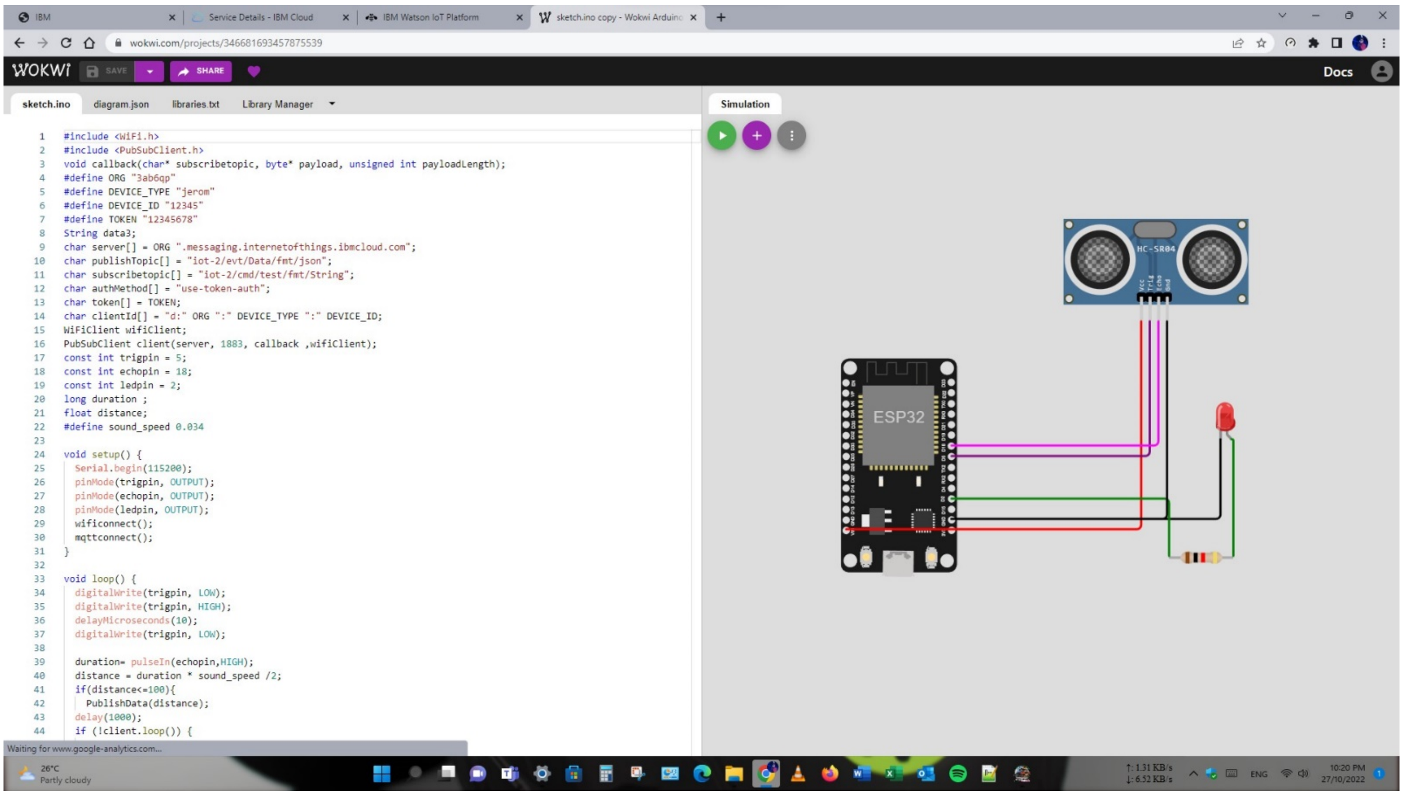
ASSIGNMENT – 04

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.

Wowki Project Link: https://wokwi.com/projects/346681693457875539

Step 1: Completed Setup to build Circuit

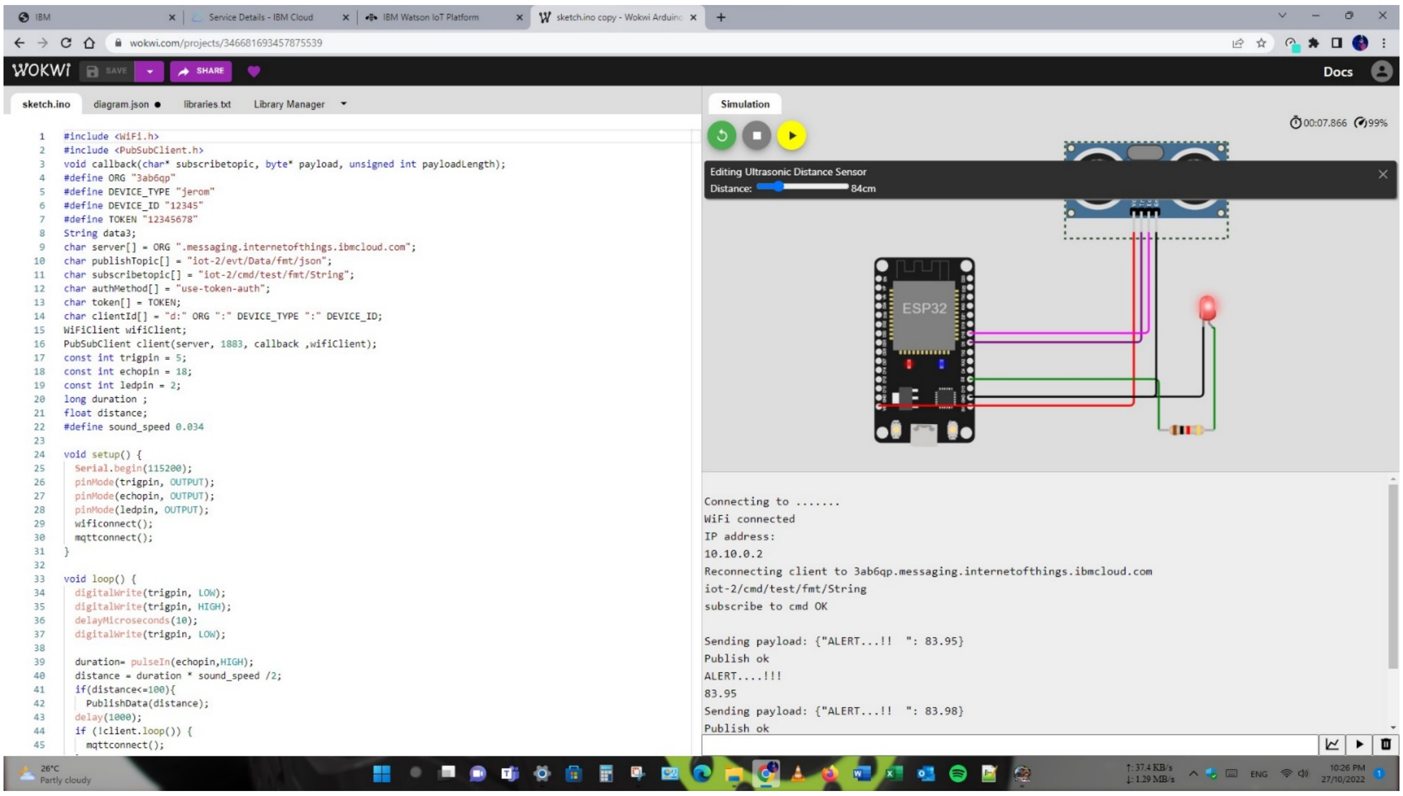


```
#include <WiFi.h>
#include <PubSubClient.h>
void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
#define ORG "3ab6qp"
#define DEVICE_TYPE "jerom"
#define DEVICE_ID "12345"
#define TOKEN "12345678"
String data3;
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
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()) {
```

Waiting for www.google-analytics.com...

Step 2: Output in WOWKI

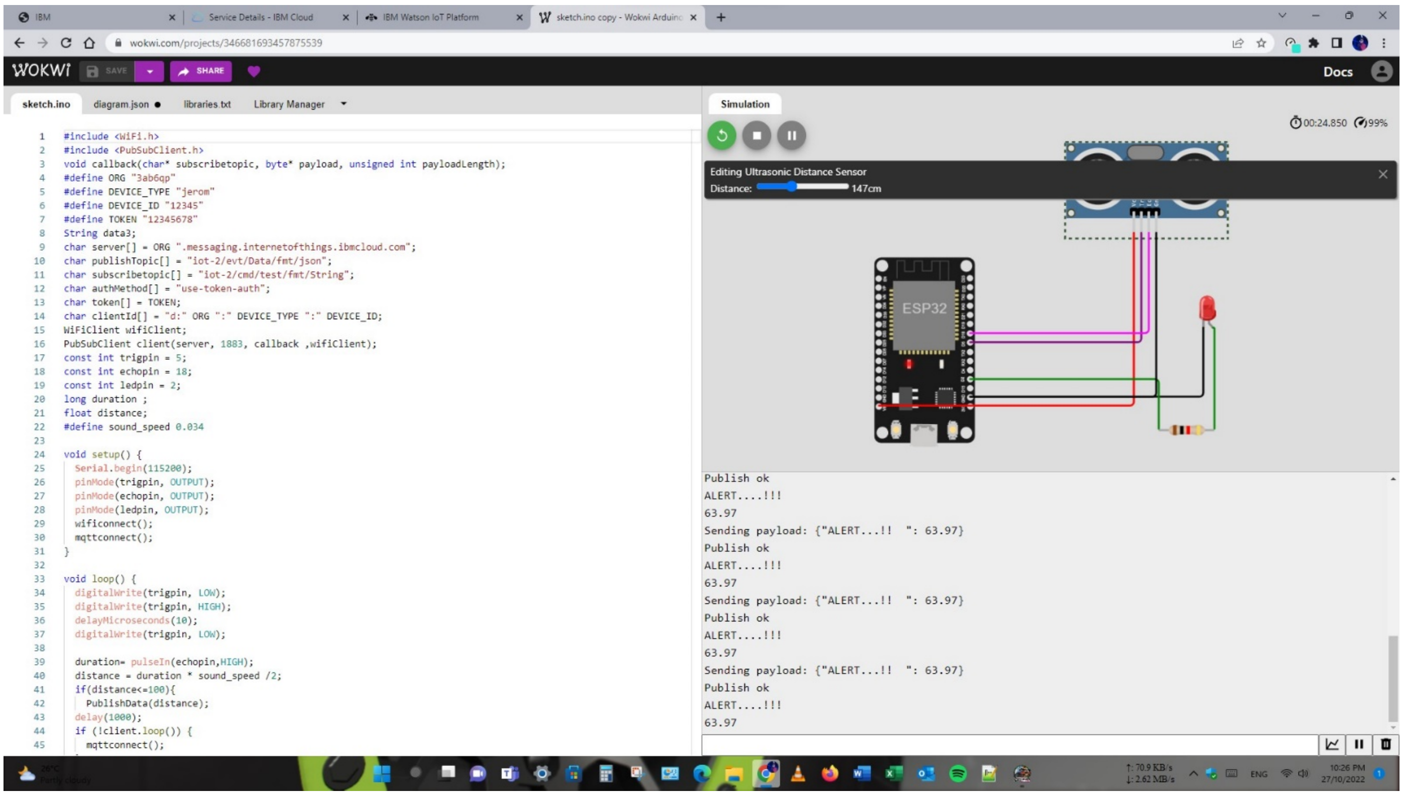
a) When distance is below 100 cms



```
#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()) {
    mqttconnect();
```

```
Connecting to .......
WiFi connected
IP address:
10.10.0.2
Reconnecting client to 3ab6qp.messaging.internetofthings.ibmcloud.com
iot-2/cmd/test/fmt/String
subscribe to cmd OK

Sending payload: {"ALERT...!!  ": 83.95}
Publish ok
ALERT....!!!
83.95
Sending payload: {"ALERT...!!  ": 83.98}
Publish ok
```

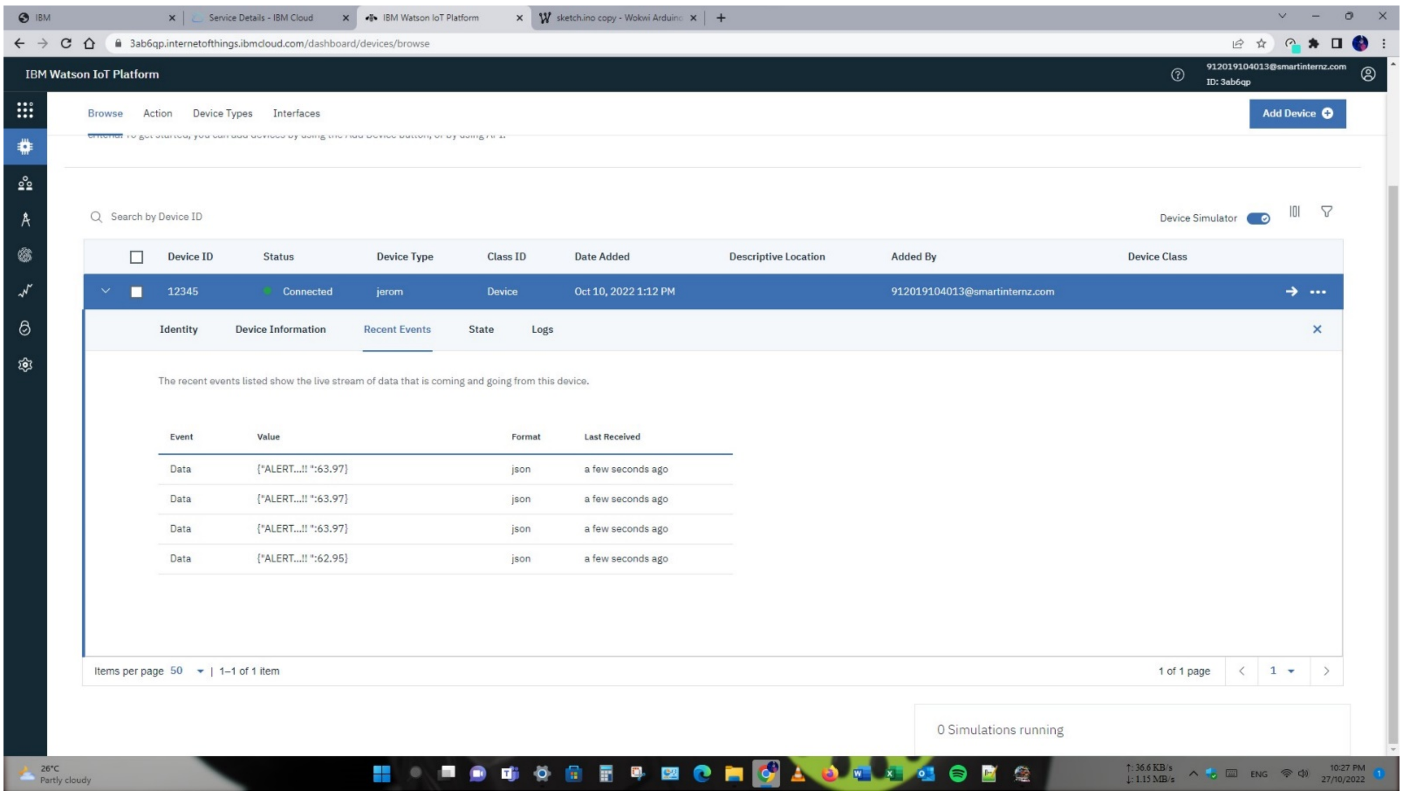
b) When Distance is above 100 cms



```
#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()) {
    mqttconnect();
```

```
Publish ok
ALERT....!!!
63.97
Sending payload: {"ALERT...!!  ": 63.97}
Publish ok
ALERT....!!!
63.97
Sending payload: {"ALERT...!!  ": 63.97}
Publish ok
ALERT....!!!
63.97
Sending payload: {"ALERT...!!  ": 63.97}
Publish ok
ALERT....!!!
63.97
```

Step 3: Output in IBM CLOUD (Watson Platform) recent events



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

Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location	Added By	Device Class
12345	Connected	jerom	Device	Oct 10, 2022 1:12 PM		912019104013@smartinternz.com	

Event	Value	Format	Last Received
Data	{"ALERT...!! ":63.97}	json	a few seconds ago
Data	{"ALERT...!! ":63.97}	json	a few seconds ago
Data	{"ALERT...!! ":63.97}	json	a few seconds ago
Data	{"ALERT...!! ":62.95}	json	a few seconds ago

Items per page 50 | 1–1 of 1 item   1 of 1 page

0 Simulations running

Program Code:

```
#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()) {
    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="";
}
```