ENVIRONMENTAL MONITORING

ABSTRACT:

Environmental monitoring is the process of collecting and analyzing data about the environment to identify and assess environmental changes. It is essential for protecting human health and the environment, and for making informed decisions about environmental management. IoT-based environmental monitoring systems typically consist of a network of sensors that collect data about various environmental parameters, such as air quality, water quality, temperature, humidity, noise levels, and soil conditions. The sensors are connected to a central hub or gateway, which transmits the data to a cloud-based platform for storage and analysis. The data can then be accessed from anywhere in the world via a web browser or mobile app.

DESIGNING SYSTEM:

❖Sensors: Temperature, humidity

❖Device: Microcontroller with a Wi-Fi module

❖Cloud platform: MQTT broker

❖Software: Python script to collect data from the sensors and publish it to the MQTT broker

❖Alerts and notifications: Email or SMS notifications when the sensor readings exceed a certain threshold This system can be easily scaled by adding more sensors and devices to the microcontroller. The data from the sensors can be visualized and analyzed using a variety of cloud-based tools.

PROBLEM:

Traditional environmental monitoring systems are often expensive, complex, and difficult to deploy. They may require manual data collection and analysis, which can be time-consuming and error-prone. Additionally, these systems may not be able to provide real-time data, which can limit their effectiveness in responding to environmental incidents.

Benefits of IoT-based environmental monitoring:

❖Reduced costs: IoT devices are typically much less expensive than traditional environmental monitoring equipment. Additionally, IoT systems can reduce labor costs by automating data collection and analysis.

❖Increased efficiency: IoT systems can provide real-time data on environmental conditions, which can help organizations to identify and respond to problems quickly.

❖Improved accuracy: IoT systems can collect data from a wide range of sensors, which can provide a more complete picture of environmental conditions.

❖Enhanced decision-making: IoT systems can provide insights into environmental trends, which can help organizations to make better decisions about resource management and environmental protection. Use cases for IoT-based environmental monitoring:

❖Air quality monitoring: IoT devices can be used to monitor air quality in cities, factories, and other indoor and outdoor environments. This data can be used to identify and address sources of air pollution, and to protect public health.

❖Water quality monitoring: IoT devices can be used to monitor water quality in rivers, lakes, and other water bodies. This data can be used to identify and address sources of water pollution, and to protect public health and the environment.

❖Soil quality monitoring: IoT devices can be used to monitor soil quality in agricultural fields and other outdoor environments. This data can be used to improve crop yields and reduce environmental impact.

❖Natural disaster monitoring: IoT devices can be used to monitor environmental conditions that could lead to natural disasters, such as floods, wildfires, and earthquakes. This data can be used to warn people of impending danger and to help them to evacuate safely.

Additional challenges

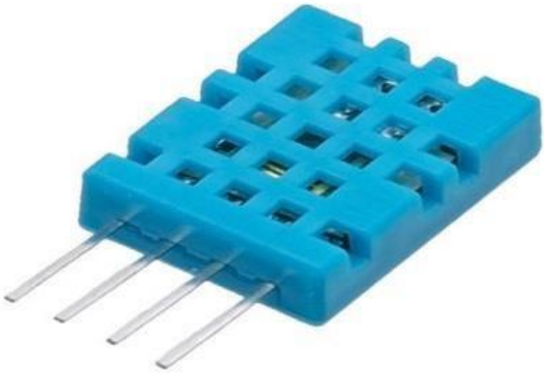
❖While IoT-based environmental monitoring systems offer a number of benefits, there are also some challenges that need to be addressed.

❖Security: IoT devices can be vulnerable to cyberattacks, so it is important to implement appropriate security measures.

❖Data privacy: IoT systems collect a lot of data, so it is important to protect the privacy of this data.

❖Scalability: IoT systems can be complex and difficult to scale, so it is important to choose a system that can meet the specific needs of the organization.

Sensors:



1. DH11 sensor (Temperature and humidity sensor)

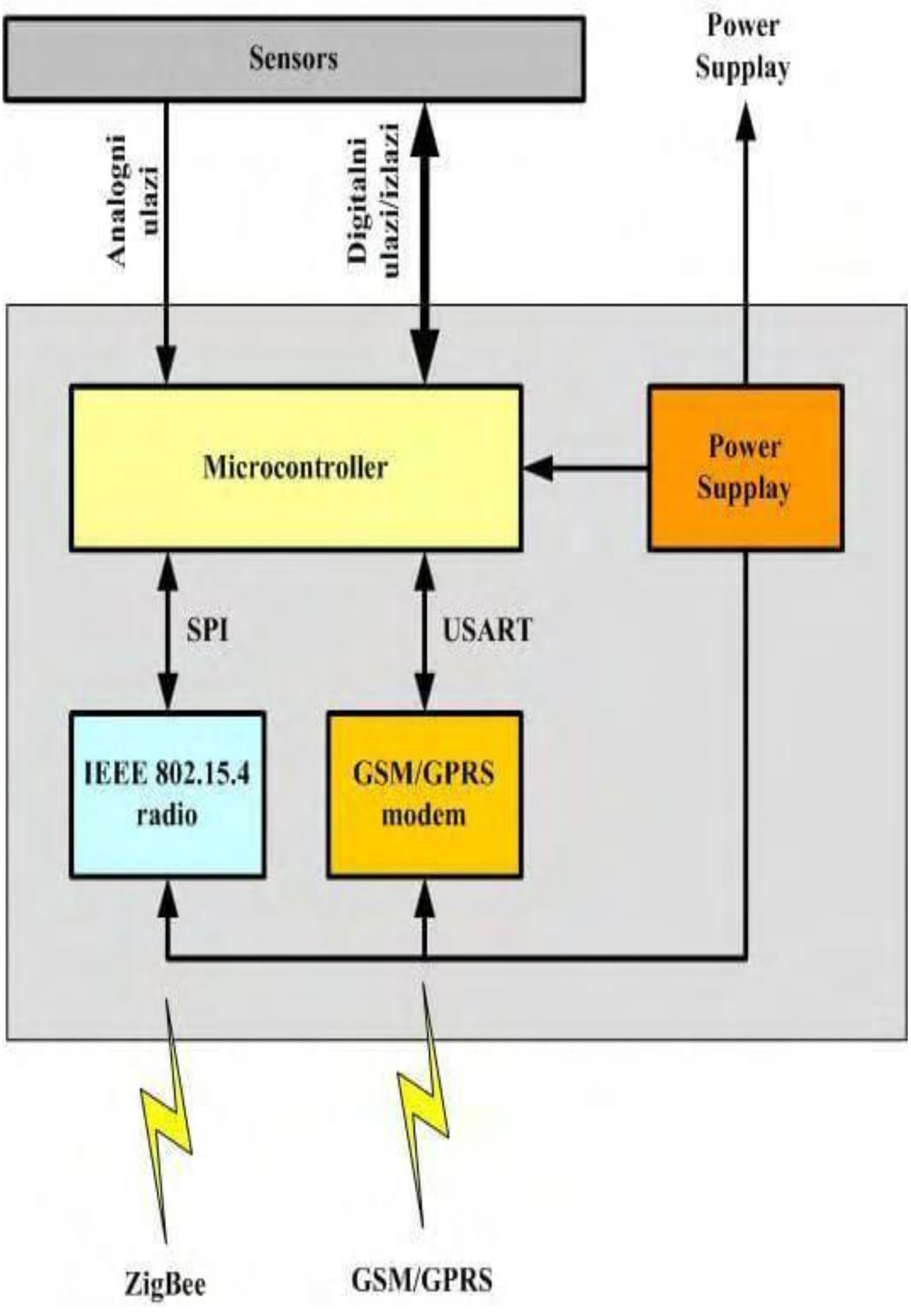
1.TEMPERATURE AND HUMIDITY SENSOR:

These sensors have been designed for various applications to measure the humidity as well as the temperature of the environment.

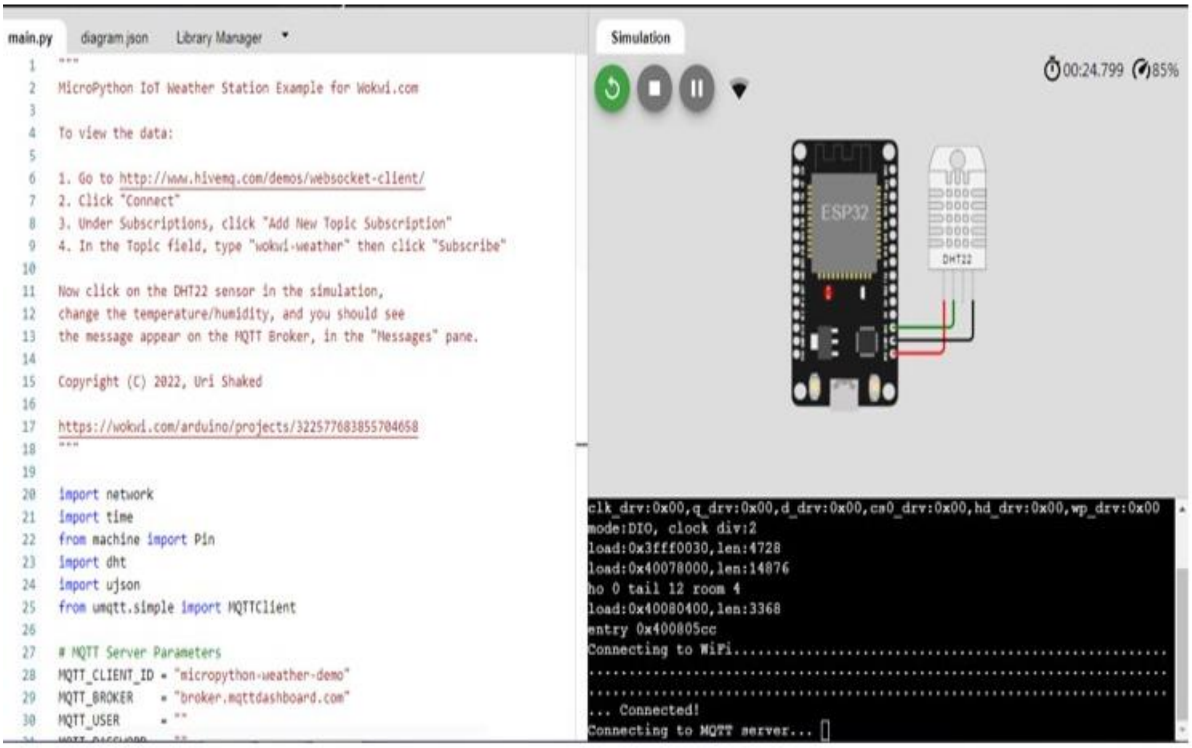
2. MICROPROCESSOR:

A microcontroller is a compact integrated circuit designed to govern a specific operation in an embedded system. A typical microcontroller includes a processor, memory and input/output (I/O) peripherals on a single chip.

Block diagram



Screenshot of the IOT devices :



```
"""
MicroPython IoT Weather Station Example for Wokwi.com

To view the data:

1. Go to http://www.hivemq.com/demos/websocket-client/
2. Click "Connect"
3. Under Subscriptions, click "Add New Topic Subscription"
4. In the Topic field, type "wokwi-weather" then click "Subscribe"

Now click on the DHT22 sensor in the simulation,
change the temperature/humidity, and you should see
the message appear on the MQTT Broker, in the "Messages" pane.

Copyright (C) 2022, Uri Shaked

https://wokwi.com/arduino/projects/322577683855704658
"""


import network
import time
from machine import Pin
import dht
import ujson
from umqtt.simple import MQTTClient

# MQTT Server Parameters
MQTT_CLIENT_ID = "micropython-weather-demo"
MQTT_BROKER    = "broker.mqttdashboard.com"
MQTT_USER      = ""
```

```
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:2
load:0x3fff0030,len:4728
load:0x40078000,len:14876
ho 0 tail 12 room 4
load:0x40080400,len:3368
entry 0x400805cc
Connecting to WiFi.......................................................
.........................................................................
.........................................................................
... Connected!
Connecting to MQTT server...
```

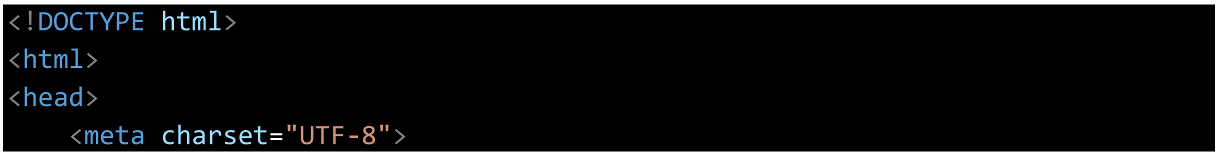
Program:

PYTHON CODE

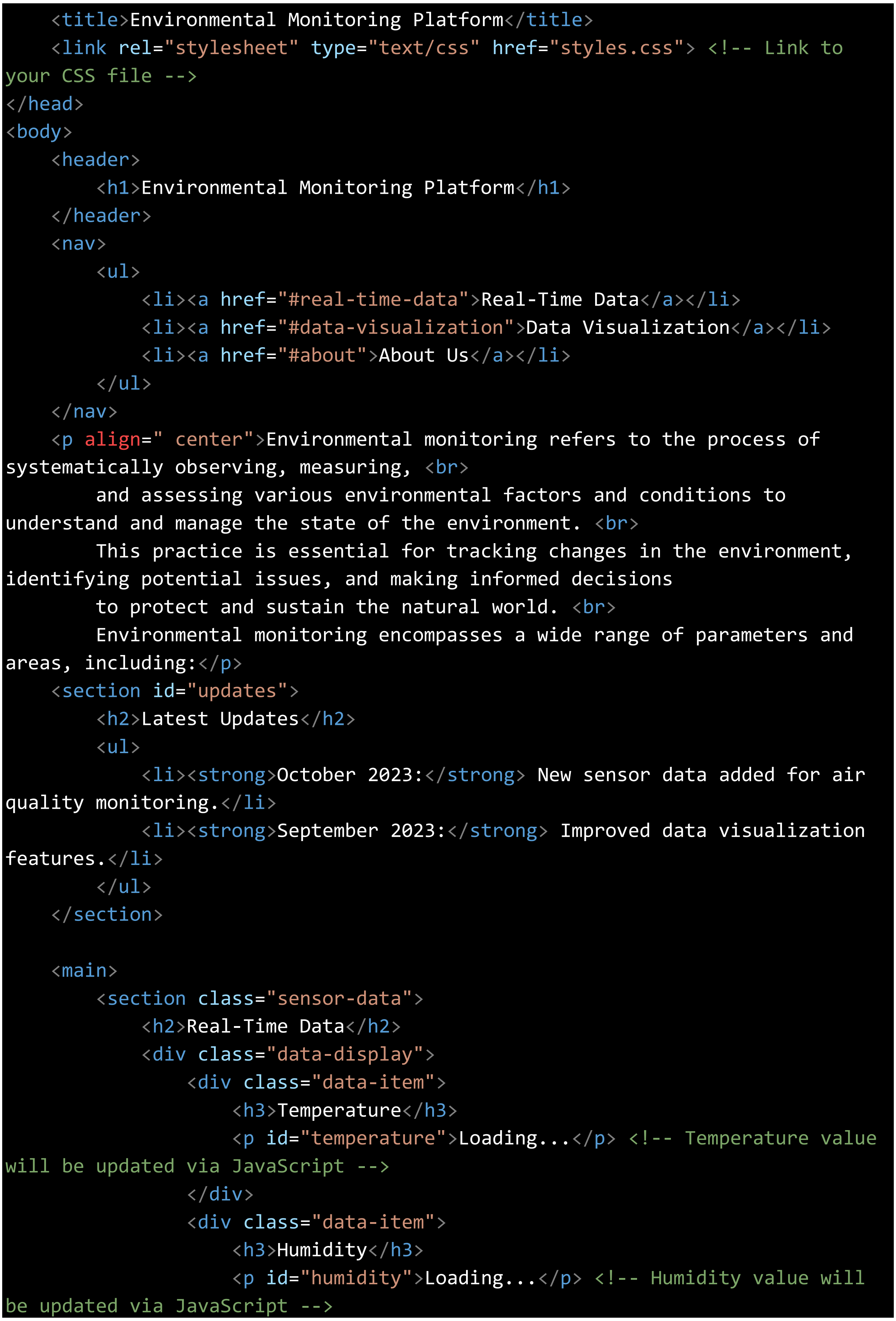
```
import Adafruit_DHT

import time


# Sensor should be set to Adafruit_DHT.DHT11,

# Adafruit_DHT.DHT22, or Adafruit_DHT.AM2302.

DHT_SENSOR = Adafruit_DHT.DHT11

DHT_PIN = 4  # GPIO pin the sensor is connected to


def get_temperature_humidity():

    humidity, temperature = Adafruit_DHT.read_retry(DHT_SENSOR, DHT_PIN)

    if humidity is not None and temperature is not None:
        return temperature, humidity
    else:
        return None, None


def log_data(temperature, humidity):
    with open("environment_log.txt", "a") as file:
        current_time = time.strftime('%Y-%m-%d %H:%M:%S')
        log_entry = f"{current_time} - Temperature: {temperature}°C, Humidity: {humidity}%\n"
        file.write(log_entry)


if __name__ == '__main__':
    while True:
        temperature, humidity = get_temperature_humidity()
        if temperature is not None and humidity is not None:
            log_data(temperature, humidity)
            print(f"Temperature: {temperature}°C, Humidity: {humidity}%")
        else:
            print("Failed to retrieve data from the sensor.")
        time.sleep(60)  # Adjust this time to set the interval for data collection.
```

Program file for web platform:

1.Html Code: (index.html)



```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
```



```
    <title>Environmental Monitoring Platform</title>
    <link rel="stylesheet" type="text/css" href="styles.css"> <!-- Link to
your CSS file -->
</head>
<body>
    <header>
        <h1>Environmental Monitoring Platform</h1>
    </header>
    <nav>
        <ul>
            <li><a href="#real-time-data">Real-Time Data</a></li>
            <li><a href="#data-visualization">Data Visualization</a></li>
            <li><a href="#about">About Us</a></li>
        </ul>
    </nav>
    <p align=" center">Environmental monitoring refers to the process of
systematically observing, measuring, <br>
        and assessing various environmental factors and conditions to
understand and manage the state of the environment. <br>
        This practice is essential for tracking changes in the environment,
identifying potential issues, and making informed decisions
        to protect and sustain the natural world. <br>
        Environmental monitoring encompasses a wide range of parameters and
areas, including:</p>
    <section id="updates">
        <h2>Latest Updates</h2>
        <ul>
            <li><strong>October 2023:</strong> New sensor data added for air
quality monitoring.</li>
            <li><strong>September 2023:</strong> Improved data visualization
features.</li>
        </ul>
    </section>

    <main>
        <section class="sensor-data">
            <h2>Real-Time Data</h2>
            <div class="data-display">
                <div class="data-item">
                    <h3>Temperature</h3>
                    <p id="temperature">Loading...</p> <!-- Temperature value
will be updated via JavaScript -->
                </div>
                <div class="data-item">
                    <h3>Humidity</h3>
                    <p id="humidity">Loading...</p> <!-- Humidity value will
be updated via JavaScript -->
```

```html
        </div>
      </div>
    </section>
    <section class="data-visualization">
      <h2>Data Visualization</h2>
      <canvas id="chart"></canvas> <!-- Add data visualization elements here, e.g., charts or graphs -->
    </section>
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
    <canvas id="lineChart"></canvas>

  </main>
  <section id="about">
    <h2>About Us</h2>
    <p>Welcome to the Environmental Monitoring Platform, where we provide real-time data and visualizations to help you monitor environmental conditions. Our mission is to promote sustainability and environmental awareness.</p>
  </section>
  <section id="contact">
    <h2>Contact Us</h2>
    <p>If you have any questions or feedback, please feel free to contact us at <a href="mailto:contact@example.com">contact@example.com</a>.</p>
  </section>
  <section id="disclaimer">
    <h2>Disclaimer</h2>
    <p>Information provided on this platform is for informational purposes only. Please consult with experts for critical decisions related to the environment.</p>
  </section>
  <footer>
    <p>&copy; 2023 Environmental Monitoring Platform</p>
  </footer>
  <script src="node.js"></script> <!-- Link to your JavaScript file for real-time updates -->
</body>
</html>
```

2.CSS Code : (Styles.css)

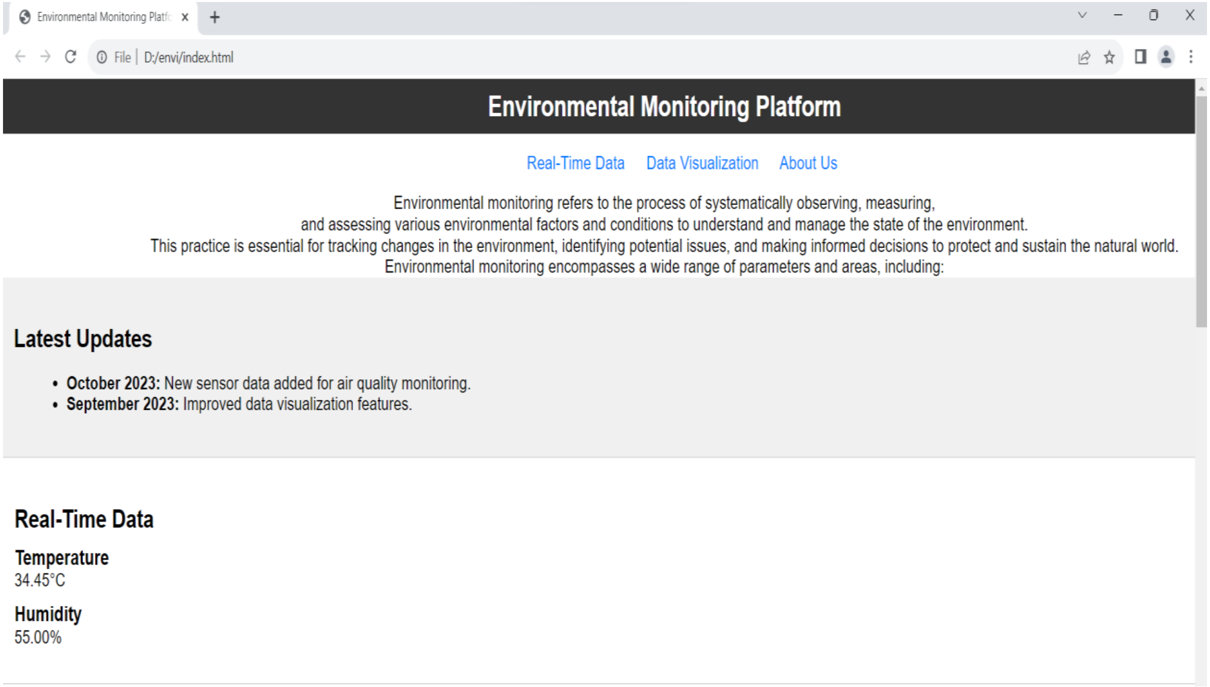
```css
/* Reset some default styles to ensure consistency across browsers */
html, body, h1, h2, h3, p {
  margin: 0;
  padding: 0;
}

body {
    font-family: Arial, sans-serif; /* Set the default font for the entire
page */
}

header {
    background-color: #333; /* Set a dark background color for the header */
    color: #fff; /* Set text color to white */
    padding: 10px;
    text-align: center;
}

h1 {
    font-size: 24px;
}

nav ul {
    list-style: none;
    text-align: center;
}

nav ul li {
    display: inline;
    margin-right: 20px;
}

nav a {
    text-decoration: none;
    color: #007bff;
}

nav a:hover {
    text-decoration: underline;
}

#updates {
    background-color: #f0f0f0; /* Set a light background color for the "Latest
Updates" section */
    padding: 20px;
}

#updates h2 {
    font-size: 22px;
}
```

```
.main-content {
    padding: 20px;
}

.sensor-data {
    border: 1px solid #ccc;
    padding: 20px;
}

.data-item {
    margin: 10px 0;
}

.data-item h3 {
    font-size: 18px;
}

.data-visualization {
    margin-top: 20px;
}

canvas#lineChart {
    width: 100%; /* Make the chart fill the available space */
    height: 300px; /* Set a fixed height for the chart */
}

h2 {
    font-size: 22px;
    margin-top: 20px;
}

ul {
    list-style-type: disc;
    margin-left: 20px;
}

footer {
    background-color: #333;
    color: #fff;
    text-align: center;
    padding: 10px;
}

/* Add specific styles for other sections like "About Us," "Contact Us," and
"Disclaimer" if needed */
```

3.Java Script : (node.js)

```
// Function to update the temperature and humidity values
function updateSensorData() {
    // Simulate data retrieval (you would replace this with actual data)
    const temperatureValue = Math.random() * 30 + 10; // Random temperature
between 10 and 40
    const humidityValue = Math.random() * 60 + 40; // Random humidity between
40 and 100

    // Update the HTML elements with the new values
    document.getElementById("temperature").textContent =
temperatureValue.toFixed(2) + "°C";
    document.getElementById("humidity").textContent = humidityValue.toFixed(2)
+ "%";
}

// Function to create a random chart for data visualization (you can replace
this with a real chart library)
function createRandomChart() {
    const ctx = document.getElementById("chart").getContext("2d");

    const data = {
        labels: ["Jan", "Feb", "Mar", "Apr", "May"],
        datasets: [
            {
                label: "Temperature",
                data: [Math.random() * 10 + 20, Math.random() * 10 + 20,
Math.random() * 10 + 20, Math.random() * 10 + 20, Math.random() * 10 + 20],
                borderColor: "#FF5733",
                borderWidth: 2,
            },
            {
                label: "Humidity",
                data: [Math.random() * 20 + 80, Math.random() * 20 + 80,
Math.random() * 20 + 80, Math.random() * 20 + 80, Math.random() * 20 + 80],
                borderColor: "#33FF57",
                borderWidth: 2,
            },
        ],
    };

    const config = {
        type: "line",
        data: data,
    };
    new Chart(ctx, config);
}

// Update sensor data and chart every 5 seconds (you can adjust the interval)
setInterval(() => {
    updateSensorData();
    createRandomChart();
}, 5000);

// Call the update functions on page load
updateSensorData();
createRandomChart();
// Function to create a line chart using Chart.js
function createLineChart() {
    const ctx = document.getElementById("lineChart").getContext("2d");

    // Sample data (replace with your actual data)
    const data = {
        labels: ["Jan", "Feb", "Mar", "Apr", "May"],
        datasets: [
            {
                label: "Temperature (°C)",
                data: [15, 18, 20, 22, 25],
                borderColor: "#FF5733",
                borderWidth: 2,
                fill: false,
            },
            {
                label: "Humidity (%)",
                data: [45, 42, 40, 38, 35],
                borderColor: "#33FF57",
                borderWidth: 2,
                fill: false,
            },
        ],
    };

    const options = {
        scales: {
            x: {
                type: "category",
                labels: data.labels,
            },
            y: {
                beginAtZero: true,
                max: 30, // Set the maximum value for the y-axis
            },
        },
    };

    const config = {
        type: "line",
        data: data,
        options: options,
    };

    new Chart(ctx, config);
}

// Call the createLineChart function when the document is ready
document.addEventListener("DOMContentLoaded", createLineChart);
```

Result:



Environmental Monitoring Platform

Environmental monitoring refers to the process of systematically observing, measuring,
and assessing various environmental factors and conditions to understand and manage the state of the environment.
This practice is essential for tracking changes in the environment, identifying potential issues, and making informed decisions to protect and sustain the natural world.
Environmental monitoring encompasses a wide range of parameters and areas, including:

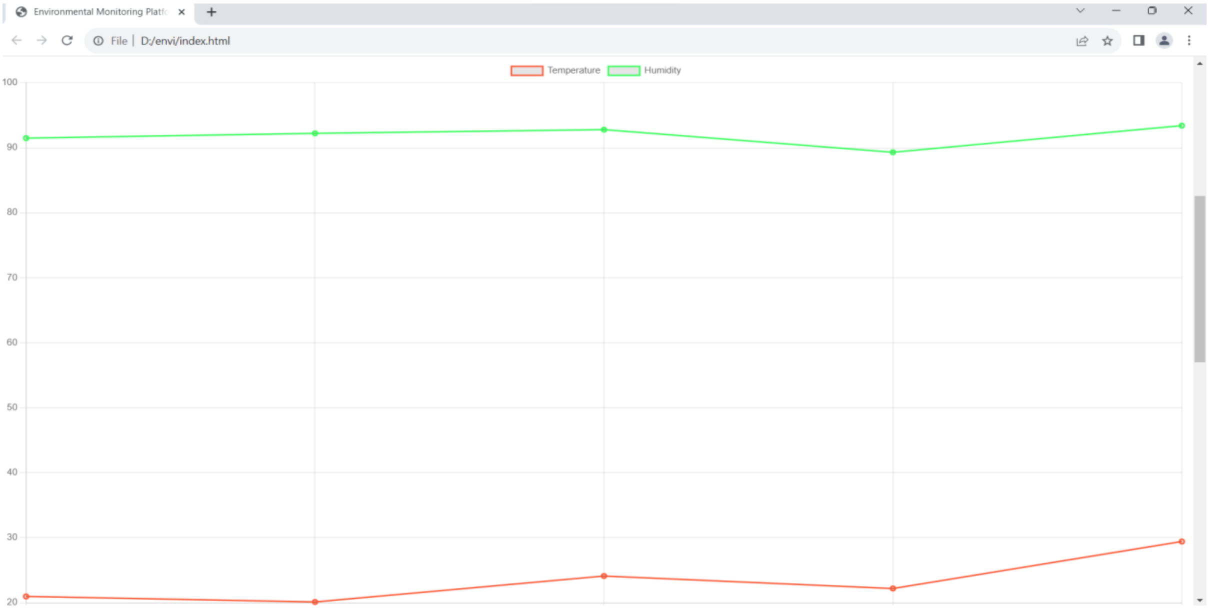
Latest Updates

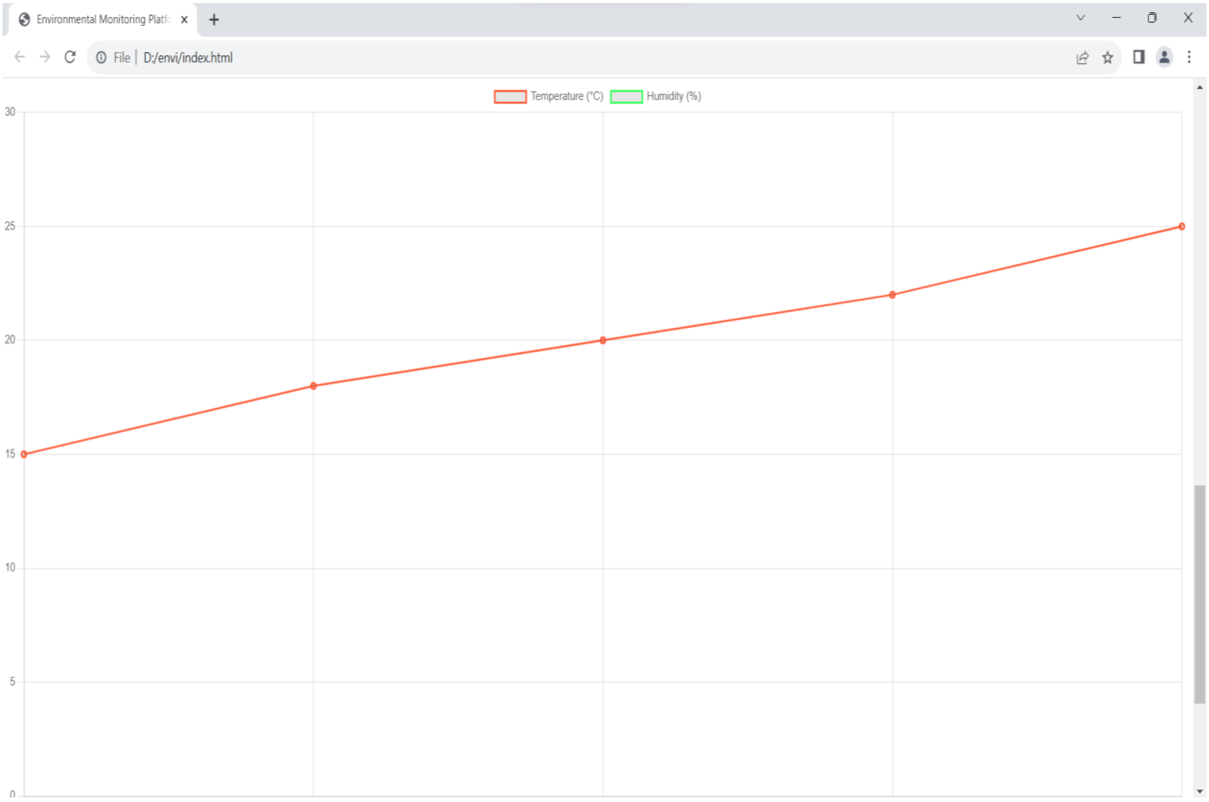
- **October 2023:** New sensor data added for air quality monitoring.
- **September 2023:** Improved data visualization features.

Real-Time Data

Temperature
34.45°C

Humidity
55.00%


Environmental Monitoring Platf
File | D:/envi/index.html
Temperature
Humidity
100
90
80
70
60
50
40
30
20


Environmental Monitoring Platf
File | D:/envi/index.html
Temperature (°C)
Humidity (%)
30
25
20
15
10
5
0

Conclusion:

In conclusion, an Environmental Monitoring System using the Internet of Things (IoT) represents a transformative and highly valuable technology for addressing a wide range of environmental challenges. This system monitor the temperature and humidity of the environment.