ENVIRONMENTAL MONITORIING

PHASE – 3

TITLE: DEPLOYMENT OF ENVIRONMENTAL SENSORS IN PUBLIC PARKS

INTRODUCTION:

In the quest for improving public park management and ensuring the well-being of park visitors, the deployment of environmental sensors for monitoring temperature, humidity, and air quality is essential. This project aims to create a sustainable and data-driven approach to maintaining public parks by accurately monitoring their environmental conditions.

SENSOR SELECTION AND PREPARATION:

- **Sensor Selection:** For this project, we've carefully chosen sensors that are robust and weather-resistant, capable of withstanding outdoor conditions. The DHT22 sensor is used for temperature and humidity, while a specialized air quality sensor is employed for monitoring air quality parameters.

- **Technical Specifications:** Our chosen sensors have the necessary technical specifications to provide accurate and reliable data. These include a wide measurement range, high accuracy, and the ability to work in outdoor environments.
- **Calibration:** Prior to deployment, all sensors are calibrated to ensure their accuracy. Calibration ensures that the sensors provide data that can be relied upon for decision-making.

SENSOR PLACEMENT IN PUBLIC PARKS:

- **Strategic Locations:** Sensors are strategically placed throughout the park. Temperature and humidity sensors are positioned in areas frequented by visitors, such as picnic areas and play zones. Air quality sensors are strategically positioned near parking lots and areas with potential pollution sources.

- **Factors Influencing Placement:** Sensor placement takes into account factors like vegetation, as shade and wind patterns can affect temperature and humidity readings. For air quality monitoring, proximity to pollution sources is a key consideration.

POWER AND CONNECTIVITY IN OUTDOOR SETTINGS:

- **Power Source:** Our sensors are powered by a combination of solar panels and battery backups. Solar panels ensure continuous power supply, even in the outdoors.

- **Connectivity:** For data transmission, we use cellular connectivity, which offers the reliability and coverage needed for remote locations within the park.

DATA VISUALIZATION AND ANALYSIS FOR PARK MANAGEMENT:

- **Data Visualization:** Sensor data is visualized through an intuitive and user-friendly interface. Park management teams can access temperature, humidity, and air quality data in real-time.
- **Data Analysis:** Data analysis tools provide insights into environmental trends and patterns. For instance, temperature and humidity data can help in planning for visitor comfort, while air quality data assists in addressing pollution issues.

MAINTENANCE AND CALIBRATION IN PUBLIC PARKS:

- **Maintenance Schedule:** Sensors are maintained on a regular schedule. Maintenance tasks include cleaning, battery checks, and equipment inspections.

- **Calibration:** Ongoing calibration ensures the sensors continue to provide accurate data. This is especially crucial in outdoor settings, where environmental conditions can change rapidly.

REPORTING AND ALERTS FOR PARK AUTHORITIES:

- **Data Reporting**: Environmental data is reported to park authorities and maintenance teams. This includes daily summaries, weekly reports, and ad-hoc reports in case of unusual environmental conditions.
- **Alerts:** The system is equipped with alert mechanisms that notify authorities of critical environmental conditions. These alerts are essential for addressing safety concerns and operational issues in the park.

PUBLIC AWARENESS AND ENGAGEMENT:

- **Public Interface:** Environmental data, such as temperature, humidity, and air quality, is made accessible to the public through informational displays within the park.
- **Educational Initiatives:** The project encourages public awareness and engagement by providing educational material on the importance of environmental monitoring. This includes information on how visitors can contribute to maintaining a healthy park environment.

PYTHON SCRIPT (For Temperature and Humidity only)

```
import time
import urequests
from machine import Pin
import dht
# WiFi credentials
ssid = "Wokwi-GUEST"
password = ""
# Beeceptor endpoint
server_url = "https://smartenviron.free.beeceptor.com/smartenviron/"
# DHT sensor configuration
dht_pin = 4  # Define the GPIO pin to which the DHT22 is connected
dht_sensor = dht.DHT22(Pin(dht_pin))
def connect_wifi():
    import network
    sta_if = network.WLAN(network.STA_IF)
    if not sta_if.isconnected():
        print("Connecting to WiFi...")
        sta_if.active(True)
        sta_if.connect(ssid, password)
        while not sta_if.isconnected():
            pass
    print("Connected to WiFi")
def send_data(temperature, humidity):
    data = "temperature={:.2f}&humidity={:.2f}".format(temperature,
humidity)
    headers = {"Content-Type": "application/x-www-form-urlencoded"}
```

```
    response = urequests.post(server_url, data=data, headers=headers)
    return response
def main():
    connect_wifi()
    while True:
        try:
            dht_sensor.measure()
            temperature = dht_sensor.temperature()
            humidity = dht_sensor.humidity()
            if not math.isnan(temperature) and not math.isnan(humidity):
                response = send_data(temperature, humidity)
                if response.status_code == 200:
                    print("Data sent to Beeceptor.")
                else:
                    print("Error in HTTP request. HTTP Response code:", response.status_code)
            else:
                print("Failed to read from DHT sensor!")
            time.sleep(60)  # Send data every 1 minute (adjust as needed)
        except Exception as e:
            print("An error occurred:", e)
if __name__ == "__main__":
    main()
```

PYTHON SCRIPT (For Temperature, Humidity and to determine the Air Monitoring System of the Atmosphere)

```
import time
from machine import Pin, ADC
import dht
import ubinascii
from umqtt.simple import MQTTClient
# MQTT broker configuration
MQTT_BROKER = "your_broker_address"
MQTT_PORT = 1883
MQTT_USER = "your_mqtt_user"
MQTT_PASSWORD = "your_mqtt_password"
MQTT_TOPIC = "air_monitoring"
CLIENT_ID = ubinascii.hexlify(ubinascii.unique_id()).decode('utf-8')
# Sensor configuration
DHT_PIN = Pin(2, Pin.IN)
DHT_SENSOR = dht.DHT22(DHT_PIN)
GAS_SENSOR_PIN = Pin(36, Pin.IN)  # Replace with the appropriate pin for your gas sensor
adc = ADC(Pin(34))  # Analog pin for gas sensor reading
# LEDs and Buzzer
GREEN_LED = Pin(25, Pin.OUT)
RED_LED = Pin(26, Pin.OUT)
BUZZER = Pin(27, Pin.OUT)
# Set initial states
GREEN_LED.off()
RED_LED.off()
BUZZER.off()
```

```
# MQTT client setup
client = MQTTClient(CLIENT_ID, MQTT_BROKER, user=MQTT_USER, password=MQTT_PASSWORD)
client.connect()
client.publish(MQTT_TOPIC, "Air Monitoring System is online.")
def send_mqtt(message):
    client.publish(MQTT_TOPIC, message)
def monitor_air_quality():
    while True:
        DHT_SENSOR.measure()
        temperature = DHT_SENSOR.temperature()
        humidity = DHT_SENSOR.humidity()
        # Gas sensor reading
        gas_sensor_value = adc.read()
        if gas_sensor_value > 40:
            RED_LED.on()
            GREEN_LED.off()
            BUZZER.on()
            send_mqtt("Harmful Gas Detected")
        else:
            RED_LED.off()
            GREEN_LED.on()
            BUZZER.off()
            send_mqtt("No Harmful Gas Detected")
        print("Temperature: {:.2f} C, Humidity: {:.2f}%, Gas Sensor Reading: {}".format(temperature, humidity, gas_sensor_value))
        time.sleep(60)  # Send data every 60 seconds
monitor_air_quality()
```