Creating a Mutual Fund Plan with Python

Mutual funds are investment plans that pool money from multiple investors to purchase a diversified portfolio of stocks, bonds, and other securities, managed by professional fund managers. A mutual fund plan is created by selecting the stocks where an investor can benefit in the long term.

Creating a Mutual Fund Plan with Python
A mutual fund plan is created by selecting the stocks where an investor can benefit in the long term. Here’s the process we can follow to create a mutual fund plan:

Step 1: Gather historical stock data, such as closing prices and growth trends over time.
Step 2: Calculate key metrics like Return on Investment (ROI) and volatility (risk) to understand how each stock has performed historically.
Step 3: Choose stocks that have a high ROI and low volatility to ensure a balance between risk and reward.
Step 4: Calculate the future value of monthly investments based on the expected ROI of the selected stocks.

In [ ]:

In [24]:

```
import pandas as pd

data = pd.read_csv(r"C:\Users\hp\Downloads\nifty50_closing_prices.csv")

data
```

Out[24]:

	Date	RELIANCE.NS	HDFCBANK.NS	ICICIBANK.NS	INFY.NS	TCS.NS	KOTAKBANK.NS	HINDUNILVR.NS	
0	2024-08-20 00:00:00+05:30	2991.899902	1637.699951	1179.449951	1872.199951	4523.299805	1805.650024	2751.050049	498
1	2024-08-21 00:00:00+05:30	2997.350098	1625.800049	1174.849976	1872.699951	4551.500000	1812.949951	2791.199951	505
2	2024-08-22 00:00:00+05:30	2996.250000	1631.300049	1191.099976	1880.250000	4502.000000	1821.500000	2792.800049	504
3	2024-08-23 00:00:00+05:30	2999.949951	1625.050049	1203.500000	1862.099976	4463.899902	1818.000000	2815.600098	505
4	2024-08-26 00:00:00+05:30	3025.199951	1639.949951	1213.300049	1876.150024	4502.450195	1812.500000	2821.149902	505
5	2024-08-27 00:00:00+05:30	3000.899902	1637.750000	1226.349976	1900.099976	4497.149902	1803.349976	2766.899902	500
6	2024-08-28 00:00:00+05:30	2996.600098	1637.099976	1223.849976	1939.099976	4506.049805	1791.300049	2764.350098	497
7	2024-08-29 00:00:00+05:30	3041.850098	1638.550049	1221.900024	1933.349976	4511.799805	1777.250000	2785.250000	505
8	2024-08-30 00:00:00+05:30	3019.250000	1636.900024	1229.199951	1943.699951	4553.750000	1780.800049	2778.000000	501
9	2024-09-02 00:00:00+05:30	3032.500000	1626.949951	1229.949951	1964.500000	4521.049805	1780.250000	2789.050049	510
10	2024-09-03 00:00:00+05:30	3018.250000	1637.349976	1247.699951	1941.250000	4512.350098	1783.800049	2794.300049	509
11	2024-09-04 00:00:00+05:30	3029.100098	1641.800049	1236.349976	1922.449951	4479.250000	1783.800049	2841.250000	506
12	2024-09-05 00:00:00+05:30	2985.949951	1645.449951	1235.949951	1933.150024	4475.950195	1777.949951	2838.449951	511
13	2024-09-06 00:00:00+05:30	2929.649902	1636.949951	1208.150024	1901.849976	4456.750000	1764.150024	2838.949951	501
14	2024-09-09 00:00:00+05:30	2924.899902	1646.500000	1235.000000	1894.650024	4449.549805	1790.150024	2921.800049	511
15	2024-09-10 00:00:00+05:30	2923.050049	1650.349976	1237.300049	1912.300049	4507.850098	1791.599976	2898.600098	513
16	2024-09-11 00:00:00+05:30	2903.000000	1643.900024	1236.349976	1910.150024	4479.350098	1789.250000	2904.149902	514

	Date	RELIANCE.NS	HDFCBANK.NS	ICICIBANK.NS	INFY.NS	TCS.NS	KOTAKBANK.NS	HINDUNILVR.NS	
17	2024-09-12 00:00:00+05:30	2959.600098	1666.599976	1252.150024	1950.449951	4517.700195	1827.449951	2956.399902	519.
18	2024-09-13 00:00:00+05:30	2945.250000	1665.949951	1250.349976	1944.099976	4522.600098	1820.349976	2932.949951	513.
19	2024-09-16 00:00:00+05:30	2942.699951	1670.949951	1262.849976	1950.250000	4513.250000	1831.300049	2867.100098	511.
20	2024-09-17 00:00:00+05:30	2944.600098	1668.800049	1268.099976	1952.550049	4505.649902	1846.650024	2873.500000	507.
21	2024-09-18 00:00:00+05:30	2926.899902	1694.800049	1288.349976	1892.150024	4346.149902	1839.699951	2875.850098	507.
22	2024-09-19 00:00:00+05:30	2939.350098	1708.500000	1292.000000	1894.199951	4296.149902	1871.949951	2911.750000	508.
23	2024-09-20 00:00:00+05:30	2971.850098	1741.199951	1338.449951	1905.750000	4284.899902	1904.500000	2977.600098	514.

24 rows × 51 columns

In [2]:
```
data.columns
```

Out[2]:
```
Index(['Date', 'RELIANCE.NS', 'HDFCBANK.NS', 'ICICIBANK.NS', 'INFY.NS',
       'TCS.NS', 'KOTAKBANK.NS', 'HINDUNILVR.NS', 'ITC.NS', 'LT.NS', 'SBIN.NS',
       'BAJFINANCE.NS', 'BHARTIARTL.NS', 'HCLTECH.NS', 'ASIANPAINT.NS',
       'AXISBANK.NS', 'DMART.NS', 'MARUTI.NS', 'ULTRACEMCO.NS', 'HDFC.NS',
       'TITAN.NS', 'SUNPHARMA.NS', 'M&M.NS', 'NESTLEIND.NS', 'WIPRO.NS',
       'ADANIGREEN.NS', 'TATASTEEL.NS', 'JSWSTEEL.NS', 'POWERGRID.NS',
       'ONGC.NS', 'NTPC.NS', 'COALINDIA.NS', 'BPCL.NS', 'IOC.NS', 'TECHM.NS',
       'INDUSINDBK.NS', 'DIVISLAB.NS', 'GRASIM.NS', 'CIPLA.NS',
       'BAJAJFINSV.NS', 'TATAMOTORS.NS', 'HEROMOTOCO.NS', 'DRREDDY.NS',
       'SHREECEM.NS', 'BRITANNIA.NS', 'UPL.NS', 'EICHERMOT.NS', 'SBILIFE.NS',
       'ADANIPORTS.NS', 'BAJAJ-AUTO.NS', 'HINDALCO.NS'],
      dtype='object')
```

In [3]:
```
data.shape
```

Out[3]:
```
(24, 51)
```

In [5]:
```
data.describe()
```

Out[5]:

	RELIANCE.NS	HDFCBANK.NS	ICICIBANK.NS	INFY.NS	TCS.NS	KOTAKBANK.NS	HINDUNILVR.NS	ITC.NS	
count	24.000000	24.000000	24.000000	24.000000	24.000000	24.000000	24.000000	24.000000	24
mean	2976.912506	1652.339579	1236.770818	1914.558324	4478.349976	1809.422918	2845.333344	507.739581	3647
std	41.290551	28.258220	36.438726	30.240685	70.822718	32.936318	65.620694	5.472559	6(
min	2903.000000	1625.050049	1174.849976	1862.099976	4284.899902	1764.150024	2751.050049	497.299988	3536
25%	2941.862488	1637.062469	1219.750031	1893.687469	4472.937622	1783.800049	2790.662476	504.962502	3597
50%	2988.924927	1640.875000	1235.474976	1911.225037	4504.050049	1804.500000	2838.699951	507.550003	3646
75%	3005.237427	1666.112457	1250.799988	1941.862488	4514.362549	1822.987488	2899.987549	511.337509	3689
max	3041.850098	1741.199951	1338.449951	1964.500000	4553.750000	1904.500000	2977.600098	519.500000	3793

8 rows × 50 columns

In [6]:
```
data.dtypes
```

Out[6]:
```
Date              object
RELIANCE.NS      float64
HDFCBANK.NS      float64
ICICIBANK.NS     float64
INFY.NS          float64
TCS.NS           float64
KOTAKBANK.NS     float64
HINDUNILVR.NS    float64
ITC.NS           float64
LT.NS            float64
SBIN.NS          float64
BAJFINANCE.NS    float64
BHARTIARTL.NS    float64
HCLTECH.NS       float64
ASIANPAINT.NS    float64
AXISBANK.NS      float64
DMART.NS         float64
MARUTI.NS        float64
ULTRACEMCO.NS    float64
```

In [7]:
```
#Before moving forward, I'll convert the date column into a datetime data type:
data['Date'] = pd.to_datetime(data['Date'])
```

In [8]:
```
#Now, let's have a look at whether this data has any null values or not:
data.isnull().sum()
```

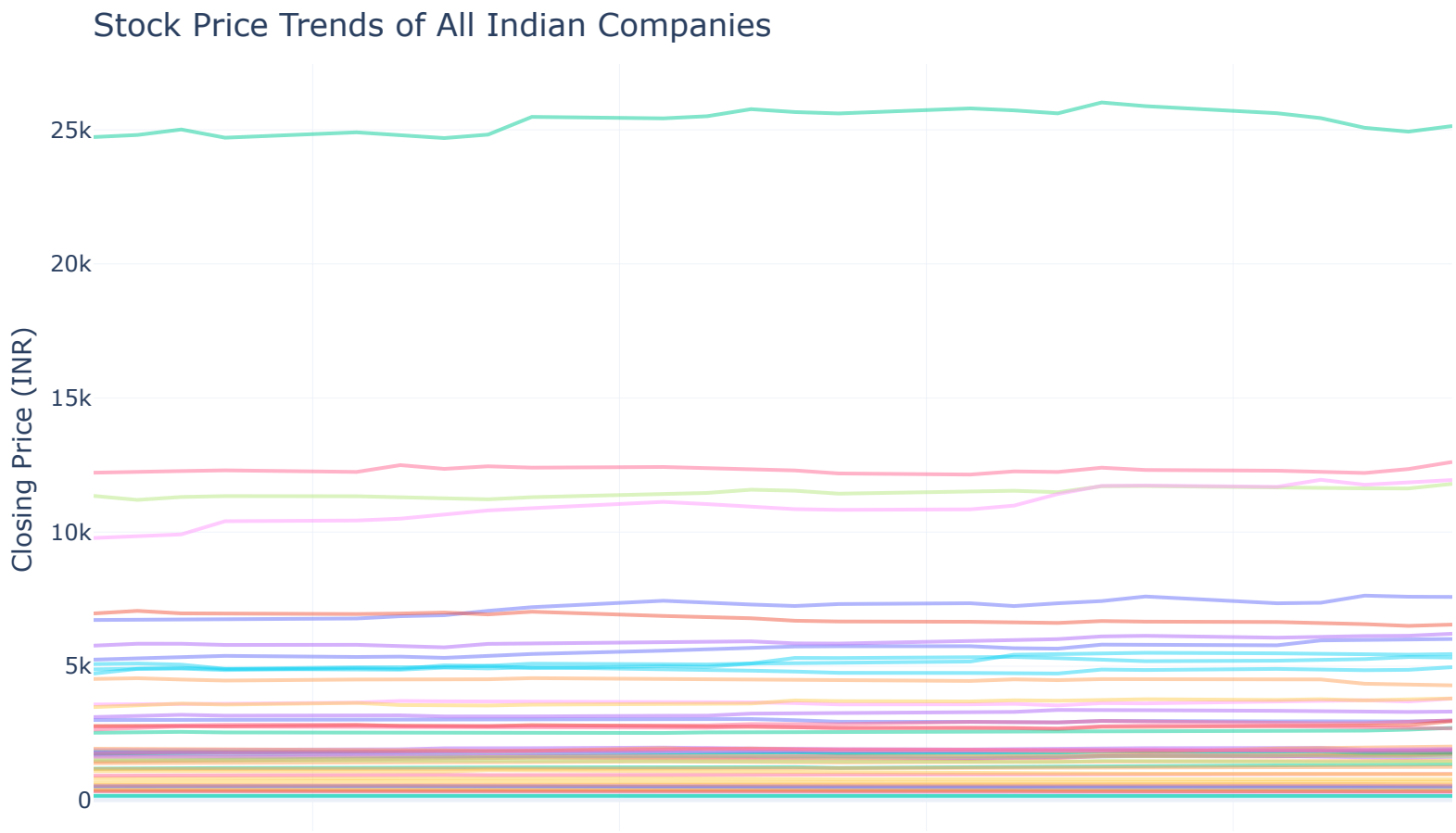
Out[8]:
```
Date              0
RELIANCE.NS       0
HDFCBANK.NS       0
ICICIBANK.NS      0
INFY.NS           0
TCS.NS            0
KOTAKBANK.NS      0
HINDUNILVR.NS     0
ITC.NS            0
LT.NS             0
SBIN.NS           0
BAJFINANCE.NS     0
BHARTIARTL.NS     0
HCLTECH.NS        0
ASIANPAINT.NS     0
AXISBANK.NS       0
DMART.NS          0
MARUTI.NS         0
ULTRACEMCO.NS     0
```

In [9]:
```
#There are 24 null values in the closing prices of HDFC. Let's fill in these null values using the forwara
data.fillna(method='ffill', inplace=True)
```

In [10]:
```
#Now, let's have a look at the stock price trends of all the companies in the data:

import plotly.graph_objs as go
import plotly.express as px
```

In [11]:

```
fig = go.Figure()

for company in data.columns[1:]:
    fig.add_trace(go.Scatter(x=data['Date'], y=data[company],
                             mode='lines',
                             name=company,
                             opacity=0.5))

fig.update_layout(
    title='Stock Price Trends of All Indian Companies',
    xaxis_title='Date',
    yaxis_title='Closing Price (INR)',
    xaxis=dict(tickangle=45),
    legend=dict(
        x=1.05,
        y=1,
        traceorder="normal",
        font=dict(size=10),
        orientation="v"
    ),
    margin=dict(l=0, r=0, t=30, b=0),
    hovermode='x',
    template='plotly_white'
)

fig.show()
```


Stock Price Trends of All Indian Companies
Closing Price (INR)
25k
20k
15k
10k
5k
0

In [12]:
```
#Let's look at the companies with the highest risks for investing:
all_companies = data.columns[1:]

volatility_all_companies = data[all_companies].std()

volatility_all_companies.sort_values(ascending=False).head(10)
```

Out[12]:
```
BAJAJ-AUTO.NS    659.810841
SHREECEM.NS      429.919834
BAJFINANCE.NS    306.658594
DIVISLAB.NS      247.674895
HEROMOTOCO.NS    247.092728
DRREDDY.NS       175.124908
ULTRACEMCO.NS    172.673053
DMART.NS         155.593701
BRITANNIA.NS     144.164343
MARUTI.NS        109.587342
dtype: float64
```

In [13]:
```
# Now, let's look at the companies with the highest growth rate for investing:

growth_all_companies = data[all_companies].pct_change() * 100

average_growth_all_companies = growth_all_companies.mean()

average_growth_all_companies.sort_values(ascending=False).head(10)
```

Out[13]:
```
BAJAJ-AUTO.NS    0.883421
BAJAJFINSV.NS    0.791730
BHARTIARTL.NS    0.735219
DIVISLAB.NS      0.634851
HEROMOTOCO.NS    0.602192
ICICIBANK.NS     0.557742
BAJFINANCE.NS    0.536819
TITAN.NS         0.393800
HINDUNILVR.NS    0.351634
BRITANNIA.NS     0.327747
dtype: float64
```

In [15]:
```
#Now, let's have a look at the companies with the highest return on investments:
initial_prices_all = data[all_companies].iloc[0]
final_prices_all = data[all_companies].iloc[-1]
```

In [16]:
```
initial_prices_all
```

```
AXISBANK.NS       1168.000000
DMART.NS          5079.200195
MARUTI.NS        12214.950195
ULTRACEMCO.NS    11349.700195
HDFC.NS                  NaN
TITAN.NS          3474.899902
SUNPHARMA.NS      1766.349976
M&M.NS            2771.300049
NESTLEIND.NS      2518.500000
WIPRO.NS           524.650024
ADANIGREEN.NS     1924.599976
TATASTEEL.NS       153.929993
JSWSTEEL.NS        917.150024
POWERGRID.NS       340.500000
ONGC.NS            327.555695
NTPC.NS            406.250000
COALINDIA.NS       524.599976
BPCL.NS            349.399994
IOC.NS             172.229996
```

In [17]:
```
final_prices_all
```

```
IOC.NS             167.050003
TECHM.NS          1622.050049
INDUSINDBK.NS     1480.199951
DIVISLAB.NS       5450.750000
GRASIM.NS         2678.250000
CIPLA.NS          1638.650024
BAJAJFINSV.NS     1916.800049
TATAMOTORS.NS      970.849976
HEROMOTOCO.NS     6013.250000
DRREDDY.NS        6551.149902
SHREECEM.NS      25141.699219
BRITANNIA.NS      6210.549805
UPL.NS             587.099976
EICHERMOT.NS      4963.149902
SBILIFE.NS        1870.250000
ADANIPORTS.NS     1438.699951
BAJAJ-AUTO.NS    11941.700195
HINDALCO.NS        694.400024
Name: 23, dtype: float64
```

In [18]:
```
roi_all_companies = ((final_prices_all - initial_prices_all) / initial_prices_all) * 100

roi_all_companies.sort_values(ascending=False).head(10)
```

Out[18]:
```
BAJAJ-AUTO.NS    22.107017
BAJAJFINSV.NS    19.642973
BHARTIARTL.NS    18.120965
DIVISLAB.NS      15.404976
HEROMOTOCO.NS    14.660402
ICICIBANK.NS     13.480860
BAJFINANCE.NS    12.797149
TITAN.NS          9.275089
HINDUNILVR.NS     8.235039
BRITANNIA.NS      7.713587
dtype: float64
```

Creating a Mutual Fund Plan Based on High ROI and Low Risk .

To create a strategy for selecting companies with high ROI and low risk, we can use a combination of ROI and volatility (standard deviation) metrics. The goal is to find companies that offer a high return on investment (ROI) but with low volatility to minimize risk.

Here are the steps we can follow for creating a mutual fund plan:

1 :Define ROI and Volatility Thresholds: We will set thresholds for ROI and volatility to select companies that provide good returns with lower risks.

2 :Rank Companies by ROI and Volatility: Rank all companies based on their ROI and volatility scores.

3 :Assign Investment Ratios: Allocate more investment to companies with higher ROI and lower volatility.

Let's start by defining thresholds and selecting companies that meet the criteria of high ROI and low volatility:

In [19]:
```
roi_threshold = roi_all_companies.median()
volatility_threshold = volatility_all_companies.median()

selected_companies = roi_all_companies[(roi_all_companies > roi_threshold) & (volatility_all_companies < v

selected_companies.sort_values(ascending=False)
```

Out[19]:
```
ICICIBANK.NS     13.480860
INDUSINDBK.NS     7.159914
JSWSTEEL.NS       7.021748
AXISBANK.NS       6.592466
HDFCBANK.NS       6.319839
SUNPHARMA.NS      5.627425
KOTAKBANK.NS      5.474481
CIPLA.NS          4.850117
NTPC.NS           4.356926
dtype: float64
```

```
The following companies meet the criteria of high ROI and low volatility:

ICICI Bank (ROI: 13.48%)
IndusInd Bank (ROI: 7.16%)
JSW Steel (ROI: 7.02%)
```

```
Axis Bank (ROI: 6.59%)
HDFC Bank (ROI: 6.32%)
Sun Pharma (ROI: 5.63%)
Kotak Bank (ROI: 5.47%)
Cipla (ROI: 4.85%)
NTPC (ROI: 4.36%)
```

To balance the investment between these companies, we can use an inverse volatility ratio for allocation. Companies with lower volatility will get a higher weight. Let’s calculate the weight for each company:

In [20]:
```
selected_volatility = volatility_all_companies[selected_companies.index]
inverse_volatility = 1 / selected_volatility

investment_ratios = inverse_volatility / inverse_volatility.sum()

investment_ratios.sort_values(ascending=False)
```

Out[20]:
```
NTPC.NS          0.280768
JSWSTEEL.NS      0.159985
AXISBANK.NS      0.092231
HDFCBANK.NS      0.089330
CIPLA.NS         0.084783
KOTAKBANK.NS     0.076642
INDUSINDBK.NS    0.074432
SUNPHARMA.NS     0.072553
ICICIBANK.NS     0.069276
dtype: float64
```

The investment ratios based on inverse volatility are as follows:

NTPC: 28.08%
JSW Steel: 15.99%
Axis Bank: 9.22%
HDFC Bank: 8.93%
Cipla: 8.48%
Kotak Bank: 7.66%
IndusInd Bank: 7.44%
Sun Pharma: 7.26%

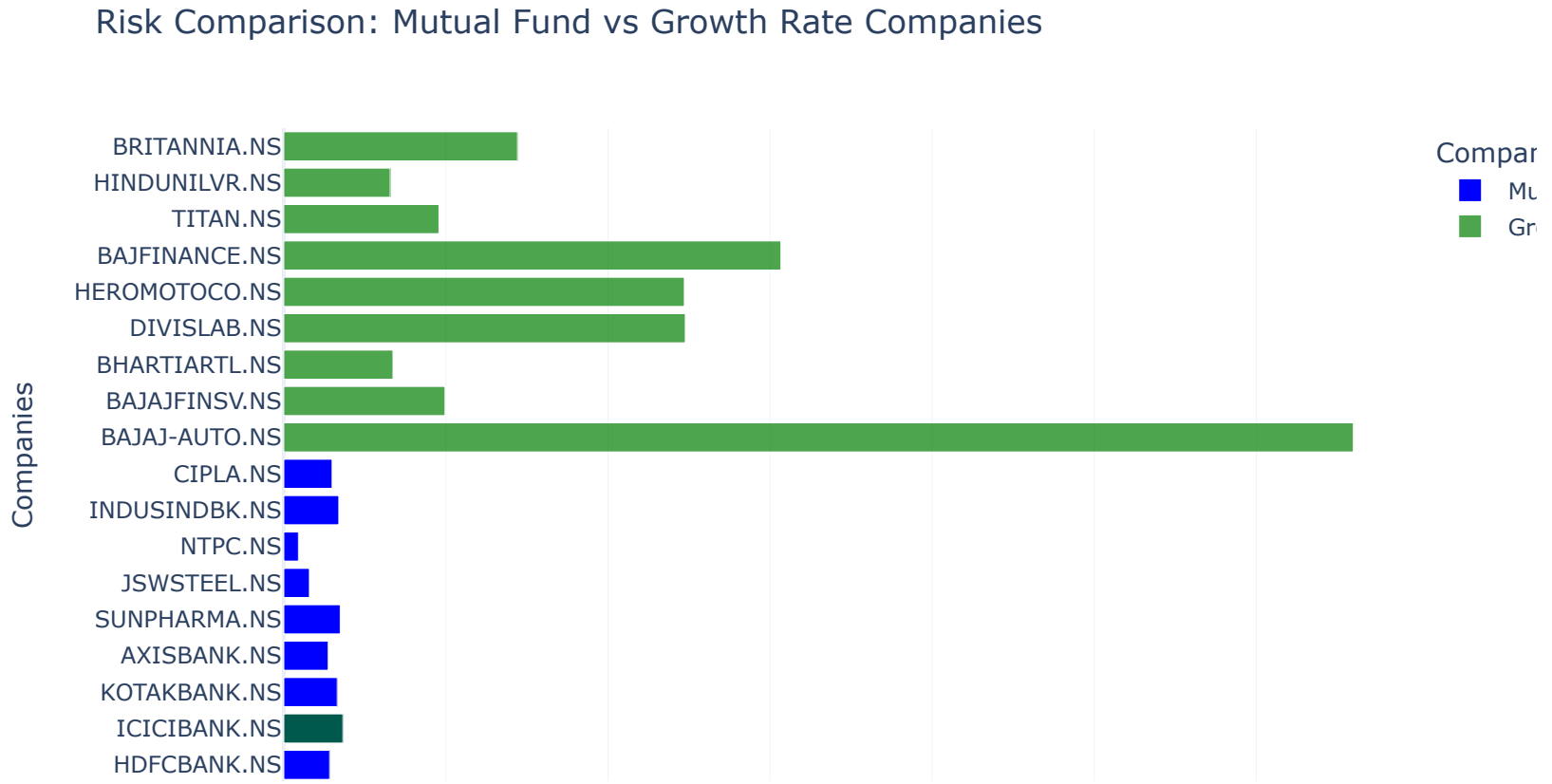
```
ICICI Bank: 6.93%
```

Analyzing Our Mutual Fund Plan

We have created a mutual fund plan for long-term investments. Now, let’s analyze and compare our mutual fund plan by comparing it with the high-performing companies in the stock market. Let’s start by comparing the risks in our mutual fund with the risk in the high growth companies:

In [21]:

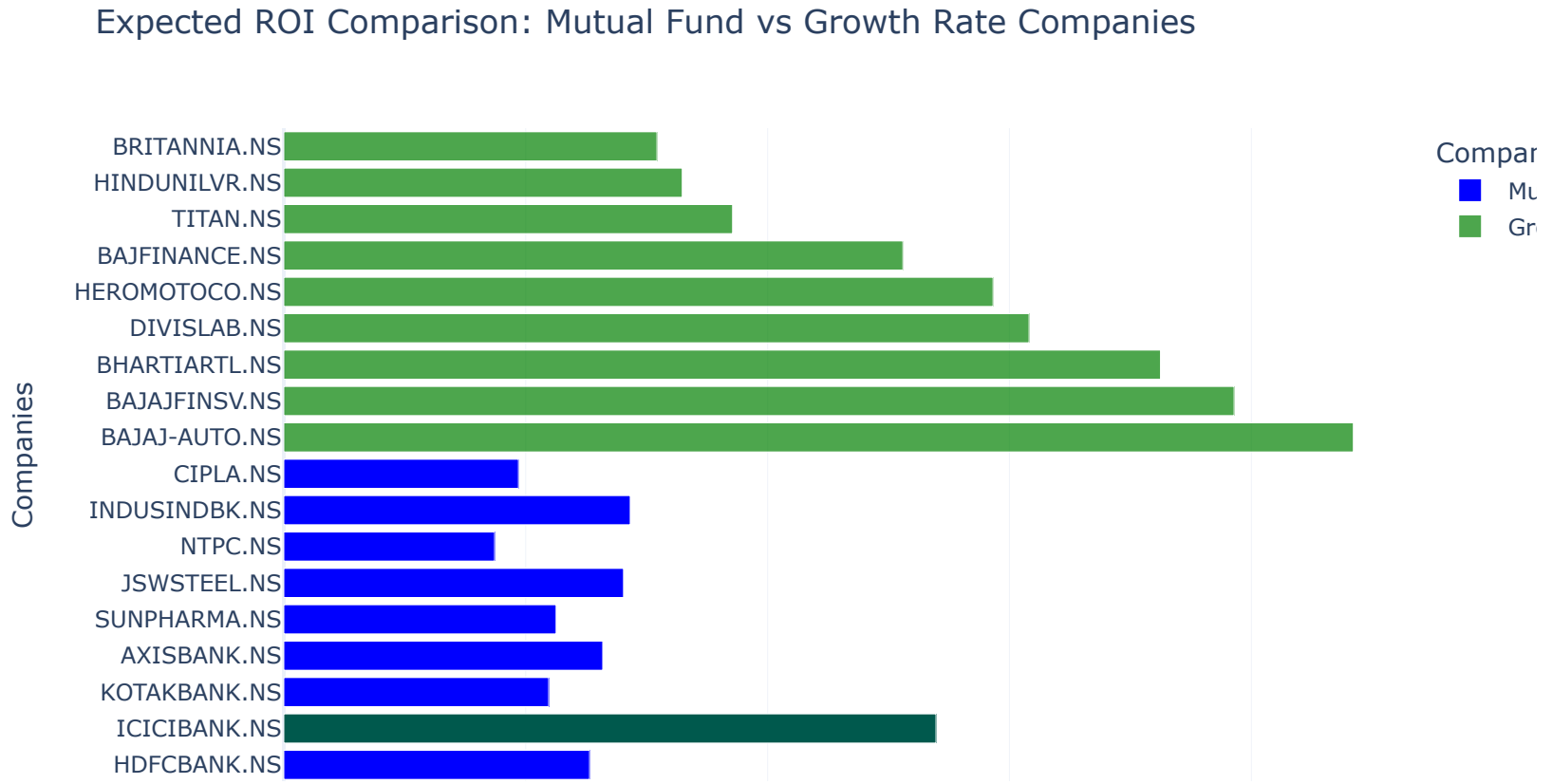
```
top_growth_companies = average_growth_all_companies.sort_values(ascending=False).head(10)
risk_growth_rate_companies = volatility_all_companies[top_growth_companies.index]
risk_mutual_fund_companies = volatility_all_companies[selected_companies.index]

fig = go.Figure()

fig.add_trace(go.Bar(
    y=risk_mutual_fund_companies.index,
    x=risk_mutual_fund_companies,
    orientation='h',  # Horizontal bar
    name='Mutual Fund Companies',
    marker=dict(color='blue')
))

fig.add_trace(go.Bar(
    y=risk_growth_rate_companies.index,
    x=risk_growth_rate_companies,
    orientation='h',
    name='Growth Rate Companies',
    marker=dict(color='green'),
    opacity=0.7
))

fig.update_layout(
    title='Risk Comparison: Mutual Fund vs Growth Rate Companies',
    xaxis_title='Volatility (Standard Deviation)',
    yaxis_title='Companies',
    barmode='overlay',
    legend=dict(title='Company Type'),
    template='plotly_white'
)

fig.show()
```



Now, let’s compare the ROI of both the groups as well:

In [22]:

```
expected_roi_mutual_fund = roi_all_companies[selected_companies.index]

expected_roi_growth_companies = roi_all_companies[top_growth_companies.index]

fig = go.Figure()

fig.add_trace(go.Bar(
    y=expected_roi_mutual_fund.index,
    x=expected_roi_mutual_fund,
    orientation='h',
    name='Mutual Fund Companies',
    marker=dict(color='blue')
))

fig.add_trace(go.Bar(
    y=expected_roi_growth_companies.index,
    x=expected_roi_growth_companies,
    orientation='h',
    name='Growth Rate Companies',
    marker=dict(color='green'),
    opacity=0.7
))

fig.update_layout(
    title='Expected ROI Comparison: Mutual Fund vs Growth Rate Companies',
    xaxis_title='Expected ROI (%)',
    yaxis_title='Companies',
    barmode='overlay',
    legend=dict(title='Company Type'),
    template='plotly_white'
)

fig.show()
```



The comparison between the risk (volatility) and expected ROI for mutual fund companies (in blue) and growth rate companies (in green) shows a clear trade-off. Mutual fund companies offer lower volatility, meaning they are less risky, but also provide lower expected returns. In contrast, growth rate companies demonstrate higher volatility, indicating more risk, but they offer much higher potential returns, especially companies like Bajaj Auto and Bajaj Finserv. This highlights a common investment dilemma: lower risk comes with a lower reward, while higher risk could yield higher returns.

```
For long-term investments, the goal is typically to find companies that offer a balance of stable returns
and manageable risk. The companies in our mutual fund exhibit low volatility, meaning they are less risky,
and their moderate returns make them solid choices for long-term, stable growth. They are well-suited for
conservative investors who want steady returns without significant fluctuations in value.
```

Calculating Expected Returns

Now, let's calculate the expected returns a person will get from our mutual fund if he/she invests ₹5000 every month.

To calculate the expected value a person will accumulate over 1 year, 3 years, 5 years, and 10 years through the mutual fund plan, we can follow these steps:

1.Assume the person is investing 5000 rupees every month.

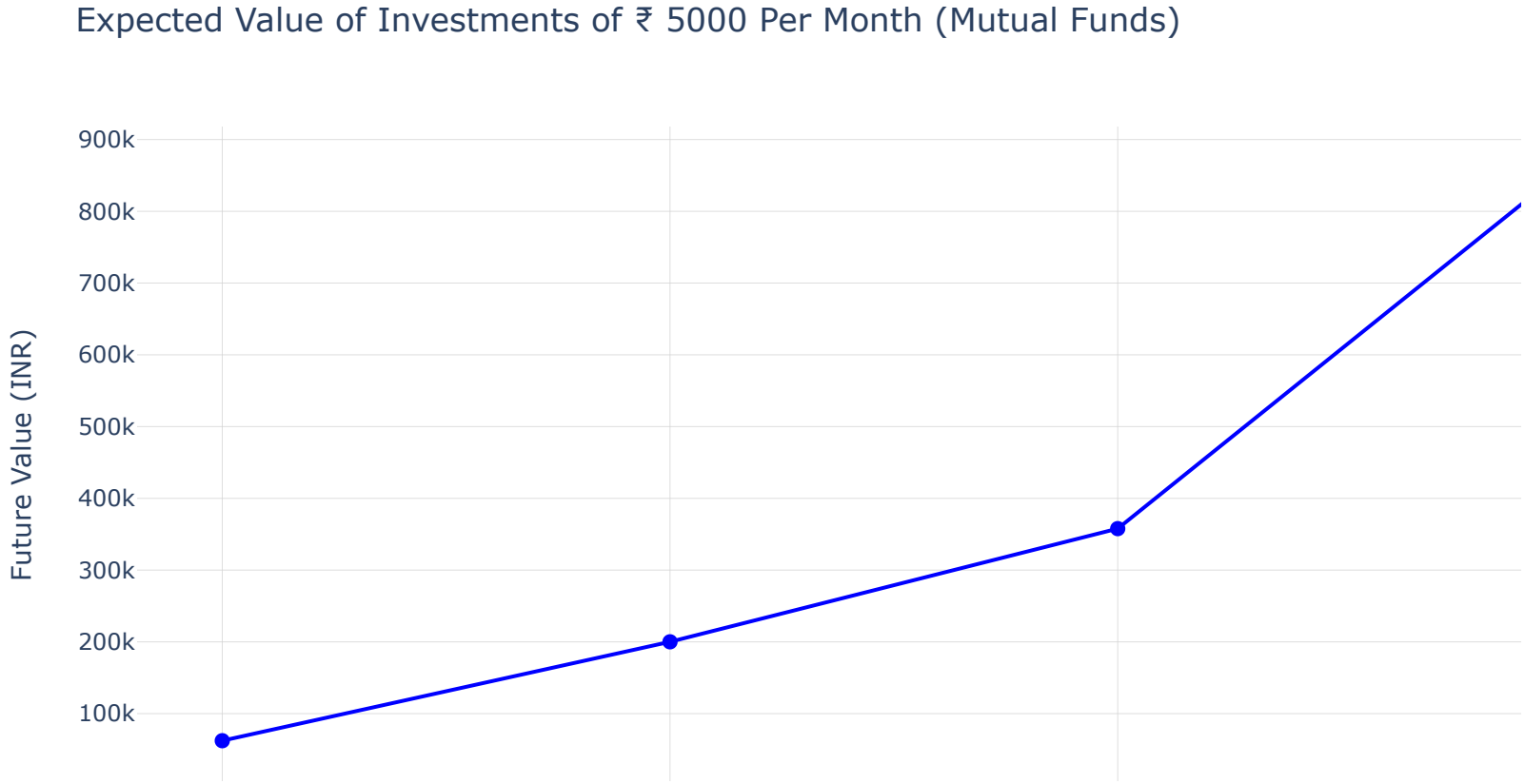
2.Use the expected ROI from the mutual fund companies to simulate the growth over time.

3.Compute the compounded value of the investments for each period (1y, 3y, 5y, and 10y).

4.Visualize the accumulated value over these periods.

In [23]:

```
import numpy as np

monthly_investment = 5000  # Monthly investment in INR
years = [1, 3, 5, 10]  # Investment periods (in years)
n = 12  # Number of times interest is compounded per year (monthly)

avg_roi = expected_roi_mutual_fund.mean() / 100  # Convert to decimal

def future_value(P, r, n, t):
    return P * (((1 + r/n)**(n*t) - 1) / (r/n)) * (1 + r/n)

future_values = [future_value(monthly_investment, avg_roi, n, t) for t in years]

fig = go.Figure()

fig.add_trace(go.Scatter(
    x=[str(year) + " year" for year in years],
    y=future_values,
    mode='lines+markers',
    line=dict(color='blue'),
    marker=dict(size=8),
    name='Future Value'
))

fig.update_layout(
    title="Expected Value of Investments of ₹ 5000 Per Month (Mutual Funds)",
    xaxis_title="Investment Period",
    yaxis_title="Future Value (INR)",
    xaxis=dict(showgrid=True, gridcolor='lightgrey'),
    yaxis=dict(showgrid=True, gridcolor='lightgrey'),
    template="plotly_white",
    hovermode='x'
)

fig.show()
```



After 1 year, the accumulated value is around ₹62,000, and by 5 years, it grows to over ₹300,000. The long-term benefit is evident, with the investment growing to nearly ₹860,000 over 10 years, which emphasises the value of consistent investing and compounding over time for long-term investors.

Summary

So, this is how a mutual fund plan is designed by investment companies for long-term investors. Mutual funds are investment plans that pool money from multiple investors to purchase a diversified portfolio of stocks, bonds, and other securities, managed by professional fund managers.

In [ ]: