IBM ASSIGNMENT 4

Date	07th NOVEMBER 2022
Team ID	PNT2022TMID04636
Project Name	Gas Leakage Monitoring and Alerting System

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

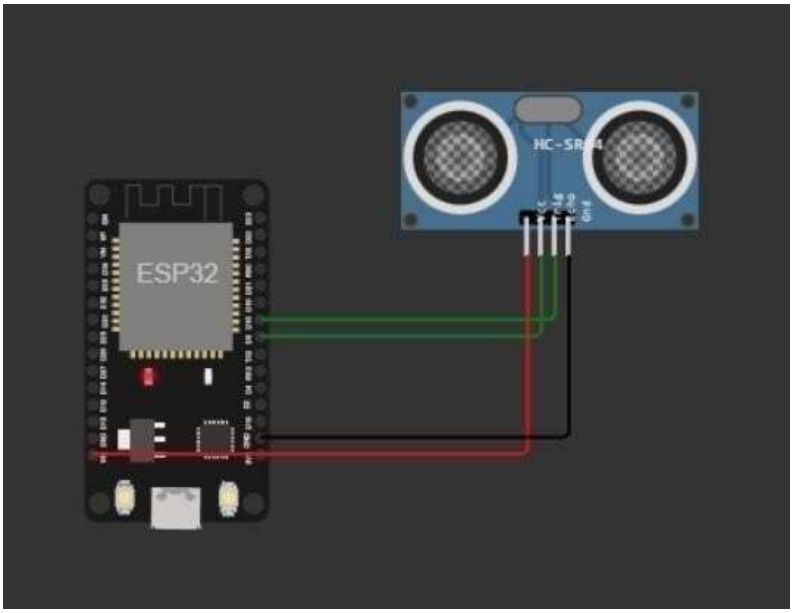
CODE

```
#include <WiFi.h>
#include  <PubSubClient.h>
WiFiClient;


#define ORG "nhpwjc"
#define DEVICE_TYPE "NodeMCU"
#define DEVICE_ID "USE YOUR ID"
#define TOKEN "USE YOUR TOKEN"
#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; float dist;


  void
setup()
{
    Serial.begin(115200); pinMode(trigpin,
    OUTPUT);
```

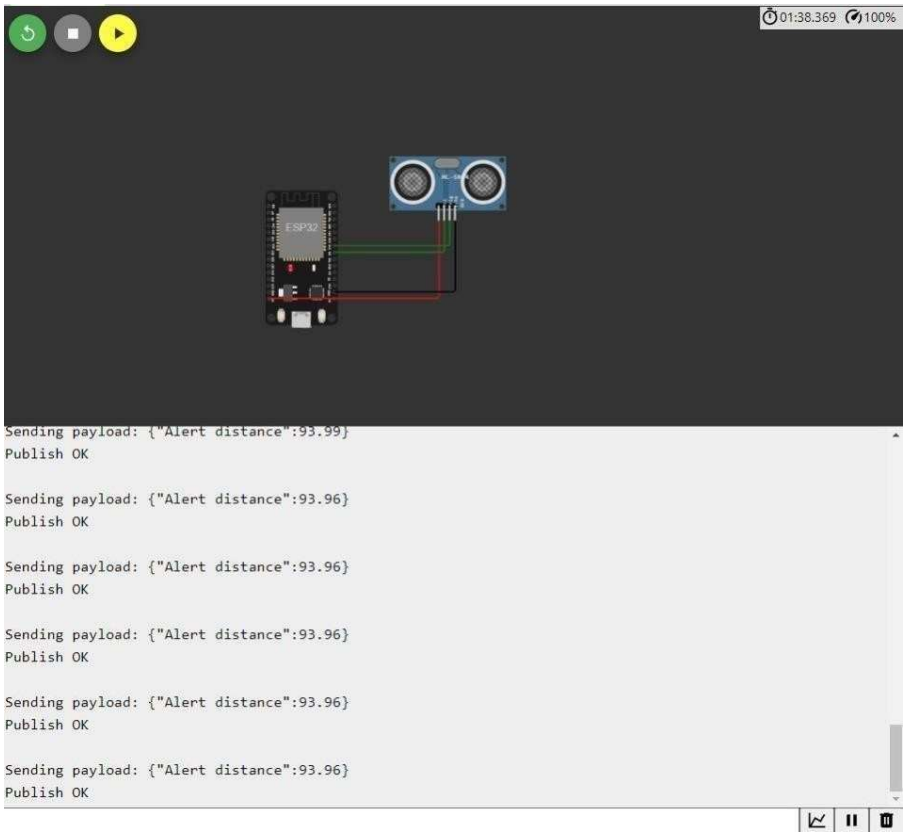
```
  pinMode(echopin,      INPUT);   wifiConnect();
  mqttConnect();
}   void   loop()   {   publishData();

  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(500);
    }
    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){
    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");
    } else {
      Serial.println("Publish FAILED");
    }
  }
}
```

CONNECTIONS



OUTPUT


01:38.369
100%
ESP32
Sending payload: {"Alert distance":93.99}
Publish OK
Sending payload: {"Alert distance":93.96}
Publish OK
Sending payload: {"Alert distance":93.96}
Publish OK
Sending payload: {"Alert distance":93.96}
Publish OK
Sending payload: {"Alert distance":93.96}
Publish OK
Sending payload: {"Alert distance":93.96}
Publish OK