

PROJECT DEVELOPMENT PHASE
DELIVERY OF SPRINT – 2

TEAM ID	PNT2022TMID04297
PROJECT NAME	Smart Waste Management System For Metropolitan Cities

CODE:

```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQTT
#include "DHT.h"// Library for dht11
#define DHTPIN 15 // what pin we're connected to
#define DHTTYPE DHT22 // define type of sensor DHT 11
#define LED 2

DHT dht (DHTPIN, DHTTYPE);//creating the instance by passing pin and typr of dht
connected

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

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

#define ORG "w5zj5y"//IBM ORGANITION ID
#define DEVICE_TYPE "abcdef"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "123456"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "?nlvmVuhCTf9JaTSuu" //Token
String data3;
float h, t;

//----- Customise the above values -----
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of event perform
and format in which data to be send
char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd REPRESENT command
type AND COMMAND IS TEST OF FORMAT STRING
char authMethod[] = "use-token-auth";// authentication method
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id

//-----
WiFiClient wifiClient; // creating the instance for wificlient
PubSubClient client(server, 1883, callback ,wifiClient); //calling the predefined client id by
passing parameter like server id,portand wificredential

void setup()// configureing the ESP32
{
```

```

Serial.begin(115200);
dht.begin();
pinMode(LED,OUTPUT);
delay(10);
Serial.println();
wificonnect();
mqttconnect();
}

void loop()// Recursive Function
{

    h = dht.readHumidity();
    t = dht.readTemperature();
    Serial.print("temp:");
    Serial.println(t);
    Serial.print("Humid:");
    Serial.println(h);

    PublishData(t, h);
    delay(1000);
    if (!client.loop()) {
        mqttconnect();
    }
}

/*.....retrieving to Cloud.....*/

void PublishData(float temp, float humid) {
    mqttconnect();//function call for connecting to ibm
    /*
        creating the String in in form JSon to update the data to ibm cloud
    */
    String payload = "{\"temp\":\"";
    payload += temp;
    payload += "\", \"Humid\":\"";
    payload += humid;
    payload += "\"}";

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

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

```

```

        Serial.println("Publish ok");// if it sucessfully upload data on the cloud then it will print
        publish ok in Serial monitor or else it will print publish failed
    } 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() //function defination for wificonnect

```

```

{

```

```

    Serial.println();
    Serial.print("Connecting to ");

```

```

    WiFi.begin("Wokwi-GUEST", "", 6);//passing the wifi credentials to establish the connection

```

```

    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)

```

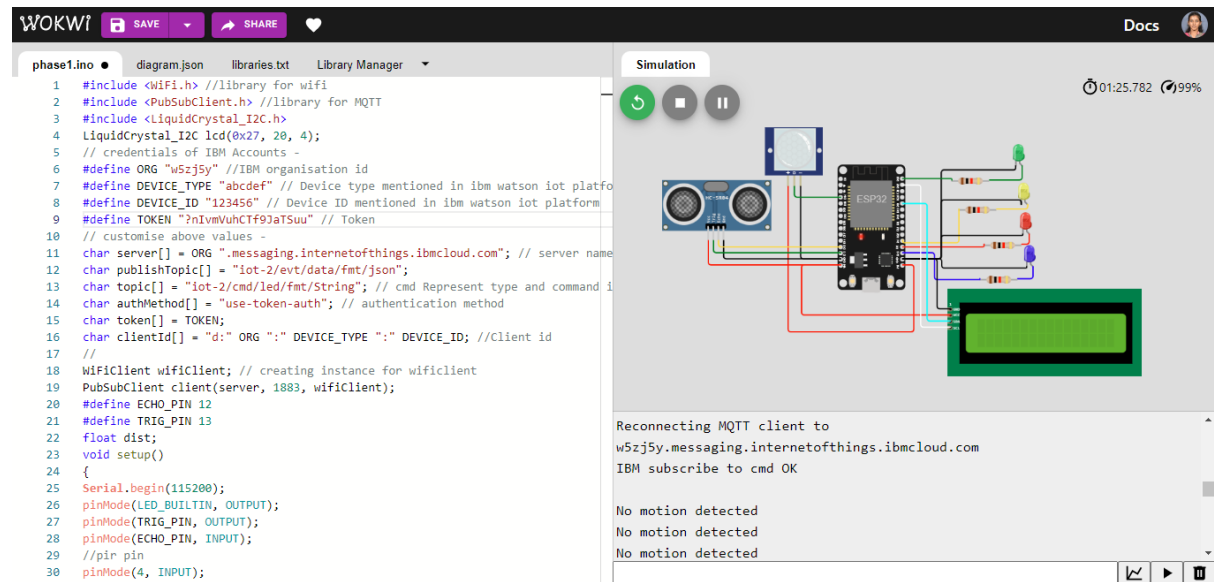
```

{

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);
if(data3=="lighton")
{
Serial.println(data3);
digitalWrite(LED,HIGH);
}
else
{
Serial.println(data3);
digitalWrite(LED,LOW);
}
data3="";
}

```

OUTPUT:



The screenshot shows the WOKWI IoT simulator interface. On the left is a code editor with a file named 'phase1.ino'. The code includes libraries for WiFi, PubSubClient, and LiquidCrystal_I2C. It defines an IBM Watson IoT device with specific credentials and sets up an MQTT client. The main loop subscribes to a topic, receives a payload, and controls an LED based on the received data. On the right is the 'Simulation' window, which displays a 3D model of the hardware: an ESP32 microcontroller, a blue LCD screen, a green LED, and a blue push button. Below the simulation window, a text log shows the MQTT client reconnecting and the LED being turned on ('IBM subscribe to cmd OK'). At the bottom, a status bar indicates 'No motion detected' three times.

```

1 #include <WiFi.h> //library for wifi
2 #include <PubSubClient.h> //library for MQTT
3 #include <LiquidCrystal_I2C.h>
4 LiquidCrystal_I2C lcd(0x27, 20, 4);
5 // credentials of IBM Accounts -
6 #define ORG "w5zj5y" //IBM organisation id
7 #define DEVICE_TYPE "abcdef" // Device type mentioned in ibm watson iot platfo
8 #define DEVICE_ID "123456" // Device ID mentioned in ibm watson iot platform
9 #define TOKEN "?nIvmVuhCTf9JaTSuu" // Token
10 // customise above values -
11 char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; // server name
12 char publishTopic[] = "iot-2/evt/data/fmt/json";
13 char topic[] = "iot-2/cmd/led/fmt/String"; // cmd Represent type and command i
14 char authMethod[] = "use-token-auth"; // authentication method
15 char token[] = TOKEN;
16 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID; //Client id
17 //
18 WiFiClient wificlient; // creating instance for wificlient
19 PubSubClient client(server, 1883, wificlient);
20 #define ECHO_PIN 12
21 #define TRIG_PIN 13
22 float dist;
23 void setup()
24 {
25   Serial.begin(115200);
26   pinMode(LED_BUILTIN, OUTPUT);
27   pinMode(TRIG_PIN, OUTPUT);
28   pinMode(ECHO_PIN, INPUT);
29   //pir pin
30   pinMode(4, INPUT);

```

Simulation

01:25.782 99%

Reconnecting MQTT client to
w5zj5y.messaging.internetofthings.ibmcloud.com
IBM subscribe to cmd OK

No motion detected
No motion detected
No motion detected

WOKWI

SAVE

SHARE

Docs

phase1.ino

diagram.json

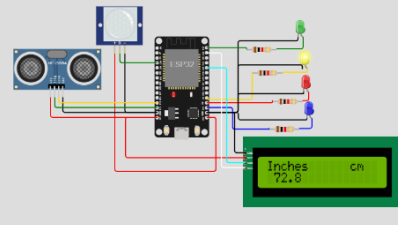
libraries.txt

Library Manager

```
1 #include <WiFi.h> //library for wifi
2 #include <PubSubClient.h> //library for MQTT
3 #include <LiquidCrystal_I2C.h>
4 LiquidCrystal_I2C lcd(0x27, 20, 4);
5 // credentials of IBM Accounts -
6 #define ORG "w5zj5y" //IBM organisation id
7 #define DEVICE_TYPE "abcdef" // Device type mentioned in ibm watson iot platform
8 #define DEVICE_ID "123456" // Device ID mentioned in ibm watson iot platform
9 #define TOKEN "?nIvmVuhCTf9JaTSuu" // Token
10 // customise above values -
11 char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; // server name
12 char publishTopic[] = "iot-2/evt/data/fmt/json";
13 char topic[] = "iot-2/cmd/led/fmt/String"; // cmd Represent type and command id
14 char authMethod[] = "use-token-auth"; // authentication method
15 char token[] = TOKEN;
16 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID; //Client id
17 //
18 WiFiClient wificlient; // creating instance for wificlient
19 PubSubClient client(server, 1883, wificlient);
20 #define ECHO_PIN 12
21 #define TRIG_PIN 13
22 float dist;
23 void setup()
24 {
25   Serial.begin(115200);
26   pinMode(LED_BUILTIN, OUTPUT);
27   pinMode(TRIG_PIN, OUTPUT);
28   pinMode(ECHO_PIN, INPUT);
29   //pir pin
30   pinMode(4, INPUT);
```

Simulation

01:51.820 69%



Sending distance: 184.96
Publish OK
Motion Detected
Lid Opened
Warning!! Trash is about to cross 50% of bin level
Lid Closed

IBM Watson IoT Platform

412519205153@smartinternz.com
ID: w5zj5y

Browse Action Device Types Interfaces

Add Device

Search by Device ID

Device Simulator

Device ID	Status	Device Type	Class ID	Date Added
123456	Connected	abcdef	Device	Nov 18, 2022 1:11 PM

Identity

Device Information

Recent Events

State

Logs

Device ID

123456

Device Type

abcdef

Date Added

Nov 18, 2022 1:11 PM

Added By

412519205153@smartinternz.com

Connection Status

Connected
Connection Time: Nov 18, 2022 3:11 PM
Client Address: 50.31.197.64 Insecure

Items per page 50 | 1-1 of 1 item

1 Simulation running

IBM Watson IoT Platform

412519205153@smartinternz.com
ID: w5zj5y

Browse Action Device Types Interfaces

Add Device

123456 Disconnected

Search by Device ID

Device Simulator

Device ID	Status	Device Type	Class ID	Date Added
123456	Disconnected	abcdef	Device	Nov 18, 2022 1:11 PM

Identity

Device Information

Recent Events

State

Logs

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

Event	Value	Format	Last Received
event_1	{"randomNumber":64}	json	a few seconds ago
event_1	{"randomNumber":12}	json	a few seconds ago
event_1	{"randomNumber":62}	json	a few seconds ago
event_1	{"randomNumber":95}	json	a few seconds ago
event_1	{"randomNumber":44}	json	a few seconds ago

1 Simulation running

Activate Windows
Go to PC settings to activate Windows.

WOWKI SIMULATION LINK:

<https://wokwi.com/projects/348583650995995219>