

ASSIGNMENT 04

WOKWI STIMULATOR

| | |
|---------------------|--|
| Student Name | SUVITHAPRIYA |
| Student Roll Number | 713319EC113 |
| Team ID | PNT2022TMID17751 |
| Project Name | INDUSTRY-SPECIFIC INTELLIGENT FIRE MANAGEMENT SYSTEM |

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

CODE:

```
#include <WiFi.h>
```

```
#include <PubSubClient.h>
```

```
#include <ArduinoJson.h>
```

```
WiFiClient wifiClient;
```

```
#define ORG "wt19pm"
```

```
#define DEVICE_TYPE "NodeMCU"
```

```
#define DEVICE_ID "12345"
```

```
#define TOKEN "12345678"
```

```
#define speed 0.034
```

```
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
```

```
char publishTopic[] = "iot-2/evt/status1/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="";
```

```
String name="Alert";
```

```
String icon="";
```

```
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(".");  
    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()  
{
```

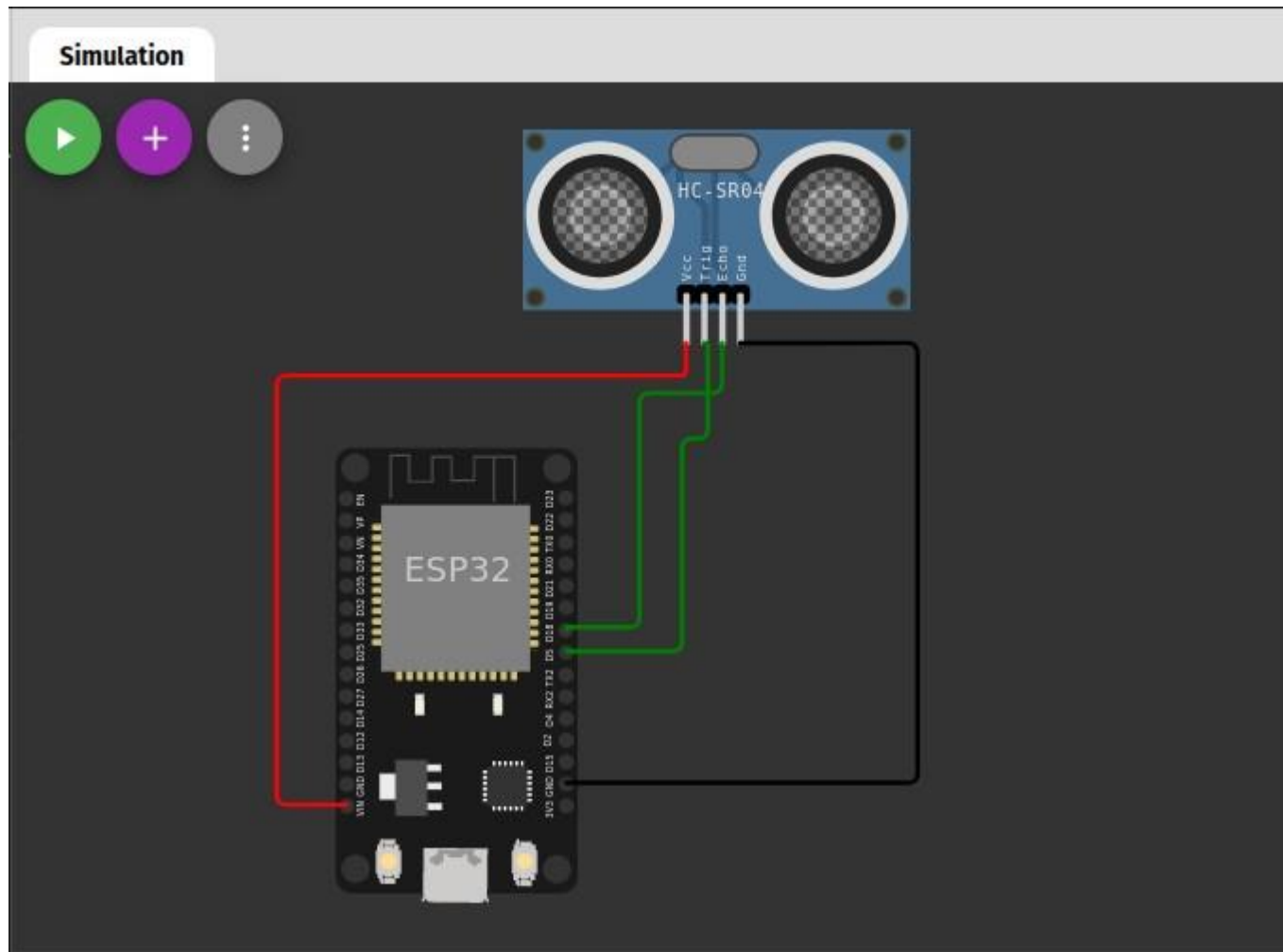
```
digitalWrite(trigpin,LOW);  
digitalWrite(trigpin,HIGH);  
delayMicroseconds(10);  
digitalWrite(trigpin,LOW);  
duration=pulseIn(echopin,HIGH);  
dist=duration*speed/2;
```

```
if(dist<100){  
    dist=100-dist;  
    icon="no-trash";  
}else{  
    dist=0;  
    icon="trash";  
}
```

```
DynamicJsonDocument doc(1024);  
String payload;  
doc["Name"]=name;  
doc["Icon"]=icon;  
doc["FillPercent"]=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");  
}  
}
```

CONNECTIONS:



WOKWI LINK:

<https://wokwi.com/projects/346647891630096979>

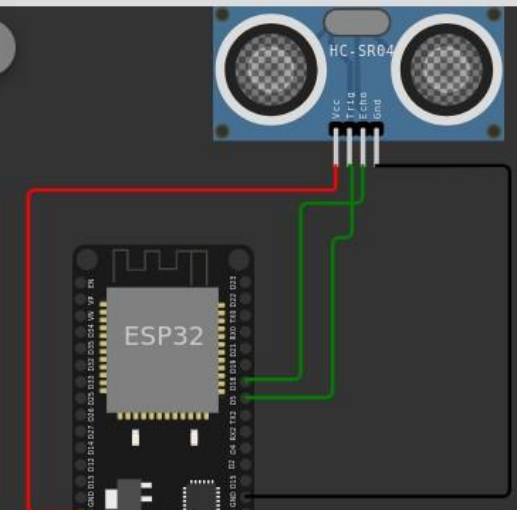
OUTPUT :

WOKWI SAVE SHARE Docs

esp32-blink.ino • diagram.json • libraries.txt • Library Manager

```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 #include <ArduinoJson.h>
4
5 WiFiClient wifiClient;
6
7 #define ORG "wt19pm"
8 #define DEVICE_TYPE "NodeMCU"
9 #define DEVICE_ID "12345"
10 #define TOKEN "12345678"
11 #define speed 0.034
12
13 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
14 char publishTopic[] = "iot-2/evt/status1/fmt/json";
15 char topic[] = "iot-2/cmd/home/fmt/String";
16 char authMethod[] = "use-token-auth";
17 char token[] = TOKEN;
18 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
19 PubSubClient client(server, 1883, wifiClient);
20 void publishData();
21
22 const int trigpin=5;
23 const int echopin=18;
24 String command;
25 String data="";
26 String name="Alert";
27 String icon="";
28
29 long duration;
30 int dist;
```

Simulation



Publish OK

Sending payload: {"Name":"Alert","Icon":"trash","FillPercent":0}
Publish OK

Sending payload: {"Name":"Alert","Icon":"trash","FillPercent":0}
Publish OK



Browse

Action

Device Types

Interfaces

Add Device +

| <input type="checkbox"/> | Device ID | Status | Device Type | Class ID | Date Added | |
|--|---|--------------|-------------------|----------|-----------------------|-------|
| ▼ <input type="checkbox"/> | 12345 | Disconnected | NodeMCU | Device | Oct 25, 2022 10:22 PM | → ... |
| 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 | | | | | | |
| status1 | {"Name": "Alert", "Icon": "trash", "FillPercent": 0} | json | a few seconds ago | | | |
| status1 | {"Name": "Alert", "Icon": "trash", "FillPercent": 0} | json | a few seconds ago | | | |
| status1 | {"Name": "Alert", "Icon": "trash", "FillPercent": 0} | json | a few seconds ago | | | |
| status1 | {"Name": "Alert", "Icon": "trash", "FillPercent": 0} | json | a few seconds ago | | | |
| status1 | {"Name": "Alert", "Icon": "fa-trash-o", "FillPercent": 0} | json | 2 minutes ago | | | |