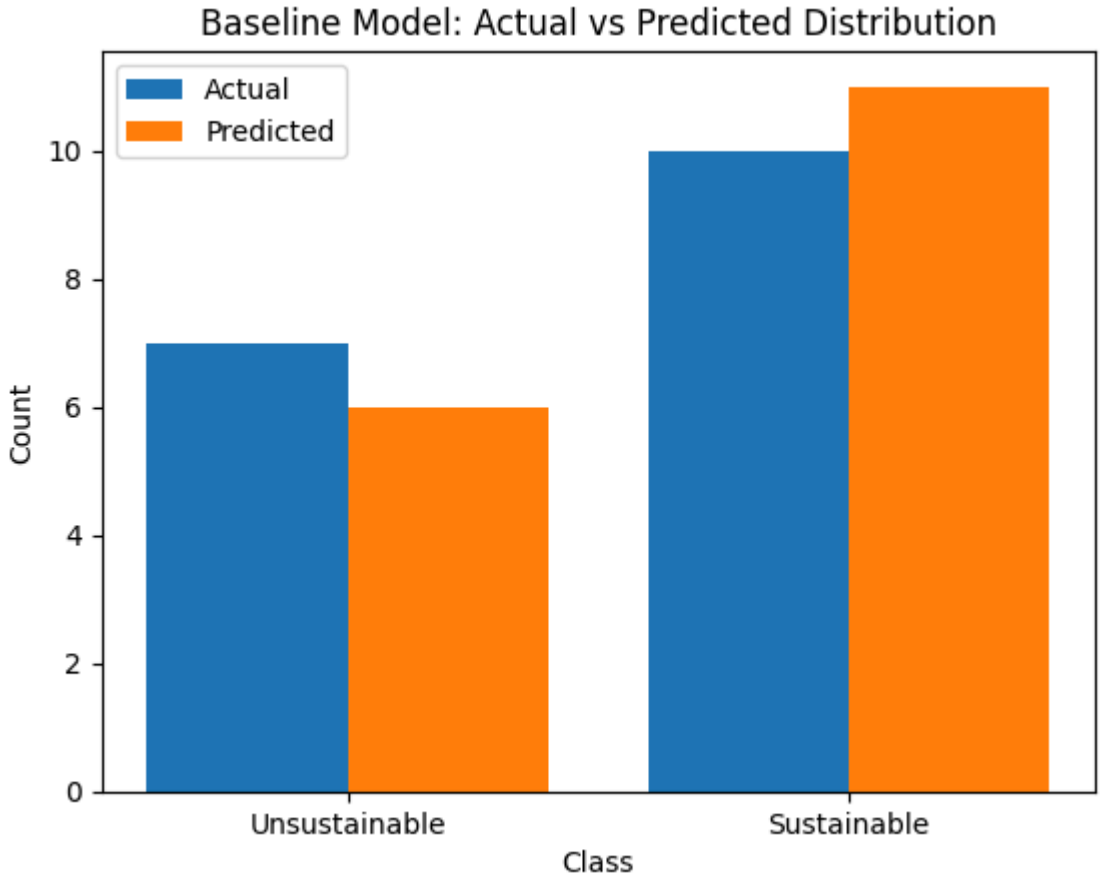
In [3]:

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix

df = pd.read_excel("capstone dataset.xlsx")

le_city = LabelEncoder()
le_country = LabelEncoder()

df["City Name"] = le_city.fit_transform(df["City Name"])
df["Country"] = le_country.fit_transform(df["Country"])

df["Sustainability_Score"] = (
    df["Green Space (%)"] +
    df["Access to Clean Water (%)"] +
    df["Waste Recycling Rate (%)"]
) - (
    df["Air Quality Index (PM2.5)"]
)

labels = ["Unsustainable", "Sustainable"]

actual_counts = np.bincount(y_test)
pred_counts = np.bincount(y_pred)

x = np.arange(len(labels))

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

X = df.drop(["Sustainability_Score", "Sustainable"], axis=1)
y = df["Sustainable"]

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=0
)

scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

baseline_model = LogisticRegression(max_iter=1000)
baseline_model.fit(X_train, y_train)

y_pred = baseline_model.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
print("Baseline Accuracy:", accuracy)

print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
plt.figure()
plt.bar(x - 0.2, actual_counts, width=0.4, label="Actual")
```

```
plt.bar(x + 0.2, pred_counts, width=0.4, label="Predicted")

plt.xticks(x, labels)
plt.xlabel("Class")
plt.ylabel("Count")
plt.title("Baseline Model: Actual vs Predicted Distribution")
plt.legend()
plt.show()
```

```
Baseline Accuracy: 0.8235294117647058
Confusion Matrix:
 [[5 2]
 [1 9]]
```



Visualizations: Patterns, Predictions, and Feature insights

In [5]:

```
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 LabelEncoder, StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, accuracy_score
from sklearn.ensemble import RandomForestClassifier

df = pd.read_excel("capstone dataset.xlsx")

le_city = LabelEncoder()
```

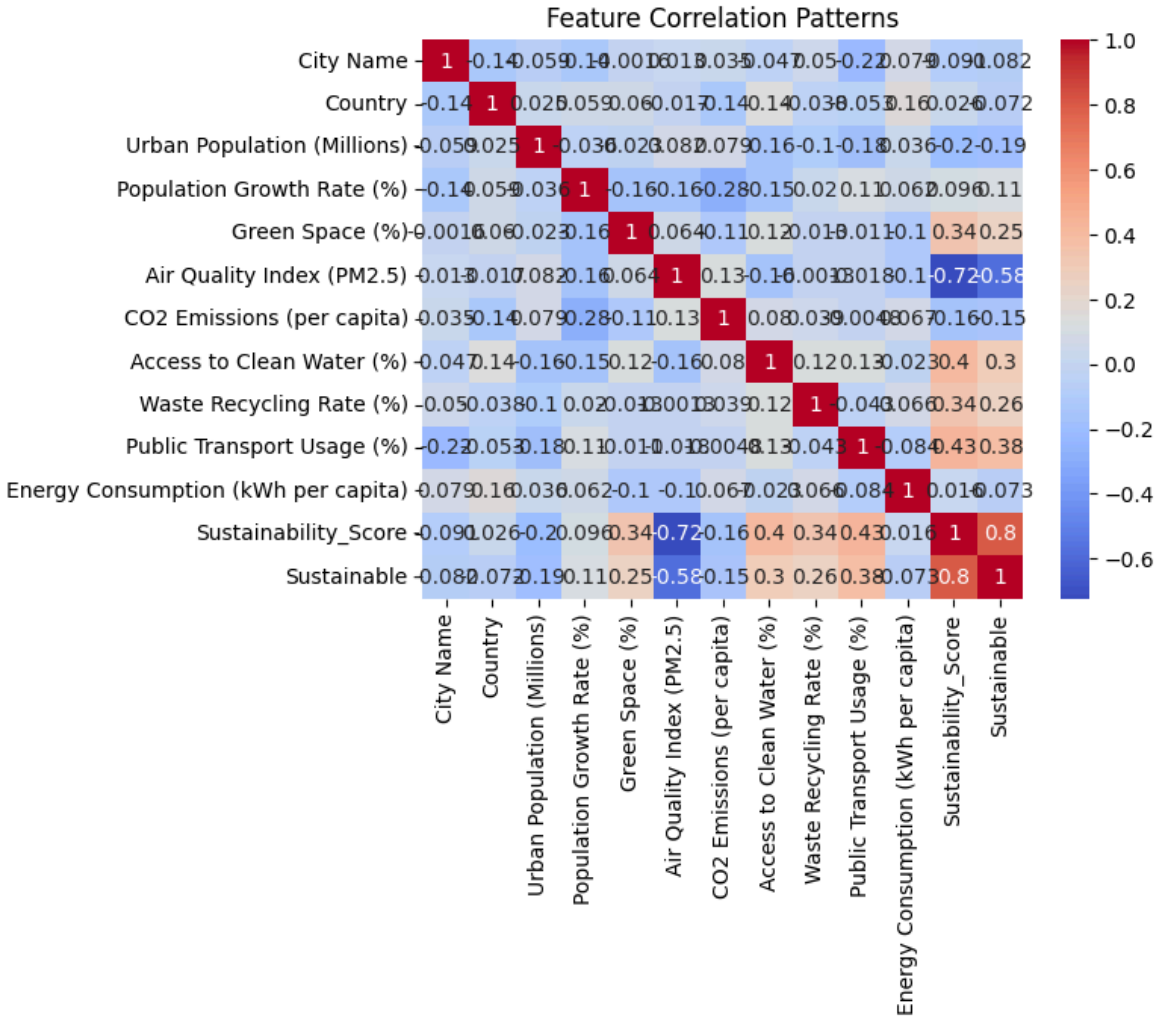
```
le_country = LabelEncoder()
df["City Name"] = le_city.fit_transform(df["City Name"])
df["Country"] = le_country.fit_transform(df["Country"])

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
)

X = df.drop(["Sustainability_Score", "Sustainable"], axis=1)
y = df["Sustainable"]

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)

scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

model = LogisticRegression(max_iter=2000)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)

print("Baseline Accuracy:", accuracy_score(y_test, y_pred))
```

```
Baseline Accuracy: 0.9411764705882353
```

Pattern Visualization

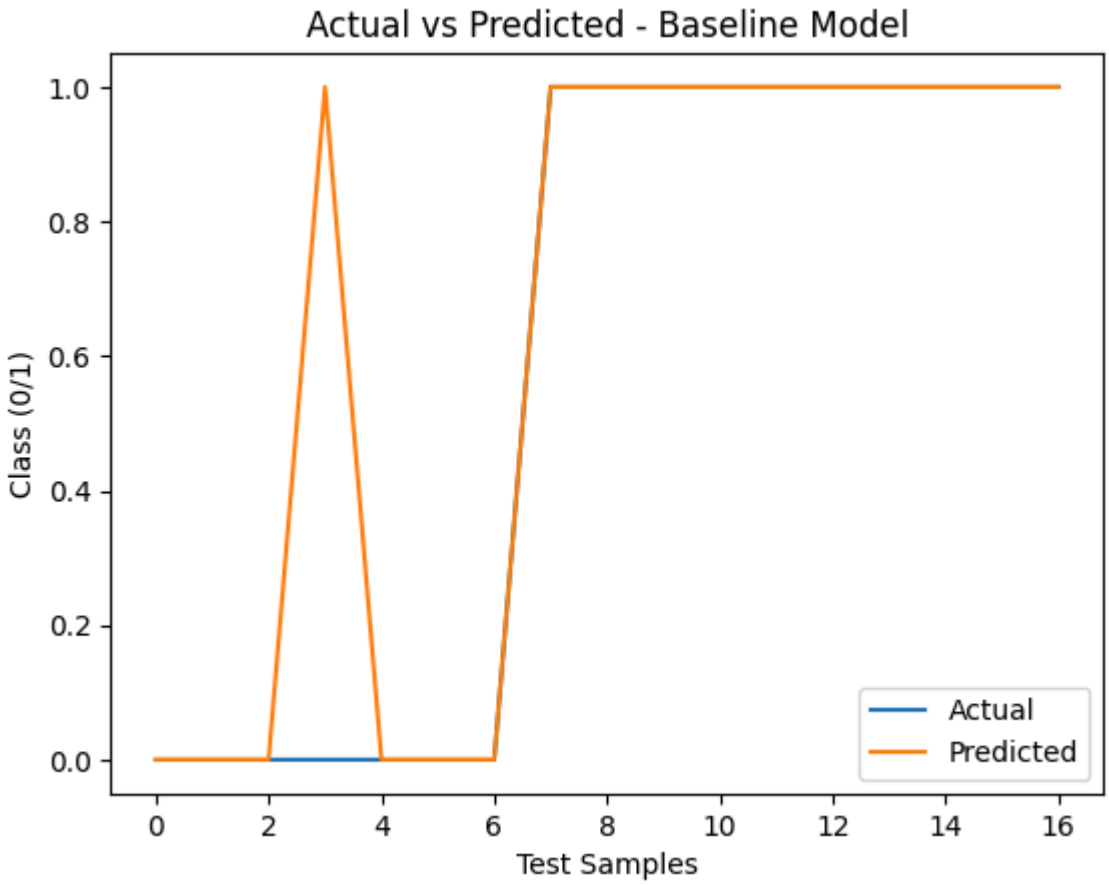
In [6]:

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



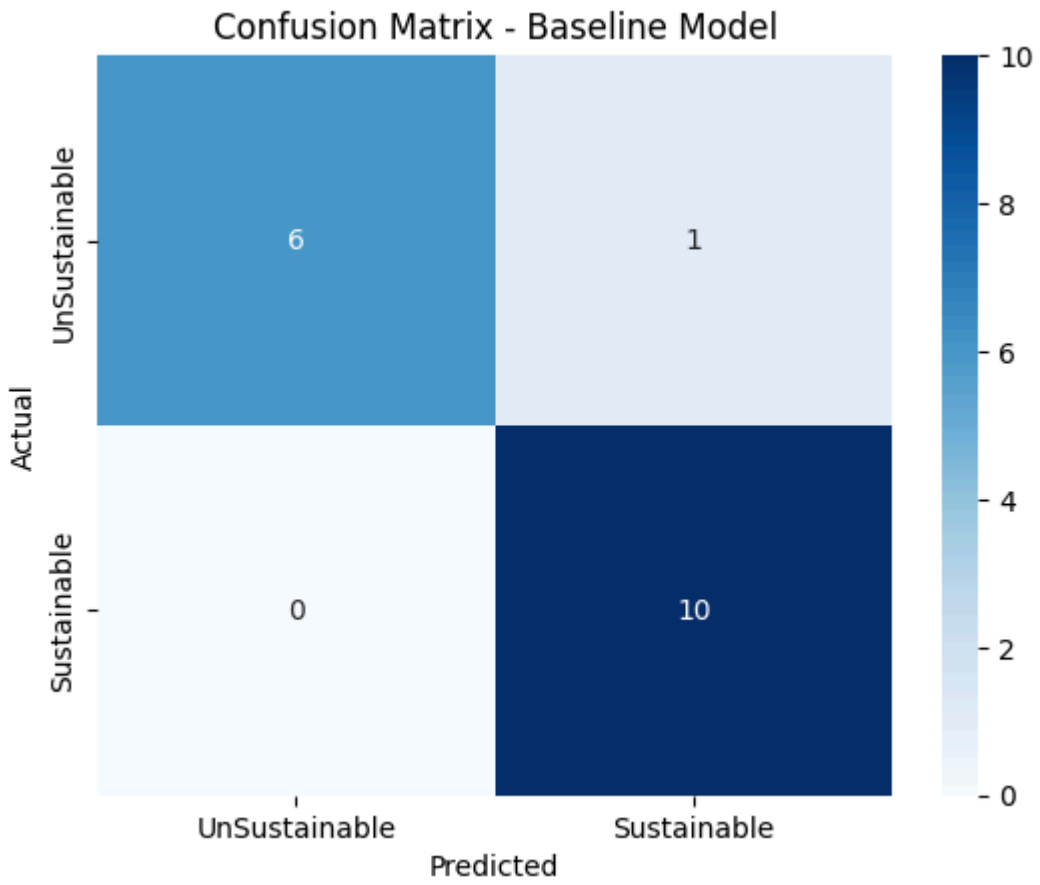
Prediction Visualization

In [7]:
```
indices = np.argsort(y_test)
plt.figure()
plt.plot(y_test.iloc[indices].values, label="Actual")
plt.plot(y_pred[indices], label="Predicted")
plt.xlabel("Test Samples")
plt.ylabel("Class (0/1)")
plt.title("Actual vs Predicted - Baseline Model")
plt.legend()
plt.show()
```



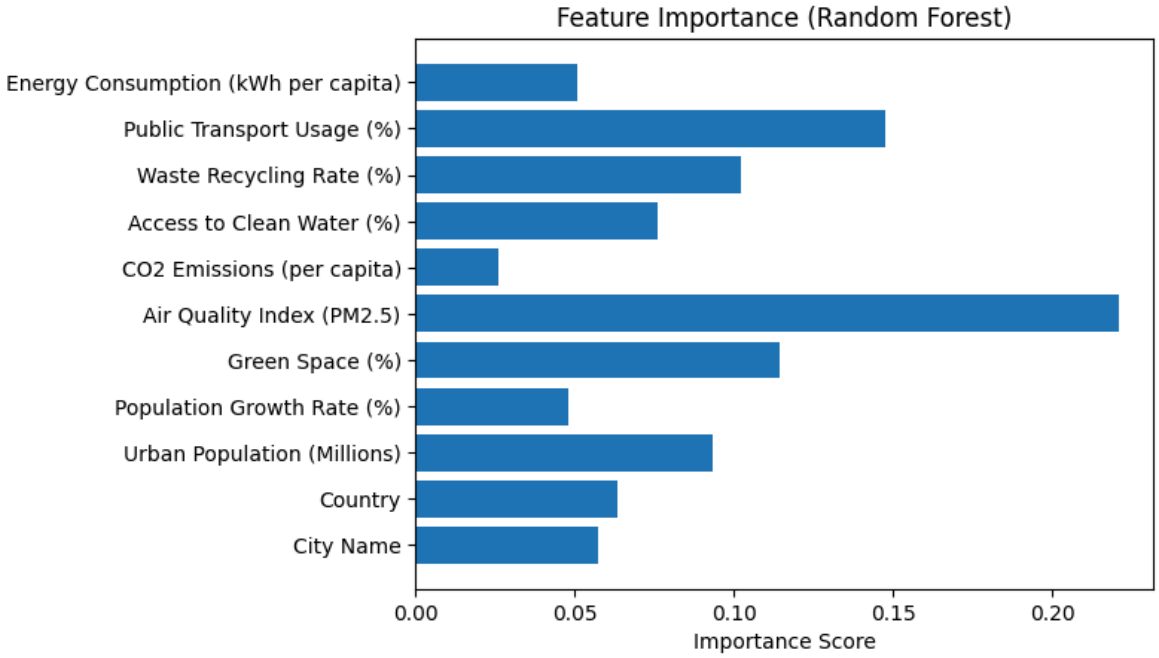
Confusion Matrix

In [8]:
```
cm = confusion_matrix(y_test, y_pred)
plt.figure()
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues",
            xticklabels=["UnSustainable", "Sustainable"],
            yticklabels=["UnSustainable", "Sustainable"])
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix - Baseline Model")
plt.show()
```



Feature Insight

In [9]:
```
rf = RandomForestClassifier(n_estimators=100, random_state=42)
rf.fit(X_train, y_train)

importances = rf.feature_importances_
features = X.columns

plt.figure()
plt.barh(features, importances)
plt.xlabel("Importance Score")
plt.title("Feature Importance (Random Forest)")
plt.show()
```


Feature Importance (Random Forest)
Energy Consumption (kWh per capita)
Public Transport Usage (%)
Waste Recycling Rate (%)
Access to Clean Water (%)
CO2 Emissions (per capita)
Air Quality Index (PM2.5)
Green Space (%)
Population Growth Rate (%)
Urban Population (Millions)
Country
City Name
0.00
0.05
0.10
0.15
0.20
Importance Score

In [ ]: