Assignment-4

Name	Thulasiram.B
Roll Number	201619205048

Problem Statement:

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

Source Code:

```
#include <WiFi.h>
#include <PubSubClient.h>
void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
#define ORG "vkk3lh"//IBM ORGANITION ID
#define DEVICE_TYPE "ESP-32"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "2019504030"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "9876543210" //Token
String data3;
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/distance/fmt/json";
char subscribeTopic[] = "iot-2/cmd/test/fmt/String";
char authMethod[] = "use-token-auth";
char token[] = TOKEN;
char clientID[] = "d:"ORG":"DEVICE_TYPE":"DEVICE_ID;
WiFiClient wifiClient;
PubSubClient client(server, 1883, callback, wifiClient);
#define ECHO_PIN 12
#define TRIG_PIN 13
#define led 2
void setup() {
// put your setup code here, to run once:
Serial.begin(115200);
pinMode(led, OUTPUT);
pinMode(TRIG_PIN, OUTPUT);
pinMode(ECHO_PIN, INPUT);
wificonnect();
mqttconnect();
}
float readDistanceCM() {
digitalWrite(TRIG_PIN, LOW);// Clear the trigger
delayMicroseconds(2);
digitalWrite(TRIG_PIN, HIGH);// Sets the trigger pin to HIGH state for 10 microseconds
delayMicroseconds(10);
digitalWrite(TRIG_PIN, LOW);
int duration = pulseIn(ECHO_PIN, HIGH);
//Serial.println(duration);
```

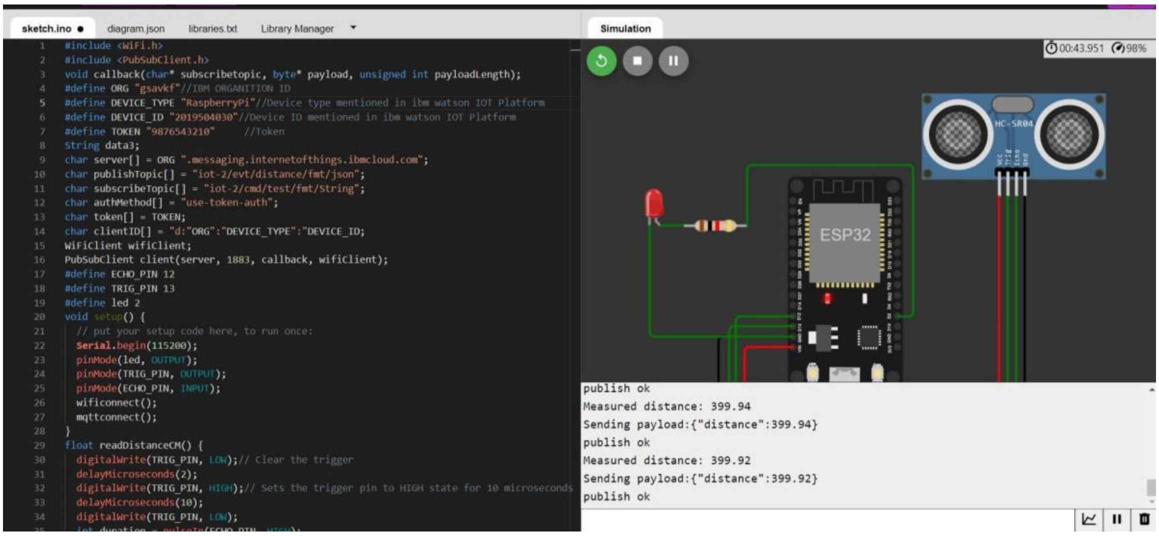
```
//duration = pulseIn(ECHO_PIN, HIGH);
return duration * 0.017;
//Serial.println(duration);
}
void loop() {
float distance = readDistanceCM();
//Serial.println(distance);
bool isNearby = distance < 100;
digitalWrite(led, isNearby);
Serial.print("Measured distance: ");
Serial.println(distance);
if (distance < 100) {
PublishData2(distance);
} else {
PublishData1(distance);
}
//PublishData(distance);
delay(1000);
if (!client.loop()) {
mqttconnect();
}
//delay(2000);
}
void PublishData1(float dist) {
mqttconnect();
String payload = "{\"distance\":";
payload += dist;
payload += "}";
Serial.print("Sending payload:");
Serial.println(payload);
if (client.publish(publishTopic, (char*)payload.c_str())) {
Serial.println("publish ok");
} else {
Serial.println("publish failed");
}
}
void PublishData2(float dist) {
mqttconnect();
String payload = "{\"ALERT\":";
payload += dist;
payload += "}";
Serial.print("Sending payload:");
Serial.println(payload);
if (client.publish(publishTopic, (char*)payload.c_str())) {
Serial.println("publish ok");
} else {
Serial.println("publish failed");
}
}
void mqttconnect() {
if (!client.connected()) {
Serial.print("Reconnecting to ");
Serial.println(server);
while (!!!client.connect(clientID, authMethod, token)) {
Serial.print(".");
delay(500);
}
initManagedDevice();
Serial.println();
}
}
void wificonnect() {
Serial.println();
Serial.print("Connecting to");
WiFi.begin("Wokwi-GUEST", "", 6);
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++) {
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 link

https://wokwi.com/projects/347143134975623762

output :

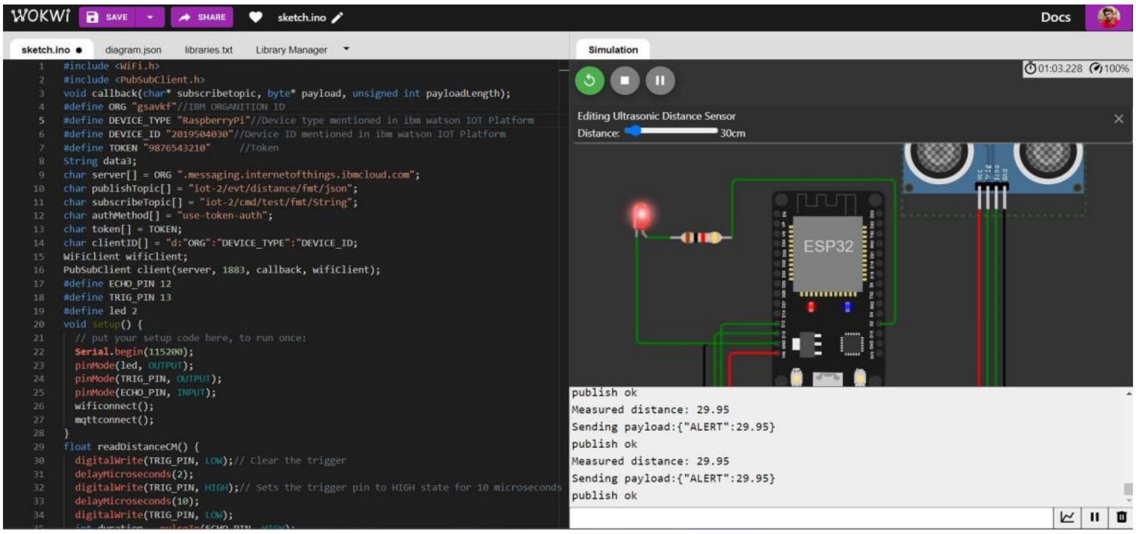
Normal Case:



```
#include <WiFi.h>
#include <PubSubClient.h>
void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
#define ORG "gsavkf"//IBM ORGANITION ID
#define DEVICE_TYPE "RaspberryPi"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "2019504030"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "9876543210"     //Token
String data3;
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/distance/fmt/json";
char subscribeTopic[] = "iot-2/cmd/test/fmt/String";
char authMethod[] = "use-token-auth";
char token[] = TOKEN;
char clientID[] = "d:"ORG":"DEVICE_TYPE":"DEVICE_ID;
WiFiClient wifiClient;
PubSubClient client(server, 1883, callback, wifiClient);
#define ECHO_PIN 12
#define TRIG_PIN 13
#define led 2
void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);
  pinMode(led, OUTPUT);
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
  wificonnect();
  mqttconnect();
}
float readDistanceCM() {
  digitalWrite(TRIG_PIN, LOW);// Clear the trigger
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);// Sets the trigger pin to HIGH state for 10 microseconds
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);
  int duration = pulseIn(ECHO_PIN, HIGH);
```

```
publish ok
Measured distance: 399.94
Sending payload:{"distance":399.94}
publish ok
Measured distance: 399.92
Sending payload:{"distance":399.92}
publish ok
```

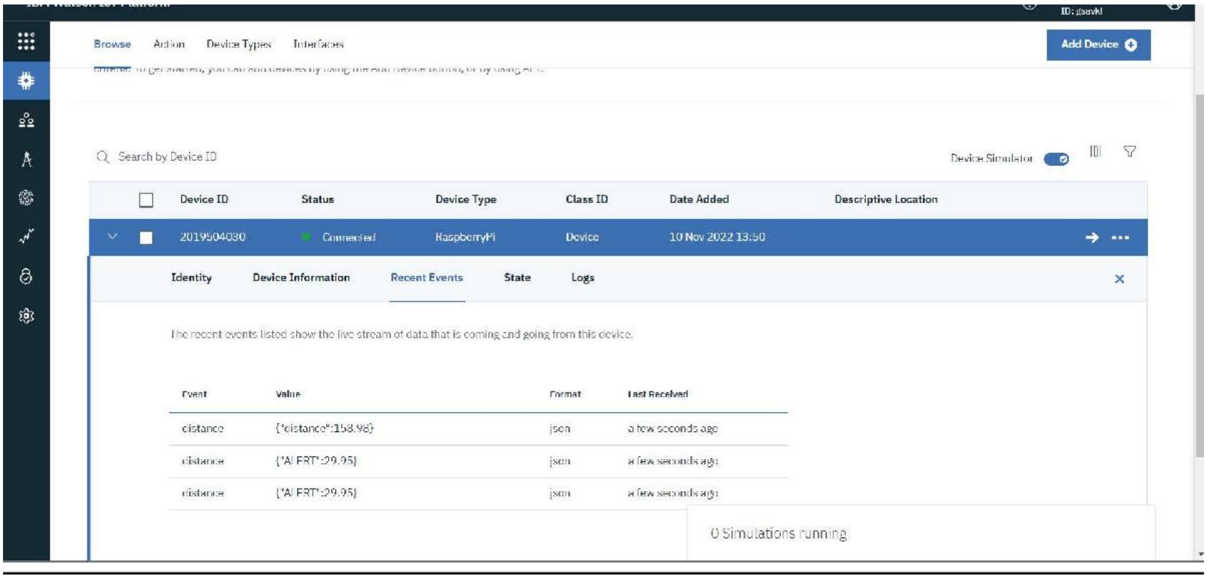
Alert Case:



```
#include <WiFi.h>
#include <PubSubClient.h>
void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
#define ORG "gsavkf"//IBM ORGANITION ID
#define DEVICE_TYPE "RaspberryPi"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "2019504030"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "9876543210"     //Token
String data3;
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/distance/fmt/json";
char subscribeTopic[] = "iot-2/cmd/test/fmt/String";
char authMethod[] = "use-token-auth";
char token[] = TOKEN;
char clientID[] = "d:"ORG":"DEVICE_TYPE":"DEVICE_ID;
WiFiClient wifiClient;
PubSubClient client(server, 1883, callback, wifiClient);
#define ECHO_PIN 12
#define TRIG_PIN 13
#define led 2
void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);
  pinMode(led, OUTPUT);
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
  wificonnect();
  mqttconnect();
}
float readDistanceCM() {
  digitalWrite(TRIG_PIN, LOW);// Clear the trigger
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);// Sets the trigger pin to HIGH state for 10 microseconds
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);
  int duration = pulseIn(ECHO_PIN, HIGH);
```

```
publish ok
Measured distance: 29.95
Sending payload:{"ALERT":29.95}
publish ok
Measured distance: 29.95
Sending payload:{"ALERT":29.95}
publish ok
```

IBM Cloud Storage:



□	Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
□	2019504030	● Connected	RaspberryPi	Device	10 Nov 2022 13:50	

The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
distance	{"distance":158.98}	json	a few seconds ago
distance	{"ALERT":29.95}	json	a few seconds ago
distance	{"ALERT":29.95}	json	a few seconds ago

0 Simulations running