Project – 8

TITLE – Smart water fountain

-submitted by :A.NITHIKA

PHASE 3:

Deploy IOT sensors (E.g., Temperature sensors, Pressure sensors) in public water fountains to monitor water flow and detect malfunctions. Develop a python script on the IOT sensors to send real-time water fountain status data to the platform.

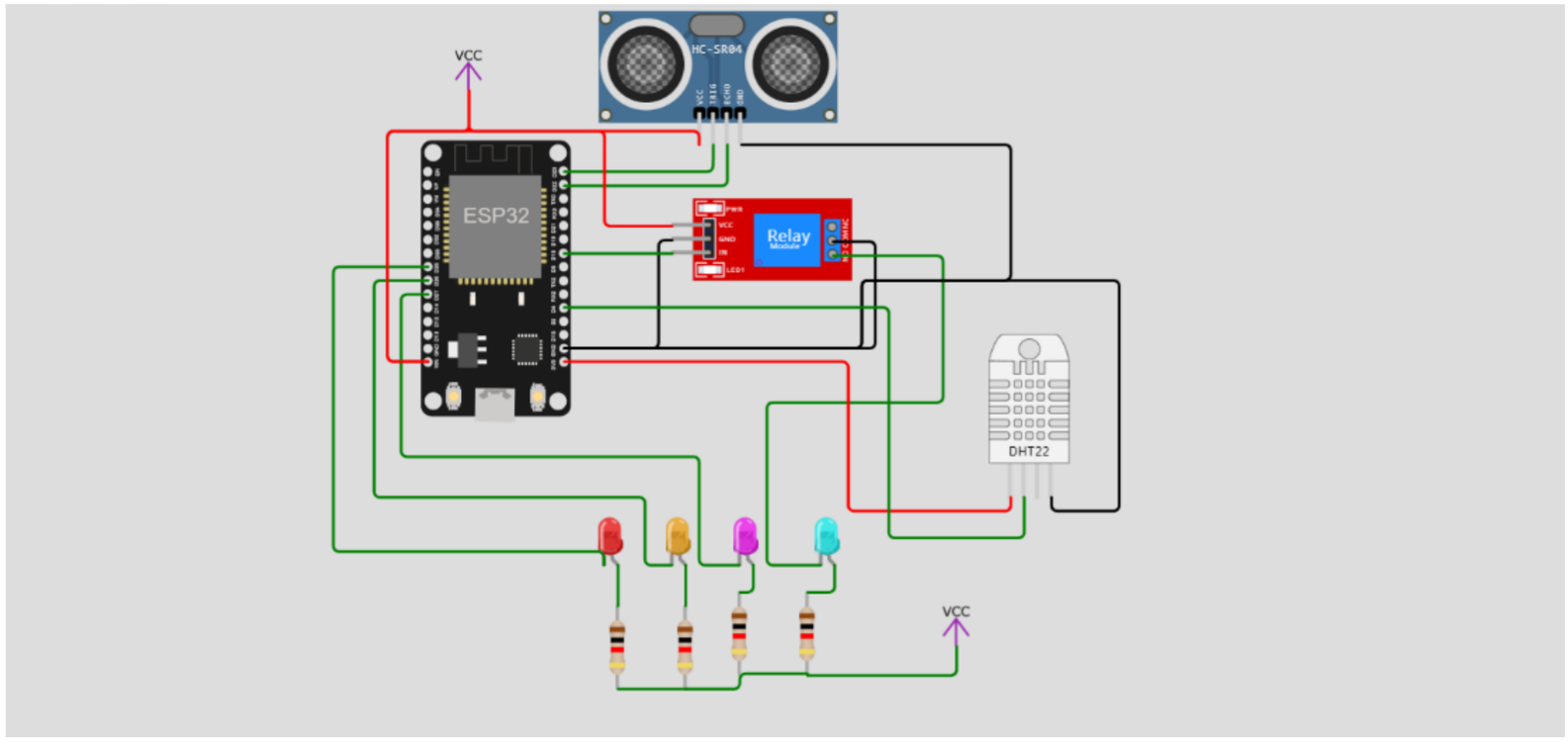
SMART WATER FOUNTAIN:

Fountain wirelessly communicate with base station. Base stations collect and transmit usage, filter, and system information to the cloud via Ethernet. Wireless communications use a low-power unlicenced band for improved security and power savings.

HARDWARE SETUP OF SMART WATER FOUNTAIN:

- ESP32 development board
- HC-SR04 ultra sonic distance sensor
- DHT22
- ATmega328p microcontroller IC
- Soil moisture sensor
- Real-time clock (DS3231)
- Relay module
- Arduino mini
- LED

SIMULATION DIAGRAM



ABOUT CIRCUIT LAYOUT:

- The Arduino board connected to the relay through a digital pin, Relay is connected to the pump.

- Relay Module:
  - GND to GND
  - VCC to VCC
  -Vin to D18
- Ultrasonic Sensor (HC-SR04):
  - VCC to VCC
  - GND to GND
  - Trig to Node MCU GPIO (D23)
  - Echo to Node MCU GPIO (D22)
- LED
- Meters and sensors (is used to transmit the output to real-time)

PYTHON SCRIPT:

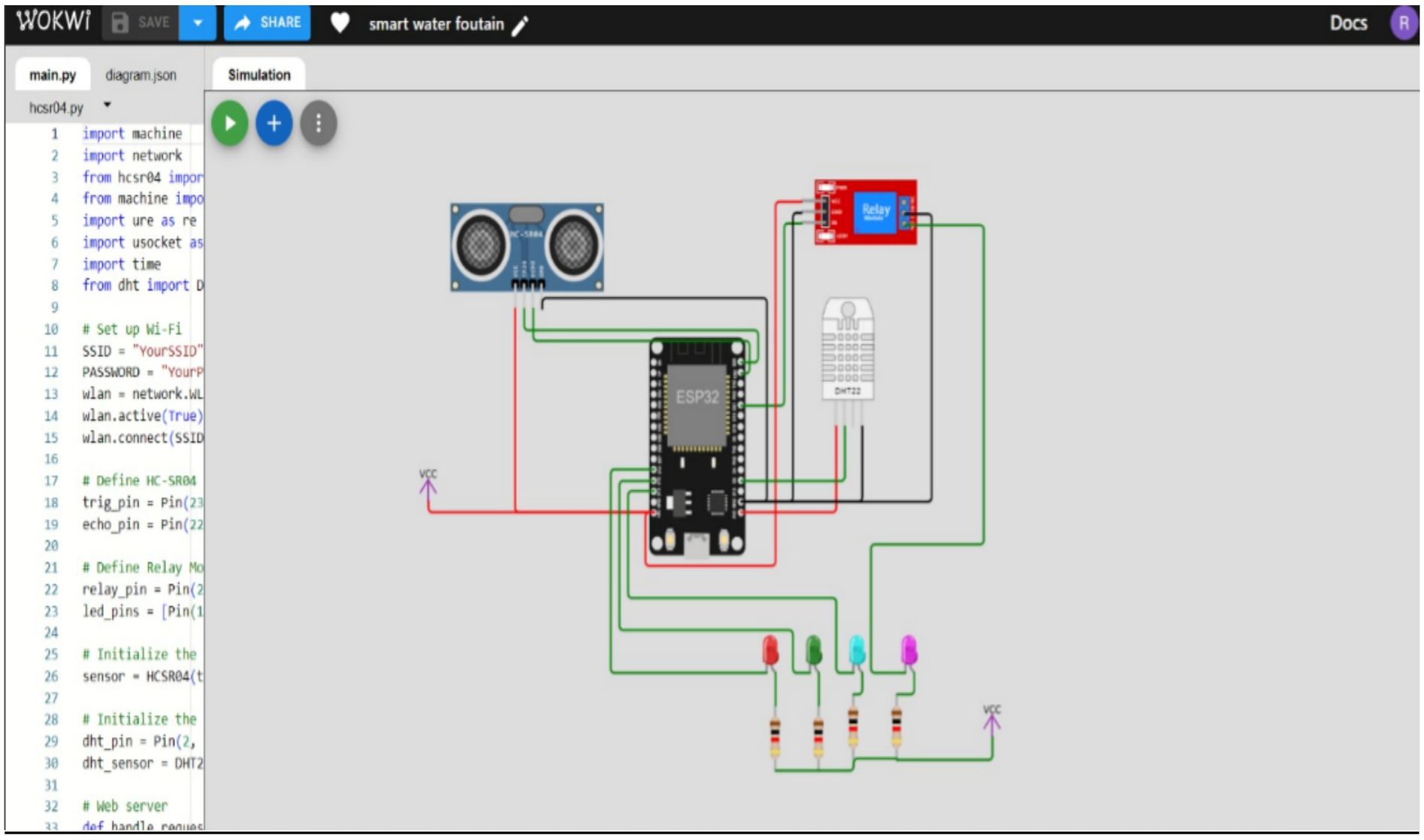
```
import machine
import network
from hcsr04 import HCSR04
from machine import Pin
import ure as re
import usocket as socket
import time
```

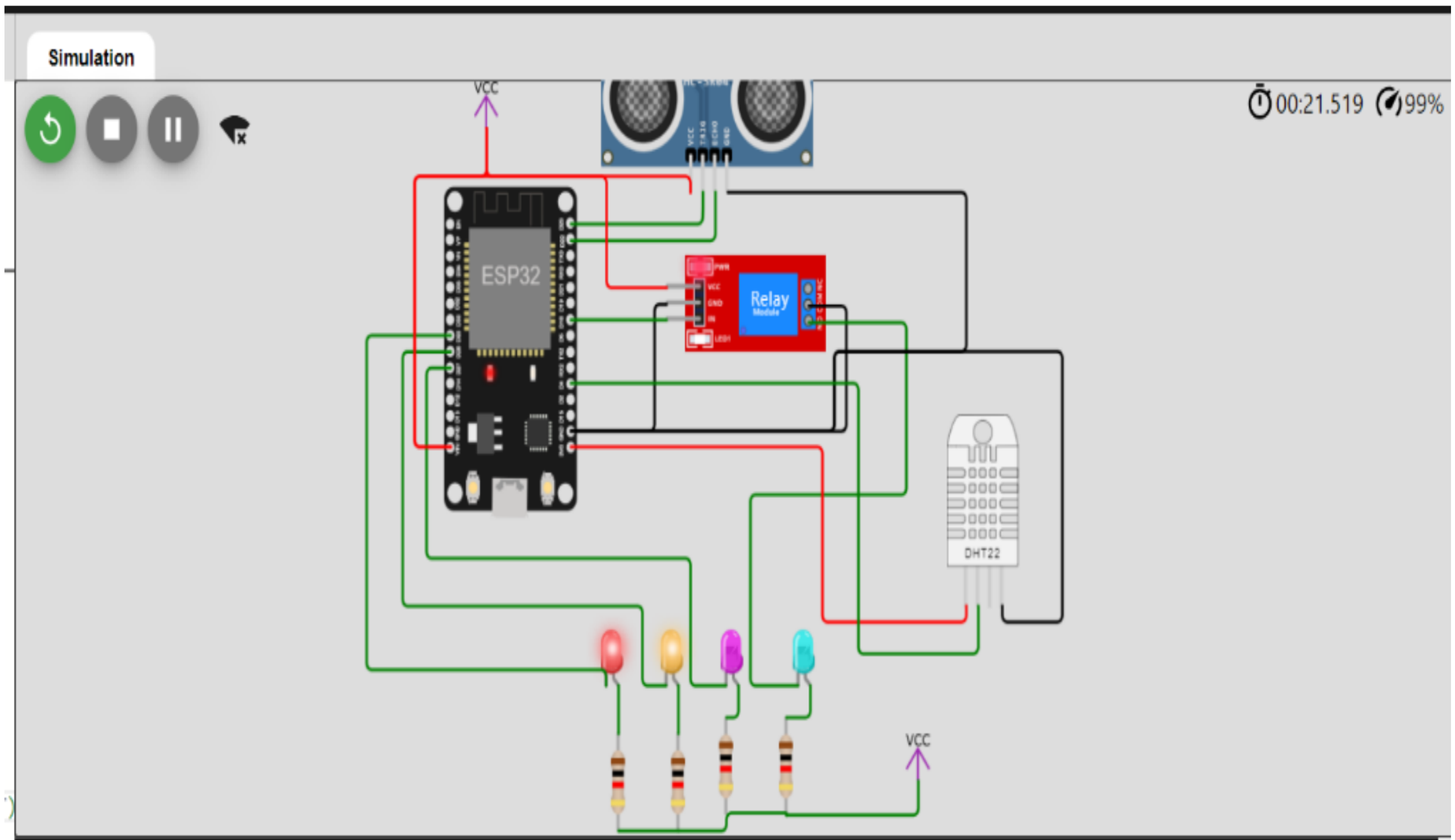
```
from dht import DHT22
# Set up Wi-Fi
SSID = "YourSSID"
PASSWORD = "YourPassword"
wlan = network.WLAN(network.STA_IF)
wlan.active(True)
wlan.connect(SSID, PASSWORD)
# Define HC-SR04 pins
trig_pin = Pin(23, Pin.OUT)
echo_pin = Pin(22, Pin.IN)
# Define Relay Module and LED pins
relay_pin = Pin(18, Pin.OUT)
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><a href='/on'>TurnPumpOn</a></p>"
    response+="<p><a href='/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)
```

OUTPUT OF SIMULATION:






```
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
Distance: 190.0 cm
Temperature: -15.6 °C
Humidity: 21.5 %
```

THANK YOU