**PROJECT TITLE: ENVIRONMENTAL MONITORING**

**PHASE- 4 DEVELOPMENT**

**Creating a real-time Environment Management platform involves a combination of front end and backend technologies. Here’s a simplified outline using C and C++ and python programming with wi-fi connection for the front end and Node.js for the back end:**

**Python:**

import network

import time

from machine import Pin,ADC

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      = ""

MQTT\_PASSWORD  = ""

MQTT\_TOPIC     = "wokwi-weather"

sensor = dht.DHT22(Pin(15))

MQ7=ADC(Pin(35))

MQ8=ADC(Pin(32))

button=Pin(34,Pin.IN)

led=Pin(33,Pin.OUT)

min\_rate=0

max\_rate=4095

print("Connecting to WiFi", end="")

sta\_if = network.WLAN(network.STA\_IF)

sta\_if.active(True)

sta\_if.connect('Wokwi-GUEST', '')

while not sta\_if.isconnected():

  print(".", end="")

  time.sleep(0.1)

print(" Connected!")

print("Connecting to MQTT server... ", end="")

client = MQTTClient(MQTT\_CLIENT\_ID, MQTT\_BROKER, user=MQTT\_USER, password=MQTT\_PASSWORD)

client.connect()

print("Connected!")

prev\_weather = ""

while True:

  CO\_sensor=(MQ7.read())\*100/(max\_rate)

  print("CO Sensor value: " + "%.2f" % CO\_sensor +"%")

  Hydrogen\_sensor=(MQ8.read())\*100/(max\_rate)

  print("Soil Sensor value: " + "%.2f" % Hydrogen\_sensor +"%")

  button\_value=button.value()

  if button\_value == True:

    led.value(000)

    print("It's Raining")

  else:

    led.value(0)

  print("Measuring weather conditions... ", end="")

  sensor.measure()

  message = ujson.dumps({

    "temp": sensor.temperature(),

    "humidity": sensor.humidity(),

  })

  if message != prev\_weather:

    print("Updated!")

    print("Reporting to MQTT topic {}: {}".format(MQTT\_TOPIC, message))

    client.publish(MQTT\_TOPIC, message)

    prev\_weather = message

  else:

    print("No change")

  time.sleep(1)

**C++:**

#include <Adafruit\_Sensor.h>

#include <DHT.h>

#include <ESP8266WiFi.h>

#include <Adafruit\_MQTT.h>

#include <Adafruit\_MQTT\_Client.h>

// Replace these with your Wi-Fi credentials.

const char\* WIFI\_SSID = "YourWiFiSSID";

const char\* WIFI\_PASS = "YourWiFiPassword";

// Replace with your Adafruit IO credentials.

#define ADAFRUIT\_IO\_USERNAME "YourAdafruitUsername"

#define ADAFRUIT\_IO\_KEY "YourAdafruitAIOKey"

// Define the DHT sensor.

#define DHT\_PIN 2           // The pin where your DHT sensor is connected.

#define DHT\_TYPE DHT22      // DHT sensor type (DHT11, DHT22, AM2302, etc.)

DHT dht(DHT\_PIN, DHT\_TYPE);

WiFiClient client;

Adafruit\_MQTT\_Client mqtt(&client, "io.adafruit.com", 1883, ADAFRUIT\_IO\_USERNAME, ADAFRUIT\_IO\_KEY);

// Define MQTT feeds.

Adafruit\_MQTT\_Publish temperature = Adafruit\_MQTT\_Publish(&mqtt, ADAFRUIT\_IO\_USERNAME "/feeds/temperature");

Adafruit\_MQTT\_Publish humidity = Adafruit\_MQTT\_Publish(&mqtt, ADAFRUIT\_IO\_USERNAME "/feeds/humidity");

void setup() {

  Serial.begin(115200);

  // Connect to Wi-Fi.

  WiFi.begin(WIFI\_SSID, WIFI\_PASS);

  while (WiFi.status() != WL\_CONNECTED) {

    delay(500);

    Serial.println("Connecting to WiFi...");

  }

  Serial.println("Connected to WiFi");

  // Connect to Adafruit IO.

  mqtt.connect();

  Serial.println("Connected to Adafruit IO");

}

void loop() {

  // Read temperature and humidity data from the DHT sensor.

  float temperatureValue = dht.readTemperature();

  float humidityValue = dht.readHumidity();

  // Publish data to Adafruit IO.

  if (!isnan(temperatureValue)) {

    temperature.publish(temperatureValue);

    Serial.print("Temperature: ");

    Serial.println(temperatureValue);

  } else {

    Serial.println("Failed to read temperature");

  }

  if (!isnan(humidityValue)) {

    humidity.publish(humidityValue);

    Serial.print("Humidity: ");

    Serial.println(humidityValue);

  } else {

    Serial.println("Failed to read humidity");

  }

  delay(60000); // Delay for 60 seconds (adjust as needed).

}

**C program:**

#include <Adafruit\_Sensor.h>

#include <DHT.h>

#include <ESP8266WiFi.h>

#include <Adafruit\_MQTT.h>

#include <Adafruit\_MQTT\_Client.h>

// Replace these with your Wi-Fi credentials.

const char\* WIFI\_SSID = "YourWiFiSSID";

const char\* WIFI\_PASS = "YourWiFiPassword";

// Replace with your Adafruit IO credentials.

#define ADAFRUIT\_IO\_USERNAME "YourAdafruitUsername"

#define ADAFRUIT\_IO\_KEY "YourAdafruitAIOKey"

// Define the DHT sensor.

#define DHT\_PIN 2           // The pin where your DHT sensor is connected.

#define DHT\_TYPE DHT22      // DHT sensor type (DHT11, DHT22, AM2302, etc.)

DHT dht(DHT\_PIN, DHT\_TYPE);

WiFiClient client;

Adafruit\_MQTT\_Client mqtt(&client, "io.adafruit.com", 1883, ADAFRUIT\_IO\_USERNAME, ADAFRUIT\_IO\_KEY);

// Define MQTT feeds.

Adafruit\_MQTT\_Publish temperature = Adafruit\_MQTT\_Publish(&mqtt, ADAFRUIT\_IO\_USERNAME "/feeds/temperature");

Adafruit\_MQTT\_Publish humidity = Adafruit\_MQTT\_Publish(&mqtt, ADAFRUIT\_IO\_USERNAME "/feeds/humidity");

void setup() {

  Serial.begin(115200);

  // Connect to Wi-Fi.

  WiFi.begin(WIFI\_SSID, WIFI\_PASS);

  while (WiFi.status() != WL\_CONNECTED) {

    delay(500);

    Serial.println("Connecting to WiFi...");

  }

  Serial.println("Connected to WiFi");

  // Connect to Adafruit IO.

  mqtt.connect();

  Serial.println("Connected to Adafruit IO");

}

void loop() {

  // Read temperature and humidity data from the DHT sensor.

  float temperatureValue = dht.readTemperature();

  float humidityValue = dht.readHumidity();

  // Publish data to Adafruit IO.

  if (!isnan(temperatureValue)) {

    temperature.publish(temperatureValue);

    Serial.print("Temperature: ");

    Serial.println(temperatureValue);

  } else {

    Serial.println("Failed to read temperature");

  }

  if (!isnan(humidityValue)) {

    humidity.publish(humidityValue);

    Serial.print("Humidity: ");

    Serial.println(humidityValue);

  } else {

    Serial.println("Failed to read humidity");

  }

  delay(60000); // Delay for 60 seconds (adjust as needed).

}

**MICROPROCESSOR PROGRAM**

**cpp**

**#include <Wire.h> // Include the Wire library for I2C communication**

**// Define the addresses of your sensors**

**const int humiditySensorAddress = 0x3F;**

**const int temperatureSensorAddress = 0x4A;**

**const int soundSensorPin = A0;**

**const int waterDetectorPin = 2;**

**const int airFlowSensorPin = A1;**

**void setup() {**

**Serial.begin(9600); // Initialize serial communication**

**Wire.begin(); // Initialize I2C communication**

**pinMode(waterDetectorPin, INPUT);**

**}**

**void loop() {**

**float humidity = readHumidity(humiditySensorAddress);**

**float temperature = readTemperature(temperatureSensorAddress);**

**int soundLevel = analogRead(soundSensorPin);**

**bool isWaterDetected = digitalRead(waterDetectorPin);**

**int airFlow = analogRead(airFlowSensorPin);**

**// Process and display the data**

**Serial.print("Temperature: ");**

**Serial.print(temperature);**

**Serial.println("°C");**

**Serial.print("Humidity: ");**

**Serial.print(humidity);**

**Serial.println("%");**

**Serial.print("Sound Level: ");**

**Serial.println(soundLevel);**

**Serial.print("Water Detected: ");**

**Serial.println(isWaterDetected ? "Yes" : "No");**

**Serial.print("Air Flow: ");**

**Serial.println(airFlow);**

**// Add code to transmit data to your desired destination or take actions based on sensor readings**

**delay(10000); // Delay for 10 seconds before taking the next reading**

**}**

**float readHumidity(int address) {**

**// Implement code to read humidity from the I2C sensor**

**// Return the humidity reading**

**}**

**float readTemperature(int address) {**

**// Implement code to read temperature from the I2C sensor**

**// Return the temperature reading**

**}**

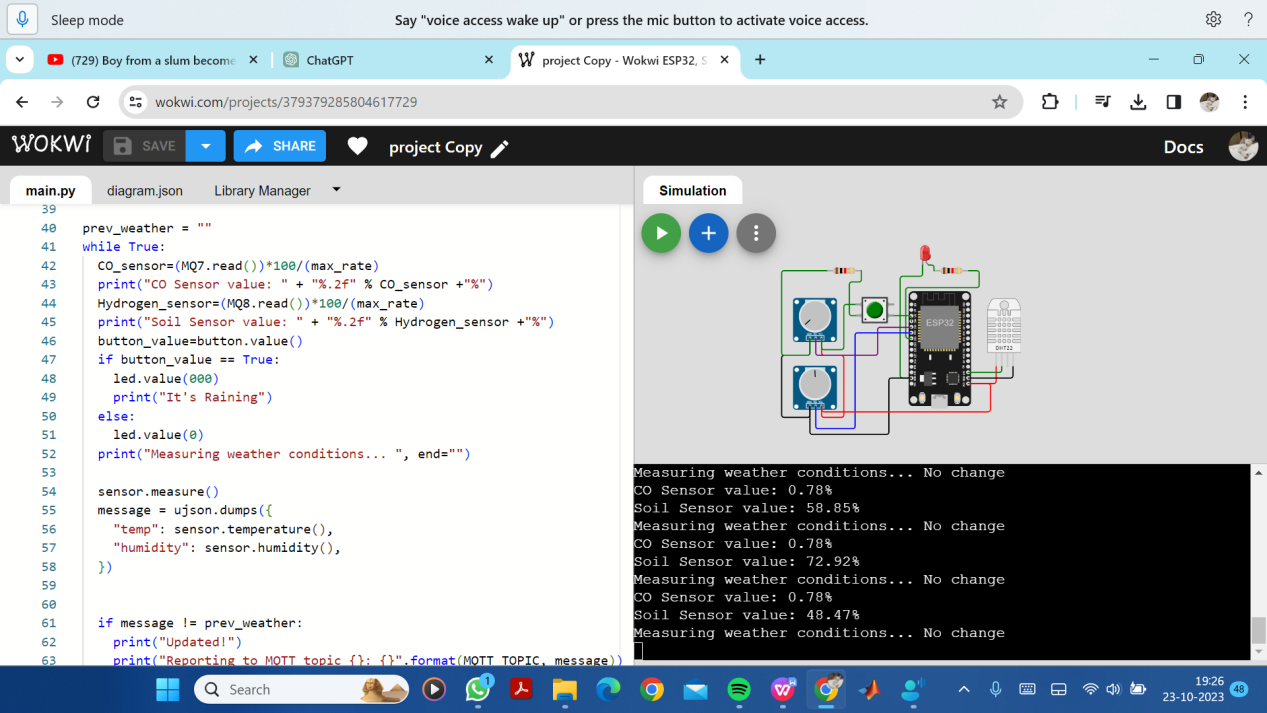
**// Implement similar functions for other sensors**

**void processSensorsData(float humidity, float temperature, int soundLevel, bool waterDetected, int airFlow) {**

**// Add your logic for data processing here**

**}**

**Result**:



**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 harnesses the power of interconnected sensors, devices, and data analytics to collect, manage, and analyze environmental data in real-time.