**PHASE 4**

**DEVELOPING OF OUR PROJECT (PART2)**

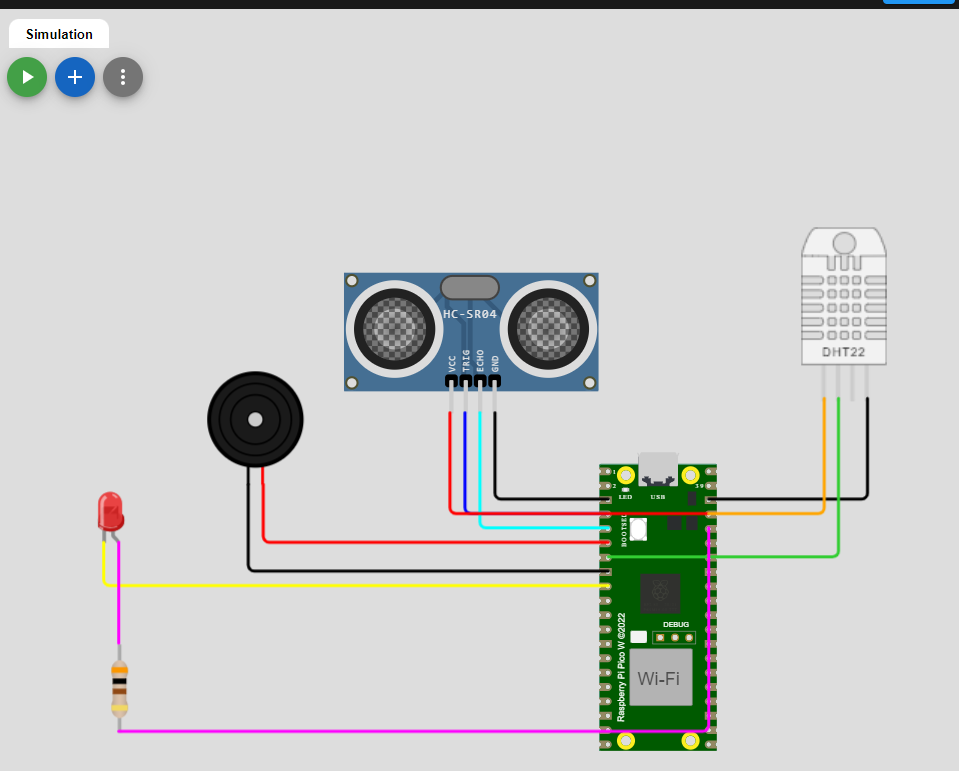
***OVERVIEW:***

In previous phase we start developing our project using ultrasonic sensor, Buzzer and the python script was developed .In this phase we further developing our project by adding Pico wifi,DHT22,LED,Resistor.

***RASPBERRY PI PICO:***

Raspberry Pi Pico W is a new Raspberry Pi product that adds Wi-Fi capability to the Raspberry Pi Pico, allowing you to connect the device to a Wi-Fi network. In this guide, you will learn how to use a Raspberry Pi Pico W, how to connect it to a Wi-Fi network, and then how to turn it into a web server to control digital outputs from a browser, and to receive sensor data.

***SIMULATION OF OUR PROJECT:***

******

[***https://wokwi.com/projects/378661131954863105***](wokwi)

***CODING:***

import time

import dht

import urequests

# Define GPIO pins

TRIG\_PIN = 2

ECHO\_PIN = 3

BUZZER\_PIN = 4

DHT\_PIN = 5

LED\_PIN = 6

THING\_SPEAK\_API\_KEY = "SS717YEOH72JBGYT"

THING\_SPEAK\_CHANNEL\_ID = " 2316433"

def distance\_measurement():

    # Trigger ultrasonic sensor

    machine.Pin(TRIG\_PIN, machine.Pin.OUT).on()

    time.sleep\_us(10)

    machine.Pin(TRIG\_PIN, machine.Pin.OUT).off()

    # Wait for echo to be HIGH (start time)

    while not machine.Pin(ECHO\_PIN, machine.Pin.IN).value():

        pass

    pulse\_start = time.ticks\_us()

    # Wait for echo to be LOW (end time)

    while machine.Pin(ECHO\_PIN, machine.Pin.IN).value():

        pass

    pulse\_end = time.ticks\_us()

    # Calculate distance

    pulse\_duration = time.ticks\_diff(pulse\_end, pulse\_start)

    distance = pulse\_duration / 58  # Speed of sound (343 m/s) divided by 2

    return distance

def read\_dht\_sensor():

    d = dht.DHT22(machine.Pin(DHT\_PIN, machine.Pin.IN))

    d.measure()

    return d.temperature(), d.humidity()

buzz\_start\_time = None  # To track when the buzzer started

while True:

    dist = distance\_measurement()

    temp, humidity = read\_dht\_sensor()

    status = "No Flooding Detected"

    if dist < 50:

        # Turn on the buzzer and LED

        machine.Pin(BUZZER\_PIN, machine.Pin.OUT).on()

        machine.Pin(LED\_PIN, machine.Pin.OUT).on()

        status = "Flooding Detected"

        buzz\_start\_time = time.ticks\_ms()

    elif buzz\_start\_time is not None and time.ticks\_diff(time.ticks\_ms(), buzz\_start\_time) >= 60000:  # 1 minute

        # Turn off the buzzer and LED after 1 minute

        machine.Pin(BUZZER\_PIN, machine.Pin.OUT).off()

        machine.Pin(LED\_PIN, machine.Pin.OUT).off()

    print(f"Distance: {dist:.2f} cm")

    print(f"Temperature: {temp:.2f}°C, Humidity: {humidity:.2f}%")

    print("Status:", status)

    # Send data to ThingSpeak

    try:

        data = {

            "api\_key": THING\_SPEAK\_API\_KEY,

            "field1": dist,

            "field2": temp,

            "field3": humidity,

        }

        response = urequests.post(f"https://api.thingspeak.com/update?api\_key=SS717YEOH72JBGYT&field1=0",

            json=data,

        )

        response.close()

    except Exception as e:

        print("Error sending data to ThingSpeak:", e)

    time.sleep(2)