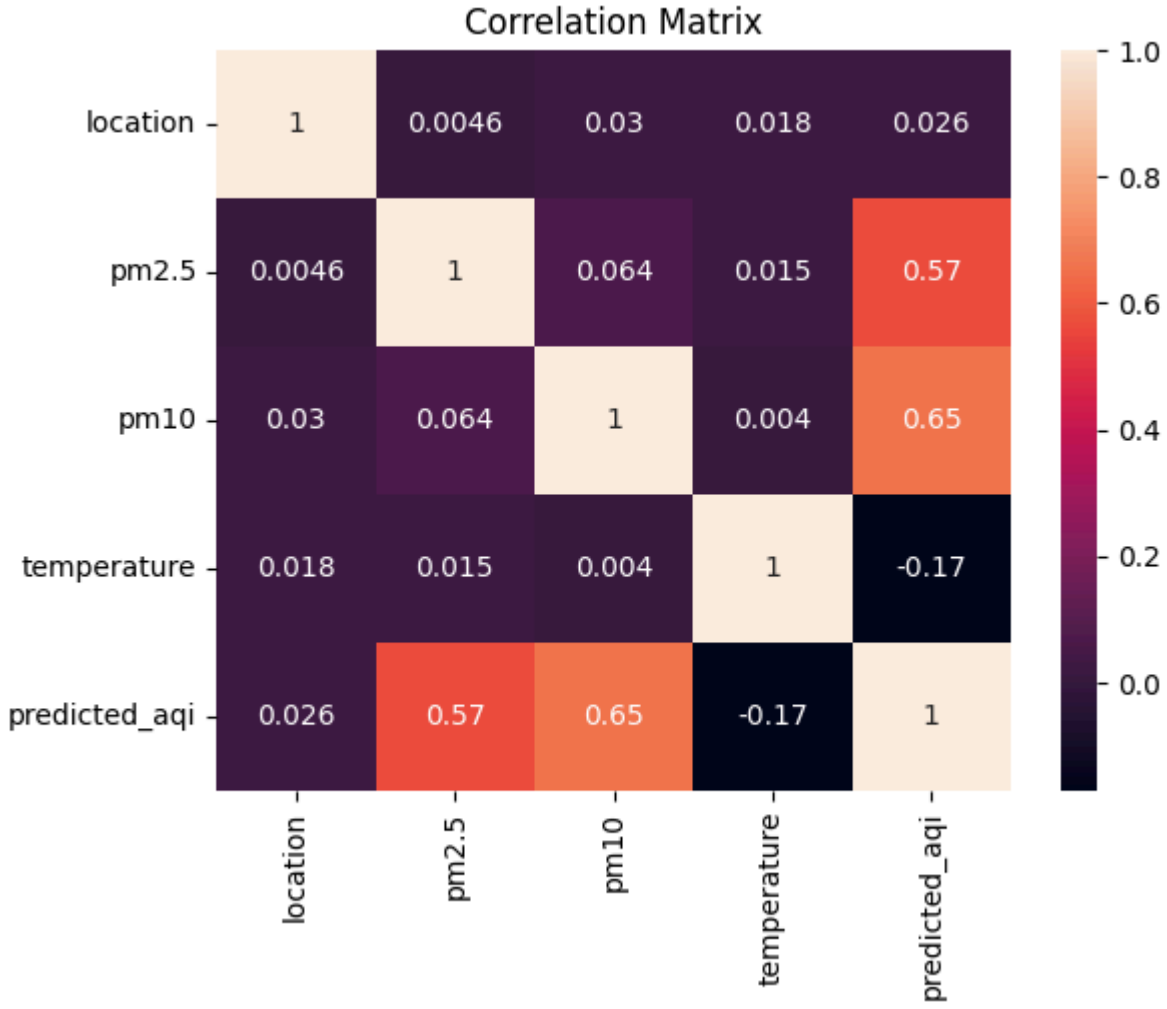
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_csv("16_air_quality_prediction.csv")
```

```
# 2. Data Cleaning & EDA
print(df.isnull().sum())  # Check for nulls
df['date'] = pd.to_datetime(df['date'])  # Convert date column
print(df.describe())  # Summary stats

# Convert 'location' column to numeric before calculating correlation
df['location'] = pd.factorize(df['location'])[0]

# Calculate correlation matrix only for numeric columns
numeric_df = df.select_dtypes(include=['number'])
sns.heatmap(numeric_df.corr(), annot=True);
plt.title("Correlation Matrix");
plt.show()
```

```
date             0
location         0
pm2.5            0
pm10             0
temperature      0
predicted_aqi    0
dtype: int64
```

	date	pm2.5	pm10	temperature
count	1460	1460.00000	1460.000000	1460.000000
mean	2023-07-02 00:00:00	35.06900	69.815788	20.006644
min	2023-01-01 00:00:00	2.94000	-0.190000	5.100000
25%	2023-04-02 00:00:00	28.59750	56.887500	12.000000
50%	2023-07-02 00:00:00	35.23000	70.025000	19.850000
75%	2023-10-01 00:00:00	41.59500	82.572500	27.800000
max	2023-12-31 00:00:00	67.08000	142.220000	35.000000
std	NaN	10.08376	19.665744	8.826885

	predicted_aqi
count	1460.000000
mean	44.493281
min	15.000000
25%	38.045000
50%	44.765000
75%	50.825000
max	78.930000
std	9.543069



```
# 3. Feature Engineering
df['month'] = df['date'].dt.month
df['day'] = df['date'].dt.day
df['location'] = df['location'].astype('category').cat.codes
```

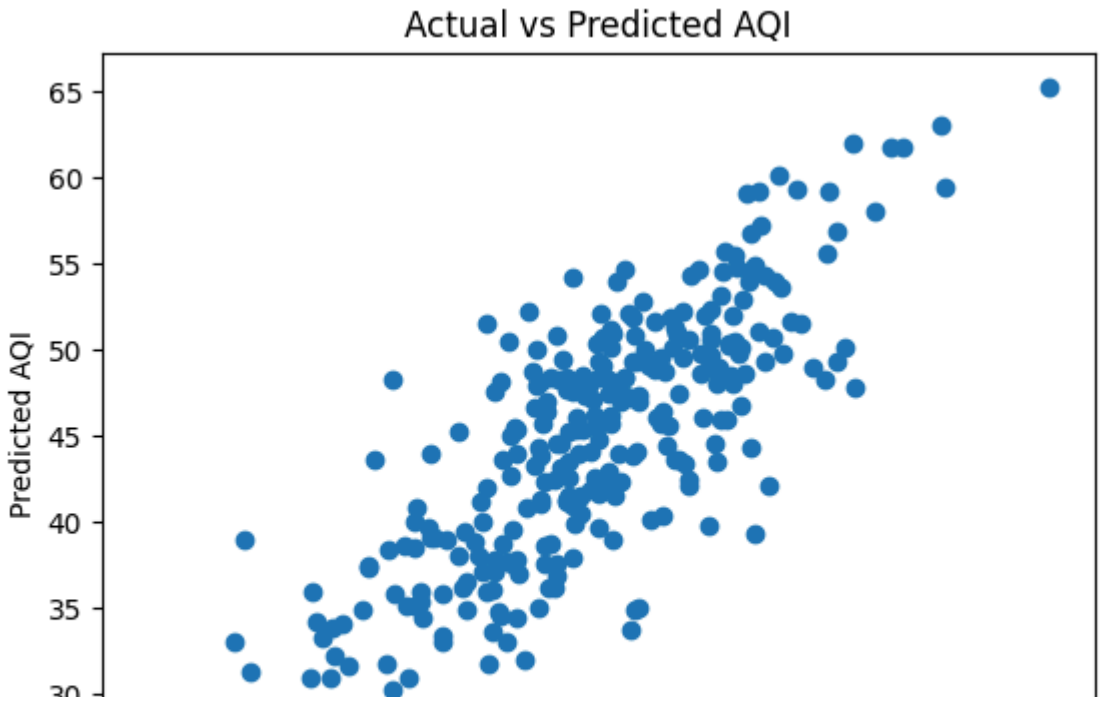
```
# 4. Model Building
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor

X = df.drop(['date', 'predicted_aqi'], axis=1)
y = df['predicted_aqi']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

model = RandomForestRegressor()
model.fit(X_train, y_train)
```

```
RandomForestRegressor
RandomForestRegressor()
```

```
# 5. Model Evaluation
from sklearn.metrics import mean_squared_error, r2_score

y_pred = model.predict(X_test)
print("MSE:", mean_squared_error(y_test, y_pred))
print("R² Score:", r2_score(y_test, y_pred))
```

```
MSE: 32.508867539863026
R² Score: 0.6353566394675392
```

```
# 6. Visualization and Interpretation
plt.scatter(y_test, y_pred)
plt.xlabel("Actual AQI")
plt.ylabel("Predicted AQI")
plt.title("Actual vs Predicted AQI")
plt.show()
```




```
# Feature importance
feat_importance = pd.Series(model.feature_importances_, index=X.columns)
feat_importance.sort_values().plot(kind='barh')
plt.title("Feature Importance")
plt.show()
```