

# ASSIGNMENT 4

NAME:M.CHANDRAKALA.

TEAM ID: PNT2022TMID30034

## QUESTION:

Write code and connections in wokwi for the ultrasonic sensor. Whenever the distance is less than 100 cm send an "alert" to the IBM cloud and display in the device recent events. Upload document with wokwi share link and images of IBM cloud

## PROGRAM:

```
#include <WiFi.h>

#include <WiFiClient.h>
#include <PubSubClient.h>
const int trigPin = 5;
const int echoPin = 18;
//define sound speed in cm/uS
#define SOUND_SPEED 0.034
#define CM_TO_INCH 0.393701
long duration;
float distanceCm;
float distanceInch;

void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
//-----credentials of IBM Accounts-----

#define ORG "qhQ3wv"//IBM ORGANITION ID
#define DEVICE_TYPE "Boobalan1"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "2001"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "(hyT84+@N8rnM(5U1M" //Token
String data3;
```

```

//----- Customise the above values -----
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of event
perform and format in which data to be send
char subscribetopic[] = "iot-2/cmd/test/fmt/String";// cmd REPRESENT command
type AND COMMAND IS TEST OF FORMAT STRING
char authMethod[] = "use-token-auth";// authentication method
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id

WiFiClient wifiClient; // creating the instance for wificlient
PubSubClient client(server, 1883, callback ,wifiClient);

void setup() {
    Serial.begin(115200); // Starts the serial communication
    pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
    pinMode(echoPin, INPUT); // Sets the echoPin as an Input
    Serial.println();
    wificonnect();
    mqttconnect();
}

void loop() {
    // Clears the trigPin
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    // Sets the trigPin on HIGH state for 10 micro seconds
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);

    // Reads the echoPin, returns the sound wave travel time in microseconds
    duration = pulseIn(echoPin, HIGH);

    // Calculate the distance
    distanceCm = duration * SOUND_SPEED/2;

    // Convert to inches
    distanceInch = distanceCm * CM_TO_INCH;

    // Prints the distance in the Serial Monitor

```

```

Serial.print("Distance (cm): ");
Serial.println(distanceCm);
Serial.print("Distance (inch): ");
Serial.println(distanceInch);

PublishData(distanceCm);
delay(1000);
if (!client.loop()) {
    mqttconnect();
}
}

void PublishData(float Cm) {
    mqttconnect();//function call for connecting to ibm
    /*
        creating the String in in form JSON to update the data to ibm cloud
    */
    String payload = "{\"Distance (cm)\":";
    payload += Cm;
    payload += "}";

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

    if (client.publish(publishTopic, (char*) payload.c_str())) {
        Serial.println("Publish ok");// if it sucessfully upload data on the cloud
        then it will print publish ok in Serial monitor or else it will print publish
        failed
    } else {
        Serial.println("Publish failed");
    }
}

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

        initManagedDevice();
    }
}

```

```

        Serial.println();
    }
}
void wificonnect() //function definition for wificonnect
{
    Serial.println();
    Serial.print("Connecting to ");

    WiFi.begin("Wokwi-GUEST", "", 6); //passing the wifi credentials to establish
the connection
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
    Serial.println("");
    Serial.println("WiFi connected");
    Serial.println("IP address: ");
    Serial.println(WiFi.localIP());
}

void initManagedDevice() {
    if (client.subscribe(subscribetopic)) {
        Serial.println((subscribetopic));
        Serial.println("subscribe to cmd OK");
    } else
    {
        Serial.println("subscribe to cmd FAILED");
    }
}

void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
{
    Serial.print("callback invoked for topic: ");
    Serial.println(subscribetopic);
    for (int i = 0; i < payloadLength; i++) {
        //Serial.print((char)payload[i]);
        data3 += (char)payload[i];
    }
}

```

**LINK** <https://wokwi.com/projects/347206323231785555>

# PROGRAM SCREEN SHOT:

The screenshot displays the Wokwi IDE interface for a new ESP32 project. The left pane shows the C++ code for the ESP32, which includes a setup function, a loop for subscribing to an MQTT topic, and a callback function that processes the received distance data. The right pane shows a simulation of the ESP32 board connected to an HC-SR04 ultrasonic sensor. The simulation window displays the sensor's output, including the distance in inches and centimeters, and the MQTT payload being sent.

```
119 Serial.print(".");
120 }
121 Serial.println("");
122 Serial.println("Wifi connected");
123 Serial.println("IP address: ");
124 Serial.println(WiFi.localIP());
125 }
126
127 void initManagedDevice() {
128   if (client.subscribe(subscribetopic)) {
129     Serial.println((subscribetopic));
130     Serial.println("subscribe to cmd OK");
131   } else
132   {
133     Serial.println("subscribe to cmd FAILED");
134   }
135 }
136 void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
137 {
138
139   Serial.print("callback invoked for topic: ");
140   Serial.println(subscribetopic);
141   for (int i = 0; i < payloadLength; i++) {
142     //Serial.print((char)payload[i]);
143     data3 += (char)payload[i];
144   }
145 }
146 }
```

Simulation window output:

```
Distance (inch): 48.02
Sending payload: {"Distance (cm)":121.97}
Publish ok
Distance (cm): 121.97
Distance (inch): 48.02
Sending payload: {"Distance (cm)":121.97}
Publish ok
```