SPRINT 4

Framework (Cloud deployment)

Date	08 November 2022
Team ID	PNT2022TMID17245
Project Name	Project - Gas Leakage Monitoring and Alerting System for Industries.

Cloud Deployment:

- On cloud, analyse and store the data and communicate wirelessly for further analysis is possible. Anyone can access the leakage data from anywhere using any Internet enabled device like PC, tablet or smart phone, and analyse it.
- The fire caused by gas leakage not only harms the owner but also people who are not far from the fire. From these problems, the authors make a design of cloud computing-based detection system of gas leak using a microcontroller NodeMCU Esp8266 that can provide notifications via smartphone in case of fire and automatically do the first treatment by turning on the exhaust. Notification sends via the smartphone appear not only when opening the application, but also when it does not open the application.

Code:

```
#include <ESP8266WiFi.h>

#include <PubSubClient.h>

WiFiClient wifiClient;


//Enter your network credentials below in ssid and password const

char* ssid = " ";

const char* password = " ";


//Provide your IBM IOT Platform credentials

#define ORG ""

#define DEVICE_TYPE ""

#define DEVICE_ID ""
```

```
#define TOKEN ""



char server[] = ORG ".messaging.internetofthings.ibmcloud.com";  char
publishTopic[] = "iot-2/evt/Data/fmt/json";
char topic[] = "iot-2/cmd/home/fmt/String"; // cmd REPRESENT command type AND COMMAND IS TEST
OF FORMAT STRING  char authMethod[] = "use-token-auth"; char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;


void callback(char* topic, byte* payload, unsigned int payloadLength); PubSubClient
client(server, 1883, callback, wifiClient);

int publishInterval = 5000; // 30 seconds long
lastPublishMillis;
String data;

void setup()
{

  Serial.begin(9600);
  pinMode(D0, OUTPUT);
  wifiConnect();
  mqttConnect();
}


void loop() {
  if (millis() - lastPublishMillis > publishInterval)
  {

    publishData();
```

```
    lastPublishMillis = millis();
  }

  if (!client.loop()) {
    mqttConnect();

  }
}

void wifiConnect() {
  Serial.print("Connecting to "); Serial.print(ssid);
  WiFi.begin(ssid, password); while (WiFi.status()
  != WL_CONNECTED) {
    delay(500);
    Serial.print(".");

  }
  Serial.print("nWiFi 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(client.subscribe(topic));
    Serial.println("subscribe to cmd OK");
  } else {
  Serial.println("subscribe to cmd FAILED");
    }
}

void callback(char* topic, byte* payload, unsigned int payloadLength) {

  Serial.print("callback invoked for topic: ");
  Serial.println(topic);

  for (int i = 0; i < payloadLength; i++)
    { //Serial.print((char)payload[i]);
    data += (char)payload[i];
  }

  Serial.println("Data: " + data ); if
  (data == "lon") { digitalWrite(D0,
  HIGH);
  }
  else if (data == "loff") {
    digitalWrite(D0, LOW);
  }
  data = "";
}
```

```
void publishData()
{

  int a = 10;
  Serial.print("Sample Value: ");
  Serial.println(a);

  String payload = "{\"d\":{\"data\":"; payload
  += a;
  payload += "}}";

  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");

  } }
```