In [15]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [16]:

```
df = pd.read_csv("Air_Quality.csv")
df.head()
```

Out[16]:

	country	state	city	station	last_update	latitude	longituc
0	India	Arunachal_Pradesh	Naharlagun	Naharlagun, Naharlagun - APSPCB	06-04-2025 15:00:00	27.103358	93.67964
1	India	Assam	Byrnihat	Central Academy for SFS, Byrnihat - PCBA	06-04-2025 15:00:00	26.071318	91.87488
2	India	Assam	Byrnihat	Central Academy for SFS, Byrnihat - PCBA	06-04-2025 15:00:00	26.071318	91.87488
3	India	Assam	Guwahati	IITG, Guwahati - PCBA	06-04-2025 15:00:00	26.202864	91.70046
4	India	Assam	Guwahati	LGBI Airport, Guwahati - PCBA	06-04-2025 15:00:00	26.108870	91.58954

In [17]:

```
# Display the first few rows
print(" ◆  First 5 Rows:")
display(df.head())
```

```
◆  First 5 Rows:
```

	country	state	city	station	last_update	latitude	longitude
0	India	Arunachal_Pradesh	Naharlagun	Naharlagun, Naharlagun - APSPCB	06-04-2025 15:00:00	27.103358	93.679645
1	India	Assam	Byrnihat	Central Academy for SFS, Byrnihat - PCBA	06-04-2025 15:00:00	26.071318	91.874880
2	India	Assam	Byrnihat	Central Academy for SFS, Byrnihat - PCBA	06-04-2025 15:00:00	26.071318	91.874880
3	India	Assam	Guwahati	IITG, Guwahati - PCBA	06-04-2025 15:00:00	26.202864	91.700464
4	India	Assam	Guwahati	LGBI Airport, Guwahati - PCBA	06-04-2025 15:00:00	26.108870	91.589544

In [18]:

```
# Check data structure
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3187 entries, 0 to 3186
Data columns (total 11 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   country        3187 non-null   object 
 1   state          3187 non-null   object 
 2   city           3187 non-null   object 
 3   station        3187 non-null   object 
 4   last_update    3187 non-null   object 
 5   latitude       3187 non-null   float64
 6   longitude      3187 non-null   float64
 7   pollutant_id   3187 non-null   object 
 8   pollutant_min  3046 non-null   float64
 9   pollutant_max  3046 non-null   float64
 10  pollutant_avg  3046 non-null   float64
dtypes: float64(5), object(6)
memory usage: 274.0+ KB
```

In [19]:

```
# Convert date column
df['last_update'] = pd.to_datetime(df['last_update'], errors='coerce')
```

In [20]:

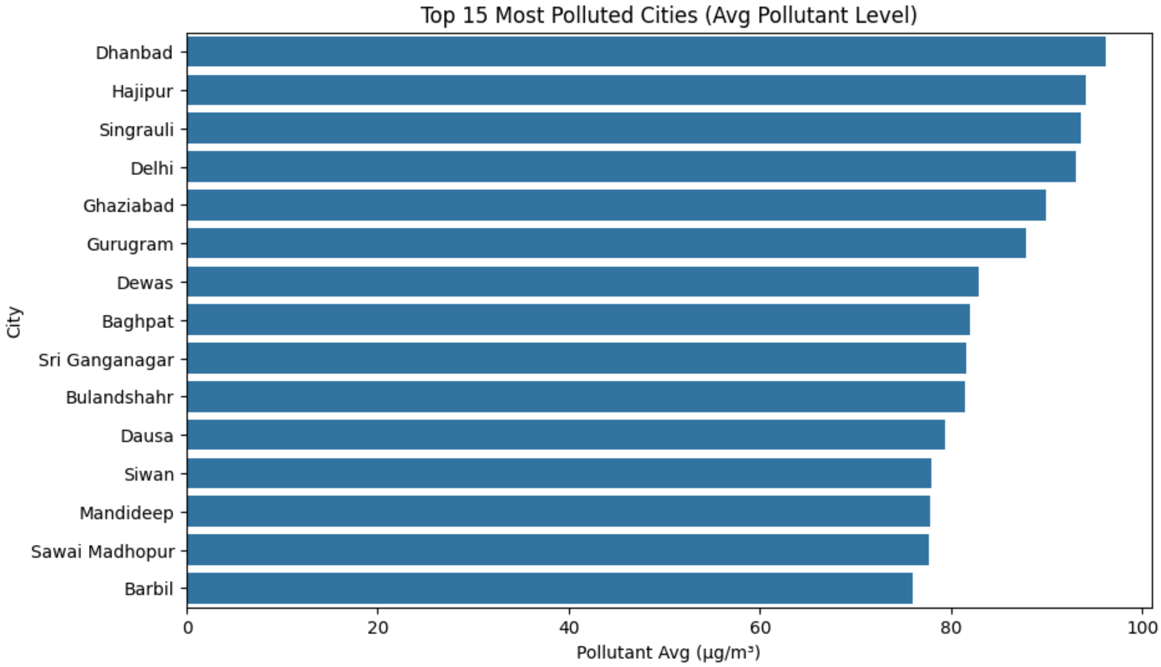
```
# Check missing values
df.isnull().sum()
```

Out[20]:

```
country            0
state              0
city               0
station            0
last_update        0
latitude           0
longitude          0
pollutant_id       0
pollutant_min    141
pollutant_max    141
pollutant_avg    141
dtype: int64
```

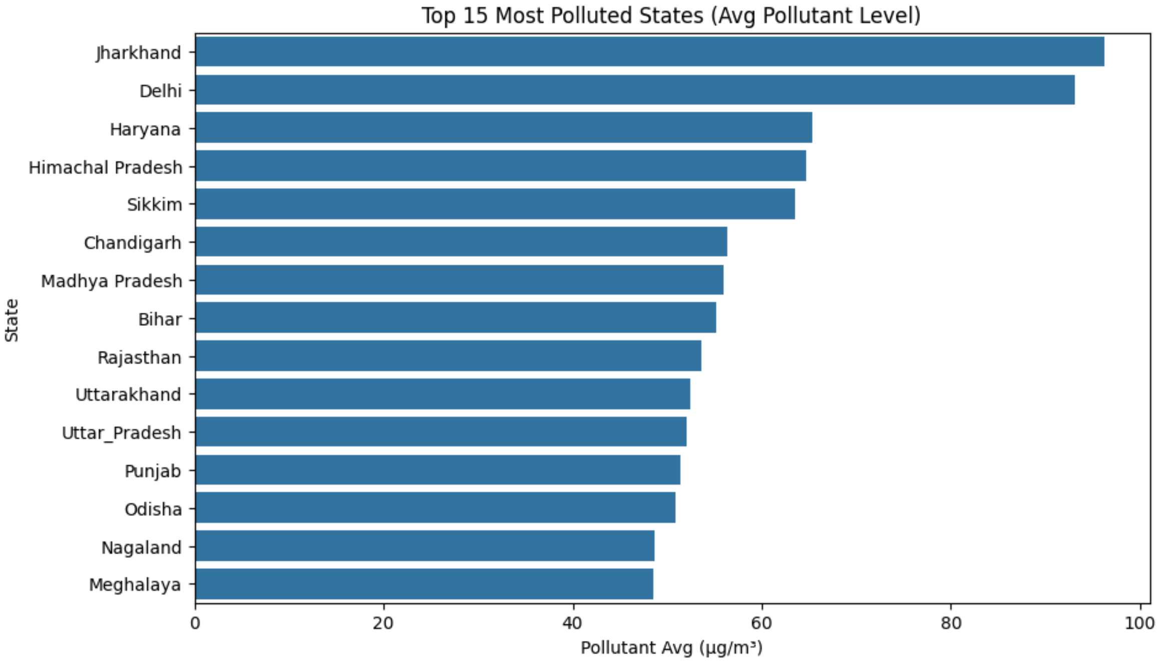
In [21]:

```
# Average pollution per city
city_avg = df.groupby('city')['pollutant_avg'].mean().sort_values(ascending=Fals

plt.figure(figsize=(10,6))
sns.barplot(x=city_avg.values, y=city_avg.index)
plt.title("Top 15 Most Polluted Cities (Avg Pollutant Level)")
plt.xlabel("Pollutant Avg (µg/m³)")
plt.ylabel("City")
plt.show()
```



In [22]:

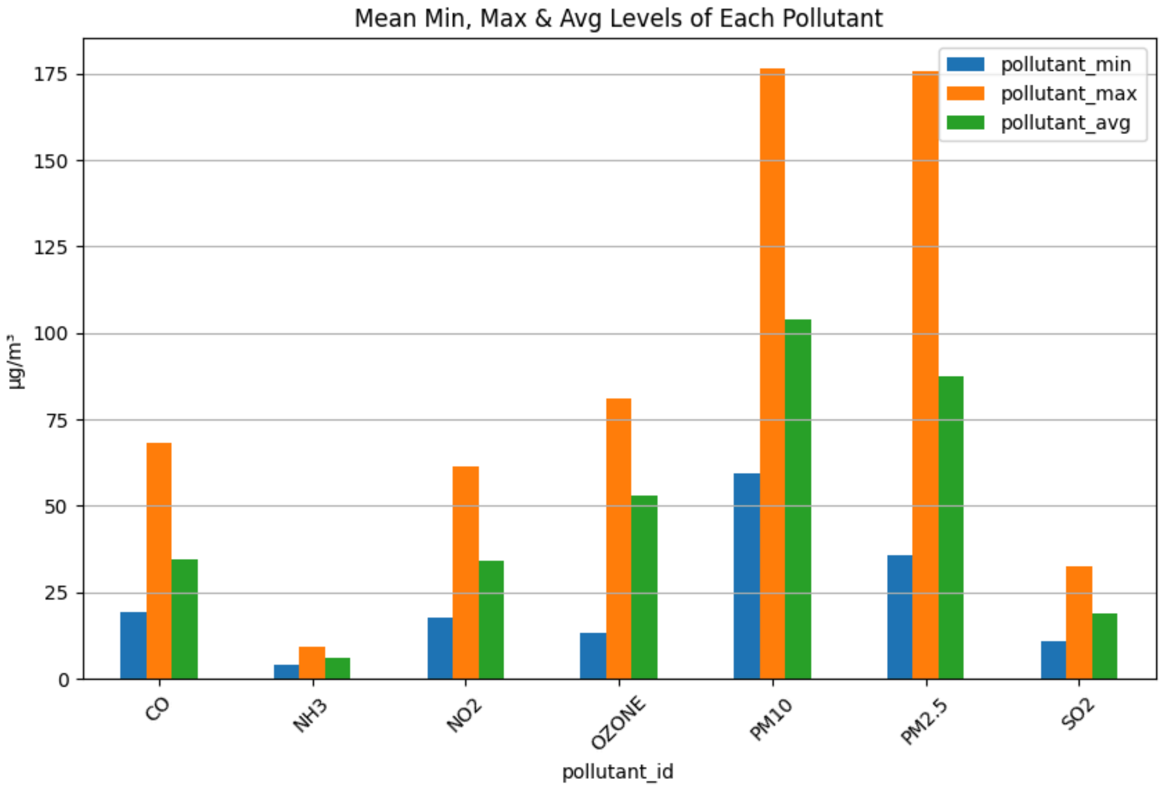
```
# Average pollution per state
state_avg = df.groupby('state')['pollutant_avg'].mean().sort_values(ascending=Fa

plt.figure(figsize=(10,6))
sns.barplot(x=state_avg.values, y=state_avg.index)
plt.title("Top 15 Most Polluted States (Avg Pollutant Level)")
plt.xlabel("Pollutant Avg (µg/m³)")
plt.ylabel("State")
plt.show()
```



In [23]:

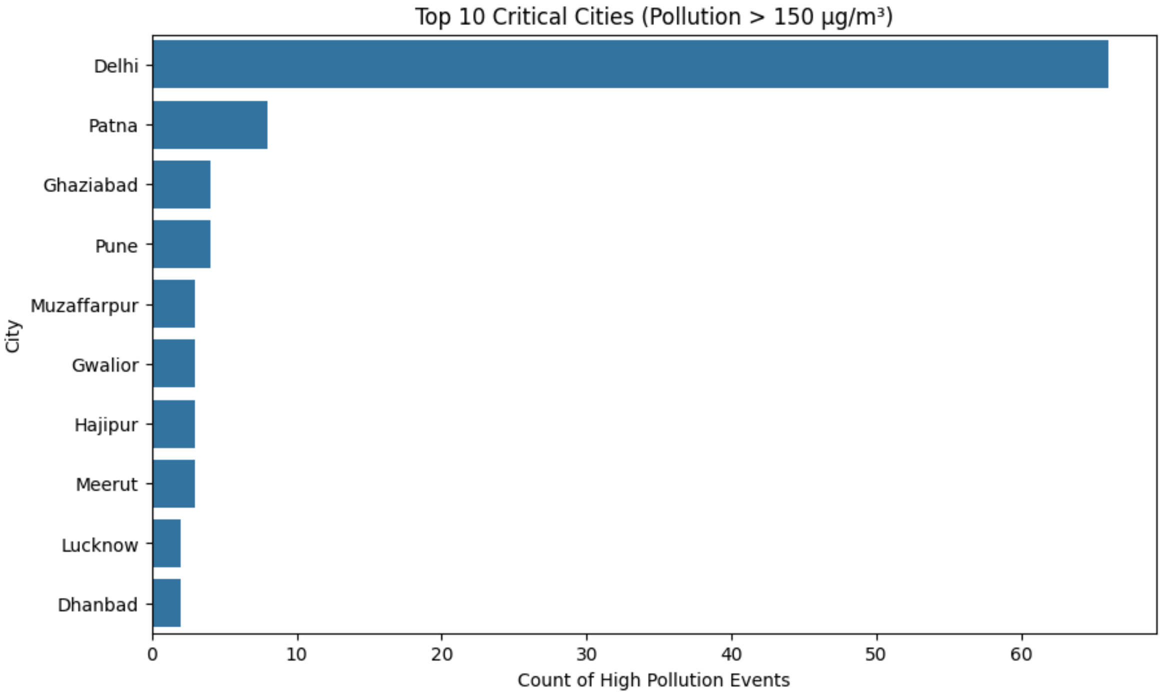
```
# Group by pollutant type
pollutant_stats = df.groupby('pollutant_id')[['pollutant_min', 'pollutant_max',
print(pollutant_stats)

# Visualize
pollutant_stats.plot(kind='bar', figsize=(10,6))
plt.title("Mean Min, Max & Avg Levels of Each Pollutant")
plt.ylabel("µg/m³")
plt.xticks(rotation=45)
plt.grid(axis='y')
plt.show()
```

```
              pollutant_min  pollutant_max  pollutant_avg
pollutant_id
CO                19.401330      68.334812      34.665188
NH3                4.137500       9.215000       6.020000
NO2               17.563063      61.461712      34.006757
OZONE             13.402299      80.928736      52.891954
PM10              59.358277     176.410431     103.795918
PM2.5             35.597315     175.648770      87.581655
SO2               11.016355      32.369159      18.738318
```



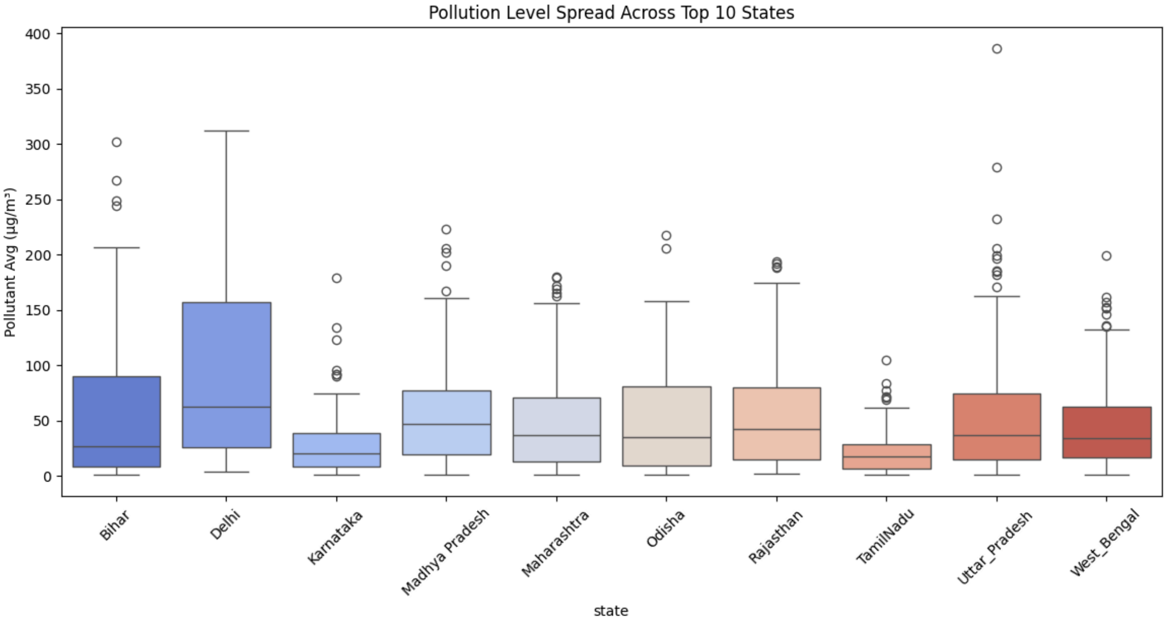
In [24]:

```python
# Define critical threshold
threshold = 150

# Cities with most high-pollution records
high_pollution = df[df['pollutant_avg'] > threshold]
critical_cities = high_pollution['city'].value_counts().head(10)

plt.figure(figsize=(10,6))
sns.barplot(x=critical_cities.values, y=critical_cities.index)
plt.title("Top 10 Critical Cities (Pollution > 150 µg/m³)")
plt.xlabel("Count of High Pollution Events")
plt.ylabel("City")
plt.show()
```

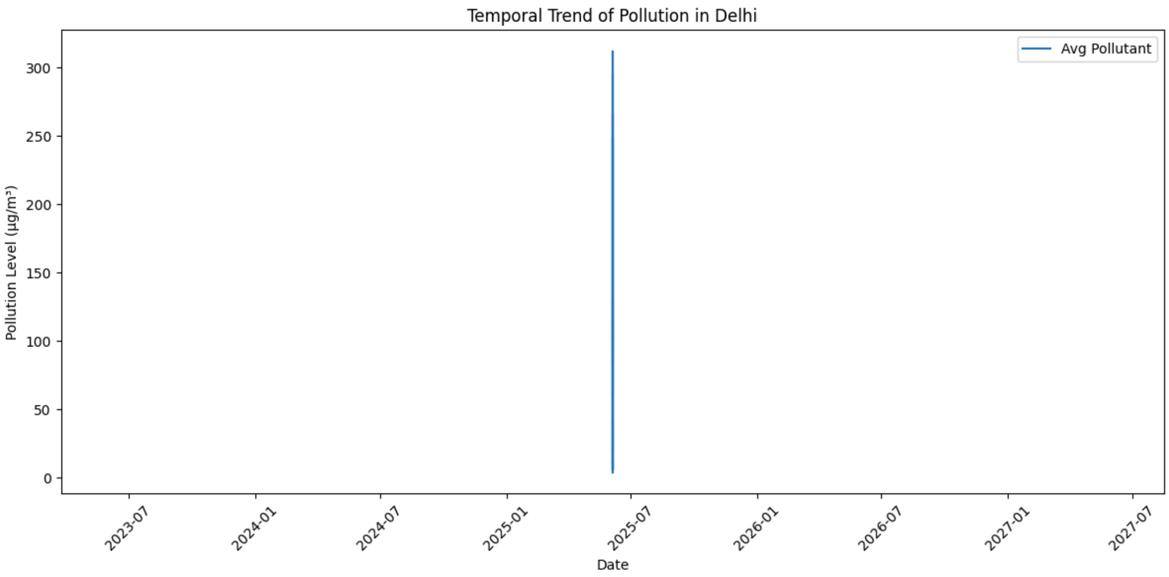


```
In [25]: # Boxplot to compare states
         plt.figure(figsize=(14,6))
         top_states = df['state'].value_counts().index[:10]
         sns.boxplot(data=df[df['state'].isin(top_states)],
                     x='state', y='pollutant_avg', palette='coolwarm')
         plt.title("Pollution Level Spread Across Top 10 States")
         plt.ylabel("Pollutant Avg (µg/m³)")
         plt.xticks(rotation=45)
         plt.show()
```

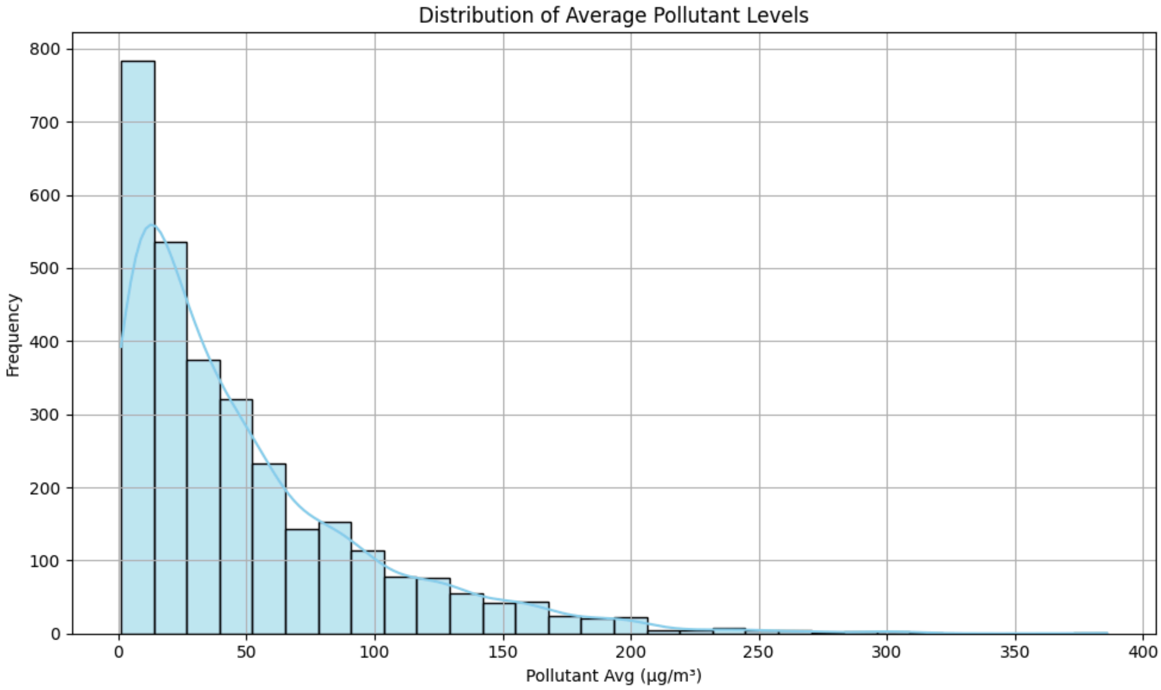
```
C:\Users\SABITHA\AppData\Local\Temp\ipykernel_10456\2056229655.py:4: FutureWarnin
g:

Passing `palette` without assigning `hue` is deprecated and will be removed in v
0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effe
ct.

  sns.boxplot(data=df[df['state'].isin(top_states)],
```



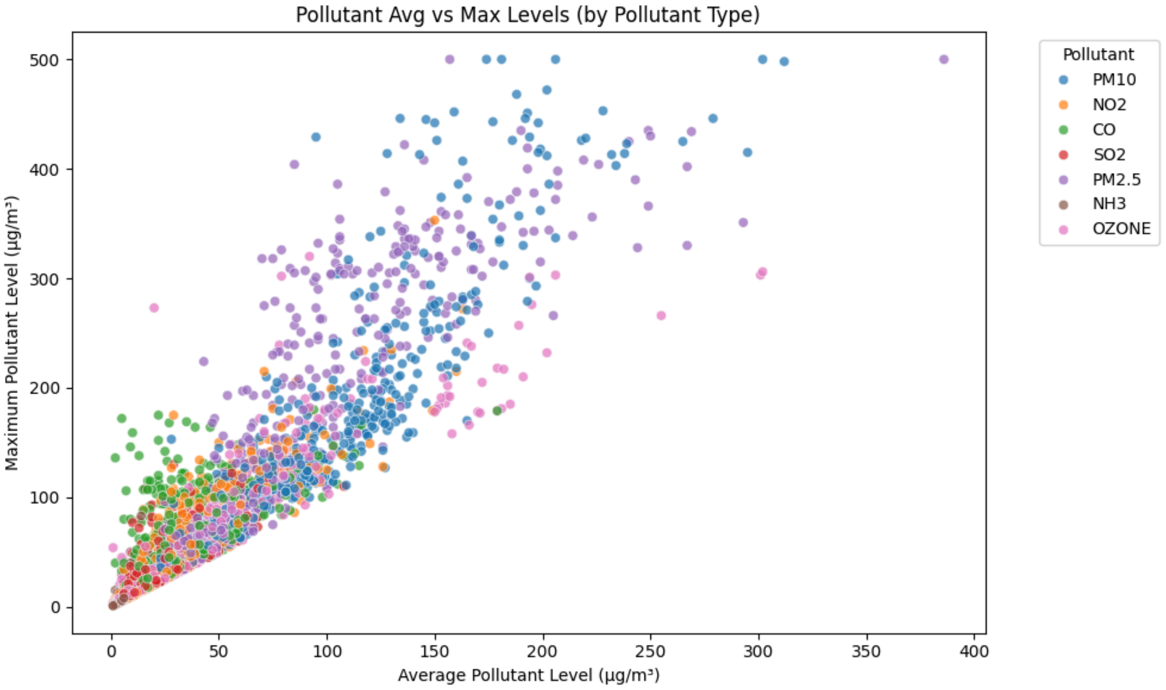
```
In [26]: # Choose a city (e.g., Delhi)
         delhi = df[df['city'] == 'Delhi'].sort_values('last_update')

         plt.figure(figsize=(12,6))
         plt.plot(delhi['last_update'], delhi['pollutant_avg'], label='Avg Pollutant')
         plt.title("Temporal Trend of Pollution in Delhi")
         plt.xlabel("Date")
         plt.ylabel("Pollution Level (µg/m³)")
         plt.xticks(rotation=45)
         plt.legend()
         plt.tight_layout()
         plt.show()
```



In [27]:
```
plt.figure(figsize=(10,6))
sns.histplot(df['pollutant_avg'], bins=30, kde=True, color='skyblue')
plt.title("Distribution of Average Pollutant Levels")
plt.xlabel("Pollutant Avg (µg/m³)")
plt.ylabel("Frequency")
plt.grid(True)
plt.tight_layout()
plt.show()
```

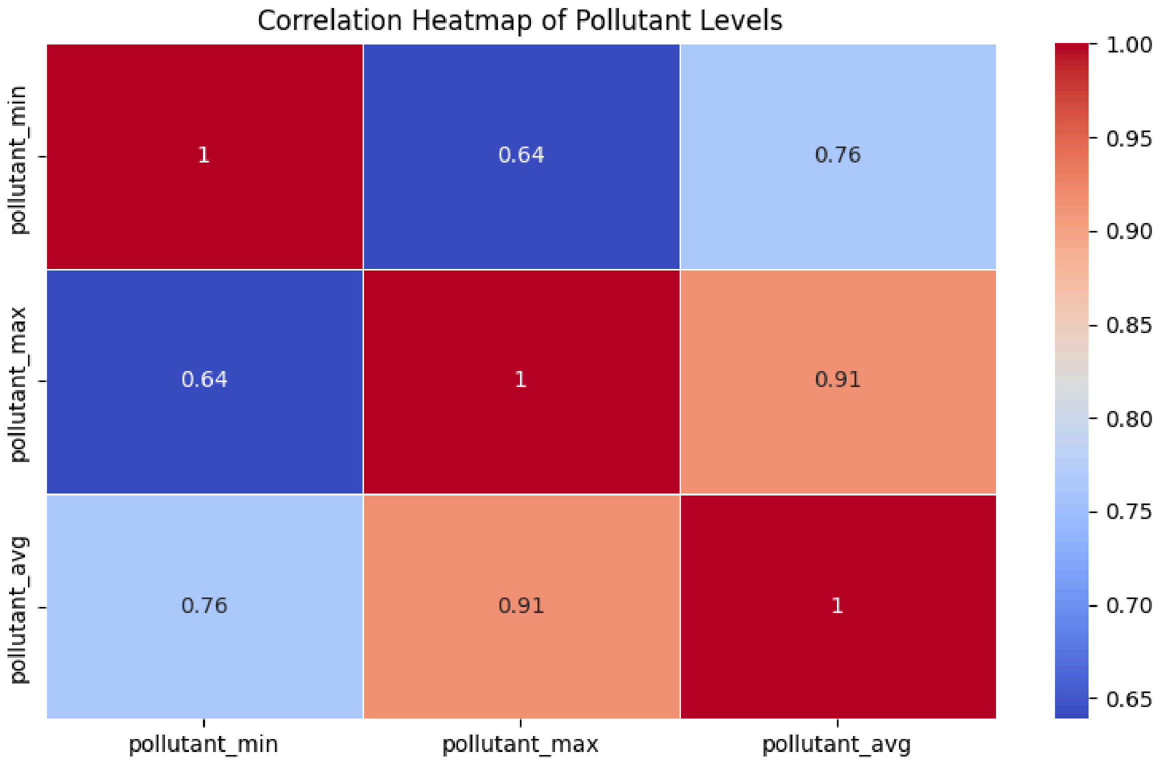


In [28]:
```
plt.figure(figsize=(10,6))
sns.scatterplot(data=df, x='pollutant_avg', y='pollutant_max', hue='pollutant_id
plt.title("Pollutant Avg vs Max Levels (by Pollutant Type)")
plt.xlabel("Average Pollutant Level (µg/m³)")
plt.ylabel("Maximum Pollutant Level (µg/m³)")
plt.legend(title='Pollutant', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.tight_layout()
plt.show()
```



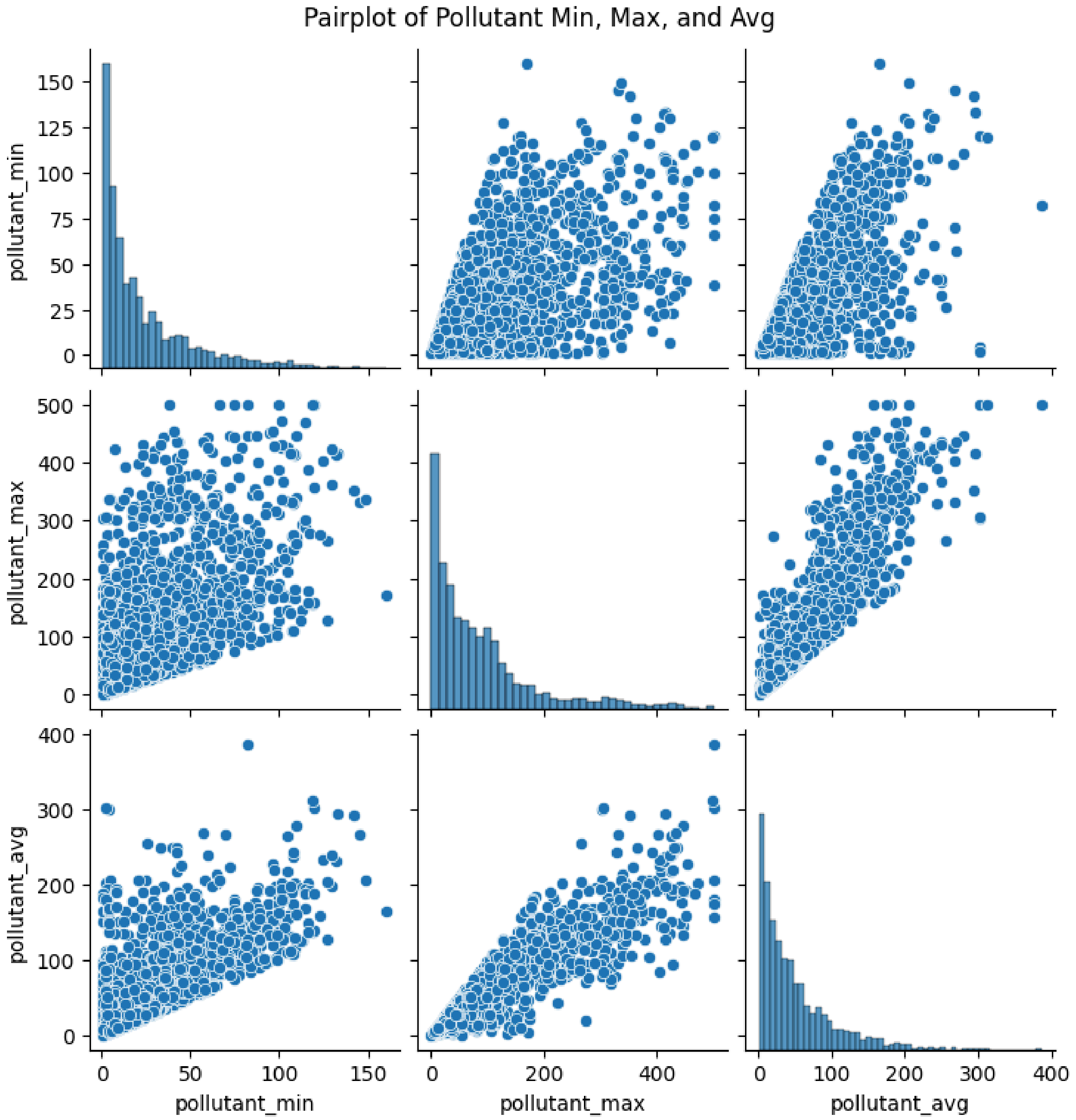
In [29]:

```
numeric_df = df[['pollutant_min', 'pollutant_max', 'pollutant_avg']]

# Compute correlation matrix
corr = numeric_df.corr()

# Plot heatmap
plt.figure(figsize=(8,5))
sns.heatmap(corr, annot=True, cmap='coolwarm', linewidths=0.5)
plt.title("Correlation Heatmap of Pollutant Levels")
plt.tight_layout()
plt.show()
```



In [30]:

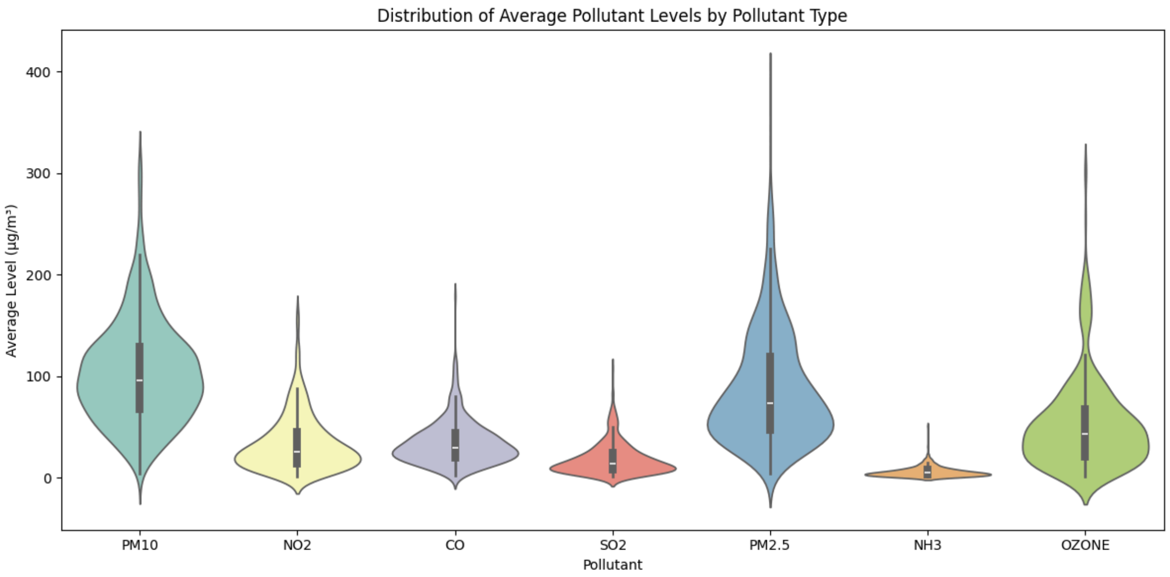
```
sns.pairplot(df[['pollutant_min', 'pollutant_max', 'pollutant_avg']])
plt.suptitle("Pairplot of Pollutant Min, Max, and Avg", y=1.02)
plt.show()
```



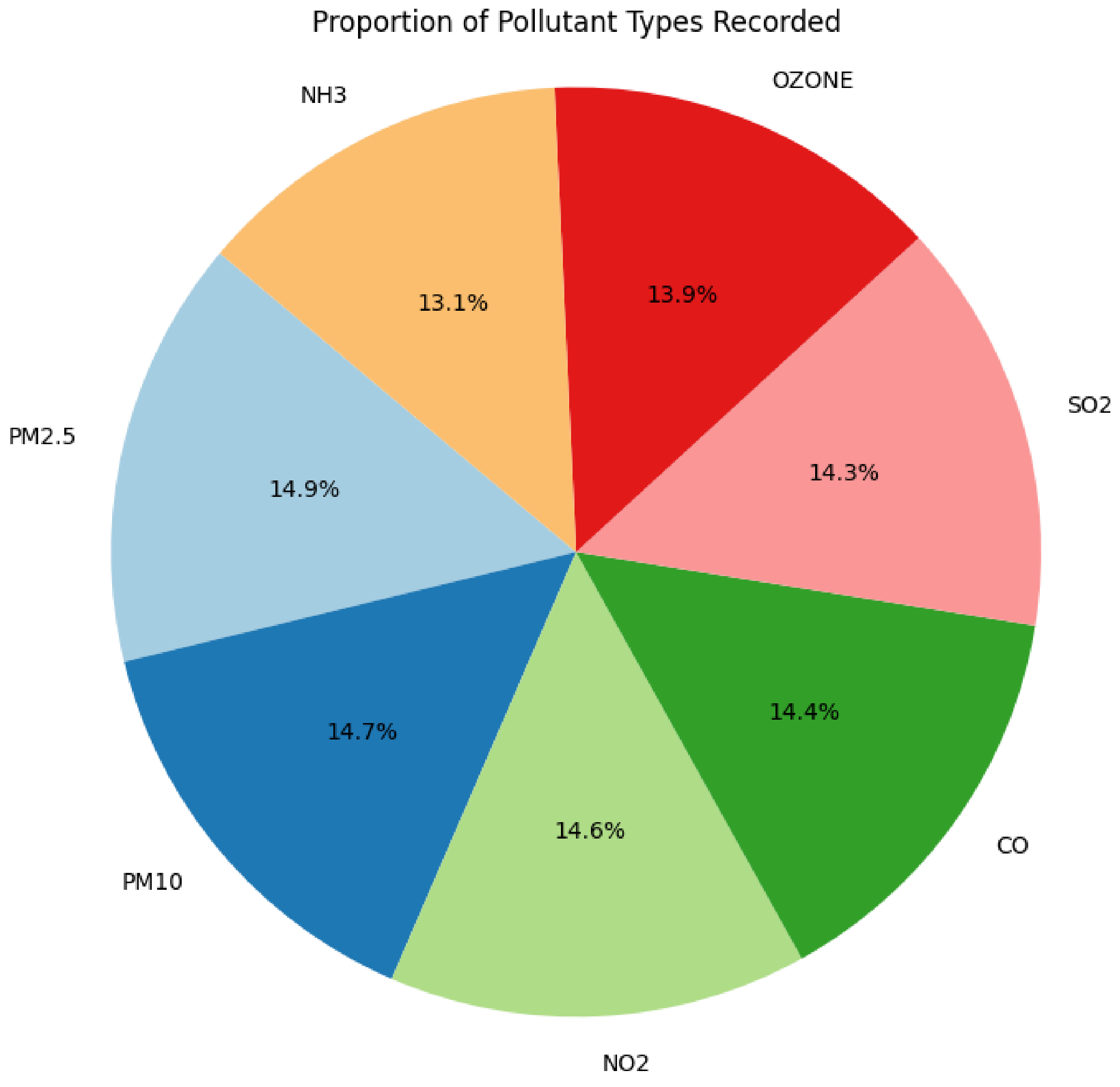
In [31]:

```
plt.figure(figsize=(12,6))
sns.violinplot(data=df, x='pollutant_id', y='pollutant_avg', palette='Set3')
plt.title("Distribution of Average Pollutant Levels by Pollutant Type")
plt.xlabel("Pollutant")
plt.ylabel("Average Level (µg/m³)")
plt.tight_layout()
plt.show()
```

```
C:\Users\SABITHA\AppData\Local\Temp\ipykernel_10456\443648726.py:2: FutureWarnin
g: 

Passing `palette` without assigning `hue` is deprecated and will be removed in v
0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effe
ct.

  sns.violinplot(data=df, x='pollutant_id', y='pollutant_avg', palette='Set3')
```

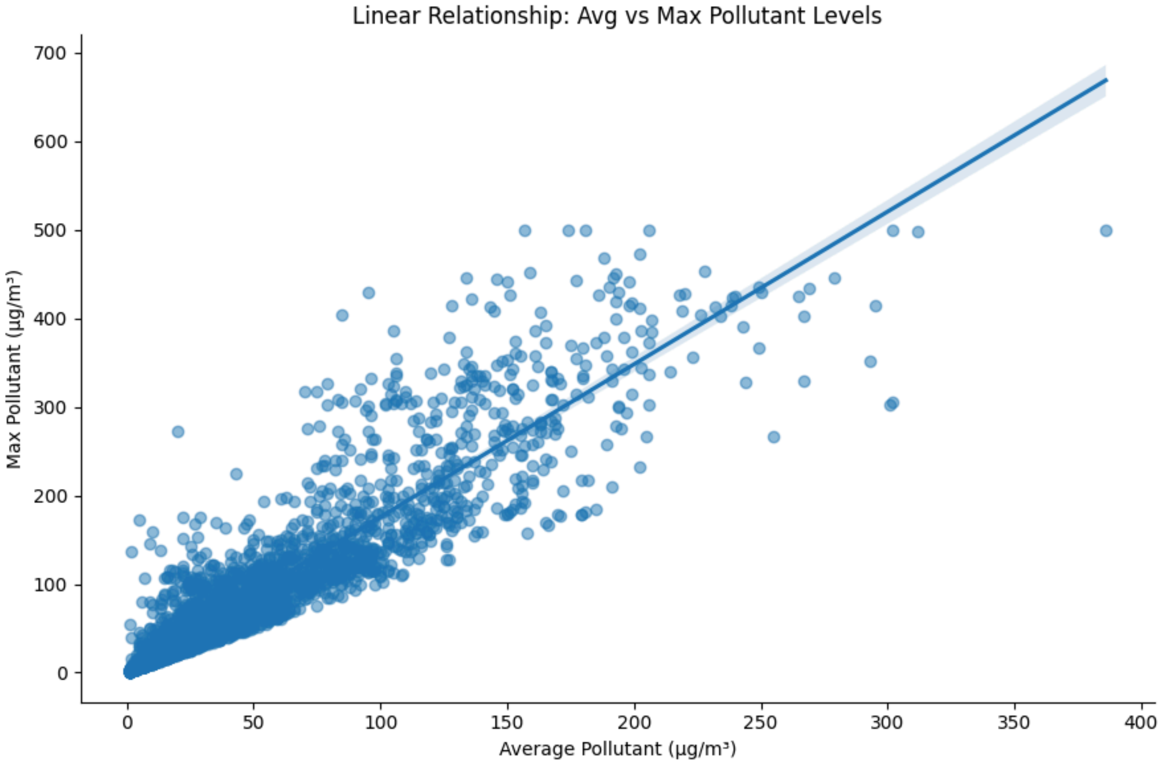


In [37]:
```
pollutant_counts = df['pollutant_id'].value_counts()

plt.figure(figsize=(8,8))
plt.pie(pollutant_counts, labels=pollutant_counts.index, autopct='%1.1f%%', star
plt.title("Proportion of Pollutant Types Recorded")
plt.axis('equal')  # Equal aspect ratio for a perfect circle
plt.show()
```



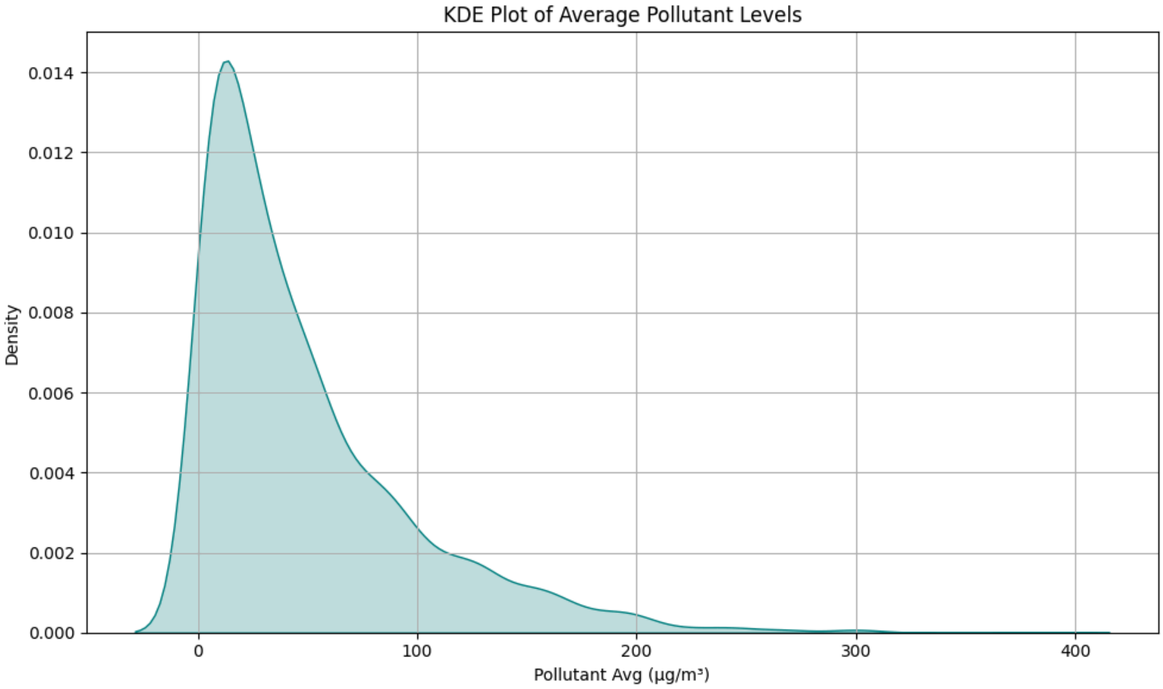
In [38]:

```
sns.lmplot(data=df, x='pollutant_avg', y='pollutant_max', height=6, aspect=1.5,
plt.title("Linear Relationship: Avg vs Max Pollutant Levels")
plt.xlabel("Average Pollutant (µg/m³)")
plt.ylabel("Max Pollutant (µg/m³)")
plt.tight_layout()
plt.show()
```



In [39]:

```
plt.figure(figsize=(10,6))
sns.kdeplot(data=df, x='pollutant_avg', fill=True, color='teal')
plt.title("KDE Plot of Average Pollutant Levels")
plt.xlabel("Pollutant Avg (µg/m³)")
plt.ylabel("Density")
plt.grid(True)
plt.tight_layout()
plt.show()
```


KDE Plot of Average Pollutant Levels
Density
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
0
100
200
300
400
Pollutant Avg (µg/m³)

In [ ]: