# SPRINT 4

|  |  |
| --- | --- |
| TEAM ID | PNT2022TMID50061 |
| PROJECT NAME | Real-Time River water Quality Monitoring and Control System |

CODE:-

#include "DHTesp.h" #include <cstdlib> #include <time.h> #include <WiFi.h>

#include <PubSubClient.h>

#define ORG "pfrrli"

#define DEVICE\_TYPE "Rasp" #define DEVICE\_ID "12345"

#define TOKEN "12345678"

#define speed 0.034

char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; char publishTopic[] = "iot-2/evt/data/fmt/json";

char authMethod[] = "use-token-auth"; char token[] = TOKEN;

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

WiFiClient wifiClient;

PubSubClient client(server, 1883, wifiClient); float temperature = 0;

int pH = 0;

String quality\_status = ""; String temperture\_status = "";

void setup() {

**Serial**.begin(99900);

wifiConnect(); mqttConnect();

}

void loop() { srand(time(0));

//initial variable int p;

temperature = random(-20,40); pH = random(0,14);

if(pH > 6.5 && pH < 8.5){ p = 0;

}

else{

p = 1;

}

//set a quality status switch (p) {

case 0:

quality\_status = "Drinkable"; break;

case 1:

quality\_status = "Not Drinkable"; break;

}

//Obivously the output.It is like json format 'cause it will help us for future sprints String payload = "{";

payload+="\"pH level is \":"; payload+=pH; payload+=",";

payload+="\"Temperature of Water\":"; payload+=(int)temperature; payload+=",";

payload+="\"Alert\":\""+quality\_status+"\"}";

**Serial**.println(payload);

if(client.publish(publishTopic, (char\*) payload.c\_str()))

{

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

}

else{

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

}

delay(1000);

if (!client.loop())

{

mqttConnect();

}

}

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);

}

**Serial**.println();

}

}

# DIAGRAM.JSON:-

{

"version": 1,

"author": "PNT2022TMID51903",

"editor": "wokwi", "parts": [

{ "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": -16.32, "left": -0.82, "attrs": {} },

{

"type": "wokwi-dht22",

"id": "dht1",

"top": -30.22,

"left": 165.89,

"attrs": { "temperature": "59.3" }

}

],

"connections": [

[ "esp:TX0", "$serialMonitor:RX", "", [] ],

[ "esp:RX0", "$serialMonitor:TX", "", [] ],

[ "dht1:SDA", "esp:D15", "green", [ "v0" ] ],

[ "dht1:VCC", "esp:3V3", "red", [ "v0" ] ],

[ "dht1:GND", "esp:GND.1", "black", [ "v0" ] ]

]

}

OUTPUT:-



