Problem statement: IoT based safety gadget for child safety monitoring and notification.

Domain: Internet of Things

Assignment 4: Distance detection using ultrasonic sensor

By,

M.VIJAY(623519106044)

## **Question-1:**

Write code and connections in wokwi for ultrasonic sensor. Whenever distance is less than 100 cmssend "alert" to IBM cloud and display in device recent events.

### **WOKWILINK:**

https://wokwi.com/projects/348839716434477651

#### CODE:

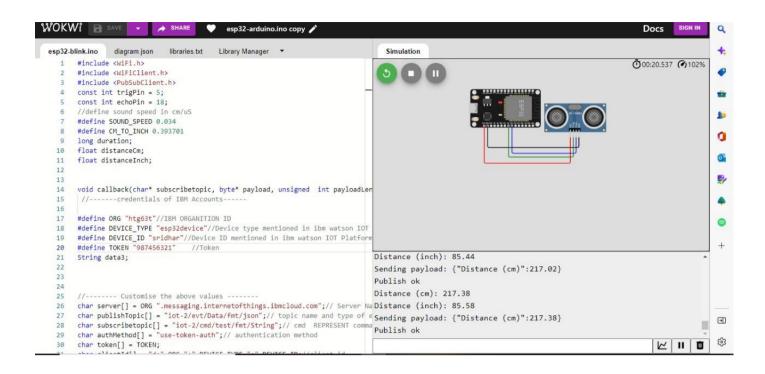
```
#include <WiFi.h>
#include <WiFiClient.h>
#include <PubSubClient.h>
const int trigPin = 5;
const int echoPin = 18;
//define sound speed in cm/uS
#define SOUND SPEED 0.034
#define CM_TO_INCH 0.393701
long duration;
float distanceCm;
float distanceInch;
void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
//----credentials of IBM Accounts----
#define ORG "pdisx0"//IBM ORGANITION ID
#define DEVICE_TYPE "esp32"//Device type mentioned in ibm watson IOT Platform
#define DEVICE ID "vijay"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "9344134341"
String data3;
//----- Customise the above values ------
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of event perform and format in
which data to be send
char subscribetopic[] = "iot-2/cmd/test/fmt/String";// cmd REPRESENT command type AND COMMAND IS TEST OF
FORMAT STRING
char authMethod[] = "use-token-auth";// authentication method
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id
WiFiClient wifiClient; // creating the instance for wificlient
PubSubClient client(server, 1883, callback ,wifiClient);
void setup() {
 Serial.begin(115200); // Starts the serial communication
 pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
 pinMode(echoPin, INPUT); // Sets the echoPin as an Input
 Serial.println();
 wificonnect();
```

```
mqttconnect();
}
void loop() {
  // Clears the trigPin
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  // Sets the trigPin on HIGH state for 10 micro seconds
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  // Reads the echoPin, returns the sound wave travel time in microseconds
  duration = pulseIn(echoPin, HIGH);
  // Calculate the distance
  distanceCm = duration * SOUND_SPEED/2;
  // Convert to inches
  distanceInch = distanceCm * CM_TO_INCH;
  // Prints the distance in the Serial Monitor
  Serial.print("Distance (cm): ");
  Serial.println(distanceCm);
  Serial.print("Distance (inch): ");
  Serial.println(distanceInch);
  PublishData(distanceCm);
  delay(1000);
  if (!client.loop()) {
    mqttconnect();
  }
}
  void PublishData(float Cm) {
  mqttconnect();//function call for connecting to ibm
     creating the String in in form JSon to update the data to ibm cloud
  String payload = "{\"Distance (cm)\":";
  payload += Cm;
  payload += "}";
  Serial.print("Sending payload: ");
  Serial.println(payload);
  if (client.publish(publishTopic, (char*) payload.c_str())) {
    Serial.println("Publish ok");// if it sucessfully upload data on the cloud then it will print publish
ok in Serial monitor or else it will print publish failed
  } else {
    Serial.println("Publish failed");
  }
void mqttconnect() {
  if (!client.connected()) {
    Serial.print("Reconnecting client to ");
    Serial.println(server);
    while (!!!client.connect(clientId, authMethod, token)) {
      Serial.print(".");
```

```
delay(500);
    }
     initManagedDevice();
     Serial.println();
  }
}
void wificonnect() //function defination for wificonnect
  Serial.println();
  Serial.print("Connecting to ");
  WiFi.begin("Wokwi-GUEST", "", 6);//passing the wifi credentials to establish the connection
  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 int payloadLength)
{
  Serial.print("callback invoked for topic: ");
  Serial.println(subscribetopic);
  for (int i = 0; i < payloadLength; i++) {</pre>
    //Serial.print((char)payload[i]);
    data3 += (char)payload[i];
  }
}
```

#### **OUTPUT:**

When object is not near to the ultrasonic sensor



# Data sent to the IBM cloud device when the object is far

