ASSIGNMENT 4

Project Name: IOT BASED SAFETY GADGET FOR CHILD SAFETY MONITORING AND NOTIFICATION

Batch Number: B5-51ME

Team members

Team Leader: m.parkavi
Team member1:R.nivetha
Team member2:J.shabina fathima
Team member3:S.sella Pavithra
Team member4:K.sobika

Question1 :

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.

WOKWI LINK :

https://wokwi.com/projects/305566932847821378

CODE :

```
#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 "4hn0jp"//IBM ORGANITION ID
#define DEVICE_TYPE "ULTRASON"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "DISTANCEDETECT"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "wuo5s?PR)2SegVk&Rx"//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);
```

esp32-blink.ino ● diagram.json ● libraries.txt ● Library Manager

```
36  pinMode(trig,OUTPUT);
37  pinMode(echo,INPUT);
38  pinMode(LED, OUTPUT);
39  delay(10);
40  wificonnect();
41  mqttconnect();
42  }
43  void loop()// Recursive Function
44  {
45
46   digitalWrite(trig,LOW);
47    digitalWrite(trig,HIGH);
48    delayMicroseconds(10);
49    digitalWrite(trig,LOW);
50    float dur = pulseIn(echo,HIGH);
51    float dist = (dur * 0.0343)/2;
52    Serial.print ("Distancein cm");
53    Serial.println(dist);
54
55
56    PublishData(dist);
57    delay(1000);
58    if (!client.loop()) {
59      mqttconnect();
60    }
61  }
62
63
64
65  /*.....retrieving to Cloud.....*/
66
67  void PublishData(float dist) {
68    mqttconnect();//function call for connecting to ibm
69    /*
70       creating the String in in form JSon to update the data to ibm cloud
```

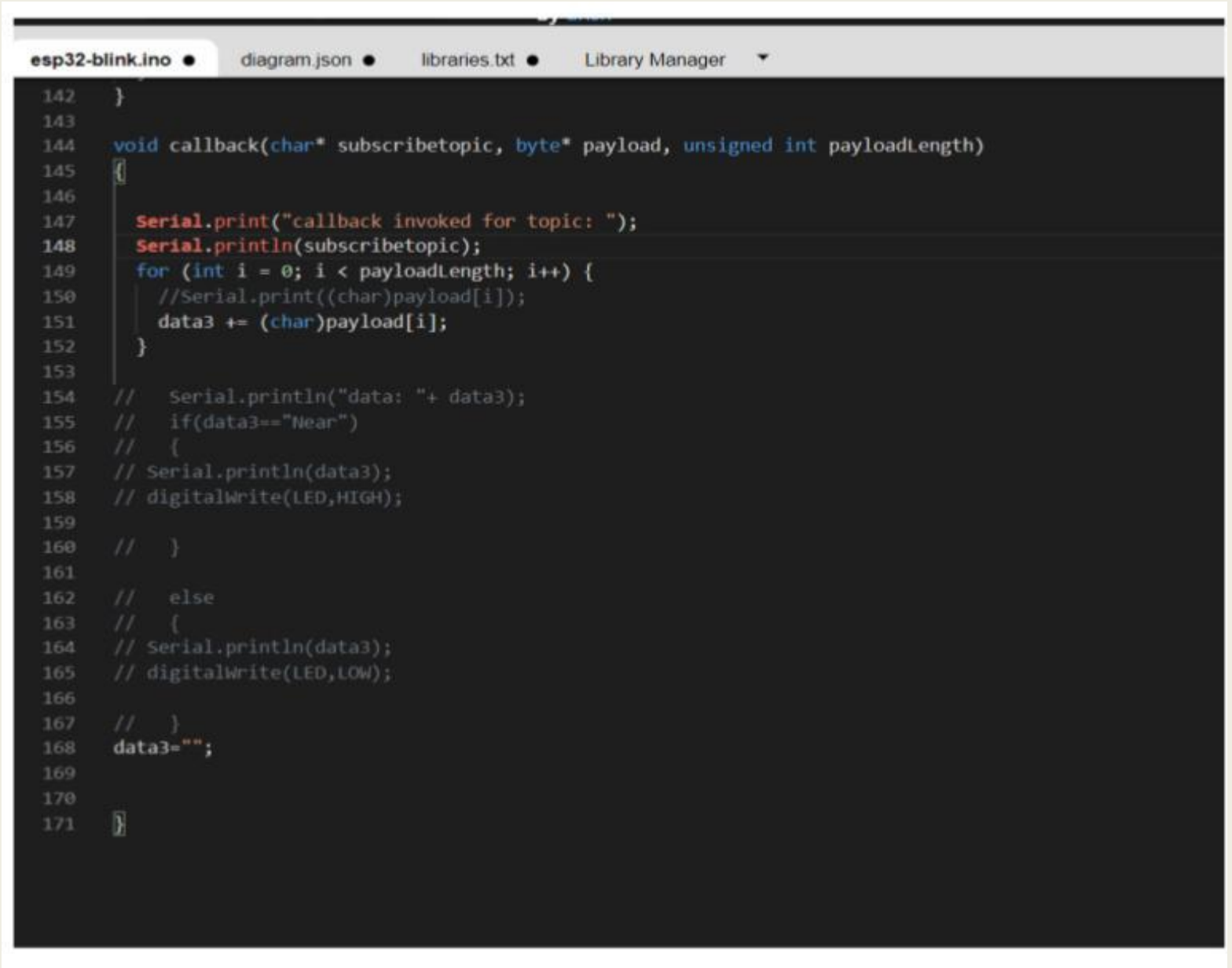
```
    creating the String in in form JSon to update the data to ibm cloud
    */
    String object;
    if (dist <100)
    {
      digitalWrite(LED,HIGH);
      Serial.println("object is near");
      object = "Near";
    }
    else
    {
      digitalWrite(LED,LOW);
      Serial.println("no object found");
      object = "No";
    }

    String payload = "{\"distance\":";
    payload += dist;
    payload += "," "\"object\":\"";
    payload += object;
    payload += "\"}";


    Serial.print("Sending payload: ");
    Serial.println(payload);
```

esp32-blink.ino ● diagram.json ● libraries.txt ● Library Manager

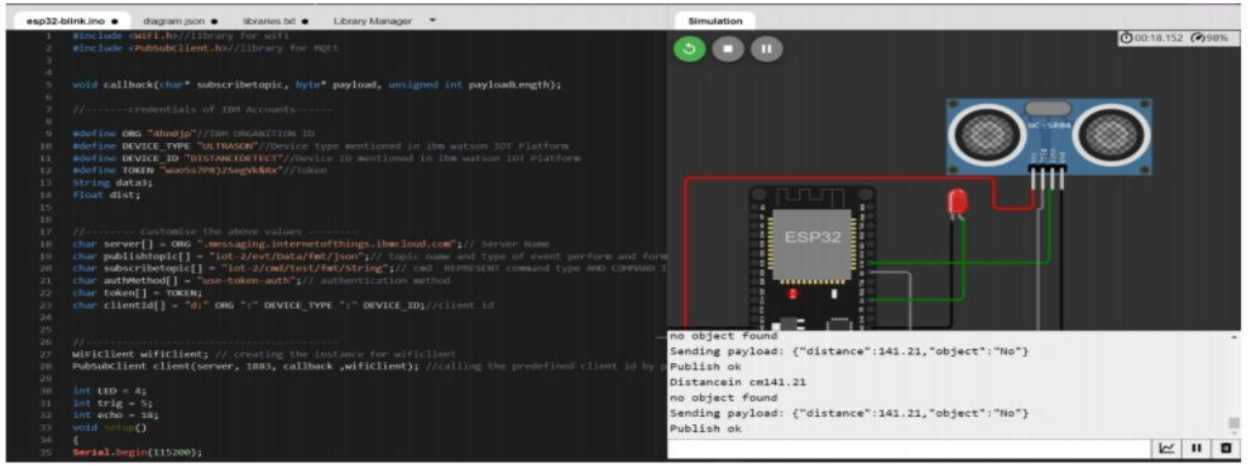
```
    if (client.publish(publishTopic, (char*) payload.c_str())) {
      Serial.println("Publish ok");// if it sucessfully upload data on the cloud then it will print publish ok in Serial monitor or else it will print publish failed
    } else {
      Serial.println("Publish failed");
    }

}
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());
```

esp32-blink.ino ● diagram.json ● libraries.txt ● Library Manager

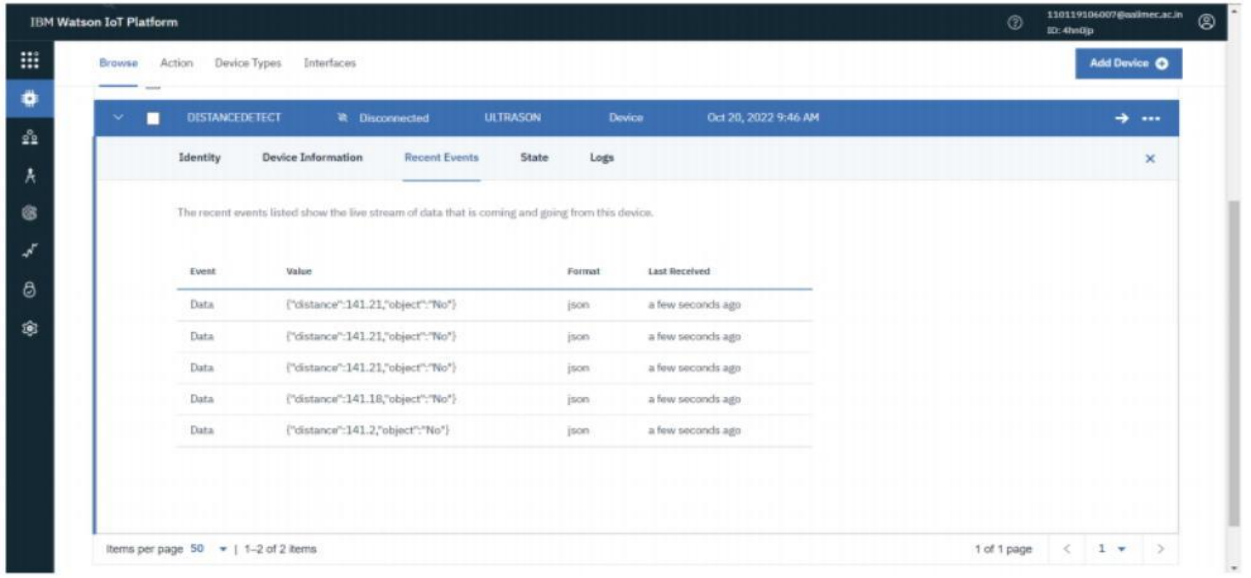
```
124   WiFi.begin("Wokwi-GUEST", "", 6);//passing the wifi credentials to establish the connection
125   while (WiFi.status() != WL_CONNECTED) {
126     delay(500);
127     Serial.print(".");
128   }
129   Serial.println("");
130   Serial.println("WiFi connected");
131   Serial.println("IP address: ");
132   Serial.println(WiFi.localIP());
133 }
134
135 void initManagedDevice() {
136   if (client.subscribe(subscribetopic)) {
137     Serial.println((subscribetopic));
138     Serial.println("subscribe to cmd OK");
139   } else {
140     Serial.println("subscribe to cmd FAILED");
141   }
142 }
143
144 void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
145 {
146
147   Serial.print("callback invoked for topic: ");
148   Serial.println(subscribetopic);
149   for (int i = 0; i < payloadLength; i++) {
150     //Serial.print((char)payload[i]);
151     data3 += (char)payload[i];
152   }
153
154 //   Serial.println("data: "+ data3);
155 //   if(data3=="Near")
156 //   {
157 // Serial.println(data3);
```


esp32-blink.ino
diagram.json
libraries.txt
Library Manager
}
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="";
}

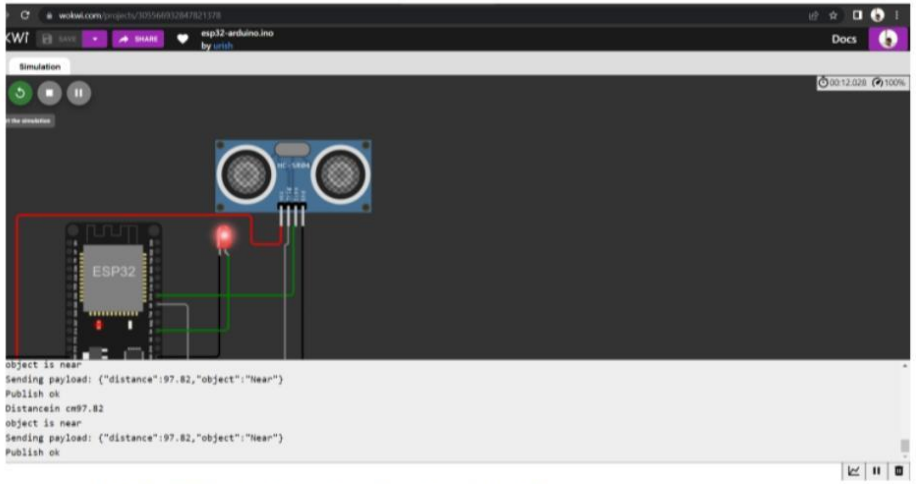
OUTPUT:



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



when object is near to the ultrasonic sensor



Data sent to the IBM Cloud Device when the object is near