ASSIGNMENT 4

NAME	TARUN G
REGISTER NUMBER	113119UG04107
TEAM ID	PNT2022TMID22491
DATE	25 October 2022
MAXIMUM MARKS	4 Marks

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.

CODE:

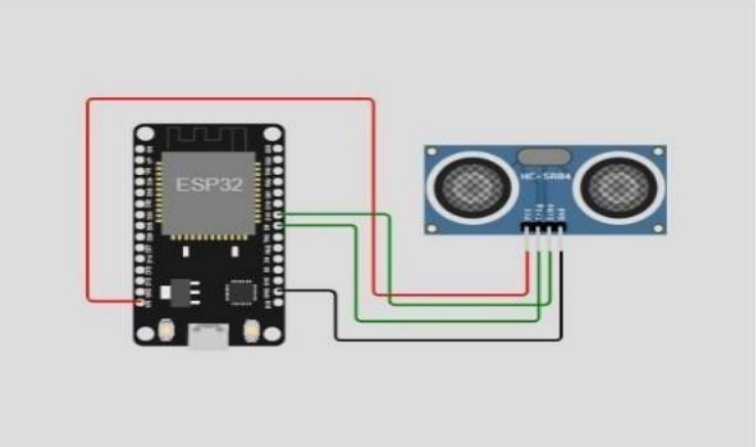
```
#include <WiFi.h>

#include

<PubSubClient.h>

void callback(char*subscribetopic,byte*payload,unsignedintpayloadLength);

//-------credentials of IBM Accounts------

#define ORG "Ashfaq1824"//IBM ORGANITION ID

#define DEVICE_TYPE "ESP32"//Device type mentioned in ibm watson IOT Platform

#define DEVICE_ID "12345"//Device ID mentioned in ibm watson IOT Platform

#define TOKEN "12345678" //Token

String data3;

char server[]=ORG".messaging.internetofthings.ibmcloud.com";
char publishTopic[]="iot-2/evt/Data/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;

WiFiClient wifiClient;

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

const int trigPin = 5;

const int echoPin = 18;
```

```
#define SOUND_SPEED

0.034 long duration;
float distance;
void setup()
{

Serial.begin(115200);

pinMode(trigPin, OUTPUT);

pinMode(echoPin, INPUT);

wificonnect();
mqttconnect();

}
void loop()
{

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

duration = pulseIn(echoPin,HIGH);
distance = duration *SOUND_SPEED/2;

Serial.print("Distance (cm): ");

Serial.println(distance);

if(distance<100)

{

Serial.println("ALERT!!");
delay(1000);

PublishData(distance);

delay(1000);
 if(!client.loop()) {

mqttconnect();

} }

delay(1000);
 }
```

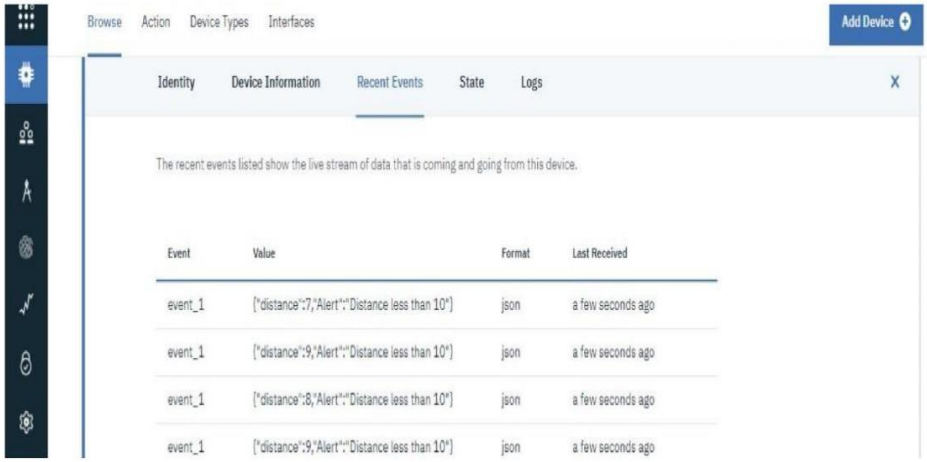
```
void PublishData(floatdist)
{

mqttconnect();

String payload="{\"Distance\":";

payload += dist;

payload += ",\"ALERT!!\":""\"Distance less than 100cms\"";

payload += "}";

Serial.print("Sending payload: ");

Serial.println(payload);

if (client.publish(publishTopic, (char*) payload.c_str())) {

Serial.println("Publish ok");

}
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()

{
```

```
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 int payloadLength)
{
Serial.print("callback invoked for topic: ");
Serial.println(subscribetopic);
for (int i = 0; i < payloadLength; i++)
{
//Serial.print((char)payload[i]);
data3 += (char)payload[i];
}
Serial.println("data: "+ data3);
data3="";
}
```

SCHEMATIC/CIRCUIT DIAGRAM



IBM CLOUD OUTPUT:



WOKWI LINK: https://wokwi.com/projects/347919574485697106