# AIR QUALITY MONITORING

## PHASE 5 : PROJECT DOCUMENTATION & SUBMISSION

### DOCUMENTATION:

**PROJECT OBJECTIVES** :

The project objectives are well-defined, encompassing real-time air quality monitoring, data sharing, public awareness, and emphasizing the health impact of air quality. This ensures a holistic approach towards achieving the project's mission.

**IOT DEVICE SETUP** :

Setting up an IoT device for air quality monitoring involves several steps:

**1.Selecting the Device**: Choose an appropriate sensor or device designed for air quality monitoring. Devices may measure various pollutants like particulate matter (PM), volatile organic compounds (VOCs), carbon monoxide (CO), etc.

**2.Connecting Sensors**: Connect the chosen sensor to a microcontroller or a development board, like Arduino or Raspberry Pi, that will collect data from the sensor.

**3.Data Transmission**: Utilize a communication protocol (Wi-Fi, Bluetooth, LoRa, etc.) to send the collected data to a server or a cloud platform for storage and analysis.

**4.Cloud Platform Integration**: Set up a cloud-based platform (like AWS, Azure, or IoT platforms) to receive, store, and process the data. You can use platforms that offer data visualization, analytics, and alerts.

**5.Visualization and Analytics**: Use software or applications to visualize and analyze the data collected. This could include creating graphs, charts, and setting thresholds for alerts.

**6.Power Supply and Enclosure**: Ensure a stable power supply for the device and consider weatherproofing or enclosing the device if it will be deployed outdoors.

**7.Testing and Calibration**: Before deployment, test the device thoroughly and calibrate the sensors for accuracy.

**<u>PLATFORM DEVELOPMENT & CODE IMPLEMENTATION</u>**:

**<u>Index.html:</u>**

```
 <!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<link rel="stylesheet" href="styles.css">
<linkhref="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.
cssrel="stylesheet">
<scriptsrc="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bund
le.min.js">
</script>
<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
<title>Air Quality Monitoring</title>
</head>
<body>
<div class="container-fluid text-center bg-primary" style="min-height: 15vh;">
<h1 class="text-white py-4"> Air Quality  Monitoring</h1>
</div>
<div class="container mt-4">
 <div class="row">
 <div class="col-md-4">
<h2>Location: <span id="location">Coimbatore</span></h2>
<p>Temperature: <span id="temperature">Loading...</span> &deg;C</p>
```

```
<p>Humidity: <span id="humidity">Loading...</span>%</p>

<p>PM2.5: <span id="pm25">Loading...</span> µg/m³</p>

<p>PM10: <span id="pm10">Loading...</span> µg/m³</p>

<button onclick="updateRandomData()" class="btn btn-primary">Update Data</button>

</div>

<div class="col-md-4">

<h2>Location: <span id="location1">Chennai</span></h2>

<p>Temperature: <span id="temperature1">Loading...</span> &deg;C</p>

<p>Humidity: <span id="humidity1">Loading...</span>%</p>

<p>PM2.5: <span id="pm251">Loading...</span> µg/m³</p>

<button onclick="updateRandomData1()" class="btn btn-primary">Update Data</button>

</div>

<div class="col-md-4">

<h2>Location: <span id="location2">Trichy</span></h2>

<p>Temperature: <span id="temperature2">Loading...</span> &deg;C</p>

<p>Humidity: <span id="humidity2">Loading...</span>%</p>

<p>PM2.5: <span id="pm252">Loading...</span> µg/m³</p>

<p>PM10: <span id="pm102">Loading...</span> µg/m³</p>

<button onclick="updateRandomData2()" class="btn btn-primary">Update Data</button>

</div>

</div>

<div class="row">
```

```
<canvas id="airQualityChart" width="400" height="200"></canvas>
</div>
</div>
<script src="./script.js"> </script>
</body>
</html>
```

**<u>Script.js :</u>**

```
function updateRandomData() {
const locationElement = document.getElementById("location"); const
temperatureElement = document.getElementById("temperature"); const
humidityElement = document.getElementById("humidity"); const
pm25Element = document.getElementById("pm25"); const
pm10Element = document.getElementById("pm10"); const
randomTemperature = (Math.random() * 40 + 10).toFixed(2);  const
randomHumidity = (Math.random() * 50 + 30).toFixed(2);  const
randomPM25 = Math.floor(Math.random() * 100);  const randomPM10 =
Math.floor(Math.random() * 150);  locationElement.textContent =
"Coimbatore";
temperatureElement.textContent = randomTemperature + " &deg;C";
humidityElement.textContent = randomHumidity + "%"; pm25Element.textContent
= randomPM25 + " µg/m³"; pm10Element.textContent = randomPM10 + " µg/m³";
}
function updateRandomData1() {
   const location1Element = document.getElementById("location1");     const
temperature1Element = document.getElementById("temperature1");     const
humidity1Element = document.getElementById("humidity1");     const
pm251Element = document.getElementById("pm251");     const
pm101Element = document.getElementById("pm101");     const
randomTemperature = (Math.random() * 40 + 10).toFixed(2);      const
randomHumidity = (Math.random() * 50 + 30).toFixed(2);      const
randomPM25 = Math.floor(Math.random() * 100);      const randomPM10 =
Math.floor(Math.random() * 150);

   location1Element.textContent = "Chennai";
   temperature1Element.textContent = randomTemperature + " &deg;C";
humidity1Element.textContent = randomHumidity + "%";
pm251Element.textContent = randomPM25 + " µg/m³";
pm101Element.textContent = randomPM10 + " µg/m³";
}
function updateRandomData2() {
   const location2Element = document.getElementById("location2");   const
  temperature2Element = document.getElementById("temperature2");   const
  humidity2Element = document.getElementById("humidity2");   const
  pm252Element = document.getElementById("pm252");

   const pm102Element = document.getElementById("pm102");
  const randomTemperature = (Math.random() * 40 + 10).toFixed(2);
```

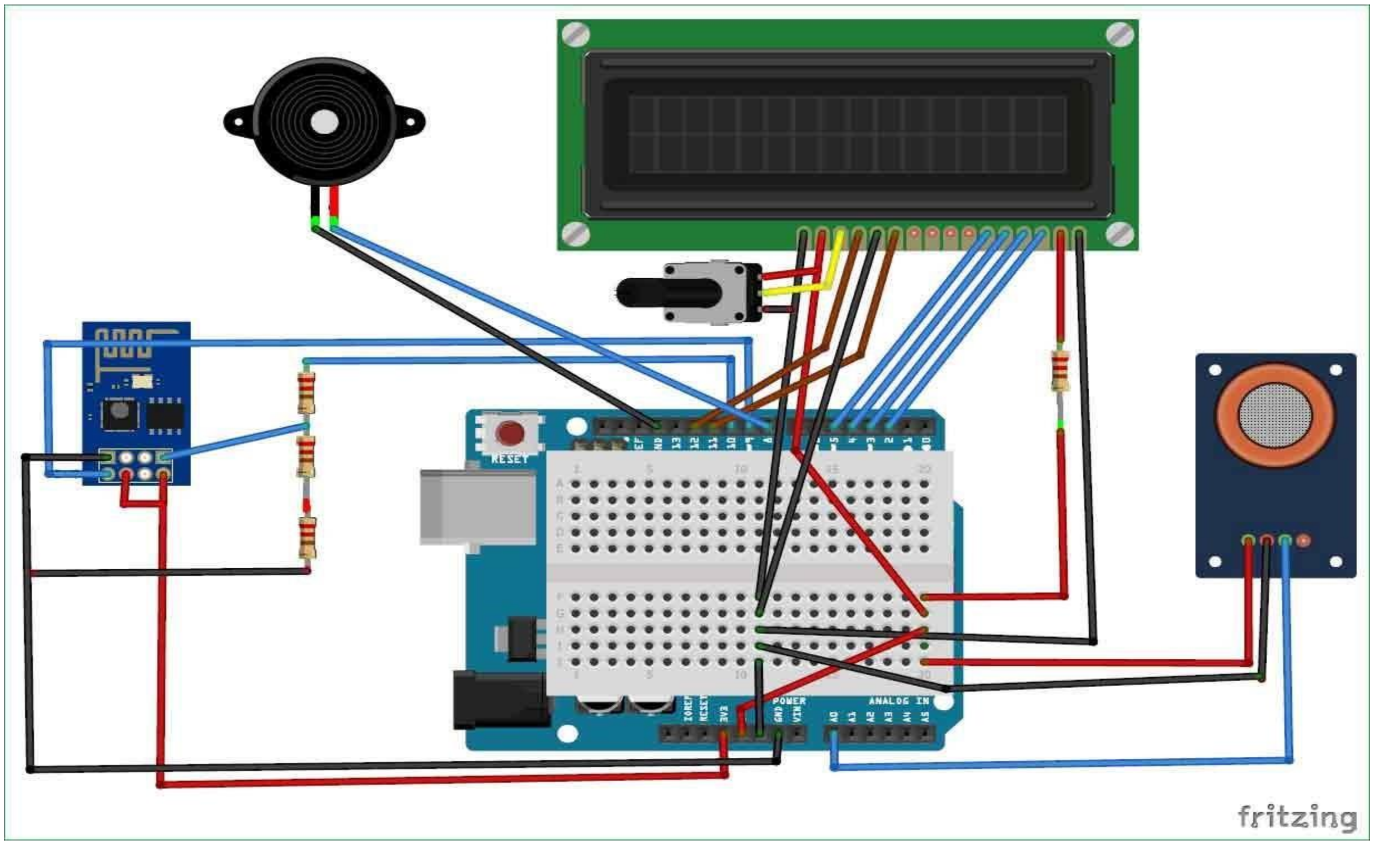
```
    const randomHumidity = (Math.random() * 50 + 30).toFixed(2);
between 0 and 100 µg/m³
    const randomPM10 = Math.floor(Math.random() * 150);


    location2Element.textContent = "Trichy";
    temperature2Element.textContent = randomTemperature + " &deg;C";
humidity2Element.textContent = randomHumidity + "%";
pm252Element.textContent = randomPM25 + " µg/m³";
pm102Element.textContent = randomPM10 + " µg/m³";
}
const historicalData = {
    labels: ["Day 1", "Day 2", "Day 3", "Day 4", "Day 5", "Day 6", "Day 7"],
pm25: [12, 15, 10, 8, 13, 9, 11],     pm10: [20, 18, 22, 19, 21, 17, 20],
  };
  const ctx = document.getElementById("airQualityChart").getContext("2d");
const airQualityChart = new Chart(ctx, {   type: "line",
  data: {
  labels: historicalData.labels, datasets:
  [
  {
        label: "PM2.5 (µg/m³)",
  data: historicalData.pm25,
        borderColor: "rgba(75, 192, 192, 1)",
        fill: false,
        },
        {
          label: "PM10 (µg/m³)",          data:
historicalData.pm10,          borderColor:
"rgba(192, 75, 75, 1)",
          fill: false,
        },
      ],
    },
    options: {
scales: {
x: {
          beginAtZero: true,
        },
y: {
          beginAtZero: true,
        },
      },
    },
  });
  function updateAirQualityChart() {
          const      newPM25Data      =      [...historicalData.pm25.map(()      =>
Math.floor(Math.random() * 20 + 10))];
          const      newPM10Data      =      [...historicalData.pm10.map(()      =>
Math.floor(Math.random() * 20 + 10))];
```

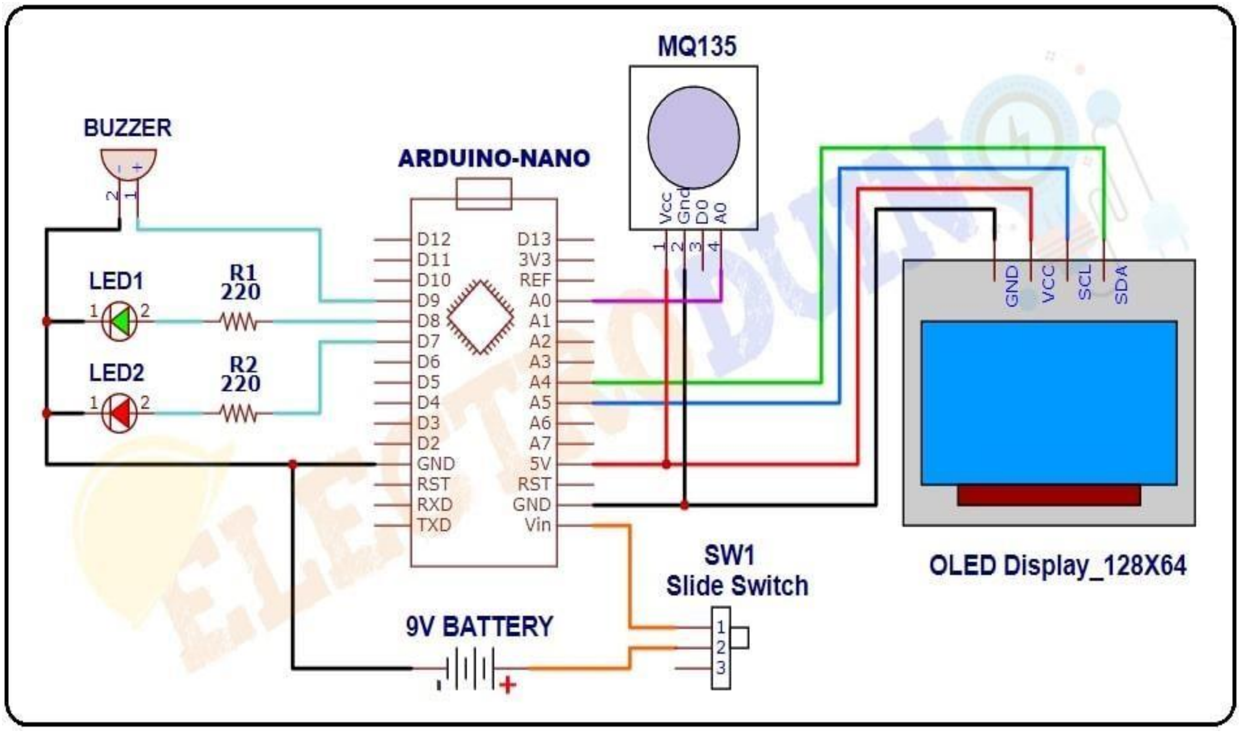
```
   airQualityChart.data.datasets[0].data = newPM25Data;
airQualityChart.data.datasets[1].data = newPM10Data;
airQualityChart.data.labels = historicalData.labels;     airQualityChart.update();
  }
```

## DIAGRAMS , SCHEMATICS , SCREENSHOT OF IOT DEVICE AND DATA-SHARING PLATFORM:

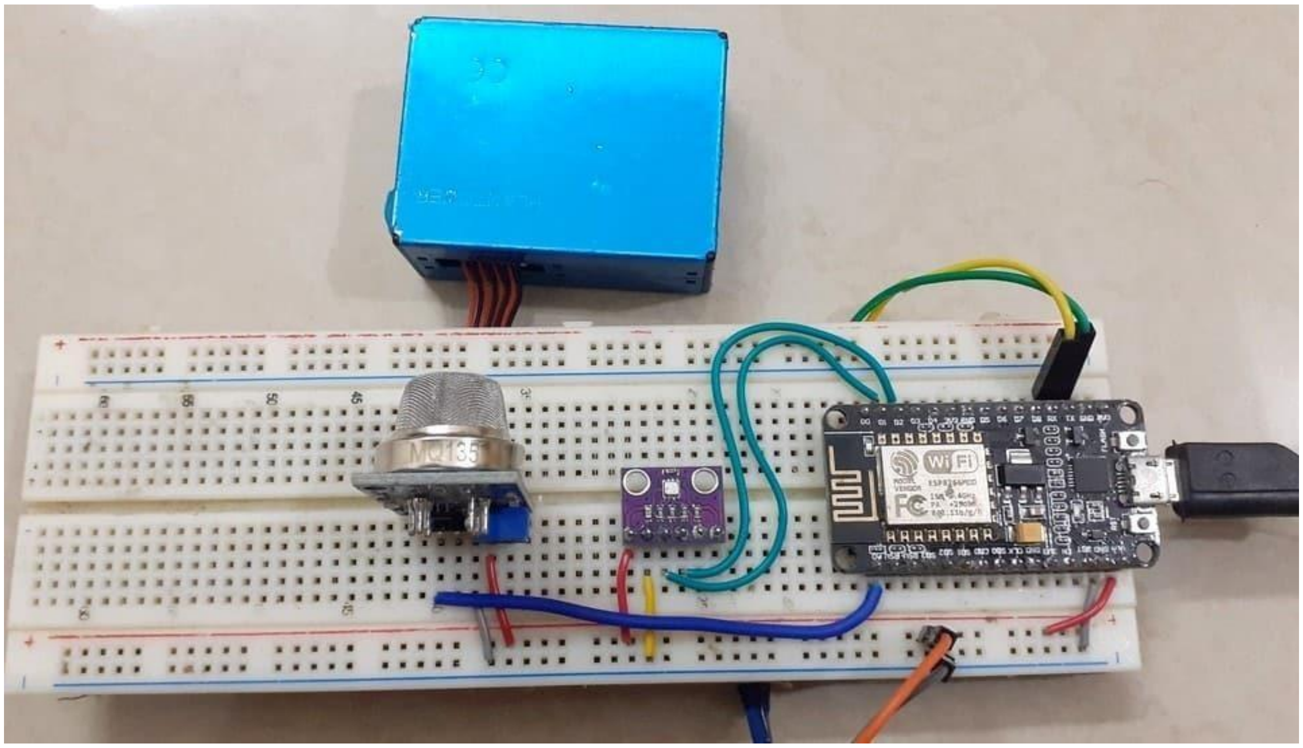
## DIAGRAM:



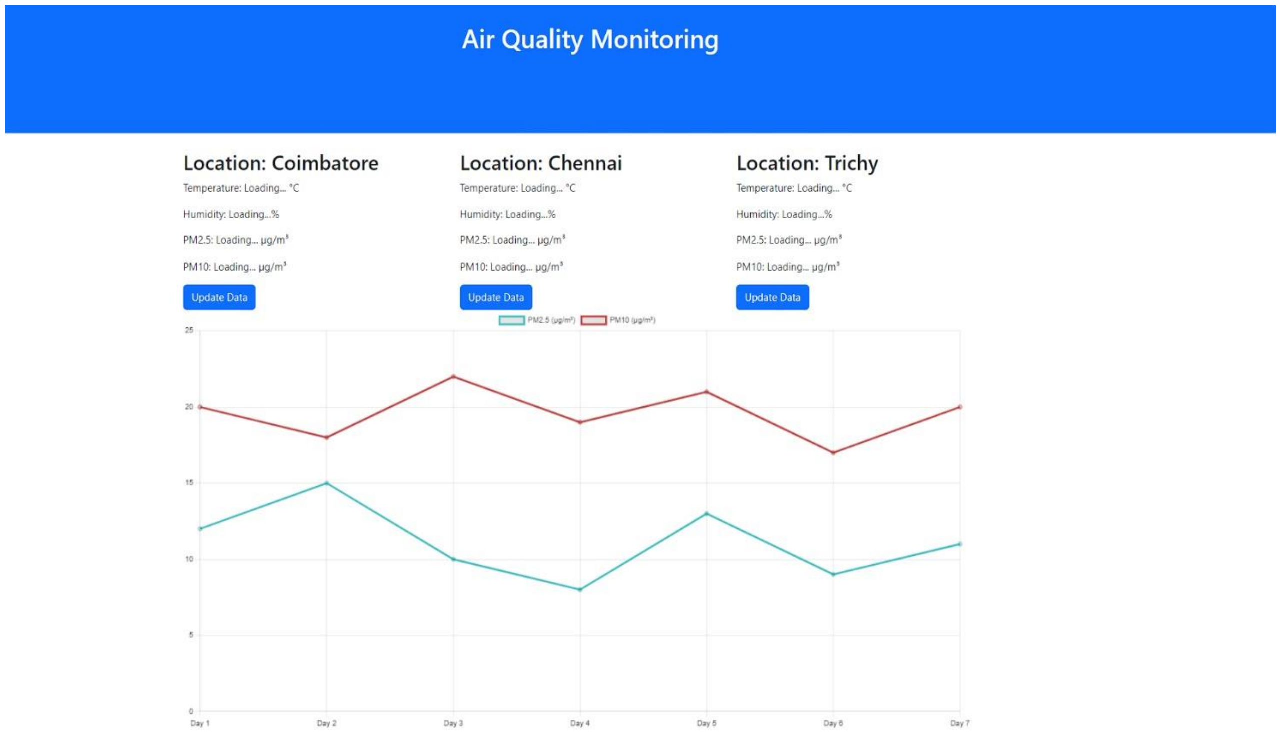
## SCHEMATICS:



**SCREENSHOT OF IOT DEVICES:**



**SCREENSHOT OF DATA SHARING PLATFORM:**



Air Quality Monitoring

Location: Coimbatore

Temperature: Loading... °C

Humidity: Loading...%

PM2.5: Loading... µg/m³

PM10: Loading... µg/m³

Location: Chennai

Temperature: Loading... °C

Humidity: Loading...%

PM2.5: Loading... µg/m³

PM10: Loading... µg/m³

Location: Trichy

Temperature: Loading... °C

Humidity: Loading...%

PM2.5: Loading... µg/m³

PM10: Loading... µg/m³

## <u>REAL TIME AIR QUALITY MONITORING SYSTEM CAN RAISE PUBLIC AWARNESS ABOUT AIR QUALITY AND HEALTH IMPACTS:</u>

A real-time air quality monitoring system can raise public awareness by providing immediate, accurate data on air pollution levels. This system allows people to access real-time information about pollutants in their area, educating them about the current air quality and its potential health impacts. By seeing this data, individuals become more informed about the risks and can take necessary precautions, such as adjusting outdoor activities, using masks, or advocating for policies to improve air quality. This increased awareness often leads to a better understanding of how air quality affects health, prompting individuals and communities to take actions that can help reduce pollution and its health impacts.

## SUBMISSION:

### GITHUB REPOSITORY LINK:

https://github.com/Subhavishnu/AQM

**INSTRUCTION ON HOW TO REPLICATE THE PROJECT:**

Setting up an air quality monitoring project involving IoT devices, a data-sharing platform, and integrating them using Python is a multi-step process. Below are general instructions for each component:

**1. SETTING UP IOT DEVICES:**

Materials Needed:

- Microcontroller (e.g., Raspberry Pi, Arduino)

- Sensors (e.g., PM sensor, CO sensor, etc.)

- Power source

- Internet connectivity (Wi-Fi or Ethernet) **Steps:**

1. Connect Sensors to Microcontroller:

   - Connect the air quality sensors to the microcontroller following their respective datasheets and pin configurations.

2. Program the Microcontroller:

   - Write code to read data from the sensors and transmit it over the internet. For example, if you're using a Raspberry Pi, you can use Python libraries like `gpiozero` or `Adafruit_Python_GPIO` to interface with sensors.

3. Set up Internet Connectivity:

   - Configure Wi-Fi or Ethernet on the microcontroller to enable data transmission.

4. Transmit Data:

   - Use a protocol like MQTT, HTTP, or any suitable communication protocol to send data from the microcontroller to a cloud-based platform or server

**2. Developing the Data-Sharing Platform:**

Options:

- Cloud Platforms (e.g., AWS, Google Cloud, Azure)

- IoT-specific Platforms (e.g., ThingsBoard, Ubidots)

- Custom Web Application with a Database (e.g., using Django or Flask) Steps:

1. Choose a Platform:

   - Select a cloud or IoT platform to host and manage your data. Configure an account.

2. Set Up Data Storage:

   - Create a database to store incoming air quality data. Configure appropriate tables or data structures.

3. Define APIs or Endpoints:

   - Set up APIs or endpoints to receive data from IoT devices. Use protocols like HTTP or MQTT for data ingestion.

4. Authentication and Security:

- Implement authentication mechanisms to ensure secure access to the platform. 5. Dashboard and Visualization:

- Create a dashboard to display real-time and historical air quality data. Use tools like charts or graphs to visualize the data.

**3. Integrating with Python:**

Using Python for Integration:

1. Install Required Libraries:

- Depending on the platform and communication protocol, install necessary Python libraries (e.g., `paho-mqtt` for MQTT communication, or use built-in libraries for HTTP)

2.Connect to the Platform:

- Write Python scripts to connect to the data-sharing platform using APIs or protocols provided by the platform.

3. Data Processing and Analysis:

   - Process the incoming data as per your project requirements. Perform any necessary data cleaning, aggregation, or analysis.

4. Automate Data Retrieval:

- Schedule Python scripts to regularly retrieve data from the platform for realtime monitoring or analysis.

Example Outputs:

**IoT Device Data Transmission:**

- Assuming you're using MQTT for communication, here's a simplified Python example using the `paho-mqtt` library:

**PYTHON:**

```
import paho.mqtt.client as mqtt def
on_connect(client, userdata, flags, rc):
print("Connected with result code "+str(rc))
client.subscribe("air_quality_data") def
on_message(client, userdata, msg):
print(msg.topic+" "+str(msg.payload)) client =
mqtt.Client() client.on_connect = on_connect
client.on_message = on_message
client.connect("mqtt.eclipse.org", 1883, 60)
client.loop_forever()
```

Platform UI (Web Application):

- For visualization, you can create a web application using a framework like Flask or Django. Here's a simplified example using Flask:

**PYTHON:**

```
from flask import Flask, render_template
app = Flask(__name__) @app.route('/')
def dashboard():
```

Retrieve data from the database and pass it to the template Render

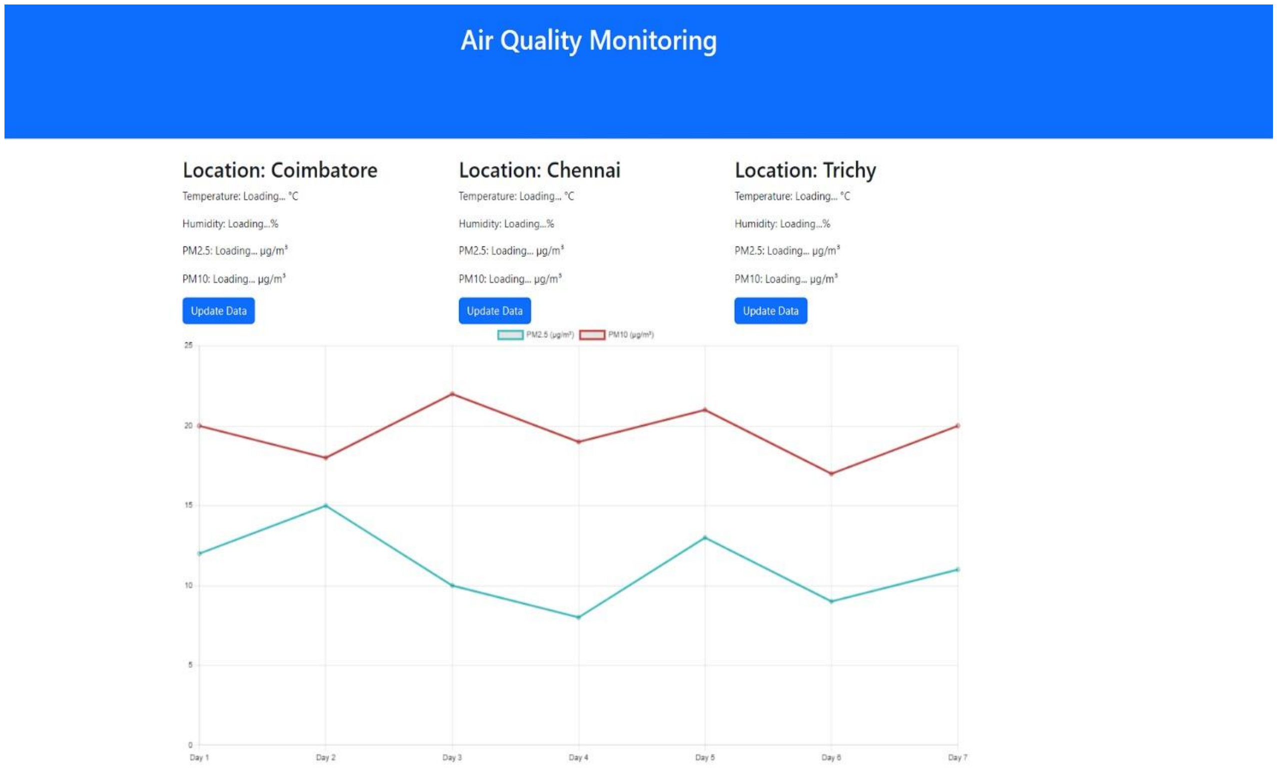
the template with the data

return render_template('dashboard.html', air_quality_data=data) if

__name__ == '__main__':

app.run(debug=True)

**<u>IOT DEVICE DATA TRANSMISSION AND PLATFORM UI:</u>**



Air Quality Monitoring

Location: Coimbatore

Temperature: Loading... °C

Humidity: Loading...%

PM2.5: Loading... μg/m³

PM10: Loading... μg/m³

Location: Chennai

Temperature: Loading... °C

Humidity: Loading...%

PM2.5: Loading... μg/m³

PM10: Loading... μg/m³

Location: Trichy

Temperature: Loading... °C

Humidity: Loading...%

PM2.5: Loading... μg/m³

PM10: Loading... μg/m³