

TEAM ID	PNT2022TMID26538
PROJECT NAME	Smart Waste Management System for Metropolitan Cities

CODE FOR DATA TRANSFER FROM SENSORS

```
#include <WiFi.h>           //Library for WiFi
#include <PubSubClient.h>    //Library for MQTT
#include <ArduinoJson.h>     //Library for ArduinoJson
```

```
WiFiClient wifiClient;
```

```
// ..... Credentials on IBM Account- .....
```

```
#define ORG "k6spbs"        //IBM Organisation ID
#define DEVICE_TYPE "MSD"   //Device mentioned in IBM Watson IOT Platform
#define DEVICE_ID "12345"   //Device ID mentioned on IBM Watson IOT Platform
#define TOKEN "123456789"   //Token
#define speed 0.034
```

```
// ..... Customise above values .....
```

```
char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; //Server Name
char publishTopic[] = "iot-2/evt/Data/fmt/json";
char topic[] = "iot-2/cmd/home/fmt/String";
char authMethod[] = "use-token-auth"; //Authentication Method
```

```

char token[] = TOKEN;

char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;           //Client id

// .....

PubSubClient client(server, 1883, wifiClient);

void publishData();

const int trigpin=5;
const int echopin=18;
String command;
String data="";
String lat="13.167558";
String lon="80.244510";
String name="point2";
String icon="fa-trash-o";
String color="green";
long duration;
int dist;

void setup()
{
  Serial.begin(115200);
  pinMode(trigpin, OUTPUT);
  pinMode(echopin, INPUT);
  wifiConnect();
  mqttConnect();
}

void loop() {

```

```
publishData();
```

```
delay(500);
```

```
if (!client.loop()) {
```

```
    mqttConnect();
```

```
}
```

```
}
```

```
// ..... Retrieving to Cloud .....
```

```
void wifiConnect() {
```

```
    Serial.print("Connecting to "); Serial.print("Wifi");
```

```
    WiFi.begin("Wokwi-GUEST", "", 6);
```

```
    while (WiFi.status() != WL_CONNECTED) {
```

```
        delay(500);
```

```
        Serial.print(".");
```

```
    }
```

```
    Serial.print("WiFi connected, IP address: "); Serial.println(WiFi.localIP());
```

```
}
```

```
void mqttConnect() {
```

```
    if (!client.connected()) {
```

```
        Serial.print("Reconnecting MQTT client to "); Serial.println(server);
```

```
        while (!client.connect(clientId, authMethod, token)) {
```

```
            Serial.print(".");
```

```
            delay(1000);
```

```
        }
```

```
        initManagedDevice();
```

```
        Serial.println();
```

```
    }
```

```
}
```

```
void initManagedDevice() {  
  if (client.subscribe(topic)) {  
    Serial.println(client.subscribe(topic));  
    Serial.println("subscribe to cmd OK");  
  } else {  
    Serial.println("subscribe to cmd FAILED");  
  }  
}
```

```
//.....Publish Smart Bin level.....
```

```
void publishData()  
{  
  digitalWrite(trigpin,LOW);  
  digitalWrite(trigpin,HIGH);  
  delayMicroseconds(10);  
  digitalWrite(trigpin,LOW);  
  duration=pulseIn(echopin,HIGH);  
  dist=duration*speed/2;  
  dist=dist/4;  
  dist=100-dist;  
  if(dist>80){  
    icon="fa-trash";  
    color="red";  
  }else{  
    icon="fa-trash-o";  
    color="green";  
  }  
}
```

```
DynamicJsonDocument doc(1024);
```

```
String payload;
```

```
doc["Name"]=name;
```

```
doc["Latitude"]=lat;
```

```
doc["Longitude"]=lon;
```

```
doc["Icon"]=icon;
```

```
doc["FillPercent"]=dist;
```

```
doc["Color"]=color;
```

```
serializeJson(doc, payload);
```

```
delay(3000);
```

```
// ..... Print on LCD .....
```

```
Serial.print("\n");
```

```
Serial.print("Sending payload: ");
```

```
Serial.println(payload);
```

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

```
    Serial.println("Publish OK");
```

```
} else {
```

```
    Serial.println("Publish FAILED");
```

```
}
```

```
}
```

```
// ..... End of Program .....
```

CIRCUIT CONFIGURATION:

WOKWI

sketch.ino

```
1 #include <WiFi.h> // library for wifi
2 #include <PubSubClient.h> // library for MQTT #include <LiquidCrystal_I2C.h> LiquidCrystal
3 lcd(0x27, 20, 4);
4
5 WiFiClient wificlient; // creating instance for wificlient PubSubClient client(server, 1
6
7 #define ECHO_PIN 12 #define TRIG_PIN 13 float dist;
8
9 void setup()
10 {
11   Serial.begin(115200); pinMode(LED_BUILTIN, OUTPUT);
12   pinMode(TRIG_PIN, OUTPUT); pinMode(ECHO_PIN, INPUT);
13   //pir pin pinMode(4, INPUT);
14
15   //ledpins pinMode(23, OUTPUT); pinMode(2, OUTPUT); pinMode(4, OUTPUT); pinMode(15, OUTPUT
16
17   lcd.init(); lcd.backlight(); lcd.setCursor(1, 0); lcd.print(""); wifiConnect(); mqttConne
18 }
19
20
21 float readcmCM()
22 {
23   digitalWrite(TRIG_PIN, LOW); delayMicroseconds(2); digitalWrite(TRIG_PIN, HIGH); delayMic
24   duration * 0.034 / 2;
25 }
26
27
28 void loop()
29 {
30
31   lcd.clear();
```

Simulation

code for data tra...docx

Show all

WOKWI

esp32-blink.ino

```
1 #include <WiFi.h> // library for wifi
2 #include <PubSubClient.h> // library for MQTT
3 #include <LiquidCrystal_I2C.h>
4 #include <mjson.h>
5 LiquidCrystal_I2C lcd(0x27, 20, 4);
6
7 //----- credentials of IBM Accounts -----
8
9 #define ORG "9gbe4w" // IBM organisation id
10 #define DEVICE_TYPE "SWMSMC" // Device type mentioned in ibm watson iot pl
11 #define DEVICE_ID "ibmproject" // Device ID mentioned in ibm watson iot plat
12 #define TOKEN "SUNA41tG6-PqJ0rk5X" // Token
13
14 //----- customise above values -----
15
16 char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; // server n
17 char publishTopic[] = "iot-2/evt/data/fmt/json"; // topic na
18 char topic[] = "iot-2/cmd/led/fmt/String"; // cmd Repr
19 char authMethod[] = "use-token-auth"; // authenti
20 char token[] = TOKEN;
21 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID; //Client id
22
23 //-----
24
25 WiFiClient wificlient; // creating i
26 PubSubClient client(server, 1883, wificlient);
27
28 #define ECHO_PIN 12
29 #define TRIG_PIN 13
30 float dist;
31 String data3;
```

Simulation

00:04.783 36%

Connecting to Wifi...WiFi connected, IP address: 10.10.0.2
Reconnecting MQTT client to 9gbe4w.messaging.internetofthings.ibmcloud.com
IBM subscribe to cmd OK

Sending payload: {169.95 }
Publish OK

code for data tra...docx

Show all