

## Assignment -4

### SmartFarmer - IoT Enabled Smart Farming Application

Assignment Date	17 October 2022
Student Name	Madhava Narayana K
Student Roll Number	910619106028
Maximum Marks	2 Marks

#### Question-1:

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

#### CODE 1 :

```
#include <WiFi.h>
#include <PubSubClient.h>
void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
#define ORG "0pmmjg"
#define DEVICE_TYPE "ESP32"
#define DEVICE_ID "06028"
#define TOKEN "12345678"
String data3;
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/Data/fmt/json";
char subscribetopic[] = "iot-2/cmd/test/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);
const int trigPin = 5;
const int echoPin = 18;
#define SOUND_SPEED 0.034
long duration;
float distance;
void setup() {
  Serial.begin(115200);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  wificonnect();
  mqttconnect();
}
void loop()
{
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  distance = duration * SOUND_SPEED/2;
  Serial.print("Distance (cm): ");
  Serial.println(distance);
  if(distance<100)
  {
    Serial.println("ALERT!!");
    delay(1000);
    PublishData(distance);
  }
}
```

```

delay(1000);
if (!client.loop()) {
  mqttconnect();
}
}
delay(1000);
}
void PublishData(float dist) {
  mqttconnect();
  String payload = "{\"Distance\":";
  payload += dist;
  payload += ", \"ALERT!!\": \"Distance less than 100cms\"";
  payload += "}";
  Serial.print("Sending payload: ");
  Serial.println(payload);

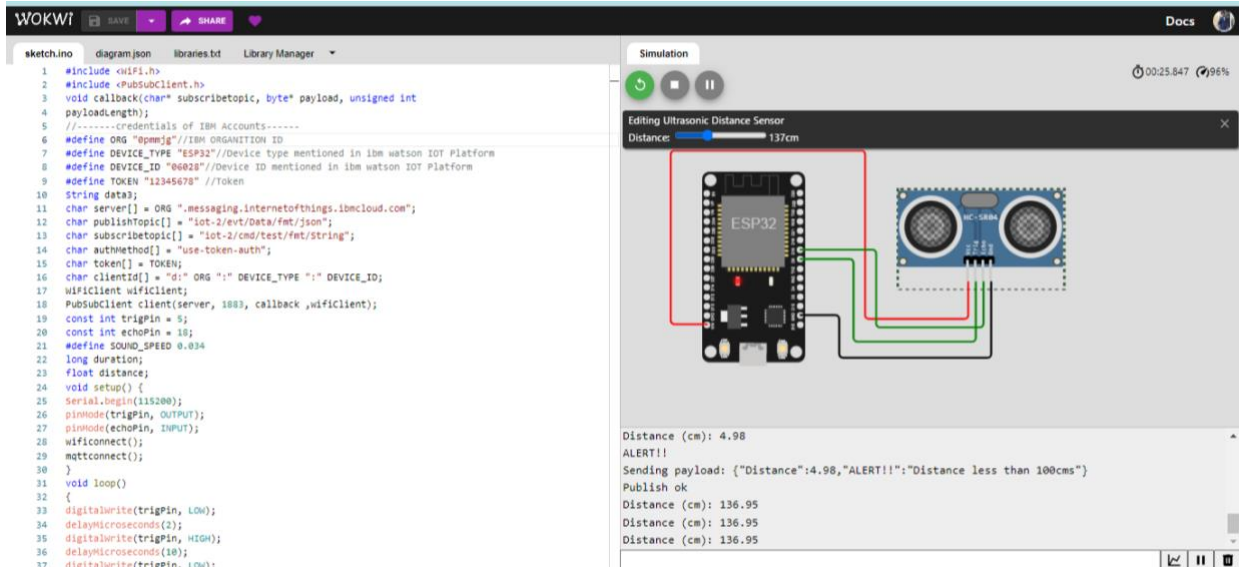
  if (client.publish(publishTopic, (char*) payload.c_str())) {
    Serial.println("Publish ok");
  } else {
    Serial.println("Publish failed");
  }
}
void mqttconnect() {
  if (!client.connected()) {
    Serial.print("Reconnecting client to ");
    Serial.println(server);
    while (!client.connect(clientId, 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++)
  {
    data3 += (char)payload[i];
  }
  Serial.println("data: " + data3);
  data3="";
}

```

Wokwi Link :

<https://wokwi.com/projects/346865633302938195>

Output and Simulation :



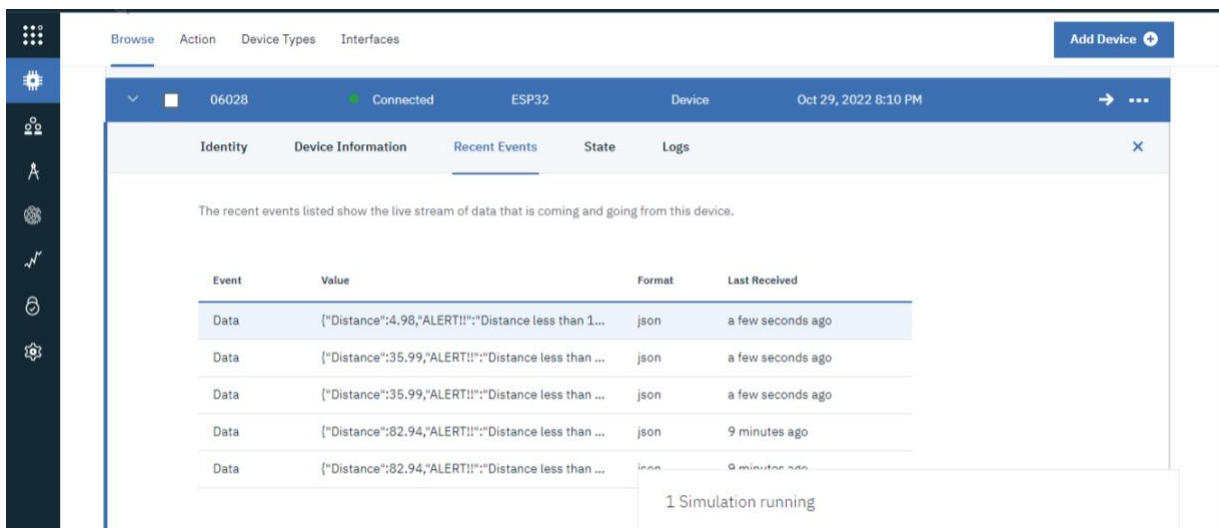
The screenshot shows the Wokwi IDE interface. On the left, the 'sketch.ino' file is open, displaying the following code:

```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 void callback(char* topic, byte* payload, unsigned int
4   payloadLength);
5 //-----Credentials of IBM Accounts-----
6 #define ORG "opmijg"//IBM ORGANIZATION ID
7 #define DEVICE_TYPE "ESP32"//Device type mentioned in ibm watson IOT Platform
8 #define DEVICE_ID "06028"//Device ID mentioned in ibm watson IOT Platform
9 #define TOKEN "12345678" //Token
10 String data;
11 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
12 char publishTopic[] = "iot-2/evt/data/fmt/json";
13 char subscribeTopic[] = "iot-2/cmd/test/fmt/string";
14 char authMethod[] = "use-token-auth";
15 char token[] = TOKEN;
16 char clientId[] = "di: ORG ":" DEVICE_TYPE ":" DEVICE_ID;
17 WiFiClient wifiClient;
18 PubSubClient client(server, 1883, callback, wifiClient);
19 const int trigPin = 5;
20 const int echoPin = 18;
21 #define SOUND_SPEED 0.034
22 long duration;
23 float distance;
24 void setup() {
25   Serial.begin(115200);
26   pinMode(trigPin, OUTPUT);
27   pinMode(echoPin, INPUT);
28   wifiConnect();
29   mqttConnect();
30 }
31 void loop() {
32 {
33   digitalWrite(trigPin, LOW);
34   delayMicroseconds(2);
35   digitalWrite(trigPin, HIGH);
36   delayMicroseconds(10);
37   digitalWrite(trigPin, LOW);
```

On the right, the 'Simulation' window shows a visual representation of the ESP32 and the HC-SR04 ultrasonic sensor. Below the simulation, the console output displays the following data:

```
Distance (cm): 4.98
ALERT!!
Sending payload: {"Distance":4.98,"ALERT!!":"Distance less than 100cms"}
Publish ok
Distance (cm): 136.95
Distance (cm): 136.95
Distance (cm): 136.95
```

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



The screenshot shows the IBM Watson IoT Platform interface. The 'Recent Events' tab is selected for device 06028. The table below lists the recent events:

Event	Value	Format	Last Received
Data	{"Distance":4.98,"ALERT!!":"Distance less than 100cms"}	json	a few seconds ago
Data	{"Distance":35.99,"ALERT!!":"Distance less than 100cms"}	json	a few seconds ago
Data	{"Distance":35.99,"ALERT!!":"Distance less than 100cms"}	json	a few seconds ago
Data	{"Distance":82.94,"ALERT!!":"Distance less than 100cms"}	json	9 minutes ago
Data	{"Distance":82.94,"ALERT!!":"Distance less than 100cms"}	json	9 minutes ago

At the bottom of the interface, a status bar indicates "1 Simulation running".