In [1]:
```
import yfinance as yf
df = yf.download(['AAPL','MSFT','TSLA'], period="1y")
```

```
C:\Users\Admin\AppData\Local\Temp\ipykernel_10624\4112102506.py:2: FutureWarning: Y
F.download() has changed argument auto_adjust default to True
  df = yf.download(['AAPL','MSFT','TSLA'], period="1y")
[*********************100%***********************]  3 of 3 completed
```

In [27]:
```
import yfinance as yf
import datetime as dt

# Define date range
start = dt.datetime(2020, 1, 1)
end = dt.datetime(2023, 1, 1)

# Fetch AAPL stock data
df = yf.download("AAPL", start=start, end=end)

print(df.head())
```

```
C:\Users\Admin\AppData\Local\Temp\ipykernel_10624\1444677212.py:9: FutureWarning: Y
F.download() has changed argument auto_adjust default to True
  df = yf.download("AAPL", start=start, end=end)
[*********************100%***********************]  1 of 1 completed
Price           Close       High        Low       Open     Volume
Ticker           AAPL       AAPL       AAPL       AAPL       AAPL
Date
2020-01-02  72.538506  72.598884  71.292296  71.545882  135480400
2020-01-03  71.833290  72.594055  71.608685  71.765667  146322800
2020-01-06  72.405678  72.444321  70.703012  70.954188  118387200
2020-01-07  72.065155  72.671348  71.845377  72.415345  108872000
2020-01-08  73.224419  73.526310  71.768094  71.768094  132079200
```

In [26]:
```
import os

def fetch_tickers_from_file(path="tickers.txt"):
    if not os.path.exists(path):
        print(f"⚠️ File {path} not found. Using default tickers.")
        return ["AAPL", "MSFT", "GOOGL", "TSLA", "AMZN"]
    with open(path) as f:
        return [line.strip() for line in f if line.strip()]
```

In [30]:
```
import os

# Make sure "data" folder exists
os.makedirs("data", exist_ok=True)
```

In [29]:
```
df.to_csv("data/all_stocks_1yr.csv")
```

In [35]:
```
df.to_csv("all_stocks_1yr.csv")
```

In [34]:
```
import sqlite3
```

```
conn = sqlite3.connect('stock_data.db')
df.to_sql('stock_prices', conn, if_exists='replace')
conn.close()
```

In [42]:
```
C:\Users\Admin\Documents\stock-data\all_stocks_1yr.csv
```

```
  Cell In[42], line 1
    C:\Users\Admin\Documents\stock-data\all_stocks_1yr.csv
       ^
SyntaxError: unexpected character after line continuation character
```

In [43]:
```
import pandas as pd

# Use raw string to avoid issues with backslashes
df = pd.read_csv(r"C:\Users\Admin\Documents\stock-data\all_stocks_1yr.csv")

print(df.head())
```

```
      Ticker                TSLA              TSLA.1              TSLA.2  \
0      Price                Open                High                 Low
1       Date                 NaN                 NaN                 NaN
2 2024-08-16  211.14999389648438   219.8000030517578   210.8000030517578
3 2024-08-19  217.07000732421875  222.97999572753906  214.08999633789062
4 2024-08-20   224.8800048828125  228.22000122070312  219.55999755859375

              TSLA.3    TSLA.4               AAPL              AAPL.1  \
0              Close    Volume               Open                High
1                NaN       NaN                NaN                 NaN
2  216.1199951171875  88765100  222.88270038911833  225.77922361016536
3  222.72000122070312  76435200  224.67435623990076  224.94310972106823
4  221.10000610351562  74001200  224.72413110884563  226.11763957317405

             AAPL.2              AAPL.3  ...               MSFT  \
0               Low               Close  ...              Close
1               NaN                 NaN  ...                NaN
2  222.6139468974481  225.00283813476562  ...   416.0345458984375
3  222.00676340736013  224.84356689453125  ...   419.0767517089844
4   224.4056062095116   225.460693359375  ...  422.32769775390625

             MSFT.1              MSFT.2              MSFT.3    MSFT.4  \
0              High                 Low                Open    Volume
1               NaN                 NaN                 NaN       NaN
2  418.8878379189571  414.87134183677813  418.15215435686315  22775600
3  419.2954725526047  414.03625113553215  416.52170147668573  15234000
4  423.3815262412174  419.18611531171354  419.24576369099765  16387600

               AMZN             AMZN.1              AMZN.2  \
0              Open               High                 Low
1               NaN                 NaN                 NaN
2  177.0399932861328  178.33999633789062  176.25999450683594
3  177.63999938964844   178.3000030517578  176.16000366210938
4   177.9199981689453  179.00999450683594  177.42999267578125

             AMZN.3    AMZN.4
0             Close    Volume
1               NaN       NaN
2  177.05999755859375  31489200
3  178.22000122070312  31129800
4   178.8800048828125  26255200

[5 rows x 26 columns]
```

In [44]:
```
r"C:\Users\Admin\Documents\stock-data\all_stocks_1yr.csv"
```

Out[44]:
```
'C:\\Users\\Admin\\Documents\\stock-data\\all_stocks_1yr.csv'
```

In [45]:
```
import pandas as pd

# Read CSV but use row 0 as a header
df = pd.read_csv(r"C:\Users\Admin\Documents\stock-data\all_stocks_1yr.csv", header=

print(df.head())
```

```
        Price        Open        High         Low       Close      Volume  \
0        Date         NaN         NaN         NaN         NaN         NaN
1  2024-08-16  211.149994  219.800003  210.800003  216.119995  88765100.0
2  2024-08-19  217.070007  222.979996  214.089996  222.720001  76435200.0
3  2024-08-20  224.880005  228.220001  219.559998  221.100006  74001200.0
4  2024-08-21  222.669998  224.660004  218.860001  223.270004  70146000.0

       Open.1      High.1       Low.1     Close.1  ...     Close.3  \
0         NaN         NaN         NaN         NaN  ...         NaN
1  222.882700  225.779224  222.613947  225.002838  ...  416.034546
2  224.674356  224.943110  222.006763  224.843567  ...  419.076752
3  224.724131  226.117640  224.405606  225.460693  ...  422.327698
4  225.470666  226.923894  224.007474  225.351212  ...  421.671539

       High.3       Low.3      Open.3    Volume.3      Open.4      High.4  \
0         NaN         NaN         NaN         NaN         NaN         NaN
1  418.887838  414.871342  418.152154  22775600.0  177.039993  178.339996
2  419.295473  414.036251  416.521701  15234000.0  177.639999  178.300003
3  423.381526  419.186115  419.245764  16387600.0  177.919998  179.009995
4  423.918366  419.265610  421.611861  16067300.0  179.919998  182.389999

        Low.4     Close.4    Volume.4
0         NaN         NaN         NaN
1  176.259995  177.059998  31489200.0
2  176.160004  178.220001  31129800.0
3  177.429993  178.880005  26255200.0
4  178.889999  180.110001  35599100.0

[5 rows x 26 columns]
```

In [46]:

```python
import pandas as pd

df = pd.read_csv(
    r"C:\Users\Admin\Documents\stock-data\all_stocks_1yr.csv",
    header=1   # skip the extra header row
)

print(df.head())
```

```
        Price        Open        High         Low       Close      Volume  \
0        Date         NaN         NaN         NaN         NaN         NaN
1  2024-08-16  211.149994  219.800003  210.800003  216.119995  88765100.0
2  2024-08-19  217.070007  222.979996  214.089996  222.720001  76435200.0
3  2024-08-20  224.880005  228.220001  219.559998  221.100006  74001200.0
4  2024-08-21  222.669998  224.660004  218.860001  223.270004  70146000.0

       Open.1      High.1       Low.1     Close.1  ...     Close.3  \
0         NaN         NaN         NaN         NaN  ...         NaN
1  222.882700  225.779224  222.613947  225.002838  ...  416.034546
2  224.674356  224.943110  222.006763  224.843567  ...  419.076752
3  224.724131  226.117640  224.405606  225.460693  ...  422.327698
4  225.470666  226.923894  224.007474  225.351212  ...  421.671539

       High.3       Low.3      Open.3    Volume.3      Open.4      High.4  \
0         NaN         NaN         NaN         NaN         NaN         NaN
1  418.887838  414.871342  418.152154  22775600.0  177.039993  178.339996
2  419.295473  414.036251  416.521701  15234000.0  177.639999  178.300003
3  423.381526  419.186115  419.245764  16387600.0  177.919998  179.009995
4  423.918366  419.265610  421.611861  16067300.0  179.919998  182.389999

        Low.4     Close.4    Volume.4
0         NaN         NaN         NaN
1  176.259995  177.059998  31489200.0
2  176.160004  178.220001  31129800.0
3  177.429993  178.880005  26255200.0
4  178.889999  180.110001  35599100.0

[5 rows x 26 columns]
```

In [50]:

```python
tickers = ["TSLA", "AAPL", "MSFT", "AMZN"]
tidy_list = []

for i, ticker in enumerate(tickers):
    cols = ["Open", "High", "Low", "Close", "Volume"]

    # if i > 0, use the ".i" suffix (Open.1, Open.2, etc.)
    if i == 0:
        sub = df[["Price"] + cols].copy()
    else:
        sub = df[["Price"] + [f"{c}.{i}" for c in cols]].copy()
        sub.columns = ["Price"] + cols

    sub["Ticker"] = ticker
    tidy_list.append(sub)

# Combine all tickers
tidy = pd.concat(tidy_list)

# Rename Date column
tidy.rename(columns={"Price": "Date"}, inplace=True)

print(tidy.head(10))
```

```
         Date        Open        High         Low       Close      Volume  \
0        Date         NaN         NaN         NaN         NaN         NaN
1  2024-08-16  211.149994  219.800003  210.800003  216.119995  88765100.0
2  2024-08-19  217.070007  222.979996  214.089996  222.720001  76435200.0
3  2024-08-20  224.880005  228.220001  219.559998  221.100006  74001200.0
4  2024-08-21  222.669998  224.660004  218.860001  223.270004  70146000.0
5  2024-08-22  223.820007  224.800003  210.320007  210.660004  79514500.0
6  2024-08-23  214.460007  221.479996  214.210007  220.320007  81525200.0
7  2024-08-26  218.750000  219.089996  211.009995  213.210007  59301200.0
8  2024-08-27  213.250000  215.660004  206.940002  209.210007  62821400.0
9  2024-08-28  209.720001  211.839996  202.589996  205.750000  64116400.0

  Ticker
0   TSLA
1   TSLA
2   TSLA
3   TSLA
4   TSLA
5   TSLA
6   TSLA
7   TSLA
8   TSLA
9   TSLA
```

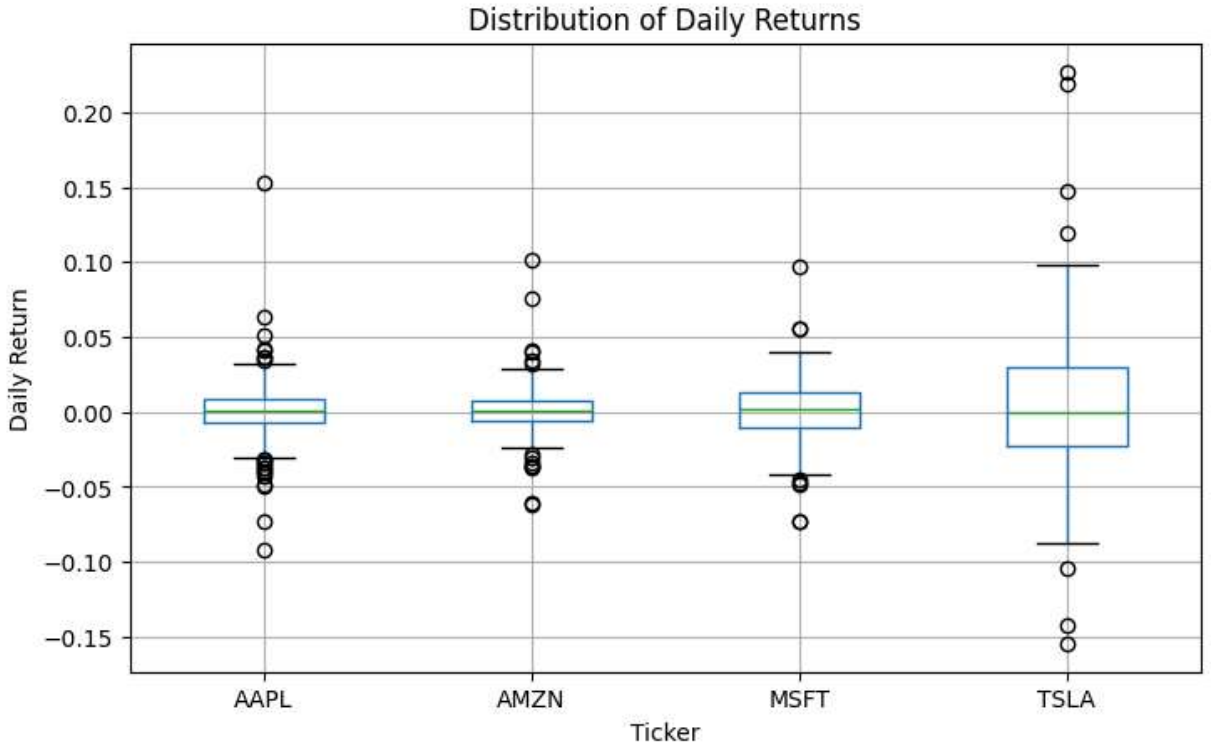
In [49]:
```
tidy.to_csv(r"C:\Users\Admin\Documents\stock-data\tidy_stocks.csv", index=False)
```

In [53]:
```
# ✅ 1. Summary statistics
print("Summary stats per ticker:")
print(tidy.groupby("Ticker")[["Open", "Close", "Volume"]].describe())

# ✅ 2. Closing price trends
plt.figure(figsize=(12,6))
for ticker in tidy["Ticker"].unique():
    subset = tidy[tidy["Ticker"] == ticker]
    plt.plot(subset["Date"], subset["Close"], label=ticker)

plt.title("Stock Closing Prices (1 Year)")
plt.xlabel("Date")
plt.ylabel("Closing Price (USD)")
plt.legend()
plt.grid(True)
plt.show()

# ✅ 3. Daily returns
tidy["Return"] = tidy.groupby("Ticker")["Close"].pct_change()

# Plot returns distribution
tidy.boxplot(column="Return", by="Ticker", figsize=(8,5))
plt.title("Distribution of Daily Returns")
plt.suptitle("")
plt.ylabel("Daily Return")
plt.show()

# ✅ 4. Correlation between tickers
# Pivot data so each ticker is a column
pivot_close = tidy.pivot(index="Date", columns="Ticker", values="Close")
```

```
corr = pivot_close.pct_change().corr()
print("\nCorrelation matrix of returns:")
print(corr)

plt.figure(figsize=(6,4))
plt.imshow(corr, cmap="coolwarm", interpolation="none")
plt.colorbar(label="Correlation")
plt.xticks(range(len(corr)), corr.columns, rotation=45)
plt.yticks(range(len(corr)), corr.index)
plt.title("Correlation of Stock Returns")
plt.show()
```

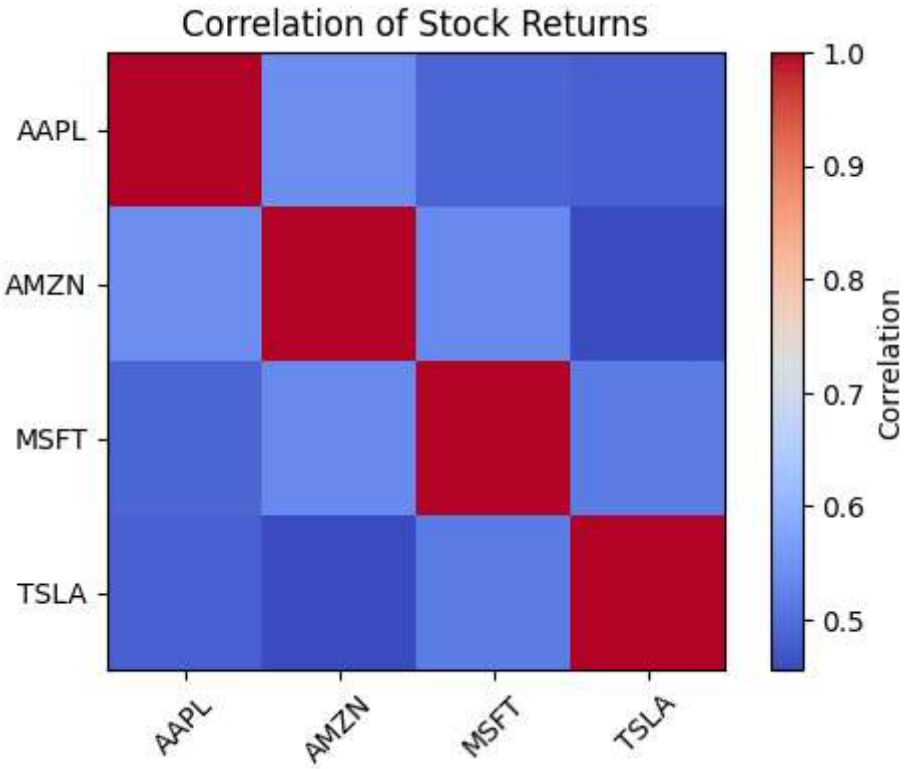
```
Summary stats per ticker:
         Open                                                          \
        count        mean        std         min         25%         50%
Ticker
AAPL    250.0  221.268002  15.937920  171.530131  209.734468  223.262230
AMZN    250.0  433.158199  41.399450  350.237034  409.850796  422.929150
MSFT    250.0  173.480706  14.200689  141.378846  163.107090  170.130947
TSLA    250.0  307.512280  64.659165  208.630005  250.057503  309.539993

                            Close              ...              \
               75%         max  count        mean  ...         75%
Ticker                                         ...
AAPL    232.184425  257.276679  250.0  221.555997  ...  232.252090
AMZN    450.572495  555.229980  250.0  433.296683  ...  452.682091
MSFT    184.690941  204.130005  250.0  173.477628  ...  184.870537
TSLA    345.624992  475.899994  250.0  307.359840  ...  344.937500

                  Volume                                                \
                 max  count        mean           std         min         25%
Ticker
AAPL    258.103729  250.0  53867470.4  2.773949e+07  23234700.0  39510475.0
AMZN    535.640015  250.0  21385680.0  8.382255e+06   7164500.0  16397250.0
MSFT    205.893341  250.0  32680754.8  1.534023e+07  10403300.0  22155025.0
TSLA    479.859985  250.0  98859748.4  3.739205e+07  37167600.0  74083300.0


                50%          75%          max
Ticker
AAPL    46742450.0   56652700.0  318679900.0
AMZN    19314400.0   23452450.0   64263700.0
MSFT    29599450.0   38488325.0  127490100.0
TSLA    89033600.0  115660600.0  287499800.0

[4 rows x 24 columns]
```


```
C:\Users\Admin\AppData\Local\Temp\ipykernel_10624\1272865072.py:19: FutureWarning: T
he default fill_method='ffill' in SeriesGroupBy.pct_change is deprecated and will be
removed in a future version. Either fill in any non-leading NA values prior to calli
ng pct_change or specify 'fill_method=None' to not fill NA values.
  tidy["Return"] = tidy.groupby("Ticker")["Close"].pct_change()
```




```
C:\Users\Admin\AppData\Local\Temp\ipykernel_10624\1272865072.py:32: FutureWarning: T
he default fill_method='pad' in DataFrame.pct_change is deprecated and will be remov
ed in a future version. Either fill in any non-leading NA values prior to calling pc
t_change or specify 'fill_method=None' to not fill NA values.
  corr = pivot_close.pct_change().corr()
```

```
Correlation matrix of returns:
Ticker      AAPL      AMZN      MSFT      TSLA
Ticker
AAPL    1.000000  0.541019  0.489579  0.482008
AMZN    0.541019  1.000000  0.536297  0.455668
MSFT    0.489579  0.536297  1.000000  0.518426
TSLA    0.482008  0.455668  0.518426  1.000000
```



In [ ]: