ASSIGNMENT 4

**Date** - 31 October 2022

**Team ID** - PNT2022TMID42331

**Project Name - Industry-specific intelligent fire management system**

**Team Members:**

1. Vinupriya K P- Team Leader

2. Jeevitha K -Team Member

3. Anne Shifana S R- Team Member

4. Vishnupriya E – Team Member

5. Monisha R - Team Member

**QUESTION:** Write code and connections in wokwi for ultrasonic sensor. Whenever distance is less than 100 cms send "alert" to ibm cloud and display in device recent events.

**CODE:**

#include <WiFi.h>

#include <PubSubClient.h>

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

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

#define ORG "Ashfaq1824"//IBM ORGANITION ID

#define DEVICE\_TYPE "ESP32"//Device type mentioned in ibm watson IOT Platform

#define DEVICE\_ID "12345"//Device ID mentioned in ibm watson IOT Platform

#define TOKEN "12345678" //Token

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;

#define SOUND\_SPEED 0.034

 long duration;

 float distance;

 void setup() {

**Serial**.begin(115200);

pinMode(trigPin, OUTPUT);

pinMode(echoPin, INPUT);

wificonnect();

 mqttconnect();

} void loop() {

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

duration = pulseIn(echoPin, HIGH);

 distance = duration \* SOUND\_SPEED/2;

**Serial**.print("Distance (cm): ");

**Serial**.println(distance);

if(distance<100)

{

**Serial**.println("ALERT!!");

 delay(1000);

PublishData(distance);

delay(1000);

if(!client.loop()) {

mqttconnect();

} }

delay(1000); }

void PublishData(float dist) {

mqttconnect();

String payload = "{\"Distance\":";

payload += dist;

payload += ",\"ALERT!!\":""\"Distance less than 100cms\"";

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];

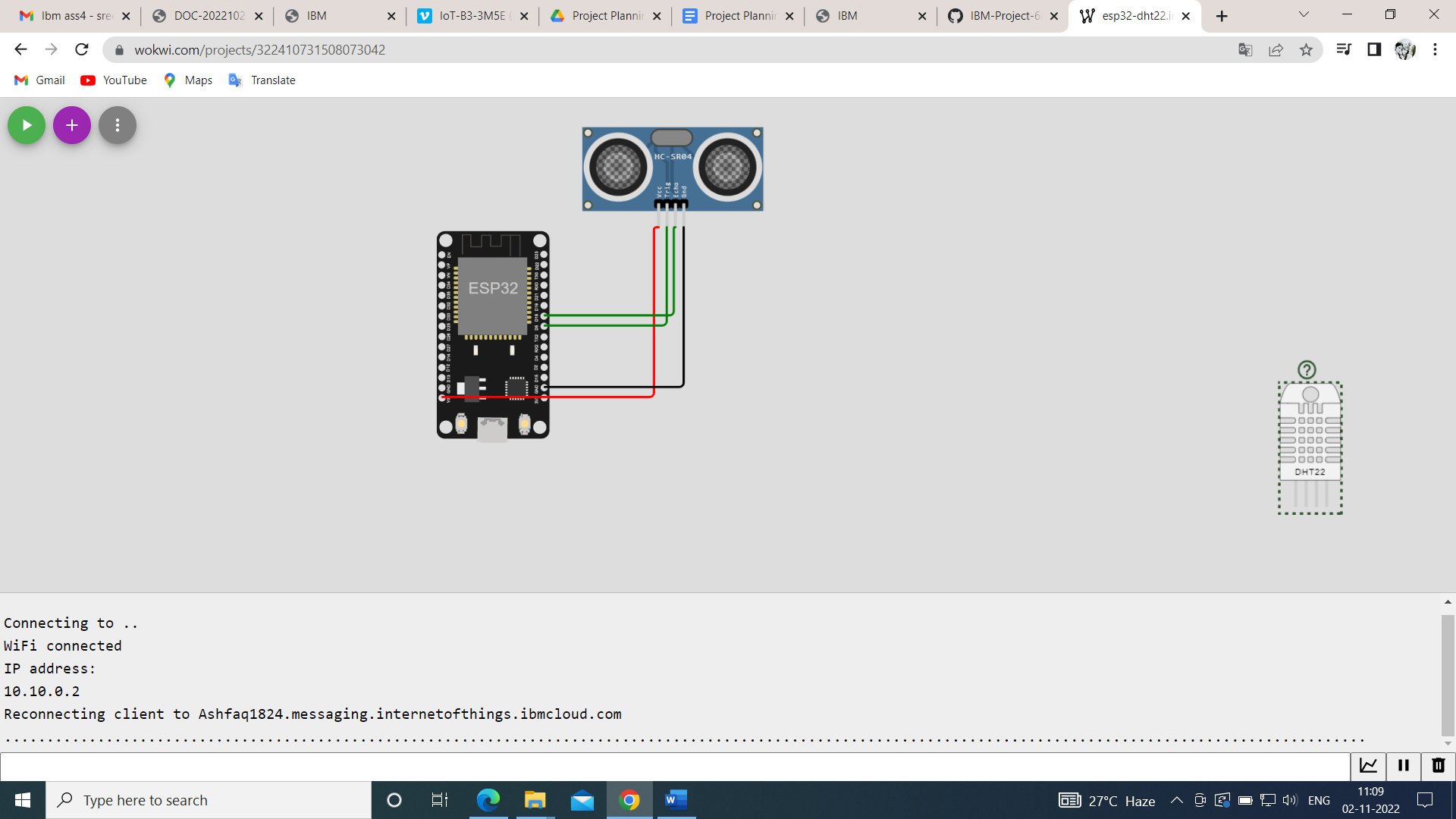
}

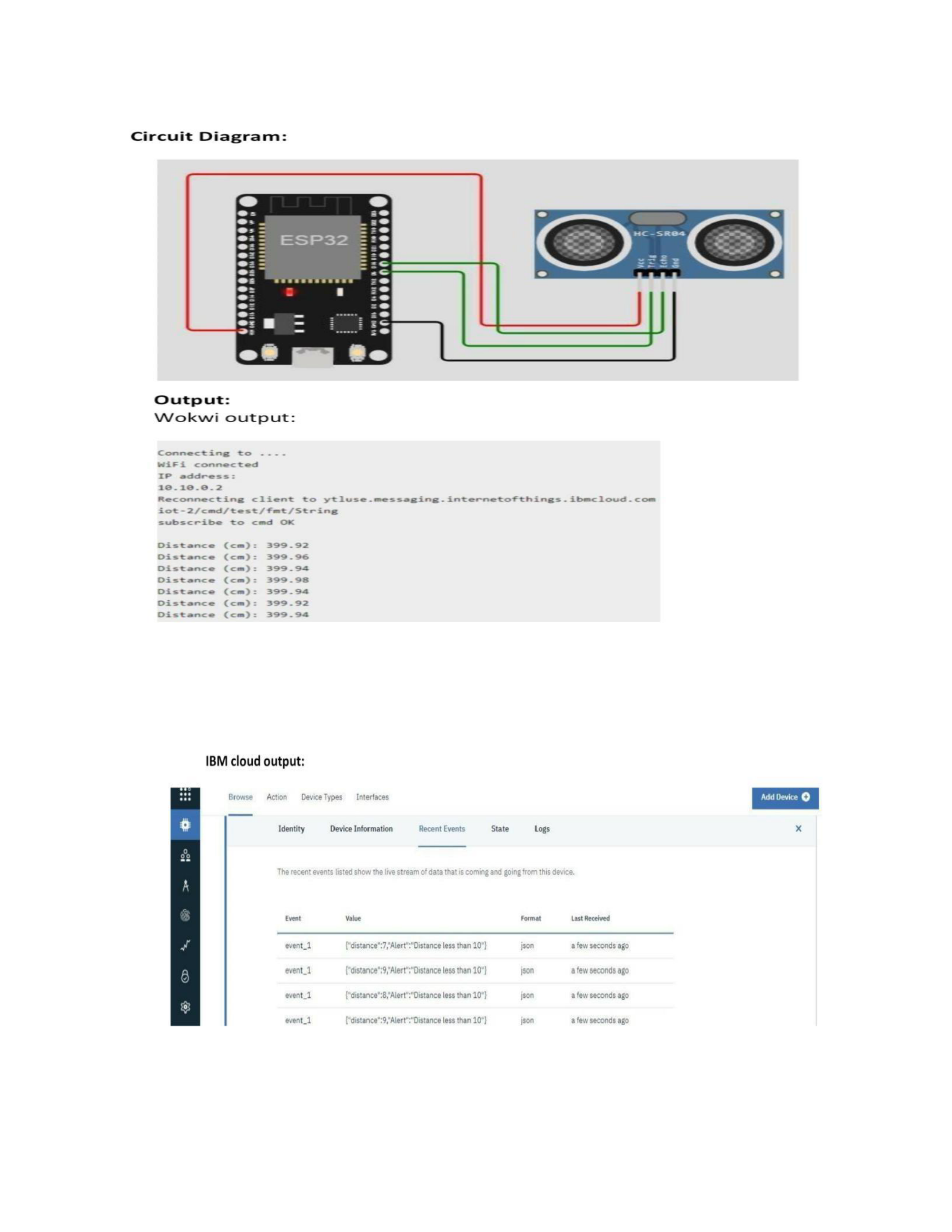
**Serial**.println("data: "+ data3);

data3="";

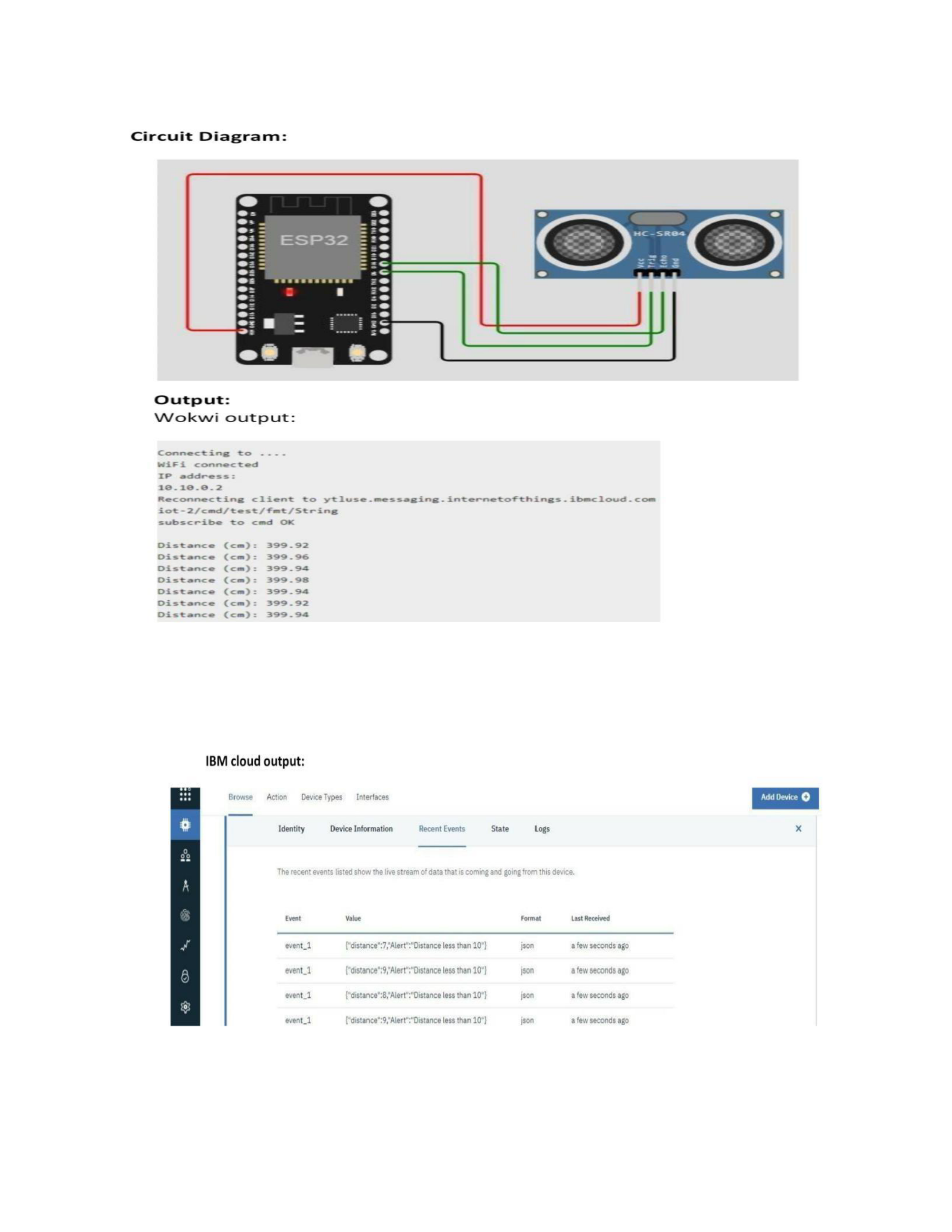
}

**SCHEMATIC/CIRCUIT DIAGRAM:**



****

**IBM CLOUD OUTPUT:**

****

**WOKWI LINK:**

https://wokwi.com/projects/322410731508073042