In [1]:

```
# import the necessary packages
# this ipynb doesn't use all these libraries, but some may come in handy
# when running some spatial analysis further along the line
import geopandas as gpd
import pandas as pd
import matplotlib.pyplot as plt
import os
import re
import xlrd
import datetime
import csv
import seaborn as sbn
import statsmodels.api as sm
#import mapclassify
#import pysal
#import libpysal as lp
#import shapely.geometry
#import mapclassify as mc
#import numpy as np
#import esda
```

In [2]:

```
air = gpd.read_file("C:\\Users\\Ryan Siu\\Desktop\\FOURTH\\GGR473\\GROUP PROJECT\\Shapefiles\\CombinedAirData\\Combi

columns = ['cgndb_id', 'geographic']
air_subset = air[columns]

# Keeping only unique 'cgndb_id' values
unique_ids = air_subset.drop_duplicates(subset='cgndb_id')

print(unique_ids)
```

```
     cgndb_id         geographic
0       FAFFD             Barrie
180     FALIF           Brampton
347     FALJI          Brantford
463     FAMXK         Burlington
641     FAYJG         Grand Bend
803     FAZKI             Dorset
987     FBKKK             Guelph
1129    FBLJL           Hamilton
1244    FBLKS  Hamilton Mountain
1415    FCAEN             London
1527    FCCOT           Cornwall
1718    FCFUU          North Bay
1857    FCGKZ           Oakville
1995    FCIBD       Port Stanley
2130    FCKTB            Rexdale
2241    FCNJT           Sandwich
2399    FCTOV            Sudbury
2544    FCWFX        Thunder Bay
2709    FCWOV           Tiverton
2866    FCWYG            Toronto
2996    FDATE        Walkerville
3136    FDCHU           Westdale
3310    FDEGT            Windsor
3452    FDGED          Newmarket
3610    FDGEJ       Peterborough
3767    FDGEM             Milton
3853    FDJFN     St. Catharines
3979    FDMOP             Oshawa
4176    FDQBU        Scarborough
4317    FDQBX         North York
4432    FDSUS        Parry Sound
4593    FDZCP   Sault Ste. Marie
4743    FEAKO        Mississauga
5219    FEARV             Sarnia
5516    FEBWC          Kitchener
5913    FEUTC       Chatham-Kent
6700    FEUZB            Toronto
7235    FEVJR           Kingston
7670    FEVJS         Belleville
7907    FEVJV           Petawawa
8371    FEVNS           Hamilton
8937    FEVNT             Ottawa
```

In [3]:

```
# Group by 'station_id' and calculate the mean of 'air_quality'
average_air_quality = air.groupby('cgndb_id')['air_qualit'].mean().reset_index()

# List of specific station IDs you want to print
toronto_stations = ['FCKTB','FCWYG', 'FDQBU', 'FDQBX', 'FEUZB']

# Filter the average_air_quality DataFrame for specific station IDs
specific_stations = average_air_quality[average_air_quality['cgndb_id'].isin(toronto_stations)]

# This will give you a DataFrame with average air quality for specific station IDs
specific_stations
```

Out[3]:

	cgndb_id	air_qualit
14	FCKTB	2.678108
19	FCWYG	2.679000
28	FDQBU	2.760496
29	FDQBX	2.594087
36	FEUZB	2.665776

In [4]:

```
air['the_date'] = pd.to_datetime(air['the_date'])

# create a new column to store the day of the week
air['day'] = air['the_date'].dt.day_name()
air
```

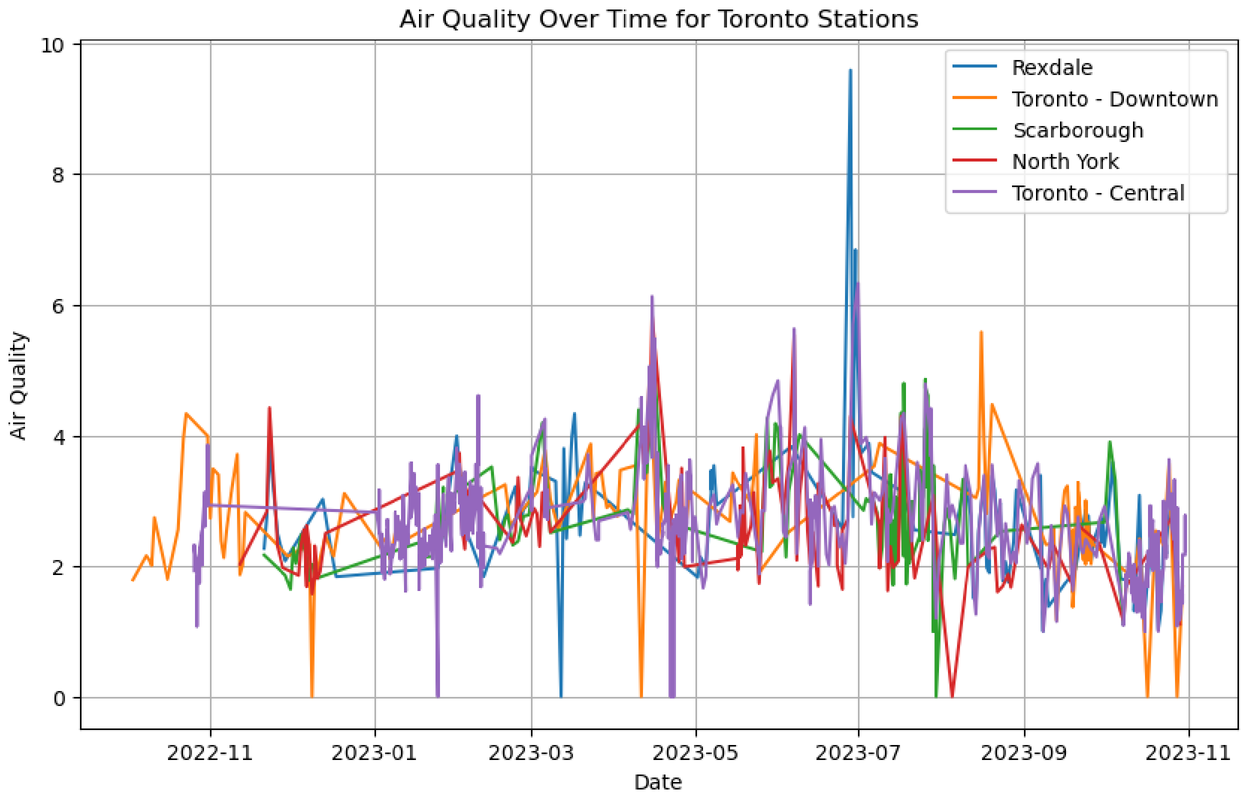
Out[4]:

	cgndb_id	geo_lat	geo_long	province_t	geo_locati	geo_decisi	concise_co	generic_ca	generic_te	geographic	geo_names_	the_c
0	FAFFD	44.37124	-79.67697	Ontario	Simcoe	1959-01-01	CITY	Populated Place	City	Barrie	8.4	20 10
1	FAFFD	44.37124	-79.67697	Ontario	Simcoe	1959-01-01	CITY	Populated Place	City	Barrie	8.4	20 10
2	FAFFD	44.37124	-79.67697	Ontario	Simcoe	1959-01-01	CITY	Populated Place	City	Barrie	8.4	20 02
3	FAFFD	44.37124	-79.67697	Ontario	Simcoe	1959-01-01	CITY	Populated Place	City	Barrie	8.4	20 10
4	FAFFD	44.37124	-79.67697	Ontario	Simcoe	1959-01-01	CITY	Populated Place	City	Barrie	8.4	20 10
...	...	...	...	...	...	...	...	...	...	...	...	
9499	FEVNT	45.33339	-75.58429	Ontario	Carleton; Russell	2001-01-01	CITY	Populated Place	City	Ottawa	8.4	20 10
9500	FEVNT	45.33339	-75.58429	Ontario	Carleton; Russell	2001-01-01	CITY	Populated Place	City	Ottawa	8.4	20 10
9501	FEVNT	45.33339	-75.58429	Ontario	Carleton; Russell	2001-01-01	CITY	Populated Place	City	Ottawa	8.4	20 02
9502	FEVNT	45.33339	-75.58429	Ontario	Carleton; Russell	2001-01-01	CITY	Populated Place	City	Ottawa	8.4	20 06
9503	FEVNT	45.33339	-75.58429	Ontario	Carleton; Russell	2001-01-01	CITY	Populated Place	City	Ottawa	8.4	20 07

9504 rows × 17 columns

In [5]:

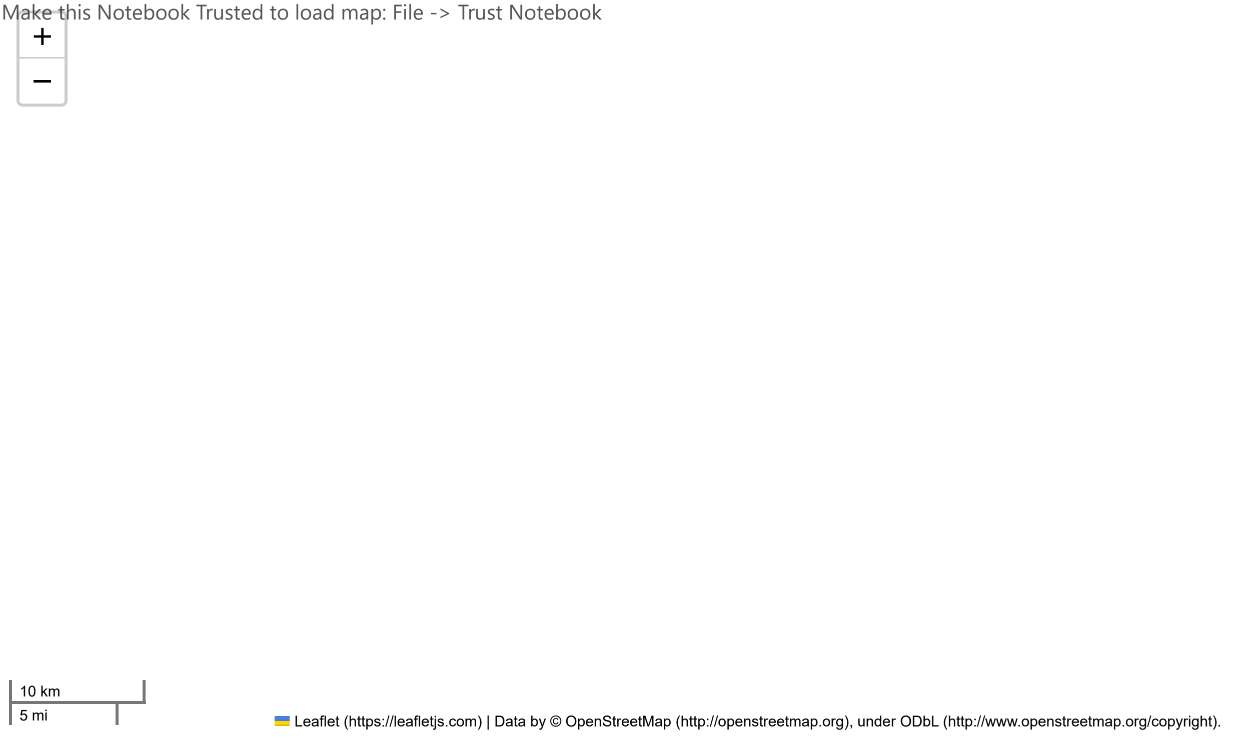
```python
# Filter data for specific stations
toronto_stations = ['FCKTB', 'FCWYG', 'FDQBU', 'FDQBX', 'FEUZB']
filtered_data = air[air['cgndb_id'].isin(toronto_stations)]

# Grouping by 'cgndb_id' in filtered data
grouped = filtered_data.groupby('cgndb_id')

# Define a dictionary to map cgndb_id to custom labels
custom_labels = {
    'FCKTB': 'Rexdale',
    'FCWYG': 'Toronto - Downtown',
    'FDQBU': 'Scarborough',
    'FDQBX': 'North York',
    'FEUZB': 'Toronto - Central'
}

# Plotting a line graph for each cgndb_id (specific stations) with custom labels
plt.figure(figsize=(10, 6))
for name, group in grouped:
    group = group.sort_values('the_date')
    plt.plot(group['the_date'], group['air_qualit'], label=custom_labels[name])

plt.xlabel('Date')
plt.ylabel('Air Quality')
plt.title('Air Quality Over Time for Toronto Stations')
plt.legend()
plt.grid(True)
plt.show()
```



```
In [6]: # # Define the file path where you want to save the shapefile
        # output_shapefile = 'airdata.shp'

        # # Save the GeoDataFrame as a shapefile
        # air.to_file(output_shapefile)
```

```
In [7]: air_buffer = gpd.read_file("C:\\Users\\Ryan Siu\\Desktop\\FOURTH\\GGR473\\GROUP PROJECT\\Shapefiles\\BufferData\\Air
        air_buffer.explore()
```

Out[7]: Make this Notebook Trusted to load map: File -> Trust Notebook


+
−
10 km
5 mi
Leaflet (https://leafletjs.com) | Data by © OpenStreetMap (http://openstreetmap.org), under ODbL (http://www.openstreetmap.org/copyright).

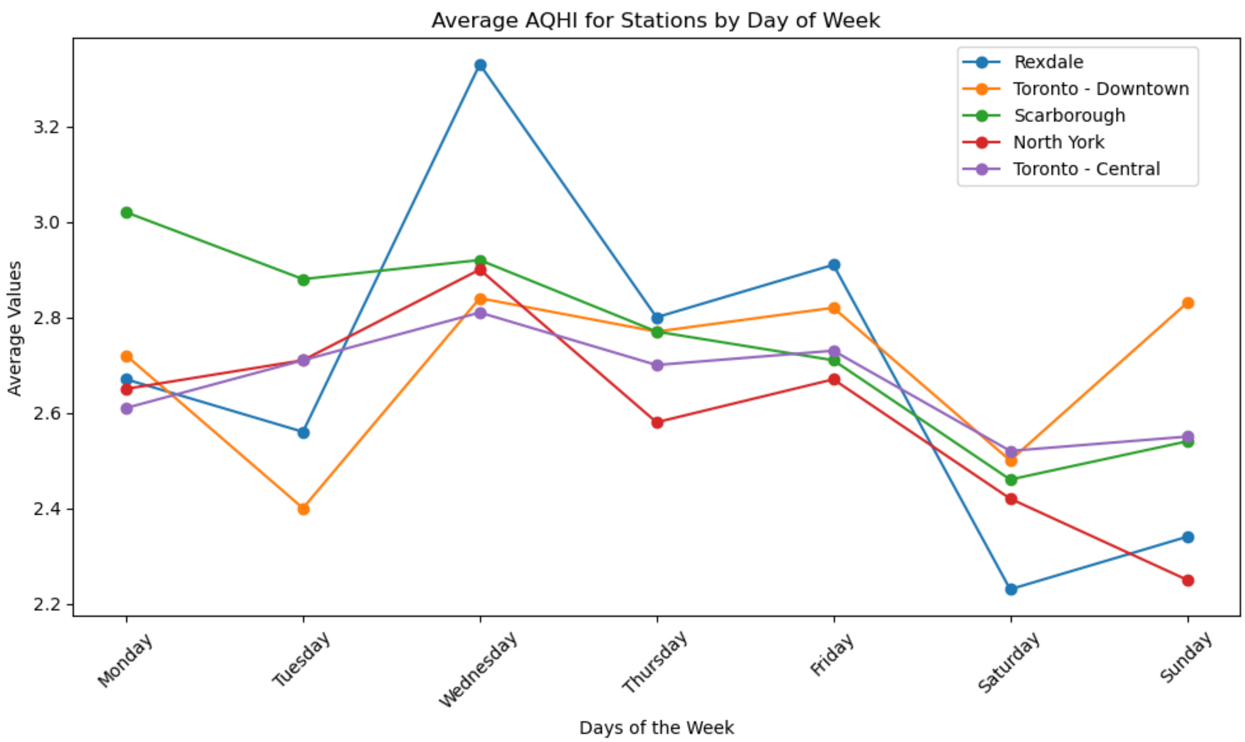
```
In [8]: air_buffer
```

Out[8]:

	buffer_id	avg_traffi	Shape_Leng	Shape_Area	dawn_avg	morn_avg	noon_avg	even_avg	mon_avg	tues_avg	wed_avg	thurs_av
0	FCKTB	14900.500000	6273.096929	3.121445e+06	3.05	2.21	2.58	2.94	2.67	2.56	3.33	2.8
1	FCWYG	15622.866667	6273.096987	3.121445e+06	2.79	2.12	2.56	3.08	2.72	2.40	2.84	2.7
2	FDQBU	17594.166667	6273.096961	3.121445e+06	3.16	2.54	2.54	2.67	3.02	2.88	2.92	2.7
3	FDQBX	21748.117647	6273.096943	3.121445e+06	2.80	2.11	2.47	2.72	2.65	2.71	2.90	2.5
4	FEUZB	21565.666667	6273.096961	3.121445e+06	2.80	2.55	2.63	2.69	2.61	2.71	2.81	2.7

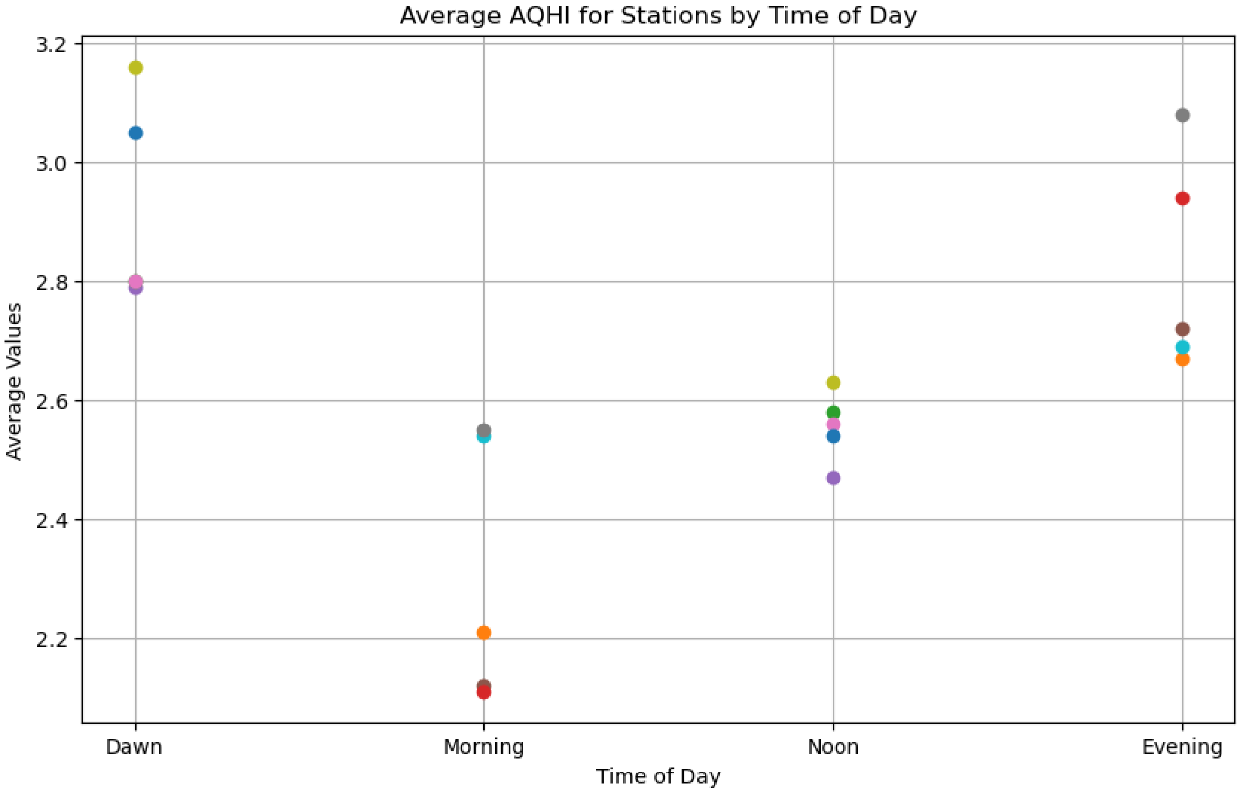
In [9]:

```
stations = ['FCKTB', 'FCWYG', 'FDQBU', 'FDQBX', 'FEUZB']
days = ['mon_avg', 'tues_avg', 'wed_avg', 'thurs_avg', 'fri_avg', 'sat_avg', 'sun_avg']
day_labels = ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday']

# Filter the data for the specified stations
filtered_df = air_buffer[air_buffer['buffer_id'].isin(stations)]

# Set up the plot
plt.figure(figsize=(10, 6))

# Plotting lines for each station
for station in stations:
    station_data = filtered_df[filtered_df['buffer_id'] == station]
    plt.plot(days, station_data[days].values.flatten(), marker='o', label=station)

plt.xlabel('Days of the Week')
plt.ylabel('Average Values')
plt.title('Average AQHI for Stations by Day of Week')

# Set custom x-axis labels
plt.xticks(days, day_labels, rotation=45)

# Custom legend labels
custom_labels = {
    'FCKTB': 'Rexdale',
    'FCWYG': 'Toronto - Downtown',
    'FDQBU': 'Scarborough',
    'FDQBX': 'North York',
    'FEUZB': 'Toronto - Central'
}

# Get current handles and labels
handles, labels = plt.gca().get_legend_handles_labels()

# Update legend labels using custom labels dictionary
updated_labels = [custom_labels[label] for label in labels]

# Set the updated labels
plt.legend(handles, updated_labels, bbox_to_anchor=(0.75, 1), loc='upper left')

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



In [10]:

```python
import matplotlib.pyplot as plt

# Your geodataframe
# Assuming your geodataframe is named 'gdf'

buffer_ids = ['FCKTB', 'FCWYG', 'FDQBU', 'FDQBX', 'FEUZB']

# Filter the geodataframe for the specified buffer_ids
filtered_gdf = air_buffer[air_buffer['buffer_id'].isin(buffer_ids)]

# Set up the time of day columns
time_of_day_columns = ['dawn_avg', 'morn_avg', 'noon_avg', 'even_avg']
time_of_day_labels = ['Dawn', 'Morning', 'Noon', 'Evening']  # Custom labels for x-axis

plt.figure(figsize=(10, 6))

for buffer_id in buffer_ids:
    buffer_data = filtered_gdf[filtered_gdf['buffer_id'] == buffer_id]
    for i, time_col in enumerate(time_of_day_columns):
        plt.plot(time_col, buffer_data[time_col], marker='o', label=f'{buffer_id} - {time_col}', linewidth=2)

plt.xlabel('Time of Day')
plt.ylabel('Average Values')
plt.title('Average AQHI for Stations by Time of Day')
#plt.legend()
plt.grid(True)

# Set custom x-axis labels
plt.xticks(range(len(time_of_day_columns)), time_of_day_labels)

plt.show()
```


Average AQHI for Stations by Time of Day
Average Values
3.2
3.0
2.8
2.6
2.4
2.2
Dawn
Morning
Noon
Evening
Time of Day