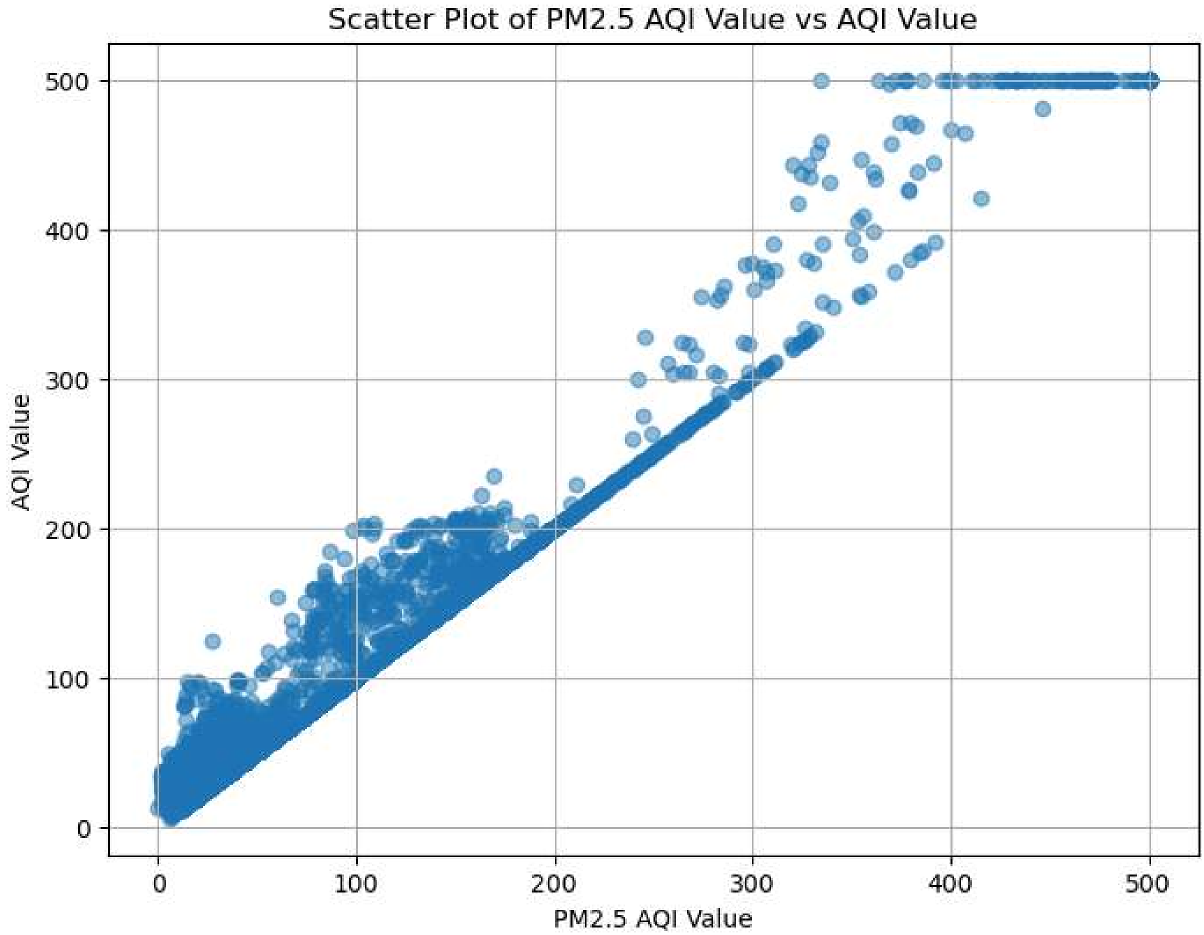
DSC530

Week 12 Project

Andrew Finch

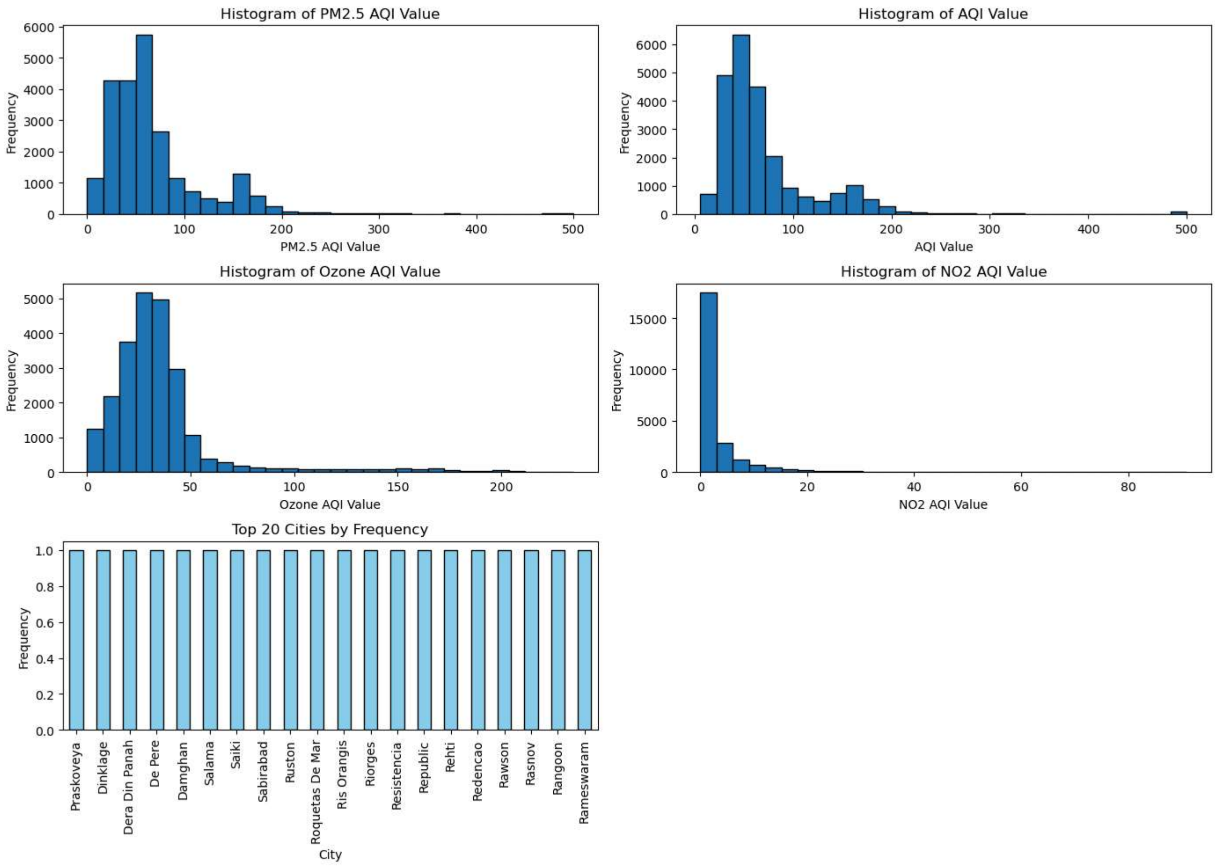
In [6]:
```
import pandas as pd
import matplotlib.pyplot as plt

file_path = r'C:\Users\finch\DSC540\Week5and6\Project\global air pollution dataset.
data = pd.read_csv(file_path)

# Scatter plot for comparing PM2.5 AQI Value and AQI Value
plt.figure(figsize=(8, 6))
plt.scatter(data['PM2.5 AQI Value'], data['AQI Value'], alpha=0.5)
plt.title('Scatter Plot of PM2.5 AQI Value vs AQI Value')
plt.xlabel('PM2.5 AQI Value')
plt.ylabel('AQI Value')
plt.grid(True)
plt.show()

# Pearson correlation coefficient
correlation = data['PM2.5 AQI Value'].corr(data['AQI Value'])
print("Pearson Correlation between PM2.5 AQI Value and AQI Value:", correlation)
```




```
Pearson Correlation between PM2.5 AQI Value and AQI Value: 0.9843265891583709
```

In [8]:
```
import pandas as pd
import matplotlib.pyplot as plt

file_path = r'C:\Users\finch\DSC540\Week5and6\Project\global air pollution dataset.
data = pd.read_csv(file_path)

plt.figure(figsize=(14, 10))

# List of variables
variables = ['PM2.5 AQI Value', 'AQI Value', 'Ozone AQI Value', 'NO2 AQI Value']

for i, var in enumerate(variables, 1):
    plt.subplot(3, 2, i)
    data[var].plot(kind='hist', bins=30, edgecolor='black')
    plt.title(f'Histogram of {var}')
    plt.xlabel(var)
    plt.ylabel('Frequency')

plt.subplot(3, 2, 5)
data['City'].value_counts().head(20).plot(kind='bar', color='skyblue', edgecolor='b
plt.title('Top 20 Cities by Frequency')
plt.xlabel('City')
plt.ylabel('Frequency')
```

```
plt.tight_layout()
plt.show()
```



In [12]:
```
import pandas as pd

file_path = r'C:\Users\finch\DSC540\Week5and6\Project\global air pollution dataset.
data = pd.read_csv(file_path)

variables = ['PM2.5 AQI Value', 'AQI Value', 'Ozone AQI Value', 'NO2 AQI Value']

descriptive_stats = {}

for var in variables:
    mean_val = data[var].mean()
    mode_val = data[var].mode()[0]
    std_dev = data[var].std()
    min_val = data[var].min()
    max_val = data[var].max()

    descriptive_stats[var] = {
        "Mean": mean_val,
        "Mode": mode_val,
        "Standard Deviation": std_dev,
        "Min (Left Tail)": min_val,
        "Max (Right Tail)": max_val
    }


descriptive_stats_df = pd.DataFrame(descriptive_stats).T
```

```
print("Descriptive Statistics for Selected Variables:")
print(descriptive_stats_df)
```

```
Descriptive Statistics for Selected Variables:
                      Mean  Mode  Standard Deviation  Min (Left Tail)  \
PM2.5 AQI Value  68.519755  50.0           54.796443              0.0
AQI Value        72.010868  50.0           56.055220              6.0
Ozone AQI Value  35.193709  30.0           28.098723              0.0
NO2 AQI Value     3.063334   0.0            5.254108              0.0

                 Max (Right Tail)
PM2.5 AQI Value             500.0
AQI Value                   500.0
Ozone AQI Value             235.0
NO2 AQI Value                91.0
```

In [14]:

```
import pandas as pd
import numpy as np

file_path = r'C:\Users\finch\DSC540\Week5and6\Project\global air pollution dataset.
data = pd.read_csv(file_path)

def calculate_pmf(values):
    # Calculate the frequency of each unique value
    unique, counts = np.unique(values, return_counts=True)
    probabilities = counts / counts.sum()
    return dict(zip(unique, probabilities))

urban_cities = ["New York", "Tokyo", "London", "Paris"]
rural_cities = ["Praskoveya", "Ruston", "Sabirabad", "Damghan", "Dinklage"]

# Filter data for urban and rural PM2.5 AQI values
urban_pm25_values = data[data['City'].isin(urban_cities)]['PM2.5 AQI Value']
rural_pm25_values = data[data['City'].isin(rural_cities)]['PM2.5 AQI Value']

# Calculate PMF for urban and rural PM2.5 AQI values
urban_pmf = calculate_pmf(urban_pm25_values)
rural_pmf = calculate_pmf(rural_pm25_values)

pmf_results = {
    "Urban PM2.5 AQI PMF": urban_pmf,
    "Rural PM2.5 AQI PMF": rural_pmf
}

pmf_results
```

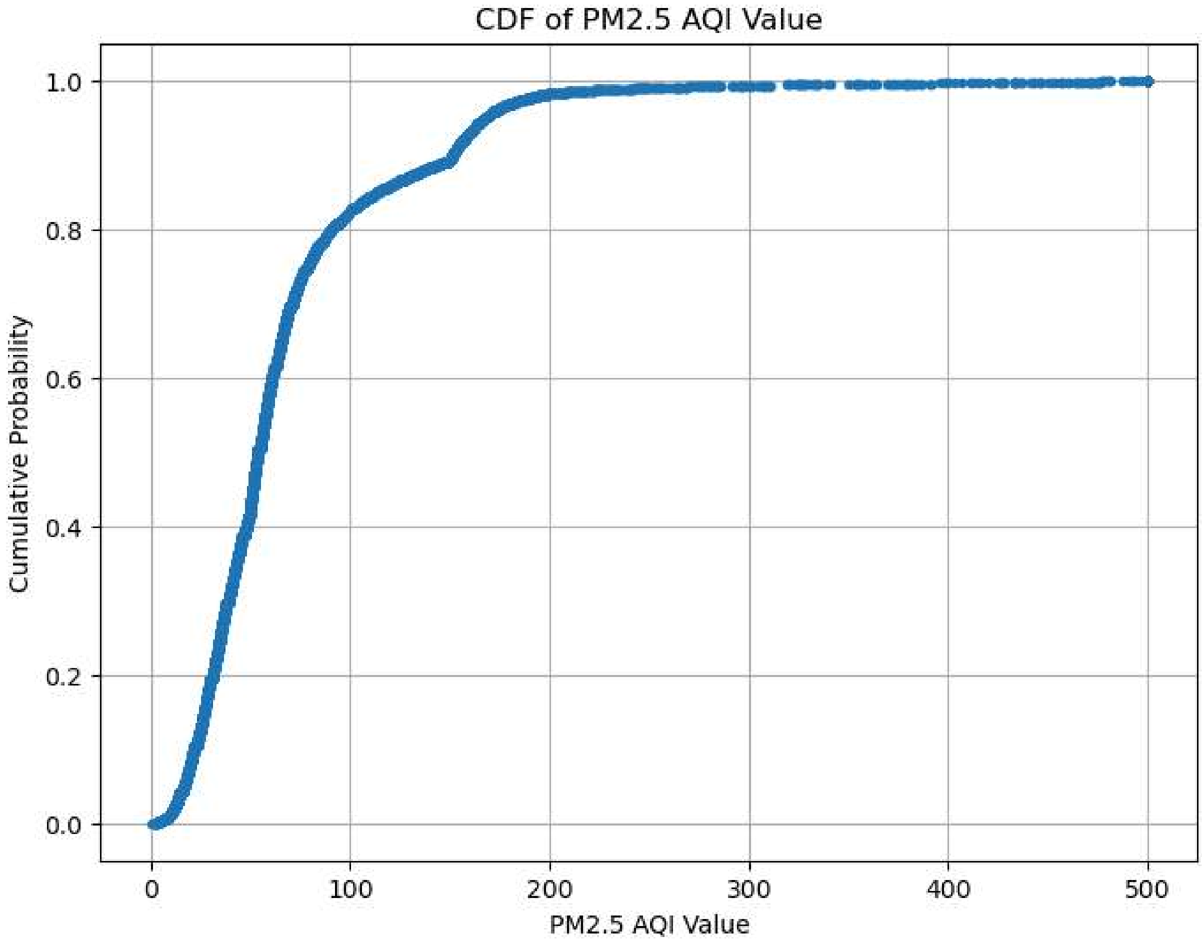
Out[14]:

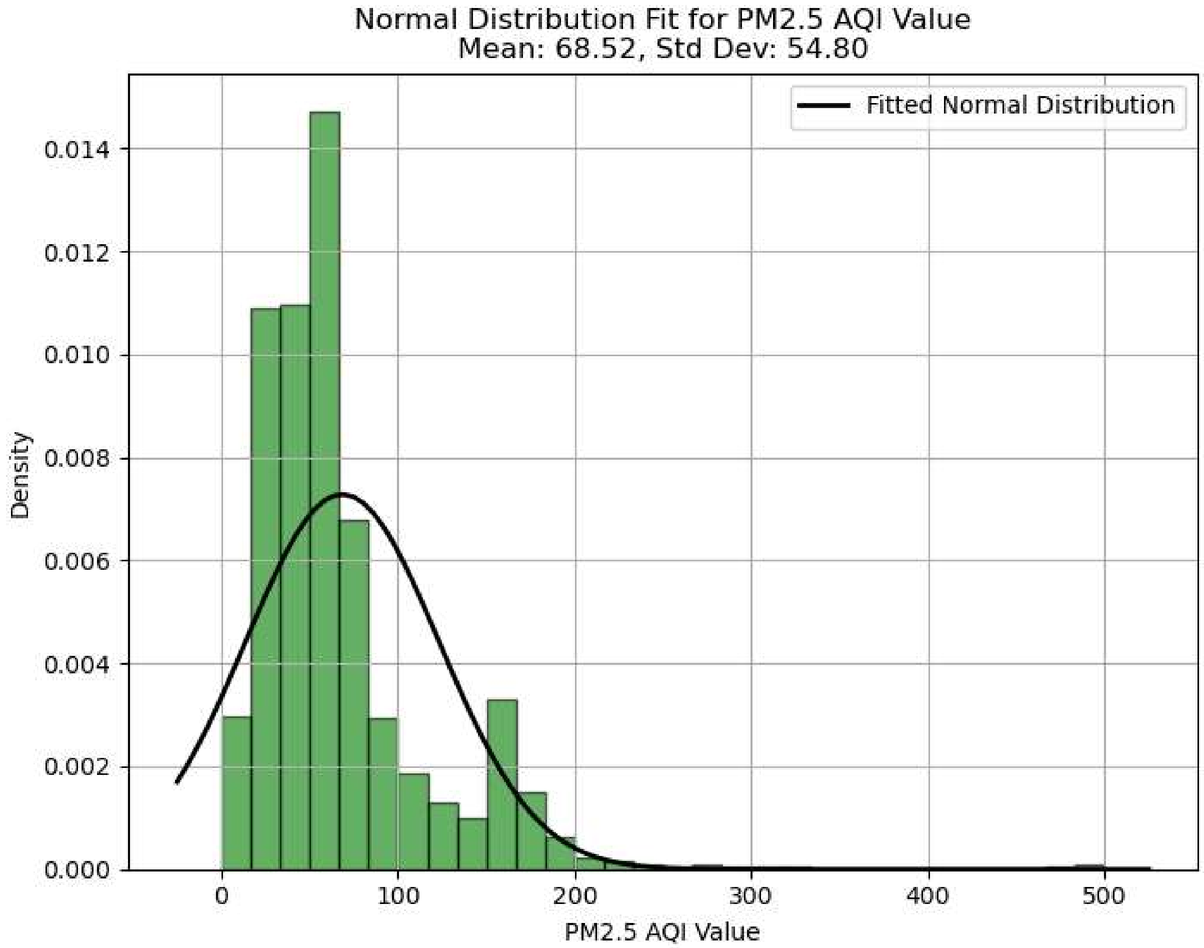
```
{'Urban PM2.5 AQI PMF': {40: 0.25, 70: 0.25, 72: 0.25, 79: 0.25},
 'Rural PM2.5 AQI PMF': {36: 0.2, 51: 0.2, 64: 0.2, 75: 0.2, 91: 0.2}}
```

In [16]:

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from scipy.stats import norm

file_path = r'C:\Users\finch\DSC540\Week5and6\Project\global air pollution dataset.
```

```
data = pd.read_csv(file_path)

pm25_values = data['PM2.5 AQI Value'].dropna()
sorted_values = np.sort(pm25_values)
cdf = np.arange(1, len(sorted_values) + 1) / len(sorted_values)

plt.figure(figsize=(8, 6))
plt.plot(sorted_values, cdf, marker='.', linestyle='none')
plt.title('CDF of PM2.5 AQI Value')
plt.xlabel('PM2.5 AQI Value')
plt.ylabel('Cumulative Probability')
plt.grid(True)
plt.show()

mean, std_dev = norm.fit(pm25_values)

# normal distribution
plt.figure(figsize=(8, 6))
plt.hist(pm25_values, bins=30, density=True, alpha=0.6, color='g', edgecolor='black
xmin, xmax = plt.xlim()
x = np.linspace(xmin, xmax, 100)
p = norm.pdf(x, mean, std_dev)
plt.plot(x, p, 'k', linewidth=2, label='Fitted Normal Distribution')

plt.title(f'Normal Distribution Fit for PM2.5 AQI Value\nMean: {mean:.2f}, Std Dev:
plt.xlabel('PM2.5 AQI Value')
plt.ylabel('Density')
plt.legend()
plt.grid(True)
plt.show()
```

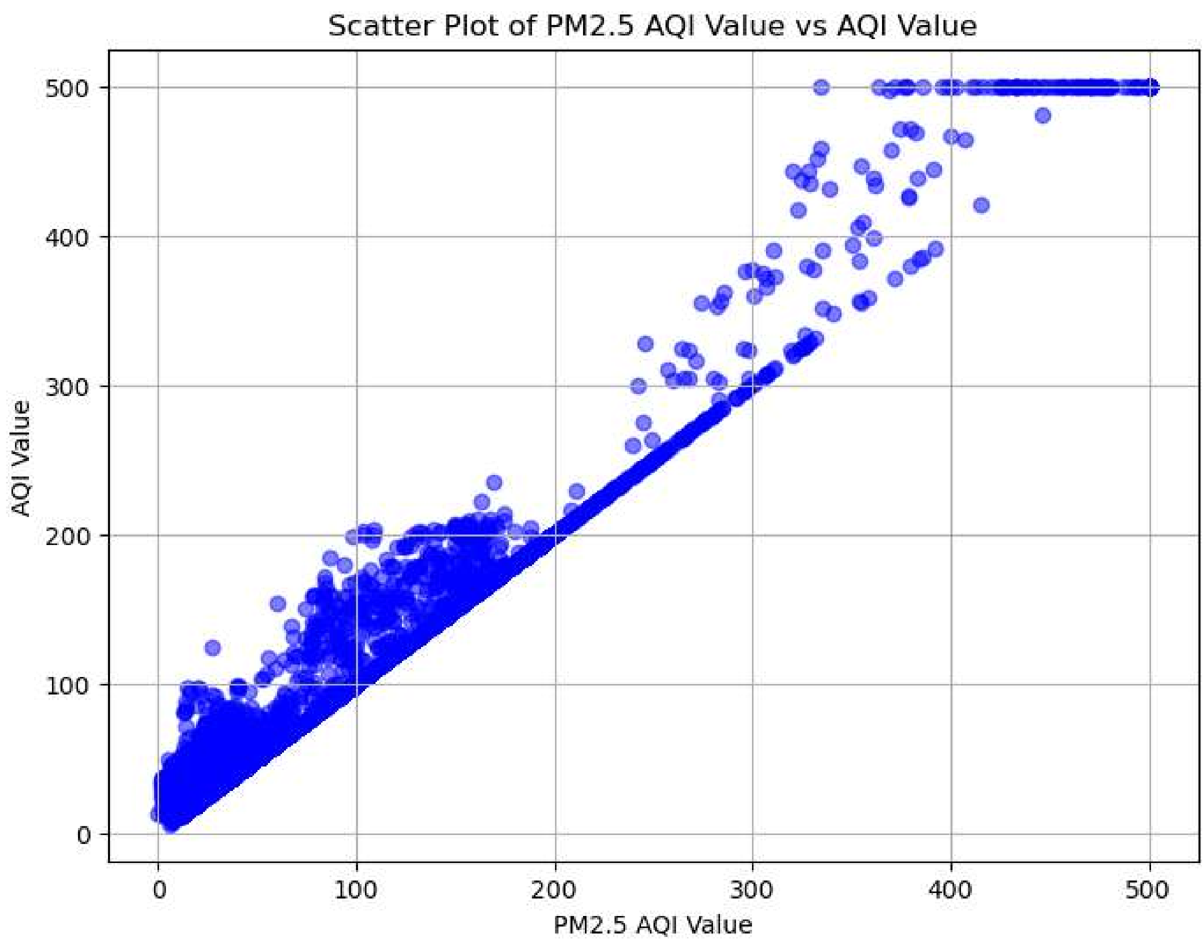

CDF of PM2.5 AQI Value
Cumulative Probability
PM2.5 AQI Value
0.0
0.2
0.4
0.6
0.8
1.0
0
100
200
300
400
500

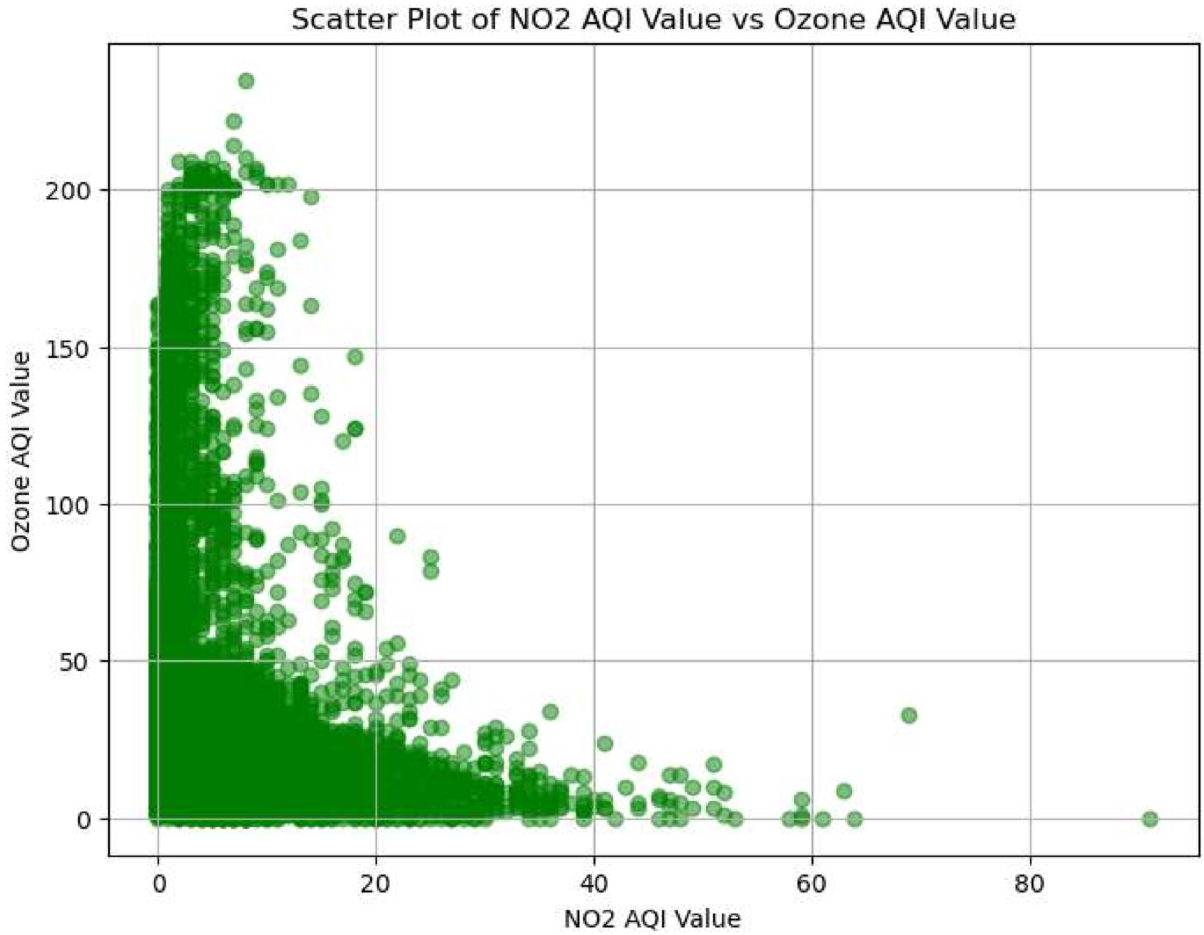


In [18]:

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

file_path = r'C:\Users\finch\DSC540\Week5and6\Project\global air pollution dataset.
data = pd.read_csv(file_path)

# Scatter Plot 1: PM2.5 AQI Value vs. AQI Value
plt.figure(figsize=(8, 6))
plt.scatter(data['PM2.5 AQI Value'], data['AQI Value'], alpha=0.5, color='blue')
plt.title('Scatter Plot of PM2.5 AQI Value vs AQI Value')
plt.xlabel('PM2.5 AQI Value')
plt.ylabel('AQI Value')
plt.grid(True)
plt.show()

# Calculate Pearson's correlation and covariance for PM2.5 AQI Value vs AQI Value
corr_pm25_aqi = data['PM2.5 AQI Value'].corr(data['AQI Value'])
cov_pm25_aqi = data['PM2.5 AQI Value'].cov(data['AQI Value'])

# Scatter Plot 2: NO2 AQI Value vs. Ozone AQI Value
plt.figure(figsize=(8, 6))
plt.scatter(data['NO2 AQI Value'], data['Ozone AQI Value'], alpha=0.5, color='green
plt.title('Scatter Plot of NO2 AQI Value vs Ozone AQI Value')
plt.xlabel('NO2 AQI Value')
plt.ylabel('Ozone AQI Value')
plt.grid(True)
```

```
plt.show()

corr_no2_ozone = data['NO2 AQI Value'].corr(data['Ozone AQI Value'])
cov_no2_ozone = data['NO2 AQI Value'].cov(data['Ozone AQI Value'])

correlation_covariance_results = {
    "PM2.5 vs AQI": {
        "Pearson's Correlation": corr_pm25_aqi,
        "Covariance": cov_pm25_aqi
    },
    "NO2 vs Ozone": {
        "Pearson's Correlation": corr_no2_ozone,
        "Covariance": cov_no2_ozone
    }
}

correlation_covariance_results
```





Out[18]:
```
{'PM2.5 vs AQI': {"Pearson's Correlation": 0.9843265891583709,
  'Covariance': 3023.4838233143237},
 'NO2 vs Ozone': {"Pearson's Correlation": -0.18181765117472903,
  'Covariance': -26.84241973988119}}
```

In [20]:
```
import pandas as pd
from scipy.stats import ttest_ind

file_path = r'C:\Users\finch\DSC540\Week5and6\Project\global air pollution dataset.
data = pd.read_csv(file_path)

urban_cities = ["New York", "Tokyo", "London", "Paris"]
rural_cities = ["Praskoveya", "Ruston", "Sabirabad", "Damghan", "Dinklage"]

# PM2.5 AQI values for urban and rural cities
urban_pm25_values = data[data['City'].isin(urban_cities)]['PM2.5 AQI Value']
rural_pm25_values = data[data['City'].isin(rural_cities)]['PM2.5 AQI Value']

# Perform a two-sample t-test
t_stat, p_value = ttest_ind(urban_pm25_values, rural_pm25_values, alternative='grea

test_results = {
    "T-statistic": t_stat,
    "P-value": p_value
}

test_results
```

Out[20]: {'T-statistic': 0.1441873406986751, 'P-value': 0.4447127570664692}

In [22]:
```
import pandas as pd
import statsmodels.api as sm

file_path = r'C:\Users\finch\DSC540\Week5and6\Project\global air pollution dataset.
data = pd.read_csv(file_path)

y = data['PM2.5 AQI Value']

X = data[['AQI Value', 'NO2 AQI Value', 'Ozone AQI Value']]

# Add a constant to the independent variables matrix for intercept
X = sm.add_constant(X)

model = sm.OLS(y, X).fit()

model_summary = model.summary()
print(model_summary)
```

```
                            OLS Regression Results                            
==============================================================================
Dep. Variable:        PM2.5 AQI Value   R-squared:                       0.973
Model:                            OLS   Adj. R-squared:                  0.973
Method:                 Least Squares   F-statistic:                 2.846e+05
Date:                Sun, 10 Nov 2024   Prob (F-statistic):               0.00
Time:                        22:16:52   Log-Likelihood:                -84744.
No. Observations:               23463   AIC:                         1.695e+05
Df Residuals:                   23459   BIC:                         1.695e+05
Df Model:                           3                                         
Covariance Type:            nonrobust                                         
===================================================================================
                      coef    std err          t      P>|t|      [0.025      0.975]
-----------------------------------------------------------------------------------
const               1.6415      0.111     14.755      0.000       1.423       1.860
AQI Value           0.9850      0.001    811.413      0.000       0.983       0.987
NO2 AQI Value       0.1525      0.012     12.670      0.000       0.129       0.176
Ozone AQI Value    -0.1284      0.002    -53.607      0.000      -0.133      -0.124
==============================================================================
Omnibus:                    19569.786   Durbin-Watson:                   1.983
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           904371.378
Skew:                          -3.747   Prob(JB):                         0.00
Kurtosis:                      32.477   Cond. No.                         187.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly spe
cified.
```

In [ ]: