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

#include <WiFiClient.h>

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

// creating the instance by passing pin and typr of dht connected

 float distance;

#define sound\_speed 0.034

 int trigpin=18;

 int echopin=19;

 int led=5;

 int LED=9;

 long duration;

 String message;// creating the instance by passing pin and typr of dht connected

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

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

#define ORG "q4wtcm"//IBM ORGANITION ID

#define DEVICE\_TYPE "Arduino"//Device type mentioned in ibm watson IOT Platform

#define DEVICE\_ID "123456"//Device ID mentioned in ibm watson IOT Platform

#define TOKEN "12345678"     //Token

String data3;

float h, t;

//-------- 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/command/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

void setup()// configureing the ESP32

{

**Serial**.begin(115200);

pinMode(trigpin,OUTPUT);

pinMode(echopin,INPUT);

pinMode(led,OUTPUT);

  delay(10);

**Serial**.println();

  wificonnect();

  mqttconnect();

}

void loop()// Recursive Function

{

digitalWrite(trigpin,LOW);

digitalWrite(trigpin,HIGH);

delay(1000);

digitalWrite(trigpin,LOW);

duration=pulseIn(echopin,HIGH);

distance=duration\*sound\_speed/2;

**Serial**.println("distance"+String(distance)+"cm");

 if(distance<100)

 {

   message="Alert";

   digitalWrite(led,HIGH);

   } else

{

  message="No problem";

  digitalWrite(led,LOW);

  }

  delay(1000);

  PublishData(distance,message);

  // if (!client.loop()) {

  //   mqttconnect();

  // }

}

/\*.....................................retrieving to Cloud...............................\*/

void PublishData(float d, String a) {

  mqttconnect();//function call for connecting to ibm

  /\*

     creating the String in in form JSon to update the data to ibm cloud

  \*/

  String payload = "{\"distance\":";

  payload += d; payload += "}";

  payload += "," "{\"message\":";

  payload += a;

  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=="lighton")

  {

**Serial**.println(data3);

digitalWrite(LED,HIGH);

  }

  else

  {

**Serial**.println(data3);

digitalWrite(LED,LOW);

  }

data3="";

}

**OUTPUT SNIPS**





