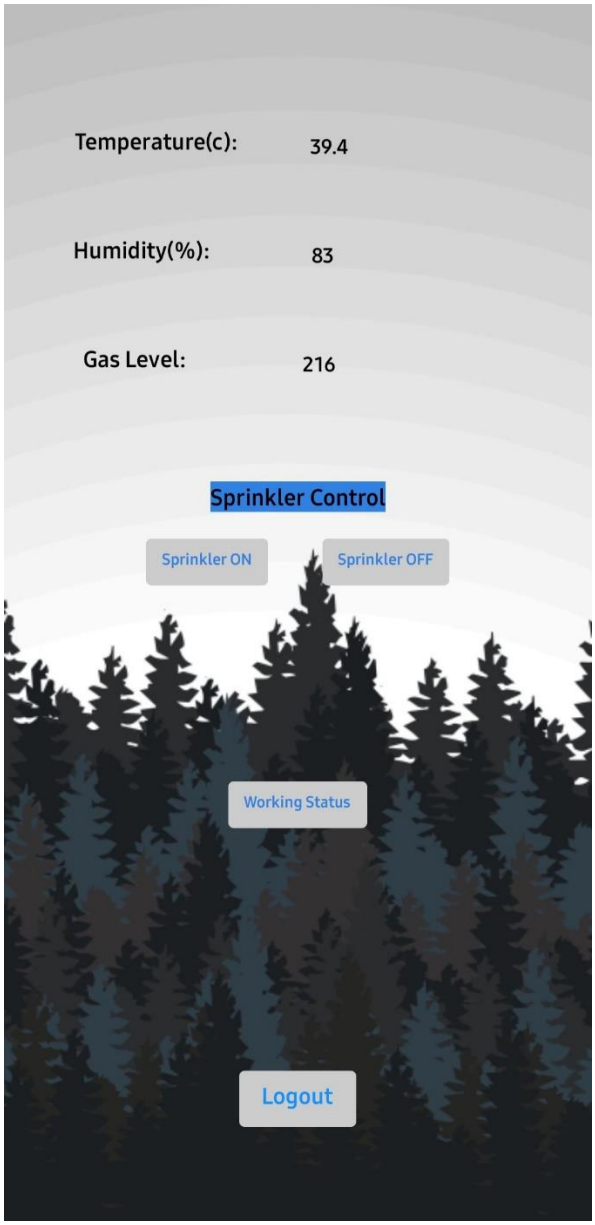
PROJECT DEVELOPMENT PHASE

SPRINT-3

Team ID	PNT2022TMID32813
Project Name	Project - INDUSTRY-SPECIFIC INTELLIGENCE FIRE MANAGEMENT SYSTEM

USER STORY :

AS a user, I can get temperature, humidity and gas level parameters values and get alert messages.



PYTHON:

USECASE: RANDOM VALUES OF TEMPERATUR ,HUMIDITY ,GAS AND SYSTEM CONDITION GENERATED FROM PYTHON

ibm with fast2sms.py - C:/Users/acer/AppData/Local/Programs/Python/Python37/ibm with fast2sms.py (3.7.0)

File Edit Format Run Options Window Help

```
import wiotp.sdk.device
import time
import random
import requests
myConfig = {
    "identity": {
        "orgId": "ls2adz",
        "typeId": "ardiuno",
        "deviceId":"0910"
    },
    "auth": {
        "token": "12345678"
    }
}

def myCommandCallback(cmd):
    print("Message received from IBM IoT Platform: %s" % cmd.data['command'])
    m=cmd.data['command']

client = wiotp.sdk.device.DeviceClient(config=myConfig, logHandlers=None)
client.connect()
count=0
while True:

    s1=['Good','Need maintannce','bad condition','Needs checking']
    s2=['Good','Need maintanence','bad condition','Low water level','No water']
    s3=['Good','No electricity','bad condition','Needs checking']
    s4=['Good','Not working','bad condition','Needs checking']
    random.shuffle(s1)
    random.shuffle(s2)
    random.shuffle(s3)
    random.shuffle(s4)

    temp=random.randint(-40,84)
    humid=random.randint(0,100)
    gas=random.randint(0,100)

    if(temp>68 and gas>80):
        print("\n")
        myData={'temp':str(temp)+chr(176)+"C", 'humid':str(humid)+" %", 'gas':str(gas)+" %", 'sensors':str(s1[0]), 'sprinklers':str(s2[0]), 'exhaust':str(s3[0]), 'alarm':str(s4[0]), 'condit
        message="ALERT MESSAGE FROM FIRE MANAGEMENT SYSTEM:\n\n"+'Temperature:'+str(temp)+" C"+'\nHumidity:'+str(humid)+" %"+'\nGas-level:'+str(gas)+" %"+"\nCondition:Turn On Harzard-Protec
        url = " https://www.fast2sms.com/dev/bulkV2?authorization=oxXvdFwBIzPDuOfpJnALG0VhUkj2YSQN6cTRie8qtErglbK49ltgWTBzkZclr4mPLwOp2nfEKDqoFAGH&route=q&message="+message+"%0A%0AHIGH%20TE
        response = requests.request("GET", url)
        print(response.text)
        print(message)
        print("Turn On Harzard-Protection System")
```

Ln: 1 Col: 0

OUTPUT:

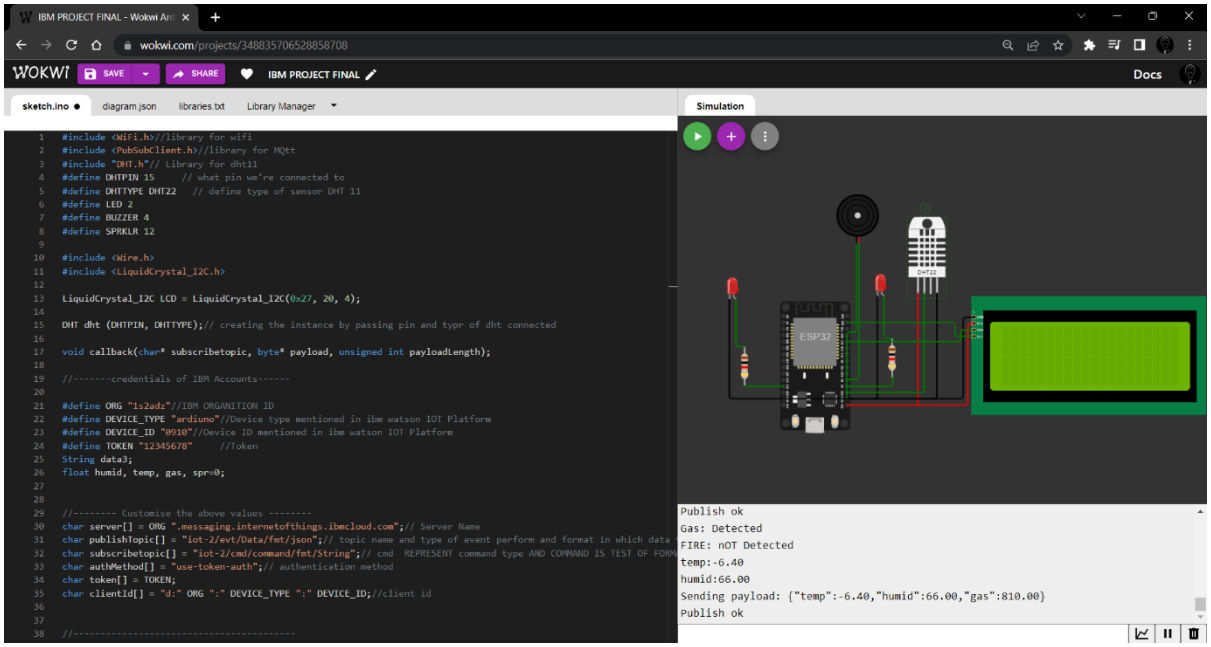
Python 3.7.0 Shell

File Edit Shell Debug Options Window Help

```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
 RESTART: C:/Users/acer/AppData/Local/Programs/Python/Python37/ibm with fast2sms.py

2022-11-20 20:01:39,420  wiotp.sdk.device.client.DeviceClient  INFO    Connected successfully: d:ls2adz:ardiuno:0910

ALERT MESSAGE FROM FIRE MANAGEMENT SYSTEM:

Temperature:-16 C
Humidity:60 %
Gas-level:14 %
Condition:SAFE

Sensors:Need maintannce
Sprinklers:bad condition
Exhaust:bad condition
Alarm:Good

SAFE
Published data Successfully: %s {'temp': '-16°C', 'humid': '60 %', 'gas': '14 %', 'sensors': 'Need maintannce', 'sprinklers': 'bad condition', 'exhaust': 'bad condition', 'alarm': 'Good',
'condition': 'SAFE'}


ALERT MESSAGE FROM FIRE MANAGEMENT SYSTEM:

Temperature:-31 C
Humidity:69 %
Gas-level:33 %
Condition:SAFE

Sensors:Need maintannce
Sprinklers:No water
Exhaust:Needs checking
Alarm:Good

SAFE
Published data Successfully: %s {'temp': '-31°C', 'humid': '69 %', 'gas': '33 %', 'sensors': 'Need maintannce', 'sprinklers': 'No water', 'exhaust': 'Needs checking', 'alarm': 'Good', 'con
dition': 'SAFE'}


ALERT MESSAGE FROM FIRE MANAGEMENT SYSTEM:

Temperature:10 C
Humidity:66 %
Gas-level:29 %
Condition:SAFE
```

Ln: 54 Col: 0

WOKWI :



Sketch.ino :

```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt
#include "DHT.h"// Library for dht11
#define DHTPIN 15     // what pin we're connected to
#define DHTTYPE DHT22   // define type of sensor DHT 11
#define LED 2
#define BUZZER 4
#define SPRKLR 12

#include <Wire.h>
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C LCD = LiquidCrystal_I2C(0x27, 20, 4);

DHT dht (DHTPIN, DHTTYPE);// creating the instance by passing pin and typr of
dht connected

void callback(char* subscribetopic, byte* payload, unsigned int
payloadLength);

//-------credentials of IBM Accounts------

#define ORG "1s2adz"//IBM ORGANITION ID
#define DEVICE_TYPE "ardiuno"//Device type mentioned in ibm watson IOT
Platform
```

```
#define DEVICE_ID "0910"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "12345678"     //Token
String data3;
float humid, temp, gas, spr=0;


//-------- Customise the above values --------
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of
event perform and format in which data to be send
char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd  REPRESENT
command type AND COMMAND IS TEST OF FORMAT STRING
char authMethod[] = "use-token-auth";// authentication method
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id


//-----------------------------------------
WiFiClient wifiClient; // creating the instance for wificlient
PubSubClient client(server, 1883, callback ,wifiClient); //calling the
predefined client id by passing parameter like server id,portand
wificredential


void setup()// configureing the ESP32
{
  Serial.begin(115200);
  dht.begin();
  pinMode(LED,OUTPUT);
  pinMode(BUZZER,OUTPUT);
  digitalWrite(LED,LOW);
  digitalWrite(BUZZER,LOW);
  pinMode(SPRKLR,OUTPUT);
  digitalWrite(SPRKLR,LOW);
  delay(10);
  Serial.println();
  wificonnect();
  mqttconnect();
  LCD.init();
  LCD.backlight();
  LCD.setCursor(0, 0);
  LCD.print("Connecting to ");
  LCD.setCursor(0, 1);
  LCD.print("WiFi ");
  delay(1000);
  LCD.clear();
}

void loop()// Recursive Function
```

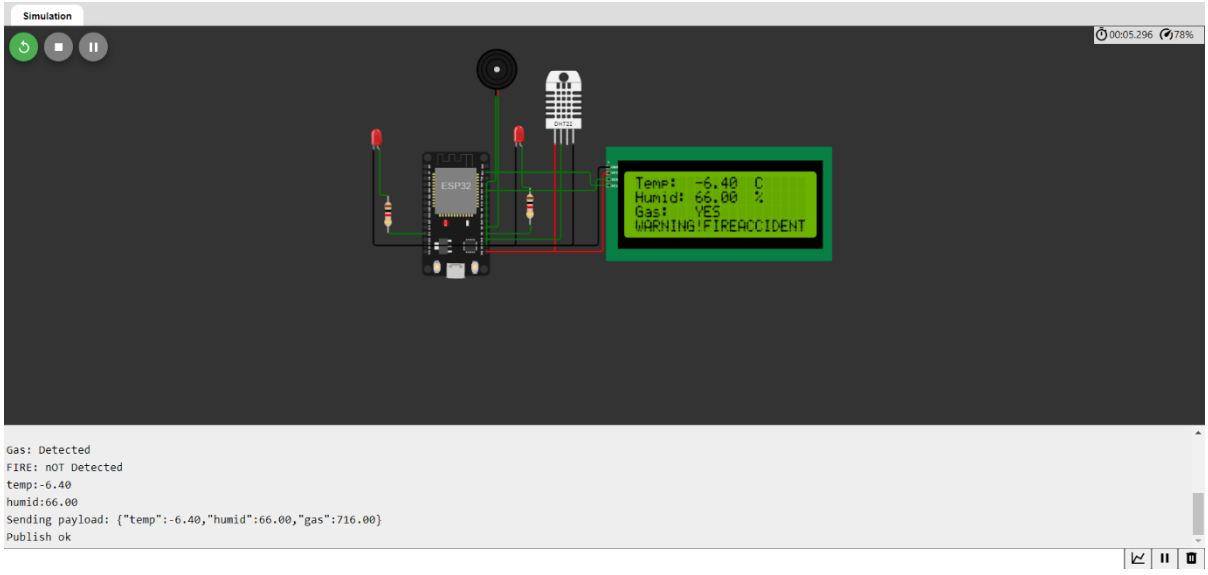
```
{
    LCD.setCursor(0,2);
    LCD.print("Gas: ");
    LCD.setCursor(0, 0);
    LCD.print("Temp: ");
    LCD.setCursor(14, 0);
    LCD.print("C");
    LCD.setCursor(0, 1);
    LCD.print("Humid: ");
    LCD.setCursor(14, 1);
    LCD.print("%");
  humid = dht.readHumidity();
  temp = dht.readTemperature();
  gas = random(0,900);

  if (gas>400)
  {
    Serial.print("Gas: ");
    Serial.println("Detected");
    digitalWrite(BUZZER,HIGH);
    LCD.setCursor(7, 2);
    LCD.print("YES");
    LCD.setCursor(0, 3);
    LCD.print("WARNING!FIREACCIDENT");
    digitalWrite(SPRKLR,HIGH);
    digitalWrite(SPRKLR,HIGH);
  }
  else{
      Serial.print("Gas: ");
      Serial.println("Not Detected");
      digitalWrite(BUZZER,LOW);
      LCD.setCursor(7, 2);
      LCD.print(" NO");
      LCD.setCursor(0, 3);
      LCD.print("                    ");
      digitalWrite(SPRKLR,LOW);
      spr=0;
    }
  if(temp>60)
  {
      Serial.print("FIRE: ");
      Serial.println("Detected");
      spr=1;
      digitalWrite(SPRKLR,HIGH);
      spr=1;
  }
  else
  {
```

```
      Serial.print("FIRE: ");
      Serial.println("nOT Detected");
      spr=0;
      digitalWrite(SPRKLR,LOW);
      spr=0;
  }
  Serial.print("temp:");
  Serial.println(temp);
  LCD.setCursor(7, 0);
  LCD.print(temp);
  Serial.print("humid:");
  Serial.println(humid);
  LCD.setCursor(7, 1);
  LCD.print(humid);

  PublishData(temp, humid, gas);
  delay(1000);
  if (!client.loop()) {
    mqttconnect();
  }
}



/*.....................................retrieving to
Cloud.............................*/

void PublishData(float temp, float humid, float gas) {
  mqttconnect();//function call for connecting to ibm
  /*
     creating the String in in form JSon to update the data to ibm cloud
  */
  String payload = "{\"temp\":";
  payload += temp;
  payload += "," "\"humid\":";
  payload += humid;
  payload += "," "\"gas\":";
  payload += gas;
  payload += "}";


  Serial.print("Sending payload: ");
  Serial.println(payload);


  if (client.publish(publishTopic, (char*) payload.c_str())) {
    Serial.println("Publish ok");// if it sucessfully upload data on the cloud
then it will print publish ok in Serial monitor or else it will print publish
failed
```

```
  } else {
    Serial.println("Publish failed");
  }

}


void mqttconnect() {
  if (!client.connected()) {
    Serial.print("Reconnecting client to ");
    Serial.println(server);
    while (!!!client.connect(clientId, authMethod, token)) {
      Serial.print(".");
      delay(500);
    }

     initManagedDevice();
     Serial.println();
  }
}
void wificonnect() //function defination for wificonnect
{
  Serial.println();
  Serial.print("Connecting to ");

  WiFi.begin("Wokwi-GUEST", "", 6);//passing the wifi credentials to establish
the connection
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
  LCD.setCursor(0, 0);
  LCD.print("Connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}

void initManagedDevice() {
  if (client.subscribe(subscribetopic)) {
    Serial.println((subscribetopic));
    Serial.println("subscribe to cmd OK");
  } else {
    Serial.println("subscribe to cmd FAILED");
  }
}

void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
```

```
{

  Serial.print("callback invoked for topic: ");
  Serial.println(subscribetopic);
  for (int i = 0; i < payloadLength; i++) {
    //Serial.print((char)payload[i]);
    data3 += (char)payload[i];
  }
  Serial.println("data: "+ data3);
  if(data3=="sprinkleron")
  {
Serial.println(data3);
digitalWrite(LED,HIGH);
  }
  else
  {
Serial.println(data3);
digitalWrite(LED,LOW);
  }
data3="";
}
```

OUTPUT:



LINK: https://wokwi.com/projects/348835706528858708