

SPRINT - 3

Date	14 November
Team ID	PNT2022TMID35105
Project Name	Smart waste management system for metropolitan cities
Points	20

Created a IOT device to sense the level of bins and do code for device and send to Node Red using the API keys from Watson platform

CODE :

```
#include <WiFi.h>
#include <PubSubClient.h>
void callback(char* subscribetopic, byte* payload, unsigned int
payloadLength);

#define ORG "fzv53v"
#define DEVICE_TYPE "ESP32"
#define DEVICE_ID "ESP32_iot"
#define TOKEN "7(Ro*nk83MB7ZpKjMU"
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;
float level;

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;  
  level = 400 - distance;  
  Serial.print("Distance (cm): ");  
  Serial.println(level);  
  if(level>300)  
  {  
    Serial.println("ALERT!!");  
    delay(1000);  
    PublishData(level);  
    delay(1000);  
    if (!client.loop()) {  
      mqttconnect();  
    }  
  }  
  else  
  {  
    Publishdata2(level);  
    delay(1000);  
    if (!client.loop()) {  
      mqttconnect();  
    }  
  }  
  delay(1000);  
}
```

```
void PublishData(float dist) {  
  mqttconnect();  
  String payload = "{\"Level\":";  
  payload += dist;  
  payload += ", \"ALERT!!\": \"\" \"Bin Level less than 100 Units \";  
  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 Publishdata2(float dist) {  
    mqttconnect();  
    String payload = "{\"Level\":\"";  
    payload += dist;  
    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="";
}
```

Sensor circuit:

The image shows a WOKWI IoT simulator interface. On the left, the code for an ESP32 microcontroller is displayed, showing MQTT subscription and data publishing logic. The code includes a loop that publishes distance data to an MQTT topic and a callback function that prints the received data. On the right, the simulation window shows the physical connection between the ESP32 and an HC-SR04 ultrasonic sensor. The sensor is connected to the ESP32's GPIO pins. The simulation output shows the device successfully connecting to the MQTT broker and receiving data.

```
esp32_mqtt_keeplives.ino
52 }
53 }
54 else
55 {
56   Publishdata2(level);
57   delay(1000);
58   if (!client.loop()) {
59     mqttconnect();
60   }
61 }
62 delay(1000);
63 }
64 }
65
66 void PublishData(float dist) {
67   mqttconnect();
68   String payload = "{\"Level\":\"";
69   payload += dist;
70   payload += "\",\"ALERT!!\":\"Bin Level less than 100 Units \"\"";
71   payload += "\"}";
72   Serial.print("Sending payload: ");
73   Serial.println(payload);
74 }
75 if (client.publish(publishTopic, (char*) payload.c_str())) {
76   Serial.println("Publish ok");
77 } else {
78   Serial.println("Publish failed");
79 }
80 }
81 //
82 void Publishdata2(float dist) {
83   mqttconnect();
84   String payload = "{\"Level\":\"";
85   payload += dist;
86   payload += "\"}";
87   Serial.print("Sending payload: ");
```

Simulation

10.10.0.2
Reconnecting client to fiv53v.messaging.internetofthings.ibmcloud.com
iot-2/cmd/test/fmt/String
subscribe to cmd OK
Distance (cm): 398.01
ALERT!!

Watson IoT Platform:

IBM Watson IoT Platform

akashm3802@gmail.com
ID: fvz53v

Browse Action Device Types Interfaces

Browse Devices

All Devices Diagnose

This table shows a summary of all devices that have been added. It can be filtered, organized, and searched on using different criteria. To get started, you can add devices by using the Add Device button, or by using API.

Search by Device ID

Device Simulator

<input type="checkbox"/>	Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
> <input type="checkbox"/>	Bin_1	Disconnected	Bin	Device	Nov 10, 2022 6:50 PM	
> <input type="checkbox"/>	ESP32_iot	Connected	ESP32	Device	Nov 8, 2022 7:26 PM	
> <input type="checkbox"/>	weather_today	Disconnected	weather_device	Device	Nov 8, 2022 12:20 PM	
> <input type="checkbox"/>	weatheribm	Disconnected	weather2	Device	Nov 9, 2022 9:01 PM	

Items per page 50 | 1-4 of 4 items

1 of 1 page

0 Simulations running

IBM Watson IoT Platform

akashm3802@gmail.com
ID: fvz53v

Browse Action Device Types Interfaces

Recent Events

The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
Data	{"Level":398.01,"ALERT!":"Bin Level less than 1..."}	json	a few seconds ago
Data	{"Level":398.01,"ALERT!":"Bin Level less than 1..."}	json	a few seconds ago
Data	{"Level":398.01,"ALERT!":"Bin Level less than 1..."}	json	a few seconds ago

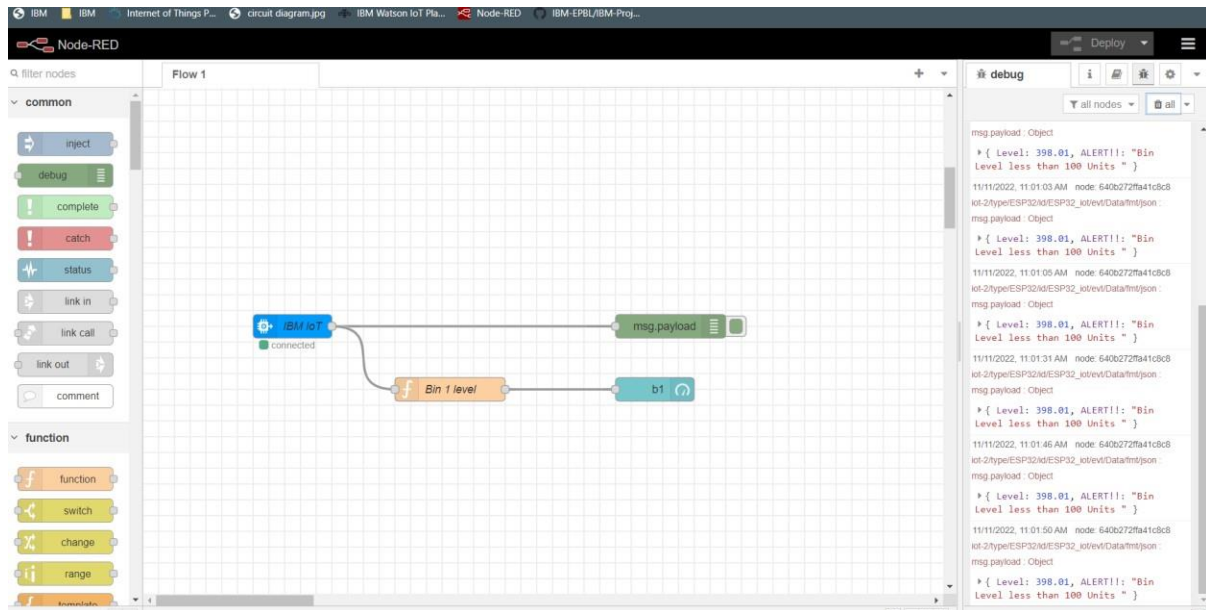
> ☐ weather_today Disconnected weather_device Device Nov 8, 2022 12:20 PM

> ☐ weatheribm Disconnected weather2 Device Nov 9, 2022 9:01 PM

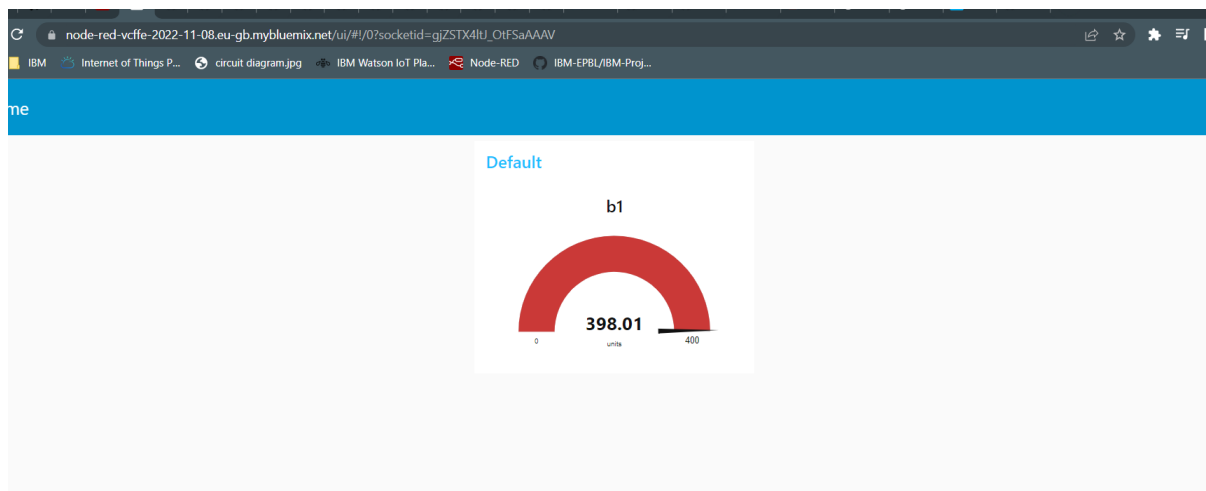
Items per page 50 | 1-4 of 4 items

1 of 1 page

Node-RED Connections :



Web UI :



Run the code here : <https://wokwi.com/projects/348010712871731794>