**SMART WATER FOUNTAIN**

**DESCRIPTION :**

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

**PROGRAM :**

import machine

import time

# Pin assignments for the ultrasonic sensor

TRIGGER\_PIN = 23  # GPIO23 for trigger

ECHO\_PIN = 22     # GPIO22 for echo

# Pin assignment for the LED

LEAK\_LED\_PIN = 19  # GPIO19 for the LED

# Set the pin modes

trigger = machine.Pin(TRIGGER\_PIN, machine.Pin.OUT)

echo = machine.Pin(ECHO\_PIN, machine.Pin.IN)

leak\_led = machine.Pin(LEAK\_LED\_PIN, machine.Pin.OUT)

# Function to measure distance using the ultrasonic sensor

def measure\_distance():

    # Generate a short trigger pulse

    trigger.value(0)

    time.sleep\_us(5)

    trigger.value(1)

    time.sleep\_us(10)

    trigger.value(0)

    # Measure the echo pulse duration to calculate distance

    pulse\_start = pulse\_end = 0

    while echo.value() == 0:

        pulse\_start = time.ticks\_us()

    while echo.value() == 1:

        pulse\_end = time.ticks\_us()

    pulse\_duration = pulse\_end - pulse\_start

    # Calculate distance in centimeters (assuming the speed of sound is 343 m/s)

    distance = (pulse\_duration \* 0.0343) / 2  # Divide by 2 for one-way travel

    return distance

# Function to check for a water leak

def check\_for\_leak():

    # Measure the distance from the ultrasonic sensor

    distance = measure\_distance()

    # Set the threshold distance for detecting a leak (adjust as needed)

    threshold\_distance = 10  # Adjust this value based on your tank setup

    if distance < threshold\_distance:

        # If the distance is less than the threshold, a leak is detected

        return True

    else:

        return False

# Main loop

while True:

    if check\_for\_leak():

        # Blink the LED to indicate a leak

        leak\_led.value(1)  # LED ON

        time.sleep(0.5)

        leak\_led.value(0)  # LED OFF

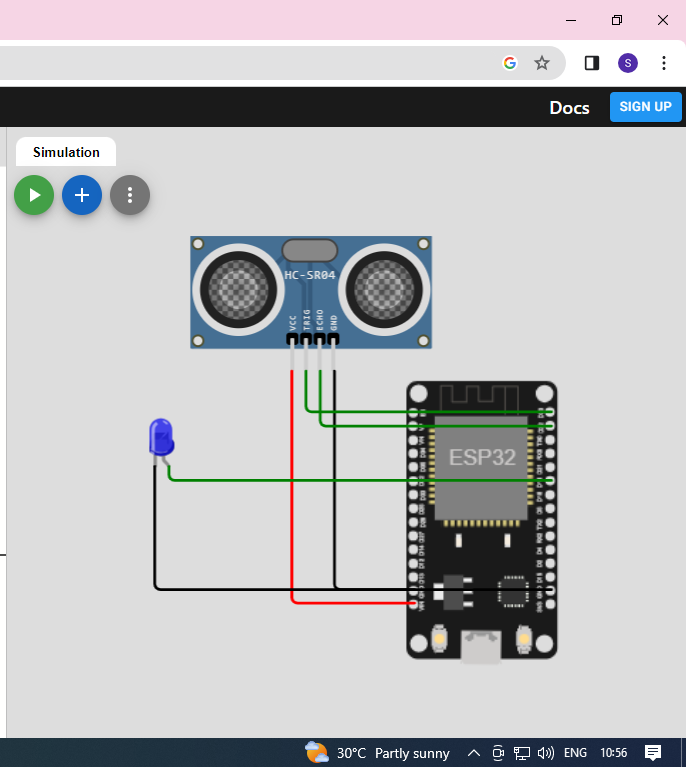
        time.sleep(0.5)

    else:

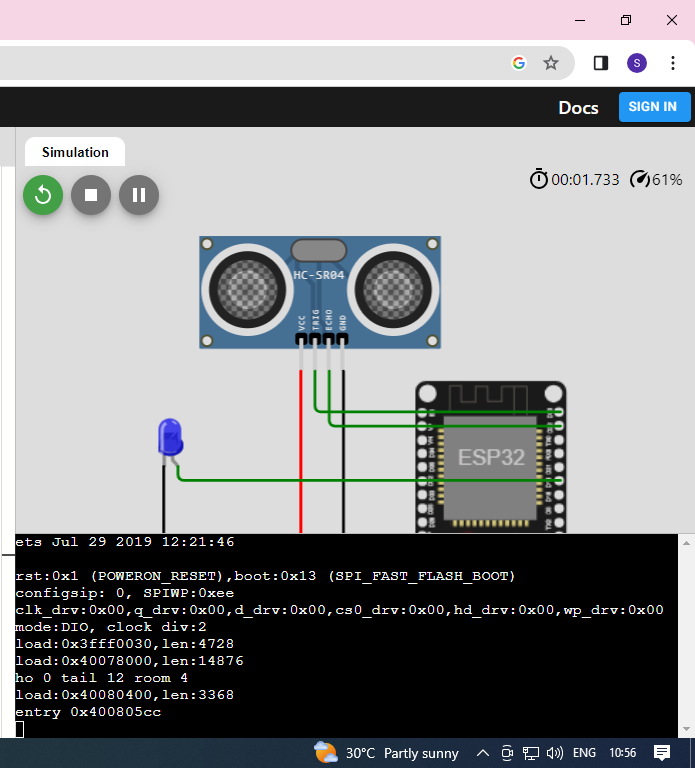
        leak\_led.value(0)  # LED OFF

    time.sleep(1)  # Delay between measurements

**SENSOR :**

****

**OUTPUT :**

****