# PYTHON SCRIPT FOR GAS LEAKAGE DETECTION AND MONITORING SYSTEM

| Date | 19 Nov 2022 |
|---|---|
| Team ID | PNT2022TMID29935 |
| Project Name | Gas leakage detection and monitoring system |

**PYTHON CODE:**

**import time**

**import sys**

**import ibmiotf.application**

**import ibmiotf.device**

**import random**

**#Provide your IBM Watson Device Credentials**

**organization = "x6troc"**

**deviceType = "PNT2022TMID29935"**

**deviceId = "6374679606"**

**authMethod = "token"**

**authToken = "eSJ(wSv_kaOwuZ?yIl"**

**# Initialize GPIO**

```python
def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status=="lighton":
        print ("led is on")
    else :
        print ("led is off")


    #print(cmd)
  try:
deviceOptions = {"org": organization, "type": deviceType, "id": deviceId,
"auth-method": authMethod, "auth-token": authToken}
deviceCli = ibmiotf.device.Client(deviceOptions)
#..........................................


except Exception as e:
print("Caught exception connecting device: %s" % str(e))
sys.exit()


# Connect and send a datapoint "hello" with value "world" into the cloud as
an event of type "greeting" 10 times
deviceCli.connect()


while True:
        #Get Sensor Data from DHT11
```

```python
gas=random.randint(0,200)

    data = { 'gas' : gas  }


    def myOnPublishCallback():

        if gas>100:

            data = { 'gas' : gas }

            print ("Published gas_level = %s ppm" % gas, "//Gas alert!!")


    success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0,
on_publish=myOnPublishCallback)


    if not success:

        print("Not connected to IoTF")

    time.sleep(1)


    deviceCli.commandCallback = myCommandCallback


# Disconnect the device and application from the cloud

deviceCli.disconnect()
```