

SPRINT 4

Framework (Cloud deployment)

Date	15 November 2022
Team ID	PNT2022TMID27080
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.

5.5 Receiving commands in IBM cloud using Python program:

#IBM Watson IOT Platform

#pip install wiotp-sdk import

wiotp.sdk.device import

time

import random

#Provide your IBM Watson Device Credentials myConfig

```
= {  
    "identity": {  
        "orgId": "q26y5w",  
        "typeId": "TestDeviceType",  
        "deviceId": "2022"  
    },  
    "auth": {  
        "token": "uu(rZQw9292EfSFGDX"  
    }  
}
```

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()
```

```
#Conditions while True:
```

```
temp=random.randint(0,125)
```

```
hum=random.randint(0,100)
```

```
pre=random.randint(0,100)
```

```
haz=random.randint(0,100)
```

```
myData={'Temperature':temp,
```

```
        'Humidity':hum,
```

```
        'Pressure':pre,
```

```
        'HazardousGas':haz
```

```
    }
```

```
    client.publishEvent(eventId="status", msgFormat="json", data=myData, qos=0, onPublish=None)
```

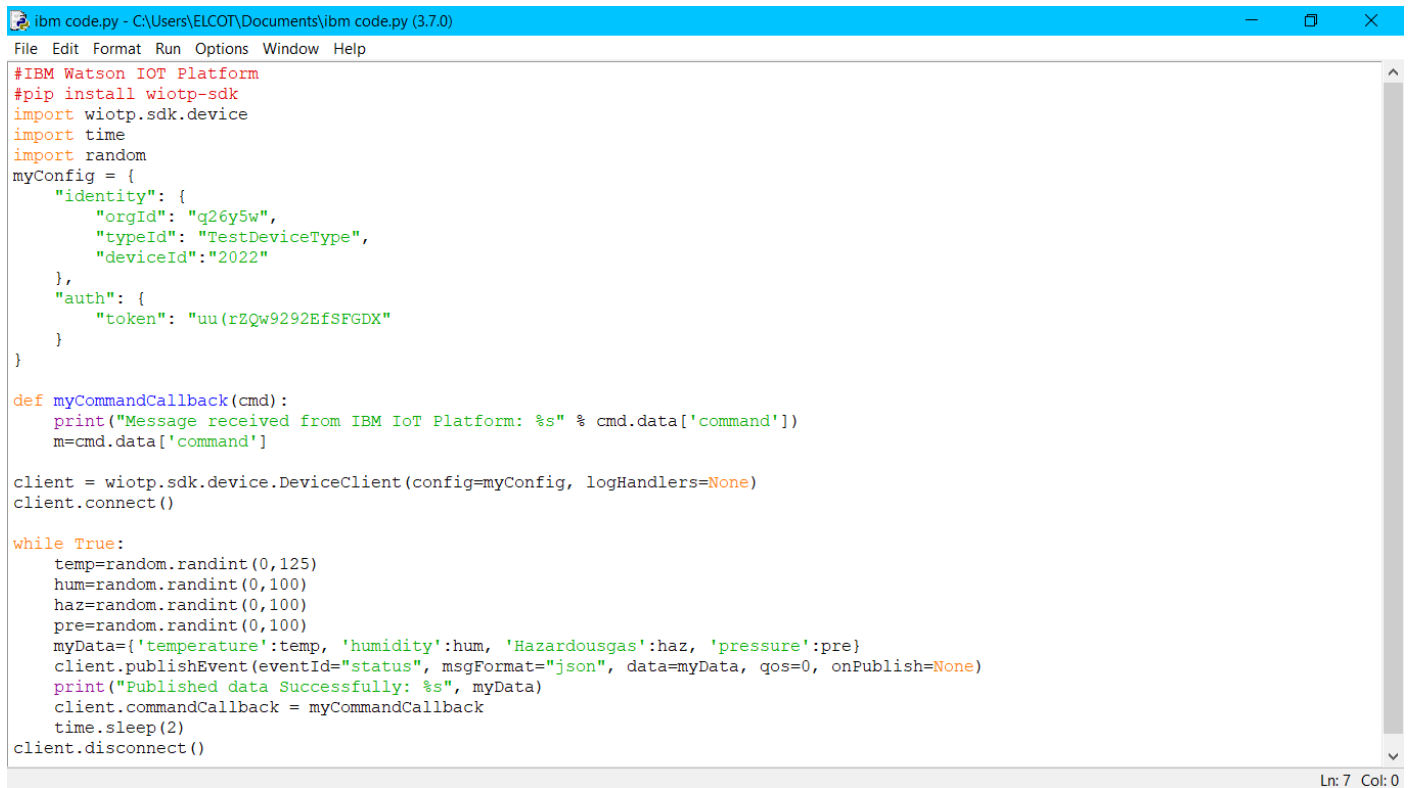
```
    print("Published data Successfully: %s", myData)    if(haz>90):    print("Exhaust Fan is ON")
```

```
    else:
```

```
        print("Exhaust Fan is OFF")
```

```
    client.commandCallback = myCommandCallback
```

```
    time.sleep(2) client.disconnect()
```



```
ibm code.py - C:\Users\ELCOT\Documents\ibm code.py (3.7.0)
File Edit Format Run Options Window Help
#IBM Watson IoT Platform
#pip install wiotp-sdk
import wiotp.sdk.device
import time
import random
myConfig = {
    "identity": {
        "orgId": "g26y5w",
        "typeId": "TestDeviceType",
        "deviceId": "2022"
    },
    "auth": {
        "token": "uu(rZQw9292EfSFGDX"
    }
}

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()

while True:
    temp=random.randint(0,125)
    hum=random.randint(0,100)
    haz=random.randint(0,100)
    pre=random.randint(0,100)
    myData={'temperature':temp, 'humidity':hum, 'Hazardousgas':haz, 'pressure':pre}
    client.publishEvent(eventId="status", msgFormat="json", data=myData, qos=0, onPublish=None)
    print("Published data Successfully: %s", myData)
    client.commandCallback = myCommandCallback
    time.sleep(2)
client.disconnect()
Ln: 7 Col: 0
```

```
*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\ELCOT\Documents\ibm code.py =====
2022-11-18 14:13:53,161 wiotp.sdk.device.client.DeviceClient INFO Connected successfully: d:q26y5w:TestDeviceType:2022Published
data Successfully: %s
{'temperature': 6, 'humidity': 52, 'Hazardousgas': 16, 'pressure': 81}
Published data Successfully: %s {'temperature': 76, 'humidity': 28, 'Hazardousgas': 41, 'pressure': 51}
Published data Successfully: %s {'temperature': 57, 'humidity': 3, 'Hazardousgas': 4, 'pressure': 26}
Published data Successfully: %s {'temperature': 125, 'humidity': 86, 'Hazardousgas': 39, 'pressure': 100}
Published data Successfully: %s {'temperature': 116, 'humidity': 26, 'Hazardousgas': 9, 'pressure': 95}
Published data Successfully: %s {'temperature': 109, 'humidity': 62, 'Hazardousgas': 49, 'pressure': 69}
Published data Successfully: %s {'temperature': 15, 'humidity': 35, 'Hazardousgas': 46, 'pressure': 91}
Published data Successfully: %s {'temperature': 32, 'humidity': 21, 'Hazardousgas': 26, 'pressure': 23}
```

6.Observations & Results:

```
*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\ELCOT\Documents\ibm code.py =====
2022-11-18 14:13:53,161 wiotp.sdk.device.client.DeviceClient INFO Connected successfully: d:q26y5w:TestDeviceType:2022Published
data Successfully: %s
{'temperature': 6, 'humidity': 52, 'Hazardousgas': 16, 'pressure': 81}
Published data Successfully: %s {'temperature': 76, 'humidity': 28, 'Hazardousgas': 41, 'pressure': 51}
Published data Successfully: %s {'temperature': 57, 'humidity': 3, 'Hazardousgas': 4, 'pressure': 26}
Published data Successfully: %s {'temperature': 125, 'humidity': 86, 'Hazardousgas': 39, 'pressure': 100}
Published data Successfully: %s {'temperature': 116, 'humidity': 26, 'Hazardousgas': 9, 'pressure': 95}
Published data Successfully: %s {'temperature': 109, 'humidity': 62, 'Hazardousgas': 49, 'pressure': 69}
Published data Successfully: %s {'temperature': 15, 'humidity': 35, 'Hazardousgas': 46, 'pressure': 91}
Published data Successfully: %s {'temperature': 32, 'humidity': 21, 'Hazardousgas': 26, 'pressure': 23}
```

IBM x IBM x Node x Deve x https x IBM- x IBM- x Node x Reso x +

q26y5w.internetofthings.ibmcloud.com/dashboard/devices/browse

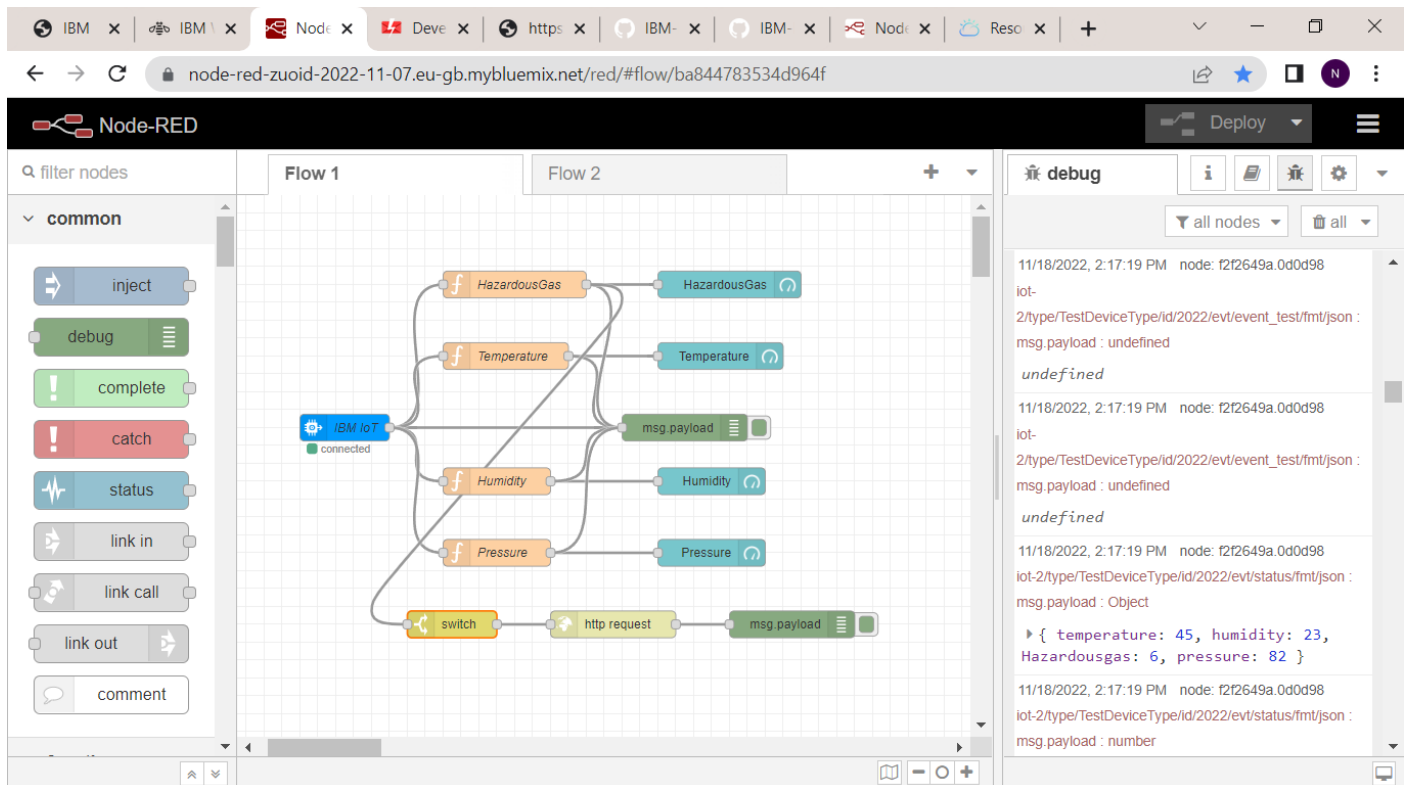
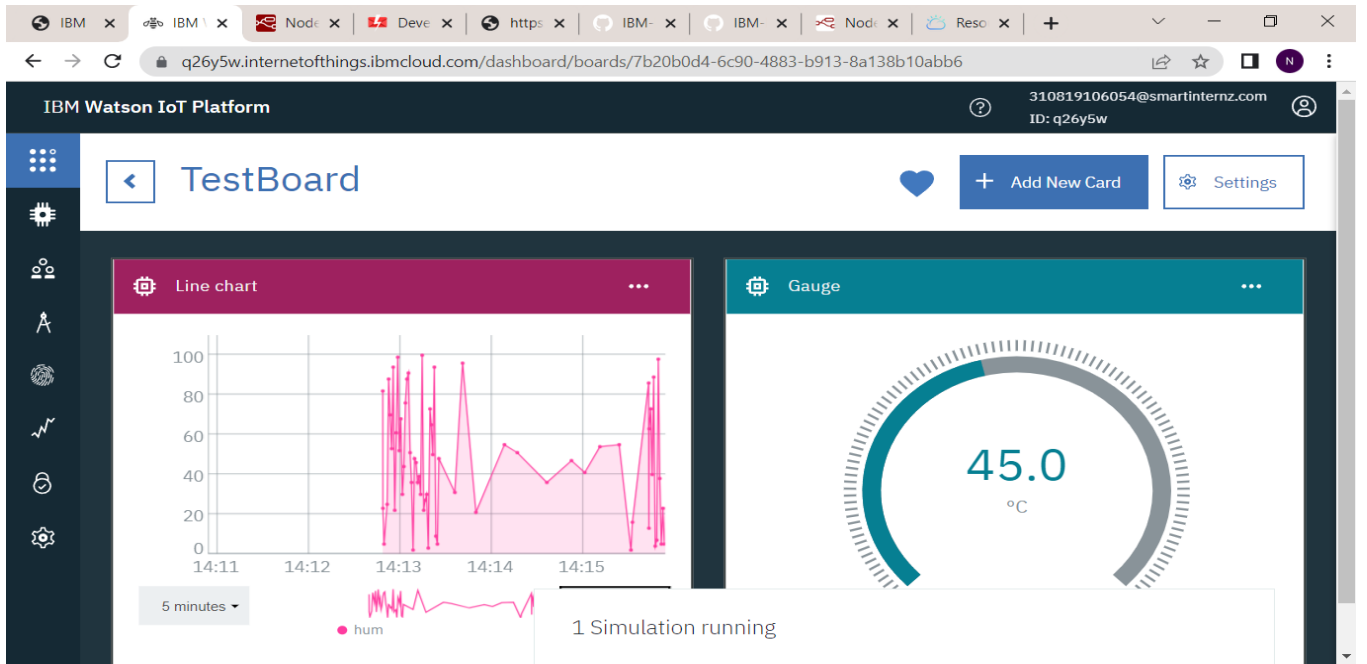
IBM Watson IoT Platform 310819106054@smartinternz.com ID: q26y5w

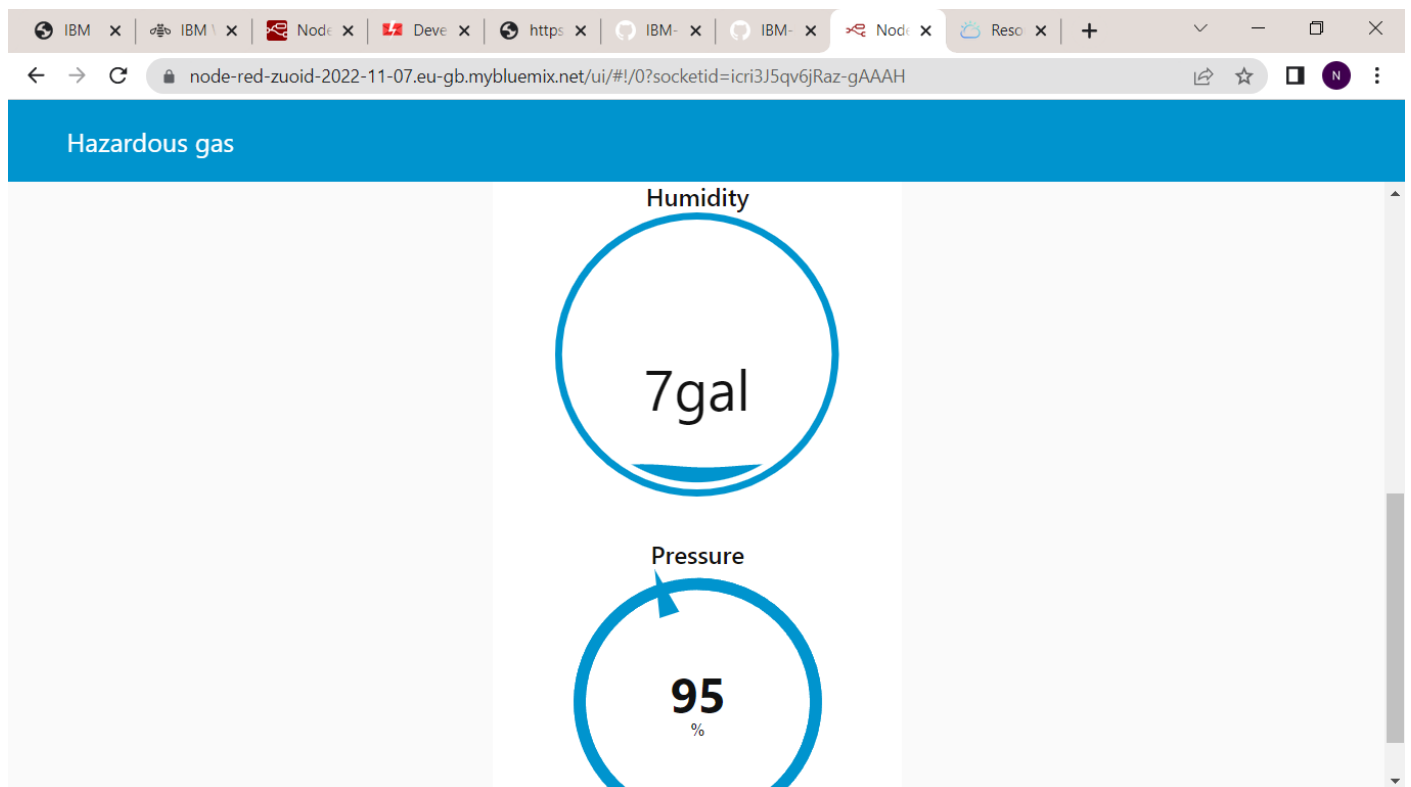
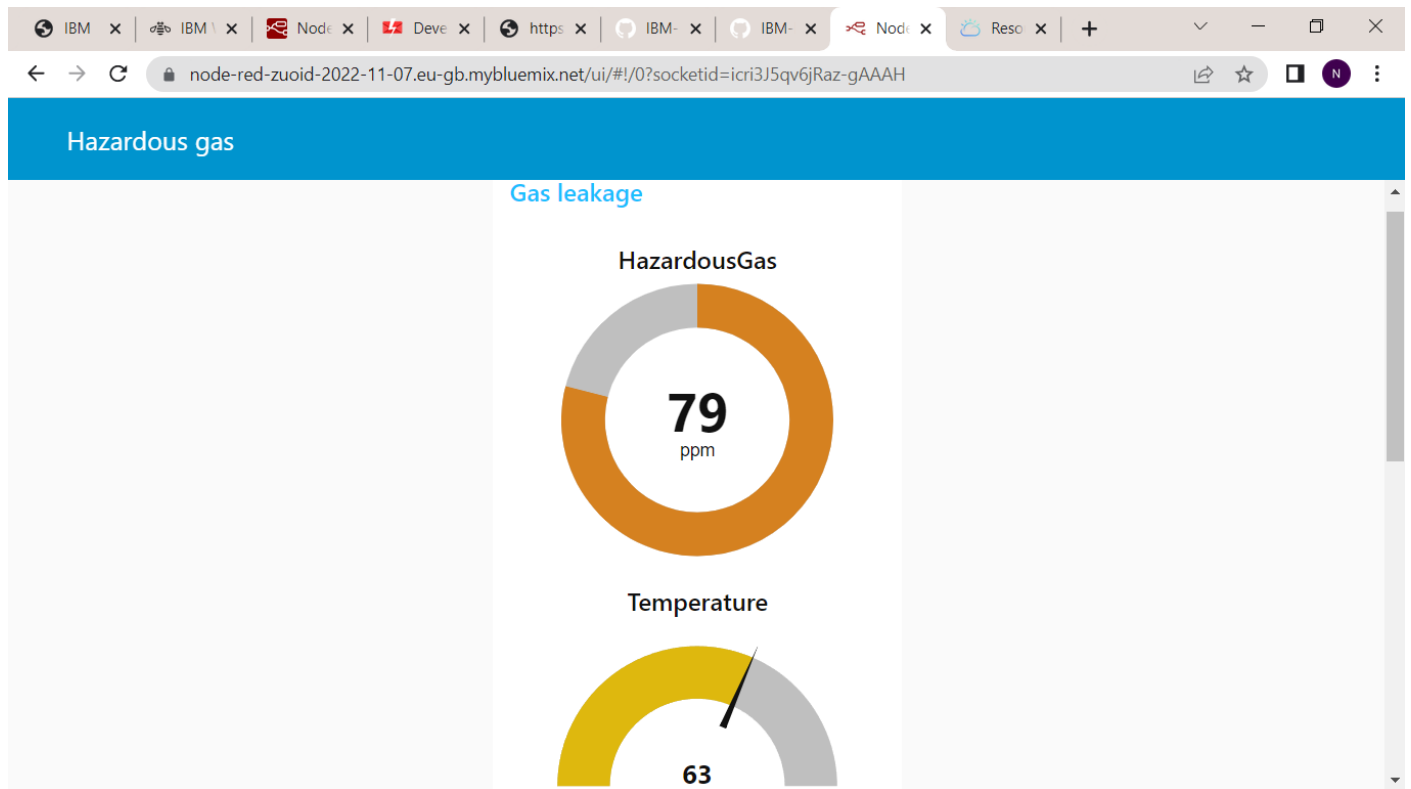
Browse Action Device Types Interfaces Add Device +

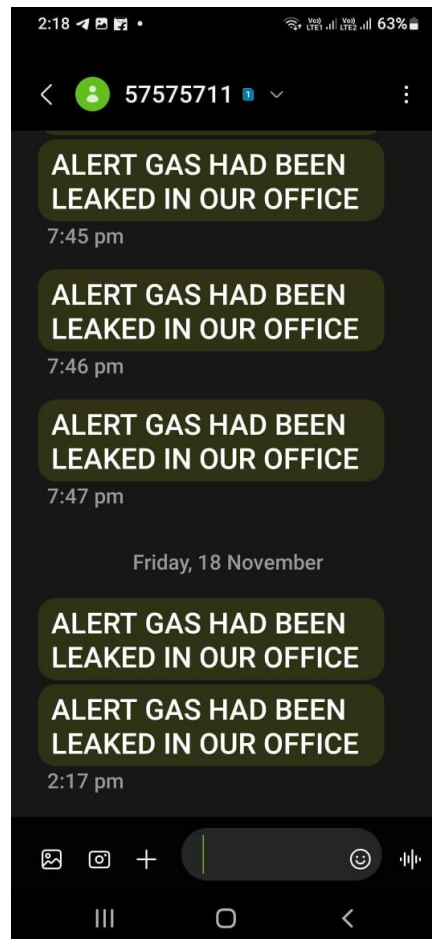
The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
status	{"temperature":73,"humidity":66,"Hazardousgas..."}	json	a few seconds ago
event_test	{"temp":50,"hum":79}	json	a few seconds ago
event_test	{"temp":79,"hum":66}	json	a few seconds ago
status	{"temperature":41,"humidity":12,"Hazardousgas..."}	json	a few seconds ago
event_test	{"temp":90,"hum":54}	json	a few seconds ago

1 Simulation running







7. Advantages & Disadvantages:

Advantages:

- Alert us when Hazardous gas levels increases in the atmosphere.
- Prevent fire and explosions, avoiding huge losses.
- Supervise gas concentration levels.
- Ensure worker's health.
- Real-time updates about leakage.
- Cost-effective installation.
- Data analytics for improved decisions.
- Measure oxygen level accuracy.

Disadvantages:

- Only one gas can be measured with each instrument.
- Poor stability leads to greater environmental impact.
- When heavy dust, steam or fog blocks the input of the sensor.

8. Conclusion and Future Work:

In this paper we use IOT technology for enhancing the existing safety standards. While making this prototype has been to bring a revolution in the field of safety against the leakage of harmful and toxic

gases in environment and hence nullify any major or minor hazard being caused due to them. We have used the IOT technology to make a Gas Leakage Detector for society which having Smart Alerting techniques involving sending text message to the concerned authority and an ability performing data analytics on sensor. This system will be able to detect the gas in environment using the gas sensors. This will prevent form the major harmful problem.

9.References:

IBM cloud reference: <https://cloud.ibm.com/>

IoT simulator: <https://watson-iot-sensor-simulator.mybluemix.net/> Fast

2 SMS: <https://www.fast2sms.com/>