Developing an interactive dashboard for analyzing cloud resource usage with machine learning models

Step 1: Data Collection

In [ ]:

```
import pandas as pd
import matplotlib.pyplot as plt

# Data Collection
data = pd.read_csv('performance_data.csv')
```

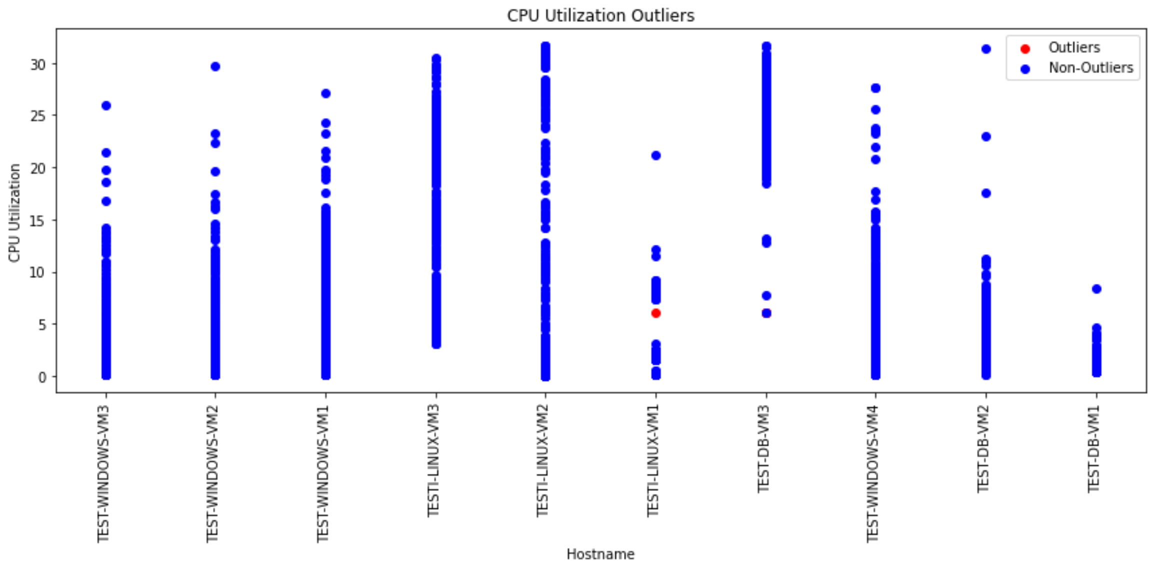
Step 2: Data Preprocessing and Handling Outliers

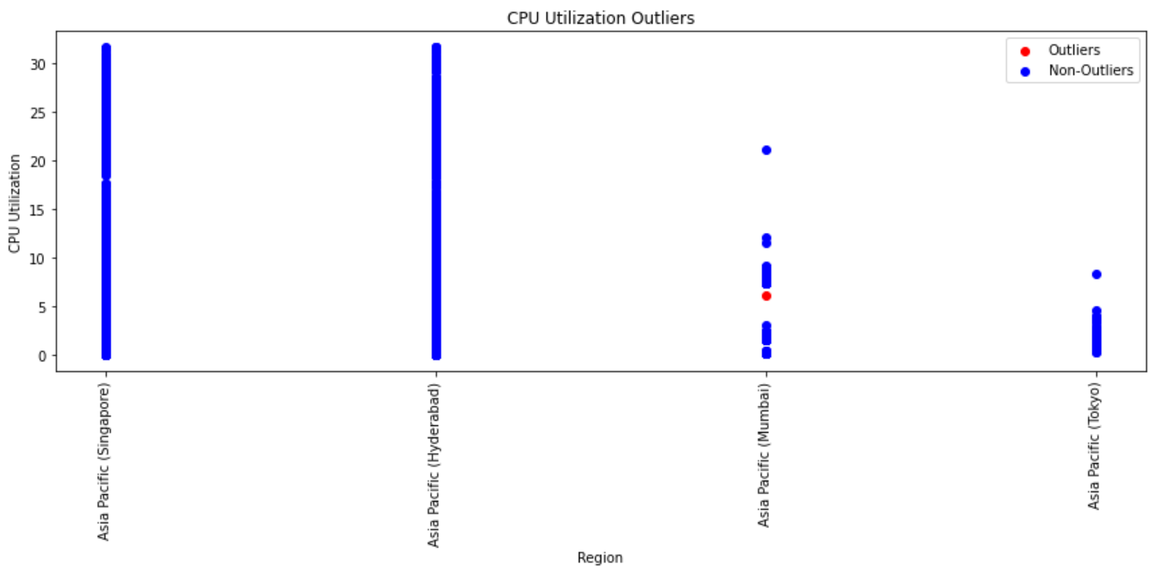
In [3]:

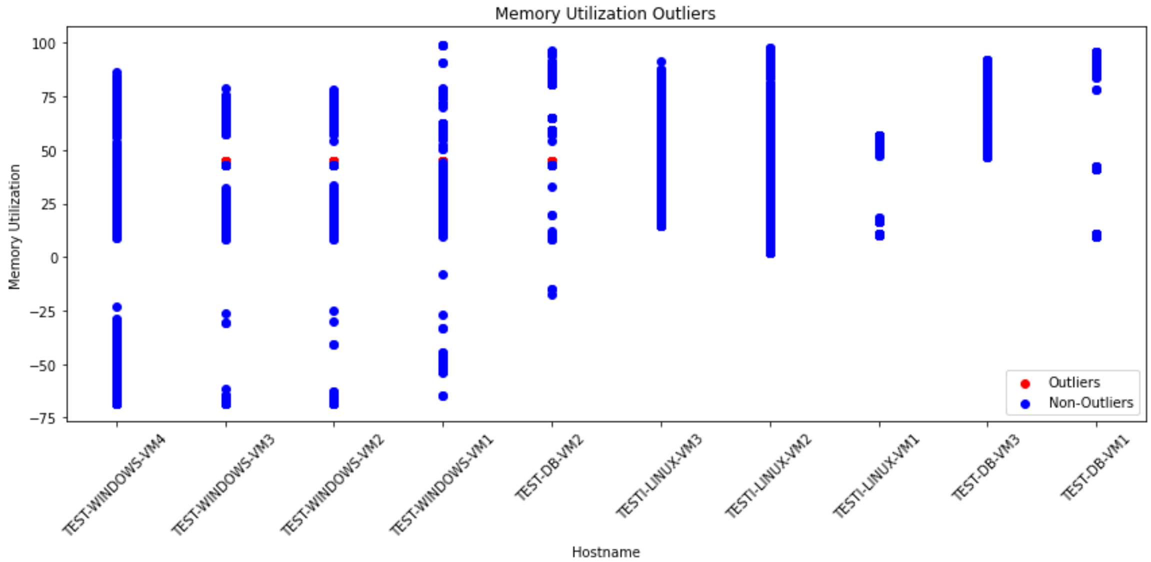
```
# Data Preprocessing
# Convert date columns to datetime format
date_columns = ['Date_Mem Utilization', 'Date_CPU Utilization', 'Date_Disk Utilizatio
n']
for col in date_columns:
    df[col] = pd.to_datetime(df[col])

# Handling Outliers - Detect and Remove/Adjust Outliers
# consider using the Z-score method to detect outliers

def detect_outliers_zscore(data, threshold=3):
    z_scores = (data - data.mean()) / data.std()
    return abs(z_scores) > threshold

# Detect outliers for CPU, Memory, and Disk Utilization columns
outliers_cpu = detect_outliers_zscore(df['CPU Utilization'])
outliers_memory = detect_outliers_zscore(df['Memory Utilization'])
outliers_disk = detect_outliers_zscore(df['Disk Utilization'])

# Remove or replace outliers (e.g., replace with mean, median, or drop the rows)
df['CPU Utilization'][outliers_cpu] = df['CPU Utilization'].mean()
df['Memory Utilization'][outliers_memory] = df['Memory Utilization'].mean()
df['Disk Utilization'][outliers_disk] = df['Disk Utilization'].mean()

# Visualize Outliers with respect to Hostname and Regions

# Plotting CPU Utilization Outliers
plt.figure(figsize=(12, 6))
plt.scatter(df[outliers_cpu]['Hostname'], df[outliers_cpu]['CPU Utilization'], c='red',
label='Outliers')
plt.scatter(df[~outliers_cpu]['Hostname'], df[~outliers_cpu]['CPU Utilization'], c='blu
e', label='Non-Outliers')
plt.xlabel('Hostname')
plt.ylabel('CPU Utilization')
plt.title('CPU Utilization Outliers')
plt.legend()
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()

# Plotting CPU Utilization Outliers
plt.figure(figsize=(12, 6))
plt.scatter(df[outliers_cpu]['Region'], df[outliers_cpu]['CPU Utilization'], c='red', l
abel='Outliers')
plt.scatter(df[~outliers_cpu]['Region'], df[~outliers_cpu]['CPU Utilization'], c='blu
e', label='Non-Outliers')
plt.xlabel('Region')
plt.ylabel('CPU Utilization')
plt.title('CPU Utilization Outliers')
plt.legend()
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()


# Plotting Memory Utilization Outliers
plt.figure(figsize=(12, 6))
plt.scatter(df[outliers_memory]['Hostname'], df[outliers_memory]['Memory Utilization'],
c='red', label='Outliers')
```

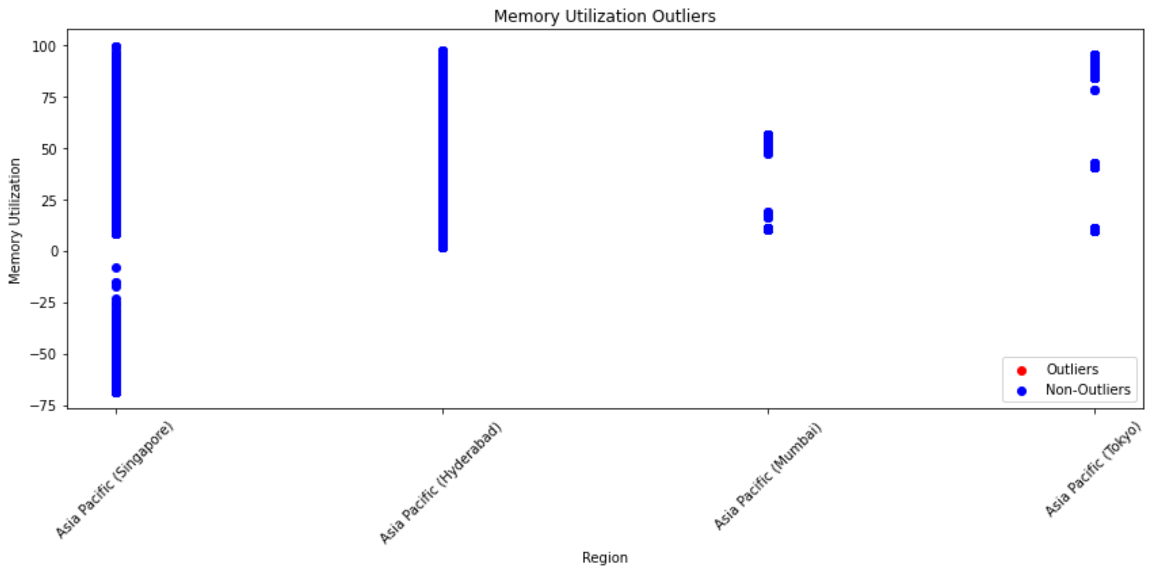
```
plt.scatter(df[~outliers_memory]['Hostname'], df[~outliers_memory]['Memory Utilizatio
n'], c='blue', label='Non-Outliers')
plt.xlabel('Hostname')
plt.ylabel('Memory Utilization')
plt.title('Memory Utilization Outliers')
plt.legend()
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

# Plotting Memory Utilization Outliers
plt.figure(figsize=(12, 6))
plt.scatter(df[outliers_memory]['Region'], df[outliers_memory]['Memory Utilization'], c
='red', label='Outliers')
plt.scatter(df[~outliers_memory]['Region'], df[~outliers_memory]['Memory Utilization'],
c='blue', label='Non-Outliers')
plt.xlabel('Region')
plt.ylabel('Memory Utilization')
plt.title('Memory Utilization Outliers')
plt.legend()
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

# Plotting Disk Utilization Outliers
plt.figure(figsize=(12, 6))
plt.scatter(df[outliers_disk]['Region'], df[outliers_disk]['Disk Utilization'], c='re
d', label='Outliers')
plt.scatter(df[~outliers_disk]['Region'], df[~outliers_disk]['Disk Utilization'], c='bl
ue', label='Non-Outliers')
plt.xlabel('Region')
plt.ylabel('Disk Utilization')
plt.title('Disk Utilization Outliers')
plt.legend()
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()


# Plotting Disk Utilization Outliers
plt.figure(figsize=(12, 6))
plt.scatter(df[outliers_disk]['Hostname'], df[outliers_disk]['Disk Utilization'], c='re
d', label='Outliers')
plt.scatter(df[~outliers_disk]['Hostname'], df[~outliers_disk]['Disk Utilization'], c
='blue', label='Non-Outliers')
plt.xlabel('Hostname')
plt.ylabel('Disk Utilization')
plt.title('Disk Utilization Outliers')
plt.legend()
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

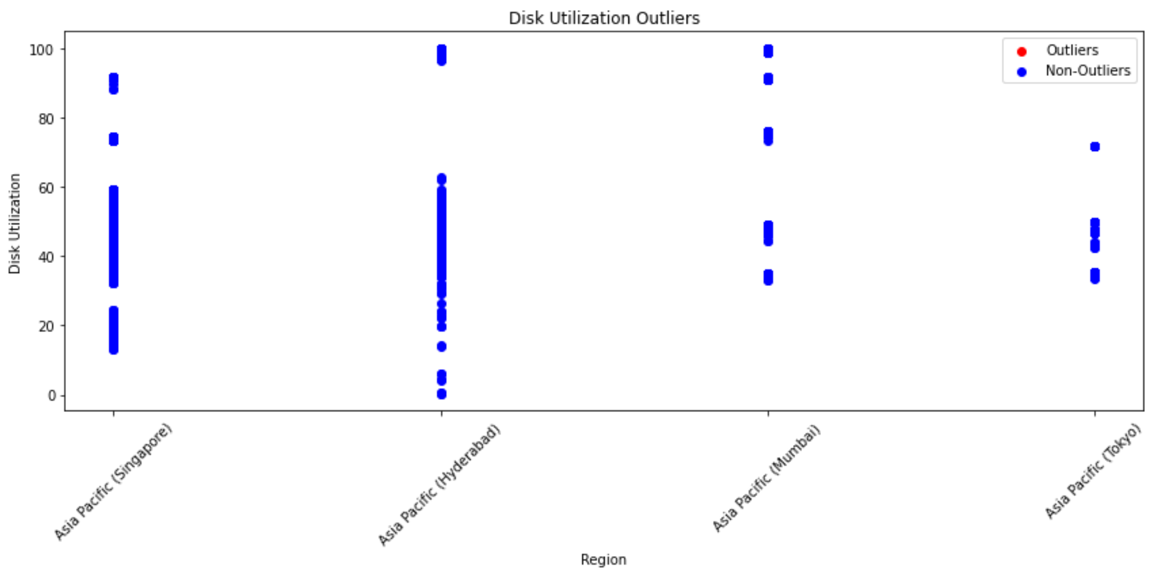
```
C:\Users\poorn\AppData\Local\Temp/ipykernel_14224/1163059819.py:27: Settin
gWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df['CPU Utilization'][outliers_cpu] = df['CPU Utilization'].mean()
C:\Users\poorn\AppData\Local\Temp/ipykernel_14224/1163059819.py:28: Settin
gWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df['Memory Utilization'][outliers_memory] = df['Memory Utilization'].mea
n()
C:\Users\poorn\AppData\Local\Temp/ipykernel_14224/1163059819.py:29: Settin
gWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df['Disk Utilization'][outliers_disk] = df['Disk Utilization'].mean()
```

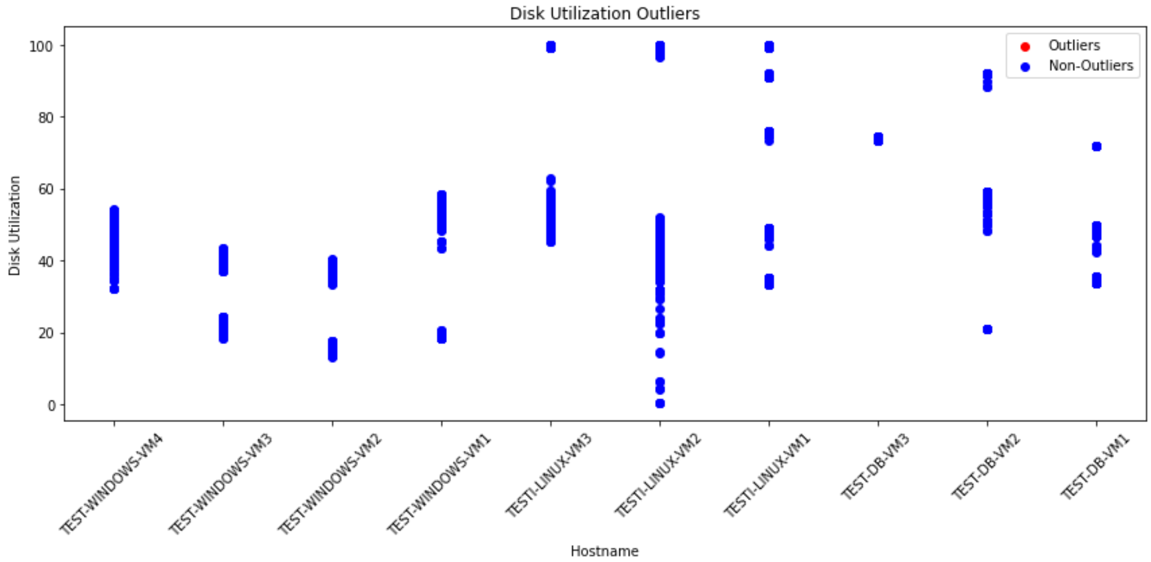





Memory Utilization Outliers
Memory Utilization
100
75
50
25
0
−25
−50
−75
Outliers
Non-Outliers
TEST-WINDOWS-VM4
TEST-WINDOWS-VM3
TEST-WINDOWS-VM2
TEST-WINDOWS-VM1
TEST-DB-VM2
TESTI-LINUX-VM3
TESTI-LINUX-VM2
TESTI-LINUX-VM1
TEST-DB-VM3
TEST-DB-VM1
Hostname


Memory Utilization Outliers
Memory Utilization
100
75
50
25
0
−25
−50
−75
Outliers
Non-Outliers
Asia Pacific (Singapore)
Asia Pacific (Hyderabad)
Asia Pacific (Mumbai)
Asia Pacific (Tokyo)
Region


Disk Utilization Outliers
Disk Utilization
100
80
60
40
20
0
Outliers
Non-Outliers
Asia Pacific (Singapore)
Asia Pacific (Hyderabad)
Asia Pacific (Mumbai)
Asia Pacific (Tokyo)
Region


Disk Utilization Outliers
Outliers
Non-Outliers
Disk Utilization
100
80
60
40
20
0
Hostname
TEST-WINDOWS-VM4
TEST-WINDOWS-VM3
TEST-WINDOWS-VM2
TEST-WINDOWS-VM1
TEST-LINUX-VM3
TEST-LINUX-VM2
TEST-LINUX-VM1
TEST-DB-VM3
TEST-DB-VM2
TEST-DB-VM1

Step3: Data Transformation

In [6]:

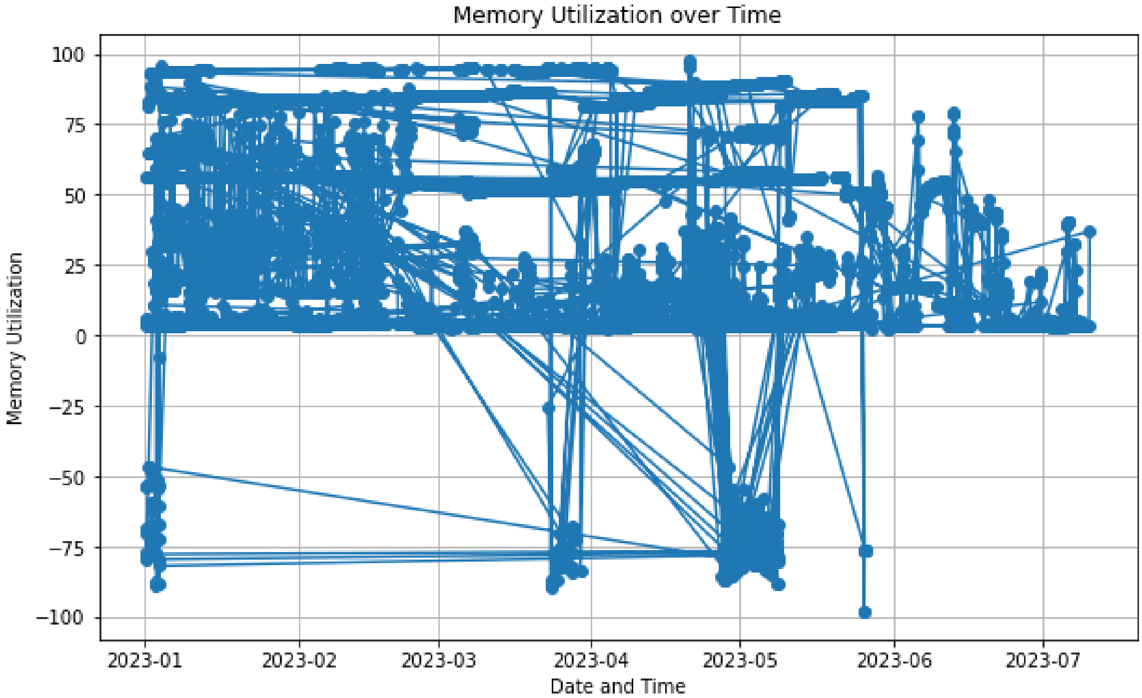
```
# Convert 'Date_Mem Utilization', 'Date_CPU Utilization', and 'Date_Disk Utilization' c
olumns to datetime format
data['Date_Mem Utilization'] = pd.to_datetime(data['Date_Mem Utilization'])
data['Date_CPU Utilization'] = pd.to_datetime(data['Date_CPU Utilization'])
data['Date_Disk Utilization'] = pd.to_datetime(data['Date_Disk Utilization'])


print(data.head())
```

```
     Account ID      Account name          Hostname Instance type       OS  
\
0  9.580000e+11  MtechProject1605  TEST-WINDOWS-VM4     m5.xlarge  Windows
1  9.580000e+11  MtechProject1605  TEST-WINDOWS-VM4     m5.xlarge  Windows
2  9.580000e+11  MtechProject1605  TEST-WINDOWS-VM4     m5.xlarge  Windows
3  9.580000e+11  MtechProject1605  TEST-WINDOWS-VM4     m5.xlarge  Windows
4  9.580000e+11  MtechProject1605  TEST-WINDOWS-VM4     m5.xlarge  Windows

                  Region  Memory Utilization Date_Mem Utilization  \
0  Asia Pacific (Singapore)          -69.559819  2023-05-02 07:30:00
1  Asia Pacific (Singapore)          -64.377584  2023-05-02 09:30:00
2  Asia Pacific (Singapore)          -65.760491  2023-05-02 11:30:00
3  Asia Pacific (Singapore)          -78.340094  2023-05-02 14:30:00
4  Asia Pacific (Singapore)          -68.747094  2023-05-03 05:30:00

   CPU Utilization Date_CPU Utilization  Disk Utilization  \
0         0.403854  2023-01-03 02:30:00         51.061435
1         0.234507  2023-01-03 00:30:00         48.548180
2         0.249844  2023-01-02 23:30:00         43.443316
3         3.442852  2023-01-02 12:30:00         49.372037
4         5.810633  2023-01-02 07:30:00         47.809481

  Date_Disk Utilization
0   2023-02-13 05:30:00
1   2023-02-16 05:30:00
2   2023-02-16 05:30:00
3   2023-02-15 05:30:00
4   2023-02-23 05:30:00
```

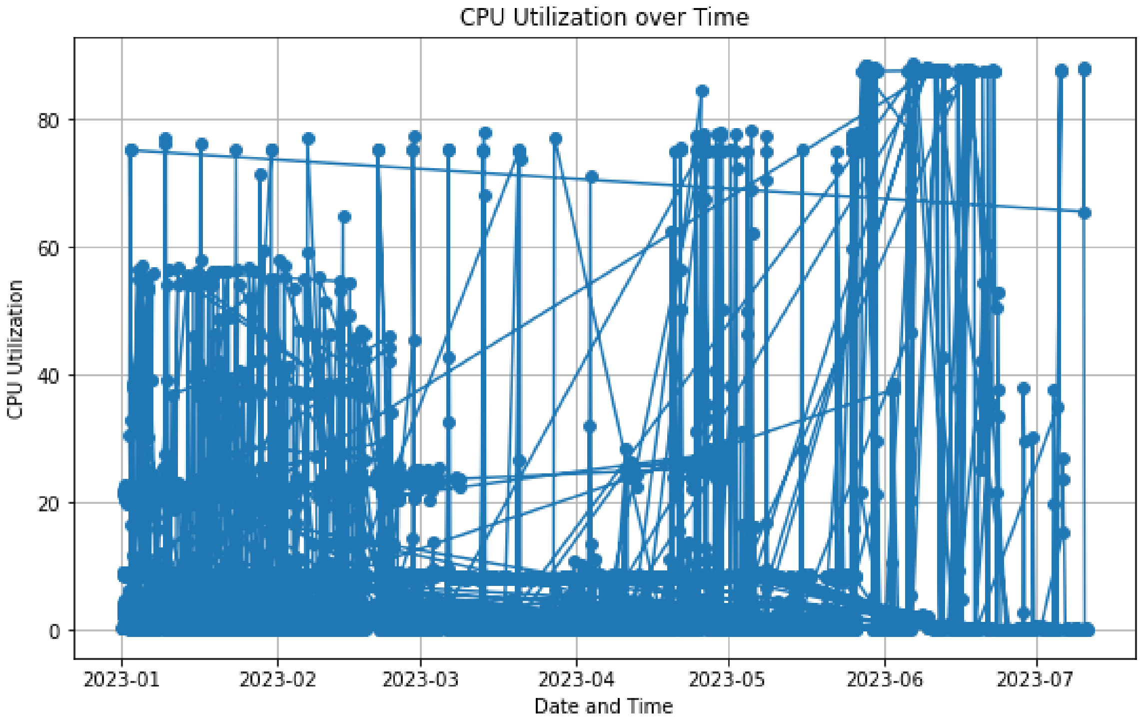
Step 4: Data Exploration

In [5]:

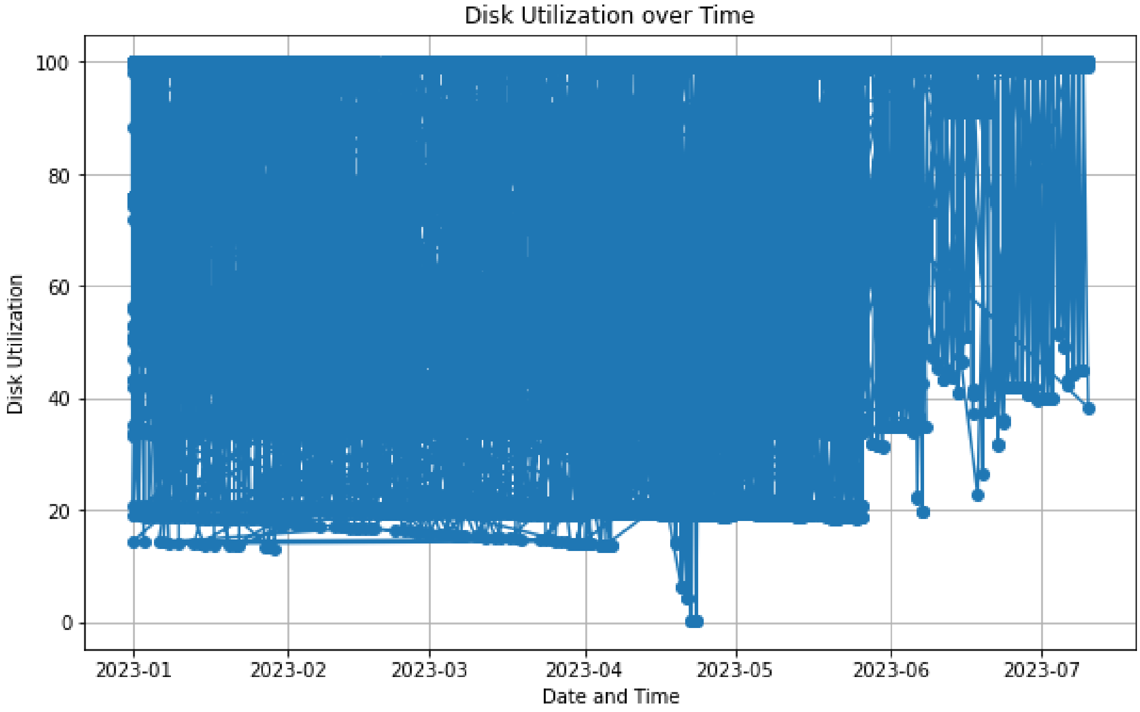
```
import pandas as pd
import matplotlib.pyplot as plt


# Data Exploration - Memory Utilization
plt.figure(figsize=(10, 6))
plt.plot(data['Date_Mem Utilization'], data['Memory Utilization'], marker='o', linestyl
e='-')
plt.xlabel('Date and Time')
plt.ylabel('Memory Utilization')
plt.title('Memory Utilization over Time')
plt.grid(True)
plt.show()

# Data Exploration - CPU Utilization
plt.figure(figsize=(10, 6))
plt.plot(data['Date_CPU Utilization'], data['CPU Utilization'], marker='o', linestyle
='-')
plt.xlabel('Date and Time')
plt.ylabel('CPU Utilization')
plt.title('CPU Utilization over Time')
plt.grid(True)
plt.show()

# Data Exploration - Disk Utilization
plt.figure(figsize=(10, 6))
plt.plot(data['Date_Disk Utilization'], data['Disk Utilization'], marker='o', linestyle
='-')
plt.xlabel('Date and Time')
plt.ylabel('Disk Utilization')
plt.title('Disk Utilization over Time')
plt.grid(True)
plt.show()
```



Memory Utilization over Time
Memory Utilization
100
75
50
25
0
−25
−50
−75
−100
2023-01
2023-02
2023-03
2023-04
2023-05
2023-06
2023-07
Date and Time



CPU Utilization over Time
CPU Utilization
80
60
40
20
0
2023-01
2023-02
2023-03
2023-04
2023-05
2023-06
2023-07
Date and Time


Disk Utilization over Time
Disk Utilization
Date and Time
0
20
40
60
80
100
2023-01
2023-02
2023-03
2023-04
2023-05
2023-06
2023-07

Step 5: Data Preprocessing and Feature Engineering

In [7]:

```
#Data Preprocessing
# Convert date columns to datetime format with utc=True
data['Date_Mem Utilization'] = pd.to_datetime(data['Date_Mem Utilization'], utc=True)
data['Date_CPU Utilization'] = pd.to_datetime(data['Date_CPU Utilization'], utc=True)
data['Date_Disk Utilization'] = pd.to_datetime(data['Date_Disk Utilization'], utc=True)

#Feature Engineering - Date Components
data['Year_Mem'] = data['Date_Mem Utilization'].dt.year
data['Month_Mem'] = data['Date_Mem Utilization'].dt.month
data['Day_Mem'] = data['Date_Mem Utilization'].dt.day
data['Hour_Mem'] = data['Date_Mem Utilization'].dt.hour
data['Minute_Mem'] = data['Date_Mem Utilization'].dt.minute

data['Year_CPU'] = data['Date_CPU Utilization'].dt.year
data['Month_CPU'] = data['Date_CPU Utilization'].dt.month
data['Day_CPU'] = data['Date_CPU Utilization'].dt.day
data['Hour_CPU'] = data['Date_CPU Utilization'].dt.hour
data['Minute_CPU'] = data['Date_CPU Utilization'].dt.minute

data['Year_Disk'] = data['Date_Disk Utilization'].dt.year
data['Month_Disk'] = data['Date_Disk Utilization'].dt.month
data['Day_Disk'] = data['Date_Disk Utilization'].dt.day
data['Hour_Disk'] = data['Date_Disk Utilization'].dt.hour
data['Minute_Disk'] = data['Date_Disk Utilization'].dt.minute

# Print the first few rows to check the new features
print(data.head())
```

```
      Account ID      Account name          Hostname Instance type       OS  \
0  9.580000e+11  MtechProject1605  TEST-WINDOWS-VM4     m5.xlarge  Windows
1  9.580000e+11  MtechProject1605  TEST-WINDOWS-VM4     m5.xlarge  Windows
2  9.580000e+11  MtechProject1605  TEST-WINDOWS-VM4     m5.xlarge  Windows
3  9.580000e+11  MtechProject1605  TEST-WINDOWS-VM4     m5.xlarge  Windows
4  9.580000e+11  MtechProject1605  TEST-WINDOWS-VM4     m5.xlarge  Windows

                     Region  Memory Utilization      Date_Mem Utilization  \
0  Asia Pacific (Singapore)          -69.559819 2023-05-02 07:30:00+00:00
1  Asia Pacific (Singapore)          -64.377584 2023-05-02 09:30:00+00:00
2  Asia Pacific (Singapore)          -65.760491 2023-05-02 11:30:00+00:00
3  Asia Pacific (Singapore)          -78.340094 2023-05-02 14:30:00+00:00
4  Asia Pacific (Singapore)          -68.747094 2023-05-03 05:30:00+00:00

   CPU Utilization      Date_CPU Utilization  ...  Year_CPU Month_CPU  \
0         0.403854 2023-01-03 02:30:00+00:00  ...      2023         1
1         0.234507 2023-01-03 00:30:00+00:00  ...      2023         1
2         0.249844 2023-01-02 23:30:00+00:00  ...      2023         1
3         3.442852 2023-01-02 12:30:00+00:00  ...      2023         1
4         5.810633 2023-01-02 07:30:00+00:00  ...      2023         1

   Day_CPU  Hour_CPU  Minute_CPU  Year_Disk  Month_Disk  Day_Disk  Hour_Di
sk  \
0        3         2          30       2023           2        13
5
1        3         0          30       2023           2        16
5
2        2        23          30       2023           2        16
5
3        2        12          30       2023           2        15
5
4        2         7          30       2023           2        23
5

   Minute_Disk
0           30
1           30
2           30
3           30
4           30

[5 rows x 27 columns]
```

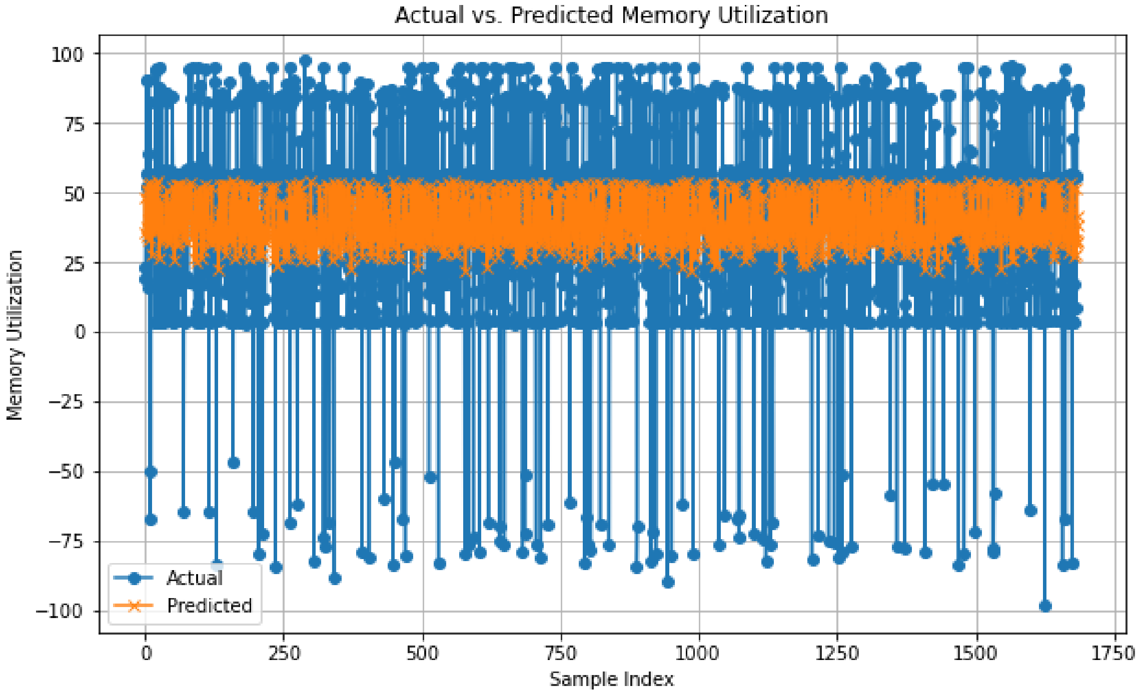
Step 6: Model Selection

a)Linear Regression (Time Series Forecasting)

In [8]:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, r2_score
import matplotlib.pyplot as plt

# Select the features and target variable
X = data[['Year_Mem', 'Month_Mem', 'Day_Mem', 'Hour_Mem', 'Minute_Mem']].values
y = data['Memory Utilization'].values

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=4
2)

# Machine Learning Model - Linear Regression (Time Series Forecasting)
model = LinearRegression()
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)

# Model Evaluation
mae = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Absolute Error (MAE): {mae}")
print(f"R-squared (R2): {r2}")

# Plot the actual vs. predicted memory utilization
plt.figure(figsize=(10, 6))
plt.plot(y_test, label='Actual', marker='o')
plt.plot(y_pred, label='Predicted', marker='x')
plt.xlabel('Sample Index')
plt.ylabel('Memory Utilization')
plt.title('Actual vs. Predicted Memory Utilization')
plt.legend()
plt.grid(True)
plt.show()
```

```
Mean Absolute Error (MAE): 31.040568211465136
R-squared (R2): 0.033535602828514643
```



MAE is 31.040568211465136, which means, on average, the model's predictions deviate from the actual values by approximately 31.04 units.

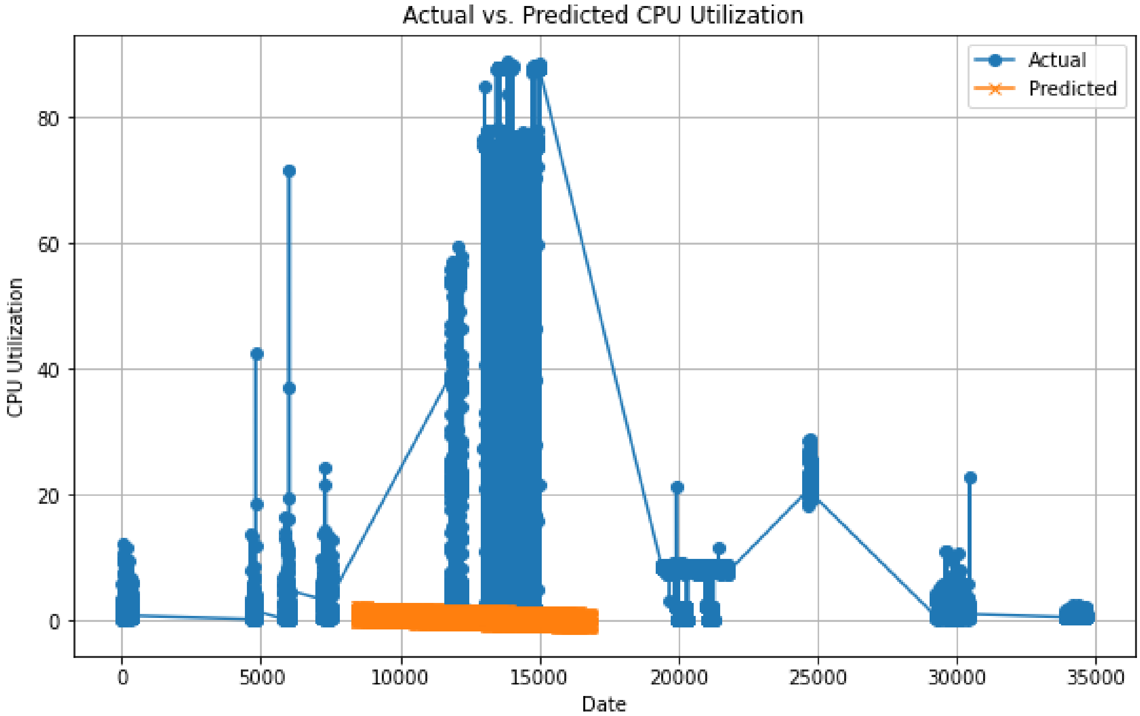
R2 value is 0.033535602828514643, which suggests that the model explains only about 3.35% of the variance in the data. This indicates that the model's predictions are not very accurate in capturing the underlying patterns in the data.

b)Time Series Forecasting - SARIMA

In [9]:

```
import pandas as pd
import matplotlib.pyplot as plt
import statsmodels.api as sm

# Fit SARIMA model
model = sm.tsa.SARIMAX(data['CPU Utilization'], order=(1, 1, 1), seasonal_order=(1, 1,
1, 24))
results = model.fit()

# Make predictions
forecast = results.get_forecast(steps=len(data))  # Forecast for the entire dataset
forecast_mean = forecast.predicted_mean

# Plot actual vs. predicted CPU utilization
plt.figure(figsize=(10, 6))
plt.plot(data['CPU Utilization'], label='Actual', marker='o')
plt.plot(forecast_mean, label='Predicted', marker='x')
plt.xlabel('Date')
plt.ylabel('CPU Utilization')
plt.title('Actual vs. Predicted CPU Utilization')
plt.legend()
plt.grid(True)
plt.show()

# Evaluate the model
mae = mean_absolute_error(data['CPU Utilization'], forecast_mean)
r2 = r2_score(data['CPU Utilization'], forecast_mean)

print(f"Mean Absolute Error (MAE): {mae}")
print(f"R-squared (R2): {r2}")
```

```
C:\Users\poorn\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.
py:578: ValueWarning: An unsupported index was provided and will be ignore
d when e.g. forecasting.
  warnings.warn('An unsupported index was provided and will be'
C:\Users\poorn\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.
py:578: ValueWarning: An unsupported index was provided and will be ignore
d when e.g. forecasting.
  warnings.warn('An unsupported index was provided and will be'
C:\Users\poorn\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.
py:376: ValueWarning: No supported index is available. Prediction results
will be given with an integer index beginning at `start`.
  warnings.warn('No supported index is available.'
```




```
Mean Absolute Error (MAE): 7.138885717052515
R-squared (R2): -0.20020900153555088
```

MAE is 7.138885717052515, which means, on average, the model's predictions deviate from the actual values by approximately 7.14 units.

R2 value is -0.20020900153555088, which suggests that the model does not fit the data well and performs worse than just using the mean of the target variable for predictions. A negative R2 indicates that the model's predictions are not meaningful and the model is not explaining the variance in the data.

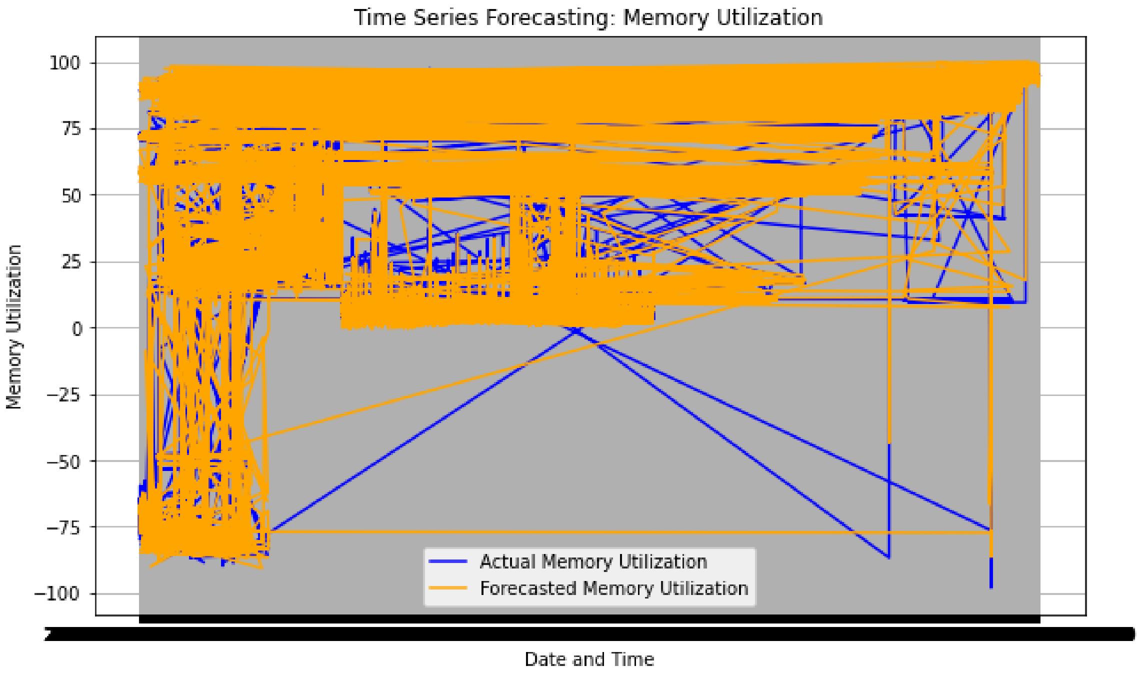
c) Time Series Forecasting - Triple Exponential Smoothing

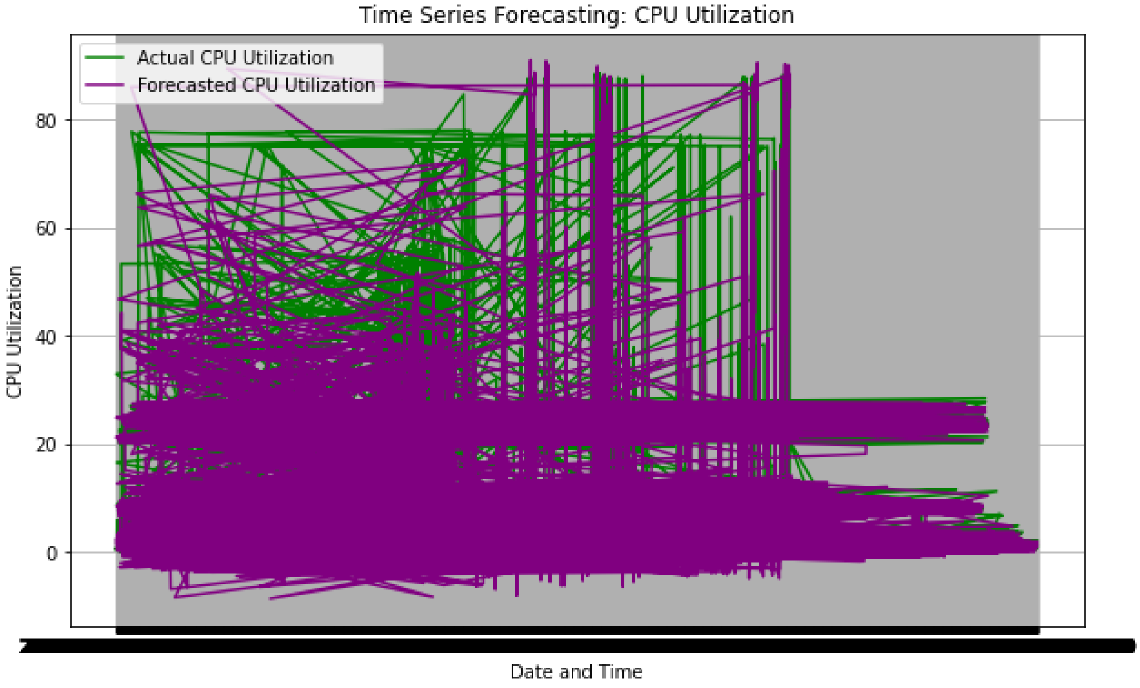
In [10]:

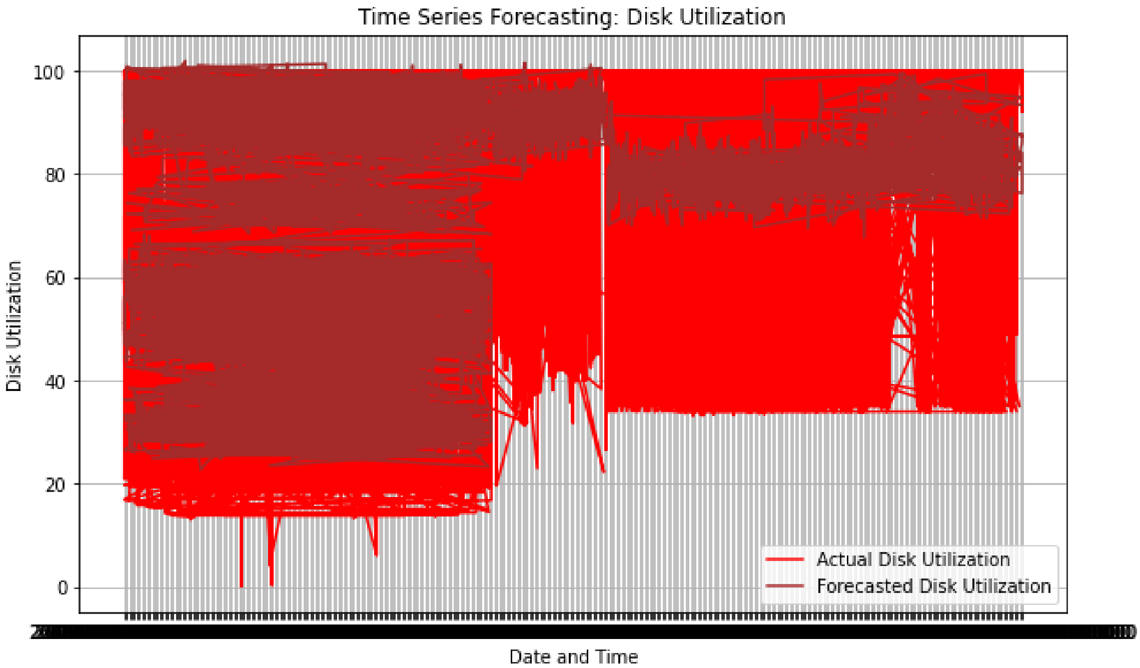
```
import pandas as pd
import matplotlib.pyplot as plt
import statsmodels.api as sm
from sklearn.metrics import mean_absolute_error, r2_score

# Drop rows with missing values
data.dropna(inplace=True)

# Function to perform Triple Exponential Smoothing
def triple_exponential_smoothing(data, column):
    model = sm.tsa.ExponentialSmoothing(data[column], trend='add', seasonal='add', seas
onal_periods=24)
    fitted_model = model.fit()
    forecast = fitted_model.fittedvalues
    return forecast

# Time Series Forecasting - Memory Utilization
data['Forecast_Memory'] = triple_exponential_smoothing(data, 'Memory Utilization')

# Time Series Forecasting - CPU Utilization
data['Forecast_CPU'] = triple_exponential_smoothing(data, 'CPU Utilization')

# Time Series Forecasting - Disk Utilization
data['Forecast_Disk'] = triple_exponential_smoothing(data, 'Disk Utilization')

# Calculate MAE and R2 for Memory Utilization forecast
mae_memory = mean_absolute_error(data['Memory Utilization'], data['Forecast_Memory'])
r2_memory = r2_score(data['Memory Utilization'], data['Forecast_Memory'])
print("Memory Utilization - Mean Absolute Error (MAE):", mae_memory)
print("Memory Utilization - R-squared (R2):", r2_memory)

# Calculate MAE and R2 for CPU Utilization forecast
mae_cpu = mean_absolute_error(data['CPU Utilization'], data['Forecast_CPU'])
r2_cpu = r2_score(data['CPU Utilization'], data['Forecast_CPU'])
print("CPU Utilization - Mean Absolute Error (MAE):", mae_cpu)
print("CPU Utilization - R-squared (R2):", r2_cpu)

# Calculate MAE and R2 for Disk Utilization forecast
mae_disk = mean_absolute_error(data['Disk Utilization'], data['Forecast_Disk'])
r2_disk = r2_score(data['Disk Utilization'], data['Forecast_Disk'])
print("Disk Utilization - Mean Absolute Error (MAE):", mae_disk)
print("Disk Utilization - R-squared (R2):", r2_disk)

# Time Series Visualization - Memory Utilization
plt.figure(figsize=(10, 6))
plt.plot(data['Date_Mem Utilization'], data['Memory Utilization'], label='Actual Memory
Utilization', color='blue')
plt.plot(data['Date_Mem Utilization'], data['Forecast_Memory'], label='Forecasted Memor
y Utilization', color='orange')
plt.xlabel('Date and Time')
plt.ylabel('Memory Utilization')
plt.title('Time Series Forecasting: Memory Utilization')
plt.legend()
plt.grid(True)
plt.show()

# Time Series Visualization - CPU Utilization
plt.figure(figsize=(10, 6))
plt.plot(data['Date_CPU Utilization'], data['CPU Utilization'], label='Actual CPU Utili
```

```
zation', color='green')
plt.plot(data['Date_CPU Utilization'], data['Forecast_CPU'], label='Forecasted CPU Util
ization', color='purple')
plt.xlabel('Date and Time')
plt.ylabel('CPU Utilization')
plt.title('Time Series Forecasting: CPU Utilization')
plt.legend()
plt.grid(True)
plt.show()

# Time Series Visualization - Disk Utilization
plt.figure(figsize=(10, 6))
plt.plot(data['Date_Disk Utilization'], data['Disk Utilization'], label='Actual Disk Ut
ilization', color='red')
plt.plot(data['Date_Disk Utilization'], data['Forecast_Disk'], label='Forecasted Disk U
tilization', color='brown')
plt.xlabel('Date and Time')
plt.ylabel('Disk Utilization')
plt.title('Time Series Forecasting: Disk Utilization')
plt.legend()
plt.grid(True)
plt.show()
```

```
C:\Users\poorn\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.
py:578: ValueWarning: An unsupported index was provided and will be ignore
d when e.g. forecasting.
  warnings.warn('An unsupported index was provided and will be'
C:\Users\poorn\anaconda3\lib\site-packages\statsmodels\tsa\holtwinters\mod
el.py:427: FutureWarning: After 0.13 initialization must be handled at mod
el creation
  warnings.warn(
C:\Users\poorn\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.
py:578: ValueWarning: An unsupported index was provided and will be ignore
d when e.g. forecasting.
  warnings.warn('An unsupported index was provided and will be'
C:\Users\poorn\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.
py:578: ValueWarning: An unsupported index was provided and will be ignore
d when e.g. forecasting.
  warnings.warn('An unsupported index was provided and will be'

Memory Utilization - Mean Absolute Error (MAE): 4.529509797307567
Memory Utilization - R-squared (R2): 0.8879486781917751
CPU Utilization - Mean Absolute Error (MAE): 4.0746768462411165
CPU Utilization - R-squared (R2): 0.44229987977697327
Disk Utilization - Mean Absolute Error (MAE): 13.636924187578092
Disk Utilization - R-squared (R2): 0.5217871702306971
```




Time Series Forecasting: CPU Utilization
Actual CPU Utilization
Forecasted CPU Utilization
CPU Utilization
80
60
40
20
0
Date and Time


Time Series Forecasting: Disk Utilization
Disk Utilization
100
80
60
40
20
0
Actual Disk Utilization
Forecasted Disk Utilization
Date and Time

Interpretation of the metrics for each resource utilization:

Memory Utilization:

MAE: 4.53 On average, the forecasted memory utilization values differ from the actual memory utilization values by approximately 4.53 units. R-squared: 0.89 The model explains around *88.79%* of the variance in memory utilization, which indicates a good fit of the model to the data.

CPU Utilization:

MAE: 4.07 On average, the forecasted CPU utilization values differ from the actual CPU utilization values by approximately 4.07 units. R-squared: 0.44 The model explains around 44.23% of the variance in CPU utilization, which indicates a moderate fit of the model to the data. The model may benefit from improvement.

Disk Utilization:

MAE: 13.64 On average, the forecasted disk utilization values differ from the actual disk utilization values by approximately 13.64 units. R-squared: 0.52 The model explains around 52.18% of the variance in disk utilization, which indicates a moderate fit of the model to the data. Like CPU utilization, the model may benefit from improvement.

Conclusion:

The model seems to perform well for memory utilization with a high R-squared value, indicating a good fit to the data. However, there is room for improvement for CPU and disk utilization models, as indicated by lower R-squared values. Further model tuning or trying different models might help improve the accuracy of the forecasts for CPU and disk utilization.