In [1]:
```
1 pip install yfinance
```

```
Requirement already satisfied: yfinance in c:\users\asus\anaconda3\lib\site-p
ackages (0.2.37)Note: you may need to restart the kernel to use updated packa
ges.

Requirement already satisfied: pandas>=1.3.0 in c:\users\asus\anaconda3\lib\s
ite-packages (from yfinance) (2.0.3)
Requirement already satisfied: numpy>=1.16.5 in c:\users\asus\anaconda3\lib\s
ite-packages (from yfinance) (1.24.3)
Requirement already satisfied: requests>=2.31 in c:\users\asus\anaconda3\lib
\site-packages (from yfinance) (2.31.0)
Requirement already satisfied: multitasking>=0.0.7 in c:\users\asus\anaconda3
\lib\site-packages (from yfinance) (0.0.11)
Requirement already satisfied: lxml>=4.9.1 in c:\users\asus\anaconda3\lib\sit
e-packages (from yfinance) (4.9.2)
Requirement already satisfied: appdirs>=1.4.4 in c:\users\asus\anaconda3\lib
\site-packages (from yfinance) (1.4.4)
Requirement already satisfied: pytz>=2022.5 in c:\users\asus\anaconda3\lib\si
te-packages (from yfinance) (2022.7)
Requirement already satisfied: frozendict>=2.3.4 in c:\users\asus\anaconda3\l
ib\site-packages (from yfinance) (2.4.0)
Requirement already satisfied: peewee>=3.16.2 in c:\users\asus\anaconda3\lib
\site-packages (from yfinance) (3.17.1)
Requirement already satisfied: beautifulsoup4>=4.11.1 in c:\users\asus\anacon
da3\lib\site-packages (from yfinance) (4.12.2)
Requirement already satisfied: html5lib>=1.1 in c:\users\asus\anaconda3\lib\s
ite-packages (from yfinance) (1.1)
Requirement already satisfied: soupsieve>1.2 in c:\users\asus\anaconda3\lib\s
ite-packages (from beautifulsoup4>=4.11.1->yfinance) (2.4)
Requirement already satisfied: six>=1.9 in c:\users\asus\anaconda3\lib\site-p
ackages (from html5lib>=1.1->yfinance) (1.16.0)
Requirement already satisfied: webencodings in c:\users\asus\anaconda3\lib\si
te-packages (from html5lib>=1.1->yfinance) (0.5.1)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\asus\anacon
da3\lib\site-packages (from pandas>=1.3.0->yfinance) (2.8.2)
Requirement already satisfied: tzdata>=2022.1 in c:\users\asus\anaconda3\lib
\site-packages (from pandas>=1.3.0->yfinance) (2023.3)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\asus\anac
onda3\lib\site-packages (from requests>=2.31->yfinance) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\asus\anaconda3\lib\si
te-packages (from requests>=2.31->yfinance) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\asus\anaconda3
\lib\site-packages (from requests>=2.31->yfinance) (1.26.16)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\asus\anaconda3
\lib\site-packages (from requests>=2.31->yfinance) (2023.7.22)
```

In [2]:
```
import pandas as pd
import yfinance as yf
from datetime import datetime
import plotly.express as px

```

In [3]:
```
start_date = datetime.now() - pd.DateOffset(months=60)
end_date = datetime.now()

tickers = ['AAPL', 'MSFT']

df_list = []

for ticker in tickers:
    data = yf.download(ticker, start=start_date, end=end_date)
    df_list.append(data)

df = pd.concat(df_list, keys=tickers, names=['Ticker', 'Date'])
print(df.head())
```

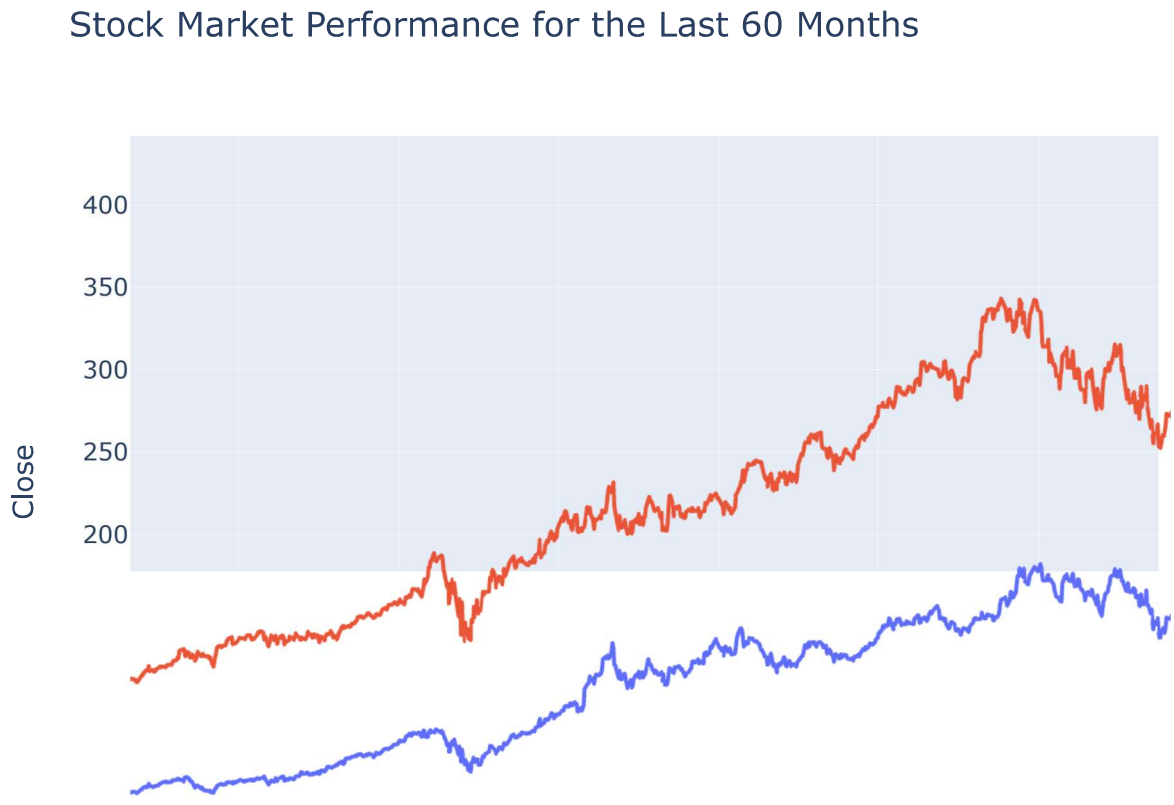
```
[**********100%%**********]  1 of 1 completed
[**********100%%**********]  1 of 1 completed

                        Open       High        Low      Close  Adj Close  \
Ticker Date
AAPL   2019-02-28  43.580002  43.727501  43.230000  43.287498  41.672829
       2019-03-01  43.570000  43.787498  43.222500  43.742500  42.110859
       2019-03-04  43.922501  44.437500  43.492500  43.962502  42.322659
       2019-03-05  43.985001  44.000000  43.634998  43.882500  42.245636
       2019-03-06  43.667500  43.872501  43.485001  43.630001  42.002560

                       Volume
Ticker Date
AAPL   2019-02-28  112861600
       2019-03-01  103544800
       2019-03-04  109744800
       2019-03-05   78949600
       2019-03-06   83241600
```

In [4]:
```
df = df.reset_index()
print(df.head())
```

```
  Ticker       Date       Open       High        Low      Close  Adj Close  \
0   AAPL 2019-02-28  43.580002  43.727501  43.230000  43.287498  41.672829
1   AAPL 2019-03-01  43.570000  43.787498  43.222500  43.742500  42.110859
2   AAPL 2019-03-04  43.922501  44.437500  43.492500  43.962502  42.322659
3   AAPL 2019-03-05  43.985001  44.000000  43.634998  43.882500  42.245636
4   AAPL 2019-03-06  43.667500  43.872501  43.485001  43.630001  42.002560

      Volume
0  112861600
1  103544800
2  109744800
3   78949600
4   83241600
```

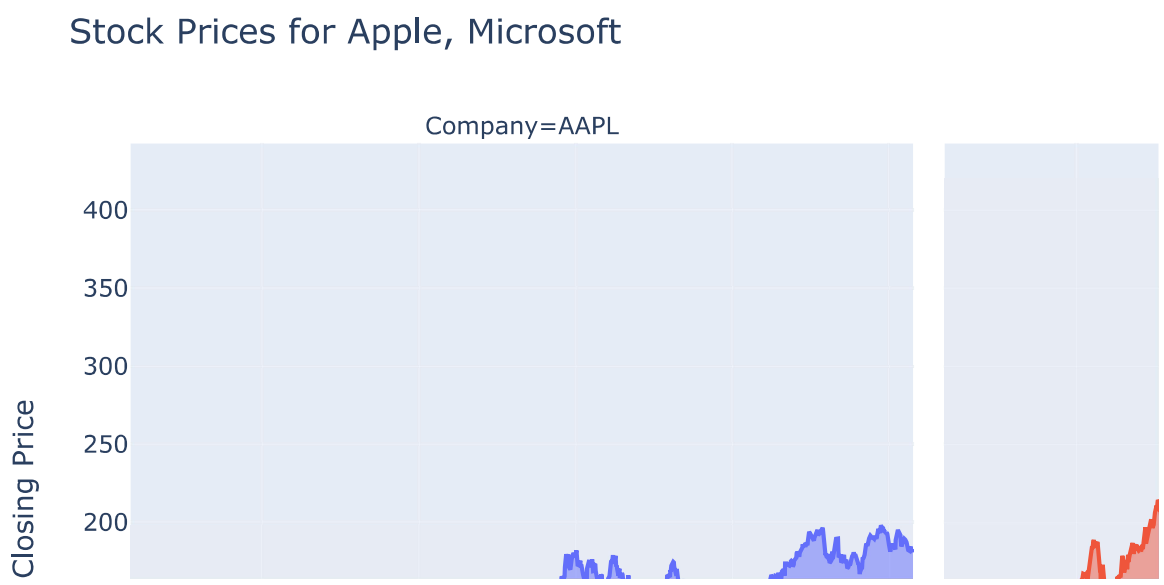
In [5]:

```
fig = px.line(df, x='Date',
              y='Close',
              color='Ticker',
              title="Stock Market Performance for the Last 60 Months")
fig.show()
```



In [6]:

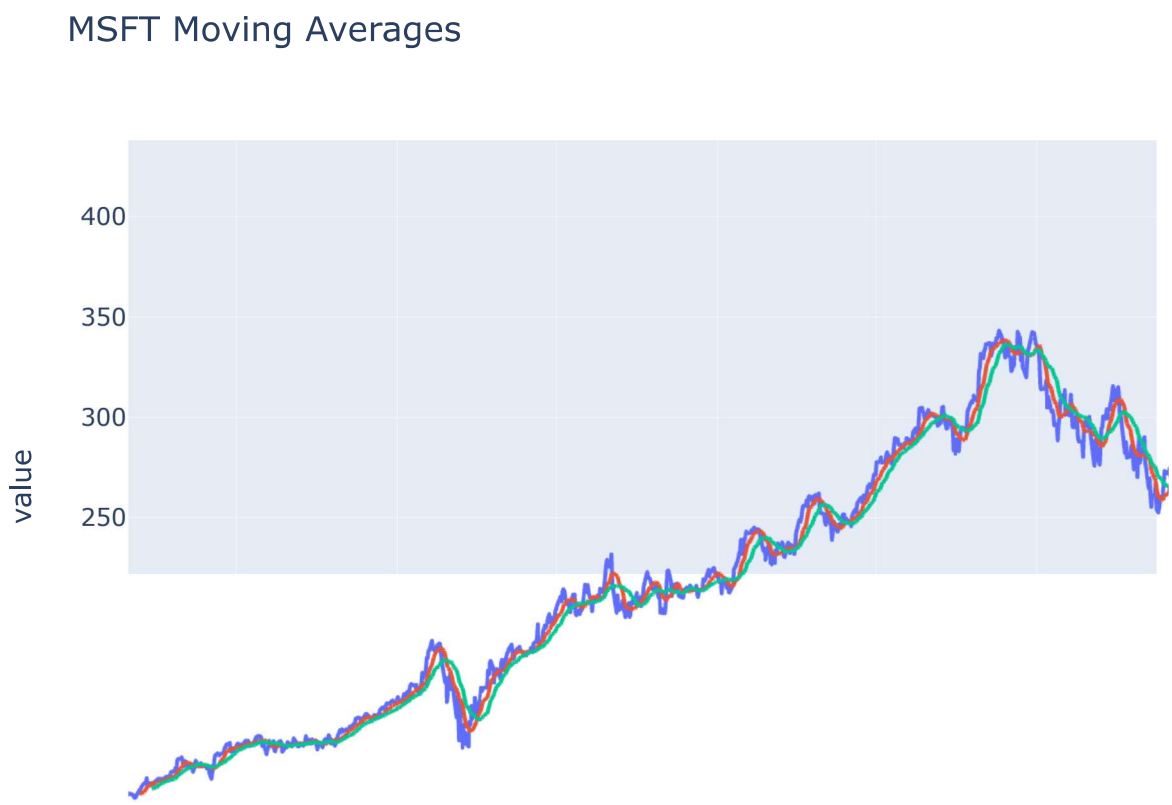
```
fig = px.area(df, x='Date', y='Close', color='Ticker',
              facet_col='Ticker',
              labels={'Date':'Date', 'Close':'Closing Price', 'Ticker':'Co
              title='Stock Prices for Apple, Microsoft')
fig.show()
```



In [7]:
```
df['MA10'] = df.groupby('Ticker')['Close'].rolling(window=10).mean().reset
df['MA20'] = df.groupby('Ticker')['Close'].rolling(window=20).mean().reset

for ticker, group in df.groupby('Ticker'):
    print(f'Moving Averages for {ticker}')
    print(group[['MA10', 'MA20']])
```

```
Moving Averages for AAPL
            MA10        MA20
0            NaN         NaN
1            NaN         NaN
2            NaN         NaN
3            NaN         NaN
4            NaN         NaN
...          ...         ...
1253  185.297000  187.395999
1254  184.792999  186.889499
1255  184.212999  186.306999
1256  183.443999  185.743999
1257  182.992000  185.289000

[1258 rows x 2 columns]
Moving Averages for MSFT
            MA10        MA20
1258         NaN         NaN
1259         NaN         NaN
1260         NaN         NaN
1261         NaN         NaN
1262         NaN         NaN
...          ...         ...
2511  409.536996  407.437996
2512  409.296997  407.892496
2513  408.919998  408.165996
2514  407.619000  408.346497
2515  406.841000  408.234497

[1258 rows x 2 columns]
```

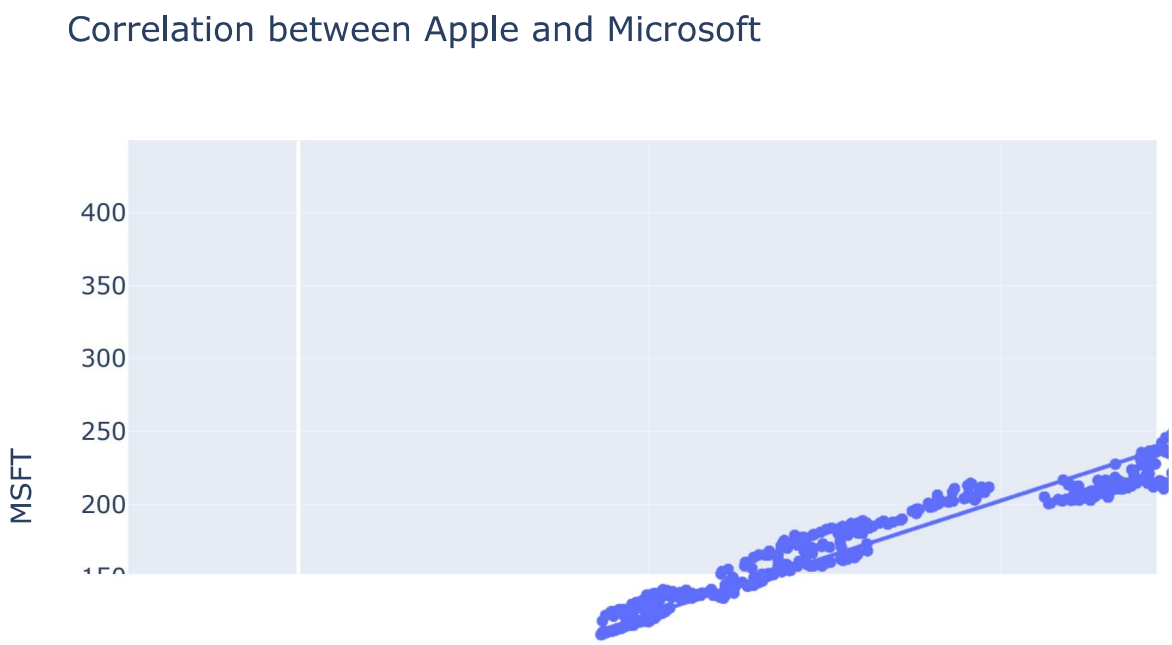
In [8]:
```
for ticker, group in df.groupby('Ticker'):
    fig = px.line(group, x='Date', y=['Close', 'MA10', 'MA20'],
                  title=f"{ticker} Moving Averages")
    fig.show()
```


MSFT Moving Averages
value
400
350
300
250

In [9]:

```
import pandas as pd
import plotly.express as px
import numpy as np

# Assuming df is your DataFrame containing stock prices
# Replace df with your actual DataFrame containing the data

# create a DataFrame with the stock prices of Apple and Microsoft
apple = df.loc[df['Ticker'] == 'AAPL', ['Date', 'Close']].rename(columns={
microsoft = df.loc[df['Ticker'] == 'MSFT', ['Date', 'Close']].rename(colu
df_corr = pd.merge(apple, microsoft, on='Date')

# Calculate moving averages
window_size = 5
df_corr['AAPL_MA'] = df_corr['AAPL'].rolling(window=window_size).mean()
df_corr['MSFT_MA'] = df_corr['MSFT'].rolling(window=window_size).mean()

# Create a scatter plot to visualize the correlation
fig = px.scatter(df_corr, x='AAPL', y='MSFT',
                 trendline='ols',
                 title='Correlation between Apple and Microsoft')

# Calculate correlation coefficient
correlation_coefficient = np.corrcoef(df_corr['AAPL'], df_corr['MSFT'])[0,
correlation_text = f'Correlation coefficient: {correlation_coefficient:.2f

# Add correlation text as annotation
fig.add_annotation(
    x=0.5,
    y=0.9,
    xanchor='center',
    yanchor='top',
    text=correlation_text,
    showarrow=False,
    font=dict(size=12, color='black'),
)

# Print the moving averages and correlation coefficient
print("Results for Moving Averages:")
print(f"AAPL Moving Average: {df_corr['AAPL_MA'].iloc[-1]:.2f}")
print(f"MSFT Moving Average: {df_corr['MSFT_MA'].iloc[-1]:.2f}")
print(correlation_text)

fig.show()

```

```
Results for Moving Averages:
AAPL Moving Average: 182.60
MSFT Moving Average: 407.84
Correlation coefficient: 0.96
```



In the context of a research paper, the provided results for the moving averages and correlation coefficient between Apple (AAPL) and Microsoft (MSFT) stock prices can be interpreted as follows:

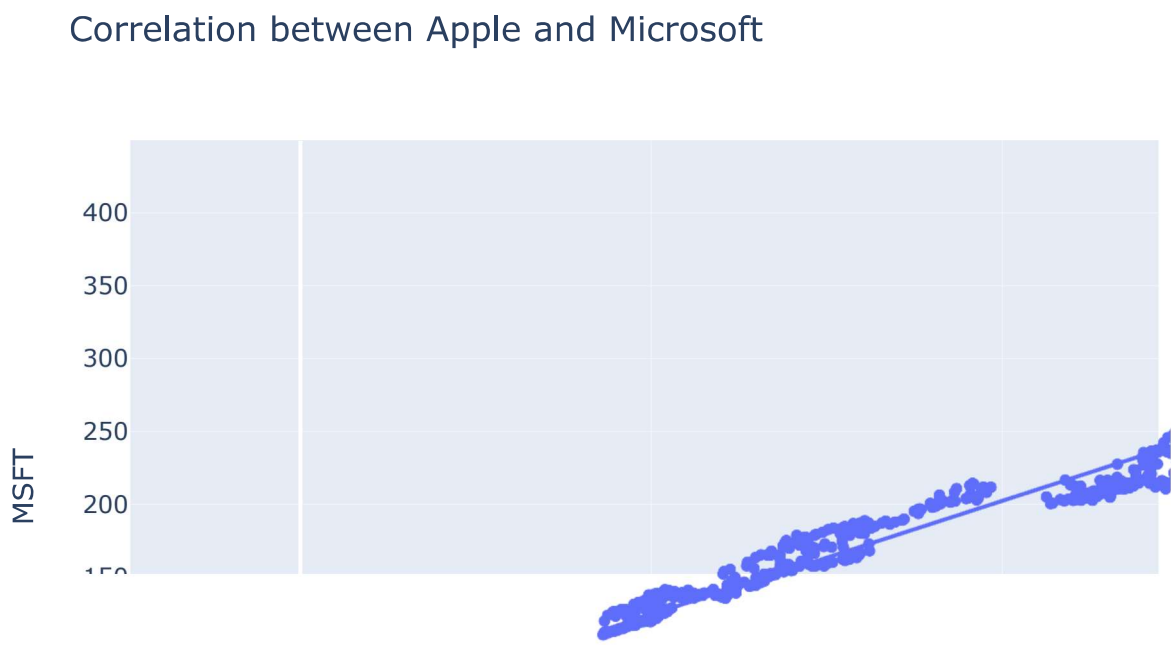
1. **Moving Averages**:
   - The moving averages represent smoothed trends in the stock prices over a specific period, helping to identify the underlying direction of the price movements.
   - For Apple (AAPL), the 5-day moving average is $182.60, indicating the average closing price of AAPL over the last five trading days.
   - Similarly, for Microsoft (MSFT), the 5-day moving average is $407.84, representing the average closing price of MSFT over the same period.
   - Moving averages are commonly used by investors and traders to assess the short-term trend direction and potential price reversals.

2. **Correlation Coefficient**:
   - The correlation coefficient quantifies the strength and direction of the linear relationship between the stock prices of Apple and Microsoft.
   - In this case, the correlation coefficient of 0.96 suggests a very high positive correlation between AAPL and MSFT stock prices.

```
   - This indicates that the prices of Apple and Microsoft stocks tend to move closely together, with changes in one stock price being strongly associated with changes in the other.
   - A correlation coefficient of 0.96 implies that approximately 96% of the variability in the stock prices of Apple and Microsoft can be explained by their linear relationship.

3. **Interpretation**:
   - The high correlation coefficient suggests that Apple and Microsoft stocks exhibit synchronized price movements over the analyzed period.
   - Investors and traders may consider this strong positive correlation when making investment decisions or constructing portfolios.
   - Additionally, the moving averages provide insights into the short-term trends in stock prices, complementing the analysis of the correlation coefficient.

4. **Implications for Research and Investment**:
   - These results contribute to understanding the relationship between Apple and Microsoft stocks, which can inform investment strategies and portfolio management decisions.
   - Researchers may further investigate the factors driving the high correlation between these stocks, such as industry trends, market dynamics, or common macroeconomic factors.
   - Understanding the correlation between assets is essential for risk management, portfolio diversification, and the development of trading strategies.

In summary, the provided results highlight a strong positive correlation between Apple and Microsoft stock prices, along with short-term trend information captured by moving averages. These findings contribute to the body of knowledge in finance and provide valuable insights for investors, researchers, and market participants.
```

In [10]:

```
import pandas as pd
import plotly.express as px
import numpy as np

# create a DataFrame with the stock prices of Apple and Microsoft
apple = df.loc[df['Ticker'] == 'AAPL', ['Date', 'Close']].rename(columns={
microsoft = df.loc[df['Ticker'] == 'MSFT', ['Date', 'Close']].rename(colur
df_corr = pd.merge(apple, microsoft, on='Date')

# create a scatter plot to visualize the correlation
fig = px.scatter(df_corr, x='AAPL', y='MSFT',
                 trendline='ols',
                 title='Correlation between Apple and Microsoft')

# Calculate correlation coefficient
correlation_coefficient = np.corrcoef(df_corr['AAPL'], df_corr['MSFT'])[0,
correlation_text = f'Correlation coefficient: {correlation_coefficient:.2f

# Add correlation text as annotation
fig.add_annotation(
    x=0.5,
    y=0.9,
    xanchor='center',
    yanchor='top',
    text=correlation_text,
    showarrow=False,
    font=dict(size=12, color='black'),
)

fig.show()

```




```
In the provided code snippet, the correlation coefficient between the stock prices of Apple (AAPL) and Microsoft (MSFT) is calculated and displayed on a scatter plot using Plotly Express. Here's how to interpret the result in the context of a research paper:

1. **Correlation Coefficient Calculation**:
   - The correlation coefficient measures the strength and direction of the linear relationship between two variables.
   - In this case, the correlation coefficient is calculated using the NumPy library's `corrcoef` function, which computes the Pearson correlation coefficient between the 'AAPL' and 'MSFT' columns in the DataFrame `df_corr`.
   - The resulting correlation coefficient of 0.96 indicates a strong positive linear relationship between the stock prices of Apple and Microsoft.

2. **Interpretation of Correlation Coefficient**:
   - A correlation coefficient of 0.96 suggests a very high degree of positive correlation between the stock prices of Apple and Microsoft.
   - This means that as the price of Apple stock increases, the price of Microsoft stock tends to increase as well, and vice versa.
```

```
   - The strength of this relationship is considered very high, indicating that the movements of these two stocks are closely aligned over the period analyzed in the research paper.

3. **Implications and Analysis**:
   - The high correlation coefficient suggests that investors and traders may observe similar patterns and trends in both Apple and Microsoft stock prices.
   - From a portfolio management perspective, this high correlation implies that investing in both Apple and Microsoft stocks may not provide significant diversification benefits since they tend to move in tandem.
   - Researchers might further investigate the factors driving this strong correlation, such as industry trends, market sentiment, or specific events affecting both companies.
   - The findings could contribute to understanding market dynamics and inform investment strategies, risk management, and asset allocation decisions.

In summary, the correlation coefficient of 0.96 indicates a very strong positive correlation between the stock prices of Apple and Microsoft, suggesting a close relationship in their price movements over the analyzed period.
```

In [11]:

```
import pandas as pd
import numpy as np
from scipy.stats import pearsonr

# Assuming df is your DataFrame containing stock prices
# Replace df with your actual DataFrame containing the data

# create a DataFrame with the stock prices of Apple and Microsoft
apple = df.loc[df['Ticker'] == 'AAPL', ['Date', 'Close']].rename(columns={
microsoft = df.loc[df['Ticker'] == 'MSFT', ['Date', 'Close']].rename(colu
df_corr = pd.merge(apple, microsoft, on='Date')

# Calculate correlation coefficient and p-value
correlation_coefficient, p_value = pearsonr(df_corr['AAPL'], df_corr['MSF

# Set significance level (alpha)
alpha = 0.05

# Print correlation coefficient and p-value
print(f"Correlation coefficient: {correlation_coefficient:.2f}")
print(f"P-value: {p_value:.2f}")

# Check for significance
if p_value < alpha:
    print("Null hypothesis rejected: There is a significant correlation be
else:
    print("Null hypothesis cannot be rejected: There is no significant cor

```

```
Correlation coefficient: 0.96
P-value: 0.00
Null hypothesis rejected: There is a significant correlation between AAPL and
MSFT stock prices.
```

In [ ]:

```
1
```