Project Development Phase
Sprint – 2 (Cloud Service)

Date	07 November 2022
Team ID	PNT2022TMID35867
Project Name	Project – Gas leakage monitoring and alerting system for industries

Installed Libraries – Arduino IDE:

1. PubSubClient
2. LiquidCrystal_I2C
3. ESP8266WiFi

Source Code:

```
//Libraries
#include <PubSubClient.h>
#include <LiquidCrystal_I2C.h>
#include <ESP8266WiFi.h>

//Define Variables
LiquidCrystal_I2C lcd(0x27, 16, 2);
#define Buzzer D5
#define Green D6
#define Sensor A0
const char* ssid = "Airtel-Hotspot-958A";
const char* password = "9889i1bb";

//Cloud Service and Authentication
#define ORG "wf2kmp"
#define DEVICE_TYPE "GLMASFI_IOT_Device_Cloud_Service"
#define DEVICE_ID "PNT2022TMID35867"
#define TOKEN "PNT2022TMID35867"
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char topic[] = "iot-2/evt/status/fmt/json";
char authMethod[] = "use-token-auth";
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;

//Callback function
void callback(char* topic, byte* payload, unsigned int length) {
 Serial.println("callback invoked");
}
WiFiClient wifiClient;
PubSubClient client(server, 1883, callback, wifiClient);

//Format of cloud data for different Gas Level
void PublishData1(float senso){
String payload= "{\"Normal Gas Level\":";
payload += senso;
payload+="}";
Serial.print("Sending payload: ");
Serial.println(payload);
if (client.publish(topic, (char*) payload.c_str())) {
  Serial.println("Publish ok");
```

```
} else {
  Serial.println("Publish failed");
}
}
void PublishData2(float senso){
String payload= "{\"Alert! Gas Level\":";
payload += senso;
payload+="}";
Serial.print("Sending payload: ");
Serial.println(payload);
if (client.publish(topic, (char*) payload.c_str())) {
  Serial.println("Publish ok");
} else {
  Serial.println("Publish failed");
}

}
void PublishData3(float senso){
String payload= "{\"Danger! Gas Level\":";
payload += senso;
payload+="}";
Serial.print("Sending payload: ");
Serial.println(payload);
if (client.publish(topic, (char*) payload.c_str())) {
  Serial.println("Publish ok");
} else {
  Serial.println("Publish failed");
}
}

//Setup function
void setup() {
 Serial.begin(9600);
 lcd.backlight();
 lcd.init();
 pinMode(Green, OUTPUT);
 pinMode(Buzzer, OUTPUT);
 pinMode(Sensor, INPUT);
 Serial.begin(115200);
 Serial.println();

 Serial.print("Connecting to ");
 Serial.print(ssid);
 WiFi.begin(ssid, password);
 while (WiFi.status() != WL_CONNECTED) {
  delay(500);
  Serial.print(".");
 }
 Serial.println("");
 Serial.print("WiFi connected, IP address: ");
 Serial.println(WiFi.localIP());
}

//Format of IOT device Working for different Gas Level
void notifiaction() {
 int sensor = analogRead(Sensor);
 Serial.println(sensor);
```

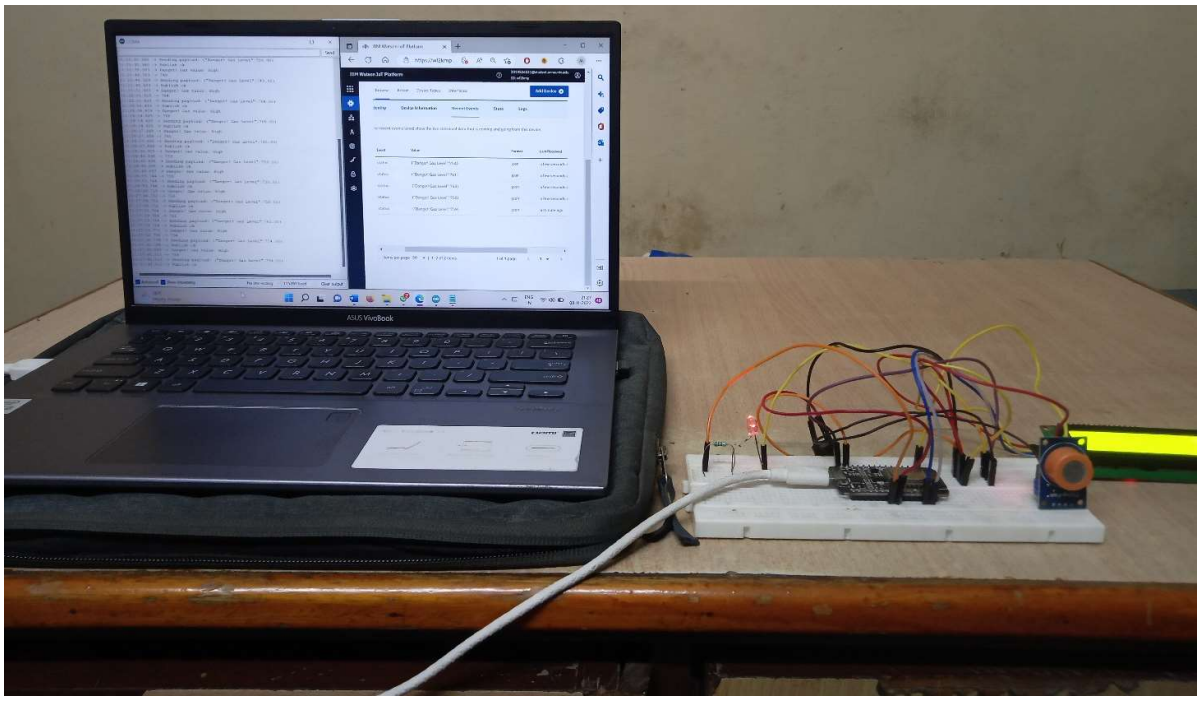
```
 if (sensor >= 740) {
  digitalWrite(Green, HIGH);
  digitalWrite(Buzzer, HIGH);
  PublishData3(sensor);
  delay(3000);
  lcd.setCursor(0, 1);
  Serial.println("Danger! Gas value: High");
 } else if (700<=sensor){
  digitalWrite(Buzzer, HIGH);
  digitalWrite(Green, HIGH);
  PublishData2(sensor);
  delay(750);
  digitalWrite(Buzzer,LOW);
  digitalWrite(Green, LOW);
  delay(1000);
  lcd.setCursor(0, 1);
  Serial.println("Gas value: Moderate - Alert");
 }
 else{
  digitalWrite(Green, LOW);
  digitalWrite(Buzzer, LOW);
  PublishData1(sensor);
  lcd.setCursor(0, 1);
  Serial.println("Gas value: Normal");
 }
 lcd.setCursor(0, 0);
 lcd.print("Value : ");
 lcd.print(sensor);
}


//Loop Function
void loop() {
 notifiaction();
 if (!!!client.connected()) {
 Serial.print("Reconnecting client to ");
 Serial.println(server);
 while (!!!client.connect(clientId, authMethod, token)) {
  Serial.print(".");
  delay(500);
 }
 Serial.println();
 }
 delay(10000);
}
```

Verification:

Concentration (ppm)	Level	IOT device Indication	Cloud Service Update
> =740	HIGH	Danger! Gas value: High	Danger + Gas sensor value
700-740	MODERATE	Gas value: Moderate - Alert	Alert + Gas sensor value
<=700	LOW	Gas value: Normal	Normal + Gas sensor value

Output:

1. **Interfacing IOT Device and Cloud Service:**



2. **Serial Monitor Output:**

COM4

```
19:45:26.207 -> Sending payload: {"Normal Gas Level":668.00}
19:45:26.207 -> Publish ok
19:45:26.207 -> Gas value: Normal
19:45:36.267 -> 685
19:45:36.267 -> Sending payload: {"Normal Gas Level":685.00}
19:45:36.267 -> Publish ok
19:45:36.267 -> Gas value: Normal
19:45:46.275 -> 710
19:45:46.275 -> Sending payload: {"Alert! Gas Level":710.00}
19:45:46.275 -> Publish ok
19:45:48.043 -> Gas value: Moderate - Alert
19:45:58.069 -> 725
19:45:58.069 -> Sending payload: {"Alert! Gas Level":725.00}
19:45:58.069 -> Publish ok
19:45:59.786 -> Gas value: Moderate - Alert
19:46:09.822 -> 754
19:46:09.822 -> Sending payload: {"Danger! Gas Level":754.00}
19:46:09.822 -> Publish ok
19:46:12.827 -> Danger! Gas value: High
19:46:22.855 -> 768
19:46:22.855 -> Sending payload: {"Danger! Gas Level":768.00}
19:46:22.855 -> Publish ok
19:46:25.863 -> Danger! Gas value: High
19:46:35.855 -> 783
19:46:35.855 -> Sending payload: {"Danger! Gas Level":783.00}
19:46:35.903 -> Publish ok
19:46:38.903 -> Danger! Gas value: High
19:46:48.914 -> 700
19:46:48.914 -> Sending payload: {"Alert! Gas Level":700.00}
19:46:48.914 -> Publish ok
19:46:50.682 -> Gas value: Moderate - Alert
19:47:00.685 -> 735
19:47:00.685 -> Sending payload: {"Alert! Gas Level":735.00}
19:47:00.685 -> Publish ok
19:47:02.458 -> Gas value: Moderate - Alert
19:47:12.455 -> 736
```

Autoscroll Show timestamp

3. IBM Watson IOT Platform Output (Cloud Service Updates):