```
import os
import sys
from tempfile import NamedTemporaryFile
from urllib.request import urlopen
from urllib.parse import unquote, urlparse
from urllib.error import HTTPError
from zipfile import ZipFile
import tarfile
import shutil

CHUNK_SIZE = 40960
DATA_SOURCE_MAPPING = 'air-quality-data-in-india:https%3A%2F%2Fstorage.googleapis.com%2Fkaggle-data-sets%2F630055%2F1377609%2Fbundle%2Far

KAGGLE_INPUT_PATH='/kaggle/input'
KAGGLE_WORKING_PATH='/kaggle/working'
KAGGLE_SYMLINK='kaggle'

!umount /kaggle/input/ 2> /dev/null
shutil.rmtree('/kaggle/input', ignore_errors=True)
os.makedirs(KAGGLE_INPUT_PATH, 0o777, exist_ok=True)
os.makedirs(KAGGLE_WORKING_PATH, 0o777, exist_ok=True)

try:
  os.symlink(KAGGLE_INPUT_PATH, os.path.join("..", 'input'), target_is_directory=True)
except FileExistsError:
  pass
try:
  os.symlink(KAGGLE_WORKING_PATH, os.path.join("..", 'working'), target_is_directory=True)
except FileExistsError:
  pass

for data_source_mapping in DATA_SOURCE_MAPPING.split(','):
    directory, download_url_encoded = data_source_mapping.split(':')
    download_url = unquote(download_url_encoded)
    filename = urlparse(download_url).path
    destination_path = os.path.join(KAGGLE_INPUT_PATH, directory)
    try:
        with urlopen(download_url) as fileres, NamedTemporaryFile() as tfile:
            total_length = fileres.headers['content-length']
            print(f'Downloading {directory}, {total_length} bytes compressed')
            dl = 0
            data = fileres.read(CHUNK_SIZE)
            while len(data) > 0:
                dl += len(data)
                tfile.write(data)
                done = int(50 * dl / int(total_length))
                sys.stdout.write(f"\r[{'=' * done}{' ' * (50-done)}] {dl} bytes downloaded")
                sys.stdout.flush()
                data = fileres.read(CHUNK_SIZE)
            if filename.endswith('.zip'):
              with ZipFile(tfile) as zfile:
                zfile.extractall(destination_path)
            else:
              with tarfile.open(tfile.name) as tarfile:
                tarfile.extractall(destination_path)
            print(f'\nDownloaded and uncompressed: {directory}')
    except HTTPError as e:
        print(f'Failed to load (likely expired) {download_url} to path {destination_path}')
        continue
    except OSError as e:
        print(f'Failed to load {download_url} to path {destination_path}')
        continue

print('Data source import complete.')
```

```
Downloading air-quality-data-in-india, 76469579 bytes compressed
[==================================================] 76469579 bytes downloaded
Downloaded and uncompressed: air-quality-data-in-india
Downloading geographic-data-of-indian-cities, 5295 bytes compressed
[==================================================] 5295 bytes downloaded
Downloaded and uncompressed: geographic-data-of-indian-cities
Data source import complete.
```

Air Quality Index for Cities in India 2015 - 2020

Table of Contents

1. Introduction
2. Data description and cleaning
3. Data analysis
4. Conclusion

1. Introduction

The project aims to develop a predictive model for urban air quality index (AQI) by integrating meteorological factors and pollution sources. By analyzing historical data on air quality parameters, such as particulate matter (PM2.5, PM10), ozone (O3), nitrogen dioxide (NO2), sulfur dioxide (SO2), and carbon monoxide (CO), alongside meteorological data like temperature, humidity, wind speed, and pollution source data such as industrial emissions and vehicular traffic, the project seeks to understand the complex interplay between these factors. Through advanced data analysis techniques and machine learning algorithms, the goal is to create a model that accurately forecasts urban AQI levels, providing valuable insights for policymakers, urban planners, and public health officials to mitigate air pollution and improve overall air quality in urban areas.

The dataset is a collection of pollutant readings across cities in India recorded between 2015 and 2020. The data consists of 26 cities in India, and is split into the following categories: -

- Date - daily readings between 2015 and 2020
- PM2.5 - Particulate Matter 2.5-micrometer in ug / m3
- PM10 - Particulate Matter 10-micrometer in ug / m3
- NO - Nitric Oxide in ug / m3
- NO2 - Nitric Dioxide in ug / m3
- NOx - Any Nitric x-oxide in ppb
- NH3 - Ammonia in ug / m3
- CO - Carbon Monoxide in mg / m3
- SO2 - Sulphur Dioxide in ug / m3
- O3 - Ozone in ug / m3
- Benzene - Benzene in ug / m3
- Toluene - Toluene in ug / m3
- Xylene - Xylene in ug / m3
- AQI - Air Quality Index
- AQI Bucket - Air Quality Index Bucket (ranging from 'very poor' to 'good')

2. Data Description and Cleaning

The data is in the form of a csv file and I have used a combination of Python and Pandas library to read the file and clean the data (i.e. drop any duplicated data and handle any missing values). Later, in section 3 there are some data visuals and for this I have used Plotly, numpy and matplotlib .

2.1 Data Description

- import libraries
- read the data from the csv file
- determine the data types in the file
- understand the data shape

```
#import libraries
import pandas as pd
import numpy as np
import os, sys
import sqlite3


import matplotlib.pyplot as plt
import numpy as np

import plotly as py
import plotly.express as px
import plotly.graph_objs as go
#import plotly.offline as offline
#offline.init_notebook_mode(connected=True)

#load data
india_air_data = pd.read_csv("/kaggle/input/air-quality-data-in-india/city_day.csv")
india_air_data.head()
```

	City	Date	PM2.5	PM10	NO	NO2	NOx	NH3	CO	SO2	O3	Benzer
0	Ahmedabad	2015-01-01	NaN	NaN	0.92	18.22	17.15	NaN	0.92	27.64	133.36	0.(
1	Ahmedabad	2015-01-02	NaN	NaN	0.97	15.69	16.46	NaN	0.97	24.55	34.06	3.6
2	Ahmedabad	2015-01-03	NaN	NaN	17.40	19.30	29.70	NaN	17.40	29.07	30.70	6.8

```
india_air_data.tail()
```

	City	Date	PM2.5	PM10	NO	NO2	NOx	NH3	CO	SO2	O3	E
29526	Visakhapatnam	2020-06-27	15.02	50.94	7.68	25.06	19.54	12.47	0.47	8.55	23.30	
29527	Visakhapatnam	2020-06-28	24.38	74.09	3.42	26.06	16.53	11.99	0.52	12.72	30.14	
29528	Visakhapatnam	2020-06-29	22.91	65.73	3.45	29.53	18.33	10.71	0.48	8.42	30.96	

```
india_air_data.columns
```

```
Index(['City', 'Date', 'PM2.5', 'PM10', 'NO', 'NO2', 'NOx', 'NH3', 'CO', 'SO2',
       'O3', 'Benzene', 'Toluene', 'Xylene', 'AQI', 'AQI_Bucket'],
      dtype='object')
```

```
india_air_data['AQI_Bucket'].unique()
```

```
array([nan, 'Poor', 'Very Poor', 'Severe', 'Moderate', 'Satisfactory',
       'Good'], dtype=object)
```

```
#data description
india_air_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 29531 entries, 0 to 29530
Data columns (total 16 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   City        29531 non-null  object 
 1   Date        29531 non-null  object 
 2   PM2.5       24933 non-null  float64
 3   PM10        18391 non-null  float64
 4   NO          25949 non-null  float64
 5   NO2         25946 non-null  float64
 6   NOx         25346 non-null  float64
 7   NH3         19203 non-null  float64
 8   CO          27472 non-null  float64
 9   SO2         25677 non-null  float64
 10  O3          25509 non-null  float64
 11  Benzene     23908 non-null  float64
 12  Toluene     21490 non-null  float64
 13  Xylene      11422 non-null  float64
 14  AQI         24850 non-null  float64
 15  AQI_Bucket  24850 non-null  object 
dtypes: float64(13), object(3)
memory usage: 3.6+ MB
```

```
india_air_data.shape
```

```
(29531, 16)
```

```
#Basic Stats
print(india_air_data['City'].nunique())
print(india_air_data['City'].unique())
```

```
26
['Ahmedabad' 'Aizawl' 'Amaravati' 'Amritsar' 'Bengaluru' 'Bhopal'
 'Brajrajnagar' 'Chandigarh' 'Chennai' 'Coimbatore' 'Delhi' 'Ernakulam'
 'Gurugram' 'Guwahati' 'Hyderabad' 'Jaipur' 'Jorapokhar' 'Kochi' 'Kolkata'
 'Lucknow' 'Mumbai' 'Patna' 'Shillong' 'Talcher' 'Thiruvananthapuram'
 'Visakhapatnam']
```

```
city_readings = india_air_data['City'].value_counts().to_frame().reset_index().rename(columns={'index':'City Name', 'City':'No. of reads
city_readings['%'] = (100* city_readings['No. of readings']/city_readings['No. of readings'].sum()).round(0)
city_readings
```

	City Name	No. of readings	%
0	Ahmedabad	2009	7.0
1	Delhi	2009	7.0
2	Mumbai	2009	7.0
3	Bengaluru	2009	7.0
4	Lucknow	2009	7.0
5	Chennai	2009	7.0
6	Hyderabad	2006	7.0
7	Patna	1858	6.0
8	Gurugram	1679	6.0
9	Visakhapatnam	1462	5.0
10	Amritsar	1221	4.0
11	Jorapokhar	1169	4.0
12	Jaipur	1114	4.0
13	Thiruvananthapuram	1112	4.0
14	Amaravati	951	3.0
15	Brajrajnagar	938	3.0
16	Talcher	925	3.0
17	Kolkata	814	3.0
18	Guwahati	502	2.0
19	Coimbatore	386	1.0
20	Shillong	310	1.0
21	Chandigarh	304	1.0
22	Bhopal	289	1.0
23	Ernakulam	162	1.0
24	Kochi	162	1.0
25	Aizawl	113	0.0

```
#Number of readings per year
india_air_data['Date'] = pd.to_datetime(india_air_data['Date'])
annual_readings = india_air_data.Date.dt.year.value_counts().to_frame().reset_index().rename(columns={'index':'Year', 'Date':'No. of rea
annual_readings['%'] = (100* annual_readings['No. of readings']/annual_readings['No. of readings'].sum()).round(0)
annual_readings
```

	Year	No. of readings	%
0	2019	7446	25.0
1	2018	6471	22.0
2	2017	4689	16.0
3	2020	4646	16.0
4	2016	3478	12.0
5	2015	2801	9.0

```
#general trend for numeric data
pollutant_columns = india_air_data[['PM2.5', 'PM10', 'NO', 'NO2', 'NOx', 'NH3', 'CO', 'SO2','O3', 'Benzene', 'Toluene', 'Xylene', 'AQI']
pollutant_columns.describe().T
```

	count	mean	std	min	25%	50%	75%	max
PM2.5	24933.0	67.450578	64.661449	0.04	28.820	48.57	80.5900	949.99
PM10	18391.0	118.127103	90.605110	0.01	56.255	95.68	149.7450	1000.00
NO	25949.0	17.574730	22.785846	0.02	5.630	9.89	19.9500	390.68
NO2	25946.0	28.560659	24.474746	0.01	11.750	21.69	37.6200	362.21
NOx	25346.0	32.309123	31.646011	0.00	12.820	23.52	40.1275	467.63
NH3	19203.0	23.483476	25.684275	0.01	8.580	15.85	30.0200	352.89
CO	27472.0	2.248598	6.962884	0.00	0.510	0.89	1.4500	175.81
SO2	25677.0	14.531977	18.133775	0.01	5.670	9.16	15.2200	193.86
O3	25509.0	34.491430	21.694928	0.01	18.860	30.84	45.5700	257.73
Benzene	23908.0	3.280840	15.811136	0.00	0.120	1.07	3.0800	455.03
Toluene	21490.0	8.700972	19.969164	0.00	0.600	2.97	9.1500	454.85
Xylene	11422.0	3.070128	6.323247	0.00	0.140	0.98	3.3500	170.37
AQI	24850.0	166.463581	140.696585	13.00	81.000	118.00	208.0000	2049.00

2.2 Data Cleaning

- locate any missing values
- check for duplicated items

```
#clean data
missing_values_count = india_air_data.isnull().sum()
total_cells = np.product(india_air_data.shape)
total_missing = missing_values_count.sum()
percent_missing_vals = ((total_missing/total_cells)*100).round(2)
print(missing_values_count[:])
print('Overall percentage of missing values: ', percent_missing_vals)
```

```
City              0
Date              0
PM2.5          4598
PM10          11140
NO             3582
NO2            3585
NOx            4185
NH3           10328
CO             2059
SO2            3854
O3             4022
Benzene        5623
Toluene        8041
Xylene        18109
AQI            4681
AQI_Bucket     4681
dtype: int64
Overall percentage of missing values:  18.73
```

Overall, 18.73% of the data missing. The reason for missing values could be a combination of records missed and data not existing. It is clear that the dataset has increased over time to include more cities across India. To gain more accurate findings I will fill in the missing values with the mean value for a given numeric column.

```
#Replace the missing values for numerical columns with mean

india_air_data['PM2.5'].fillna(india_air_data['PM2.5'].mean().round(2),inplace = True )
india_air_data['PM10'].fillna(india_air_data['PM10'].mean().round(2),inplace = True )
india_air_data['NO'].fillna(india_air_data['NO'].mean().round(2),inplace = True )
india_air_data['NO2'].fillna(india_air_data['NO2'].mean().round(2),inplace = True )
india_air_data['NOx'].fillna(india_air_data['NOx'].mean().round(2),inplace = True )
india_air_data['NH3'].fillna(india_air_data['NH3'].mean().round(2),inplace = True )
india_air_data['CO'].fillna(india_air_data['CO'].mean().round(2),inplace = True )
india_air_data['SO2'].fillna(india_air_data['SO2'].mean().round(2),inplace = True )
india_air_data['O3'].fillna(india_air_data['O3'].mean().round(2),inplace = True )
india_air_data['Benzene'].fillna(india_air_data['Benzene'].mean().round(2),inplace = True )
india_air_data['Toluene'].fillna(india_air_data['Toluene'].mean().round(2),inplace = True )
india_air_data['Xylene'].fillna(india_air_data['Xylene'].mean().round(2),inplace = True )
india_air_data['AQI'].fillna(india_air_data['AQI'].mean().round(2),inplace = True )


india_air_data.head()
```

	City	Date	PM2.5	PM10	NO	NO2	NOx	NH3	CO	SO2	O3	Ben
0	Ahmedabad	2015-01-01	67.45	118.13	0.92	18.22	17.15	23.48	0.92	27.64	133.36	
1	Ahmedabad	2015-01-02	67.45	118.13	0.97	15.69	16.46	23.48	0.97	24.55	34.06	
2	Ahmedabad	2015-01-03	67.45	118.13	17.40	19.30	29.70	23.48	17.40	29.07	30.70	

The missing numeric data has now been resolved. The column headed 'AQI Bucket' categorises the air quality into 'Poor', 'Very Poor', 'Severe', 'Moderate', 'Satisfactory', 'Good'. Let's take a closer look.

```
india_air_data['AQI_Bucket'].isnull().sum()
```

```
4681
```

In total, there are 4,681 missing descriptions in the 'AQI Bucket' column. Let's populate the missing rows in accordance with the score system implemented.

AQI Score System: -

- Good (0-50)
- Satisfactory (51-100)
- Moderate (101-200)
- Poor (201-300)
- Very poor (301-400)
- Severe (401-500)

```
india_air_data.loc[(india_air_data['AQI'] >= 0) & (india_air_data['AQI'] <= 50), 'AQI_Bucket'] = 'Good'
india_air_data.loc[(india_air_data['AQI'] >= 51) & (india_air_data['AQI'] <= 100), 'AQI_Bucket'] = 'Satisfactory'
india_air_data.loc[(india_air_data['AQI'] >= 101) & (india_air_data['AQI'] <= 200), 'AQI_Bucket'] = 'Moderate'
india_air_data.loc[(india_air_data['AQI'] >= 201) & (india_air_data['AQI'] <= 300), 'AQI_Bucket'] = 'Poor'
india_air_data.loc[(india_air_data['AQI'] >= 301) & (india_air_data['AQI'] <= 400), 'AQI_Bucket'] = 'Very poor'
india_air_data.loc[(india_air_data['AQI'] >= 401) & (india_air_data['AQI'] <= 500), 'AQI_Bucket'] = 'Severe'
```

```
india_air_data
```

	City	Date	PM2.5	PM10	NO	NO2	NOx	NH3	CO	SO2	
0	Ahmedabad	2015-01-01	67.45	118.13	0.92	18.22	17.15	23.48	0.92	27.64	133.
1	Ahmedabad	2015-01-02	67.45	118.13	0.97	15.69	16.46	23.48	0.97	24.55	34.
2	Ahmedabad	2015-01-03	67.45	118.13	17.40	19.30	29.70	23.48	17.40	29.07	30.
3	Ahmedabad	2015-01-04	67.45	118.13	1.70	18.48	17.97	23.48	1.70	18.59	36.
4	Ahmedabad	2015-01-05	67.45	118.13	22.10	21.42	37.76	23.48	22.10	39.33	39.
...	...	...	...	...	...	...	...	...	...	...	
29526	Visakhapatnam	2020-06-27	15.02	50.94	7.68	25.06	19.54	12.47	0.47	8.55	23.
29527	Visakhapatnam	2020-06-28	24.38	74.09	3.42	26.06	16.53	11.99	0.52	12.72	30.

```
#check if null values filled correctly
india_air_data['AQI_Bucket'].isnull().sum()
```

```
0
```

3. Data Analysis

3.1 Average readings for cities against each pollutant

```
#average readings by city
avg_readings = india_air_data.groupby('City').mean().reset_index().round(2)
avg_readings
```

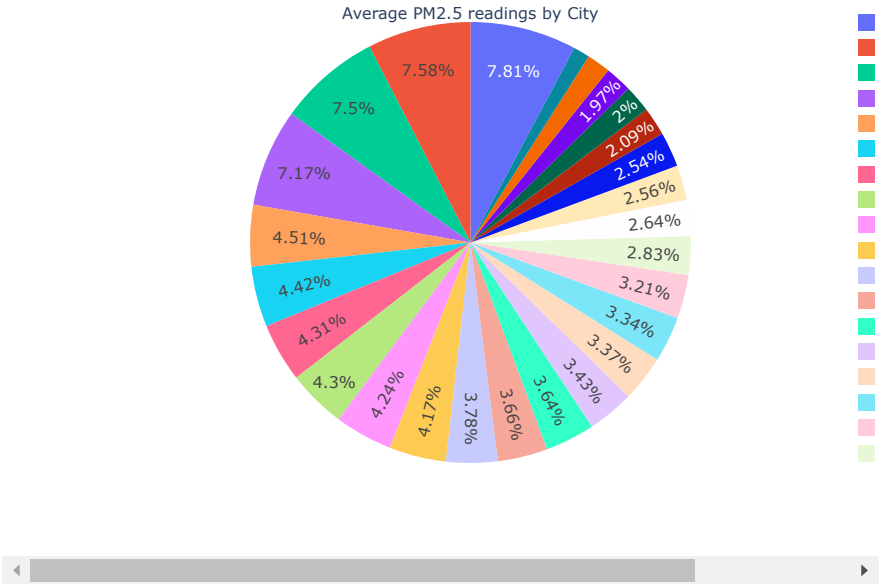
```
<ipython-input-20-28b6c2de3c12>:2: FutureWarning: The default value of numeric_only i
  avg_readings = india_air_data.groupby('City').mean().reset_index().round(2)
```

	City	PM2.5	PM10	NO	NO2	NOx	NH3	CO	SO2	O3	Be
0	Ahmedabad	67.73	117.41	20.96	49.81	42.92	23.48	16.15	42.28	37.56	
1	Aizawl	18.02	24.19	9.41	0.39	12.61	22.31	0.28	7.38	6.16	
2	Amaravati	39.61	78.78	5.20	22.54	16.36	12.65	0.79	14.28	37.91	
3	Amritsar	56.72	115.35	18.64	18.88	34.86	14.69	0.66	9.03	23.55	
4	Bengaluru	38.12	89.49	9.43	28.00	19.74	22.13	1.84	5.55	32.98	
5	Bhopal	50.60	119.29	7.37	31.26	22.74	19.21	0.92	13.11	59.06	
6	Brajrajnagar	64.73	123.09	17.37	19.52	25.99	33.48	1.87	10.76	20.47	
7	Chandigarh	42.43	85.66	10.59	11.83	15.29	30.54	0.63	10.16	20.05	
8	Chennai	51.42	109.82	9.34	17.07	17.93	54.26	1.08	8.00	32.49	
9	Coimbatore	29.95	39.44	8.83	28.78	30.90	14.01	0.96	8.65	28.88	
10	Delhi	117.15	228.41	38.96	50.76	58.57	41.91	1.98	15.83	50.62	
11	Ernakulam	25.99	50.06	23.05	12.16	24.49	20.71	1.64	3.46	34.49	
12	Gurugram	112.55	150.47	17.54	23.80	30.17	23.67	1.32	9.95	34.41	
13	Guwahati	63.69	116.60	20.04	13.60	44.28	11.09	0.74	14.66	25.09	
14	Hyderabad	48.21	96.57	7.95	28.39	19.48	17.51	0.59	9.24	33.62	
15	Jaipur	54.64	123.42	14.68	32.37	38.90	26.47	0.81	11.11	46.55	
16	Jorapokhar	66.41	142.24	12.53	13.78	32.31	12.19	1.36	27.91	32.94	
17	Kochi	31.43	67.34	71.10	15.45	68.41	9.54	1.30	17.60	3.82	
18	Kolkata	64.57	115.80	26.55	40.03	63.33	18.37	0.80	8.86	30.90	
19	Lucknow	107.57	118.13	15.26	33.19	24.10	26.40	2.13	10.10	36.88	
20	Mumbai	54.86	110.01	22.70	27.43	49.39	21.58	0.59	14.79	33.92	
21	Patna	113.82	119.02	30.28	36.51	44.46	22.95	1.59	21.29	36.84	
22	Shillong	38.39	59.10	4.09	7.66	4.33	6.75	0.45	8.00	29.15	
23	Talcher	62.61	156.55	28.07	17.34	31.97	13.92	1.91	25.60	20.36	
24	Thiruvananthapuram	29.53	55.09	3.85	9.91	8.53	6.93	0.97	5.90	34.68	
25	Visakhapatnam	50.19	107.92	13.48	35.73	25.42	12.80	0.78	12.95	37.05	

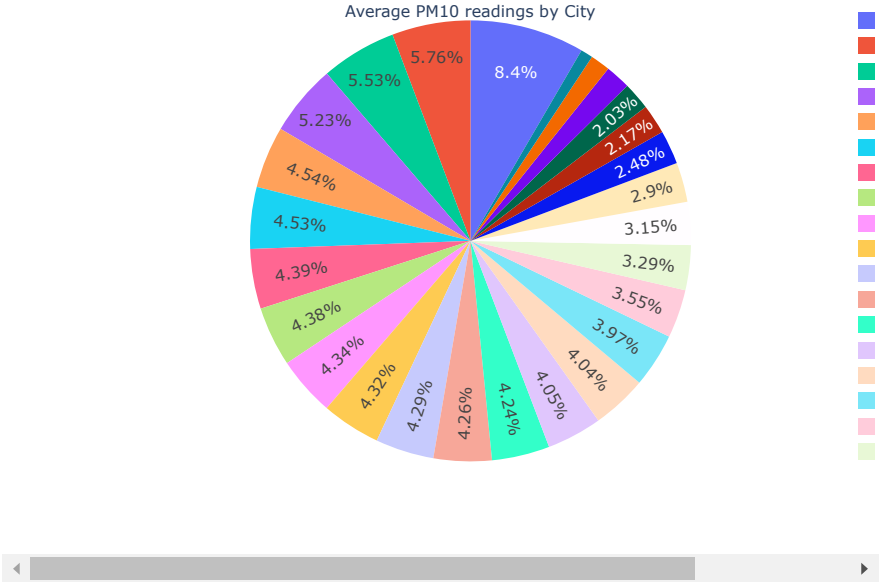
```
avg_readings.describe().T
```

	count	mean	std	min	25%	50%	75%	max
PM2.5	26.0	57.728462	27.508158	18.02	38.6950	53.030	64.6900	117.15
PM10	26.0	104.586538	42.134184	24.19	80.5000	112.680	119.2225	228.41
NO	26.0	17.971923	13.924665	3.85	9.3575	14.970	22.2650	71.10
NO2	26.0	24.084231	12.566666	0.39	14.1975	23.170	32.0925	50.76
NOx	26.0	31.056923	16.472431	4.33	19.5450	28.080	41.9150	68.41
NH3	26.0	20.751923	10.722767	6.75	13.0800	19.960	23.6225	54.26
CO	26.0	1.697692	2.994228	0.28	0.7500	0.965	1.6275	16.15
SO2	26.0	13.325000	8.294535	3.46	8.7025	10.460	14.7575	42.28
O3	26.0	31.555000	11.769350	3.82	26.0375	33.300	36.8700	59.06
Benzene	26.0	4.471154	6.613644	0.03	1.6525	3.280	4.0400	34.69
Toluene	26.0	8.425769	7.591385	0.00	4.0925	7.530	8.7000	35.37
Xylene	26.0	2.976154	1.154795	0.22	2.8075	3.070	3.0700	6.61
AQI	26.0	143.427692	66.441364	37.10	99.2850	130.585	159.8225	356.14

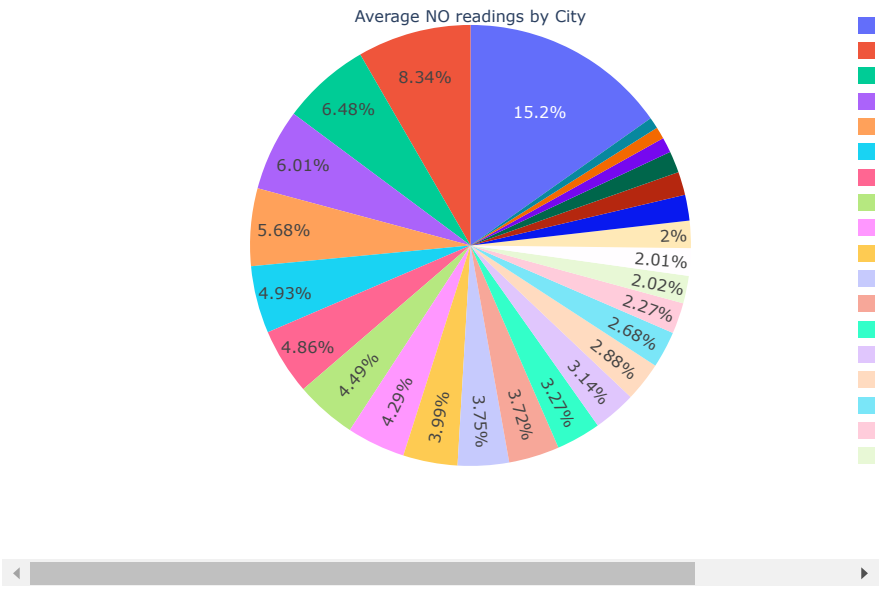
```
# avg readings for PM2.5 across all cities (pie chart)
trace1=go.Pie(labels=avg_readings['City'], values=avg_readings['PM2.5'].round(2),
              title='Average PM2.5 readings by City')
fig=go.Figure(trace1)
fig.update_traces(textposition='inside')
fig=fig.update_layout(uniformtext_minsize=12, uniformtext_mode='hide')
fig.show()
```



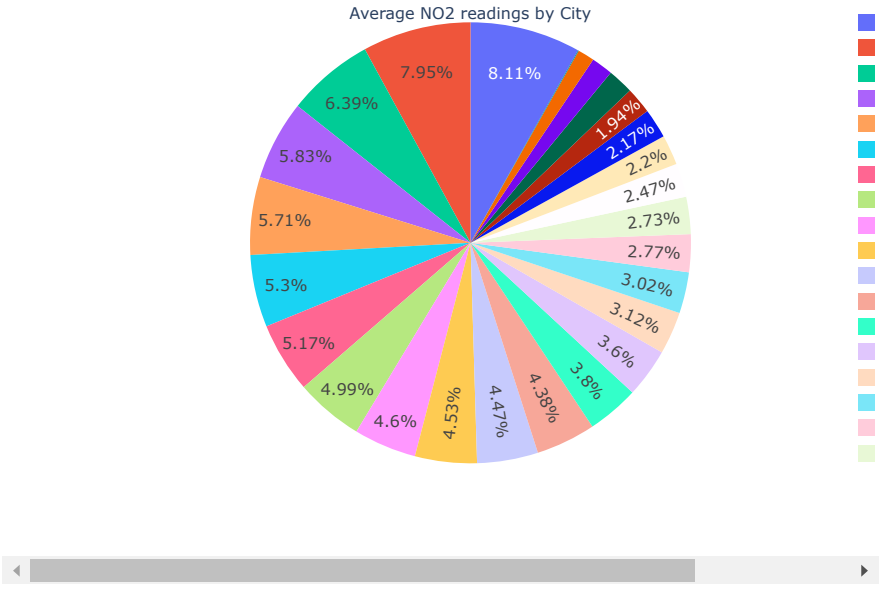

```
# avg readings for PM10 across all cities (pie chart)
trace2=go.Pie(labels=avg_readings['City'], values=avg_readings['PM10'].round(2),
              title='Average PM10 readings by City')
fig=go.Figure(trace2)
fig.update_traces(textposition='inside')
fig=fig.update_layout(uniformtext_minsize=12, uniformtext_mode='hide')
fig.show()
```



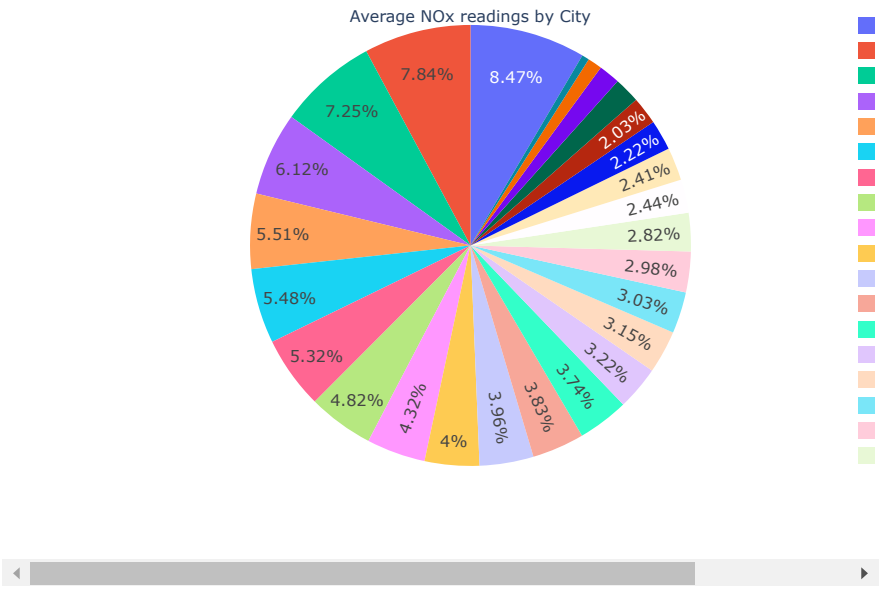

```
# avg readings for NO across all cities (pie chart)
trace3=go.Pie(labels=avg_readings['City'], values=avg_readings['NO'].round(2),
              title='Average NO readings by City')
fig=go.Figure(trace3)
fig.update_traces(textposition='inside')
fig=fig.update_layout(uniformtext_minsize=12, uniformtext_mode='hide')
fig.show()
```



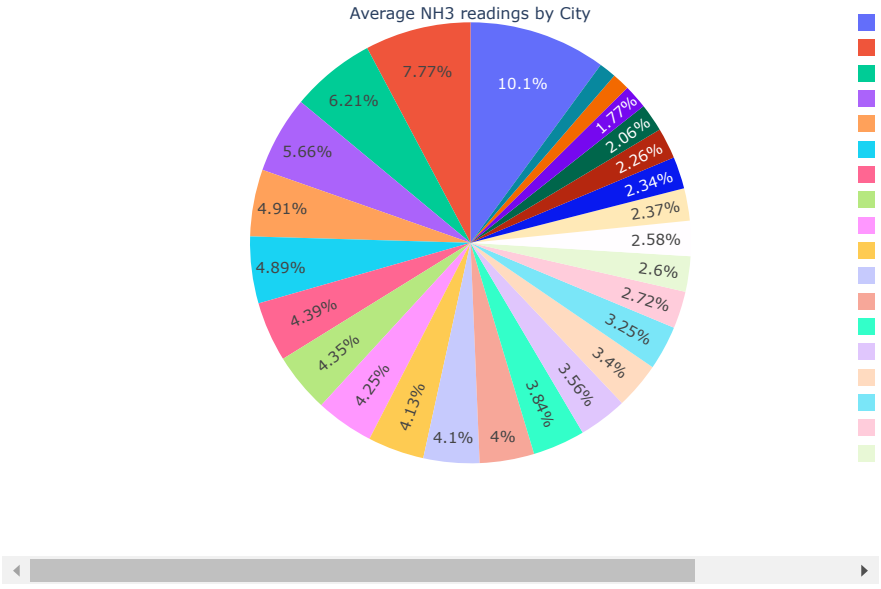

```
# avg readings for NO2 across all cities (pie chart)
trace4=go.Pie(labels=avg_readings['City'], values=avg_readings['NO2'],
              title='Average NO2 readings by City')
fig=go.Figure(trace4)
fig.update_traces(textposition='inside')
fig=fig.update_layout(uniformtext_minsize=12, uniformtext_mode='hide')
fig.show()
```



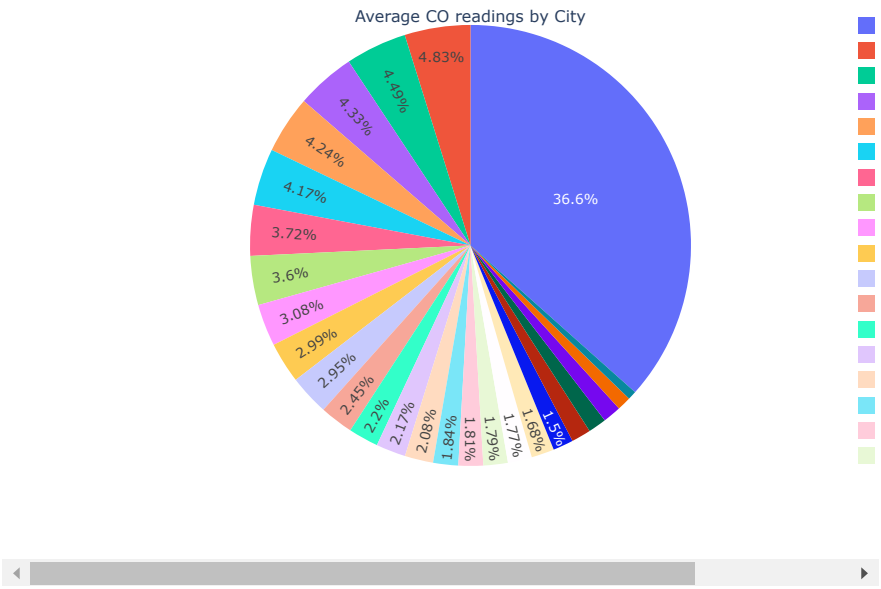

```
# avg readings for NOx across all cities (pie chart)
trace5=go.Pie(labels=avg_readings['City'], values=avg_readings['NOx'],
              title='Average NOx readings by City')
fig=go.Figure(trace5)
fig.update_traces(textposition='inside')
fig=fig.update_layout(uniformtext_minsize=12, uniformtext_mode='hide')
fig.show()
```



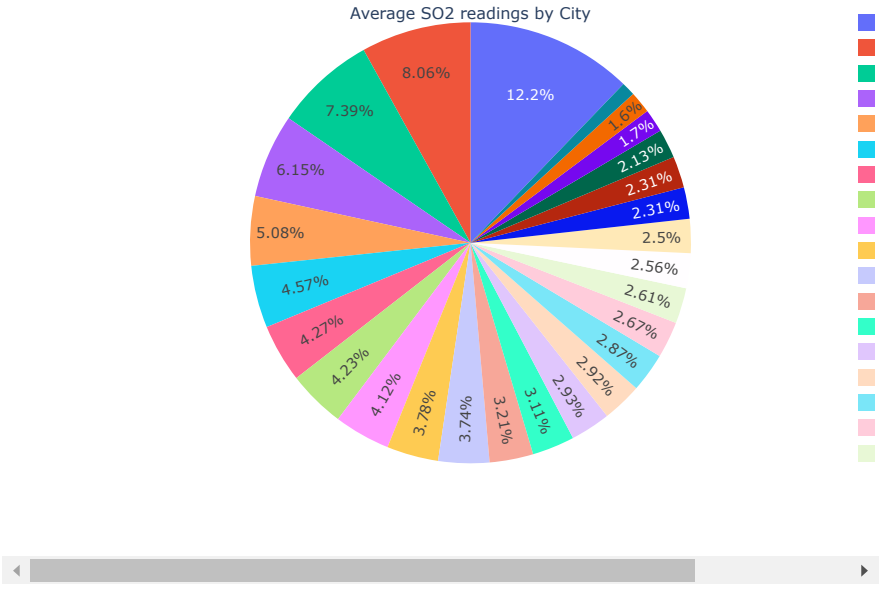

```
# avg readings for NH3 across all cities (pie chart)
trace6=go.Pie(labels=avg_readings['City'], values=avg_readings['NH3'],
              title='Average NH3 readings by City')
fig=go.Figure(trace6)
fig.update_traces(textposition='inside')
fig.update_layout(uniformtext_minsize=10, uniformtext_mode='hide')
fig.show()
```



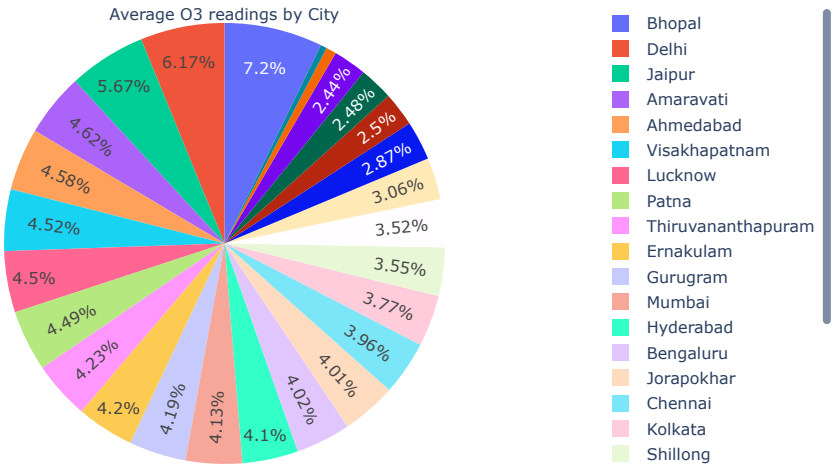

```
# avg readings for CO across all cities (pie chart)
trace7=go.Pie(labels=avg_readings['City'], values=avg_readings['CO'].round(2),
              title='Average CO readings by City')
fig=go.Figure(trace7)
fig.update_traces(textposition='inside')
fig.update_layout(uniformtext_minsize=10, uniformtext_mode='hide')
fig.show()
```



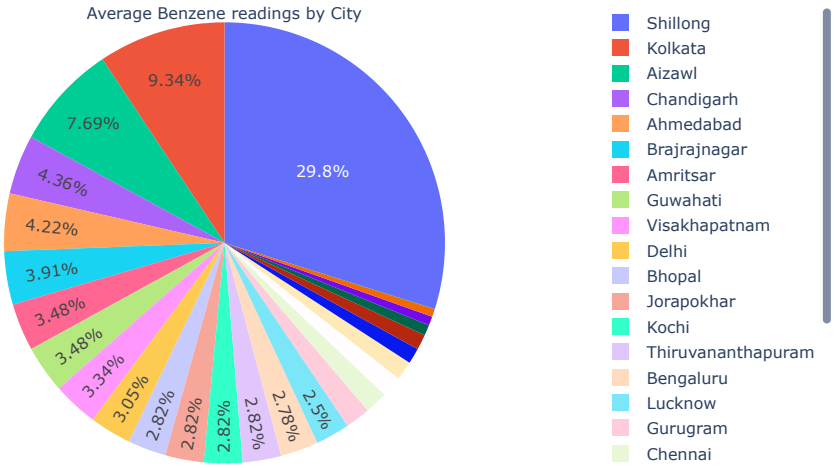

```
# avg readings for SO2 across all cities (pie chart)
trace8=go.Pie(labels=avg_readings['City'], values=avg_readings['SO2'],
              title='Average SO2 readings by City')
fig=go.Figure(trace8)
fig.update_traces(textposition='inside')
fig.update_layout(uniformtext_minsize=10, uniformtext_mode='hide')
fig.show()
```



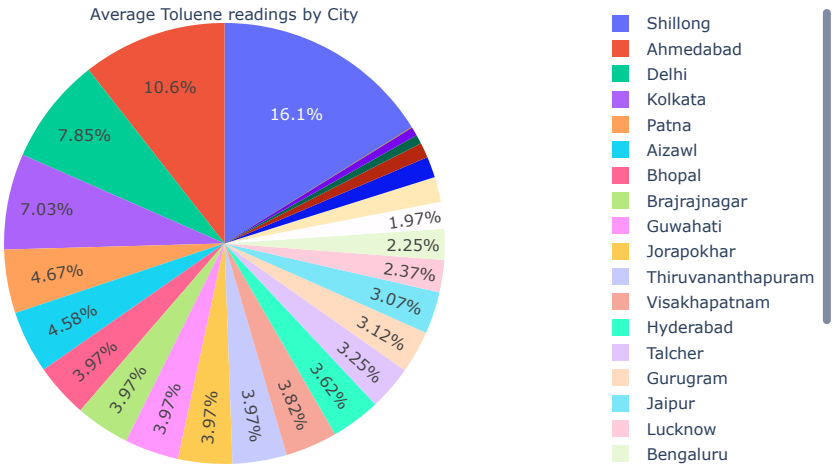

```
# avg readings for O3 across all cities (pie chart)
trace9=go.Pie(labels=avg_readings['City'], values=avg_readings['O3'],
              title='Average O3 readings by City')
fig=go.Figure(trace9)
fig.update_traces(textposition='inside')
fig.update_layout(uniformtext_minsize=10, uniformtext_mode='hide')
fig.show()
```



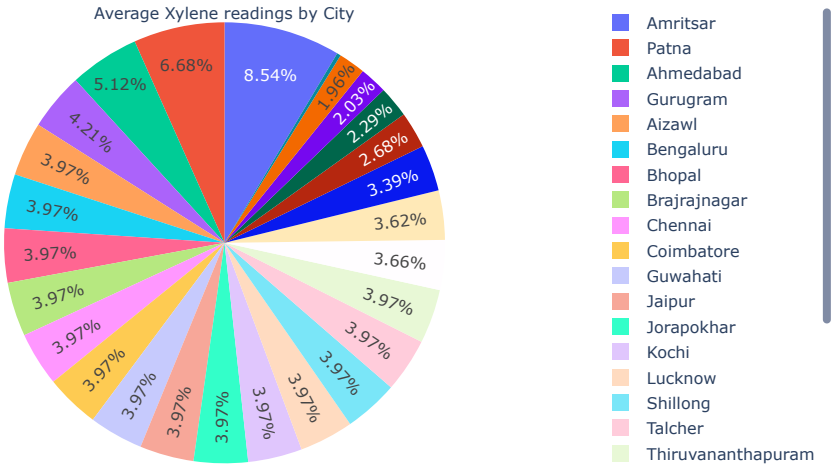

```
# avg readings for Benzene across all cities (pie chart)
trace10=go.Pie(labels=avg_readings['City'], values=avg_readings['Benzene'].round(2),
               title='Average Benzene readings by City')
fig=go.Figure(trace10)
fig.update_traces(textposition='inside')
fig.update_layout(uniformtext_minsize=12, uniformtext_mode='hide')
fig.show()
```



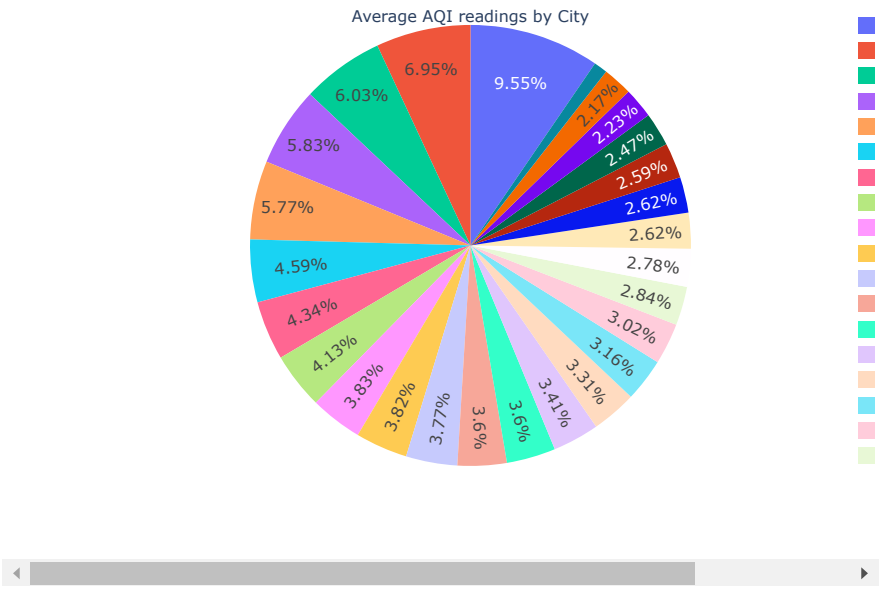

```
# avg readings for Toluene across all cities (pie chart)
trace11=go.Pie(labels=avg_readings['City'], values=avg_readings['Toluene'],
               title='Average Toluene readings by City')
fig=go.Figure(trace11)
fig.update_traces(textposition='inside')
fig=fig.update_layout(uniformtext_minsize=12, uniformtext_mode='hide')
fig.show()
```




```
# avg readings for Xylene across all cities (pie chart)
trace12=go.Pie(labels=avg_readings['City'], values=avg_readings['Xylene'],
               title='Average Xylene readings by City')
fig=go.Figure(trace12)
fig.update_traces(textposition='inside')
fig=fig.update_layout(uniformtext_minsize=12, uniformtext_mode='hide')
fig.show()
```




```
# avg readings for AQI across all cities (pie chart)
trace13=go.Pie(labels=avg_readings['City'], values=avg_readings['AQI'],
               title='Average AQI readings by City')
fig=go.Figure(trace13)
fig.update_traces(textposition='inside')
fig=fig.update_layout(uniformtext_minsize=12, uniformtext_mode='hide')
fig.show()
```



3.2 On average, which city scores highest and lowest for each pollutant?

```
avg_readings.round(2).head(5)
```

	City	PM2.5	PM10	NO	NO2	NOx	NH3	CO	SO2	O3	Benzene	Toluene	Xylene	AQI
0	Ahmedabad	67.73	117.41	20.96	49.81	42.92	23.48	16.15	42.28	37.56	4.90	23.16	3.96	356.14
1	Aizawl	18.02	24.19	9.41	0.39	12.61	22.31	0.28	7.38	6.16	8.94	10.04	3.07	37.10
2	Amaravati	39.61	78.78	5.20	22.54	16.36	12.65	0.79	14.28	37.91	0.76	2.42	1.57	103.53
3	Amritsar	56.72	115.35	18.64	18.88	34.86	14.69	0.66	9.03	23.55	4.04	4.02	6.61	123.54
4	Bengaluru	38.12	89.49	9.43	28.00	19.74	22.13	1.84	5.55	32.98	3.23	4.94	3.07	97.87

```
MaxVals=avg_readings[['PM2.5', 'PM10', 'NO', 'NO2', 'NOx', 'NH3', 'CO', 'SO2', 'O3',
       'Benzene', 'Toluene', 'Xylene', 'AQI']].dropna().idxmax()
MaxVals
```

```
PM2.5      10
PM10       10
NO         17
NO2        10
NOx        17
NH3         8
CO          0
SO2         0
O3          5
Benzene    22
Toluene    22
Xylene      3
AQI         0
dtype: int64
```

```
MaxCities=avg_readings.iloc[[10,10,17,10,17,8,0,0,5,22,22,3,0], [0]].reset_index().rename(columns={'City':'City, Max. Readings'})
MaxCities
```

	index	City, Max. Readings
0	10	Delhi
1	10	Delhi
2	17	Kochi
3	10	Delhi
4	17	Kochi
5	8	Chennai
6	0	Ahmedabad
7	0	Ahmedabad
8	5	Bhopal
9	22	Shillong
10	22	Shillong
11	3	Amritsar
12	0	Ahmedabad

```
MinVals=avg_readings[['PM2.5', 'PM10', 'NO', 'NO2', 'NOx', 'NH3', 'CO', 'SO2', 'O3',
       'Benzene', 'Toluene', 'Xylene', 'AQI']].dropna().idxmin()
MinVals
```

```
PM2.5       1
PM10        1
NO         24
NO2         1
NOx        22
NH3        22
CO          1
SO2        11
O3         17
Benzene     9
Toluene    17
Xylene     11
AQI         1
dtype: int64
```

```
MinCities=avg_readings.iloc[[20,19,22,1,16,0,22,11,11,5,5,1,1], [0]].reset_index().rename(columns={'City':'City, Min. Readings'})
MinCities
```

	index	City, Min. Readings
0	20	Mumbai
1	19	Lucknow
2	22	Shillong
3	1	Aizawl
4	16	Jorapokhar
5	0	Ahmedabad
6	22	Shillong
7	11	Ernakulam
8	11	Ernakulam
9	5	Bhopal
10	5	Bhopal
11	1	Aizawl
12	1	Aizawl

```
pollutants=avg_readings.columns.to_frame().drop(labels='City', axis=0).reset_index()
pollutants
```

	index	0
0	PM2.5	PM2.5
1	PM10	PM10
2	NO	NO
3	NO2	NO2
4	NOx	NOx
5	NH3	NH3
6	CO	CO
7	SO2	SO2
8	O3	O3
9	Benzene	Benzene
10	Toluene	Toluene
11	Xylene	Xylene
12	AQI	AQI

```
summ_max_min=pd.concat([pollutants, MaxCities, MinCities], axis=1).drop(columns=['index'])
summ_max_min.rename(columns={0:'Pollutant'} )
```

	Pollutant	City, Max. Readings	City, Min. Readings
0	PM2.5	Delhi	Mumbai
1	PM10	Delhi	Lucknow
2	NO	Kochi	Shillong
3	NO2	Delhi	Aizawl
4	NOx	Kochi	Jorapokhar
5	NH3	Chennai	Ahmedabad
6	CO	Ahmedabad	Shillong
7	SO2	Ahmedabad	Ernakulam
8	O3	Bhopal	Ernakulam
9	Benzene	Shillong	Bhopal
10	Toluene	Shillong	Bhopal
11	Xylene	Amritsar	Aizawl
12	AQI	Ahmedabad	Aizawl

From the data analysis it is clear that Delhi scores highest in the readings for PM2.5, PM10 and NO2. Kochi also scores highest against NO and NOx which is concerning given the low quantity of readings. With the exception of Mumbai and Lucknow, the cities with the lowest scores are Shillong, Ernakulam and Aizwal which are also the cities with the lowest quantity of readings between 2015 and 2020.

3.3 Pollutants Carbon Monoxide (CO) and Sulfur Dioxide (SO2)

According to the World Health Organisation(WHO), Sulfur dioxide (SO2), "is produced from the burning of fossil fuels (coal and oil) and the smelting of mineral ores that contain sulfur" and Carbon monoxide (CO), is "a toxic gas produced by the incomplete combustion of carbonaceous fuels such as wood, petrol, charcoal, natural gas and kerosene." Given the high toxicity of these pollutants in our atmosphere, there is an urgency to review our actions and innovate green energy options.

```
#avg co readings in order of high to low by city
avg_co_df=avg_readings[['City', 'CO']].sort_values('CO', ascending=False)
avg_co_df
```

	City	CO
0	Ahmedabad	16.15
19	Lucknow	2.13
10	Delhi	1.98
23	Talcher	1.91
6	Brajrajnagar	1.87
4	Bengaluru	1.84
11	Ernakulam	1.64
21	Patna	1.59
16	Jorapokhar	1.36
12	Gurugram	1.32
17	Kochi	1.30
8	Chennai	1.08
24	Thiruvananthapuram	0.97
9	Coimbatore	0.96
5	Bhopal	0.92
15	Jaipur	0.81
18	Kolkata	0.80
2	Amaravati	0.79
25	Visakhapatnam	0.78
13	Guwahati	0.74
3	Amritsar	0.66
7	Chandigarh	0.63
14	Hyderabad	0.59
20	Mumbai	0.59
22	Shillong	0.45
1	Aizawl	0.28

```
#avg so2 readings in order of high to low by city
avg_so2_df=avg_readings[['City', 'SO2']].sort_values('SO2', ascending=False)
avg_so2_df
```

	City	SO2
0	Ahmedabad	42.28
16	Jorapokhar	27.91
23	Talcher	25.60
21	Patna	21.29
17	Kochi	17.60
10	Delhi	15.83
20	Mumbai	14.79
13	Guwahati	14.66
2	Amaravati	14.28
5	Bhopal	13.11
25	Visakhapatnam	12.95
15	Jaipur	11.11
6	Brajrajnagar	10.76
7	Chandigarh	10.16
19	Lucknow	10.10
12	Gurugram	9.95
14	Hyderabad	9.24
3	Amritsar	9.03
18	Kolkata	8.86
9	Coimbatore	8.65
22	Shillong	8.00
8	Chennai	8.00
1	Aizawl	7.38
24	Thiruvananthapuram	5.90
4	Bengaluru	5.55
11	Ernakulam	3.46

```
city_annual_avg=india_air_data.copy()
city_annual_avg['Date'] = pd.to_datetime(city_annual_avg['Date'])
city_annual_avg=city_annual_avg.groupby(['City', city_annual_avg.Date.dt.year]).mean().reset_index().rename(columns={'Date':'Year'})
city_annual_avg
```

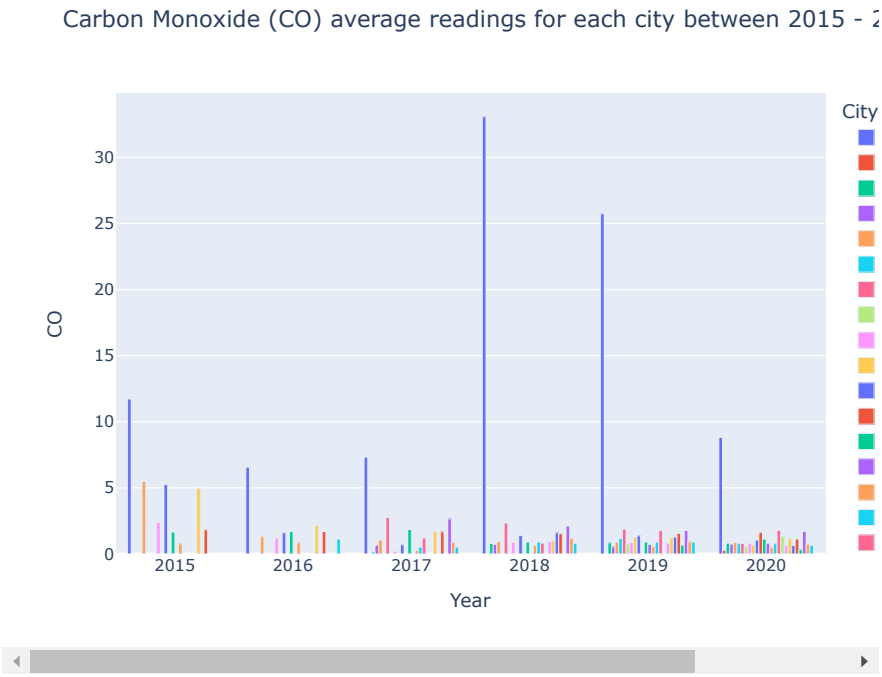
```
<ipython-input-44-467252bd2956>:3: FutureWarning:

The default value of numeric_only in DataFrameGroupBy.mean is deprecated. In a future
```

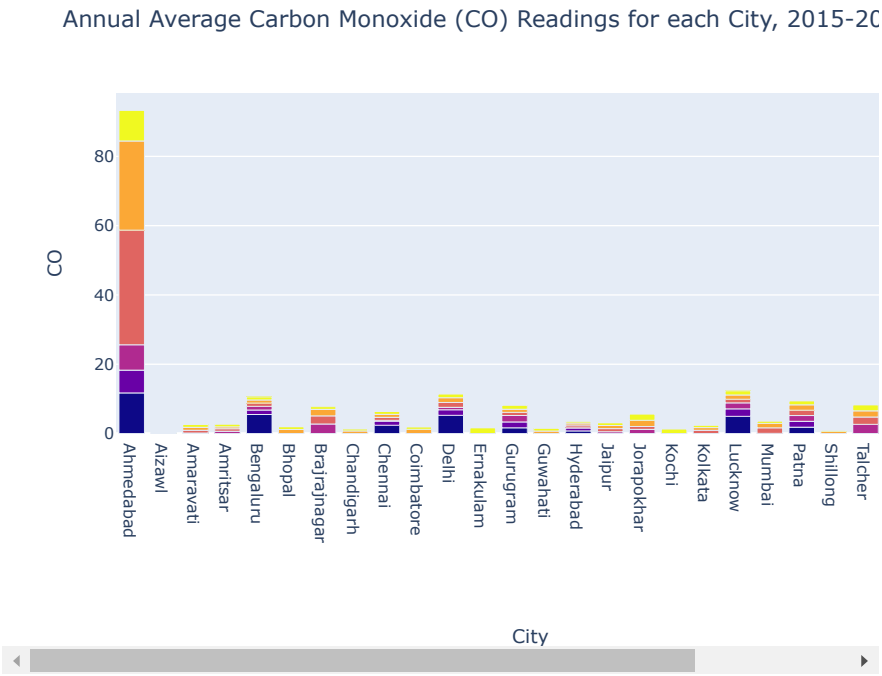
	City	Year	PM2.5	PM10	NO	NO2	NOx	
0	Ahmedabad	2015	76.479315	118.130000	14.244055	22.435068	33.122329	23.480
1	Ahmedabad	2016	65.759836	118.130000	16.654399	23.916175	30.713060	23.480
2	Ahmedabad	2017	71.711288	118.130000	19.921479	37.441507	38.070164	23.480
3	Ahmedabad	2018	74.649123	118.130000	33.158521	84.628575	60.277890	23.480
4	Ahmedabad	2019	62.206110	119.383945	25.993068	90.062959	62.681671	23.480
...	...	...	...	...	...	...	...	
98	Visakhapatnam	2016	45.718587	88.915489	16.692120	41.956957	32.827065	13.706
99	Visakhapatnam	2017	62.346740	113.471014	15.426027	30.628767	12.335781	15.574
100	Visakhapatnam	2018	51.365507	116.734904	12.708877	38.365178	30.652027	12.704
101	Visakhapatnam	2019	48.038466	115.358904	14.056795	37.482767	31.397890	10.685
102	Visakhapatnam	2020	32.397978	83.516011	6.756612	30.881585	21.708415	10.754

103 rows × 15 columns

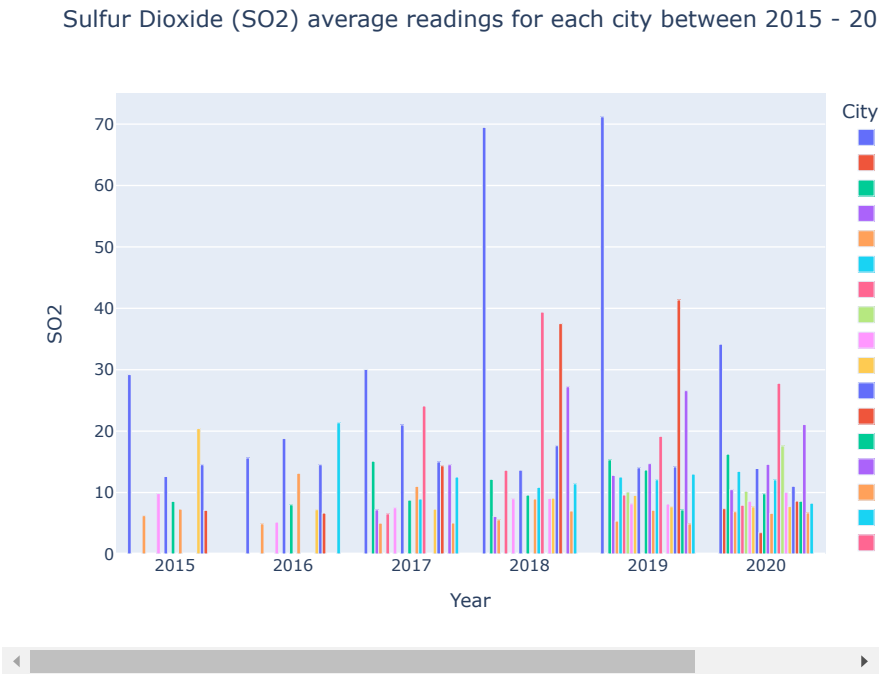
```
fig=px.bar(city_annual_avg, x='Year', y='CO', color='City', barmode='group',title='Carbon Monoxide (CO) average readings for each city b
fig.show()
```



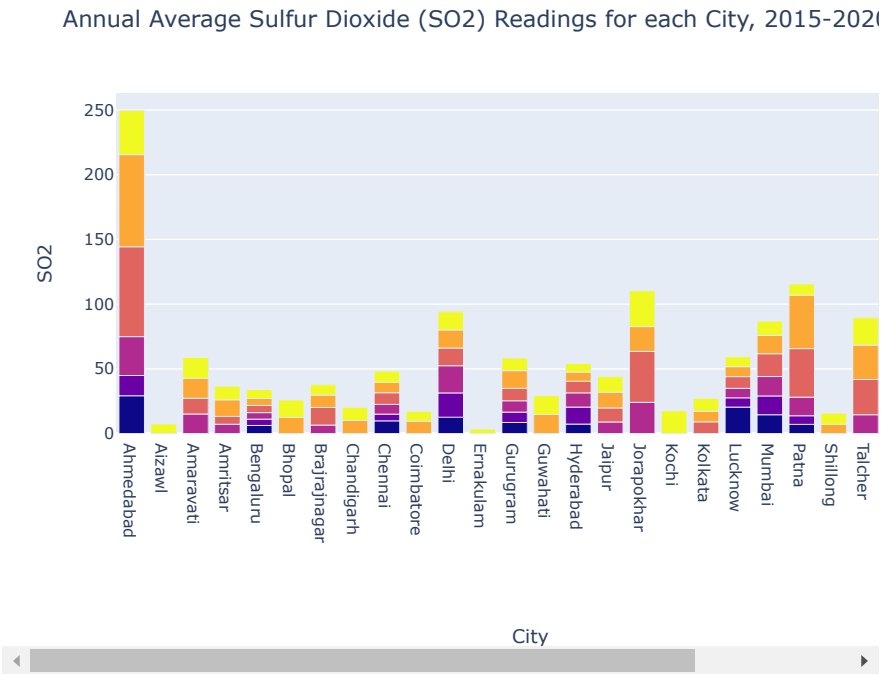

```
fig=px.bar(city_annual_avg, x='City', y='CO' , color='Year', title='Annual Average Carbon Monoxide (CO) Readings for each City, 2015-20
fig.show()
```




```
fig=px.bar(city_annual_avg, x='Year', y='SO2', color='City', barmode='group', title='Sulfur Dioxide (SO2) average readings for each city
fig.show()
```




```
fig=px.bar(city_annual_avg, x='City', y='SO2' , color='Year', title='Annual Average Sulfur Dioxide (SO2) Readings for each City, 2015-2(
fig.show()
```



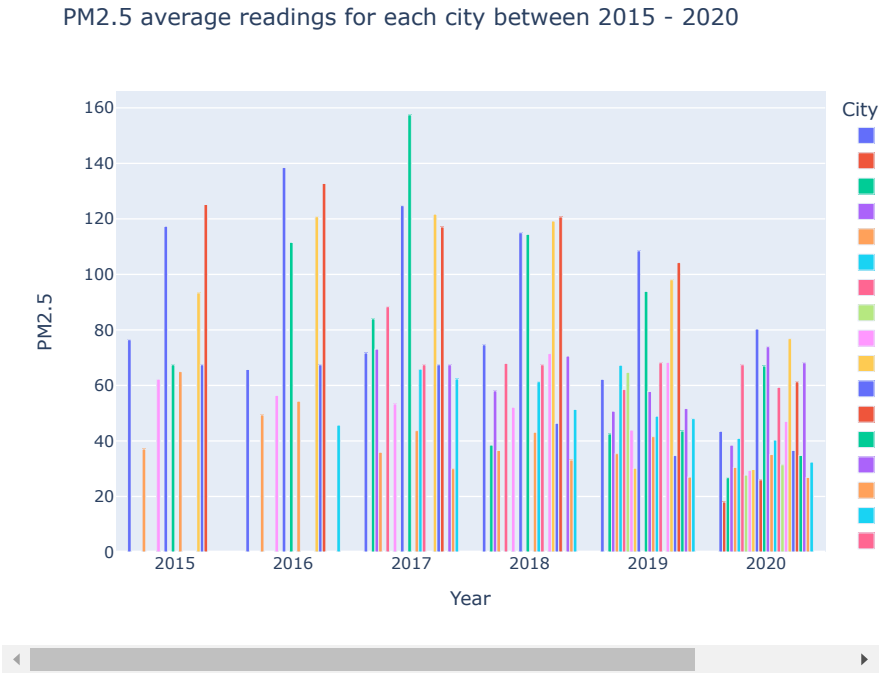
Ahmedabad scores highly on CO and SO2 between 2015 and 2020, which suggests that the city needs to consider enforcing some policies to reduce air pollution. In some cases such as Delhi and Lucknow it is evident that the local government has implemented some policies to improve the air quality as the readings have decreased over the 5-year period.

3.4 Pollutants PM2.5 and NO2

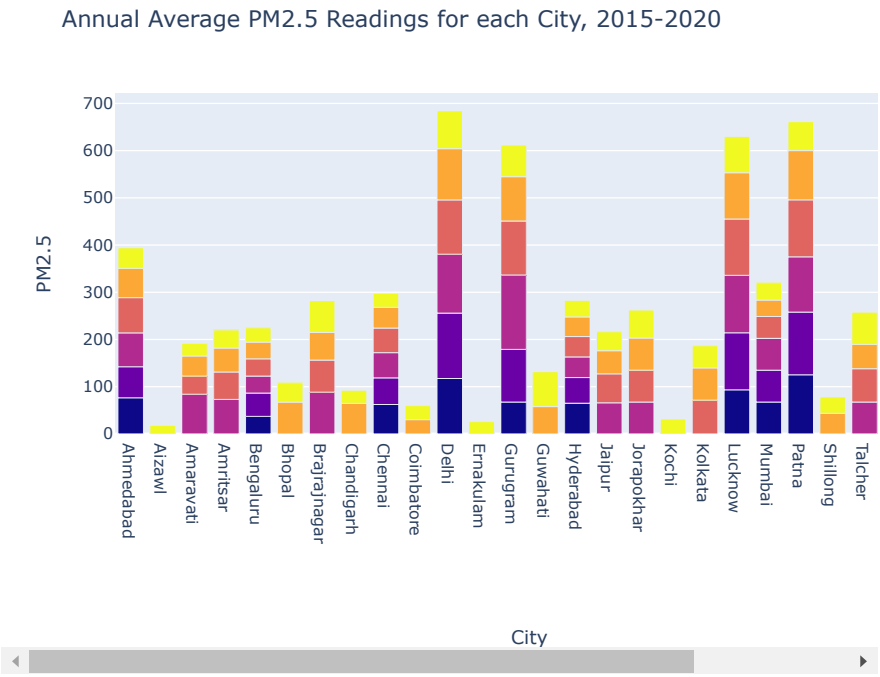
According to WHO, Particulate Matter (PM) is a, "common proxy indicator for air pollution. There is strong evidence for the negative health impacts associated with exposure to this pollutant. The major components of PM are sulfates, nitrates, ammonia, sodium chloride, black carbon, mineral dust and water." and Nitrogen Dioxide (NO2), "is a gas that is commonly released from the combustion of fuels in the transportation and industrial sectors."

```
trace5=go.Box(x=city_annual_avg['PM2.5'], name='PM2.5')
d5=[trace5]
layout=go.Layout(title='Range of PM2.5 readings')
fig=go.Figure(data=d5, layout=layout)
fig.show()
```


```
fig=px.bar(city_annual_avg, x='Year', y='PM2.5', color='City', barmode='group',title='PM2.5 average readings for each city between 2015
fig.show()
```

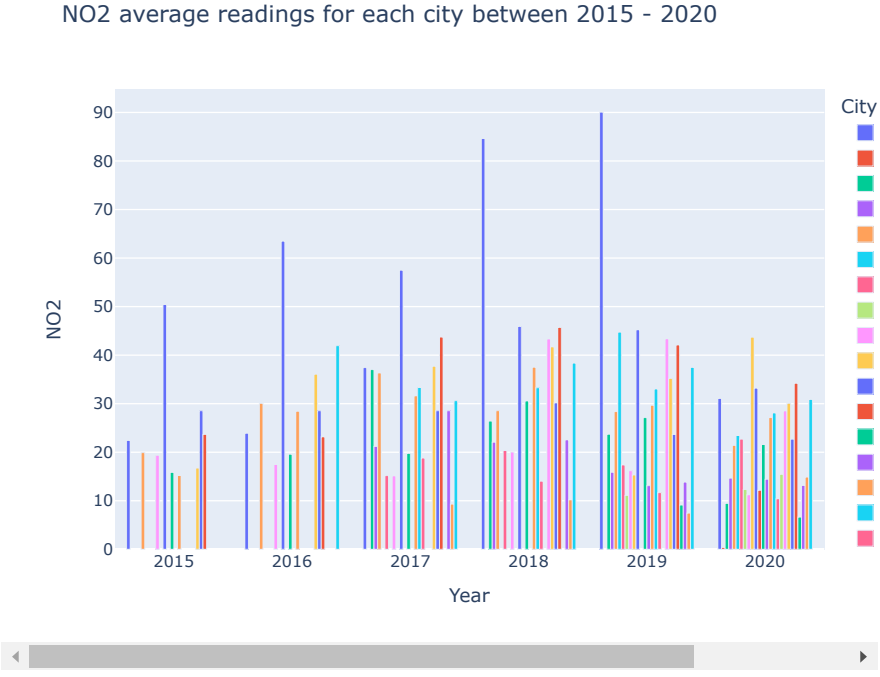



```
fig=px.bar(city_annual_avg, x='City', y='PM2.5' , color='Year', title='Annual Average PM2.5 Readings for each City, 2015-2020')
fig.show()
```

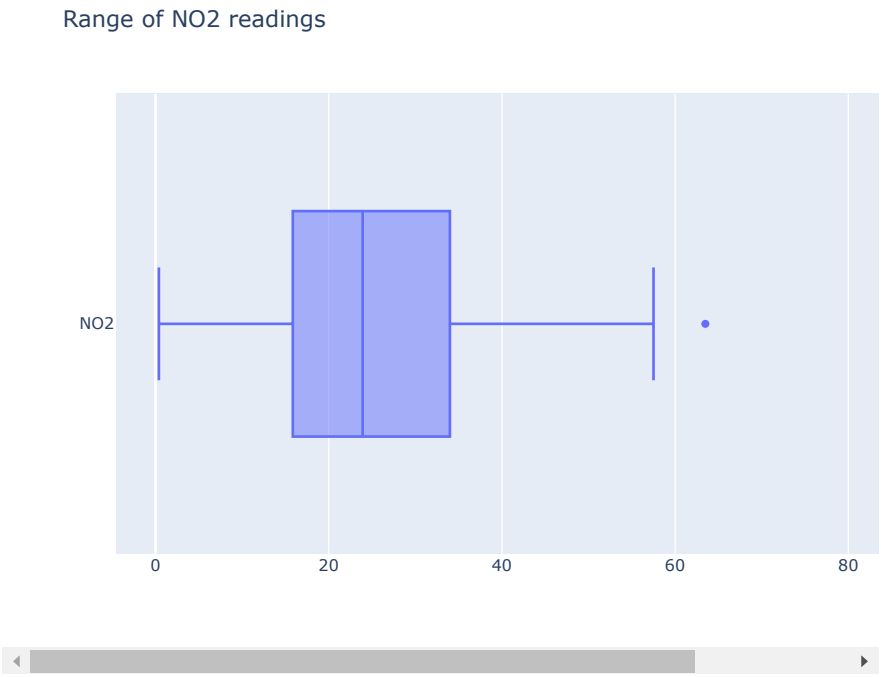


From the bar chart it is clear that the dataset for PM2.5 has expanded over the years to include more cities. Delhi, Lucknow and Patna have maintained a steady level of PM2.5. However, it is fair to say that most cities had a lower reading in 2020 -this could be due to the Pandemic when more people were working from home.

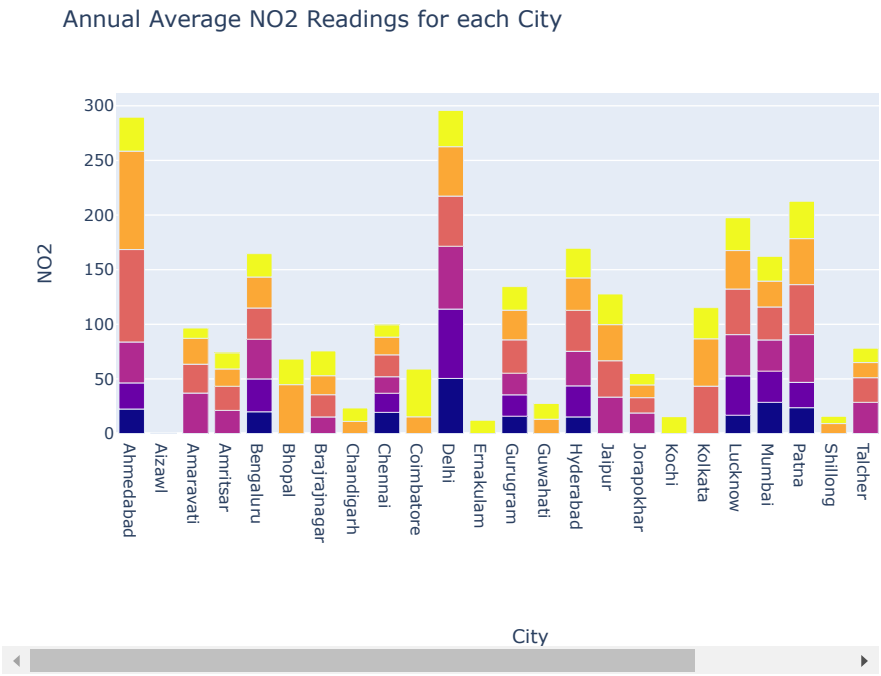
```
fig=px.bar(city_annual_avg, x='Year', y='NO2', color='City', barmode='group', title='NO2 average readings for each city between 2015 - 2
fig.show()
```




```
trace4=go.Box(x=city_annual_avg['NO2'], name='NO2')
d4=[trace4]
layout=go.Layout(title='Range of NO2 readings')
fig=go.Figure(data=d4, layout=layout)
fig.show()
```




```
fig=px.bar(city_annual_avg, x='City', y='NO2' , color='Year', title='Annual Average NO2 Readings for each City')
fig.show()
```



Similar to the PM2.5 readings, the NO2 dataset has also expanded over time to include more cities. Ahmedabad appears to have high readings for 2018 and 2019, whereas the readings for Delhi are steadily decreasing over the years. This could be due to measures implemented to try to reduce the pollution in the city.

3.5 Air Quality Index (AQI)

The AQI is measured accordingly:-

- Good (0-50)
- Satisfactory (51-100)
- Moderate (101-200)
- Poor (201-300)
- Very poor (301-400)
- Severe (401-500)

```
aqi_avg_df=city_annual_avg.copy().drop(columns=['PM2.5', 'PM10', 'NO', 'NO2', 'NOx', 'NH3', 'CO', 'SO2','O3', 'Benzene', 'Toluene', 'Xy
aqi_avg_df
```

	City	Year	AQI
0	Ahmedabad	2015	270.572384
1	Ahmedabad	2016	212.397650
2	Ahmedabad	2017	240.622356
3	Ahmedabad	2018	612.273096
4	Ahmedabad	2019	503.890356
...	...	...	...
98	Visakhapatnam	2016	109.749022
99	Visakhapatnam	2017	156.281534
100	Visakhapatnam	2018	127.482027
101	Visakhapatnam	2019	126.271342
102	Visakhapatnam	2020	86.919672

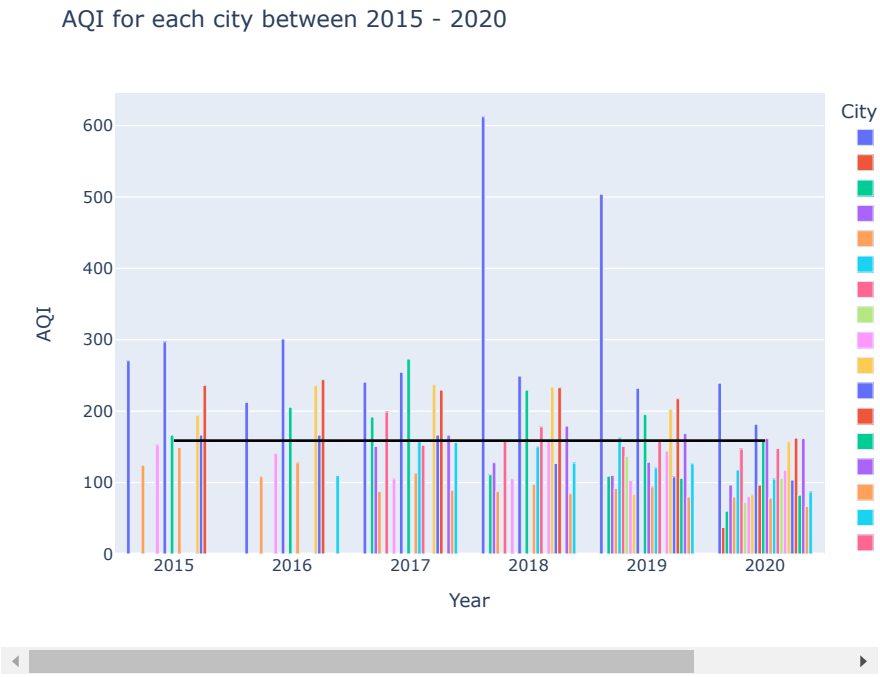
103 rows × 3 columns

```
aqi_avg_df['Average_AQI']=aqi_avg_df['AQI'].mean()
aqi_avg_df
```

	City	Year	AQI	Average_AQI
0	Ahmedabad	2015	270.572384	158.681593
1	Ahmedabad	2016	212.397650	158.681593
2	Ahmedabad	2017	240.622356	158.681593
3	Ahmedabad	2018	612.273096	158.681593
4	Ahmedabad	2019	503.890356	158.681593
...	...	...	...	...
98	Visakhapatnam	2016	109.749022	158.681593
99	Visakhapatnam	2017	156.281534	158.681593
100	Visakhapatnam	2018	127.482027	158.681593
101	Visakhapatnam	2019	126.271342	158.681593
102	Visakhapatnam	2020	86.919672	158.681593

103 rows × 4 columns

```
fig=px.bar(aqi_avg_df, x='Year', y='AQI', color='City', barmode='group', title='AQI for each city between 2015 - 2020')
l=px.line(aqi_avg_df, x='Year', y='Average_AQI').update_traces(line_color="black", name='Average AQI', showlegend=True)
fig.add_traces(l.data)
fig.show()
```



By plotting the average AQI it gives us a better idea on how the cities are performing relatively. Most cities are scoring below average which is considered to be "Moderate". The lower the AQI score, the better the rating. It is clear that Ahemedabad has consistently performed badly in the AQI ratings. The large drop between 2019 and 2020 is most likely due to the pandemic when more people were working from home amongst other factors. By increasing awareness of the problem we can help to drive practical solutions.

```
aqi_bucket_df=india_air_data[['City', 'Date', 'AQI_Bucket']]
aqi_bucket_df
```

	City	Date	AQI_Bucket
0	Ahmedabad	2015-01-01	Moderate
1	Ahmedabad	2015-01-02	Moderate
2	Ahmedabad	2015-01-03	Moderate
3	Ahmedabad	2015-01-04	Moderate
4	Ahmedabad	2015-01-05	Moderate
...	...	...	...
29526	Visakhapatnam	2020-06-27	Good
29527	Visakhapatnam	2020-06-28	Satisfactory
29528	Visakhapatnam	2020-06-29	Satisfactory
29529	Visakhapatnam	2020-06-30	Satisfactory
29530	Visakhapatnam	2020-07-01	Good

29531 rows × 3 columns

```
aqi_bucket_df_by_yr = aqi_bucket_df.groupby(['City', aqi_bucket_df.Date.dt.year, 'AQI_Bucket']).count().rename(columns={'Date':'Quantity
aqi_bucket_df_by_yr = aqi_bucket_df_by_yr.rename(columns={'Date':'Year'})
aqi_bucket_df_by_yr
```

	City	Year	AQI_Bucket	Quantity
0	Ahmedabad	2015	Good	1
1	Ahmedabad	2015	Moderate	174
2	Ahmedabad	2015	Poor	74

```
fig=px.bar(aqi_bucket_df_by_yr, x='City', y='Quantity' , color='AQI_Bucket', title='Annual Air Quality Index by City, 2015-2020')
fig.show()
```