

#### ASSIGNMENT-4

Assignment Date	15 October2022
Student Name	LAVANYA R
Student Roll Number	715519106024
Team ID	PNT2022TMID43384
Project Name	Smart Farmer - IoT Enabled Smart Farming Application
Maximum marks	2 marks

**Question:** Write code and connections in wokwi for the ultrasonic sensor. Whenever the distance is less than 100 cms 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<PubSubClient.h>
#include <ArduinoJson.h>

void callback(char* subscribetopic,byte* payload,unsigned int payloadLength);
#define ORG "o16sqk"
#define DEVICE_TYPE "NodeMCU"
#define DEVICE_ID "12345"
#define TOKEN "715519106004"
#define SOUND_SPEED 0.034
#define CM_TO_INCH 0.393701

const int trigPin = 5;
const int echoPin = 18;

long duration;
float distanceCm;
float distanceInch;
String data;

char server[]=ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/event_1/fmt/json";
char subscribeTopic[] = "iot-2/cmd/home/fmt/String";
char authMethod[] = "use-token-auth";
char token[] =TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;

WiFiClient 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
  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);

    delay(1000);

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

void PublishData(float distanceCm)
{

    mqttconnect ();
    String payload;

    if(distanceCm<100.0)
    {
        payload = "{\"Alert\":\"";
        payload += distanceCm;
        payload += "\"}";
    }
    else
    {
        payload = "{\"distanceCm\":\"";
        payload += distanceCm;

```

```
    payload += "}";  
}
```

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

```
if(client.publish(publishTopic , (char*) payload.c_str())){  
    Serial.println("Publish ok");}  
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()
```

```
{  
    Serial.println();  
    Serial.print("Connecting to");
```

```
    WiFi.begin("Wokwi-GUEST","",6);  
    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");
    }
}
}

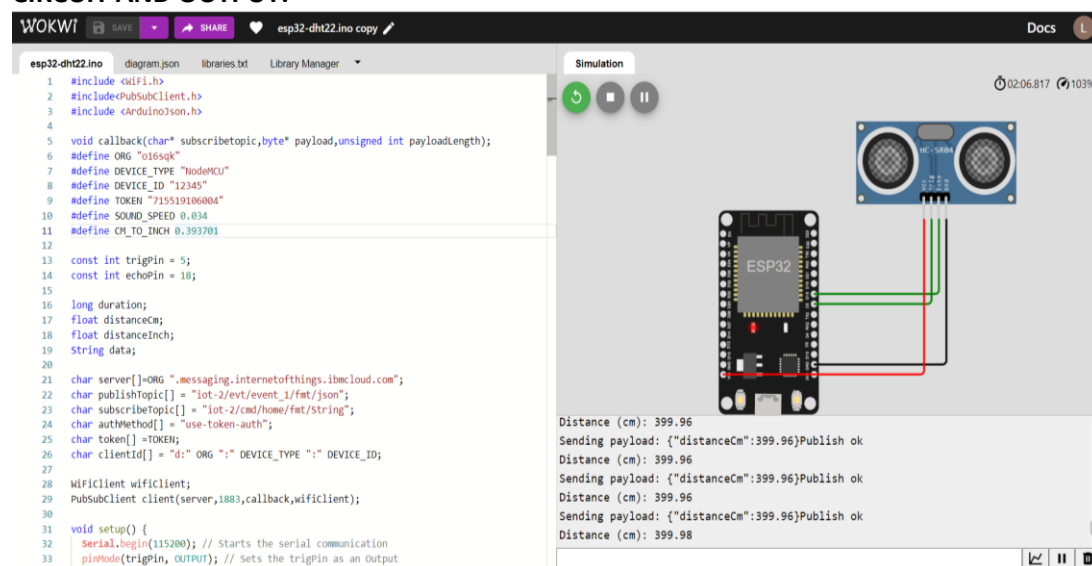
```

```

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++)
    {
        data +=(char)payload[i];
    }
    Serial.println("data: "+data);
    if(data>(char)100)
    {
        Serial.println("Alert!");
    }*/
}

```

## CIRCUIT AND OUTPUT:



The screenshot displays the Wokwi IDE interface. On the left, the code for the ESP32 is shown, including headers for WiFi, PubSubClient, and ArduinoJson. The code defines a callback function and sets up the serial communication. On the right, a simulation of the circuit is shown, featuring an ESP32 microcontroller and an HC-SR04 ultrasonic sensor. The sensor is connected to the ESP32 via four wires: VCC (red), GND (black), Trig (green), and Echo (blue). The simulation output at the bottom shows the distance measured by the sensor and the payload sent to the MQTT broker.

```

esp32-dht22.ino
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 #include <ArduinoJson.h>
4
5 void callback(char* subscribetopic,byte* payload,unsigned int payloadLength);
6 #define ORG "o16sqk"
7 #define DEVICE_TYPE "NodeMCU"
8 #define DEVICE_ID "12345"
9 #define TOKEN "715519106004"
10 #define SOUND_SPEED 0.034
11 #define CM_TO_INCH 0.393701
12
13 const int trigPin = 5;
14 const int echoPin = 18;
15
16 long duration;
17 float distanceCm;
18 float distanceInch;
19 String data;
20
21 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
22 char publishTopic[] = "iot-2/evt/event_1/fmt/json";
23 char subscribeTopic[] = "iot-2/cmd/home/fmt/String";
24 char authMethod[] = "use-token-auth";
25 char token[] = TOKEN;
26 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
27
28 WiFiClient wificlient;
29 PubSubClient client(server,1883,callback,wificlient);
30
31 void setup() {
32     Serial.begin(115200); // Starts the serial communication
33     pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output

```

Simulation output:

```

Distance (cm): 399.96
Sending payload: {"distanceCm":399.96}Publish ok
Distance (cm): 399.96
Sending payload: {"distanceCm":399.96}Publish ok
Distance (cm): 399.96
Sending payload: {"distanceCm":399.96}Publish ok
Distance (cm): 399.98

```

IBM Watson IoT Platform

715519106024@smartinternz.com  
ID: o16sqk

← Back

Device Drilldown - 12345

Connection Information

Recent Events

State

Device Information

Metadata

Diagnostics

Connection Logs

Device Actions

Recent Events

The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
event_1	{"distanceCm":399.96}	json	a few seconds ago
event_1	{"distanceCm":399.96}	json	a few seconds ago
event_1	{"distanceCm":399.96}	json	a few seconds ago
event_1	{"distanceCm":399.96}	json	a few seconds ago
event_1	{"distanceCm":399.98}	json	a few seconds ago

**LINK:** <https://wokwi.com/projects/347386870885253715>