SPRINT - 03

Date :	07 November 2022
Team ID :	PNT2022TMID47905
Project Name	SIGNS WITH SMART CONNECTIVITY FOR BETTER ROAD SAFETY

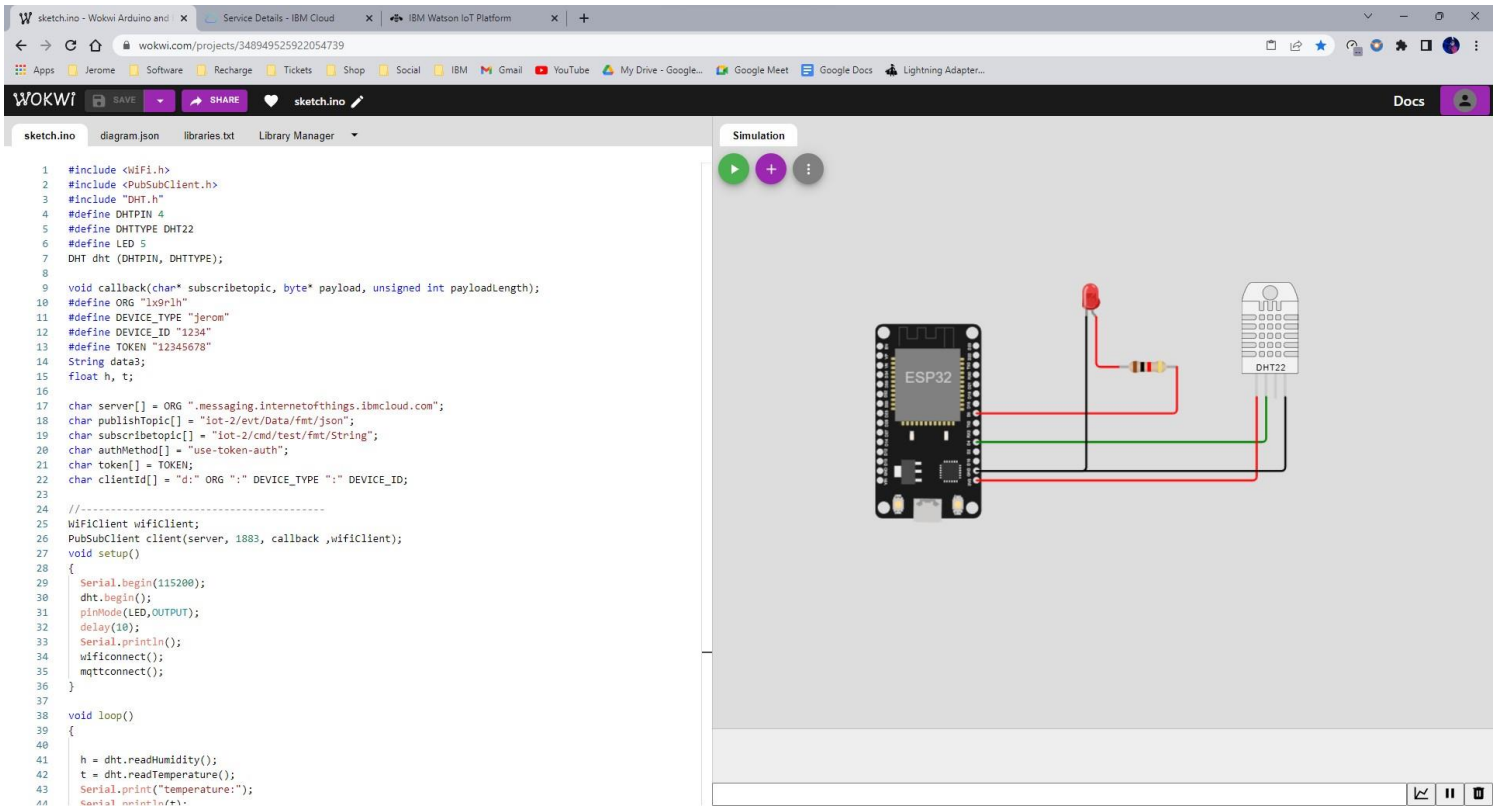
SPRINT GOAL:

Integrate the hardware to be able to access the cloud functions and provide inputs to the same.

POGRAM 01:

AIM: To find the Temperature and Humidity DHT22 and ESP32

PLATFORM: WOKWI



Service Details - IBM Cloud | IBM Watson IoT Platform | sketch.ino - Wokwi Arduino and

wokwi.com/projects/348949525922054739

WOKWI SAVE SHARE sketch.ino Docs

sketch.ino | diagram.json | libraries.txt | Library Manager

```
#include <WiFi.h>
#include <PubSubClient.h>
#include "DHT.h"
#define DHTPIN 4
#define DHTTYPE DHT22
#define LED 5
DHT dht (DHTPIN, DHTTYPE);

void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
#define ORG "lx9rlh"
#define DEVICE_TYPE "jerom"
#define DEVICE_ID "1234"
#define TOKEN "12345678"
String data3;
float h, t;

char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/Data/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);
void setup()
{
  Serial.begin(115200);
  dht.begin();
  pinMode(LED,OUTPUT);
  delay(10);
  Serial.println();
  wificonnect();
  mqttconnect();
}

void loop()
{

  h = dht.readHumidity();
  t = dht.readTemperature();
  Serial.print("temperature:");
```

Simulation

00:37.601 100%

Editing DHT22
Temperature: 30.0°C
Humidity: 44.0%

ESP32 DHT22

```
temperature:30.00
Humidity:44.00
Sending payload: {"temperature":30.00,"humidity":44.00}
Publish ok
temperature:30.00
Humidity:44.00
Sending payload: {"temperature":30.00,"humidity":44.00}
Publish ok
temperature:30.00
Humidity:44.00
Sending payload: {"temperature":30.00,"humidity":44.00}
Publish ok
temperature:30.00
Humidity:44.00
Sending payload: {"temperature":30.00,"humidity":44.00}
Publish ok
temperature:30.00
Humidity:44.00
Sending payload: {"temperature":30.00,"humidity":44.00}
Publish ok
```

Service Details - IBM Cloud | IBM Watson IoT Platform

lx9rlh.internetofthings.ibmcloud.com/dashboard/devices/browse

IBM Watson IoT Platform

912019104013@smartinternz.com
ID: lx9rlh

Browse | Action | Device Types | Interfaces

Add Device

Search by Device ID

Device Simulator

Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location	Added By	Device Class
1234	Connected	jerom	Device	Nov 22, 2022 6:43 PM		912019104013@smartinternz.com	

Identity | Device Information | Recent Events | State | Logs

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

Event	Value	Format	Last Received
Data	{"temperature":6.4,"humidity":27.5}	json	a few seconds ago
Data	{"temperature":6.4,"humidity":27.5}	json	a few seconds ago
Data	{"temperature":6.4,"humidity":27.5}	json	a few seconds ago
Data	{"temperature":6.4,"humidity":27.5}	json	a few seconds ago
Data	{"temperature":30,"humidity":28.5}	json	a few seconds ago

Items per page 50 | 1–1 of 1 item

1 of 1 page

0 Simulations running

PYTHON CODE:

```
#include <WiFi.h>
#include <PubSubClient.h>
#include "DHT.h"
#define DHTPIN 4
#define DHTTYPE DHT22
#define LED 5
DHT dht (DHTPIN, DHTTYPE);

void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
#define ORG "lx9rlh"
#define DEVICE_TYPE "jerom"
#define DEVICE_ID "1234"
#define TOKEN "12345678"
String data3;
float h, t;

char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/Data/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);
void setup()
{
  Serial.begin(115200);
  dht.begin();
  pinMode(LED,OUTPUT);
  delay(10);
  Serial.println();
  wificonnect();
  mqttconnect();
}

void loop()
{

  h = dht.readHumidity();
  t = dht.readTemperature();
  Serial.print("temperature:");
  Serial.println(t);
  Serial.print("Humidity:");
  Serial.println(h);

  PublishData(t, h);
  delay(1000);
```

```
  if (!client.loop()) {
    mqttconnect();
  }
}


void PublishData(float temp, float humid) {
  mqttconnect();

  String payload = "{\"temperature\":";
  payload += temp;
  payload += "," "\"humidity\":";
  payload += humid;
  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 client 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="";


}
```

Output link:

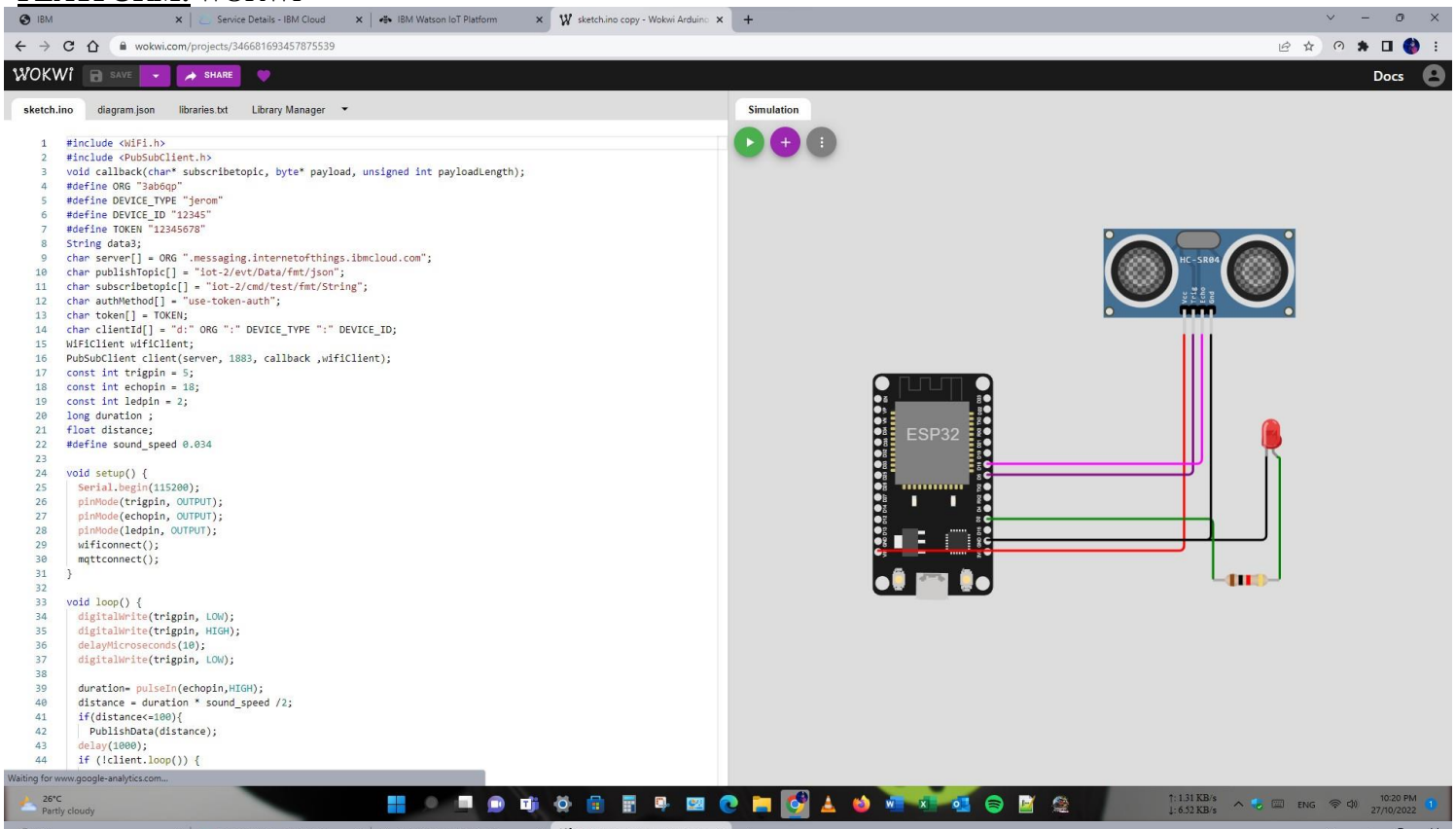
https://wokwi.com/projects/348949525922054739

By using this Wokwi we determined the Temperature and Humidity for better road safety.

POGRAM 02

AIM: Write code and connection in Wowki for ultrasonic sensor. Whenever distance is less than 100 cms send “Alert” to IBM cloud and display in device recent events by using ESP32

PLATFORM: WOKWI

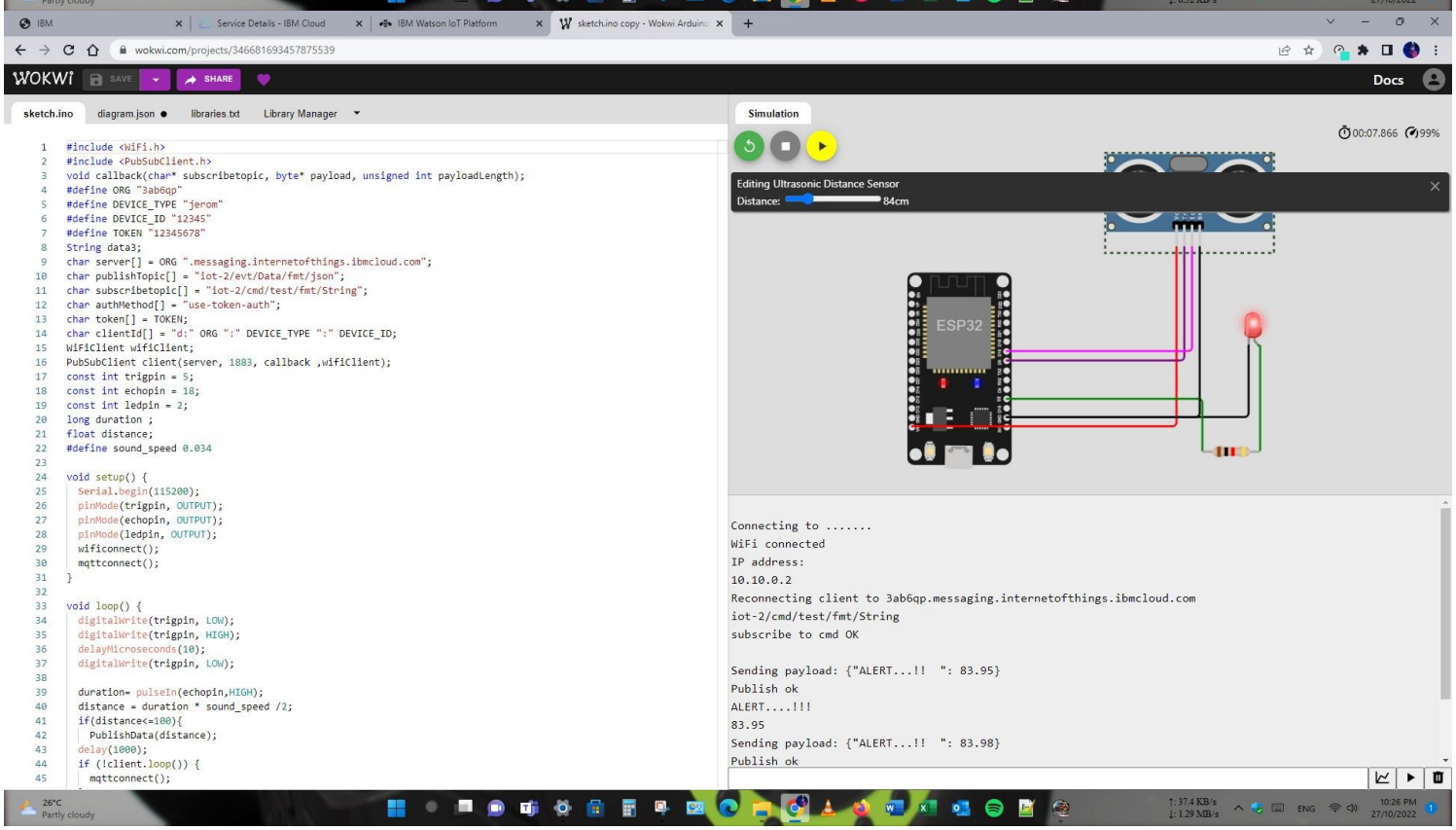


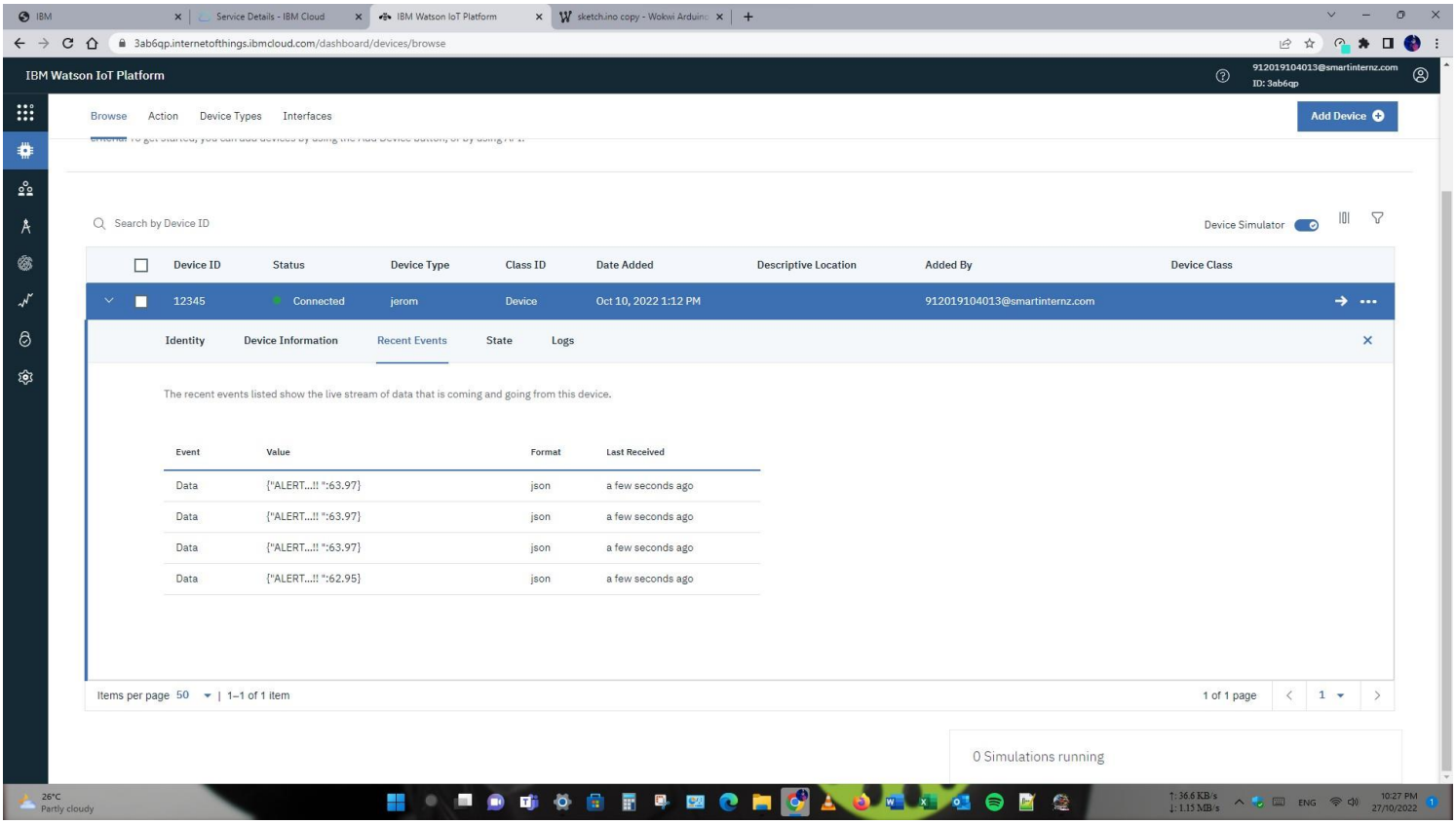

```
#include <WiFi.h>
#include <PubSubClient.h>
void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
#define ORG "3ab6qp"
#define DEVICE_TYPE "jerom"
#define DEVICE_ID "12345"
#define TOKEN "12345678"
String data3;
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/Data/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);
const int trigpin = 5;
const int echopin = 18;
const int ledpin = 2;
long duration ;
float distance;
#define sound_speed 0.034

void setup() {
  Serial.begin(115200);
  pinMode(trigpin, OUTPUT);
  pinMode(echopin, OUTPUT);
  pinMode(ledpin, OUTPUT);
  wificonnect();
  mqttconnect();
}

void loop() {
  digitalWrite(trigpin, LOW);
  digitalWrite(trigpin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigpin, LOW);

  duration= pulseIn(echopin,HIGH);
  distance = duration * sound_speed /2;
  if(distance<=100){
    PublishData(distance);
  delay(1000);
  if (!client.loop()) {
```





PYHTON CODE:

```
#include <WiFi.h>
#include <PubSubClient.h>
void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
#define ORG "lx9rlh"
#define DEVICE_TYPE "jerom"
#define DEVICE_ID "12345"
#define TOKEN "12345678"
String data3;
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/Data/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);
const int trigpin = 5;
const int echopin = 18;
const int ledpin = 2;
long duration ;
float distance;
#define sound_speed 0.034

void setup() {
  Serial.begin(115200);
  pinMode(trigpin, OUTPUT);
```

```
  pinMode(echopin, OUTPUT);
  pinMode(ledpin, OUTPUT);
  wificonnect();
  mqttconnect();
}

void loop() {
  digitalWrite(trigpin, LOW);
  digitalWrite(trigpin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigpin, LOW);

  duration= pulseIn(echopin,HIGH);
  distance = duration * sound_speed /2;
  if(distance<=100){
    PublishData(distance);
  delay(1000);
  if (!client.loop()) {
    mqttconnect();
  }
    digitalWrite(ledpin, HIGH);
    Serial.println("ALERT....!!!");
    Serial.println(distance);
  }
  else
  {
    digitalWrite(ledpin, LOW);
  }
  delay(10); // this speeds up the simulation
}

void PublishData(float distance) {
  mqttconnect();

  String payload = "{\"ALERT...!!  \": ";
  payload += distance;
  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 client 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++) {
    //Serial.print((char)payload[i]);
    data3 += (char)payload[i];
  }

  Serial.println("data: "+ data3);
  if(data3=="lighton")
  {
      Serial.println(data3);
  }
  else
  {
```

```
      Serial.println(data3);
  }
data3="";
}
```

Output link :

https://wokwi.com/projects/349037081204359762

Conclusion:

Here I showed the ALERT and DISTANCE in IBM cloud when vehicle has the distance is less than 100 cms