AIR QUALITY MONITORING

Abstract:

Air quality monitoring is a crucial aspect of environmental management, public health, and urban planning. This abstract provides an overview of the significance, methods, and key findings in air quality monitoring. It discusses the importance of continuously assessing air quality to mitigate the harmful effects of air pollutants on human health and the environment. Various monitoring techniques, including remote sensing, ground based stations, and sensor networks, are employed to collect data on air pollutants such as particulate matter, nitrogen dioxide , sulfur dioxide , carbon monoxide , and ozone . Monitoring efforts have revealed spatial and temporal variations in air quality, driven by factors like industrial emissions, traffic, weather conditions, and natural events. This data is crucial for implementing air quality management strategies, setting regulatory standards, and promoting public awareness. As air quality concerns continue to rise, the development of advanced monitoring technologies and data analysis tools remains essential for creating healthier and more sustainable urban environments.

STEPS FOR FLOWCHART:

1. Define the Purpose:

i) Clearly state the purpose of the air quality monitoring process. This could be to measure and record air pollutants, ensure compliance with regulations, or provide real-time air quality information.

2. Start/End:

i) Begin your flowchart with a "Start" symbol, typically represented by an oval or a rectangle with rounded corners.

ii) End the flowchart with an "End" symbol, which is also represented by an oval or a rectangle with rounded corners.

3. Data Collection:

i) Determine how data will be collected. This may involve various sensors, instruments, or monitoring stations.

ii) Use a symbol like a rectangle to represent data collection points.

4. Data Processing:

i) Show how the collected data is processed. This may include data filtering, validation, and conversion.

ii) Use a rectangle with rounded corners to represent data processing steps.

5. Data Analysis:

i) Illustrate the steps involved in analyzing the processed data. This may include calculations, comparisons, or statistical analysis.

ii) Use a diamond-shaped symbol to represent decision points in the analysis.

6. Reporting and Display:

i) Depict how the air quality data is presented to users. This could involve visual displays, reports, or notifications.

ii) Use a rectangle to represent reporting and display steps.

7. Data Storage:

i) Show where the data is stored for future reference and analysis. This may involve databases or data repositories.

ii) Use a cylinder-shaped symbol to represent data storage.

8. Alarm Thresholds:

i) If applicable, indicate how alarm thresholds are set and monitored for abnormal air quality conditions.

ii) Use another decision point (diamond-shaped symbol) to represent threshold checks.

9. Data Transmission:

i) Indicate how the air quality data is transmitted to relevant stakeholders, which may include government agencies, the public, or environmental organizations.

ii) Use an arrow to show the flow of data transmission.

10. Regulatory Compliance:

i) If applicable, include steps for ensuring compliance with air quality regulations and standards.

ii) Use decision points to represent compliance checks.

11. Maintenance and Calibration:

i) Show how the monitoring equipment is maintained and calibrated regularly to ensure accurate measurements.

ii) Use rectangles or other symbols to represent maintenance and calibration activities.

12. Feedback Loop:

i) If necessary, illustrate how the monitoring process incorporates feedback to improve data accuracy and reliability.

ii) Use arrows to represent feedback loops within the flowchart.

13. Documentation:

i) Include a step for documenting the monitoring process, including procedures, data records, and reports.

ii) Use a rectangle to represent documentation.

14. Review and Validation:

i) Show steps for reviewing and validating the collected data to ensure its accuracy and reliability.

ii) Use decision points to represent validation checks.

15. Continuous Monitoring:

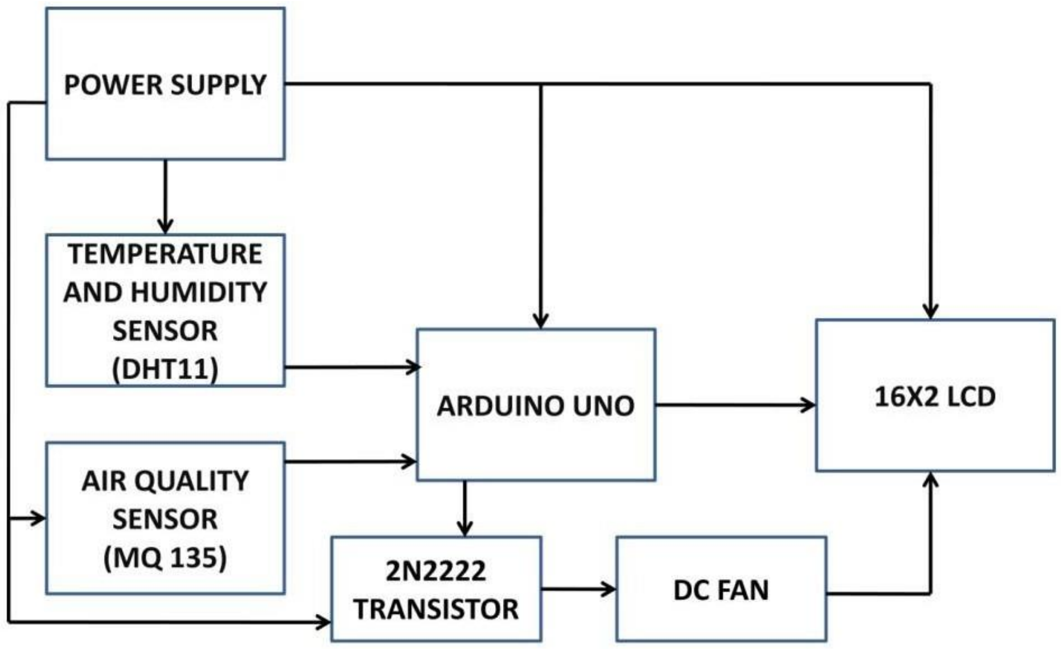
i) Indicate that the monitoring process is continuous and ongoing.

ii) Use a loop symbol or an arrow pointing back to a previous step to represent continuous monitoring.

16. End:

i) Conclude the flowchart with an "End" symbol.

BLOCK DIAGRAM:


POWER SUPPLY
TEMPERATURE AND HUMIDITY SENSOR (DHT11)
AIR QUALITY SENSOR (MQ 135)
ARDUINO UNO
16X2 LCD
2N2222 TRANSISTOR
DC FAN

BLOCK DIAGRAM DISCRIPTION:

AURDUNIO UNO:

Arduino is an open source computer hardware and software company, project, and user community that designs and manufactures single- board microcontrollers and microcontroller kits for building digital devices and interactive objects that can sense and control objects in the physical world.

16X2 LCD Panel:

A liquid-crystal display (LCD) is a flat-panel display or other electronically modulated optical device that uses the light-modulating properties of liquid crystals Liquid crystals do not emit light directly, instead using a backlight or reflector .To produce images in colour or monochrome.[1]LCDs are available to display arbitrary images (as in a general-purpose computer display) or fixed images with low information content, which can be displayed or hidden, such as preset words, digits, and seven-segment displays.

Air Quality Sensor (MQ135):

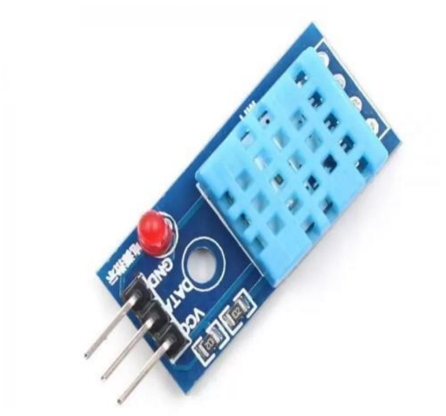
Air quality click is suitable for detecting ammonia (NH3), nitrogen oxides (NOx) benzene, smoke, CO2 and other harmful or poisonous gases that impact air quality. The MQ-135 sensor unit has a sensor layer made of tin dioxide (SnO2), an inorganic compound which has lower conductivity in clean air than when polluting gases are present. To calibrate Air quality, use the on-board potentiometer to adjust the load resistance on the sensor circuit.

Temperature and humidity sensor (DHT11):

DHT11 digital temperature and humidity sensor is a composite Sensor contains a calibrated digital signal output of the temperature and humidity. Application of a dedicated digital modules collection technology and the temperature and humidity sensing technology, to ensure that the product has high reliability and excellent long- term stability. The sensor includes a resistive sense of wet components and an NTC temperature measurement devices, and connected with a high-performance 8-bit microcontroller.

SENSORS:

A) Temperature and Humidity sensors:



B) Gas Sensor:

Definition For Sensors:

Humidity Sensor:

A humidity sensor is an electronic device that measures the humidity in its environment and converts its findings into a corresponding electrical signal. Humidity sensors vary widely in size and fu | A humidity sensor is an electronic device that measures the humidity in its environment and converts its findings into a corresponding electrical signal.

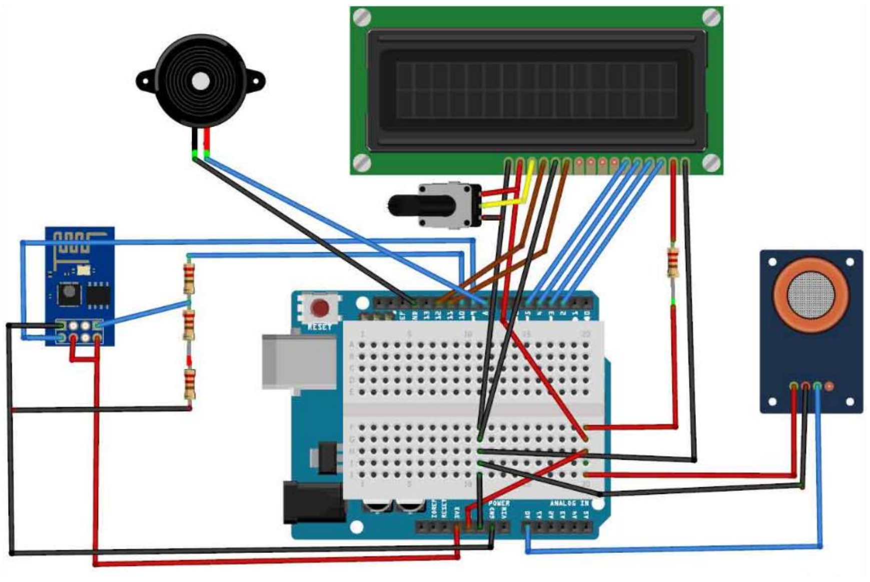
Temperature Sensor:

A temperature sensor is a device used to measure temperature. This can be air temperature, liquid temperature or the temperature of solid matter.

Gas Sensor:

Gas Sensor is the core of the gas detection system and is usually installed in the detection head. Essentially, a gas sensor is a converter that converts a certain gas volume fraction into a corresponding electrical signal.

Design of air quality monitoring system:



Creating a real-time Noise Pollution Monitoring by using HTML Program to Web Development:

1) Index.html:

```
  <!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">

  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css" rel="stylesheet">

  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.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>

            <p>PM10: <span id="pm101">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>
```

2) Script.js:

```
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");



    // Generate random air quality data

    const randomTemperature = (Math.random() * 40 + 10).toFixed(2); // Random temperature between 10 and
50 °C

    const randomHumidity = (Math.random() * 50 + 30).toFixed(2); // Random humidity between 30% and 80%

    const randomPM25 = Math.floor(Math.random() * 100); // Random PM2.5 between 0 and 100 µg/m³

    const randomPM10 = Math.floor(Math.random() * 150); // Random PM10 between 0 and 150 µg/m³




    // Update the displayed data

    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");


    // Generate random air quality data
    const randomTemperature = (Math.random() * 40 + 10).toFixed(2); // Random temperature between 10 and
50 °C
    const randomHumidity = (Math.random() * 50 + 30).toFixed(2); // Random humidity between 30% and 80%
    const randomPM25 = Math.floor(Math.random() * 100); // Random PM2.5 between 0 and 100 µg/m³
    const randomPM10 = Math.floor(Math.random() * 150); // Random PM10 between 0 and 150 µg/m³



    // Update the displayed data
    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");


    // Generate random air quality data
    const randomTemperature = (Math.random() * 40 + 10).toFixed(2); // Random temperature between 10 and
50 °C
```

```
const randomHumidity = (Math.random() * 50 + 30).toFixed(2); // Random humidity between 30% and 80%const
randomPM25 = Math.floor(Math.random() * 100); // Random PM2.5 between 0 and 100 µg/m³ const randomPM10 =
Math.floor(Math.random() * 150); // Random PM10 between 0 and 150 µg/m³

          // Update the displayed data

          location2Element.textContent = "Trichy";

          temperature2Element.textContent = randomTemperature + " &deg;C";

          humidity2Element.textContent = randomHumidity + "%";

          pm252Element.textContent = randomPM25 + " µg/m³";

          pm102Element.textContent = randomPM10 + " µg/m³";

        }

        // Sample historical air quality data

        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,

            },

            {
```

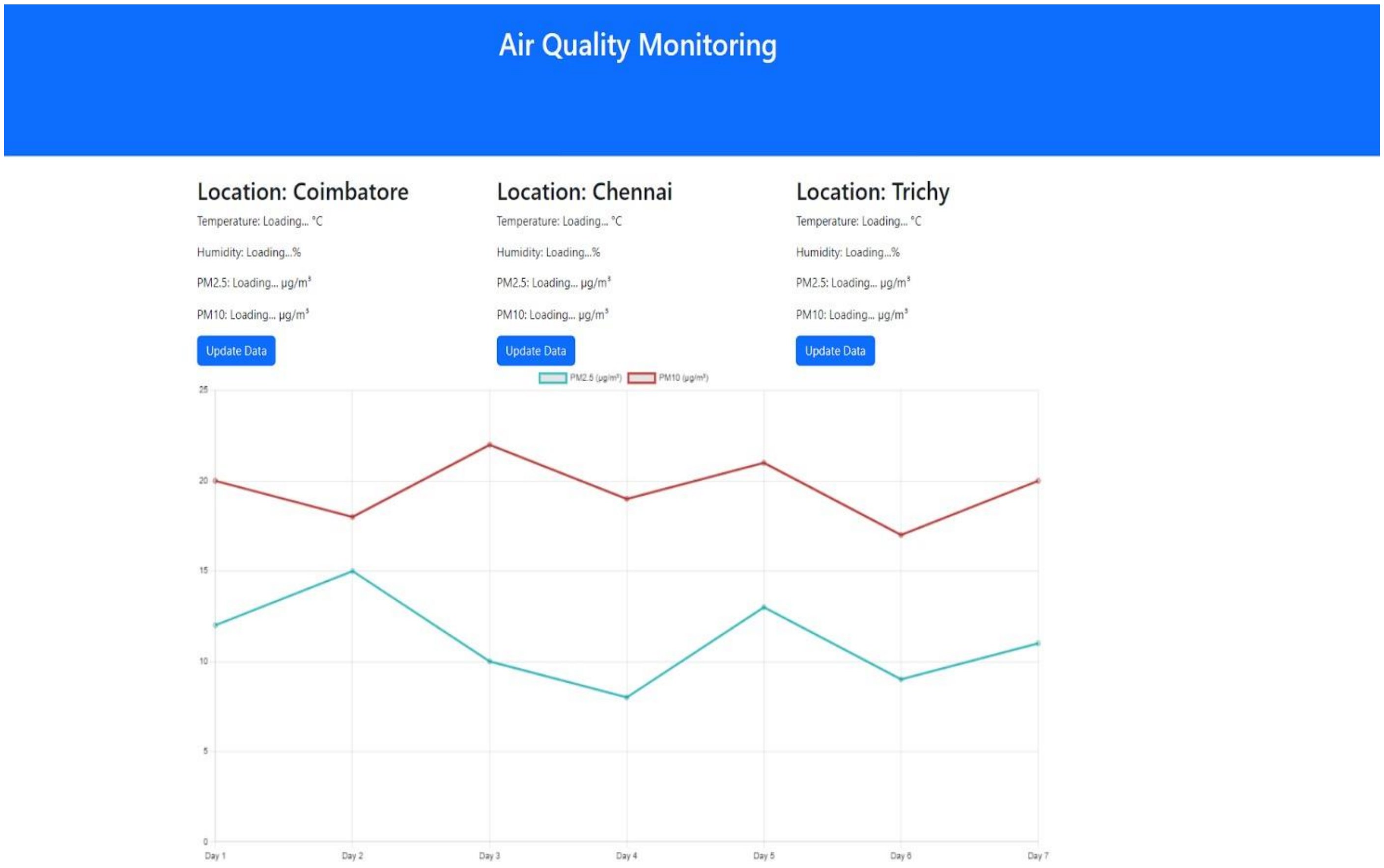
```
        label: "PM10 (µg/m³)",
        data: historicalData.pm10,
        borderColor: "rgba(192, 75, 75, 1)",
        fill: false,
      },
    ],
  },
  options: {
   scales:  {
   x: {
        beginAtZero: true,
      },
      y: {
        beginAtZero: true,
      },
    },
  },
});
function updateAirQualityChart() {
  // Generate new random historical data (for demonstration purposes)
  const newPM25Data = [...historicalData.pm25.map(() => Math.floor(Math.random() * 20 + 10))];
  const newPM10Data = [...historicalData.pm10.map(() => Math.floor(Math.random() * 20 + 10))];


  // Update the chart's data
  airQualityChart.data.datasets[0].data = newPM25Data;
  airQualityChart.data.datasets[1].data = newPM10Data;


  // Update the labels if needed
  airQualityChart.data.labels = historicalData.labels;
```

```
 // Update the chart
 airQualityChart.update();
}
```

OUTPUT:



Air Quality Monitoring

Location: Coimbatore

Temperature: Loading... °C

Humidity: Loading...%

PM2.5: Loading... µg/m³

PM10: Loading... µg/m³

Update Data

Location: Chennai

Temperature: Loading... °C

Humidity: Loading...%

PM2.5: Loading... µg/m³

PM10: Loading... µg/m³

Update Data

Location: Trichy

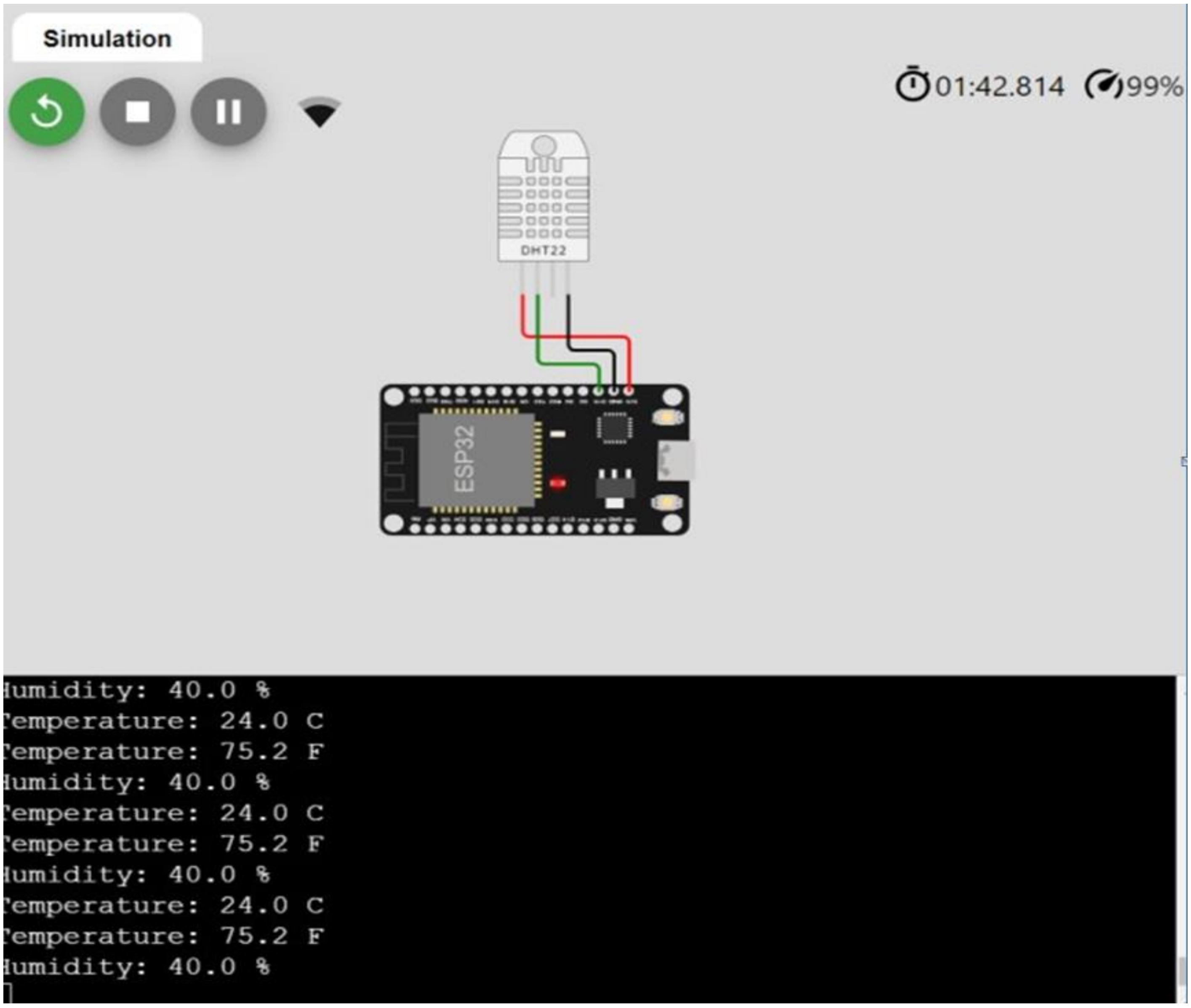
Temperature: Loading... °C

Humidity: Loading...%

PM2.5: Loading... µg/m³

PM10: Loading... µg/m³

Update Data



```
Humidity: 40.0 %
Temperature: 24.0 C
Temperature: 75.2 F
Humidity: 40.0 %
Temperature: 24.0 C
Temperature: 75.2 F
Humidity: 40.0 %
Temperature: 24.0 C
Temperature: 75.2 F
Humidity: 40.0 %
```

PROGRAM:

Import time
import Adafruit_DHT # If using a DHT sensor import
RPi.GPIO as GPIO # If using a Raspberry Pi

 # Initialize the sensors
 # For gas sensors, you will need specific libraries for your sensors
 # The following is an example for a DHT22 temperature and humidity sensorDHT_SENSOR = Adafruit_DHT.DHT22
 DHT_PIN = 4 # GPIO pin number where the DHT sensor is connected

 # Initialize IoT connectivity (e.g., MQTT, HTTP, etc.)# You'll
 need an IoT platform to send data to

```
def read_gas_sensor():
    # Use your gas sensor library to read values#
    Example:
    gas_value = your_gas_sensor.read_value()return
    gas_value

def read_temperature_humidity():
    # Read temperature and humidity from the DHT sensor
    humidity, temperature = Adafruit_DHT.read(DHT_SENSOR, DHT_PIN)return
    temperature, humidity

def send_data_to_iot_platform(data):
    # Implement IoT platform communication here# Example:
    MQTT, HTTP, etc.
    pass

while True:
    gas_data = read_gas_sensor()
    temperature, humidity = read_temperature_humidity()
    # Create a JSON payload with the sensor datasensor_data =
    {
        "gas_value": gas_data,
        "temperature": temperature,
        "humidity": humidity
    }

    # Send the data to the IoT platform
    send_data_to_iot_platform(sensor_data)


    time.sleep(60)  # Adjust the interval as needed
```

WOWKI CODE:

```
from machine import Pinfrom time
import sleep import dht
import network
sta_if = network.WLAN(network.STA_IF)if not
```

```
sta_if.isconnected():
    print('connecting to network...')
    sta_if.active(True) sta_if.connect('Wokwi-
    GUEST', "")while not sta_if.isconnected():
        pass
    print('network config:', sta_if.ifconfig())

sensor = dht.DHT22(Pin(15))while
True:
  try:
    sleep(2) sensor.measure()
    temp = sensor.temperature() hum =
    sensor.humidity() temp_f = temp * (9/5)
    + 32.0
    print('Temperature: %3.1f C' %temp)
  print('Temperature: %3.1f F' %temp_f)
  print('Humidity: %3.1f %%' %hum) except OSError
  as e:
    print('Failed to read sensor.')
```