In [3]:
```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [4]:
```python
delhi_data=pd.read_csv(r"C:\Users\Chandrika\Desktop\shadowfox datascience\(task2)delhiaqi.csv")
```

In [5]:
```python
delhi_data.head()
```

Out[5]:

	date	co	no	no2	o3	so2	pm2_5	pm10	nh3
0	2023-01-01 00:00:00	1655.58	1.66	39.41	5.90	17.88	169.29	194.64	5.83
1	2023-01-01 01:00:00	1869.20	6.82	42.16	1.99	22.17	182.84	211.08	7.66
2	2023-01-01 02:00:00	2510.07	27.72	43.87	0.02	30.04	220.25	260.68	11.40
3	2023-01-01 03:00:00	3150.94	55.43	44.55	0.85	35.76	252.90	304.12	13.55
4	2023-01-01 04:00:00	3471.37	68.84	45.24	5.45	39.10	266.36	322.80	14.19

In [6]:
```python
delhi_data.shape
```

Out[6]:
```
(561, 9)
```

In [7]:
```python
delhi_data.isnull().sum()
```

Out[7]:
```
date     0
co       0
no       0
no2      0
o3       0
so2      0
pm2_5    0
pm10     0
nh3      0
dtype: int64
```

In [8]:
```python
delhi_data.describe()
```

Out[8]:

	co	no	no2	o3	so2	pm2_5	pm10	nh3
count	561.000000	561.000000	561.000000	561.000000	561.000000	561.000000	561.000000	561.000000
mean	3814.942210	51.181979	75.292496	30.141943	64.655936	358.256364	420.988414	26.425062
std	3227.744681	83.904476	42.473791	39.979405	61.073080	227.359117	271.287026	36.563094
min	654.220000	0.000000	13.370000	0.000000	5.250000	60.100000	69.080000	0.630000
25%	1708.980000	3.380000	44.550000	0.070000	28.130000	204.450000	240.900000	8.230000
50%	2590.180000	13.300000	63.750000	11.800000	47.210000	301.170000	340.900000	14.820000
75%	4432.680000	59.010000	97.330000	47.210000	77.250000	416.650000	482.570000	26.350000
max	16876.220000	425.580000	263.210000	164.510000	511.170000	1310.200000	1499.270000	267.510000

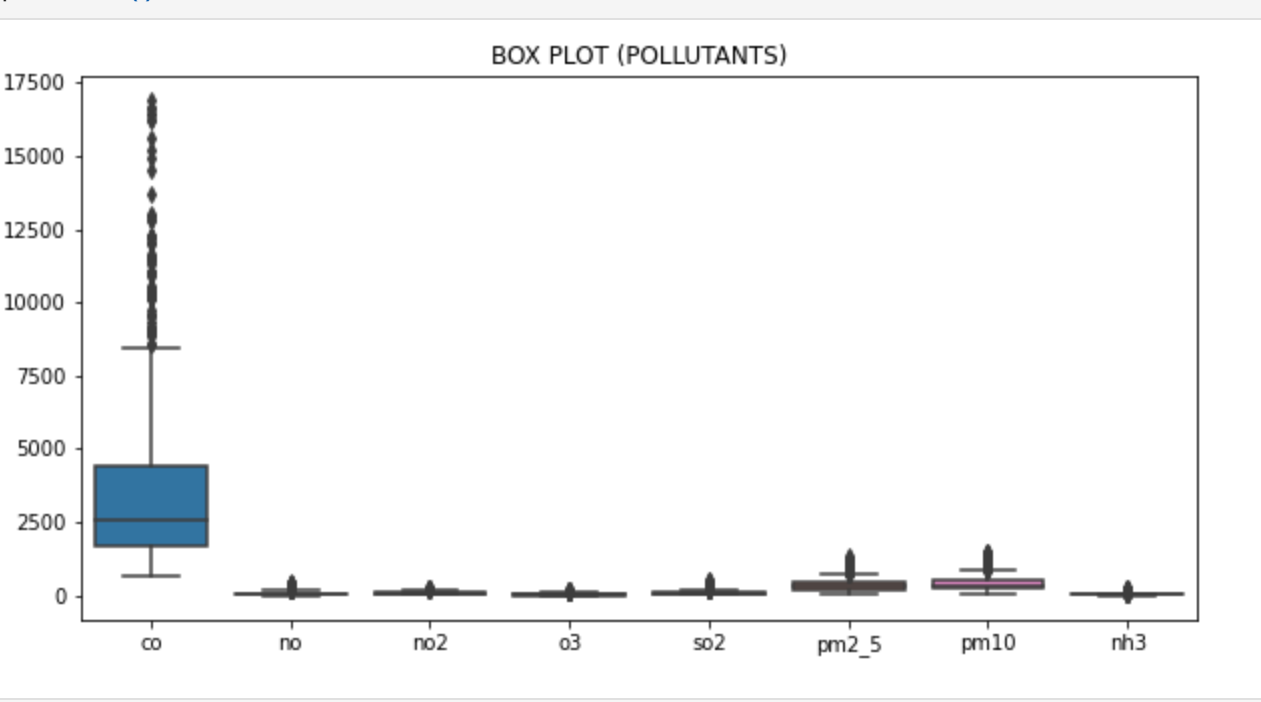
In [9]:
```python
'''it performs two operations
1)This line converts the 'date' column in the DataFrame to a datetime object , and
2)it sets the 'date' column as the index of the DataFrame'''


delhi_data['date']=pd.to_datetime(delhi_data['date'])
delhi_data.set_index('date',inplace=True)
```

In [10]:
```python
delhi_data.info()
```

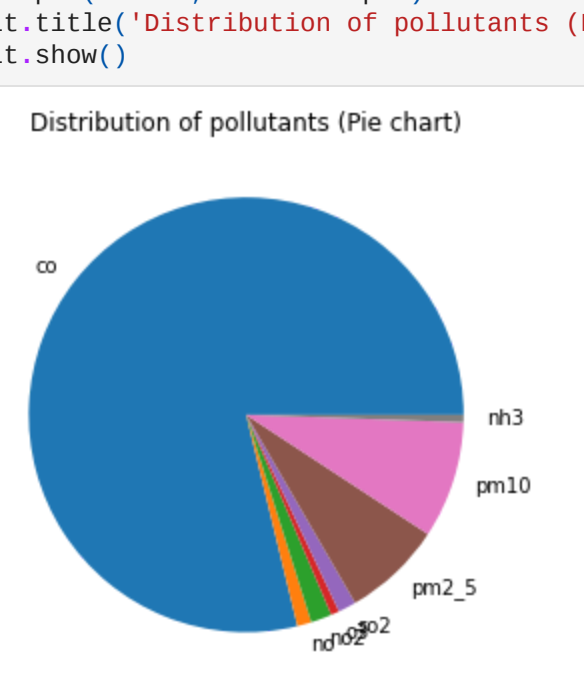
```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 561 entries, 2023-01-01 00:00:00 to 2023-01-24 08:00:00
Data columns (total 8 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   co      561 non-null    float64
 1   no      561 non-null    float64
 2   no2     561 non-null    float64
 3   o3      561 non-null    float64
 4   so2     561 non-null    float64
 5   pm2_5   561 non-null    float64
 6   pm10    561 non-null    float64
 7   nh3     561 non-null    float64
dtypes: float64(8)
memory usage: 39.4 KB
```

In [11]:
```python
cor_mat=delhi_data.corr()
```

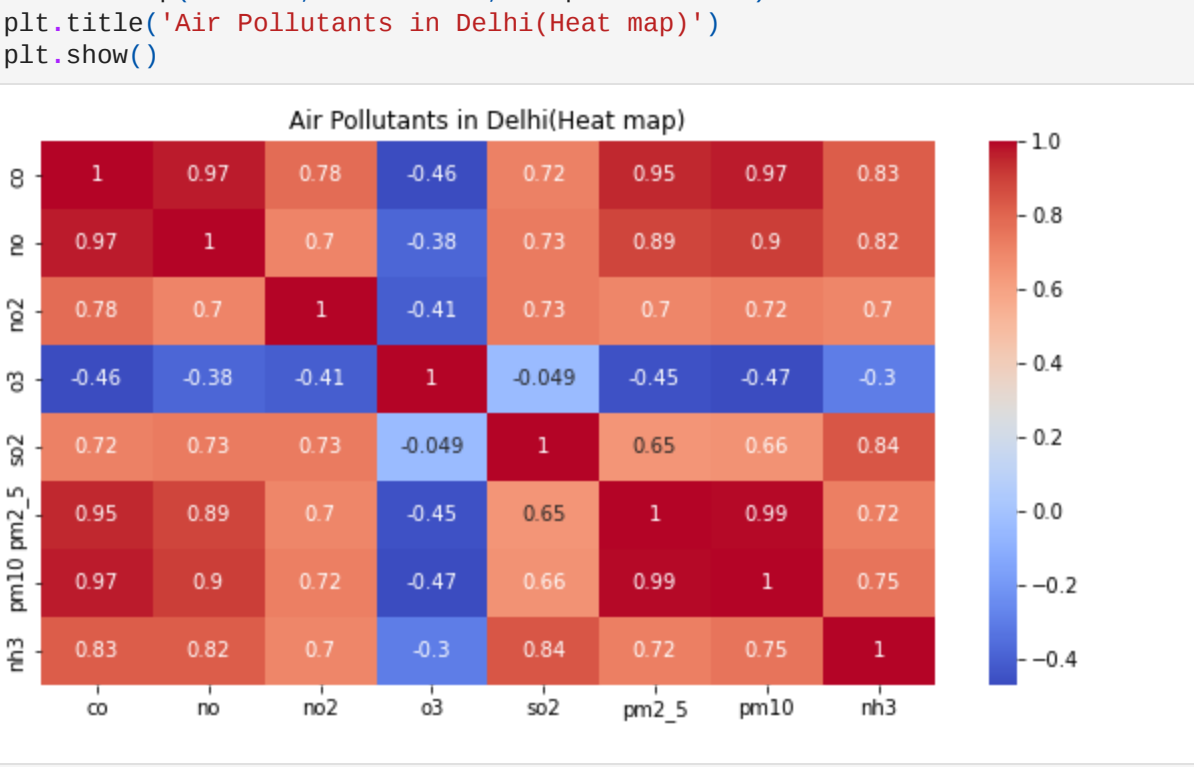
In [12]:
```python
plt.figure(figsize=(10, 5))
sns.boxplot(data=delhi_data, orient='v')
plt.title('BOX PLOT (POLLUTANTS)')
plt.show()
```



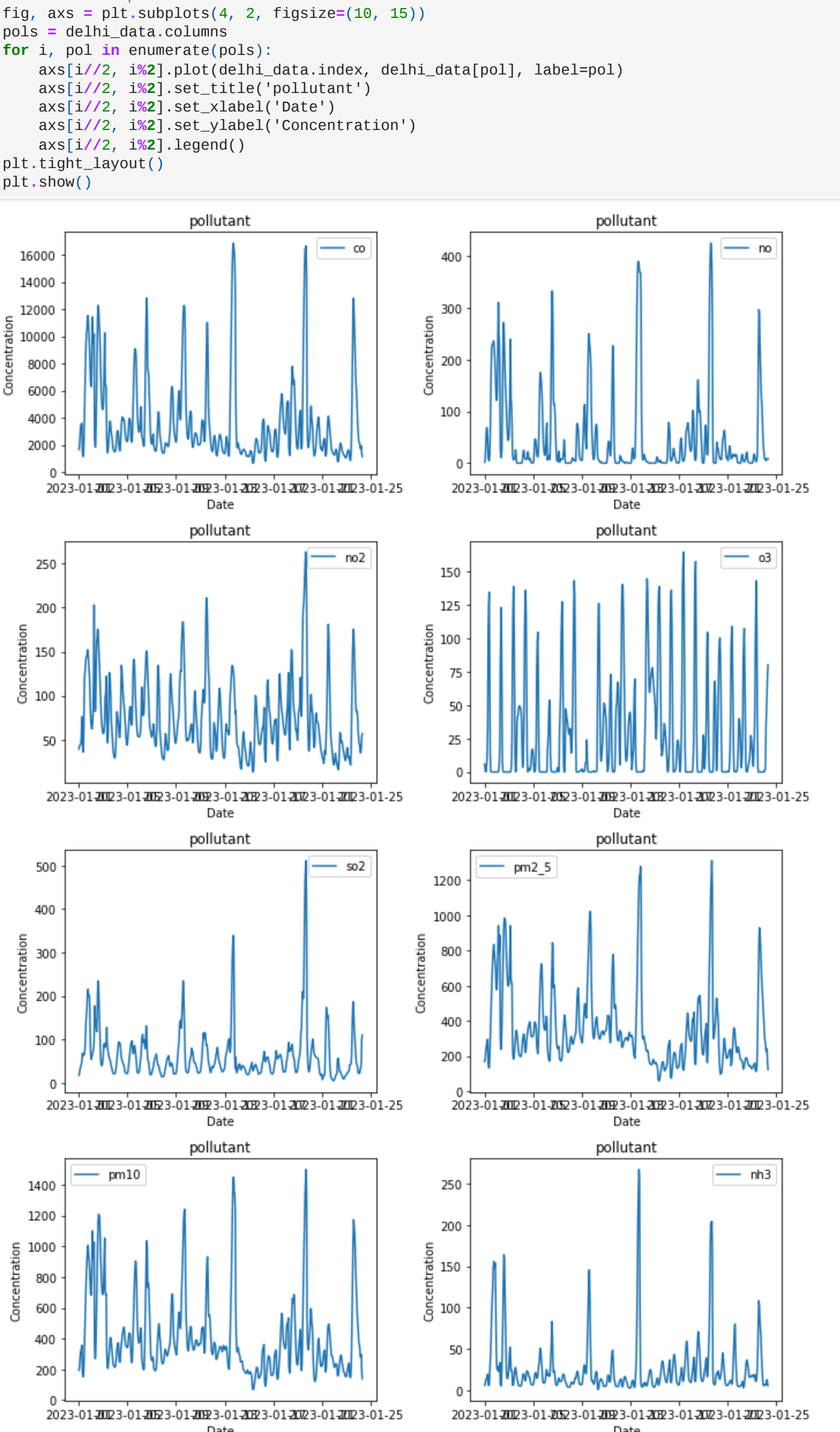
In [14]:
```python
values = delhi_data[['co','no','no2','o3','so2','pm2_5','pm10','nh3']].sum(axis=0) #columns in dataset
pol = list(['co','no','no2','o3','so2','pm2_5','pm10','nh3'])
plt.figure(figsize =(10,5))
plt.pie(values, labels = pol)
plt.title('Distribution of pollutants (Pie chart)')
plt.show()
```



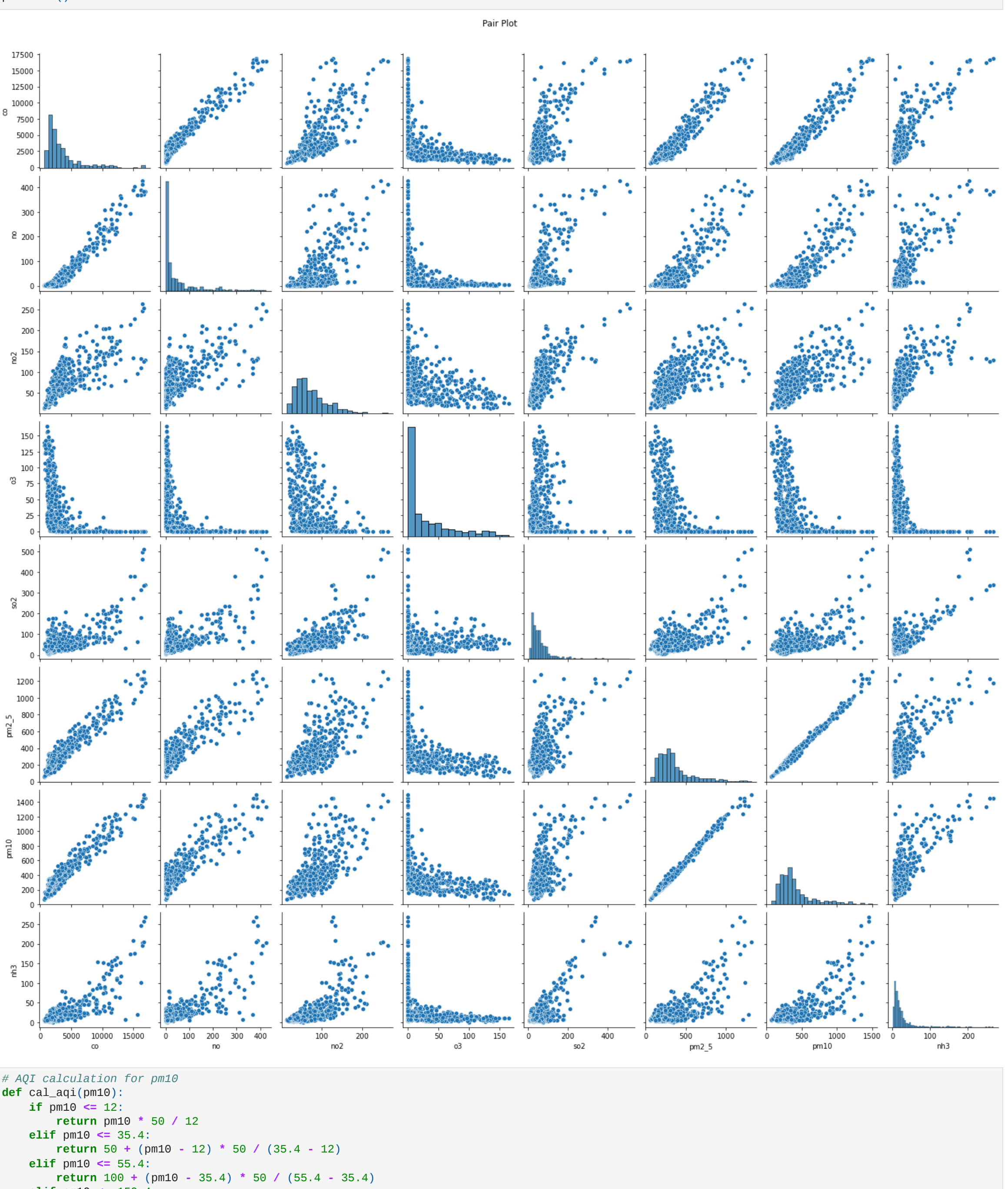
In [15]:
```python
#correlation matrix
plt.figure(figsize=(10, 5))
sns.heatmap(cor_mat, annot=True, cmap='coolwarm')
plt.title('Air Pollutants in Delhi(Heat map)')
plt.show()
```



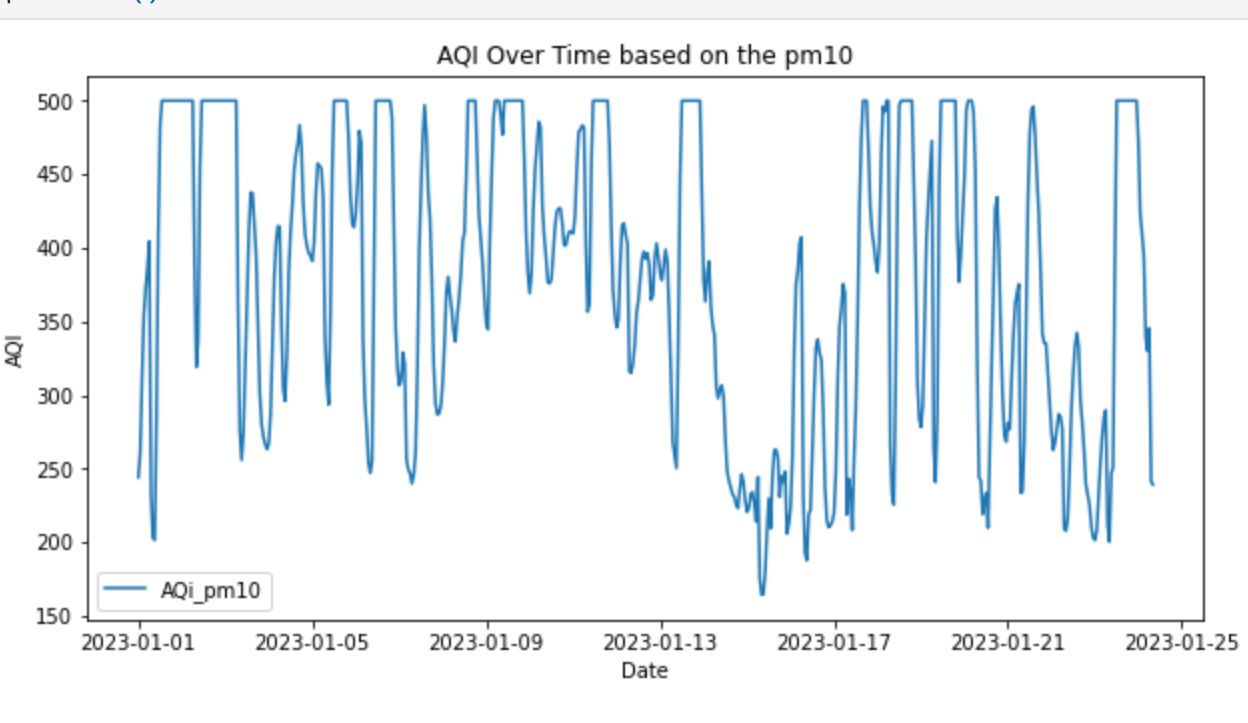
In [17]:
```python
# Time series plots
fig, axs = plt.subplots(4, 2, figsize=(10, 15))
pols = delhi_data.columns
for i, pol in enumerate(pols):
    axs[i//2, i%2].plot(delhi_data.index, delhi_data[pol], label=pol)
    axs[i//2, i%2].set_title('pollutant')
    axs[i//2, i%2].set_xlabel('Date')
    axs[i//2, i%2].set_ylabel('Concentration')
    axs[i//2, i%2].legend()
plt.tight_layout()
plt.show()
```



In [18]:
```python
sns.pairplot(delhi_data)
plt.suptitle('Pair Plot', y=1.02)
plt.show()
```



In [19]:
```python
# AQI calculation for pm10
def cal_aqi(pm10):
    if pm10 <= 12:
        return pm10 * 50 / 12
    elif pm10 <= 35.4:
        return 50 + (pm10 - 12) * 50 / (35.4 - 12)
    elif pm10 <= 55.4:
        return 100 + (pm10 - 35.4) * 50 / (55.4 - 35.4)
    elif pm10 <= 150.4:
        return 150 + (pm10 - 55.4) * 100 / (150.4 - 55.4)
    elif pm10 <= 250.4:
        return 200 + (pm10 - 150.4) * 100 / (250.4 - 150.4)
    elif pm10 <= 350.4:
        return 300 + (pm10 - 250.4) * 100 / (350.4 - 250.4)
    elif pm10 <= 500.4:
        return 400 + (pm10 - 350.4) * 100 / (500.4 - 350.4)
    else:
        return 500

delhi_data['AQi_pm10'] = delhi_data['pm10'].apply(cal_aqi)

plt.figure(figsize=(10, 5))
plt.plot(delhi_data.index, delhi_data['AQi_pm10'], label='AQi_pm10')
plt.title('AQI Over Time based on the pm10')
plt.xlabel('Date')
plt.ylabel('AQI')
plt.legend()
plt.show()
```

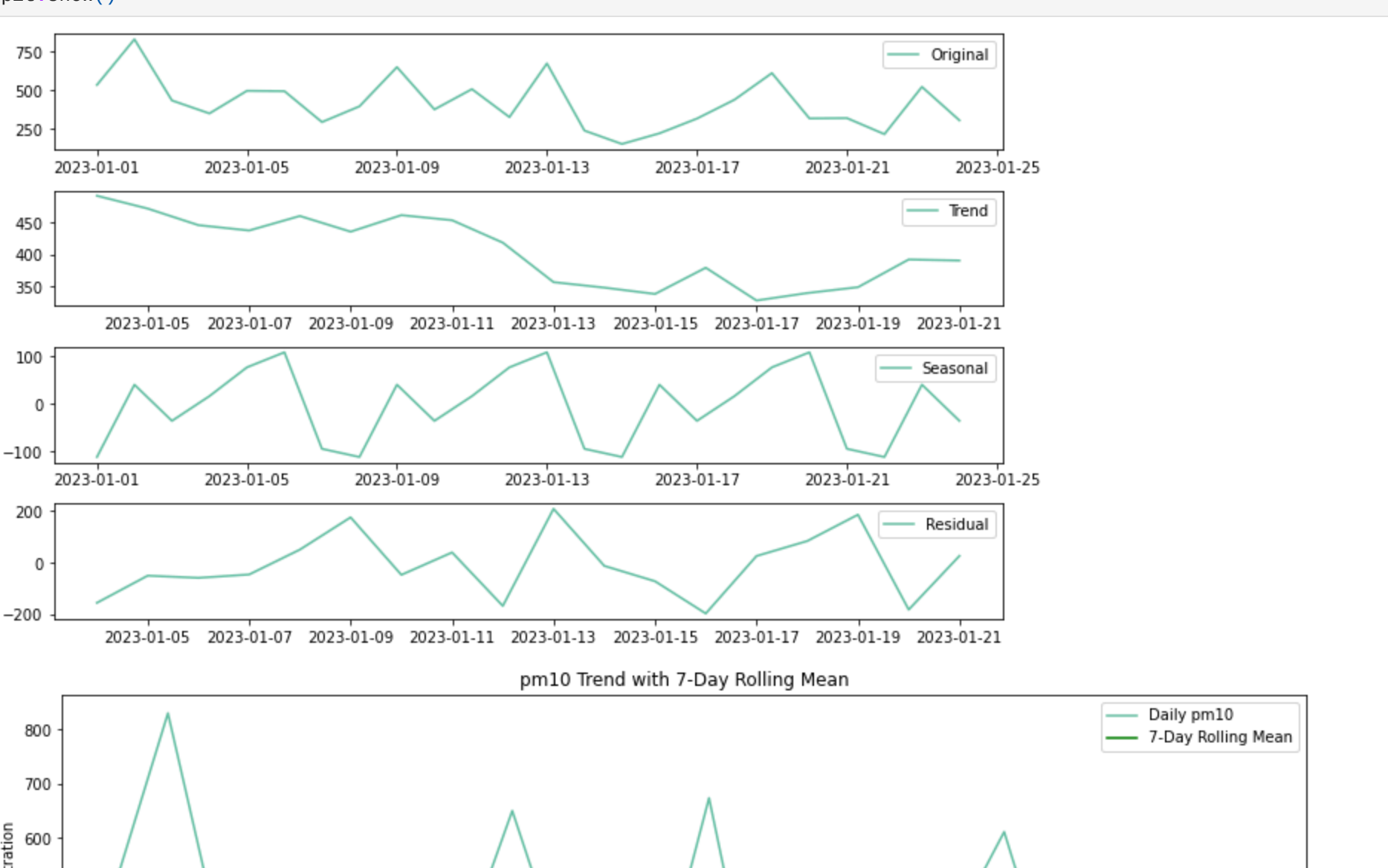


In [20]:
```python
# now we convert the date column to datetime
if 'date' in delhi_data.columns:
    delhi_data['date'] = pd.to_datetime(delhi['date'])
    delhi_data.set_index('date', inplace=True)
else:
    print("The 'date' column is not found in the dataset.")

delhi_data.head()

if 'date' in delhi_data.columns:
    features = delhi_data.drop(columns=['date'])
else:
    features = delhi_data

features.hist(bins=24, figsize=(14,8), edgecolor='black')
plt.suptitle('Histograms of Air Quality Parameters', y=1.02)
plt.tight_layout()

sns.set_palette('Set2')
plt.show()

daily_data = delhi_data.resample('D').mean()
print(daily_data.shape)
```

```
The 'date' column is not found in the dataset.
```


```
(24, 9)
```

In [27]:
```python
from statsmodels.tsa.seasonal import seasonal_decompose
decomp = seasonal_decompose(daily_data['pm10'], model='additive', period=7)
trend = decomp.trend
seasonal = decomp.seasonal
residual = decomp.resid

# Plot the decomposed components
plt.figure(figsize=(10, 6))
plt.subplot(411)
plt.plot(daily_data['pm10'], label='Original')
plt.legend(loc='best')
plt.subplot(412)
plt.plot(trend, label='Trend')
plt.legend(loc='best')
plt.subplot(413)
plt.plot(seasonal, label='Seasonal')
plt.legend(loc='best')
plt.subplot(414)
plt.plot(residual, label='Residual')
plt.legend(loc='best')
plt.tight_layout()
plt.show()

# Apply a 7-day rolling mean to smooth out the data (adjust as needed)
rolling_mean = daily_data['pm10'].rolling(window=7).mean()

# Plot the original data and the rolling mean
plt.figure(figsize=(15, 5))
plt.plot(daily_data['pm10'], label='Daily pm10')
plt.plot(rolling_mean, label='7-Day Rolling Mean', color='green')
plt.title('pm10 Trend with 7-Day Rolling Mean')
plt.xlabel('Date')
plt.ylabel('pm10 Concentration')
plt.legend()
plt.show()
```



In [28]:
```python
def stl_decomp(delhi_data):
    decomp = seasonal_decompose(delhi_data, model='additive', period=7)
    trend = decomp.trend
    seasonal = decomp.seasonal
    residual = decomp.resid

    plt.figure(figsize=(12, 8))
    plt.subplot(411)
    plt.plot(delhi_data, label='Original')
    plt.legend(loc='best')
    plt.subplot(412)
    plt.plot(trend, label='Trend')
    plt.legend(loc='best')
    plt.subplot(413)
    plt.plot(seasonal, label='Seasonal')
    plt.legend(loc='best')
    plt.subplot(414)
    plt.plot(residual, label='Residual')
    plt.legend(loc='best')
    plt.tight_layout()
    plt.show()
```

In [34]:
```python
def apply_mean(delhi_data, window):
    rolling_mean = delhi_data.rolling(window=window).mean()
    return rolling_mean

def cal_aqi(row):
    return cal_aqi

def aqi_analysis(delhi_data):
    return aqi_analysis

def day_nig_aqi_compar(delhi_data):
    return day_nig_aqi_compar

def check_convert_date_col(delhi_data):
    if 'date' in delhi_data.columns:
        delhi_data['date'] = pd.to_datetime(delhi_data['date'])
        delhi_data.set_index('date', inplace=True)
    else:
        print("The 'date' column is not found")

def extract_hour_date(delhi_data):
    if 'date' in delhi_data.columns:
        delhi_data['hour'] = delhi_data['date'].dt.hour
    else:
        print("The 'date' column is not found")

def check_convert_date_col(delhi_data):
    return check_convert_date_col

def extract_hr_date(delhi_data):
    return extract_hr_date
def cal_aqi_and_plot(delhi_data):
      return cal_aqi_and_plot

def cal_avg_aqi_day_nig(delhi_data):
     return cal_avg_aqi_day_nig

def day_vs_night_aqi_compar(delhi_data):
     return day_vs_night_aqi_compar
```

In [35]:
```python
stl_decomp(delhi_data['pm10'])

delhi_data['rolling_mean_pm10'] = apply_mean(delhi_data['pm10'], window=7)

delhi_data['AQI'] = delhi_data.apply(cal_aqi, axis=1)

aqi_analysis(delhi_data)
day_nig_aqi_compar(delhi_data)
check_convert_date_col(delhi_data)
extract_hr_date(delhi_data)
cal_aqi_and_plot(delhi_data)
cal_avg_aqi_day_nig(delhi_data)
day_vs_night_aqi_compar(delhi_data)
```

Out[35]:
```
<function __main__.day_vs_night_aqi_compar(delhi_data)>
```

Conclusion

The analysis of Delhi's air quality highlights several important points and suggests strategies to improve the situation

1. **Air Quality Levels**:
   - Delhi mostly experiences "Very Poor" and "Severe" air quality, which happens 97.33% of the time.
   - Only 2.67% of the time does the city have "Good" or "Satisfactory" air quality.
2. **Hourly Trends**:
   - AQI levels are slightly higher at night compared to the daytime.
   - The daily AQI extremes range from a high of 421.63 to a low of 330.37.
3. **Pollutant Correlations**:
   - Some pollutants show strong correlations, indicating they might come from the same sources or spread in similar ways. This suggests that addressing one pollutant may help reduce others as well.

Strategies to Improve Air Quality

1. **Reduce Carbon Monoxide (CO) Levels**:
   - Enforce strict vehicle emission standards.
   - Promote cleaner fuels like CNG and electric vehicles.
   - Improve traffic management to reduce congestion and idling.
   - Implement strict emissions regulations for industries.
   - Integrate urban planning that prioritizes mixed land-use and green spaces.
2. **Overall Air Quality Improvement**:
   - Promote renewable energy sources.
   - Adopt green building standards.
   - Enhance waste management practices.
   - Expand afforestation and urban greening initiatives.
   - Foster public awareness and education on air quality.
   - Encourage collaboration across different sectors to develop and implement comprehensive air quality management plans.

By taking these steps together, Delhi can significantly improve its air quality, protect public health, and create a more sustainable environment for everyone.