Objective

1. To calculate the Air Quality Index(AQI).
2. To find change in quality of air throughout the day.
3. To find the occurance of different air quality conditions.
4. Analyze the relation between specific pollutants
5. To find the distribution of pollutants.

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Importing Dataset

In [2]:
```
data = pd.read_csv('delhiaqi.csv')
data
```

Out[2]:

	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
...	...	...	...	...	...	...	...	...	...
556	2023-01-24 04:00:00	1762.39	4.64	37.01	33.26	30.52	231.15	289.84	6.27
557	2023-01-24 05:00:00	1735.69	6.82	34.96	46.49	34.33	225.08	280.52	9.12
558	2023-01-24 06:00:00	1922.61	8.16	40.10	56.51	43.39	242.49	296.07	12.54
559	2023-01-24 07:00:00	1361.85	9.05	52.78	71.53	100.14	165.67	191.82	7.47
560	2023-01-24 08:00:00	1134.87	8.61	56.89	80.11	110.63	123.76	140.26	5.51

561 rows × 9 columns

Air Quality Index (AQI)

In [3]:

```
co_breakpoints = [(0,1,0,60),(1.02,2,53,102),(2.04,12,104,203),
 (11.23,19,205,302),(19.32,315,303,400),(316,999997,401,500)]

no2_breakpoints = [(0,38,0,54),(43,84,50,103),(83,183,103,203),
 (182,284,205,302),(284,402,303,400),(403,999999,401,500)]

no_breakpoints = no2_breakpoints

o3_breakpoints = [(0,54,0,1),(55,79,1,2),(80,168,102,300),(123,202,198,3
00),
 (169,208,301,400),(209,999999,401,500)]

so2_breakpoints = [(0,40,0,50),(41,80,51,100),(81,380,101,200),
 (381,800,201,300),(801,1600,301,400),(1600,999999,401,500)]

pm25_breakpoints = [(0,12,0,50),(31,55,51,100),(61,150,101,200),
 (151,250,201,300),(251,350,301,400),(351,999999,401,500)]

pm10_breakpoints = pm25_breakpoints

nh3_breakpoints = [(0,200,0,50),(201,400,51,100),(401,800,101,200),
 (801,1200,201,300),(1201,1800,301,400),(1801,999999,401,500)]
```

In [4]:

```
def calculate_sub_index(value,breakpoints):
  for (low_val, high_val, low_index, high_index) in breakpoints:
    if low_val <= value <= high_val:
      return low_index + ((high_index - low_index)/(high_val - low_val))
* (value - low_val)
  return 500
```

In [5]:

```
ugm3_to_mgm3 = 1e-3
```

In [6]:

```
def calculate_aqi(row):
  row['co_sub_index'] = calculate_sub_index(row['co'] * ugm3_to_mgm3, co
_breakpoints)
  row['no2_sub_index'] = calculate_sub_index(row['no2'], no2_breakpoint
s)
  row['no_sub_index'] = calculate_sub_index(row['no'], no_breakpoints)
  row['o3_sub_index'] = calculate_sub_index(row['o3'], o3_breakpoints)
  row['so2_sub_index'] = calculate_sub_index(row['so2'], so2_breakpoint
s)
  row['pm2_5_sub_index'] = calculate_sub_index(row['pm2_5'], pm25_breakp
oints)
  row['pm10_sub_index'] = calculate_sub_index(row['pm10'], pm10_breakpoi
nts)
  row['nh3_sub_index'] = calculate_sub_index(row['nh3'], nh3_breakpoint
s)

  aqi = max(row['co_sub_index'], row['no2_sub_index'], row['no_sub_inde
x'],
            row['o3_sub_index'], row['so2_sub_index'], row['pm2_5_sub_in
dex'],
            row['pm10_sub_index'], row['nh3_sub_index'])
  return aqi
```

In [7]:

```
data['aqi'] = data.apply(calculate_aqi, axis=1)
data.head()
```

Out[7]:

	date	co	no	no2	o3	so2	pm2_5	pm10	nh3	aqi
0	2023-01-01 00:00:00	1655.58	1.66	39.41	5.90	17.88	169.29	194.64	5.83	500.00
1	2023-01-01 01:00:00	1869.20	6.82	42.16	1.99	22.17	182.84	211.08	7.66	500.00
2	2023-01-01 02:00:00	2510.07	27.72	43.87	0.02	30.04	220.25	260.68	11.40	310.68
3	2023-01-01 03:00:00	3150.94	55.43	44.55	0.85	35.76	252.90	304.12	13.55	354.12
4	2023-01-01 04:00:00	3471.37	68.84	45.24	5.45	39.10	266.36	322.80	14.19	372.80

In [8]:

```
aqi_categories = {
    (0, 50): 'Good',
    (51, 100): 'Satisfactory',
    (101, 200): 'Moderately Polluted',
    (201, 300): 'Poor',
    (301, 400): 'Very Poor',
    (401, 500): 'Severe'
}
```

In [9]:

```
def get_aqi_category(aqi):
  for (low_aqi, high_aqi), category in aqi_categories.items():
    if low_aqi <= aqi <= high_aqi:
      return category
  return 'Severe'
```

In [10]:

```
data['aqi_category'] = data['aqi'].apply(get_aqi_category)
aqi_category_distribution = data['aqi_category'].value_counts(normalize
= True) * 100
aqi_category_distribution
```

Out[10]:

	proportion
aqi_category	
Severe	55.258467
Very Poor	22.281640
Poor	20.499109
Moderately Polluted	1.960784

dtype: float64

In [11]:

```
data['date'] = pd.to_datetime(data['date'])
data['hour'] = data['date'].dt.hour
hourly_aqi = data.groupby('hour')['aqi'].mean()
```

In [14]:

```
data.head()
```

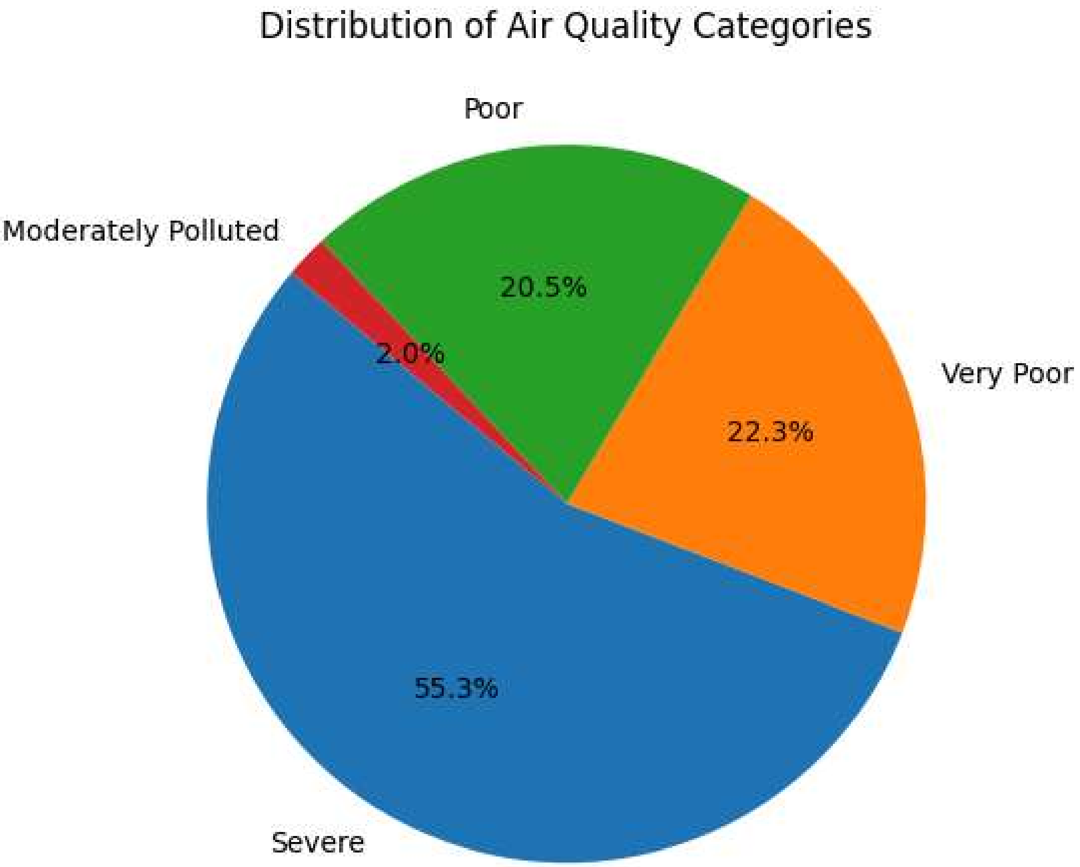
Out[14]:

	date	co	no	no2	o3	so2	pm2_5	pm10	nh3	aqi	aqi_category
0	2023-01-01 00:00:00	1655.58	1.66	39.41	5.90	17.88	169.29	194.64	5.83	500.00	Severe
1	2023-01-01 01:00:00	1869.20	6.82	42.16	1.99	22.17	182.84	211.08	7.66	500.00	Severe
2	2023-01-01 02:00:00	2510.07	27.72	43.87	0.02	30.04	220.25	260.68	11.40	310.68	Very Poor
3	2023-01-01 03:00:00	3150.94	55.43	44.55	0.85	35.76	252.90	304.12	13.55	354.12	Very Poor
4	2023-01-01 04:00:00	3471.37	68.84	45.24	5.45	39.10	266.36	322.80	14.19	372.80	Very Poor

In [15]:

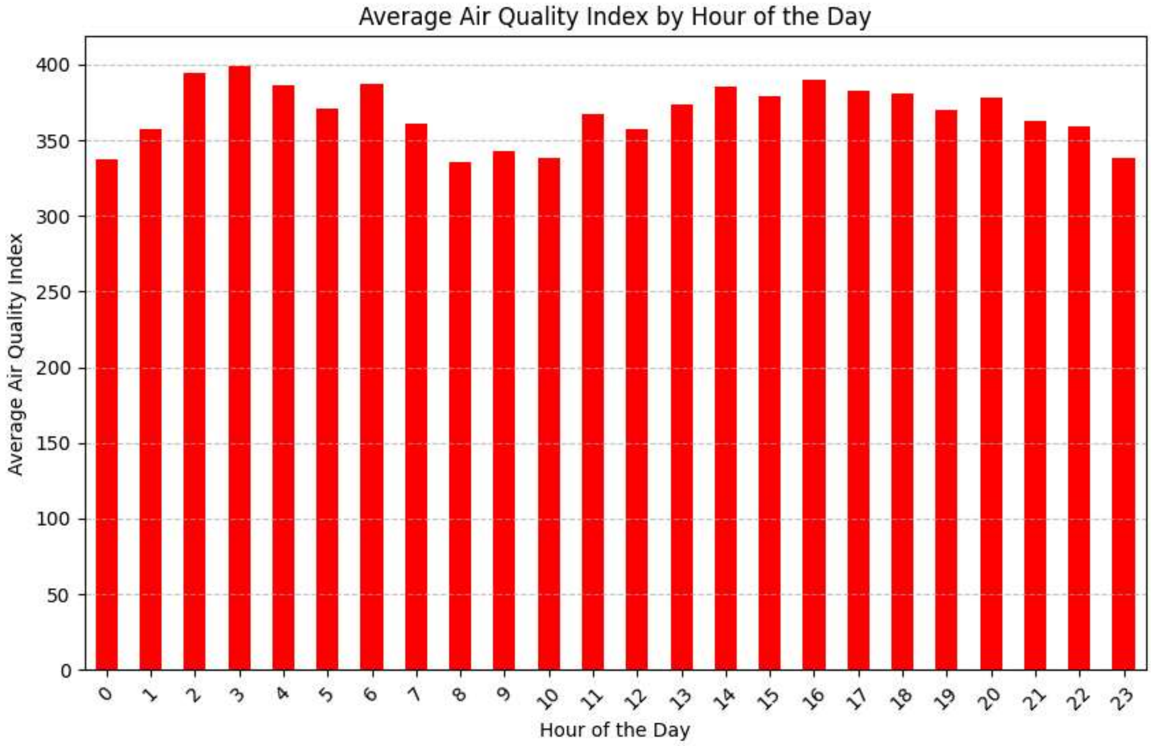
```
category_counts = data['aqi_category'].value_counts()

plt.figure(figsize=(10, 5))
plt.pie(category_counts, labels=category_counts.index, autopct='%1.1f%%', startangle=140)
plt.title('Distribution of Air Quality Categories')
plt.tight_layout()
sns.set_palette('pastel')
plt.show()
```



In [16]:

```
plt.figure(figsize=(10, 6))
hourly_aqi.plot(kind='bar', color='red')
plt.title('Average Air Quality Index by Hour of the Day')
plt.xlabel('Hour of the Day')
plt.ylabel('Average Air Quality Index')
plt.xticks(rotation=45)
plt.grid(axis='y', linestyle='--', alpha=0.7)
sns.set_palette('pastel')
plt.show()
```



In [17]:

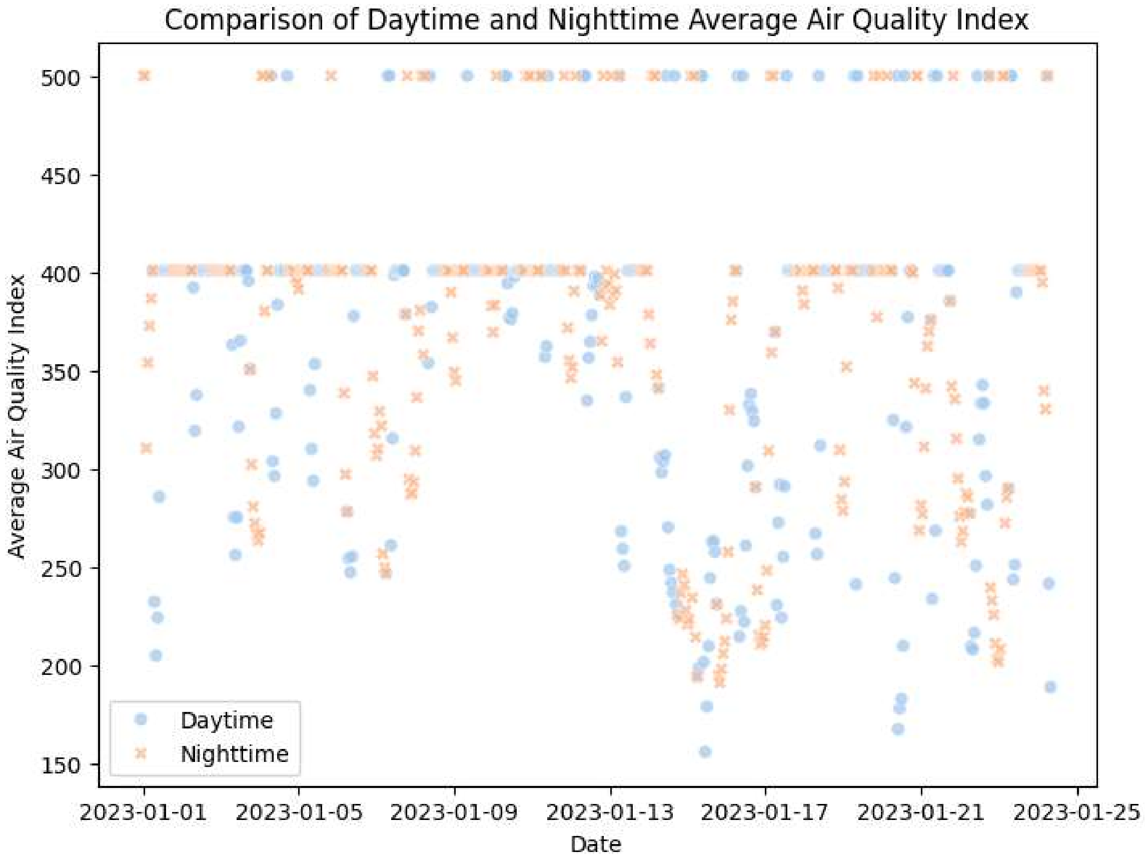
```
day_hours = (6,18)
night_hours = (18,6)

daytime_data = data[(data['hour'] >= day_hours[0]) & (data['hour'] <= da
y_hours[1])]
nighttime_data = data[(data['hour'] >= night_hours[0]) | (data['hour'] <
= night_hours[1])]

average_day_aqi = daytime_data.groupby('date')['aqi'].mean()
average_night_aqi = nighttime_data.groupby('date')['aqi'].mean()
day_night_aqi_comparison = pd.DataFrame({
    'Daytime': average_day_aqi,
    'Nighttime': average_night_aqi
})
```

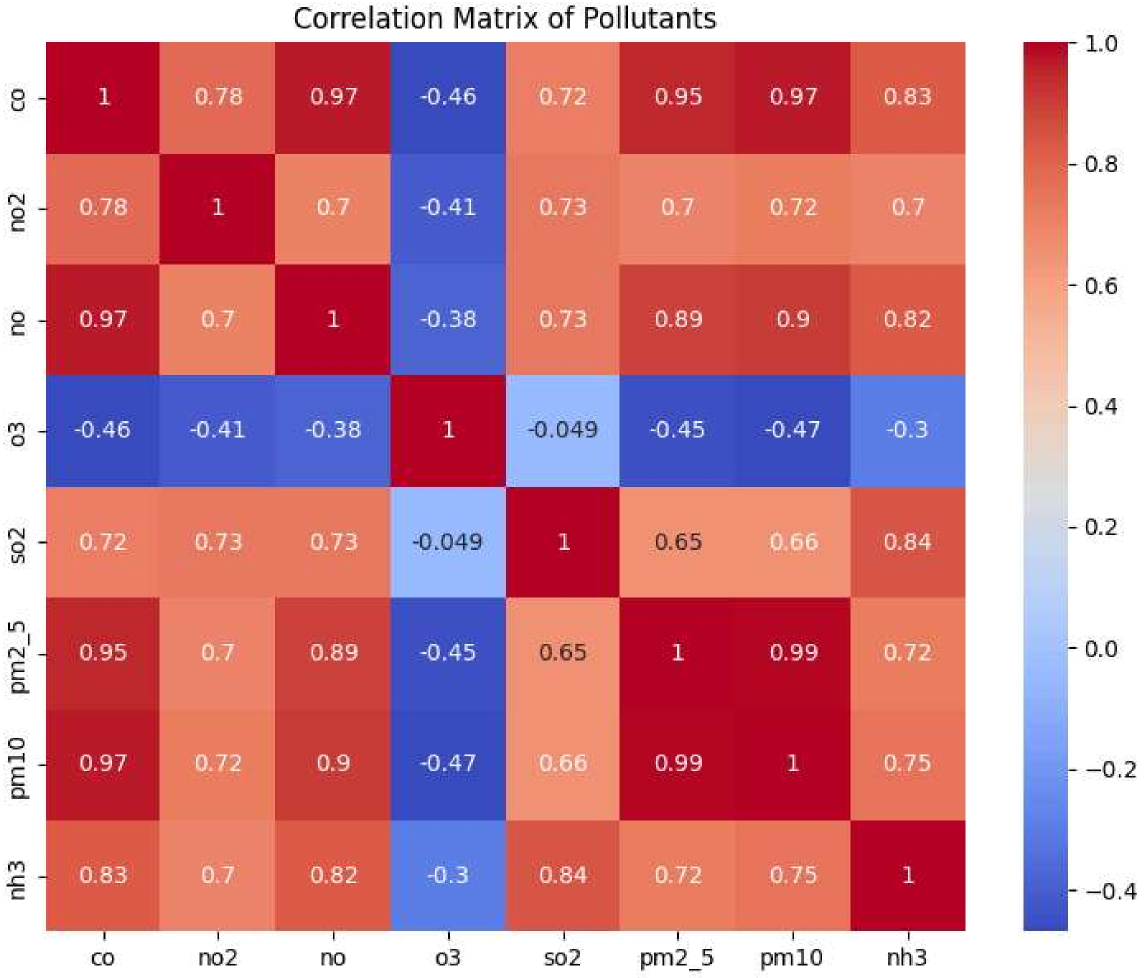
In [18]:
```
plt.figure(figsize=(8, 6))
sns.scatterplot(data=day_night_aqi_comparison, alpha=0.7)
plt.title('Comparison of Daytime and Nighttime Average Air Quality Inde
x')
plt.xlabel('Date')
plt.ylabel('Average Air Quality Index')
```

Out[18]:
```
Text(0, 0.5, 'Average Air Quality Index')
```



```
In [19]: pollutants = ['co', 'no2', 'no', 'o3', 'so2', 'pm2_5', 'pm10', 'nh3']
         correlationmatrix = data[pollutants].corr()
         plt.figure(figsize=(9, 7))
         sns.heatmap(correlationmatrix, annot=True, cmap='coolwarm')
         plt.title('Correlation Matrix of Pollutants')
```

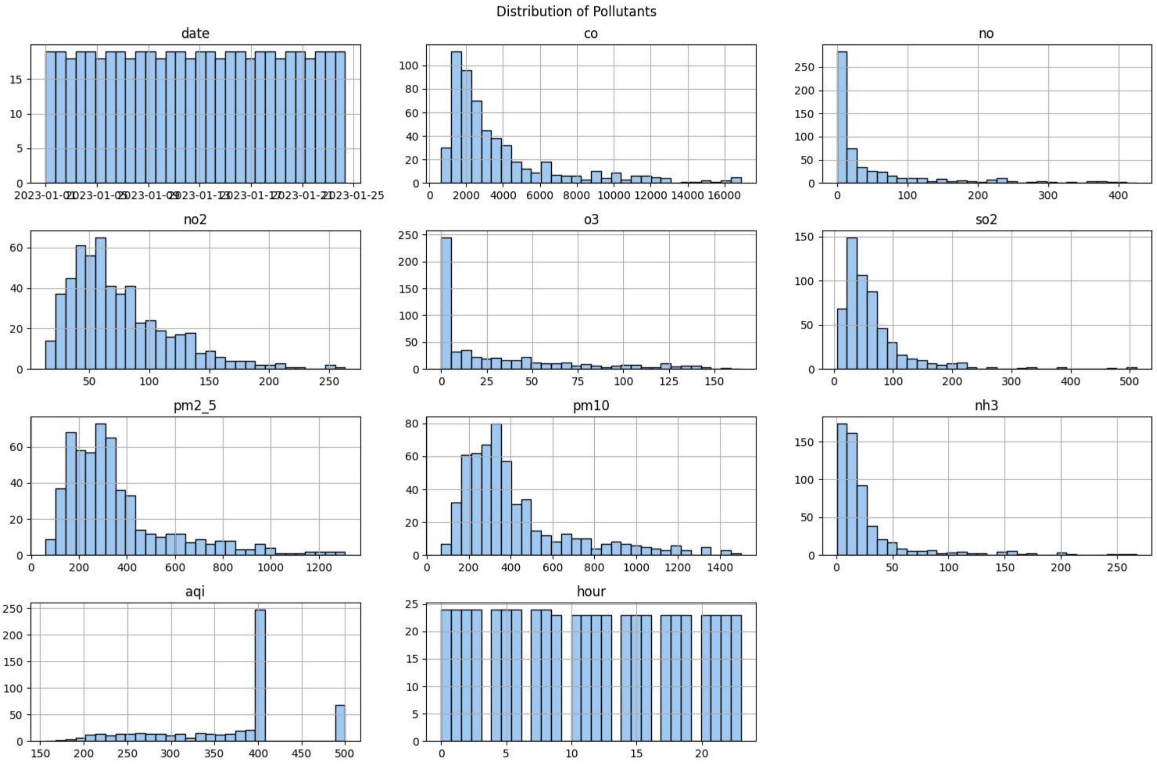
```
Out[19]: Text(0.5, 1.0, 'Correlation Matrix of Pollutants')
```



Distribution of Pollutants

In [20]:

```
features = data.drop(data[data['date'] == 'date'].index)
features.hist(bins = 30, figsize = (15,10), edgecolor = 'black')
plt.suptitle('Distribution of Pollutants')
plt.tight_layout()
sns.set_palette('pastel')
plt.show()
```


Distribution of Pollutants
date
co
no
no2
o3
so2
pm2_5
pm10
nh3
aqi
hour