Assignment - 4

Assignment Date	01 November 2022
Student Name	Shriya R
Student Roll Number	953619104037
Team ID	PNT2022TMID51029
Maximum Marks	2 Marks

Question:

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. Upload document with wokwi share link and images of IBM cloud.

Solution:

Sketch.ino

```
#include <WiFi.h>
#include <PubSubClient.h>
#include <ArduinoJson.h>
WiFiClient wifiClient;
#define ORG "mz6rat"
#define DEVICE_TYPE "arduino"
#define DEVICE_ID "54321"
#define TOKEN "26072002"
#define speed 0.034
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/Data/fmt/json";
char topic[] = "iot-2/cmd/home/fmt/String";
char authMethod[] = "use-token-auth";
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
PubSubClient client(server, 1883, wifiClient);
void publishData();
const int trigpin=5;
const int echopin=18;
String command;
String data="";
long duration;
int dist;
void setup()
{
 Serial.begin(115200);
 pinMode(trigpin, OUTPUT);
 pinMode(echopin, INPUT);
 wifiConnect();
```

```
 mqttConnect();
}
void loop() {
 publishData();
 delay(500);
 if (!client.loop()) {
 mqttConnect();
 }
}
void wifiConnect() {
 Serial.print("Connecting to "); Serial.print("Wifi");
 WiFi.begin("Wokwi-GUEST", "", 6);
 while (WiFi.status() != WL_CONNECTED) {
 delay(500);
 Serial.print(".");
 }
 Serial.print("WiFi connected, IP address: "); Serial.println(WiFi.localIP());
}
void mqttConnect() {
 if (!client.connected()) {
 Serial.print("Reconnecting MQTT client to "); Serial.println(server);
 while (!client.connect(clientId, authMethod, token)) {
 Serial.print(".");
 delay(1000);
 }
 initManagedDevice();
 Serial.println();
 }
}
void initManagedDevice() {
 if (client.subscribe(topic)) {
 Serial.println(client.subscribe(topic));
 Serial.println("subscribe to cmd OK");
 } else {
 Serial.println("subscribe to cmd FAILED");
 }
}
void publishData()
{
 digitalWrite(trigpin,LOW);
 digitalWrite(trigpin,HIGH);
 delayMicroseconds(10);
 digitalWrite(trigpin,LOW);
 duration=pulseIn(echopin,HIGH);
 dist=duration*speed/2;

 if(dist<100){
 DynamicJsonDocument doc(1024);
```

```
 String payload;
 doc["AlertDistance:"]=dist;
 serializeJson(doc, payload);
 delay(3000);
 Serial.print("\n");
 Serial.print("Sending payload: ");
 Serial.println(payload);
 if (client.publish(publishTopic, (char*) payload.c_str())) {
 Serial.println("Publish OK");
 } else {
 Serial.println("Publish FAILED");
 }
 }
}
```

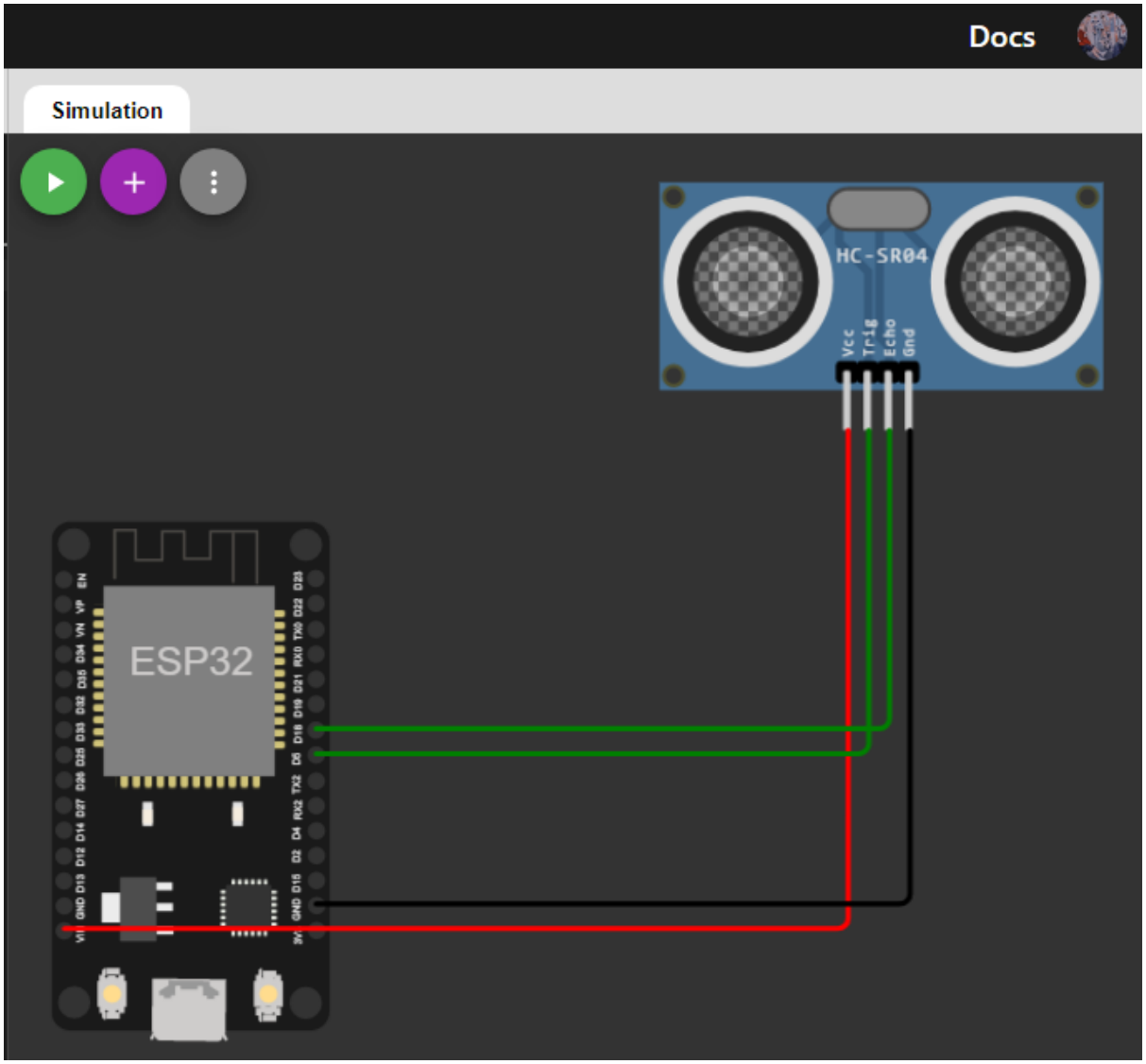
Diagram.json

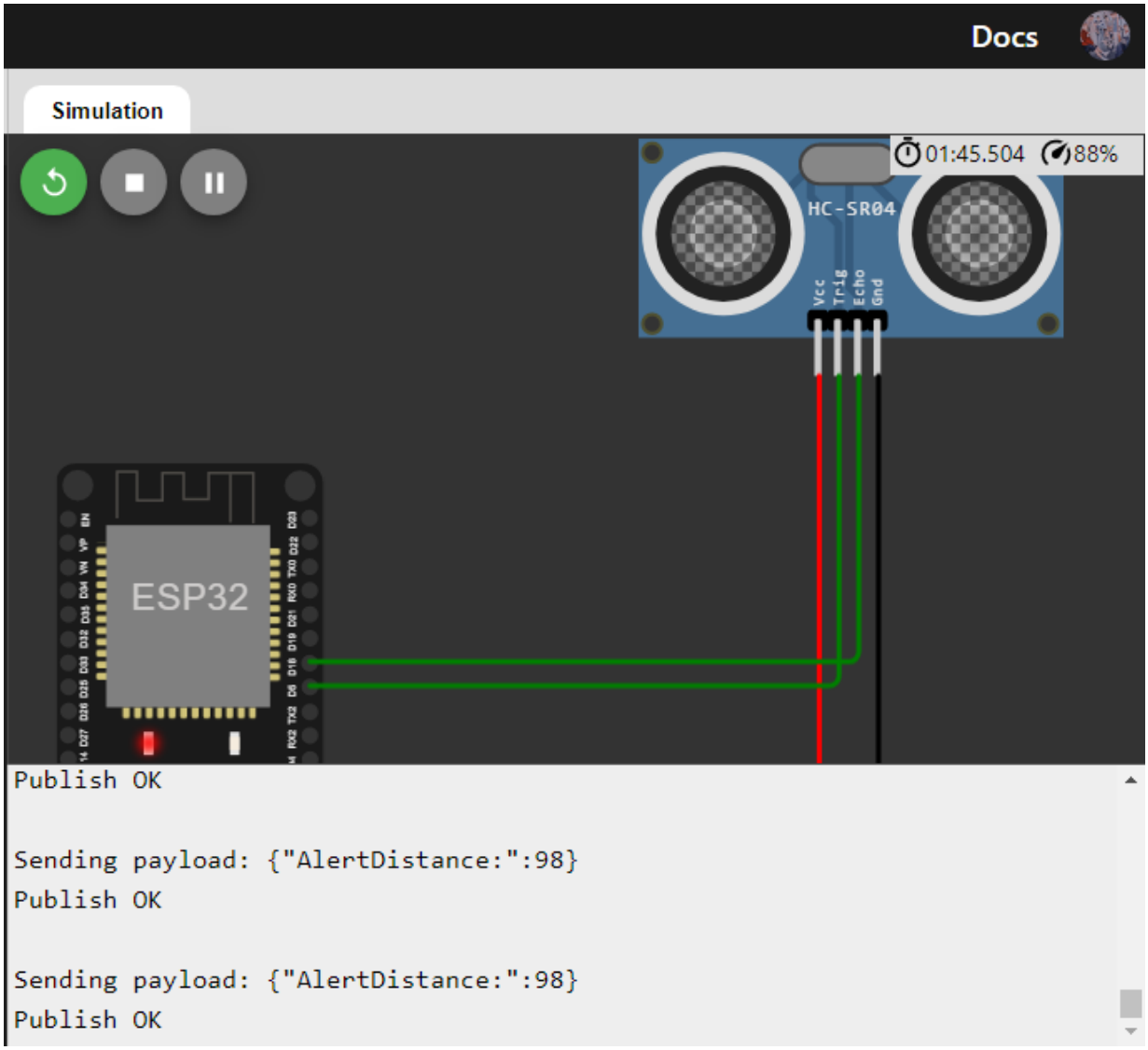
```
{
  "version": 1,
  "author": "Shriya R",
  "editor": "wokwi",
   "parts": [
    { "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": 40.66, "left": -
70.66, "attrs": {} },
    {
      "type": "wokwi-hc-sr04",
      "id": "ultrasonic1",
      "top": -86.81,
      "left": 162.37,
      "attrs": { "distance": "99" }
    }
  ],
  "connections": [
    [ "esp:TX0", "$serialMonitor:RX", "", [] ],
    [ "esp:RX0", "$serialMonitor:TX", "", [] ],
    [ "ultrasonic1:VCC", "esp:VIN", "red", [ "v0" ] ],
    [ "ultrasonic1:GND", "esp:GND.1", "black", [ "v0" ] ],
    [ "esp:D5", "ultrasonic1:TRIG", "green", [ "h0" ] ],
    [ "esp:D18", "ultrasonic1:ECHO", "green", [ "h0" ] ]
  ]
}
```

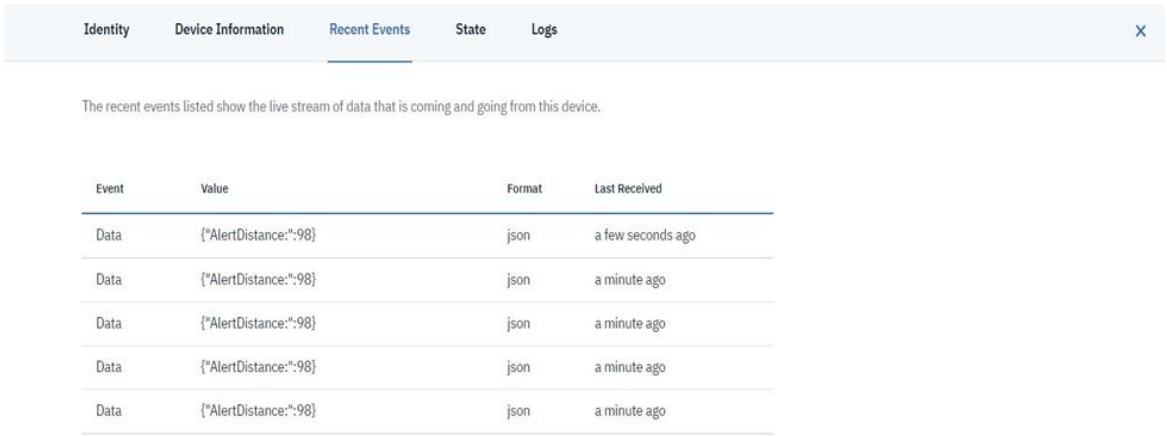
Libraries.txt

 # Wokwi Library List
 # See https://docs.wokwi.com/guides/libraries

ArduinoJson
PubSubClient







Wokwi Simulation link:

https://wokwi.com/projects/347138334045241938