FINAL DELIVERABLE

FINAL CODE

Project Title	Gas Leakage Monitoring and Alerting System
Team ID	PNT2022TMID15951
Team Members	Akshaya KS Abitha J Ashwini R Barani G

CODE:

Detect the gas Leakage

```
#include<Servo.h>

#include <TinyGPS++.h>

#include <SoftwareSerial.h>

#include<LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(32, 16, 2);

int GPSBaud = 9600;

TinyGPSPlus gps;

SoftwareSerial sgps(13, 15); //Rx ,  Tx gps

SoftwareSerial sgsm(3, 1); // Rx ,  Tx gsm

#define KNOB 3

#define LEVER 2

Servo myservo;

int gas = A5;

int sensorValue = 0;
```

```
bool gateClosed = true;



void setup()
{
  Serial.begin(9600);
  pinMode(LEVER, INPUT);
  myservo.attach(KNOB);
  myservo.write(90);
  sgsm.begin(9600);
  sgps.begin(9600);
  lcd.init();
  lcd.clear();
  lcd.backlight();
  lcd.setCursor(3,0);
  lcd.print("GAS LEAKAGE");
  lcd.setCursor(4,1);
  lcd.print("DETECTION");
  delay(3000);
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("Gas Value: ");
}

void loop()
```

```
{
  sensorValue = analogRead(gas);
  Serial.println(sensorValue);
  if(sensorValue > 500 && !gateClosed)
  {
    Serial.println("GAS DETECTED");
    lcd.setCursor(0,1);
    lcd.print("GAS DETECTED  ");
    sendSMS("GAS IS DETECTED!!");
    myservo.write(90);
    gateClosed = true;
    sendSMS("THE KNOB IS CLOSED");
    lcd.setCursor(0,1);
    lcd.print("KNOB IS CLOSED");
    delay(1000);
  }
  else if(sensorValue > 500 && gateClosed)
  {
    Serial.println("GAS DETECTED");
    lcd.setCursor(0,1);
    lcd.print("GAS DETECTED  ");
    sendSMS("GAS IS DETECTED!!");
    sendSMS("THE KNOB IS ALREADY CLOSED");
    lcd.setCursor(0,1);
    lcd.print("KNOB IS CLOSED");
    delay(1000);
  }
  else
  {
    byte buttonState = digitalRead(LEVER);
    if(buttonState == HIGH)
    {
      myservo.write(0);
      gateClosed = false;
      Serial.println("GATE IS OPENED");
    }
    else
    {
      myservo.write(90);
      gateClosed = true;
      Serial.println("GATE IS CLOSED");
    }
  }
}
void sendSMS(char*message)
{
  while (sgps.available() > 0)
    if (gps.encode(sgps.read()))
    {
      if (gps.location.isValid())
      {
        sgsm.listen();
        sgsm.print("\r");
        delay(1000);
        sgsm.print("AT+CMGF=1\r"); // AT COMMAND TO SEND SMS
        delay(1000);
        /*Replace XXXXXXXXXX to 10 digit mobile number &
        ZZ to 2 digit country code*/
        sgsm.print("AT+CMGS=\"+919025681637\"\r"); // REGISTERED
NUMBER TO SEND SMS
        delay(1000);
        //The text of the message to be sent.
        sgsm.print(message);
        sgsm.print("https://www.google.com/maps/?q="); // MAPS
        sgsm.print(gps.location.lat(), 6); // LAT
        sgsm.print(",");
        sgsm.print(gps.location.lng(), 6); // LONG    delay(1000);
        sgsm.write(0x1A);
        delay(1000);
      }
    }
}
```

For sending latitude and longitude details to IBM Watson IOT platform

```
#include <WiFi.h>
#include <PubSubClient.h>
```

```
#include <ArduinoJson.h>

WiFiClient wifiClient;

#define ORG "mz6rat"
#define DEVICE_TYPE "ESP8266"
#define DEVICE_ID "12345"
#define TOKEN "123456789"
#define speed 0.034

char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/Data/fmt/json";
char topic[] = "iot-2/cmd/home/fmt/String";
char authMethod[] = "use-token-auth";
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
PubSubClient client(server, 1883, wifiClient);
void publishData();

const int trigpin=5;
const int echopin=18;
String command;
String data="";
String lat="13.356563";
String lon="80.141428";
String name="point1";
String icon="fa-fire";

long duration;
int dist;

void setup()
{
  Serial.begin(115200);
  pinMode(trigpin, OUTPUT);
  pinMode(echopin, INPUT);
  wifiConnect();
  mqttConnect();
}

void loop() {

  publishData();
  delay(500);

  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(1000);
    }
    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 publishData()
{
  digitalWrite(trigpin,LOW);
  digitalWrite(trigpin,HIGH);
  delayMicroseconds(10);
  digitalWrite(trigpin,LOW);
  duration=pulseIn(echopin,HIGH);
  dist=duration*speed/2;
  dist=dist/4;
  dist=100-dist;
  if(dist>80){
    lat="13.356563";
    lon="80.141428";
  }else{
    lat="0.000000";
    lon="0.000000";
  }
  DynamicJsonDocument doc(1024);
  String payload;
  doc["Name"]=name;
  doc["Latitude"]=lat;
  doc["Longitude"]=lon;
  doc["Icon"]=icon;
  doc["GasPercent"]=dist;
  serializeJson(doc, payload);
  delay(3000);
  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");
  }
}
```