IBM ASSIGNMENT 4

NAME : NISHANTH M (2019504041)

**CODE :**

#include <WiFi.h>

#include <PubSubClient.h>

void callback(char\* subscribetopic,byte\* payload, unsigned int payloadLength);

#define ORG "i8xseg"

#define DEVICE\_TYPE "ESP"

#define DEVICE\_ID "2679"

#define TOKEN "9688811163"

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 14

#define TRIG\_PIN 12

#define led 27

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);

  delayMicroseconds(2);

  digitalWrite(TRIG\_PIN, HIGH);

  delayMicroseconds(10);

  digitalWrite(TRIG\_PIN, LOW);

  int duration=random(1,200);

  //Serial.println(duration);

  //duration = pulseIn(ECHO\_PIN, HIGH);

  return duration ;

  //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);

  }

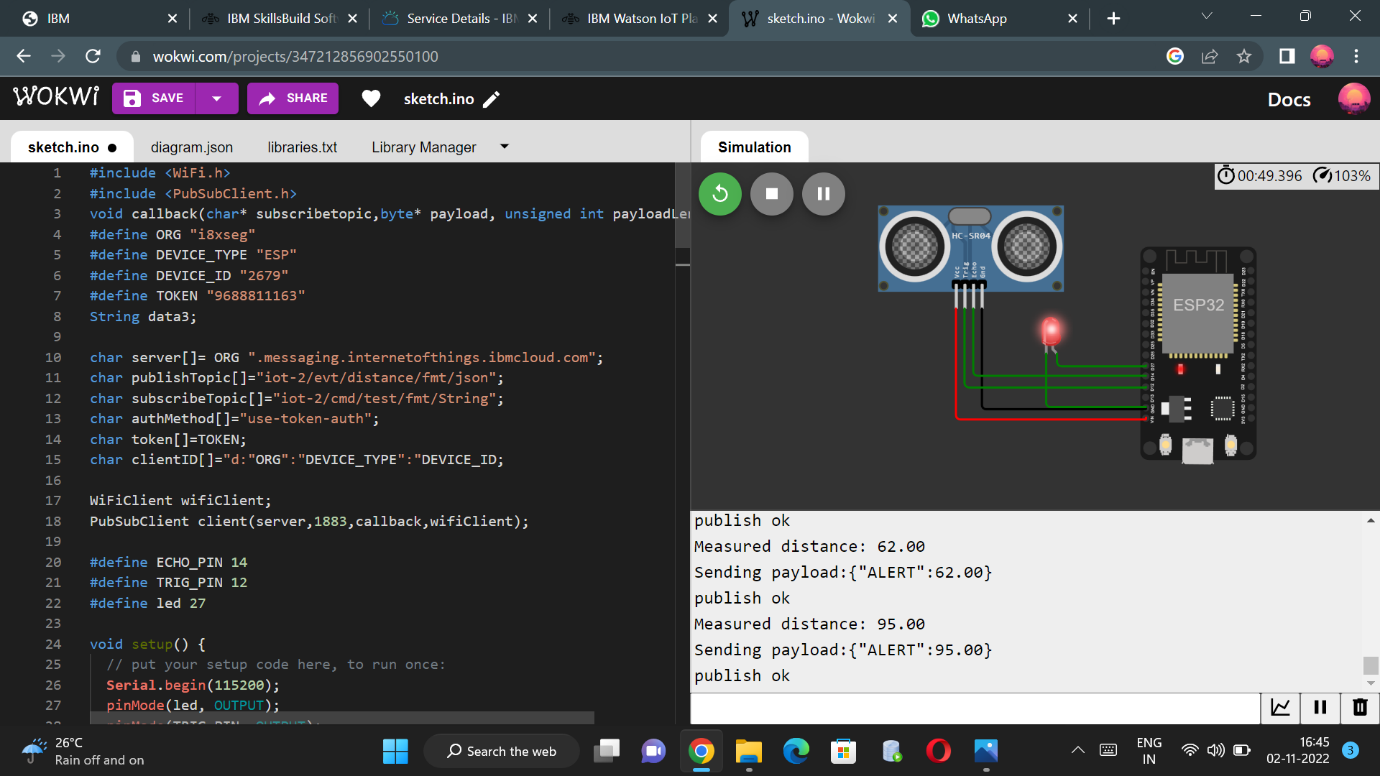
  data3="";

}

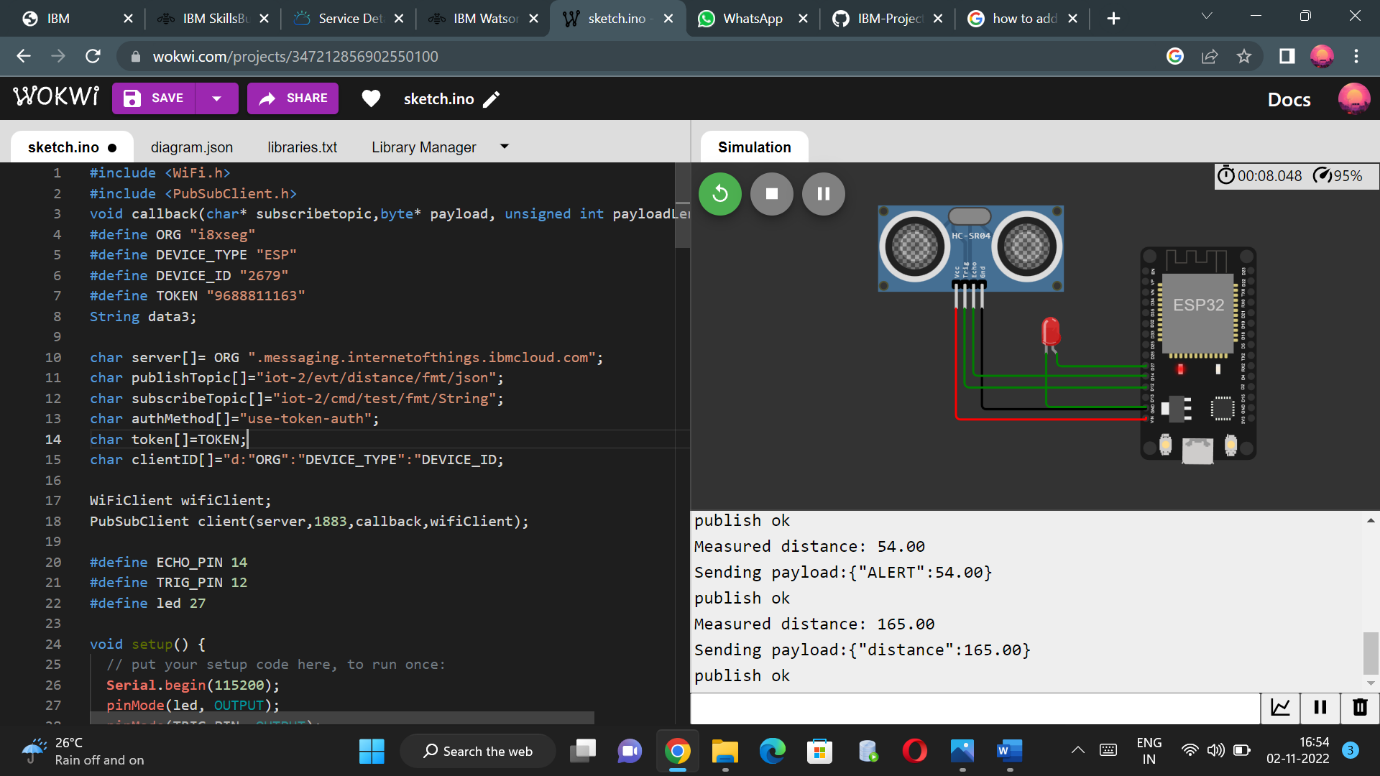
**WOWKI LINK :**

[**https://wokwi.com/projects/347212856902550100**](https://wokwi.com/projects/347212856902550100)

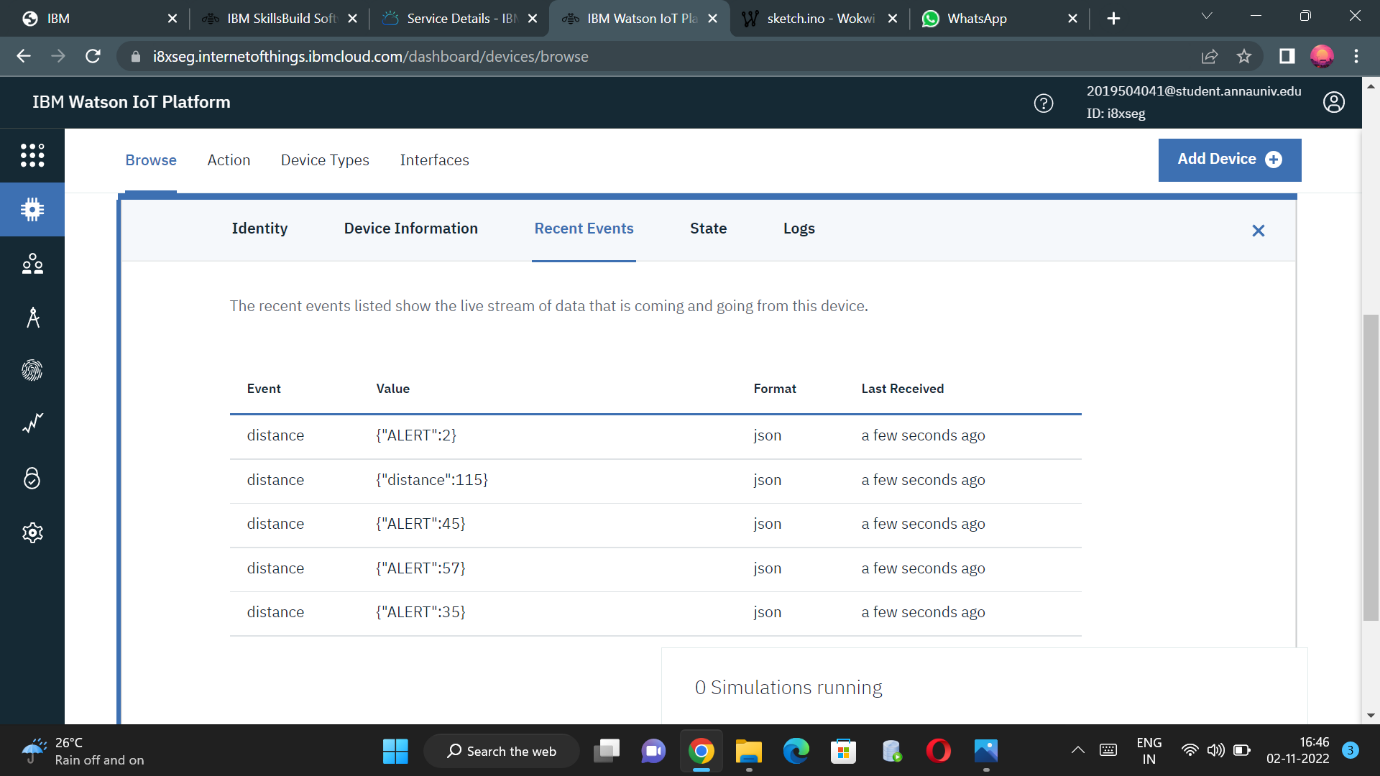
**Alert Case:**



**Normal Case :**

****

**Cloud Storage:**

****