

## SPRINT 4

DATE	5/11/2022
TEAM ID	PNT2022TMID22887
PROJECT NAME	GAS LEAKAGE MONITORING AND ALERTING SYSTEM FOR INDUSTRIES
MAXIMUM MARKS	2

### CODE DEVELOPMENT IN PYTHON IDLE:

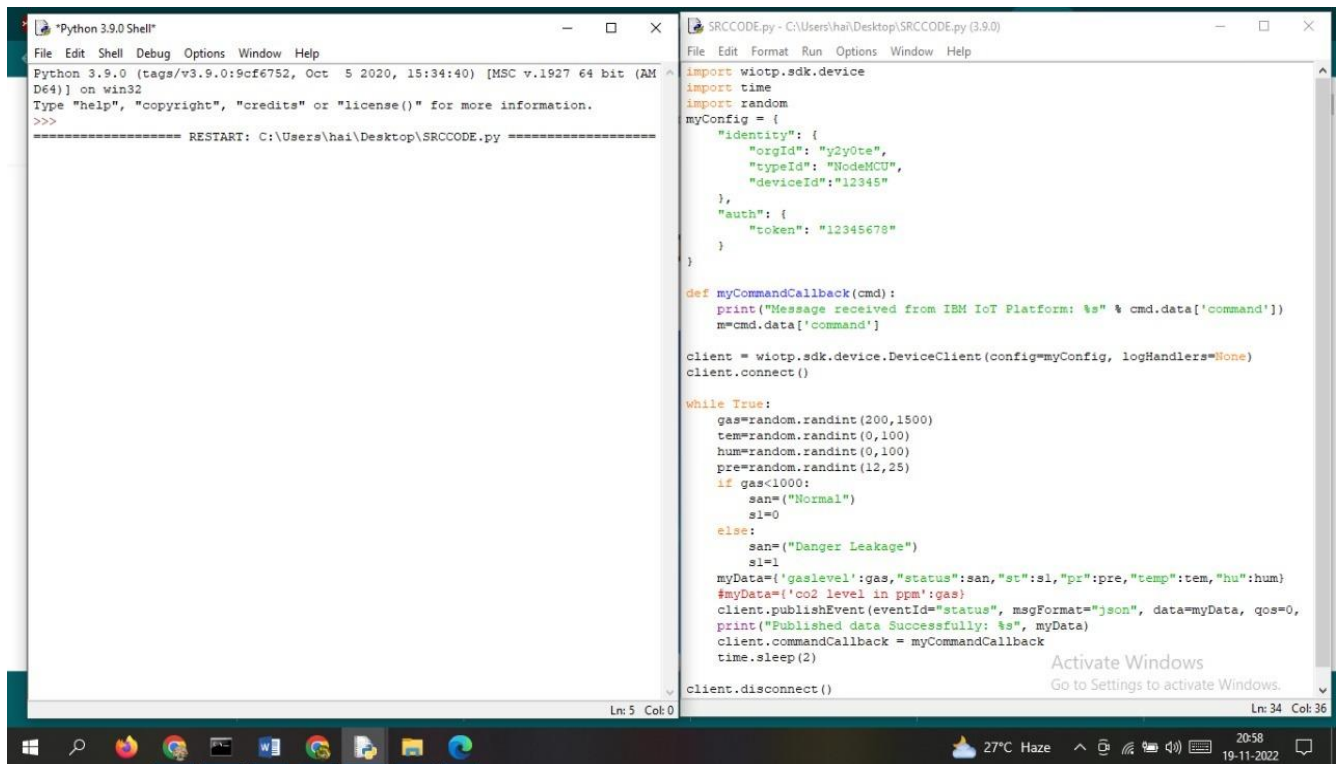
#### PYTHON PROGRAM:

```
import wiotp.sdk.device
import time
import random
myConfig = {
    "identity": {
        "orgId": "y2y0te",
        "typeId": "NodeMCU",
        "deviceId": "12345"
    },
    "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()
```

```
while True:
    gas=random.randint(200,1500)
    tem=random.randint(0,100)
    hum=random.randint(0,100)
    pre=random.randint(12,25)
    if gas<1000:
        san=("Normal")
        s1=0
    else:
        san=("Danger Leakage")
        s1=1
    myData={'gaslevel':gas,"status":san,"st":s1,"pr":pre,"temp":tem,"hu":hum}
    #myData={'co2 level in ppm':gas}
    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()
```

## CONNECTING THE DEVICE AND CREATING DEVICE ID IN IBM WATSON PLATFORM :



The image shows a Windows desktop with two application windows. The left window is a "Python 3.9.0 Shell" with a menu bar (File, Edit, Shell, Debug, Options, Window, Help) and a command prompt. It displays the Python version and architecture, and a restart command for a file named SRCCODE.py. The right window is a text editor titled "SRCCODE.py - C:\Users\hai\Desktop\SRCCODE.py (3.9.0)" with a menu bar (File, Edit, Format, Run, Options, Window, Help). It contains a Python script that imports the Watson IoT SDK, defines a configuration object, sets up a command callback, and runs a loop to publish sensor data (gas, temperature, humidity, pressure) and status (Normal or Danger Leakage) to the IBM IoT Platform. The script also includes a sleep function and a disconnect method. The Windows taskbar at the bottom shows the date and time as 19-11-2022, 20:58, and the weather as 27°C Haze.

```
Python 3.9.0 (tags/v3.9.0:9cf6752, Oct 5 2020, 15:34:40) [MSC v.1927 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\hai\Desktop\SRCCODE.py =====

import wiotp.sdk.device
import time
import random

myConfig = {
    "identity": {
        "orgId": "y2y0te",
        "typeId": "NodeMCU",
        "deviceId": "12345"
    },
    "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()

while True:
    gas=random.randint(200,1500)
    tem=random.randint(0,100)
    hum=random.randint(0,100)
    pre=random.randint(12,25)
    if gas<1000:
        san= ("Normal")
        sl=0
    else:
        san= ("Danger Leakage")
        sl=1
    myData={'gaslevel':gas,"status":san,"st":sl,"pr":pre,"temp":tem,"hu":hum}
    #myData={'co2 level in ppm':gas}
    client.publishEvent(eventId="status", msgFormat="json", data=myData, qos=0,
    print("Published data Successfully: %s", myData)
    client.commandCallback = myCommandCallback
    time.sleep(2)

client.disconnect()
```