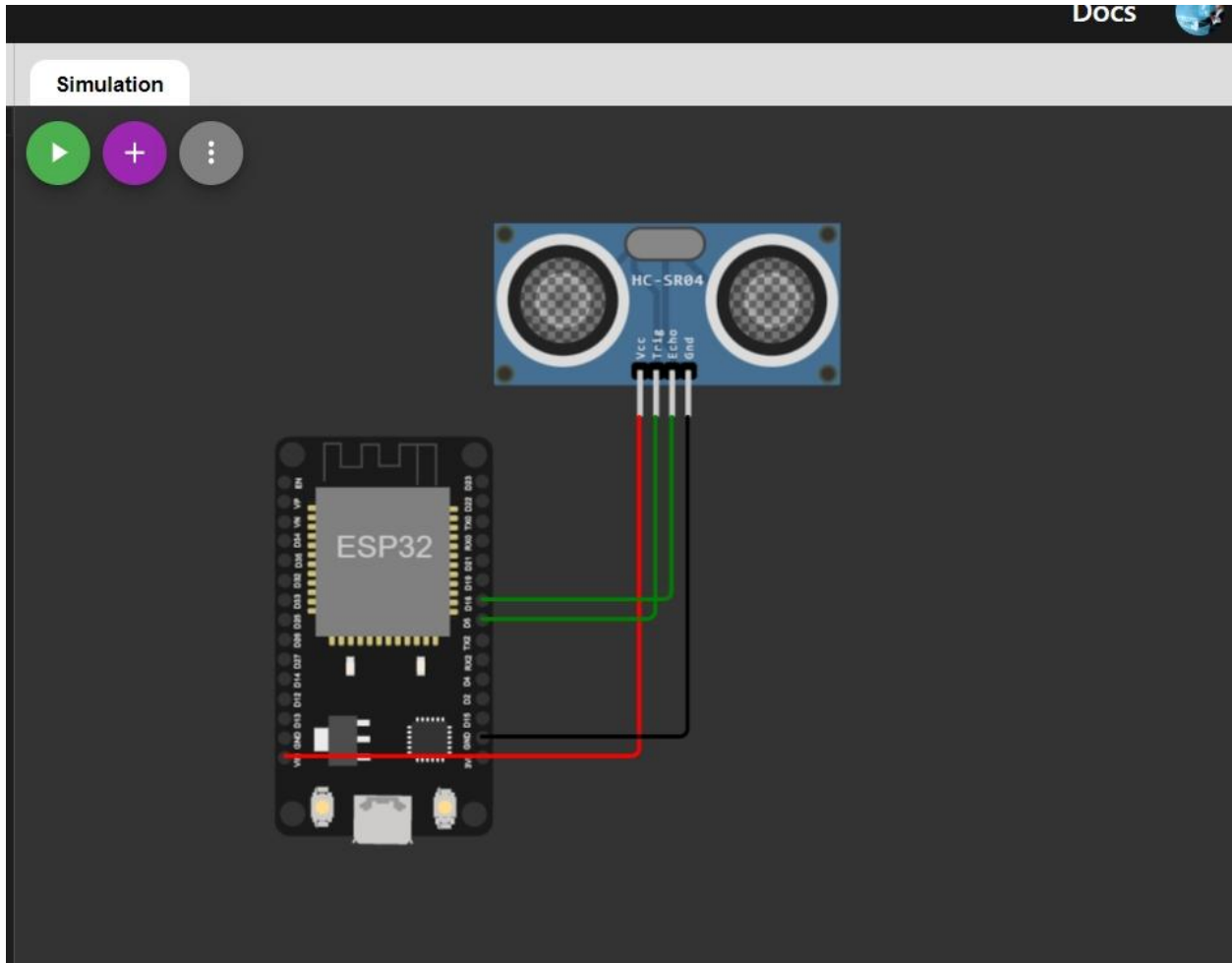


Assignment -4

Write code and connections in wowki for the ultrasonic sensor. Whenever the distance is less than 100 cms send an alert to ibm cloud and display in the device recent events,

Circuit Diagram:



Code:

```
#include <WiFi.h>
#include <PubSubClient.h>
WiFiClient wifiClient;
String data3;
#define ORG "f4cwb9"
#define DEVICE_TYPE "UltraSonicSensor"
#define DEVICE_ID "234567"
#define TOKEN "GxCVjUsU7smKp*RSrT"
#define speed 0.034
```

```

#define led 14
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/Albin B/fmt/json";
char topic[] = "iot-2/cmd/led/fmt/String";
char authMethod[] = "use-token-auth";
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
PubSubClient client(server, 1883, wifiClient);
const int trigpin=5;
const int echopin=18;
String command;
String data="";
long duration;
float dist;
void setup()
{
  Serial.begin(115200);
  pinMode(led, OUTPUT);
  pinMode(trigpin, OUTPUT);
  pinMode(echopin, INPUT);
  wifiConnect();
  mqttConnect();
}
void loop() {
  bool isNearby = dist < 100;
  digitalWrite(led, isNearby);
  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(500);
}
initManagedDevice();
Serial.println();
}
}

void initManagedDevice() {
if (client.subscribe(topic)) {
// Serial.println(client.subscribe(topic));
Serial.println("IBM 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){
String payload = "{\"Alert Distance\":\"";
payload += dist;
payload += "\"}";
Serial.print("\n");
Serial.print("Sending payload: ");
Serial.println(payload);
if (client.publish(publishTopic, (char*) payload.c_str())) {
Serial.println("Publish OK");
}
}
}

```

```

}
if(dist>100){
String payload = "{\"Distance\":\"";
payload += dist;
payload += "\"}";
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");
}
}
}
}

```

Wowki Stimulation Link:

<https://wokwi.com/projects/348427968162300500>

Output:

1) When the distance is greater than 100 cms

The screenshot displays a Wokwi simulation environment. On the left, the code for an ESP32 microcontroller is shown, with line numbers 59 to 92. The code includes a `publishData` function that triggers a publish event when the distance measured by the ultrasonic sensor is greater than 100cm. On the right, the simulation window shows an ESP32 board connected to an HC-SR04 ultrasonic sensor. A control bar for the sensor indicates a distance of 148cm. The output console at the bottom shows the following sequence of events:

```

Publish OK
Sending payload: {"Alert Distance":147.93}
Publish OK
Sending payload: {"Alert Distance":147.95}
Publish OK

```

When the distance is less than 100 cms:

