Assignment - 4

Student Name	Sujitha R
Team ID	PNT2022TMID53651
Project name	Project - Industry Specific Intelligent Fire Management system

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. Upload document with wokwi share link and images of IBM cloud.

SOLUTION:

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


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

// -------credentials of IBM Accounts-----
#define ORG "j0mda0"//IBM ORGANITION ID
#define DEVICE_TYPE "Sujitha"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "2301"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "Sujitha@2301" //Token

String data3;
float distance;
#define sound_speed 0.034
int trigpin=18;
int echopin=19;
int led=5;
int LED=9;
long duration;
String message;
```

```
//-------- 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

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 += "}";
```

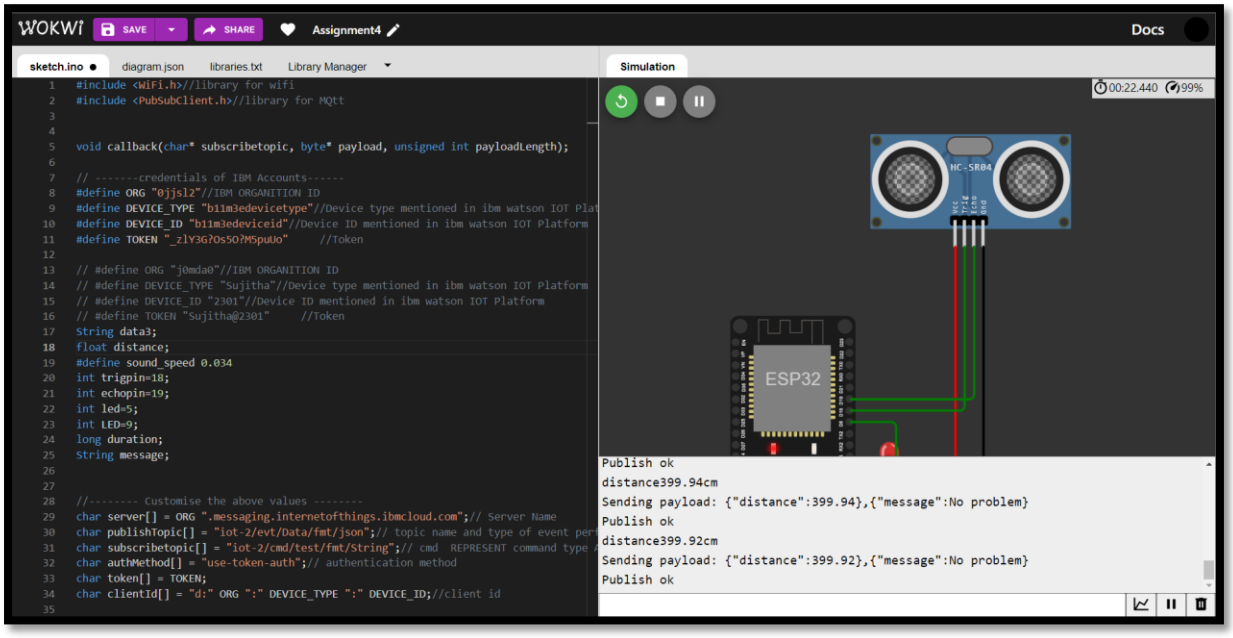
```
  payload += "," "{\"message\":";
  payload += a;
  payload += "}";

  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);
```

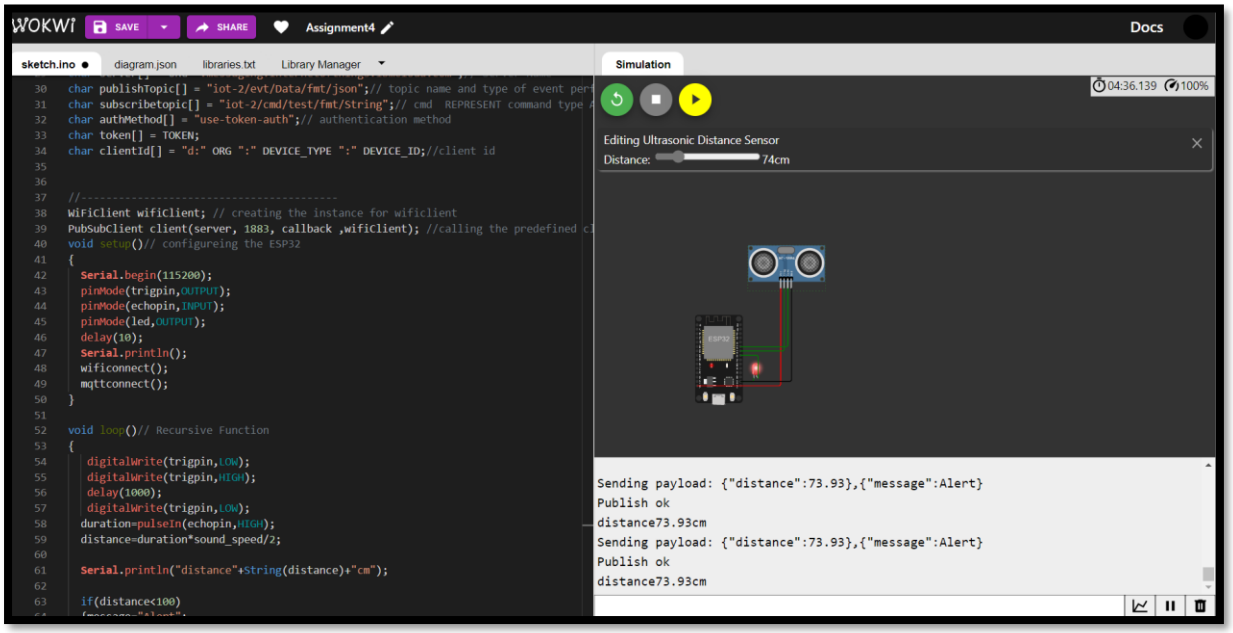
```
  if(data3=="lighton")
  {
Serial.println(data3);
digitalWrite(LED,HIGH);

  }

  else
  {
Serial.println(data3);
digitalWrite (LED,LOW);

  }
data3="";



}
```

WOKWI OUTPUT:

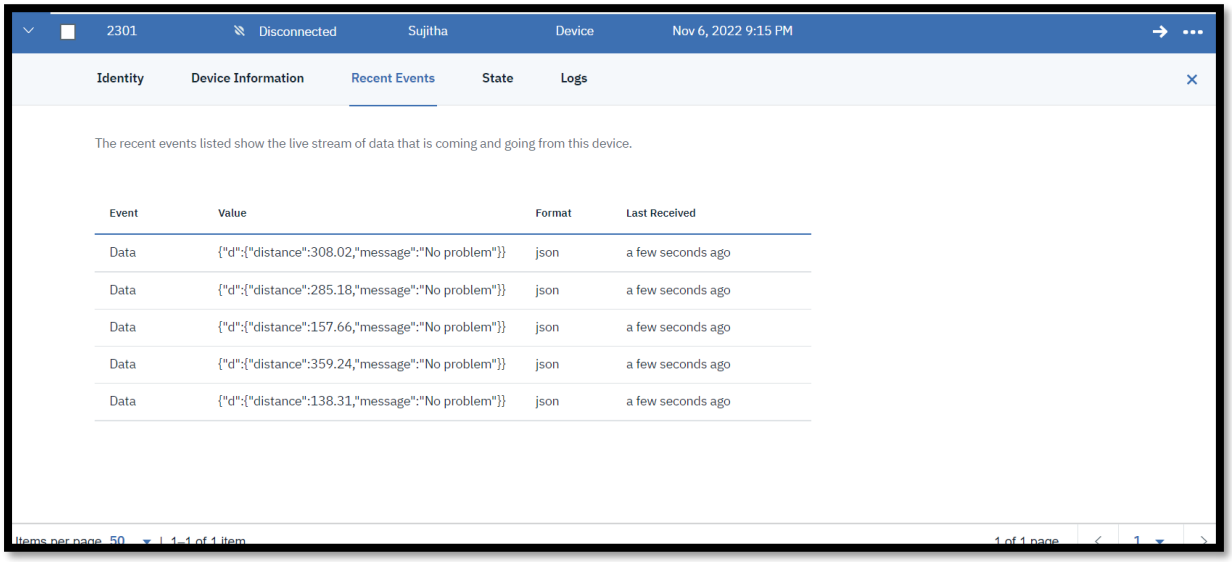
DISTANCE GREATER THAN 100:



DISTANCE LESSER THAN 100:



DEVICE RECENT EVENTS IN IBM WATSON:



WOWKI LINK:

https://wokwi.com/projects/347594038348612180