# ASSIGNMENT-4

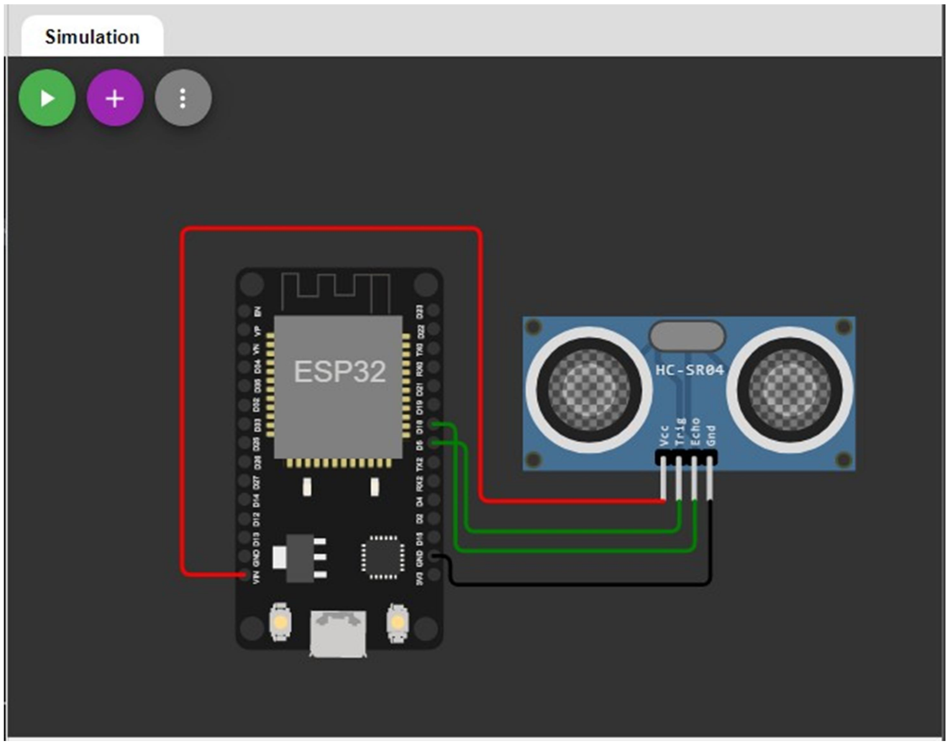
code

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

void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);

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

#define ORG "ytluse"//IBM ORGANITION ID

#define DEVICE_TYPE "2702"//Device type mentioned in ibm watson IOT Platform #define DEVICE_ID
"12345"//Device ID mentioned in ibm watson IOT Platform #define TOKEN "O+n)Eh+lNX0y3?rG!8"
//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);
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(float dist) { 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="";

}
```

**DIAGRAM.JSON**

```
{

"version": 1,

"author": "IRFANA FATHIMA A 19IT007",

"editor": "wokwi", "parts": [

{ "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": 6, "left": -66, "attrs": {}

},

{ "type": "wokwi-hc-sr04", "id": "ultrasonic1", "top": 32.56, "left": 81.02, "attrs": {} }

],

"connections": [

[ "esp:TX0", "$serialMonitor:RX", "", [] ],

[ "esp:RX0", "$serialMonitor:TX", "", [] ],

[ "esp:VIN", "ultrasonic1:VCC", "red", [ "h-31.67", "v-176.8", "h152", "v163.33" ]

],

[ "esp:D18", "ultrasonic1:ECHO", "green", [ "h11.37", "v64.67", "h121.33" ] ],

[ "esp:D5", "ultrasonic1:TRIG", "green", [ "h16.7", "v45.07", "h4" ] ],

[ "esp:GND.1", "ultrasonic1:GND", "black", [ "h8.7", "v14.7", "h138.67" ] ]

]
}
```


Simulation
ESP32
HC-SR04
Vcc
Trig
Echo
Gnd