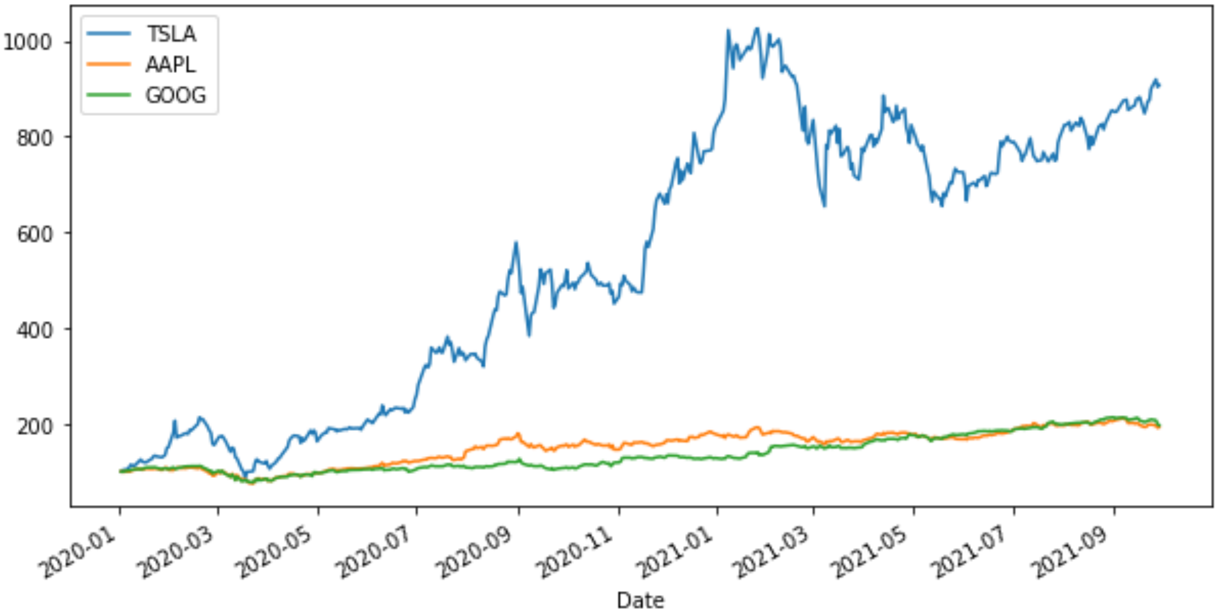
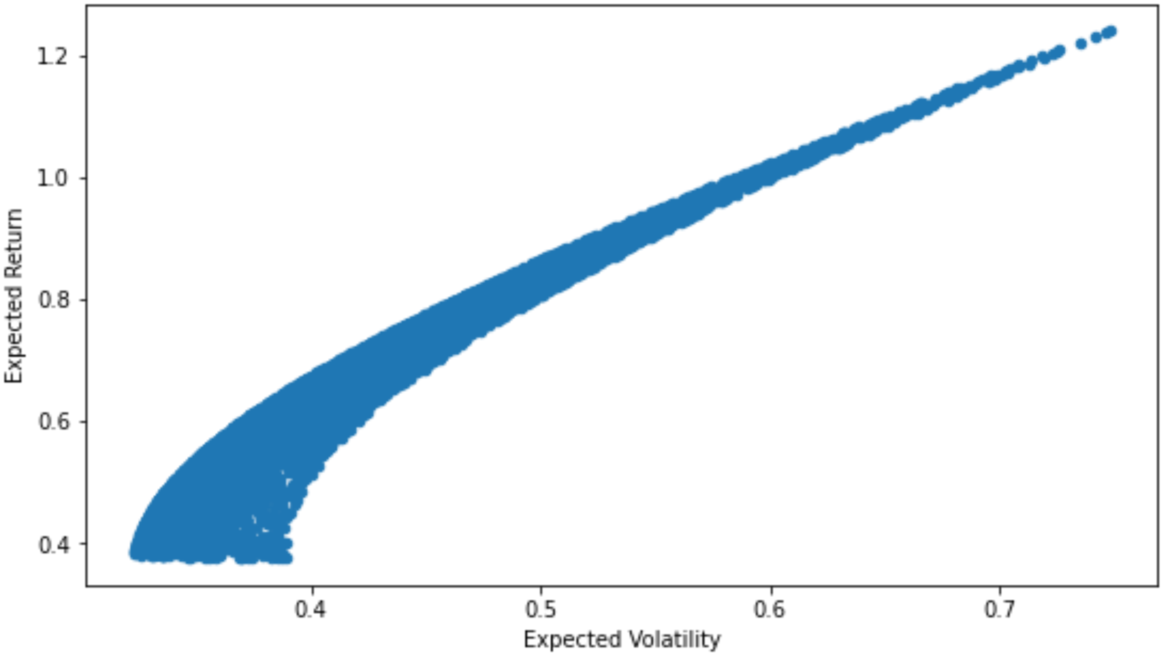
In [35]:

```
import numpy as np
import pandas as pd
import yfinance as yf
from datetime import datetime
import matplotlib.pyplot as plt


# 1get data
ticker = ['TSLA', 'AAPL', 'GOOG']
df = pd.DataFrame()
for t in ticker:
    df[t] = yf.download(t, start="2020-01-01", end="2021-9-30")['Adj C
lose']

df.to_csv('df1.csv')
# df = pd.read_csv('/Users/vl/Desktop/PycharmProjects/Fundamental/df1.
csv')

(df / df.iloc[0] * 100).plot(figsize=(10, 5))
logreturns = np.log(df / df.shift(1))
logmean = logreturns.mean()
logmeanyr = logreturns.mean() * 250
logreturns.cov() * 250
logreturns.corr()

num_ticker = len(ticker)


arr = np.random.random(10)
weights = np.random.random(num_ticker)
weights /= np.sum(weights)
sum_weights = sum(weights)


# from pypfopt.efficient_frontier import EfficientFrontier as ef
# from pypfopt import risk_models
# from pypfopt import expected_returns

# portfolio mean returns
mu = np.sum(weights * logreturns.mean()) * 250

# portfolio volatility
std = np.dot(weights.T, np.dot(logreturns.cov() * 250, weights)) ** 0.
5

# monte carlo
pfolio_returns = []
pfolio_volatilities = []
sharpe_ratio = []
for x in range(10000):
    weights = np.random.random(num_ticker)
    weights /= np.sum(weights)
    pfolio_returns.append(np.sum(weights * logreturns.mean()) * 250)
    pfolio_volatilities.append(np.sqrt(np.dot(weights.T, np.dot(logret
urns.cov() * 250, weights))))

pfolio_returns = np.array(pfolio_returns)
pfolio_volatilities = np.array(pfolio_volatilities)
portfolio = pd.DataFrame({'Return': pfolio_returns, 'Volatility': pfol
io_volatilities})

sharpe_ratio_stocks = pfolio_returns / pfolio_volatilities


portfolio.plot(x='Volatility', y='Return', kind='scatter', figsize=(9,
5))
plt.xlabel('Expected Volatility')
plt.ylabel('Expected Return')

from pypfopt import EfficientFrontier
from pypfopt import risk_models
from pypfopt import expected_returns

mu = expected_returns.mean_historical_return(df)
S = risk_models.sample_cov(df)

# Optimize for maximal Sharpe ratio
ef = EfficientFrontier(mu, S)
raw_weights = ef.max_sharpe()
cleaned_weights = ef.clean_weights()

ef.save_weights_to_file("optwit2.csv")  # saves to file
print('opt weights = ', cleaned_weights)

ef.portfolio_performance(verbose=True)
```

```
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
opt weights =  OrderedDict([('TSLA', 1.0), ('AAPL', 0.0), ('GOOG', 0
.0)])
Expected annual return: 254.8%
Annual volatility: 76.3%
Sharpe Ratio: 3.31
```

```
Out[35]: (2.5477938182746414, 0.7631256330653508, 3.3124215840069575)
```


TSLA
AAPL
GOOG
1000
800
600
400
200
2020-01
2020-03
2020-05
2020-07
2020-09
2020-11
2021-01
2021-03
2021-05
2021-07
2021-09
Date


1.2
1.0
0.8
0.6
0.4
Expected Return
0.4
0.5
0.6
0.7
Expected Volatility