Assignment -4

Assignment Date	30 October 2022
Student Name	KAVIRAJ N S
Maximum Marks	2 Marks
Student Roll Number	737819CSR86

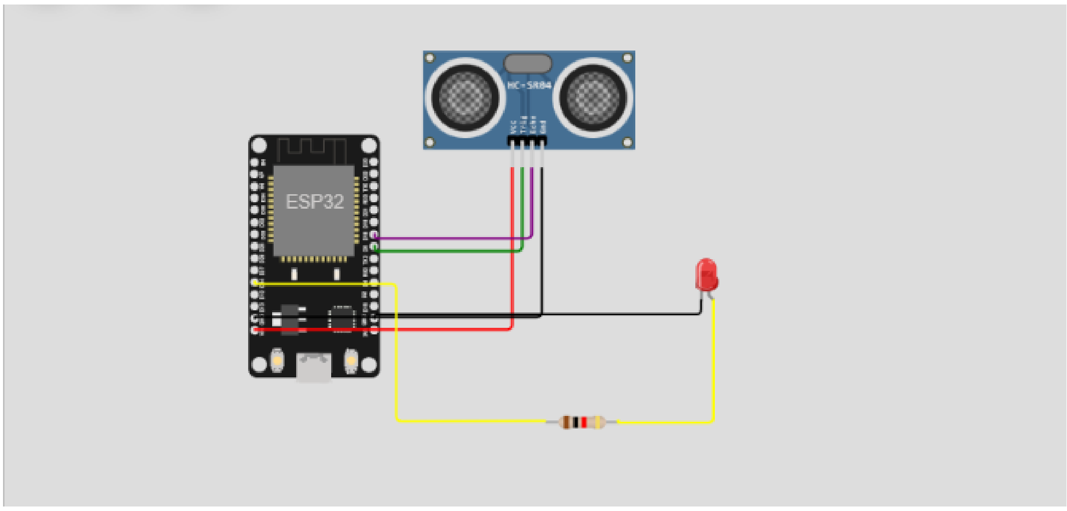
Question-1:

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

CODE:

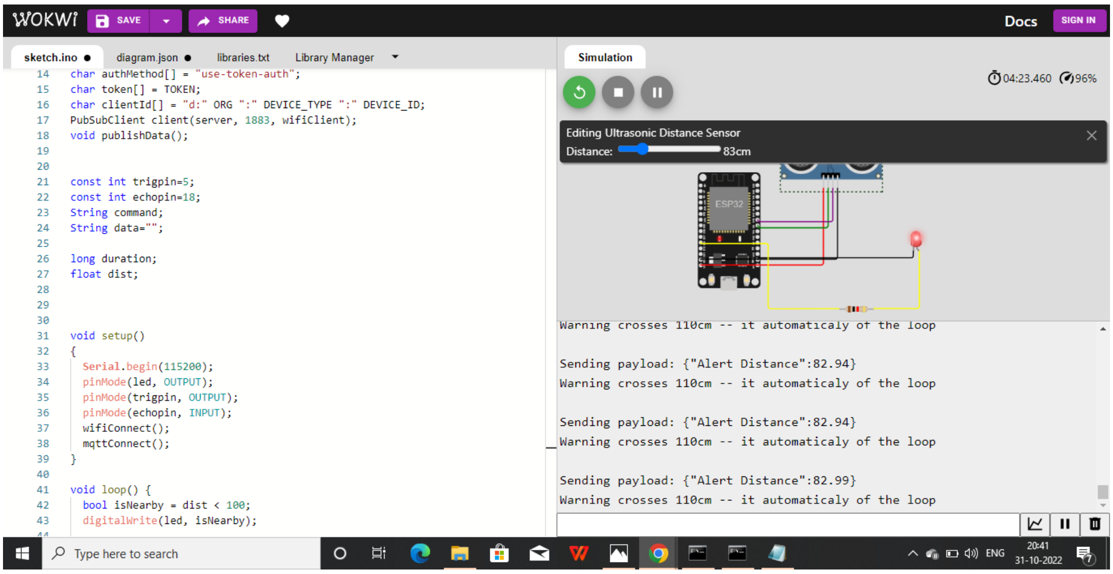
```
#include <WiFi.h>

#include <PubSubClient.h>

WiFiClient wifiClient;

String data3;


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

#define ORG "4raljz"//IBM ORGANITION ID

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

#define DEVICE_ID "assignment4"//Device ID mentioned in ibm watson IOT
Platform

#define TOKEN "vI?&bPDhf&I8q!W62k"//Token

#define speed 0.034

#define led 14


//-------- 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 topic[] = "iot-2/cmd/home/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

//-----------------------------------------

PubSubClient client(server, 1883, wifiClient);

void publishData();

const int trigpin=5;

const int echopin=18;

String command;

String data="";

long duration;

float dist;

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

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(1000);

  if (!client.loop()) {
    mqttConnect();
  }
}

void wifiConnect() {
  Serial.print("Connecting to "); Serial.print("Wifi");
  WiFi.begin("Wokwi-GUEST", "", 6);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    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(1000);
    }
    initManagedDevice();
    Serial.println();
  }
}

void initManagedDevice() {
  if (client.subscribe(topic)) {
    // Serial.println(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){
    String payload = "{\"Normal 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");
    }

  }
    if(dist>101 && dist<111){
    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())) {
      Serial.println("Warning crosses 110cm -- it automaticaly of the loop");
      digitalWrite(led,HIGH);
    }else {
      Serial.println("Publish 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++){
    dist += (char)payload[i];
  }
  Serial.println("data:"+ data3);
  if(data3=="lighton"){
    Serial.println(data3);
    digitalWrite(led,HIGH);
  }
  data3="";
}
```

CIRCUIT DIAGRAM:



OUTPUT:

1) **When Distance < 100 cm, alert with warning message occurs.**



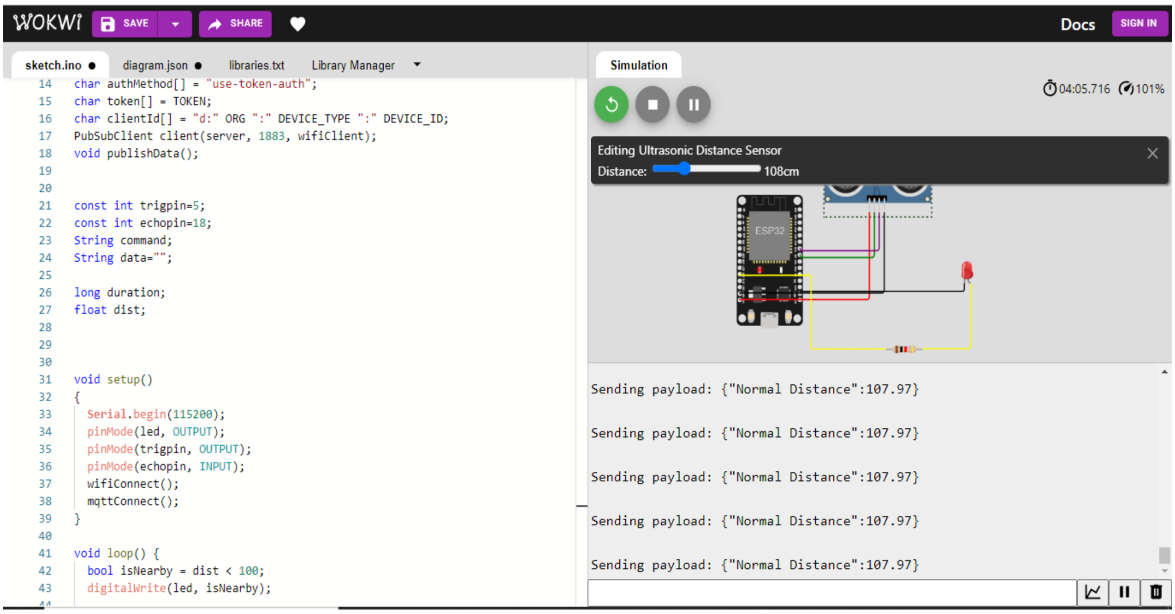
```
char authMethod[] = "use-token-auth";
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
PubSubClient client(server, 1883, wifiClient);
void publishData();


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);
```

Editing Ultrasonic Distance Sensor

Distance: 83cm

```
Warning crosses 110cm -- it automaticaly of the loop

Sending payload: {"Alert Distance":82.94}
Warning crosses 110cm -- it automaticaly of the loop

Sending payload: {"Alert Distance":82.94}
Warning crosses 110cm -- it automaticaly of the loop

Sending payload: {"Alert Distance":82.99}
Warning crosses 110cm -- it automaticaly of the loop
```

2) When distance > 100cm <110cm, it will show normal distance

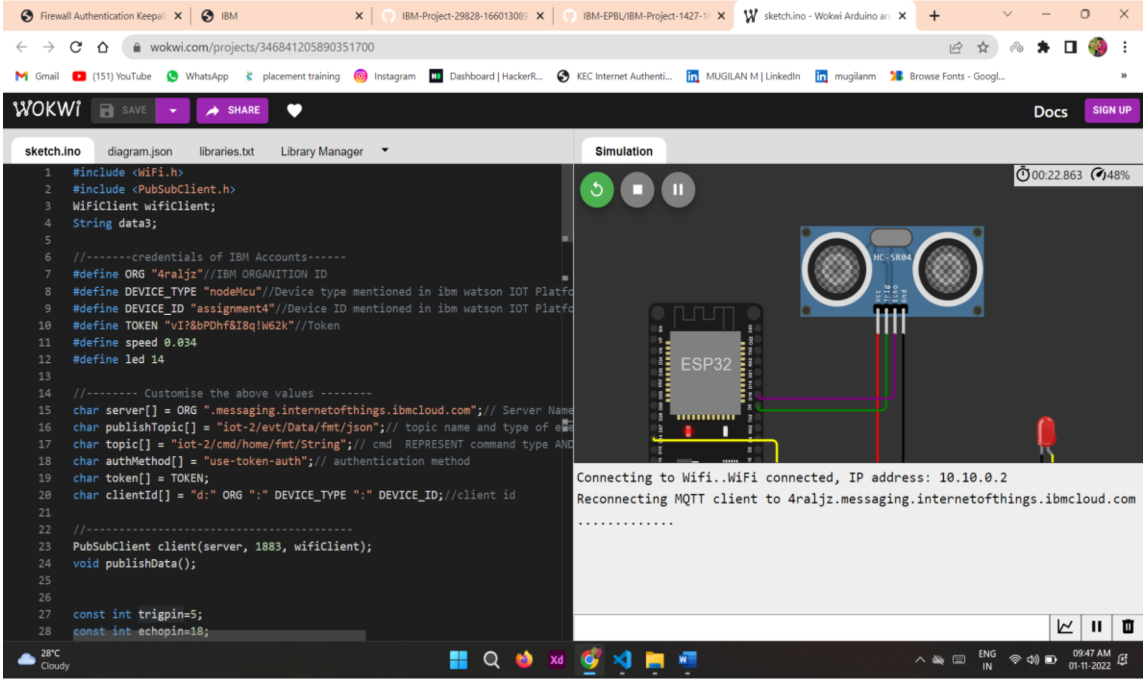


```
char authMethod[] = "use-token-auth";
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
PubSubClient client(server, 1883, wifiClient);
void publishData();


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);
```

Editing Ultrasonic Distance Sensor

Distance: 108cm

```
Sending payload: {"Normal Distance":107.97}

Sending payload: {"Normal Distance":107.97}

Sending payload: {"Normal Distance":107.97}

Sending payload: {"Normal Distance":107.97}

Sending payload: {"Normal Distance":107.97}
```



```
#include <WiFi.h>
#include <PubSubClient.h>
WiFiClient wifiClient;
String data3;

//-------credentials of IBM Accounts------
#define ORG "4raljz"//IBM ORGANITION ID
#define DEVICE_TYPE "nodeMcu"//Device type mentioned in ibm watson IOT Platfo
#define DEVICE_ID "assignment4"//Device ID mentioned in ibm watson IOT Platfo
#define TOKEN "vI?&bPDhf&I8q!W62k"//Token
#define speed 0.034
#define led 14

//-------- 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 e
char topic[] = "iot-2/cmd/home/fmt/String";// cmd  REPRESENT command type AND
char authMethod[] = "use-token-auth";// authentication method
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id

//-----------------------------------------
PubSubClient client(server, 1883, wifiClient);
void publishData();


const int trigpin=5;
const int echopin=18;
```

```
Connecting to Wifi..WiFi connected, IP address: 10.10.0.2
Reconnecting MQTT client to 4raljz.messaging.internetofthings.ibmcloud.com
.............
```