# ASSIGNMENT 04 WOKWI STIMULATOR

|  |  |
| --- | --- |
| Date | 24 October 2022 |
| Student Name | GURUTHEEP |
| Student Roll Number | 713319EC035 |
| Team ID | PNT2022TMID17583 |
| Project Name | GAS LEAKAGE MONITORING AND ALERTING SYSTEM |

**QUESTION:**

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> #include <ArduinoJson.h>

WiFiClient wifiClient; #define ORG "wt19pm"

#define DEVICE\_TYPE "NodeMCU" #define DEVICE\_ID "12345"

#define TOKEN "12345678"

#define speed 0.034

char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; char publishTopic[] = "iot-2/evt/status1/fmt/json";

char topic[] = "iot-2/cmd/home/fmt/String"; char authMethod[] = "use-token-auth"; 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="";

String name="Alert";

String icon="";

long duration; int 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(".");

**Serial**.print("\*"); delay(1000);

}

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){ dist=100-dist; icon="no-trash";

}else{ dist=0;

icon="trash";

}

DynamicJsonDocument doc(1024); String payload; doc["Name"]=name; doc["Icon"]=icon; doc["FillPercent"]=dist;

serializeJson(doc, payload); delay(3000); **Serial**.print("\n");

**Serial**.print("Sending payload: ");

**Serial**.println(payload);

if (client.publish(publishTopic, (char\*) payload.c\_str())) {

**Serial**.println("Publish OK");

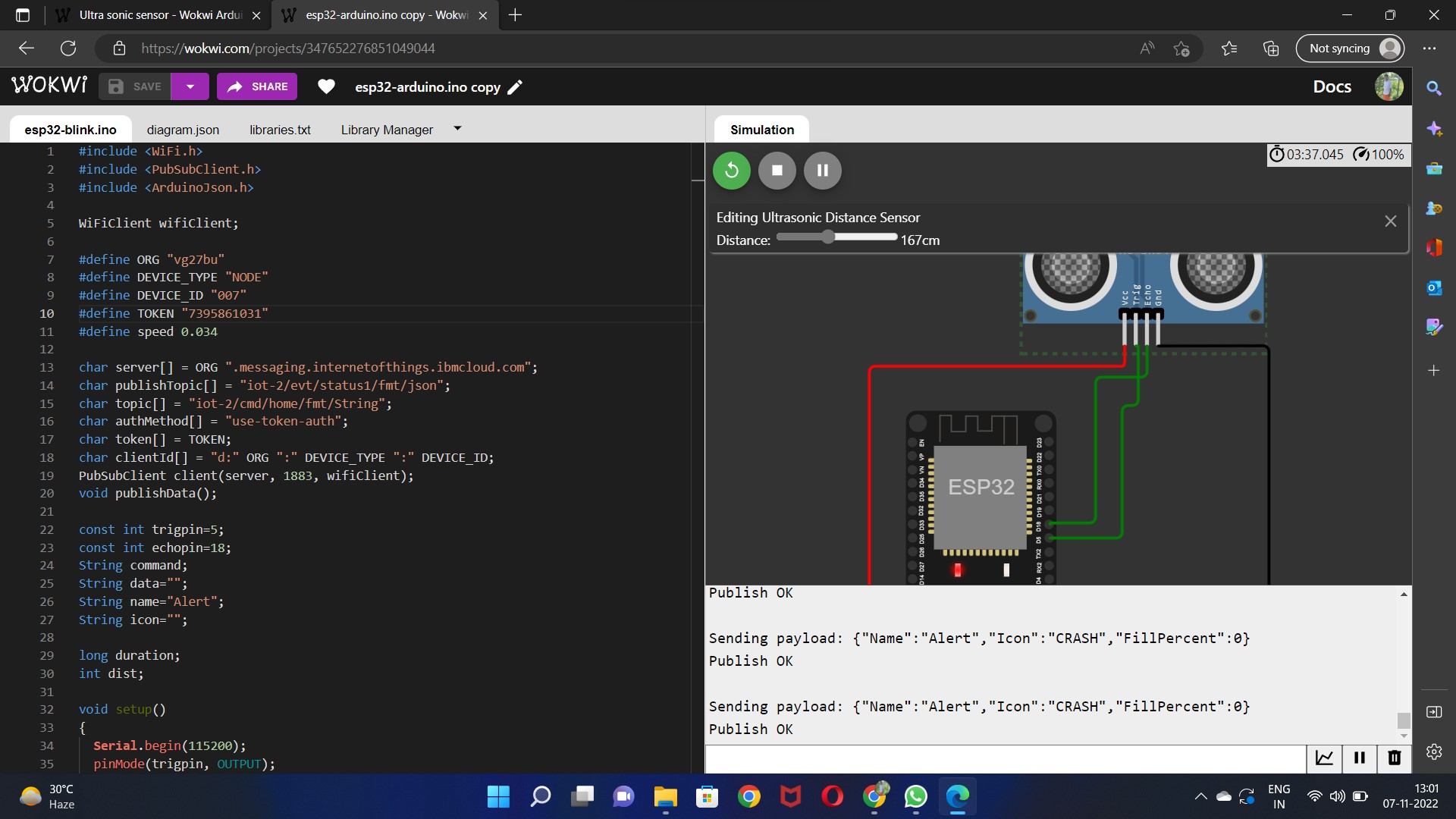
} else {

**Serial**.println("Publish FAILED");

}

}

# CONNECTOINS :



**WOKWI LINK:**

[esp32-arduino.ino copy - Wokwi Arduino and ESP32 Simulator](https://wokwi.com/projects/347652276851049044)

