```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import mean_squared_error, mean_absolute_error
from statsmodels.tsa.statespace.sarimax import SARIMAX
```

✓ 0.0s

```python
data = pd.read_csv("D:\Study\Chuoi thoi gian\TH3\Gia SMP va SMPcap 2021.csv", encoding='latin-1', delimiter=';')
data2 = np.loadtxt("D:\Study\Chuoi thoi gian\TH3\Gia SMP va SMPcap 2021.csv",encoding='latin-1', delimiter=';', skiprows=1, usecols=(3, 4, 5), dtype=float)

df = data.iloc[:, [4, 5, 6]]
df.columns = ['Column5', 'Column6', 'Column7']
```

✓ 0.0s

```python
print(df.describe())
```

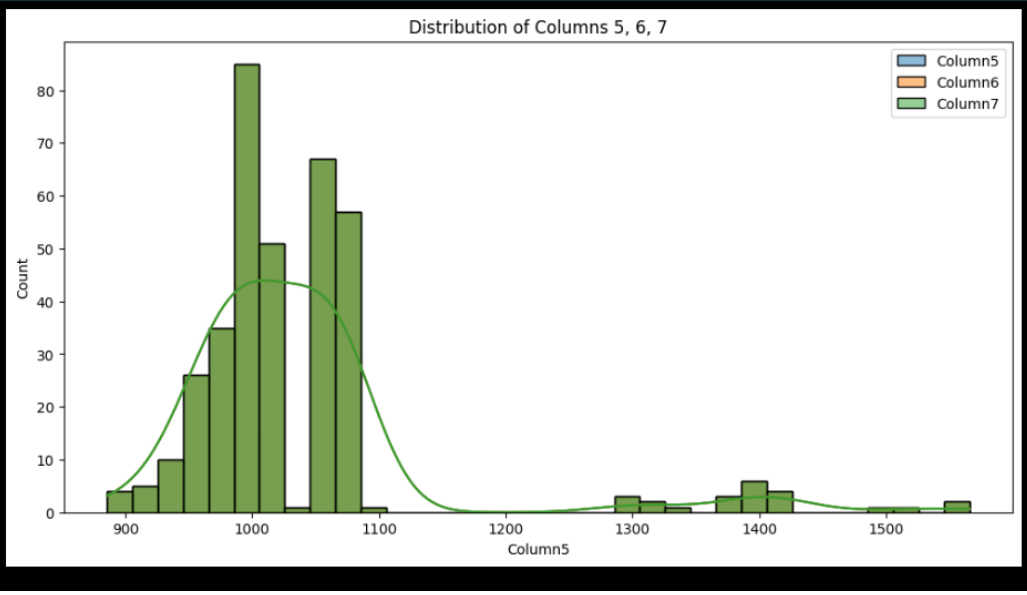
✓ 0.0s

```
          Column5      Column6      Column7
count   365.000000   365.000000   365.000000
mean   1040.228219  1040.228767  1040.228767
std     105.147113   105.146720   105.146665
min     885.700000   885.700000   885.700000
25%     988.400000   988.400000   988.400000
50%    1022.600000  1022.600000  1022.600000
75%    1061.500000  1061.500000  1061.500000
max    1565.500000  1565.500000  1565.500000
```

EDA

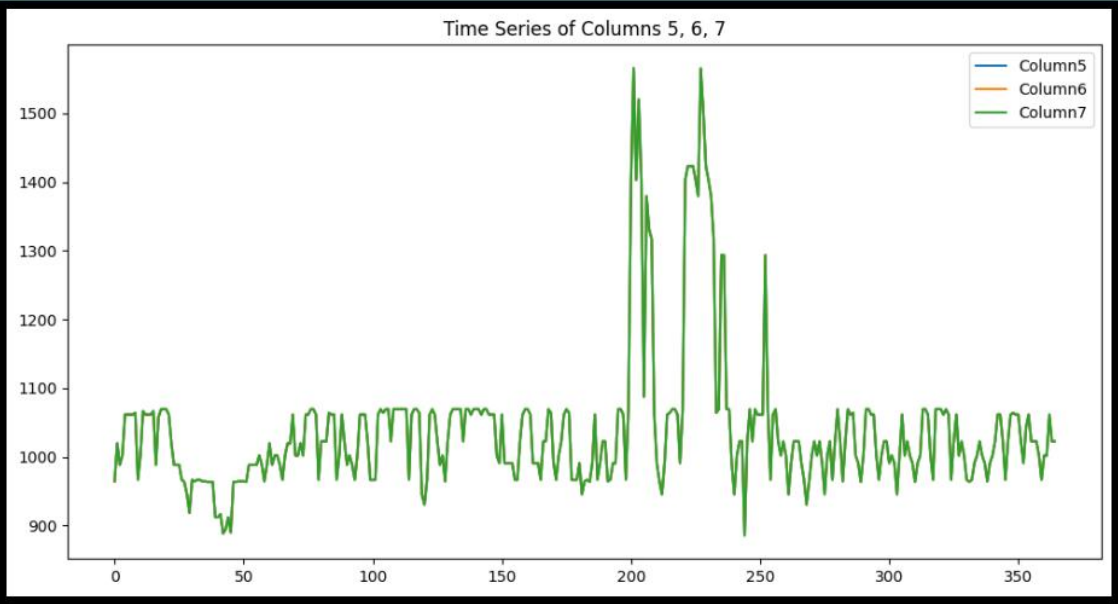
```python
# biểu đồ phân phối
plt.figure(figsize=(12, 6))
sns.histplot(df['Column5'], kde=True, label='Column5')
sns.histplot(df['Column6'], kde=True, label='Column6')
sns.histplot(df['Column7'], kde=True, label='Column7')
plt.legend()
plt.title('Distribution of Columns 5, 6, 7')
plt.show()
```

✓ 0.3s



```python
# biểu đồ thời gian
plt.figure(figsize=(12, 6))
plt.plot(df['Column5'], label='Column5')
plt.plot(df['Column6'], label='Column6')
plt.plot(df['Column7'], label='Column7')
plt.legend()
plt.title('Time Series of Columns 5, 6, 7')
plt.show()
```

✓ 0.1s



```
filtered_data2 = []

for measurement in data2:
    z = measurement.reshape(3, 1)  # Chuyển đổi đo lường thành vector cột

    # Dự đoán bước tiếp theo
    x, P = predict(x, P, F, Q)

    # Cập nhật với đo lường mới
    x, P = update(x, P, z, H, R)

    # Lưu trữ kết quả đã lọc
    filtered_data2.append(x.flatten())

filtered_data2 = np.array(filtered_data2)
```

✓ 0.0s

```
print(filtered_data2)
```

✓ 0.0s

```
[[ 877.51711712  877.51711712  877.51711712]
 [ 948.95907665  948.95907665  948.95907665]
 [ 963.78734581  963.78734581  963.78734581]
 ...
 [1020.03040877 1020.03040875 1020.03041047]
 [1020.7245998  1020.72459979 1020.72460105]
 [1021.23125081 1021.2312508  1021.23125172]]
```

```
mse_kalman = mean_squared_error(data2, filtered_data2)
mae_kalman = mean_absolute_error(data2, filtered_data2)
rmse_kalman = np.sqrt(mse_kalman)

print("Kalman Filter - MSE:", mse_kalman)
print("Kalman Filter - MAE:", mae_kalman)
print("Kalman Filter - RMSE:", rmse_kalman)
```

✓ 0.0s

```
Kalman Filter - MSE: 2793.952548984233
Kalman Filter - MAE: 32.38961736402761
Kalman Filter - RMSE: 52.85785229258027
```

SARIMA

```
series = data.iloc[:, 5]

# Điền giá trị thiếu bằng phương pháp nội suy (interpolation)
series = series.interpolate(method='linear')

# Định nghĩa các tham số của mô hình SARIMA
order = (1, 1, 1)         # Tham số (p, d, q) cho phần không mùa
seasonal_order = (1, 1, 1, 12)  # Tham số (P, D, Q, s) cho phần mùa (s = 12 cho dữ liệu theo tháng)

# Xây dựng và huấn luyện mô hình SARIMA
model = SARIMAX(series, order=order, seasonal_order=seasonal_order)
model_fit = model.fit(disp=False)

# In kết quả huấn luyện
print(model_fit.summary())
```

✓ 0.6s

```
                                SARIMAX Results
==========================================================================================
Dep. Variable:                                5   No. Observations:                  365
Model:             SARIMAX(1, 1, 1)x(1, 1, 1, 12)   Log Likelihood               -1966.078
Date:                            Tue, 21 May 2024   AIC                           3942.155
Time:                                    10:51:58   BIC                           3961.473
Sample:                                         0   HQIC                          3949.843
                                              - 365
Covariance Type:                              opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.9983      6.211      0.161      0.872     -11.175      13.172
ma.L1         -0.9992     12.971     -0.077      0.939     -26.422      24.424
ar.S.L12      -0.2393      0.038     -6.286      0.000      -0.314      -0.165
ma.S.L12      -0.9999     19.042     -0.053      0.958     -38.321      36.321
sigma2      3644.1726   8.73e+04      0.042      0.967   -1.67e+05    1.75e+05
===================================================================================
Ljung-Box (L1) (Q):                   1.25   Jarque-Bera (JB):               989.31
Prob(Q):                              0.26   Prob(JB):                         0.00
Heteroskedasticity (H):               1.61   Skew:                             0.48
Prob(H) (two-sided):                  0.01   Kurtosis:                        11.16
===================================================================================

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).
```

```python
# Dự đoán giá trị
n_forecast = len(series)
forecast = model_fit.predict(start=0, end=n_forecast-1)

# Tính toán độ đo lỗi
mse = mean_squared_error(series, forecast)
mae = mean_absolute_error(series, forecast)
rmse = np.sqrt(mse)

print("SARIMA - MSE:", mse)
print("SARIMA - MAE:", mae)
print("SARIMA - RMSE:", rmse)

# Vẽ biểu đồ kết quả dự đoán
plt.figure(figsize=(10, 6))
plt.plot(series, label='Actual')
plt.plot(forecast, label='Forecast', color='orange')
plt.title('SARIMA Forecast for Column 6')
plt.legend()
plt.show()
```

[28] ✓ 0.1s

```
SARIMA - MSE: 7106.29856380865
SARIMA - MAE: 43.264926294361445
SARIMA - RMSE: 84.29886454637838
```