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.

Table of Contents

In [1]:
```
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
```

Define Graphing Function

In this section, we will define the function `make_graph`.

In [2]:
```
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 <= '2022--07-23']
    revenue_data_specific = revenue_data[revenue_data.Date <= '2022-04-30']
    fig.add_trace(go.Scatter(x=pd.to_datetime(stock_data_specific.Date, infer_datetime_format=True), 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, infer_datetime_format=True), 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()
```

Extracting and Plotting "Tesla" Stock and Revenue Data.

Using 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`.

In [3]:
```
tesla = yf.Ticker('TSLA')
tesla_data = tesla.history(period='max')
tesla_data.head(2)
```

Out[3]:

Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
2010-06-29	3.800	5.000	3.508	4.778	93831500	0	0.0
2010-06-30	5.158	6.084	4.660	4.766	85935500	0	0.0

In [4]:
```
# Resetting the index:
tesla_data.reset_index(inplace=True)
tesla_data.head(5)
```

Out[4]:

	Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
0	2010-06-29	3.800	5.000	3.508	4.778	93831500	0	0.0
1	2010-06-30	5.158	6.084	4.660	4.766	85935500	0	0.0
2	2010-07-01	5.000	5.184	4.054	4.392	41094000	0	0.0
3	2010-07-02	4.600	4.620	3.742	3.840	25699000	0	0.0
4	2010-07-06	4.000	4.000	3.166	3.222	34334500	0	0.0

Using Webscraping to Extract Tesla Revenue Data.

Using the `requests` library to download the webpage https://www.macrotrends.net/stocks/charts/TSLA/tesla/revenue.

In [5]:
```
url = 'https://www.macrotrends.net/stocks/charts/TSLA/tesla/revenue'
html_data = requests.get(url).text
```

Parsing the html data using `beautiful_soup`.

In [6]:
```
soup = BeautifulSoup(html_data, 'html5lib')
# This is the code 'soup.find_all("tbody")[1]' to isolate the table.
```

There are two methods extracting the data into dataframe.

Using `BeautifulSoup` or the `read_html` function extract the data and store it into a dataframe.

In [7]:
```
# First method: BeautifulSoup
# Looping through the table to extract the data:

tables = soup.find_all('table')
for index,table in enumerate(tables):
    if ("Tesla Quarterly Revenue" in str(table)):
        table_index = index
```

```
print(table_index)
# print(tables[table_index].prettify())
```

```
1
```

In [8]:

```
tesla_revenue = pd.DataFrame(columns=["Date", "Revenue"])

for row in tables[table_index].tbody.find_all("tr"):
    col = row.find_all("td")
    if (col != []):
        date = col[0].text
        revenue = col[1].text
        tesla_revenue = tesla_revenue.append({"Date":date, "Revenue":revenue}, ignore_index=True)
tesla_revenue.head(2)
```

Out[8]:

	Date	Revenue
0	2022-06-30	$16,934
1	2022-03-31	$18,756

In [9]:

```
# Second method: pd.read_html

pd.read_html(str(tables[1]), flavor='bs4')
tesla_reve = pd.read_html(str(tables[1]), flavor='bs4')[0]
tesla_reve.head(2)
```

Out[9]:

	Tesla Quarterly Revenue(Millions of US $)	Tesla Quarterly Revenue(Millions of US $).1
0	2022-06-30	$16,934
1	2022-03-31	$18,756

In [10]:

```
# Removing the 'comma' and 'dollar' sign from the 'Revenue' column:
tesla_revenue["Revenue"] = tesla_revenue['Revenue'].str.replace(',',"").str.replace('$',"",  regex=True)
tesla_revenue.head(2)
```

Out[10]:

	Date	Revenue
0	2022-06-30	16934
1	2022-03-31	18756

In [11]:

```
# Removing null or empty values in the 'Revenue' column:
tesla_revenue.dropna(inplace=True)
tesla_revenue = tesla_revenue[tesla_revenue['Revenue'] != ""]
tesla_revenue.shape
```

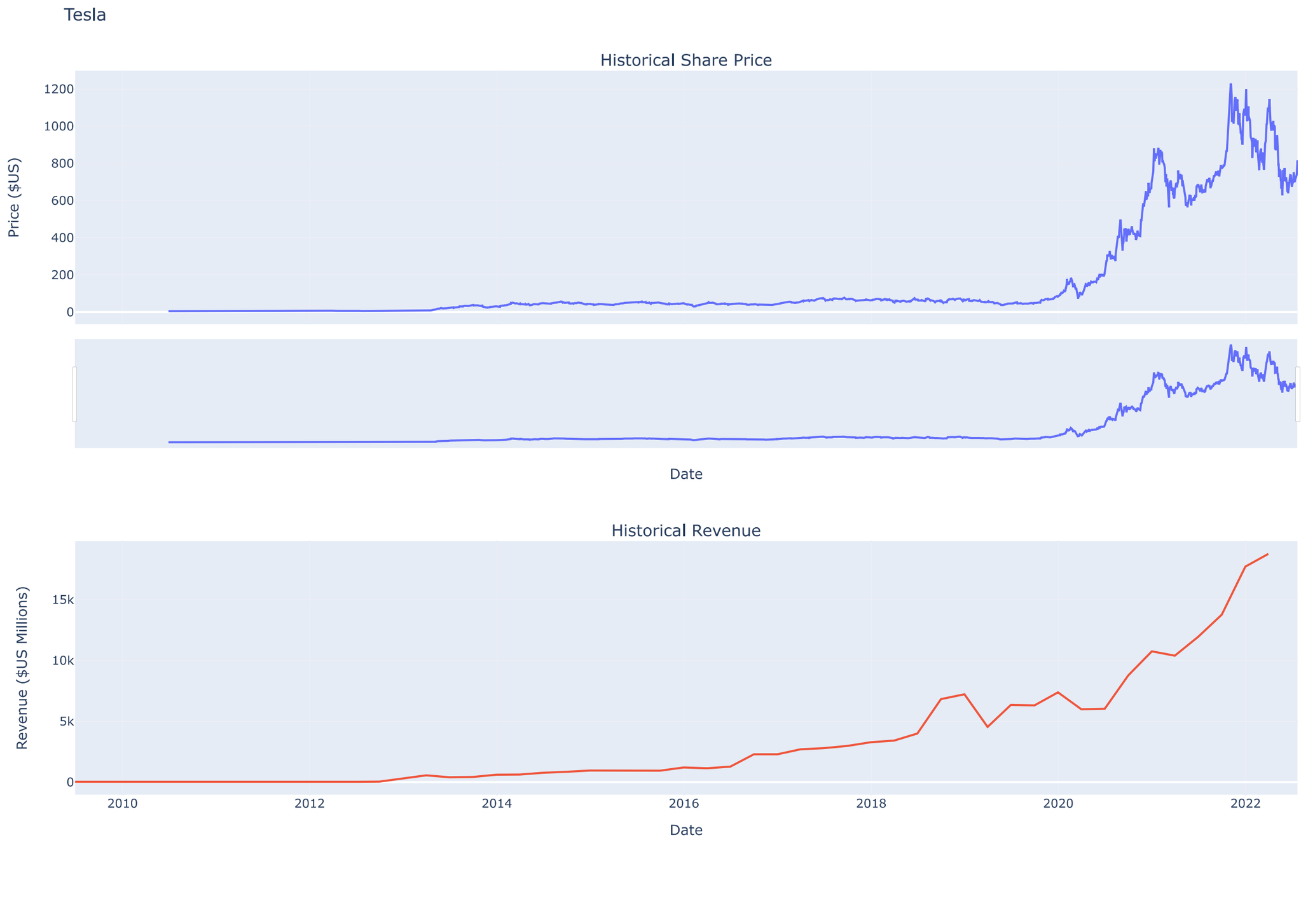
Out[11]:

```
(52, 2)
```

Plotting Tesla Stock Graph

In [12]:

```
make_graph(tesla_data, tesla_revenue, 'Tesla')
```



Extracting and Plotting "GameStop" Stock and Revenue Data.

Using 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 [13]:
```
gme = yf.Ticker('GME')
gme_data = gme.history(period='max')
gme_data.reset_index(inplace=True)
gme_data.head(5)
```

Out[13]:

	Date	Open	High	Low	Close	Volume	Dividends	Stock Splits
0	2002-02-13	1.620129	1.693350	1.603296	1.691667	76216000	0.0	0.0
1	2002-02-14	1.712707	1.716074	1.670626	1.683250	11021600	0.0	0.0
2	2002-02-15	1.683251	1.687459	1.658002	1.674834	8389600	0.0	0.0
3	2002-02-19	1.666418	1.666418	1.578047	1.607504	7410400	0.0	0.0
4	2002-02-20	1.615920	1.662210	1.603296	1.662210	6892800	0.0	0.0

In [14]:
```
# Using Webscraping to Extract GME Revenue Data
url1 = "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-PY0220EN-SkillsNetwork/labs/project/stock.html"
html_data = requests.get(url1).text
# Parsing the html data.
soup1 = BeautifulSoup(html_data, 'html5lib')
```

In [15]:
```
# In this section using "pd.read_html" method:
tables = soup1.find_all('table')
gme_revenue = pd.read_html(str(tables[1]), flavor='bs4')[0]
gme_revenue.columns = ['Date', 'Revenue']
# Replacing the unwanted character to none.
gme_revenue["Revenue"] = gme_revenue['Revenue'].str.replace(',|\$',"", regex=True)
gme_revenue.dropna(inplace=True)
# Extracting the not empty data.
gme_revenue = gme_revenue[gme_revenue['Revenue'] != ""]
gme_revenue.head(2)
```

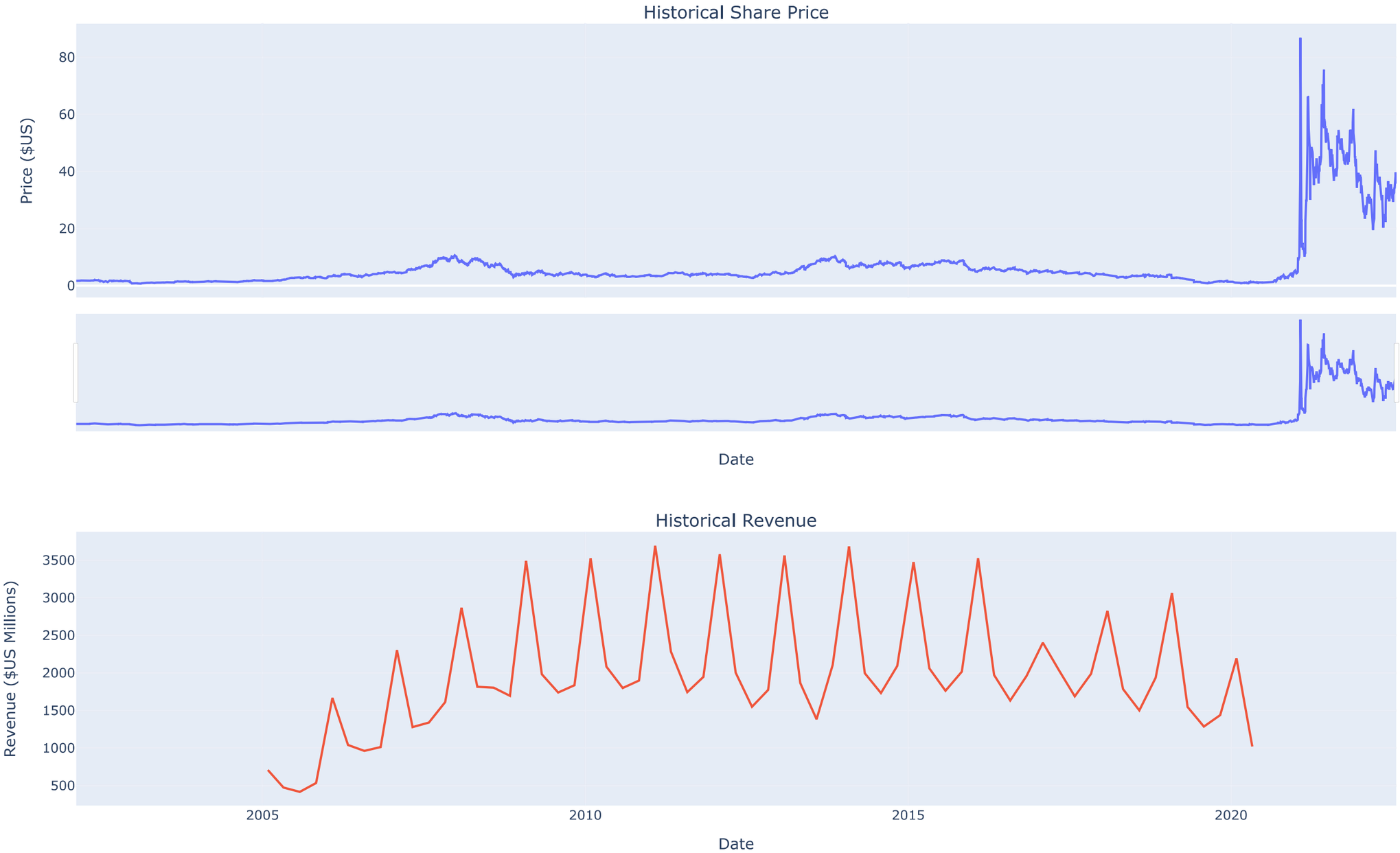
Out[15]:

	Date	Revenue
0	2020-04-30	1021
1	2020-01-31	2194

Plotting GameStop Stock Graph

In [16]:
```
make_graph(gme_data, gme_revenue, 'GameStop')
```

GameStop



End of the Project

In [ ]: