Assignment-4

Smart Solution For Railways

Team ID:PNT2022TMID52217

write code and connections in wowki for ultrasonic sensor.whenever distance is less than 100cm send “alert” to ibm cloud and display in device recent events.upload document with wowki share link and images of ibm cloud.

```
#include <WiFi.h>
#include <PubSubClient.h>
void callback(char* subscribetopic,byte* payload, unsigned intpayloadLength);#define ORG
"fdd82r"
#define DEVICE_TYPE "Pi"
#define DEVICE_ID "123"
#define TOKEN "12345678"
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;

WiFiClientwifiClient;
PubSubClient client(server,1883,callback,wifiClient);

#define ECHO_PIN 2
#define TRIG_PIN 4
#define led 5

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);
    intduration=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("p
        ublish 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("p
        ublish 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{
```

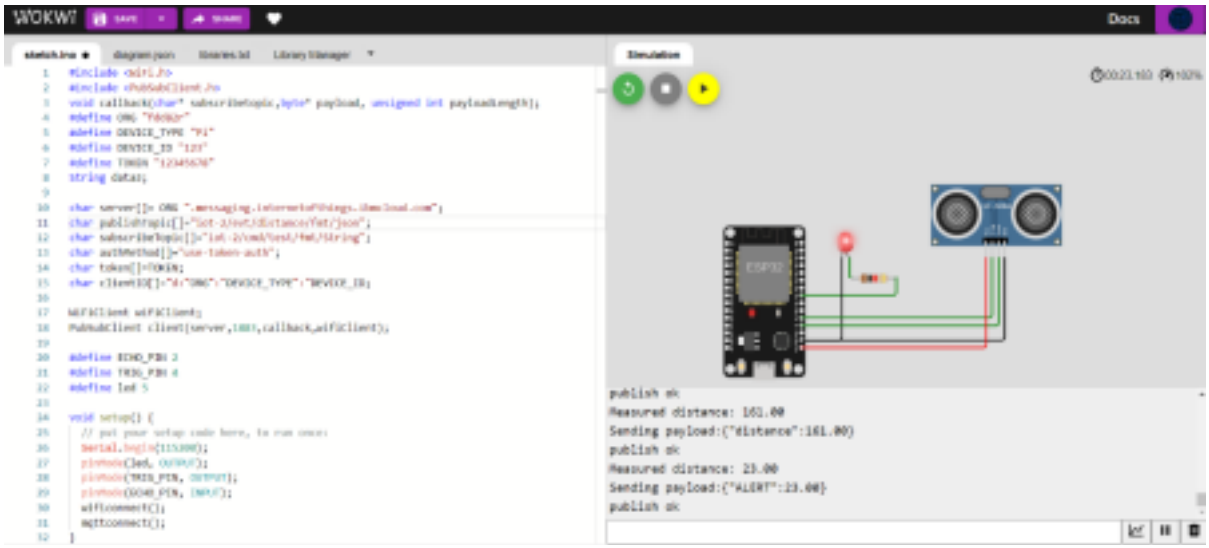
```
        Serial.println("subscribe to cmd failed");
    }
}

void callback(char* subscribeTopic, byte* payload, unsigned intpayloadLength){
    Serial.print("callback invoked for topic:");
    Serial.println(subscribeTopic);
    for(inti=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="";
}
```

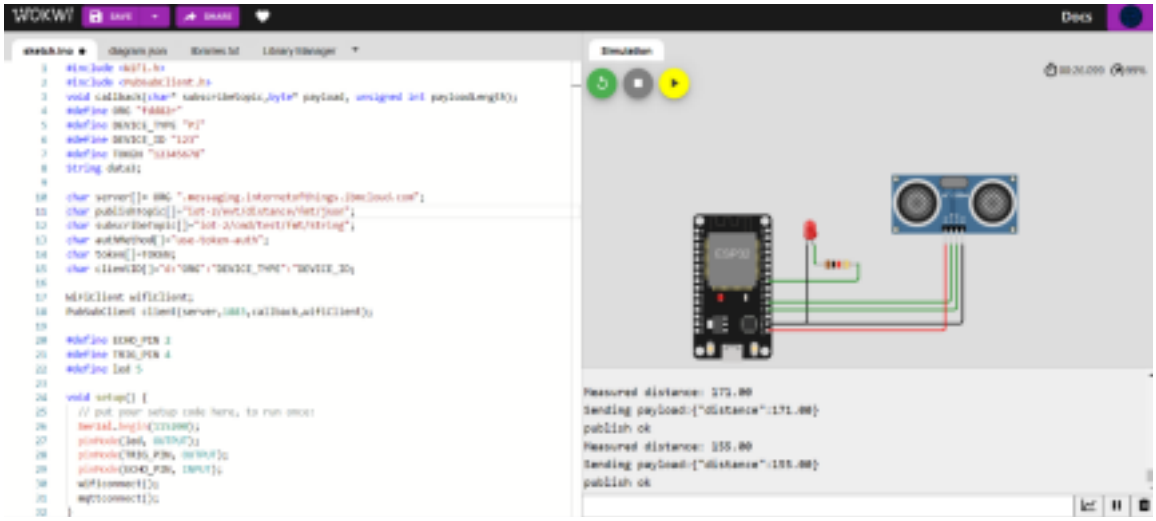
WOKWI project link

https://wokwi.com/projects/346327942785139284

Alert Case:



Normal Case:



Cloud storage: