# Delivery of Sprint-2

|  |  |
| --- | --- |
| **DATE** | 5 November 2022 |
| **TEAM ID** | PNT2022TMID05666 |
| **PROJECT NAME** | SMART WASTE MANAGEMENT FOR METROPOLITAN CITIES |

**Code for Data Transfer from Sensors**

#include <WiFi.h> // library for wifi

#include <PubSubClient.h> // library for MQTT #include <LiquidCrystal\_I2C.h> LiquidCrystal\_I2C

lcd(0x27, 20, 4);

// credentials of IBM Accounts -

|  |  |
| --- | --- |
| #define ORG "ktymlx" | // IBM organisation id |
| #define DEVICE\_TYPE "new" | // Device type mentioned in ibm watson iot platform |
| #define DEVICE\_ID "09874" | // Device ID mentioned in ibm watson iot platform |
| #define TOKEN "hariwignesh123" | // Token |

// 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/led/fmt/String"; // cmd Represent type and command is test format of strings char authMethod[] = "usetoken- auth"; // authentication method char token[] = TOKEN;

char clientId[] = "d:" ORG ":" DEVICE\_TYPE ":" DEVICE\_ID; //Client id

//

WiFiClient wifiClient; // creating instance for wificlient PubSubClient client(server, 1883, wifiClient);

#define ECHO\_PIN 12 #define TRIG\_PIN 13 float dist;

void setup()

{

**Serial**.begin(115200); pinMode(LED\_BUILTIN, OUTPUT);

pinMode(TRIG\_PIN, OUTPUT); pinMode(ECHO\_PIN, INPUT);

//pir pin pinMode(4, INPUT);

//ledpins pinMode(23, OUTPUT); pinMode(2, OUTPUT); pinMode(4, OUTPUT); pinMode(15, OUTPUT);

lcd.init(); lcd.backlight(); lcd.setCursor(1, 0); lcd.print(""); wifiConnect(); mqttConnect();

}

float readcmCM()

{

digitalWrite(TRIG\_PIN, LOW); delayMicroseconds(2); digitalWrite(TRIG\_PIN, HIGH); delayMicroseconds(10); digitalWrite(TRIG\_PIN, LOW); int duration = pulseIn(ECHO\_PIN, HIGH); return

duration \* 0.034 / 2;

}

void loop()

{

lcd.clear();

publishData(); delay(500); if (!client.loop())

{

mqttConnect(); // function call to connect to IBM

}

}

/\* -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(500);

}

initManagedDevice(); **Serial**.println();

}

}

void initManagedDevice()

{

if (client.subscribe(topic))

{

**Serial**.println("IBM subscribe to cmd OK");

}

else

{

**Serial**.println("subscribe to cmd FAILED");

}

}

void publishData()

{

float cm = readcmCM();

if(digitalRead(34)) //PIR motion detection

{

**Serial**.println("Motion Detected"); **Serial**.println("Lid Opened"); digitalWrite(15, HIGH);

}

else

{

digitalWrite(15, LOW);

}

if(digitalRead(34)== true)

{

if(cm <= 100) //Bin level detection

{

digitalWrite(2, HIGH);

**Serial**.println("High Alert!!!,Trash bin is about to be full");

**Serial**.println("Lid Closed"); lcd.print("Full! Don't use"); delay(2000); lcd.clear(); digitalWrite(4, LOW);

digitalWrite(23, LOW);

}

else if(cm > 150 && cm < 250)

{

digitalWrite(4, HIGH);

**Serial**.println("Warning!!,Trash is about to cross 50% of bin level"); digitalWrite(2, LOW);

digitalWrite(23, LOW);

}

else if(cm > 250 && cm <=400)

{

digitalWrite(23, HIGH); **Serial**.println("Bin is available"); digitalWrite(2,LOW); digitalWrite(4, LOW);

}

delay(10000); **Serial**.println("Lid Closed");

}

else

{

**Serial**.println("No motion detected");

}

if(cm <= 100)

{

digitalWrite(21,HIGH);

String payload = "{\"High Alert!!\":\""; payload += cm; payload

+= "left\" }";

**Serial**.print("\n"); **Serial**.print("Sending payload: ");

**Serial**.println(payload); if (client.publish(publishTopic, (char\*) payload.c\_str())) // if data is uploaded to cloud successfully,prints publish ok or prints publish failed

{

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

}

}

if(cm <= 250)

{

digitalWrite(22,HIGH);

String payload = "{\"Warning!!\":\""; payload += dist; payload += "left\" }"; **Serial**.print("\n");

**Serial**.print("Sending distance: "); **Serial**.println(cm); if(client.publish(publishTopic, (char\*) payload.c\_str()))

{

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

}

else

{

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

}

}

float inches = (cm / 2.54); //print on LCD lcd.setCursor(0,0); lcd.print("Inches"); lcd.setCursor(4,0); lcd.setCursor(12,0); lcd.print("cm"); lcd.setCursor(1,1); lcd.print(inches, 1); lcd.setCursor(11,1); lcd.print(cm, 1); lcd.setCursor(14,1); delay(1000); lcd.clear();

}

# Connection Diagram

