

ASSIGNMENT 4

DATE	12 November 2022
TEAM ID	PNT2022TMID36752
NAME	JAYACHANDRAN.R
Maximum Marks	2 Marks

Question 1:

Write code and connections in work for ultrasonic sensor. Whenever distance is less than 100cmssend"alert"toibmcloudanddisplayindevicerecentevents.

CODE:

```
#include <WiFi.h>
```

```
#include <PubSubClient.h>
```

```
void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
```

```
//-----credentials of IBM Accounts-----
```

```
#define ORG "910vsj"//IBM ORGANITION ID
```

```
#define DEVICE_TYPE "Assignment4"//Device type mentioned in ibm watson IOT Platform
```

```
#define DEVICE_ID "d2327"//Device ID mentioned in ibm watson IOT Platform
```

```
#define TOKEN "ibm12345" //Token
```

```
String data3;
```

```
float dist;
```

```
//----- 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); //calling the predefined client id by passing
parameter like server id,portand wificredential


int LED = 4;

int trig = 5;

int echo = 18;

void setup()
{
  Serial.begin(115200);
  pinMode(trig,OUTPUT);
  pinMode(echo,INPUT);
  pinMode(LED, OUTPUT);
  delay(10);
  wificonnect();
  mqttconnect();
}

void loop()// Recursive Function
{

  digitalWrite(trig,LOW);
  digitalWrite(trig,HIGH);
  delayMicroseconds(10);
  digitalWrite(trig,LOW);
  float dur = pulseIn(echo,HIGH);

```

```
float dist = (dur * 0.0343)/2;  
Serial.print ("Distancein cm");  
Serial.println(dist);
```

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

```
/.....retrieving to Cloud...../
```

```
void PublishData(float dist) {  
    mqttconnect();//function call for connecting to ibm  
    /*  
        creating the String in in form JSon to update the data to ibm cloud  
    */  
    String object;  
    if (dist <100)  
    {  
        digitalWrite(LED,HIGH);  
        Serial.println("object is near");  
        object = "Near";  
    }  
    else  
    {  
        digitalWrite(LED,LOW);
```

```
Serial.println("no object found");  
object = "No";  
}
```

```
String payload = "{\"distance\":";  
payload += dist;  
payload += "," "\"object\":"\"";  
payload += object;  
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 defination 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];  
    }  
    // Serial.println("data: "+ data3);  
    // if(data3=="Near")  
    // {  
    // Serial.println(data3);  
    // digitalWrite(LED,HIGH);  
  
    // }  
  
    // else  
    // {  
    // Serial.println(data3);  
    // digitalWrite(LED,LOW);  
  
    // }  
    data3="";
```

}

OUTPUT:

The screenshot shows the IBM Watson IoT Platform dashboard. The top navigation bar includes 'Browse', 'Action', 'Device Types', and 'Interfaces'. The main content area displays the details for a device named 'd2327', which is 'Connected' and assigned to 'Assignment4'. The 'Recent Events' tab is selected, showing a table of events. The table has columns for 'Event', 'Value', 'Format', and 'Last Received'. The events listed are all 'Data' events with a 'json' format, each containing a distance value and a 'No' object, received 'a few seconds ago'.

Event	Value	Format	Last Received
Data	{"distance":403.54,"object":"No"}	json	a few seconds ago
Data	{"distance":403.49,"object":"No"}	json	a few seconds ago
Data	{"distance":403.47,"object":"No"}	json	a few seconds ago
Data	{"distance":403.49,"object":"No"}	json	a few seconds ago
Data	{"distance":403.49,"object":"No"}	json	a few seconds ago

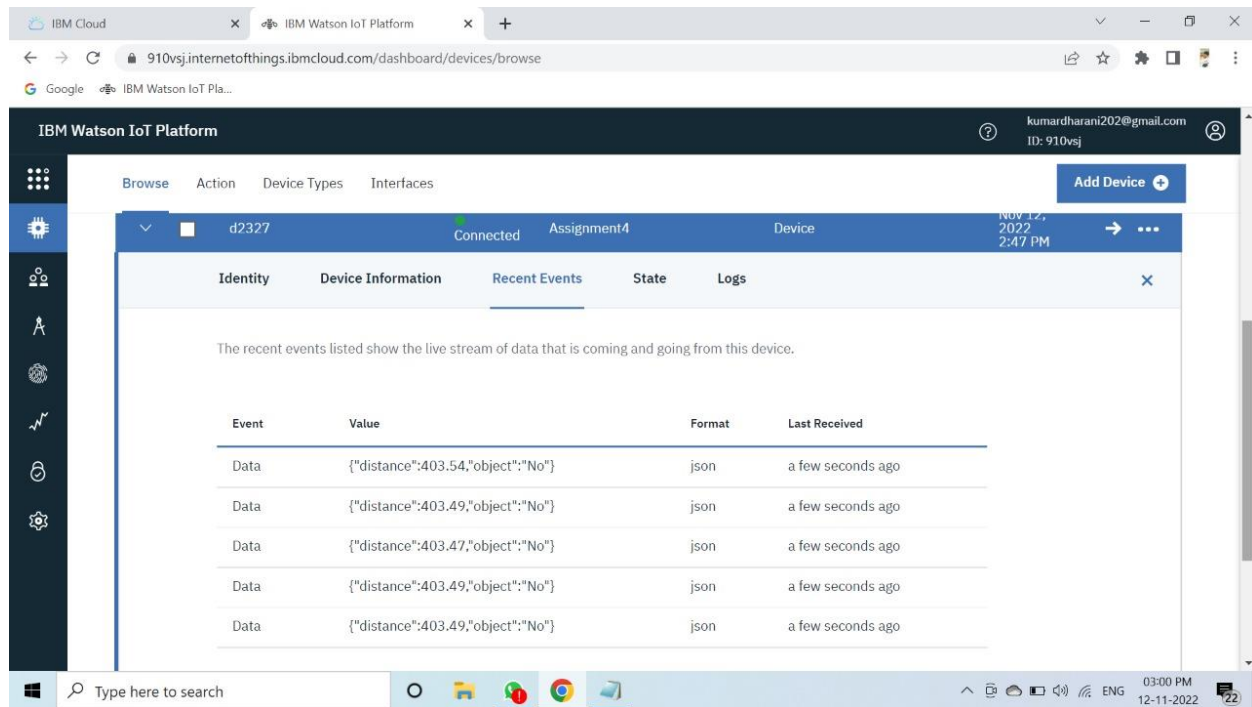
Data send to the IBM cloud device when the object is far

The screenshot shows the Wokwi Arduino IDE interface. The left pane displays a sketch named 'sketch.ino' with the following code:

```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3
4
5 void callback(char* topic, byte* payload, unsigned int length) {
6
7   //-----credentials of IBM Accounts-----
8
9   #define ORG "910vsj"//IBM ORGANIZATION ID
10  #define DEVICE_TYPE "Node mcu"//Device type mentioned in ibm watson IOT
11  #define DEVICE_ID "dharani2327"//Device ID mentioned in ibm watson IOT
12  #define TOKEN "dharani2327" //Token
13  String data;
14  float dist;
15  //----- Customise the above values -----
16  char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server
17  char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of
18  char subscribeTopic[] = "iot-2/cmd/test/fmt/String";// cmd REPRESENT co
19  char authMethod[] = "use-token-auth";// authentication method
20  char token[] = TOKEN;
21  char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id
22
23  //-----
24  WiFiClient wifiClient; // creating the instance for wifiClient
25  PubSubClient client(server, 1883, callback ,wifiClient); //calling the p
26
```

The right pane shows a simulation of the hardware setup, featuring an ESP32 microcontroller board connected to an HC-SR04 ultrasonic sensor via jumper wires. The sensor's VCC is connected to the ESP32's 5V pin, GND to GND, and the trig pin to a digital pin. The echo pin is connected to an analog pin.

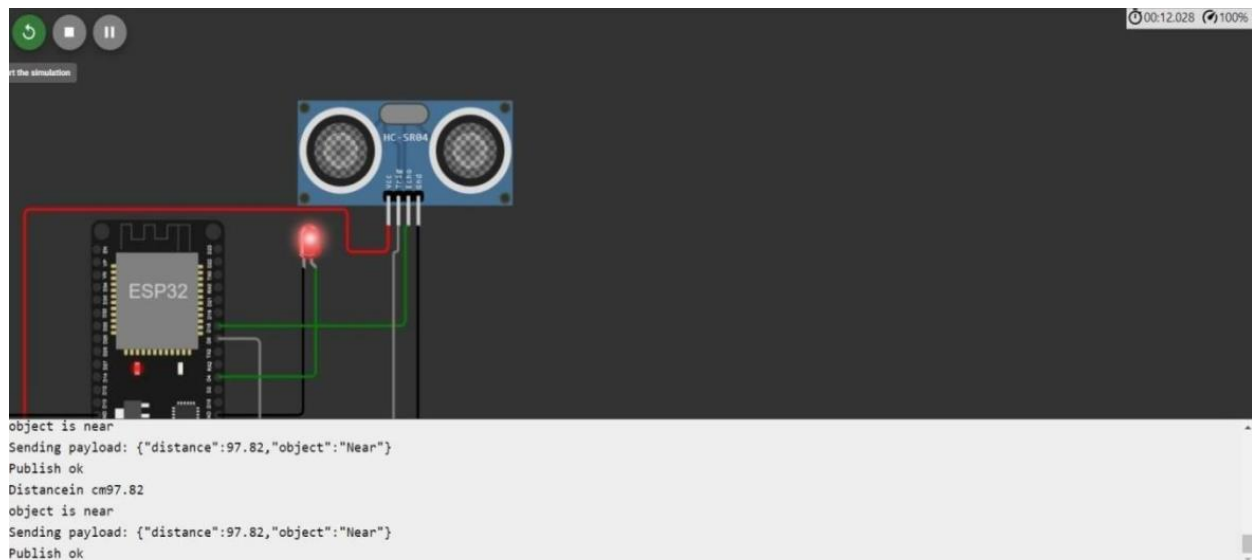
DatasenttotheIBMCloudDevicewhentheobjectisnear



The screenshot shows the IBM Watson IoT Platform dashboard. The top navigation bar includes 'Browse', 'Action', 'Device Types', and 'Interfaces'. A sidebar on the left contains icons for various functions. The main content area displays details for device 'd2327', which is 'Connected' and assigned to 'Assignment4'. The 'Recent Events' tab is selected, showing a table of events. The table has columns for 'Event', 'Value', 'Format', and 'Last Received'. The events are all 'Data' type, with values representing distance and object status in JSON format. The status is consistently 'No'.

Event	Value	Format	Last Received
Data	{"distance":403.54,"object":"No"}	json	a few seconds ago
Data	{"distance":403.49,"object":"No"}	json	a few seconds ago
Data	{"distance":403.47,"object":"No"}	json	a few seconds ago
Data	{"distance":403.49,"object":"No"}	json	a few seconds ago
Data	{"distance":403.49,"object":"No"}	json	a few seconds ago

Whenobjecticsneartotheultrasonicsensor



The screenshot shows a simulation of an ESP32 microcontroller connected to an HC-SR04 ultrasonic sensor. The sensor is connected to the ESP32 via a breadboard. A red LED is connected to the sensor's output. The simulation interface includes a 'Start the simulation' button and a status bar at the bottom showing '00:12.028' and '100%'. The console output shows the following messages:

```
object is near
Sending payload: {"distance":97.82,"object":"Near"}
Publish ok
Distancein cm97.82
object is near
Sending payload: {"distance":97.82,"object":"Near"}
Publish ok
```