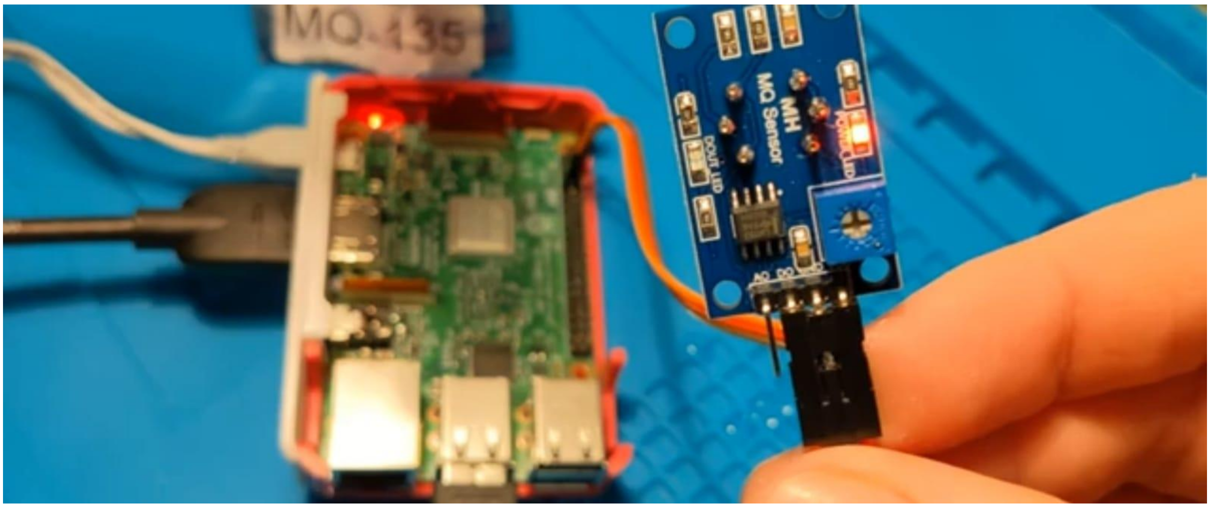
AIR QUALITY MONITORING SYSTEM

Building an air quality monitoring system is a complex project that typically involves collecting data Raspberry Pi from various sensors and sources. To get started, we'll create a simplified example by using a synthetic dataset. In practice, it would obtain real-time data from air quality sensors and sources, but this example will demonstrate the loading and preprocessing steps.



generate a synthetic dataset to simulate air quality data.

```
import pandas as pd import

numpy as np

# Create a synthetic dataset

data = pd.DataFrame({

   'Timestamp': pd.date_range(start='2023-01-01', periods=100,
freq='H'),

   'CO2 (ppm)': np.random.randint(350, 700, 100),

   'PM2.5 (µg/m³)': np.random.randint(5, 50, 100),

   'Temperature (°C)': np.random.uniform(10, 30, 100),

'Humidity (%)': np.random.uniform(30, 70, 100)

})
```

Handling Imbalanced Data (If Applicable):

If your dataset has imbalanced classes, consider techniques like oversampling, undersampling, or using different evaluation metrics to handle the class imbalance.

These are general preprocessing steps, and the specifi steps you need to perform depend on the nature of your dataset and the algorithms you plan to use. Always tailor your preprocessing steps to the requirements of your specific project.

Exploratory Data Analysis (EDA):

. Explore the synthetic dataset to understand its structure.

```
# Display the first few rows of the dataset print(data.head())

# Check data types and missing values
print(data.info()) # Summary statistics
print(data.describe())
```

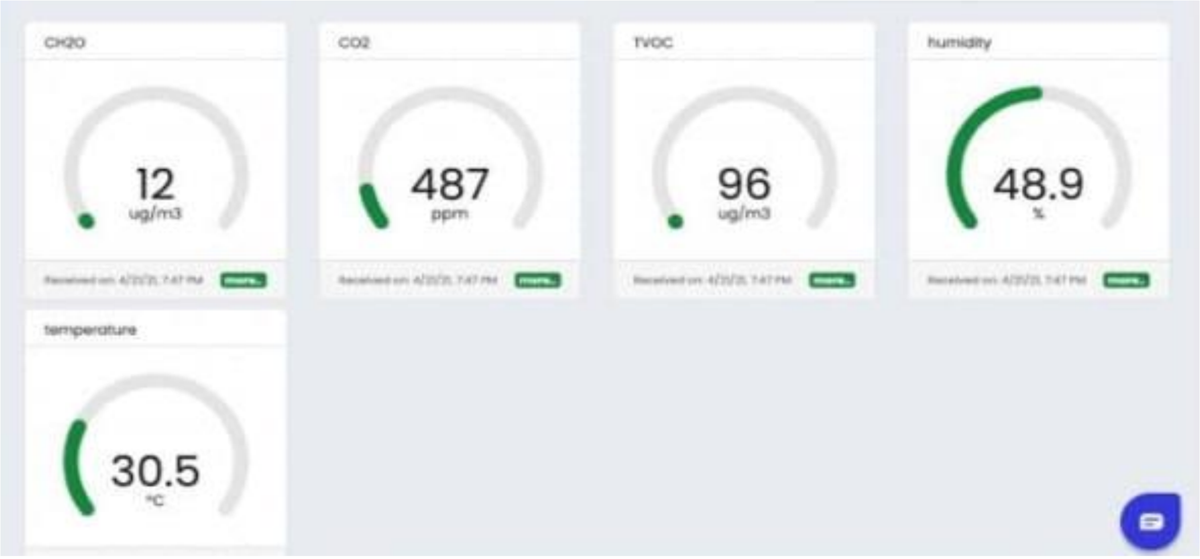
Data Preprocessing:

In a real-world scenario, air quality data might require more extensive preprocessing, including handling missing data, smoothing sensor noise, and more. For this synthetic dataset, we'll keep it simple.

```
# No missing values in this synthetic dataset

# Smooth sensor noise (optional)

# Convert the 'Timestamp' column to the datetime format data['Timestamp']

= pd.to_datetime(data['Timestamp'])

# Set the 'Timestamp' column as the DataFrame's index (useful for time-series data)
data.set_index('Timestamp', inplace=True)
```

You can save this synthetic dataset for further analysis or visualization.

```
data.to_csv('air_quality_data.csv')
```


CH2O
12
ug/m3
Received on: 4/21/21, 7:47 PM
more..
CO2
487
ppm
Received on: 4/21/21, 7:47 PM
more..
TVOC
96
ug/m3
Received on: 4/21/21, 7:47 PM
more..
humidity
48.9
%
Received on: 4/21/21, 7:47 PM
more..
temperature
30.5
°C