

SPRINT - 3

Date	11 November 2022
Team ID	PNT2022TMID21140
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 "0kzyfe"           // IBM organisation id
#define DEVICE_TYPE "final"    // Device type mentioned in ibm watson iot platform
#define DEVICE_ID "1234"       // Device ID mentioned in ibm watson iot platform
#define TOKEN "&sVA*)3VSDPzwM6!Jx" // TokenString
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 screenshot displays the WOKWI simulation interface. On the left, the code for `esp32_mqtt_keepalives.ino` is shown, which includes MQTT client setup, sensor pin definitions, and a loop that sends distance data via MQTT. On the right, the simulation shows an ESP32 board connected to an HC-SR04 ultrasonic sensor. The MQTT log at the bottom shows the device sending distance data and triggering an alert when the distance is less than 100 units.

```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 void callback(char* subscribetopic, byte* payload, unsigned int
4 payloadLength);
5
6 #define ORG "0kzyfc" // IBM organisation ID
7 #define DEVICE_TYPE "Ultrasonic_sensor" // Device type
8 #define DEVICE_ID "987654321" // Device ID mentioned in B
9 #define TOKEN "F6Dj7L9+v8IG2IAF" // Token
10
11 String data3;
12 char server[] = ORG "messaging.internetofthings.ibmcloud.com";
13 char publishTopic[] = "iot-2/evt/Data/fmt/json";
14 char subscribetopic[] = "iot-2/cmd/test/fmt/String";
15 char authMethod[] = "use-token-auth";
16 char token[] = TOKEN;
17 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
18 WiFiClient wifiClient;
19 PubSubClient client(server, 1883, callback ,wifiClient);
20
21 const int trigPin = 5;
22 const int echoPin = 18;
23 #define SOUND_SPEED 0.034
24 long duration;
25 float distance;
26 float level;
27 void setup() {
28   Serial.begin(115200);
29   pinMode(trigPin, OUTPUT);
30   pinMode(echoPin, INPUT);
31   wifiConnect();
32   mqttconnect();
33 }
34 void loop()
35 {
36   digitalWrite(trigPin, LOW);
37   delayMicroseconds(2);
38   digitalWrite(trigPin, HIGH);
39   delayMicroseconds(10);
40   digitalWrite(trigPin, LOW);
41   duration = pulseIn(echoPin, HIGH);
42   distance = (duration / 2) / SOUND_SPEED;
43   level = 100 - (distance * 100 / 310.02);
44   if (level < 100) {
45     Serial.println("Level: " + level);
46     client.publish(publishTopic, level);
47     Serial.println("Publish ok");
48     delay(1000);
49   }
50 }
```

Simulation

01:21.560 49%

ALERT!!
Sending payload: {"Level":310.02,"ALERT!!":"Bin Level less than 100 Units "}
Publish ok
Distance (cm): 310.02
ALERT!!
Sending payload: {"Level":310.02,"ALERT!!":"Bin Level less than 100 Units "}
Publish ok

Watson IoT Platform:

The screenshot displays the IBM Watson IoT Platform interface. The top navigation bar includes 'Browse', 'Action', 'Device Types', and 'Interfaces'. A sidebar on the left contains icons for various functions. The main content area shows details for a specific device with ID 987654321, which is an 'Ultrasonic_sensor' and is currently 'Connected'. The device was added on Nov 12, 2022 at 1:48 PM by the user 'rsangeetrsangeet07@gmail.com'. The connection status is 'Connected' with a connection time of Nov 15, 2022 at 2:05 PM and a client address of 106.198.11.75 using SecureToken.

Device ID	987654321
Device Type	Ultrasonic_sensor
Date Added	Nov 12, 2022 1:48 PM
Added By	rsangeetrsangeet07@gmail.com
Connection Status	Connected Connection Time: Nov 15, 2022 2:05 PM Client Address: 106.198.11.75 SecureToken

Bin	Status	Device Name	Device Type	Last Seen
BIN1	Connected	BIN_1	Device	Nov 15, 2022 12:18 PM
BIN2	Connected	BIN_2	Device	Nov 15, 2022 12:22 PM
BIN3	Connected	BIN_3	Device	Nov 15, 2022 12:24 PM

Items per page: 50 | 1-5 of 5 items | 1 of 1 page

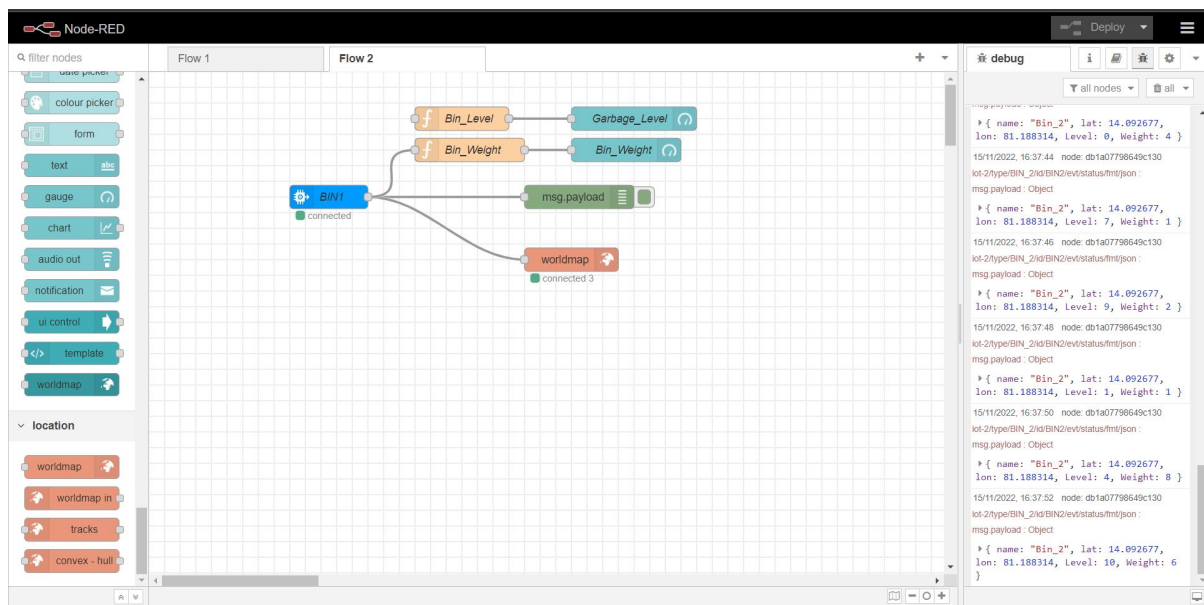
This screenshot shows the 'Recent Events' tab for the same device. It indicates that the events listed are a live stream of data. The events are shown in a table with columns for Event, Value, Format, and Last Received.

Event	Value	Format	Last Received
status	{"Level":266,"Alert":"Bin is available"}	json	a few seconds ago
status	{"Level":212,"Alert":"Warning!! Trash is about to ...	json	a few seconds ago
status	{"Level":389,"Alert":"Bin is available"}	json	a few seconds ago

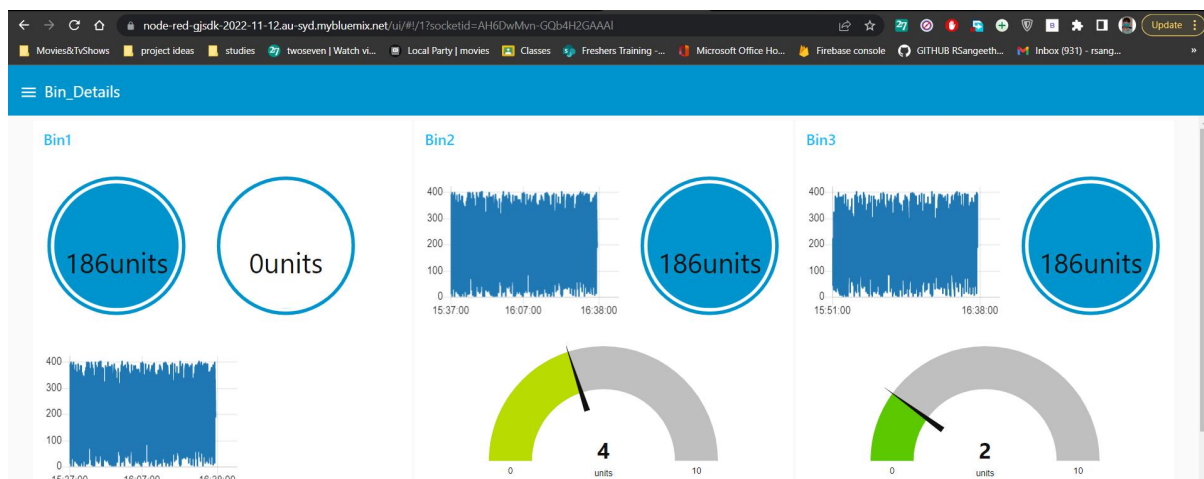
Bin	Status	Device Name	Device Type	Last Seen
BIN1	Connected	BIN_1	Device	Nov 15, 2022 12:18 PM
BIN2	Connected	BIN_2	Device	Nov 15, 2022 12:22 PM
BIN3	Connected	BIN_3	Device	No

0 Simulations running

Node-RED Connections :



Web UI :



Run the code here : <https://wokwi.com/projects/348375948659262034>