**SMART WATER FOUNTAIN**

**Phase 4: Development Part 2**

In this part you will continue building your project. Continue building the project by developing the water fountain status platform. Use web development technologies (e.g., HTML, CSS, JavaScript) to create a platform that displays real-time water fountain status. Design the platform to receive and display real-time water fountain data, including water flow rate and malfunction alerts.

**WOWKI SIMULATION PYTHON SCRIPT FOR SMART WATER FOUNTAIN**

import json

import machine

import network

from hcsr04 import HCSR04

from machine import Pin

import ure as re

import usocket as socket

import time

from dht import DHT22

# Set up Wi-Fi

SSID = "YourSSID"

PASSWORD = "YourPassword"

wlan = network.WLAN(network.STA\_IF)

wlan.active(True)

wlan.connect(SSID, PASSWORD)

# Define HC-SR04 pins

trig\_pin = Pin(23, Pin.OUT)

echo\_pin = Pin(22, Pin.IN)

# Define Relay Module and LED pins

relay\_pin = Pin(18, Pin.OUT)

led\_pins = [Pin(25, Pin.OUT), Pin(26, Pin.OUT), Pin(27, Pin.OUT)]

# Initialize the HC-SR04 sensor

sensor = HCSR04(trigger\_pin=trig\_pin, echo\_pin=echo\_pin)

# Initialize the DHT22 sensor

dht\_pin = Pin(4, Pin.IN)

dht\_sensor = DHT22(dht\_pin)

# Web server

def handle\_request(client):

request = client.recv(1024).decode('utf-8')

if 'GET /on' in request:

relay\_pin.on()

elif 'GET /off' in request:

relay\_pin.off()

distance = sensor.distance\_cm()

water\_level = "High" if distance < 10 else "Low"

dht\_sensor.measure()

temperature = dht\_sensor.temperature()

humidity = dht\_sensor.humidity()

response="HTTP/1.1200OK\r\nContent-Type: text/html\r\n\r\n"

response+=f"<html><body><h1>Water Level and Temperature/Humidity Monitoring</h1>"

response += f"<p>Distance: {distance} cm</p>"

response += f"<p>Water Level: {water\_level}</p>"

response+=f"<p>Temperature:{temperature}°C</p>"

response+=f"<p>Humidity:{humidity}%</p>" response+="<p><ahref='/on'>TurnPumpOn</a></p>"

response+="<p><ahref='/off'>TurnPumpOff</a></p>"

response += "</body></html>"

client.send(response)

client.close()

def run\_server():

s=socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

s.bind(('', 80))

s.listen(5)

while True:

client, addr = s.accept()

handle\_request(client)

# Main loop

while True:

distance = sensor.distance\_cm()

dht\_sensor.measure()

temperature = dht\_sensor.temperature()

humidity = dht\_sensor.humidity()

print("Distance:", distance, "cm")

print("Temperature:", temperature, "°C")

print("Humidity:", humidity, "%")

# Control the water pump based on distance

if distance < 10:

relay\_pin.on()

else:

relay\_pin.off()

# Indicate water level using LEDs

if distance < 10:

for i in range(3):

led\_pins[i].on()

else:

for i in range(3):

led\_pins[i].off()

# Run the web server

run\_server()

# Delay for a while to avoid excessive measurements

time.sleep(2)

if 'GET /data' in request:

# Measure the distance, water level, temperature, and humidity

distance = sensor.distance\_cm()

water\_level = "High" if distance < 10 else "Low"

dht\_sensor.measure()

temperature = dht\_sensor.temperature()

humidity = dht\_sensor.humidity()

# Create a dictionary to hold the data

data = {

"distance": distance,

"water\_level": water\_level,

"temperature": temperature,

"humidity": humidity

}

# Return the data as JSON

response = "HTTP/1.1 200 OK\r\nContent-Type: application/json\r\n\r\n"

response += json.dumps(data)

client.send(response)

client.close()

PLATFORM THE DISPLAY THE REALTIME RESULT

HTML code:We have placeholders for displaying the distance, water level, temperature, and humidity. JavaScript to fetch data from your Python program at the /data endpoint and update the placeholders.

<!DOCTYPE html>

<html>

<head>

<title>Water Level and Temperature/Humidity Monitoring</title>

</head>

<body>

<h1>SMART WATER FOUNTAIN USING IOT</h1>

<h2>Water Level and Temperature/Humidity Monitoring</h2>

<p>Distance: <span id="distance">Waiting for data...</span> cm</p>

<p>Water Level: <span id="waterLevel">Waiting for data...</span> </p>

<p>Temperature: <span id="temperature">Waiting for data...</span>° C</p>

<p>Humidity: <span id="humidity">Waiting for data...</span>%</p>

<p><a href="/on">Turn Pump On</a></p>

<p><a href="/off">Turn Pump Off</a></p>

<script>

**// JavaScript code to update the data on the HTML page**

function updateData() {

fetch('/data') // You should create an endpoint to fetch the data from your Python code

.then(response => response.json())

.then(data => {

document.getElementById('distance').innerText = data.distance + " cm";

document.getElementById('waterLevel').innerText=data.water\_level;

document.getElementById('temperature').innerText = data.temperature + "°C";

document.getElementById('humidity').innerText = data.humidity + "%";

});

} // Call the updateData function initially

updateData();

// Periodically update the data (e.g., every 5 seconds)

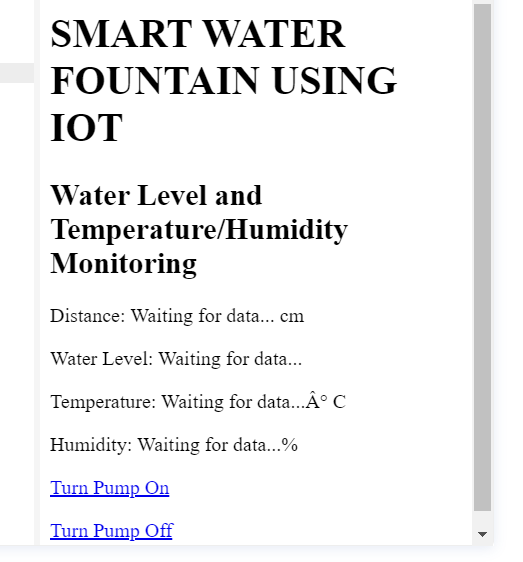
setInterval(updateData, 5000);

</script>

</body>

</html>

OUTPUT OF THE WEB PLATFORM



THANK YOU