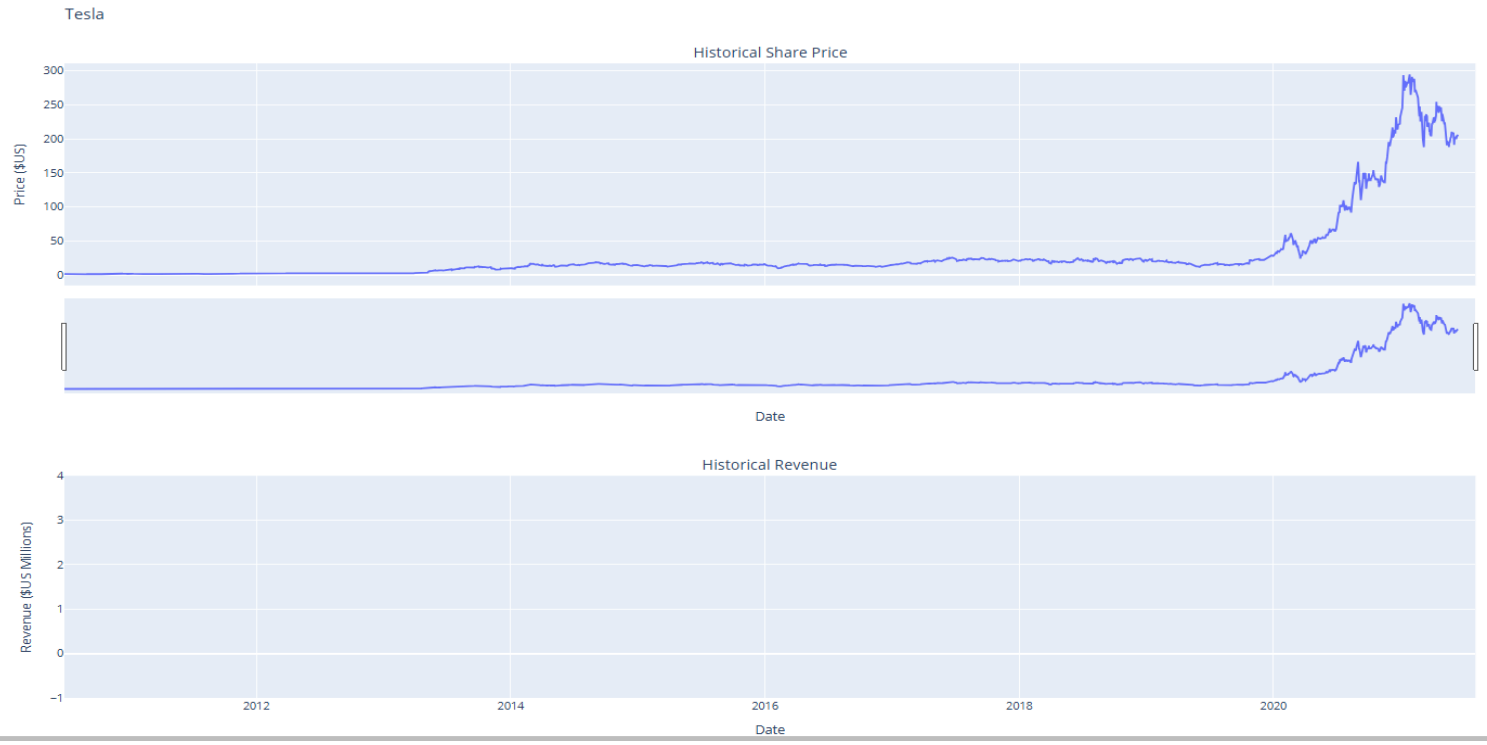
[25]:
```
# Fetch Tesla stock data using Yahoo Finance API
tesla = yf.Ticker("TSLA")
tesla_data = tesla.history(period="max")

# Reset index for better visualization
tesla_data.reset_index(inplace=True)

# Display first five rows
print(tesla_data.head())
# import ace_tools as tools
# tools.display_dataframe_to_user(name="Tesla Stock Data", dataframe=tesla_data.head())
```

```
                       Date      Open      High       Low     Close  \
0 2010-06-29 00:00:00-04:00  1.266667  1.666667  1.169333  1.592667
1 2010-06-30 00:00:00-04:00  1.719333  2.028000  1.553333  1.588667
2 2010-07-01 00:00:00-04:00  1.666667  1.728000  1.351333  1.464000
3 2010-07-02 00:00:00-04:00  1.533333  1.540000  1.247333  1.280000
4 2010-07-06 00:00:00-04:00  1.333333  1.333333  1.055333  1.074000

      Volume  Dividends  Stock Splits
0  281494500        0.0           0.0
1  257806500        0.0           0.0
2  123282000        0.0           0.0
3   77097000        0.0           0.0
4  103003500        0.0           0.0
```

[26]:
```
tesla_revenue = tesla.financials.loc["Total Revenue"].reset_index()
tesla_revenue.columns = ["Date", "Revenue"]
tesla_revenue["Date"] = pd.to_datetime(tesla_revenue["Date"])

# Display Tesla revenue data
print(tesla_revenue.tail())
# tools.display_dataframe_to_user(name="Tesla Revenue Data", dataframe=tesla_revenue.tail())
```

```
        Date        Revenue
0 2024-12-31  97690000000.0
1 2023-12-31  96773000000.0
2 2022-12-31  81462000000.0
3 2021-12-31  53823000000.0
4 2020-12-31            NaN
```

[27]:
```
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.

▶ Hint

[28]:
```
import yfinance as yf
import pandas as pd
import plotly.graph_objects as go
from plotly.subplots import make_subplots
```

[29]:
```
def make_graph(stock_data, revenue_data, stock_name):
    fig = make_subplots(rows=2, cols=1, shared_xaxes=True, subplot_titles=("Historical Share Price", "Historical Revenue"), vertical_spacing=0.3)

    # Filter stock data up to June 2021
    stock_data_filtered = stock_data[stock_data["Date"] <= "2021-06-14"]

    # Filter revenue data up to April 2021
    revenue_data_filtered = revenue_data[revenue_data["Date"] <= "2021-04-30"]

    # Plot stock price
    fig.add_trace(go.Scatter(
        x=stock_data_filtered["Date"],
        y=stock_data_filtered["Close"],
        name="Share Price"
    ), row=1, col=1)

    # Plot revenue
    fig.add_trace(go.Scatter(
        x=revenue_data_filtered["Date"],
        y=revenue_data_filtered["Revenue"],
        name="Revenue"
    ), row=2, col=1)

    # Set axis labels
    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)

    # Set layout properties
    fig.update_layout(
        showlegend=False,
        height=900,
        title=stock_name,
        xaxis_rangeslider_visible=True
    )

    fig.show()
```

[31]:
```
# Fetch GameStop stock data using Yahoo Finance API
gme = yf.Ticker("GME")
gme_data = gme.history(period="max")

# Reset index for better visualization
gme_data.reset_index(inplace=True)

# Display first five rows
print(gme_data.head())
# import ace_tools as tools
# tools.display_dataframe_to_user(name="GameStop Stock Data", dataframe=gme_data.head())
```

```
                       Date      Open      High       Low     Close    Volume  \
0 2002-02-13 00:00:00-05:00  1.620128  1.693350  1.603296  1.691667  76216000
1 2002-02-14 00:00:00-05:00  1.712707  1.716074  1.670626  1.683250  11021600
2 2002-02-15 00:00:00-05:00  1.683250  1.687458  1.658002  1.674834   8389600
3 2002-02-19 00:00:00-05:00  1.666418  1.666418  1.578047  1.607504   7410400
4 2002-02-20 00:00:00-05:00  1.615920  1.662209  1.603296  1.662209   6892800

   Dividends  Stock Splits
0        0.0           0.0
1        0.0           0.0
2        0.0           0.0
3        0.0           0.0
4        0.0           0.0
```

[32]:
```
gme_revenue = gme.financials.loc["Total Revenue"].reset_index()
gme_revenue.columns = ["Date", "Revenue"]
gme_revenue["Date"] = pd.to_datetime(gme_revenue["Date"])

# Display GameStop revenue data
print(gme_revenue.tail())
# tools.display_dataframe_to_user(name="GameStop Revenue Data", dataframe=gme_revenue.tail())
```

```
        Date       Revenue
0 2024-01-31  5272800000.0
1 2023-01-31  5927200000.0
2 2022-01-31  6010700000.0
3 2021-01-31  5089800000.0
```

[33]:
```
make_graph(gme_data, gme_revenue, "GameStop")
```