

ASSIGNMENT 4

Name	Suganya.S
Team ID	PNT2022TMID34205
Project Name	IOT Based Smart Crop Protection System

Write code and connections in wokwi for ultrasonic sensor.

Whenever distance is less than 100 cms send "alert" to ibm cloud and display in device recent events.

Upload document with wokwi share link and images of ibm cloud

CODE:

```

#include <WiFi.h>
#include <PubSubClient.h> WiFiClient wifiClient;

#define ORG "nhpwjc"
#define DEVICE_TYPE "NodeMCU"
#define DEVICE_ID "USE YOUR ID"
#define TOKEN "USE YOUR TOKEN"
#define speed 0.034
char server[] = ORG
".messaging.internetofthings.ibmcloud.com"; char publishTopic[] = "iot-
2/evt/Data/fmt/json"; char topic[] = "iot-2/cmd/home/fmt/String"; char
authMethod[] = "use-tokenauth"; char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID; PubSubClient client(server,
1883, wifiClient); void publishData();

const int trigpin=5;
const int echopin=18;
String command;
String data=""; long
duration; float
dist;

void
setup()
{
  Serial.begin(115200); pinMode(trigpin,
  OUTPUT);

```

```

    pinMode(echopin, INPUT); wifiConnect();
    mqttConnect();
} void loop() { publishData(); delay(500);

    if (!client.loop()) { mqttConnect(); } }

void wifiConnect() {
    Serial.print("Connecting to "); Serial.print("Wifi");
    WiFi.begin("Wokwi-GUEST", "", 6); while (WiFi.status() !=
    WL_CONNECTED) { delay(500);
        Serial.print("."); }
    Serial.print("WiFi connected, IP address: "); Serial.println(WiFi.localIP());
}

void mqttConnect() { if
    (!client.connected()) {
        Serial.print("Reconnecting MQTT client to "); Serial.println(server); while (!client.connect(clientId,
        authMethod, token)) { Serial.print("."); delay(500);
        } initManagedDevice(); Serial.println(); } }

void initManagedDevice() { if
    (client.subscribe(topic)) {
        // Serial.println(client.subscribe(topic)); Serial.println("subscribe to cmd OK");
    } else {
        Serial.println("subscribe to cmd FAILED"); } } void publishData()
{ digitalWrite(trigpin, LOW); digitalWrite(trigpin, HIGH);

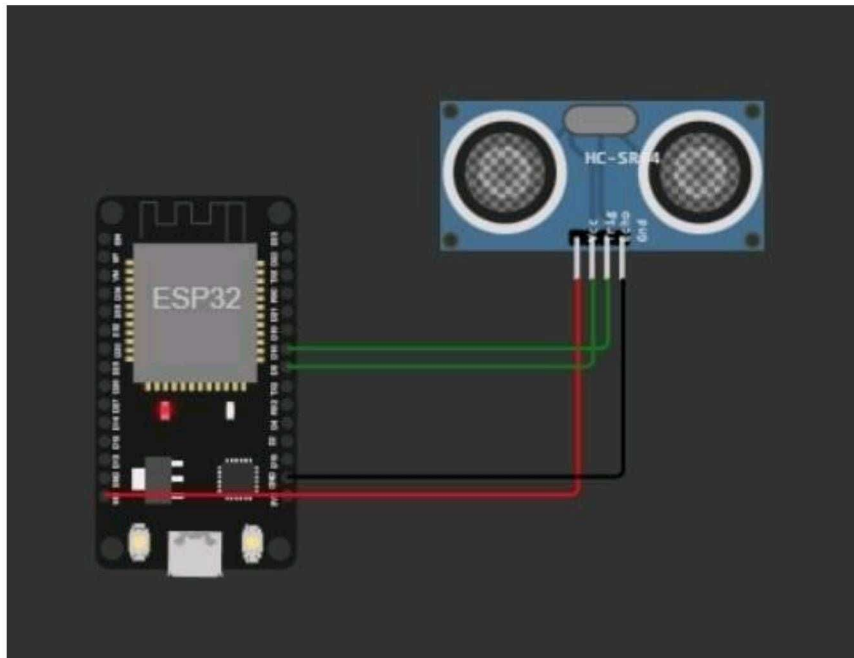
```

```

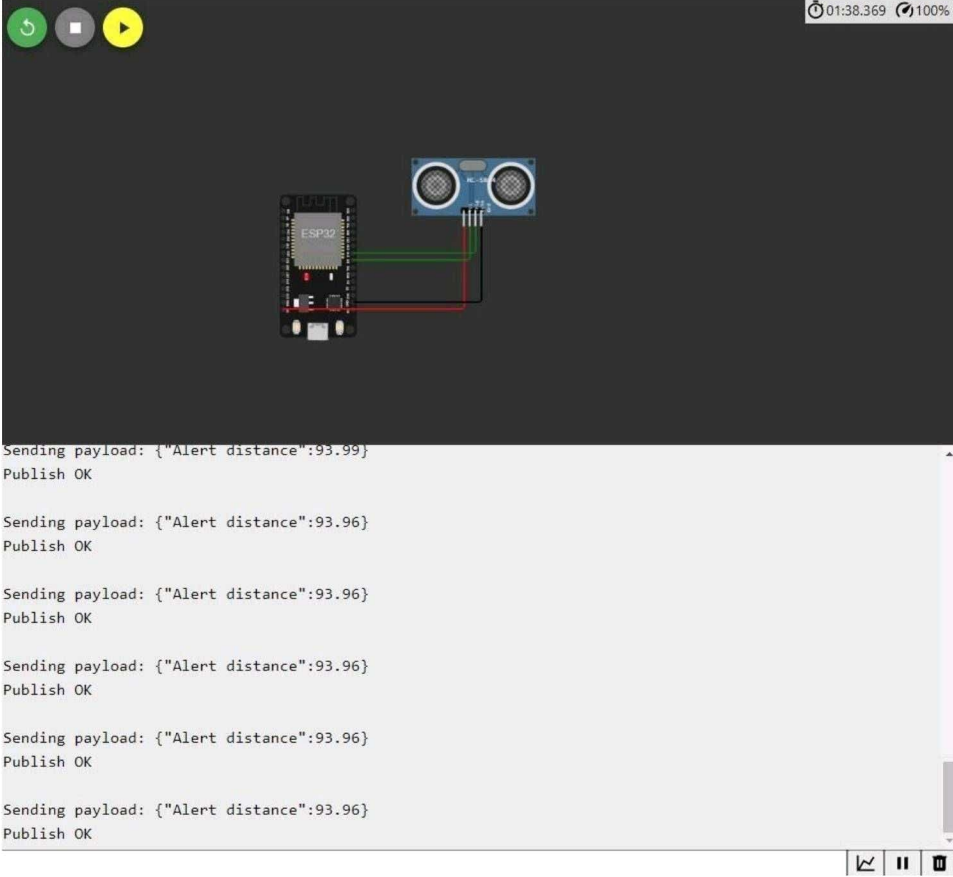
delayMicroseconds(10);
digitalWrite(trigpin,LOW);
duration=pulseIn(echopin,HIGH);
dist=duration*speed/2; if(dist<100){
  String payload = "{\nAlert distance\n:"; payload +=
  dist; payload += "\n"; Serial.print("\n");
  Serial.print("Sending payload: "); Serial.println(payload);
  if (client.publish(publishTopic, (char*) payload.c_str())) { Serial.println("Publish OK");
  } else {
    Serial.println("Publish FAILED"); }
}
}

```

CONNECTIONS:



OUTPUT:



The screenshot shows a microcontroller IDE interface. At the top, there are three circular buttons: a green one with a refresh icon, a grey one with a square icon, and a yellow one with a play icon. In the top right corner, a timer displays '01:38.369' and a battery icon indicates '100%'.

The central part of the interface features a circuit diagram on a dark background. It depicts an ESP32 microcontroller board connected to a blue HC-SR04 ultrasonic sensor. The sensor is connected to the ESP32 via four colored wires: red for VCC, black for GND, green for TRIG, and blue for ECHO.

Below the circuit diagram, the serial monitor window is open, displaying a series of messages. Each message consists of two lines: 'Sending payload: {"Alert distance":93.99}' followed by 'Publish OK'. This sequence is repeated five times. The serial monitor has a vertical scrollbar on the right side.

At the bottom right of the IDE window, there are three icons: a pencil icon for editing, a pause icon, and a trash can icon for clearing the serial monitor.

