```
In [7]: import yfinance as yf

        # Specify the ticker symbol
        ticker_symbol = "TSLA"

        # Fetch stock data
        stock_data = yf.Ticker(ticker_symbol)

        # Get historical data
        historical_data = stock_data.history(period="1y")  # Adjust the period as needed (1y for 1 year)

        # Print the historical data
        print(historical_data.head())
```

```
                                 Open        High         Low       Close  \
Date
2023-02-27 00:00:00-05:00  202.029999  209.419998  201.259995  207.630005
2023-02-28 00:00:00-05:00  210.589996  211.229996  203.750000  205.710007
2023-03-01 00:00:00-05:00  206.210007  207.199997  198.520004  202.770004
2023-03-02 00:00:00-05:00  186.740005  193.750000  186.009995  190.899994
2023-03-03 00:00:00-05:00  194.800003  200.479996  192.880005  197.789993

                              Volume  Dividends  Stock Splits
Date
2023-02-27 00:00:00-05:00  161028300        0.0           0.0
2023-02-28 00:00:00-05:00  153144900        0.0           0.0
2023-03-01 00:00:00-05:00  156852800        0.0           0.0
2023-03-02 00:00:00-05:00  181500700        0.0           0.0
2023-03-03 00:00:00-05:00  154193300        0.0           0.0
```

In [11]:

```python
import requests
from bs4 import BeautifulSoup

# URL of the webpage containing Tesla revenue data
url = "https://www.macrotrends.net/stocks/charts/TSLA/tesla/revenue"

# Send a GET request to the webpage
response = requests.get(url)

# Check if the request was successful (status code 200)
if response.status_code == 200:
    # Parse the HTML content using BeautifulSoup
    soup = BeautifulSoup(response.text, 'html.parser')

    # Locate the HTML element containing the revenue data (this is just an example)
    revenue_element = soup.find('span', {'class': 'revenue'})

    # Extract the revenue data
    if revenue_element:
        tesla_revenue = revenue_element.text
        print("Tesla Revenue:", tesla_revenue)
    else:
        print("Revenue data not found on the webpage.")

else:
    print("Failed to retrieve webpage. Status code:", response.status_code)
```

```
Failed to retrieve webpage. Status code: 403
```

In [12]:
```python
import yfinance as yf

# Set the ticker symbol for Tesla
ticker_symbol = "TSLA"

# Create a Ticker object
tesla_ticker = yf.Ticker(ticker_symbol)

# Get historical data for the past year
historical_data = tesla_ticker.history(period="1y")

# Print the historical data
print(historical_data.head())
```

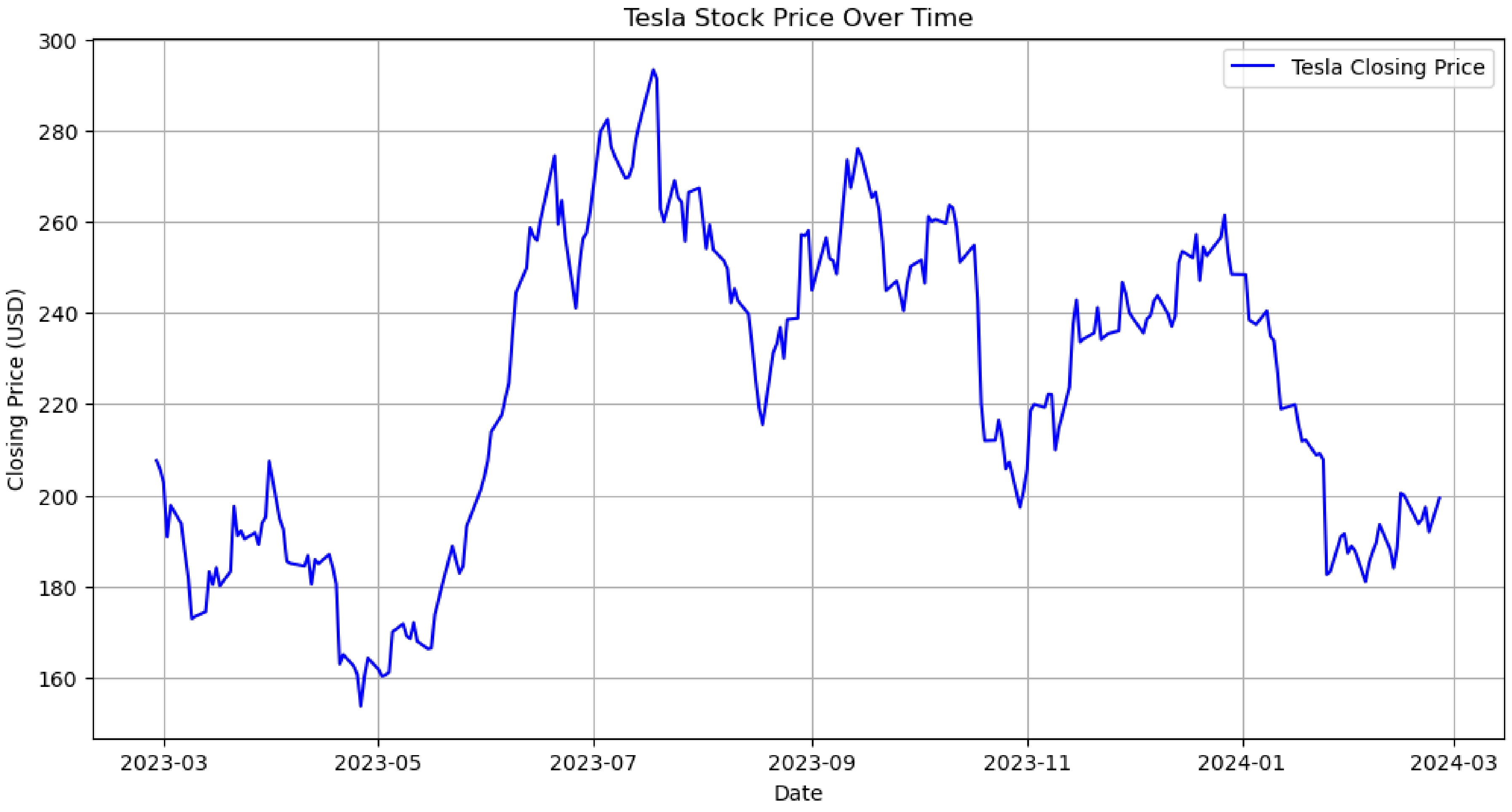
```
                                 Open        High         Low       Close  \
Date
2023-02-27 00:00:00-05:00  202.029999  209.419998  201.259995  207.630005
2023-02-28 00:00:00-05:00  210.589996  211.229996  203.750000  205.710007
2023-03-01 00:00:00-05:00  206.210007  207.199997  198.520004  202.770004
2023-03-02 00:00:00-05:00  186.740005  193.750000  186.009995  190.899994
2023-03-03 00:00:00-05:00  194.800003  200.479996  192.880005  197.789993

                              Volume  Dividends  Stock Splits
Date
2023-02-27 00:00:00-05:00  161028300        0.0           0.0
2023-02-28 00:00:00-05:00  153144900        0.0           0.0
2023-03-01 00:00:00-05:00  156852800        0.0           0.0
2023-03-02 00:00:00-05:00  181500700        0.0           0.0
2023-03-03 00:00:00-05:00  154193300        0.0           0.0
```

In [14]:
```python
import requests
from bs4 import BeautifulSoup

# URL of the webpage containing GME revenue data
url = "https://www.macrotrends.net/stocks/charts/TSLA/tesla/revenue"

# Send a GET request to the webpage
response = requests.get(url)

# Check if the request was successful (status code 200)
if response.status_code == 200:
    # Parse the HTML content using BeautifulSoup
    soup = BeautifulSoup(response.text, 'html.parser')

    # Locate the HTML element containing the revenue data (this is just an example)
    revenue_element = soup.find('span', {'class': 'revenue'})

    # Extract the revenue data
    if revenue_element:
        gme_revenue = revenue_element.text
        print("GME Revenue:", gme_revenue)
    else:
        print("Revenue data not found on the webpage.")

else:
    print("Failed to retrieve webpage. Status code:", response.status_code)
```

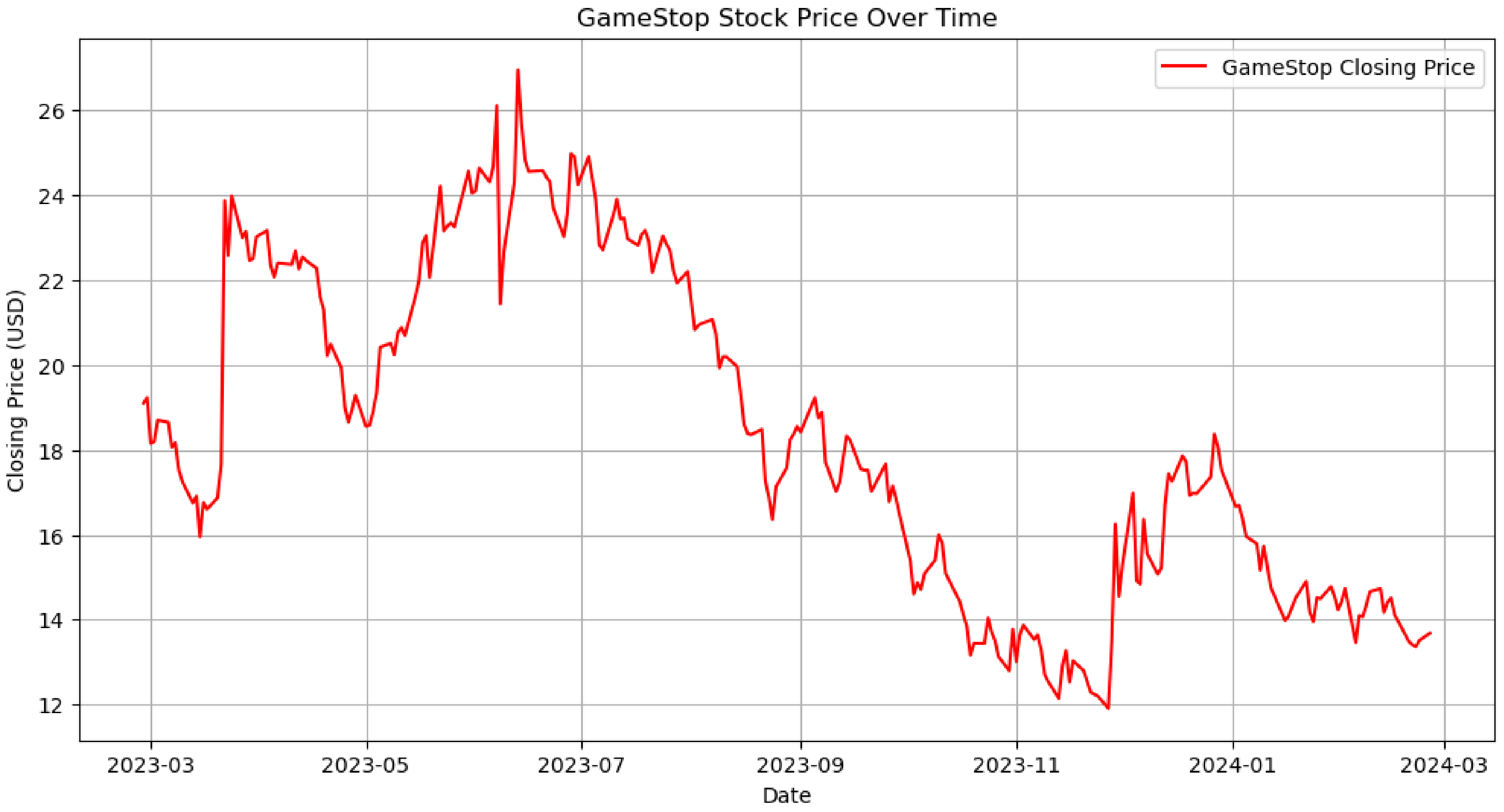
```
Failed to retrieve webpage. Status code: 403
```

In [16]:

```
import yfinance as yf
import matplotlib.pyplot as plt

# Set the ticker symbol for Tesla
ticker_symbol = "TSLA"

# Create a Ticker object
tesla_ticker = yf.Ticker(ticker_symbol)

# Get historical data for the past year
historical_data = tesla_ticker.history(period="1y")

# Plot the closing prices
plt.figure(figsize=(12, 6))
plt.plot(historical_data.index, historical_data['Close'], label='Tesla Closing Price', color='blue')

# Customize the plot
plt.title('Tesla Stock Price Over Time')
plt.xlabel('Date')
plt.ylabel('Closing Price (USD)')
plt.legend()
plt.grid(True)

# Show the plot
plt.show()
```


Tesla Stock Price Over Time
Tesla Closing Price
Closing Price (USD)
Date
300
280
260
240
220
200
180
160
2023-03
2023-05
2023-07
2023-09
2023-11
2024-01
2024-03

In [17]:
```python
import yfinance as yf
import matplotlib.pyplot as plt

# Set the ticker symbol for GameStop
ticker_symbol = "GME"

# Create a Ticker object
gme_ticker = yf.Ticker(ticker_symbol)

# Get historical data for the past year
historical_data = gme_ticker.history(period="1y")

# Plot the closing prices
plt.figure(figsize=(12, 6))
plt.plot(historical_data.index, historical_data['Close'], label='GameStop Closing Price', color='red')

# Customize the plot
plt.title('GameStop Stock Price Over Time')
plt.xlabel('Date')
plt.ylabel('Closing Price (USD)')
plt.legend()
plt.grid(True)

# Show the plot
plt.show()
```


GameStop Stock Price Over Time
GameStop Closing Price
Closing Price (USD)
26
24
22
20
18
16
14
12
2023-03
2023-05
2023-07
2023-09
2023-11
2024-01
2024-03
Date

```
In [ ]:
```