

# ASSIGNMENT 4

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.

## Wokwi link:

<https://wokwi.com/projects/347029515960058450>

## CODE:

```
#include <WiFi.h>

#include <PubSubClient.h>

#include <ArduinoJson.h>

WiFiClient wifiClient;

#define ORG "xnt2bc"

#define DEVICE_TYPE "Shalini"

#define DEVICE_ID "Shalini26"

#define TOKEN "Shalu@26"

#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-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="";

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(".");
      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)  
  {  
    DynamicJsonDocument doc(1024);  
    String payload;  
    doc["Distance Alert:"]=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");

}

}

}

}

```

## OUTPUT:

The screenshot displays the Wokwi IoT simulator interface. On the left, the 'sketch.ino' file is open, showing a C++ program for an ESP32. The code includes headers for `<WiFi.h>`, `<PubSubClient.h>`, and `<ArduinoJson.h>`. It defines a `WiFiClient` object, an MQTT server, a publish topic, a device ID, a token, and sensor pins. The main logic involves publishing distance data to the MQTT server.

On the right, the 'Simulation' window shows a virtual circuit with an ESP32 microcontroller and an Ultrasonic Distance Sensor. A dialog box for the sensor shows the distance is set to 65cm. Below the circuit, the console output shows the following sequence of events:

```

Publish OK
Sending payload: {"Distance Alert":64}
Publish OK
Sending payload: {"Distance Alert":64}
Publish OK

```

The bottom of the interface shows the Windows taskbar with the search bar and system tray icons.

The screenshot displays the IBM Watson IoT Platform interface. At the top, the browser address bar shows the URL: `xnt2bc.internetofofthings.ibmcloud.com/dashboard/devices/browse`. The dashboard header includes the IBM Watson IoT Platform logo and a user profile section with the email `922119106091@smartinternz.com` and ID `xnt2bc`.

The main content area is titled "Browse" and shows a list of devices. The first device is "Sha\_26.15deviceid" with a status of "Disconnected". The second device is "Shalini26" with a status of "Connected". The "Shalini26" device is selected, and its details are shown below.

The "Shalini26" device details page has tabs for "Identity", "Device Information", "Recent Events", "State", and "Logs". The "Recent Events" tab is active, showing a live stream of data. The text states: "The recent events listed show the live stream of data that is coming and going from this device."

The "Recent Events" table lists the following data:

Event	Value	Format	Last Received
Data	{"Distance Alert":88}	json	a few seconds ago
Data	{"Distance Alert":27}	json	a few seconds ago
Data	{"Distance Alert":88}	json	a few seconds ago
Data	{"Distance Alert":88}	json	a few seconds ago
Data	{"Distance Alert":64}	json	a few seconds ago

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