

IBM ASSIGNMENT - 4

Date	02 November 2022
Team ID	PNT2022TMID01040
Name	Child Safety Monitoring and Notification

QUESTION :

Write code and connections in wokwi for ultrasonic sensor. Whenever distance is less than 100 cms send "alert" to IBM cloud and display in device recent events.

CODE :

```
#include <WiFi.h>           // library for wifi

#include <PubSubClient.h>    // library for MQTT

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

#define ORG "rwazv5"        // IBM organisation id
#define DEVICE_TYPE "NodeRed" // Device type mentioned in ibm watson iot
platform #define DEVICE_ID "12345" // Device ID mentioned in ibm
watson iot platform #define TOKEN "vC@S3TBre6(97jAOJ_" // Token
#define speed
0.034 #define led
14 String data3;
int LED = 4;

//.....customise above values.....-

char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; // server name
char publishTopic[] = "iot-2/evt/sreedhar/fmt/json"; // topic name and type of event perform
and format in which data to be send
char topic[] = "iot-2/cmd/led/fmt/String"; // cmd Represent type and command is test
format of strings
char authMethod[] = "use-token-auth"; // authentication method
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID; //Client id

//.....

WiFiClient wifiClient; // creating instance for wificlient
```

```
PubSubClient client(server, 1883, wifiClient); // calling the predefined client id by passing parameter like server id,port and  
wifi credential
```

```
const int trigpin=5;  
const int echopin=18;  
String  
command;  
String data="";
```

```
long duration;  
float dist;
```

```
void setup()  
{  
  Serial.begin(115200);  
  pinMode(led, OUTPUT);  
  pinMode(trigpin, OUTPUT);  
  pinMode(echopin, INPUT);  
  wifiConnect(); mqttConnect();  
}
```

```
void loop() { bool isNearby  
=    dist    <  
    100;  
digitalWrite(led,  
isNearby);
```

```
publishData(  
); delay(500);
```

```
if (!client.loop())  
{  
  mqttConnect();          // function call to connect to ibm  
}  
}
```

```
/* .....retrieving to cloud..... */
```

```
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(500);
        }
        initManagedDevice();
        Serial.println();
    }
}

void
initManagedDevice() { if
(client.subscribe(topic))
{
    Serial.println("IBM 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)
    {
        digitalWrite(LED, HIGH);
        String payload = "{\"Alert
Distance\"."; payload += dist;
        payload += "}";
    }
}

```

```

Serial.print("\n");
Serial.print("Sending payload: "); Serial.println(payload); if (client.publish(publishTopic, (char*)
payload.c_str())) // if data is uploaded to cloud successfully,prints publish ok else prints publish
failed
{
    Serial.println("Publish OK");
}

}

if(dist>100)
{
    digitalWrite(LED,HIGH);
    String payload =
    "{\"Distance\":\""; payload +=
    dist;
    payload += "}";

    Serial.print("\n");
    Serial.print("Sending payload:
    "); Serial.println(payload);
    if(client.publish(publishTopic, (char*) payload.c_str()))
    {
        Serial.println("Publish OK");
    }
    else
    {
        digitalWrite(LED,LOW);
        Serial.println("Publish FAILED");
    }

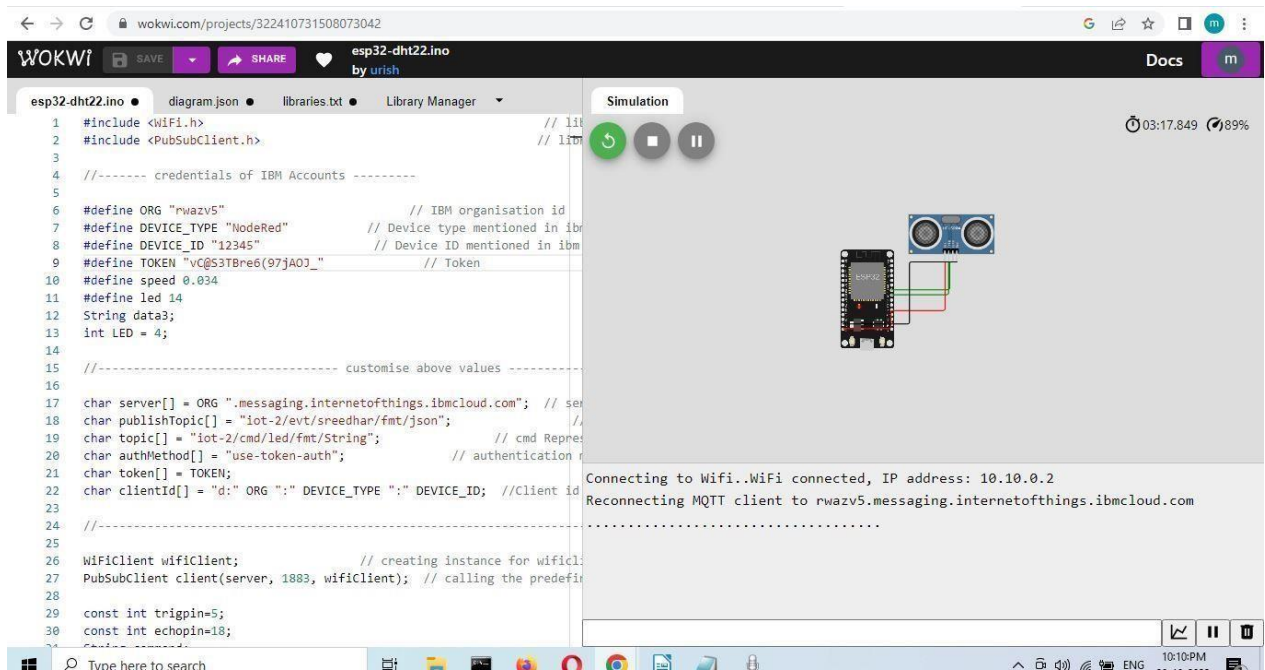
}

}

```

OUTPUT :

Code simulation on wokwi



Data sent to IBM Cloud with distance

