In [2]:
```
!pip install openpyxl
```

```
Collecting openpyxl
  Downloading openpyxl-3.1.5-py2.py3-none-any.whl.metadata (2.5 kB)
Collecting et-xmlfile (from openpyxl)
  Downloading et_xmlfile-2.0.0-py3-none-any.whl.metadata (2.7 kB)
Downloading openpyxl-3.1.5-py2.py3-none-any.whl (250 kB)
Downloading et_xmlfile-2.0.0-py3-none-any.whl (18 kB)
Installing collected packages: et-xmlfile, openpyxl

   ---------------------------------------- 0/2 [et-xmlfile]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
   -------------------- ------------------- 1/2 [openpyxl]
```

```
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
-------------------- ------------------- 1/2 [openpyxl]
```

```
   ---------------------- ---------------------- 1/2 [openpyxl]
   ---------------------- ---------------------- 1/2 [openpyxl]
   ---------------------- ---------------------- 1/2 [openpyxl]
   ---------------------- ---------------------- 1/2 [openpyxl]
   ---------------------- ---------------------- 1/2 [openpyxl]
   ---------------------------------------- 2/2 [openpyxl]

Successfully installed et-xmlfile-2.0.0 openpyxl-3.1.5
[notice] A new release of pip is available: 25.3 -> 26.0.1
[notice] To update, run: python.exe -m pip install --upgrade pip
```

In [2]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
```

In [3]:
```python
f = pd.read_excel("capstone dataset.xlsx")
print(df.head())
print(df.info())

print("\nStatistical Summary:")
print(df.describe())
```

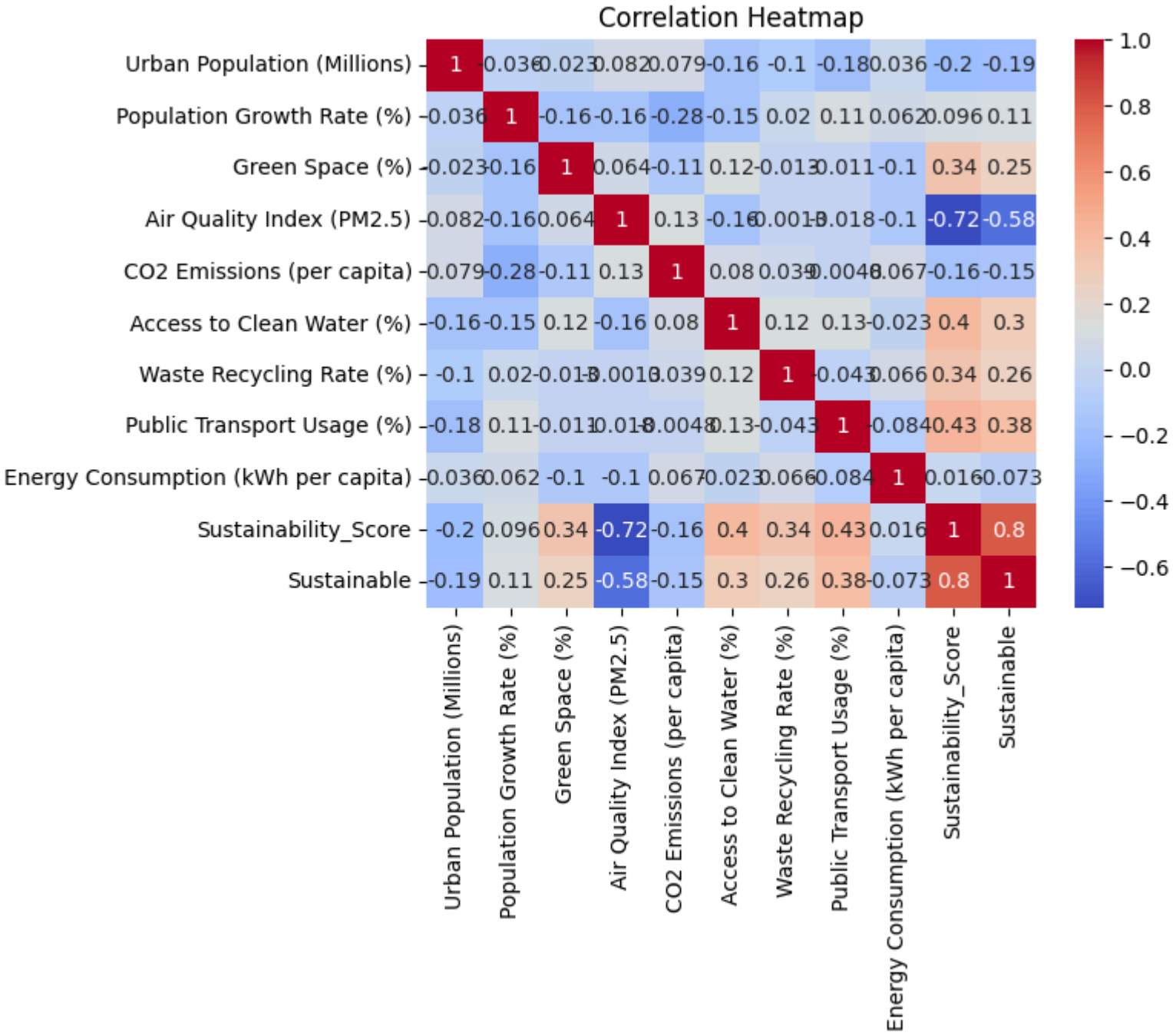
```
     City Name Country  Urban Population (Millions)  \
0      Chennai   India                          1.4
1        Delhi   India                         10.8
2       Mumbai   India                         11.5
3     New York     USA                         13.6
4  Los Angeles     USA                         13.8

   Population Growth Rate (%)  Green Space (%)  Air Quality Index (PM2.5)  \
0                         0.8               44                         99
1                         1.2               24                         64
2                         2.7               13                         50
3                         0.8               27                         30
4                         2.8               41                         85

   CO2 Emissions (per capita)  Access to Clean Water (%)  \
0                         4.9                         98
1                         5.6                         93
2                         1.8                         96
3                         4.9                         99
4                         0.9                         98

   Waste Recycling Rate (%)  Public Transport Usage (%)  \
0                        41                          59
1                        59                          64
2                        46                          71
3                        42                          75
4                        52                          51

   Energy Consumption (kWh per capita)  Sustainability_Score  Sustainable
0                                 1594                138.1            0
1                                 1006                170.4            1
2                                 1352                174.2            1
3                                 1652                208.1            1
4                                  927                156.1            1
<class 'pandas.DataFrame'>
RangeIndex: 82 entries, 0 to 81
Data columns (total 13 columns):
 #   Column                              Non-Null Count  Dtype
---  ------                              --------------  -----
 0   City Name                           82 non-null     str
 1   Country                             82 non-null     str
```

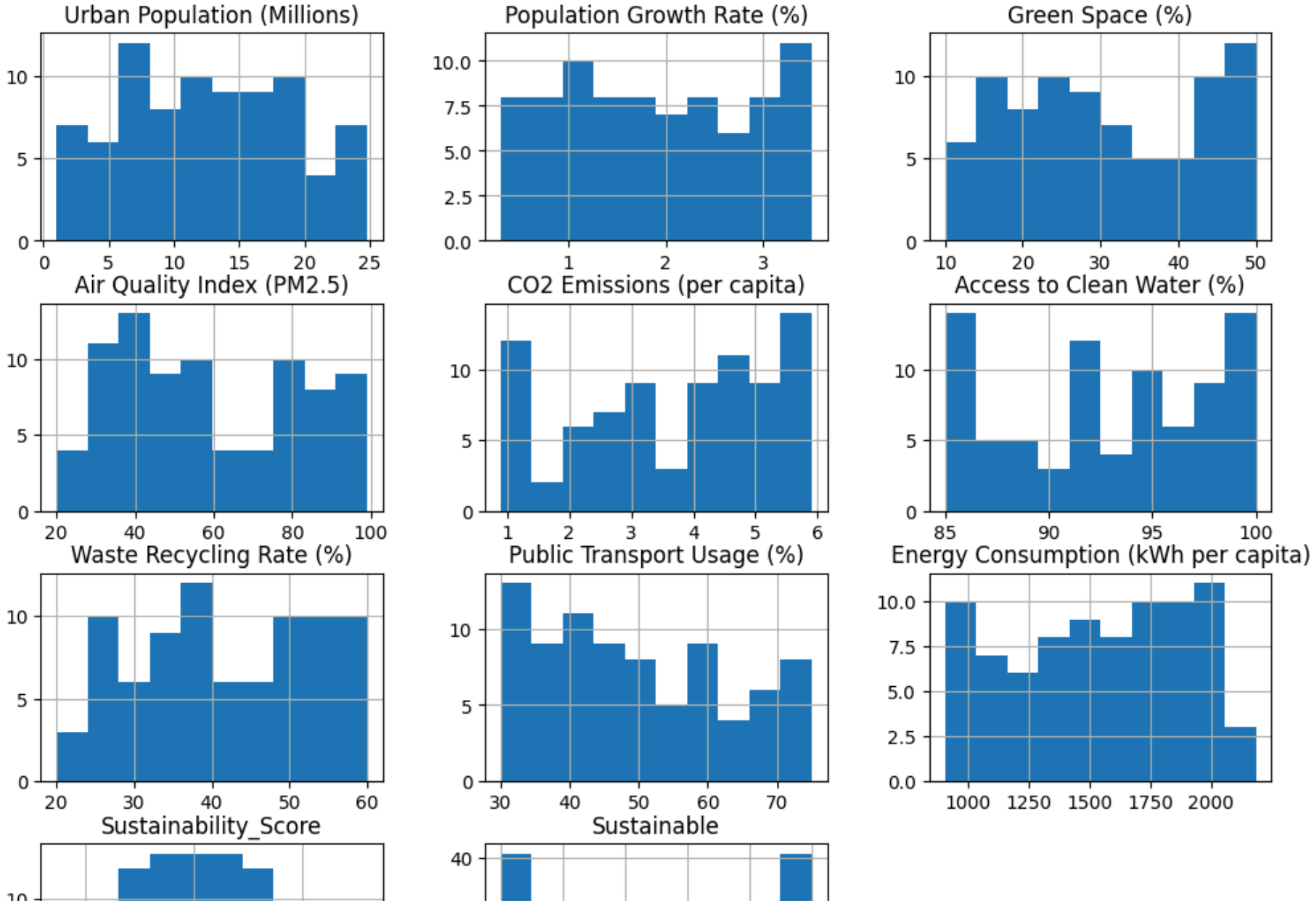
```
 2   Urban Population (Millions)          82 non-null     float64
 3   Population Growth Rate (%)           82 non-null     float64
 4   Green Space (%)                      82 non-null     int64
 5   Air Quality Index (PM2.5)            82 non-null     int64
 6   CO2 Emissions (per capita)           82 non-null     float64
 7   Access to Clean Water (%)            82 non-null     int64
 8   Waste Recycling Rate (%)             82 non-null     int64
 9   Public Transport Usage (%)           82 non-null     int64
 10  Energy Consumption (kWh per capita)  82 non-null     int64
 11  Sustainability_Score                 82 non-null     float64
 12  Sustainable                          82 non-null     int64
dtypes: float64(4), int64(7), str(2)
memory usage: 8.5 KB
None

Statistical Summary:
       Urban Population (Millions)  Population Growth Rate (%)  \
count                    82.000000                   82.000000
mean                     12.521951                    1.909756
std                       6.608151                    0.947070
min                       1.000000                    0.300000
25%                       6.950000                    1.100000
50%                      11.750000                    1.800000
75%                      17.900000                    2.775000
max                      24.800000                    3.500000

       Green Space (%)  Air Quality Index (PM2.5)  CO2 Emissions (per capita)  \
count        82.000000                  82.000000                   82.000000
mean         30.207317                  58.926829                    3.648780
std          12.271354                  23.273352                    1.567119
min          10.000000                  20.000000                    0.900000
25%          19.250000                  38.250000                    2.625000
50%          29.000000                  56.500000                    3.900000
75%          43.000000                  82.000000                    4.900000
max          50.000000                  99.000000                    5.900000

       Access to Clean Water (%)  Waste Recycling Rate (%)  \
count                  82.000000                 82.000000
mean                   92.731707                 41.304878
std                     4.858615                 11.373229
min                    85.000000                 20.000000
```

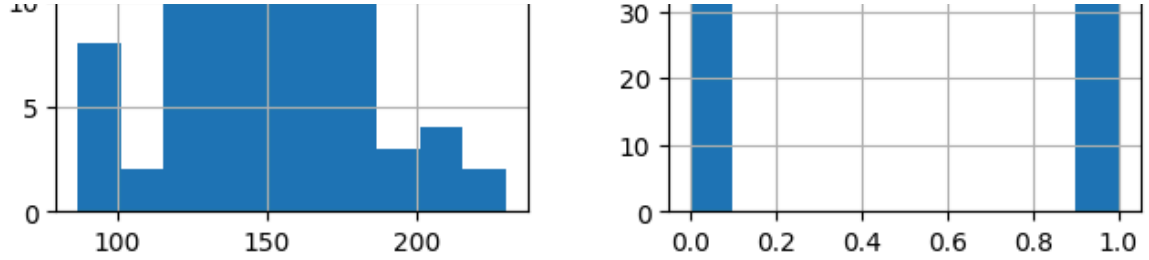
```
25%                    89.000000                 33.000000
50%                    93.000000                 40.500000
75%                    97.000000                 51.000000
max                   100.000000                 60.000000

       Public Transport Usage (%)  Energy Consumption (kWh per capita)  \
count                   82.000000                            82.000000
mean                    49.317073                          1534.500000
std                     13.633544                           363.293448
min                     30.000000                           908.000000
25%                     38.000000                          1252.750000
50%                     46.000000                          1550.500000
75%                     59.000000                          1868.500000
max                     75.000000                          2183.000000

       Sustainability_Score  Sustainable
count             82.000000    82.000000
mean             150.985366     0.500000
std               32.886716     0.503077
min               86.700000     0.000000
25%              128.825000     0.000000
50%              151.000000     0.500000
75%              173.075000     1.000000
max              230.100000     1.000000
```

In [4]:

```
plt.figure()
sns.heatmap(df.corr(numeric_only=True), annot=True, cmap="coolwarm")
plt.title("Correlation Heatmap")
plt.show()

df.hist(figsize=(12,10))
plt.suptitle("Feature Distributions")
plt.show()
```


Correlation Heatmap
Urban Population (Millions)
Population Growth Rate (%)
Green Space (%)
Air Quality Index (PM2.5)
CO2 Emissions (per capita)
Access to Clean Water (%)
Waste Recycling Rate (%)
Public Transport Usage (%)
Energy Consumption (kWh per capita)
Sustainability_Score
Sustainable
1 -0.036 0.023 0.082 0.079 -0.16 -0.1 -0.18 0.036 -0.2 -0.19
-0.036 1 -0.16 -0.16 -0.28 -0.15 0.02 0.11 0.062 0.096 0.11
-0.023 -0.16 1 0.064 -0.11 0.12 -0.013 0.011 -0.1 0.34 0.25
-0.082 -0.16 0.064 1 0.13 -0.16 0.0018 -0.018 -0.1 -0.72 -0.58
-0.079 -0.28 -0.11 0.13 1 0.08 0.039 0.0048 0.067 -0.16 -0.15
-0.16 -0.15 0.12 -0.16 0.08 1 0.12 0.13 -0.023 0.4 0.3
-0.1 0.02 -0.013 0.0018 0.039 0.12 1 -0.043 0.066 0.34 0.26
-0.18 0.11 -0.011 0.018 0.0048 0.13 -0.043 1 -0.084 0.43 0.38
-0.036 0.062 -0.1 -0.1 0.067 0.023 0.066 0.084 1 0.016 0.073
-0.2 0.096 0.34 -0.72 -0.16 0.4 0.34 0.43 0.016 1 0.8
-0.19 0.11 0.25 -0.58 -0.15 0.3 0.26 0.38 -0.073 0.8 1
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6


Feature Distributions
Urban Population (Millions)
Population Growth Rate (%)
Green Space (%)
Air Quality Index (PM2.5)
CO2 Emissions (per capita)
Access to Clean Water (%)
Waste Recycling Rate (%)
Public Transport Usage (%)
Energy Consumption (kWh per capita)
Sustainability_Score
Sustainable



In [5]:
```
df["Sustainability_Score"] = (
    df["Green Space (%)"] +
    df["Access to Clean Water (%)"] +
    df["Waste Recycling Rate (%)"] +
    df["Public Transport Usage (%)"]
) - (
    df["Air Quality Index (PM2.5)"] +
    df["CO2 Emissions (per capita)"]
)

df["Sustainable"] = df["Sustainability_Score"].apply(
    lambda x: 1 if x > df["Sustainability_Score"].median() else 0
)

print(df[["Sustainability_Score", "Sustainable"]].head())
```

```
   Sustainability_Score  Sustainable
0                 138.1            0
1                 170.4            1
2                 174.2            1
3                 208.1            1
4                 156.1            1
```

In [6]:
```
X = df.drop(["City Name", "Country", "Sustainability_Score", "Sustainable"], axis=1)
y = df["Sustainable"]
```

In [7]:
```
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)
```

In [8]:
```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

In [9]:
```
model = LogisticRegression()
model.fit(X_train, y_train)
```

Out[9]:

▾ LogisticRegression

▸ Parameters

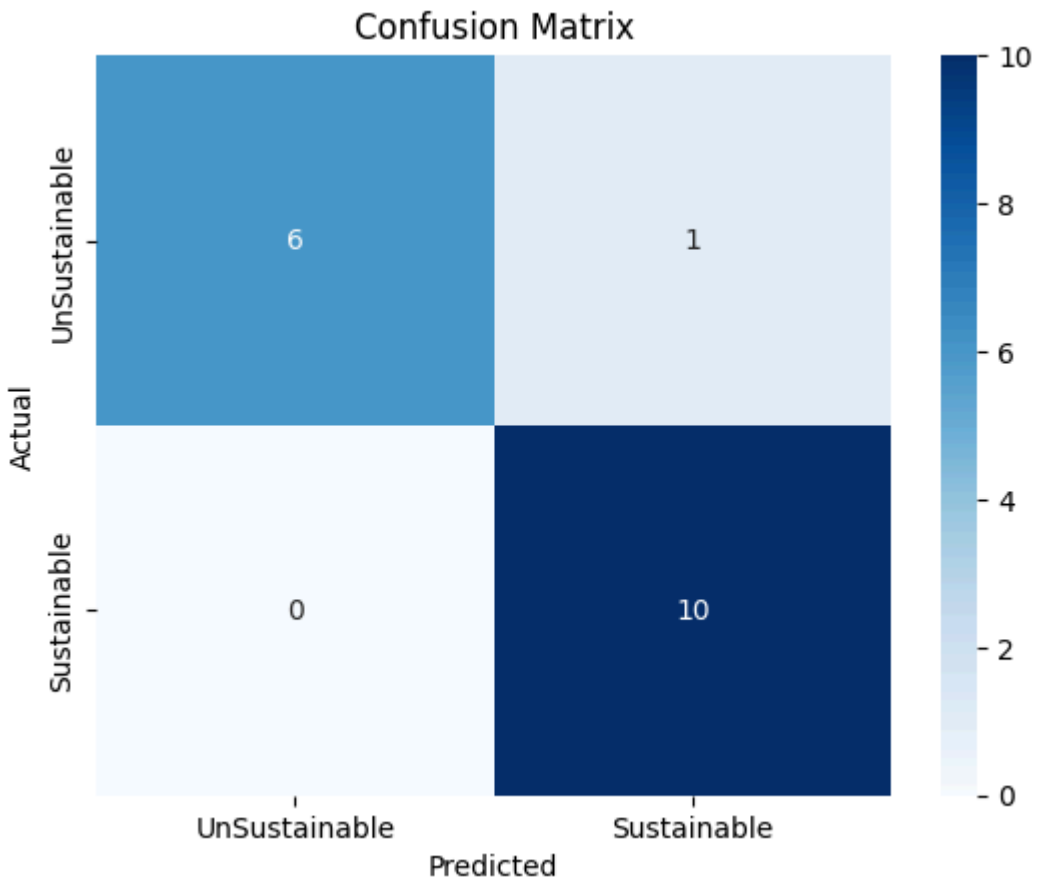
In [10]:
```
y_pred = model.predict(X_test)
```

In [11]:
```
print("\nAccuracy:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
```

```
Accuracy: 0.9411764705882353

Classification Report:
              precision    recall  f1-score   support

           0       1.00      0.86      0.92         7
           1       0.91      1.00      0.95        10

    accuracy                           0.94        17
   macro avg       0.95      0.93      0.94        17
weighted avg       0.95      0.94      0.94        17
```

In [12]:
```
conf_matrix = confusion_matrix(y_test, y_pred)

plt.figure()
sns.heatmap(conf_matrix, annot=True, fmt="d", cmap="Blues",
            xticklabels=["UnSustainable", "Sustainable"],
            yticklabels=["UnSustainable", "Sustainable"])
plt.xlabel("Predicted")
plt.ylabel("Actual")
```

```
plt.title("Confusion Matrix")
plt.show()
```



In [ ]: