Experiment 7

Name: Umang Kirit Lodaya **SAP ID: 60009200032**

Batch: D11

Aim: Perform CPPI and Drawdown Constraints on specified data source.

Objective:

- Learn about CPPI and drawdown constraints.
- Implement CPPI and drawdown constraints in Python.
- Test the CPPI and drawdown constraints on a specified data source.

Theory:

CPPI stands for constant proportion portfolio insurance. It is a risk management strategy that aims to protect the investor's capital while still allowing them to participate in market growth. CPPI works by allocating a fixed proportion of the investor's wealth to the risky assets and the remaining proportion to cash. The risky assets are rebalanced periodically, and the cash is used to buy more risky assets when the market declines.

The CPPI strategy is implemented as follows:

1. The investor specifies a target floor, which is the minimum value that the portfolio is allowed to fall to.
2. The investor also specifies a multiplier, which determines how much of the portfolio is allocated to risky assets.
3. The portfolio is rebalanced periodically, and the risky assets are rebalanced to a target value that is equal to the floor * multiplier.
4. If the market declines, the cash in the portfolio is used to buy more risky assets, so that the portfolio stays above the floor.
5. If the market rises, the risky assets are sold, and the proceeds are either reinvested in the risky assets or withdrawn from the portfolio.

The formula for the CPPI strategy is as follows:

$$CPPI = floor * multiplier * (1 + returns)^t$$

where:

- CPPI is the value of the CPPI portfolio at time t
- floor is the target floor
- multiplier is the CPPI multiplier
- returns is the return of the risky assets over the period t

Drawdown constraints are a way to limit the amount of loss that an investor can suffer. They work by preventing the investor's portfolio from falling below a certain level. Drawdown constraints can be implemented in a variety of ways, but they typically involve selling some of the risky assets in the portfolio when the market declines.

Drawdown constraints are implemented as follows:

1. The investor specifies a maximum drawdown, which is the maximum percentage loss that the portfolio is allowed to suffer.
2. If the portfolio falls below the maximum drawdown, the investor sells some of the risky assets in the portfolio to bring the portfolio back above the maximum drawdown level.

The formula for the drawdown constraint is as follows:

$$drawdown = max(0, portfolio\_value - floor) / floor$$

where:

- drawdown is the current drawdown
- portfolio_value is the current value of the portfolio
- floor is the target floor

Lab Experiment to be done by students:

1. Download the specified data source.
2. Implement the CPPI and drawdown constraints in Python.
3. Test the CPPI and drawdown constraints on the specified data source.
4. Analyze the results.

NAME: UMANG KIRIT LODAYA

SAP ID: 60009200032

BATCH: D11

In [12]:
```
import yfinance as yf
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [13]:
```
# Step 1: Import the specified data source from yfinance
tickers = ["AAPL", "BND"]
data = yf.download(tickers, period="5y", start="2018-01-01")
```

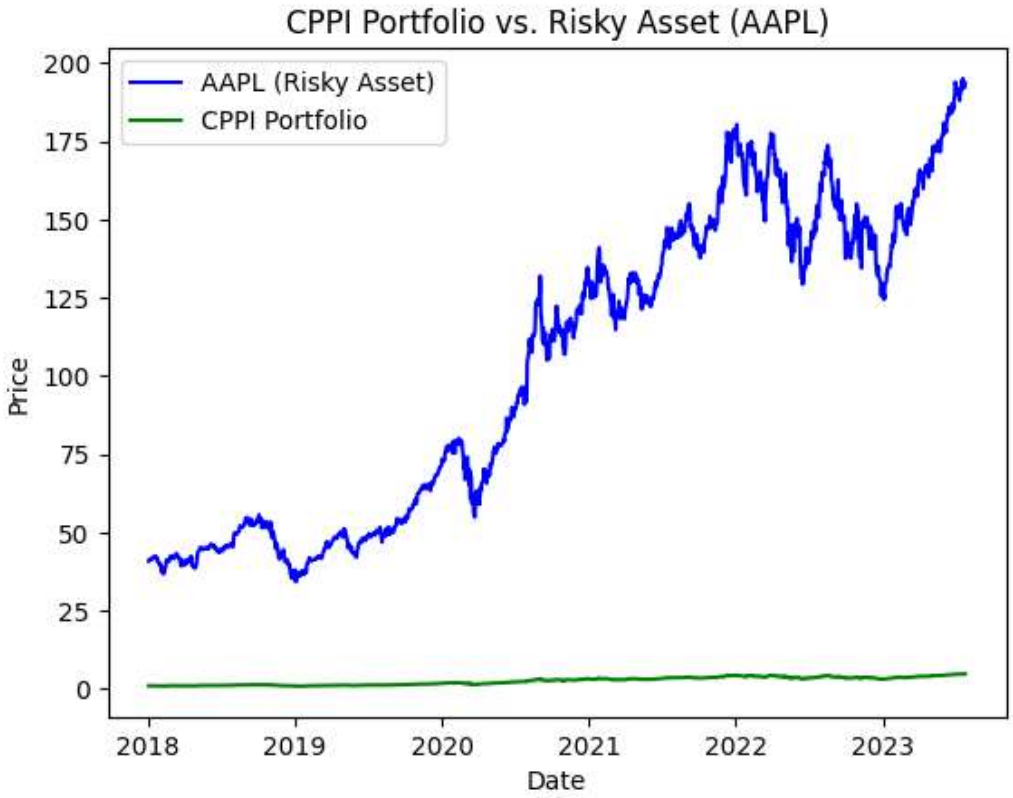
```
[**********************100%**********************]  2 of 2 c
pleted
```

In [14]:
```
# Step 2: Implement CPPI and drawdown constraints
def run_cppi_strategy(data, floor=0.8, m=3):
    df = data.copy()
    risky_asset = df["Adj Close"]["AAPL"]
    risk_free_asset = df["Adj Close"]["BND"]
    floor_value = df["Adj Close"]["AAPL"] * floor

    # CPPI Parameters
    account_value = np.ones(len(df))  # Initial investment val
ue is set to 1
    cushion = account_value - floor_value
    cppi_value = account_value.copy()
    m_multiplier = m

    for i in range(1, len(df)):
        cushion[i] = max(account_value[i] - floor_value[i], 0)
        account_value[i] = cppi_value[i - 1] * (1 + risky_asse
t[i] / risky_asset[i - 1] - 1)  # Rebalance daily
        cppi_value[i] = max(account_value[i], cushion[i] * m_m
ultiplier)

    return cppi_value

def apply_drawdown_constraint(data, max_drawdown=0.2):
    df = data.copy()
    risky_asset = df["Adj Close"]["AAPL"]

    peak_value = 0
    for i in range(len(df)):
        if risky_asset[i] > peak_value:
            peak_value = risky_asset[i]
        elif (peak_value - risky_asset[i]) / peak_value > max_
drawdown:
            df["Adj Close"]["AAPL"][i:] = 0

    return df
```

In [15]:
```
# Step 3: Test the CPPI and drawdown constraints on the spec
ied data source
cppi_portfolio = run_cppi_strategy(data)
data_with_drawdown_constraint = apply_drawdown_constraint(dat
a)
```

```
<ipython-input-14-8d24f49ef5fe>:30: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFr
ame

See the caveats in the documentation: https://pandas.pydata.or
g/pandas-docs/stable/user_guide/indexing.html#returning-a-view
-versus-a-copy
  df["Adj Close"]["AAPL"][i:] = 0
```

In [18]:
```
# Step 4: Analyze the results
# Plot the performance of CPPI portfolio and the risky asset
plt.plot(data.index, data["Adj Close"]["AAPL"], label="AAPL (Risky Asset)", color="blue")
plt.plot(data.index, cppi_portfolio, label="CPPI Portfolio", color="green")
plt.title("CPPI Portfolio vs. Risky Asset (AAPL)")
plt.xlabel("Date")
plt.ylabel("Price")
plt.legend()
plt.show()
```



In [19]:
```
# Plot the performance with drawdown constraint
plt.plot(data_with_drawdown_constraint.index, data_with_drawdo
wn_constraint["Adj Close"]["AAPL"], label="AAPL (With Drawdown
Constraint)", color="red")
plt.title("AAPL Price with Drawdown Constraint")
plt.xlabel("Date")
plt.ylabel("Price")
plt.legend()
plt.show()
```