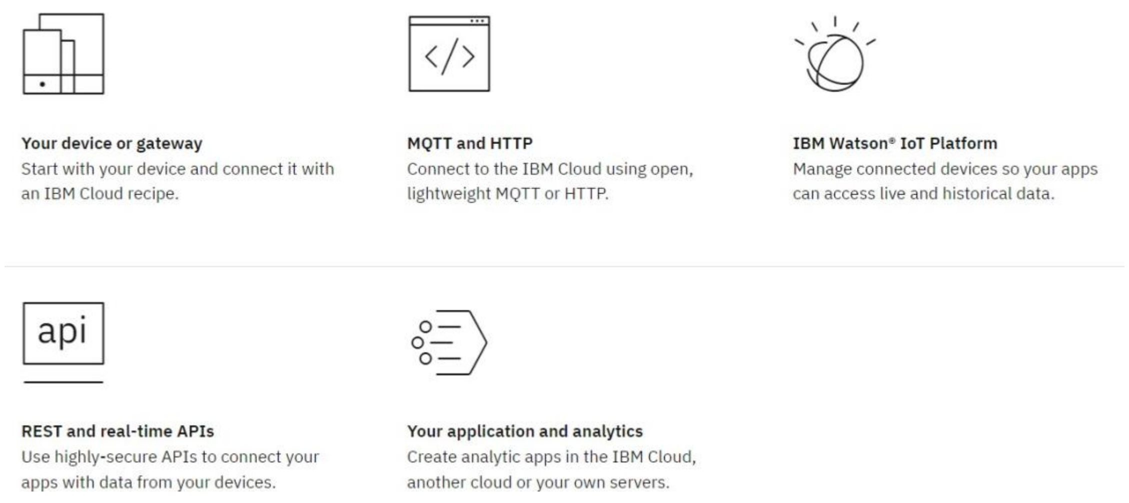
# Sprint- 2

| | |
|---|---|
| **Team ID** | **PNT2022TMID11539** |
| **Project Title** | **Gas Leakage Monitoring And Alerting System** |
| **Date** | **15.11.2022** |

**IBM Watson and Python Integration:**

By using Watson IoT Platform, you can collect connected device data and perform analytics on real-time data. The IBM Watson IoT Platform is a fully managed, Cloud-hosted service that provides device management capabilities as well as data collection and management in a time series format.



**Your device or gateway**
Start with your device and connect it with an IBM Cloud recipe.

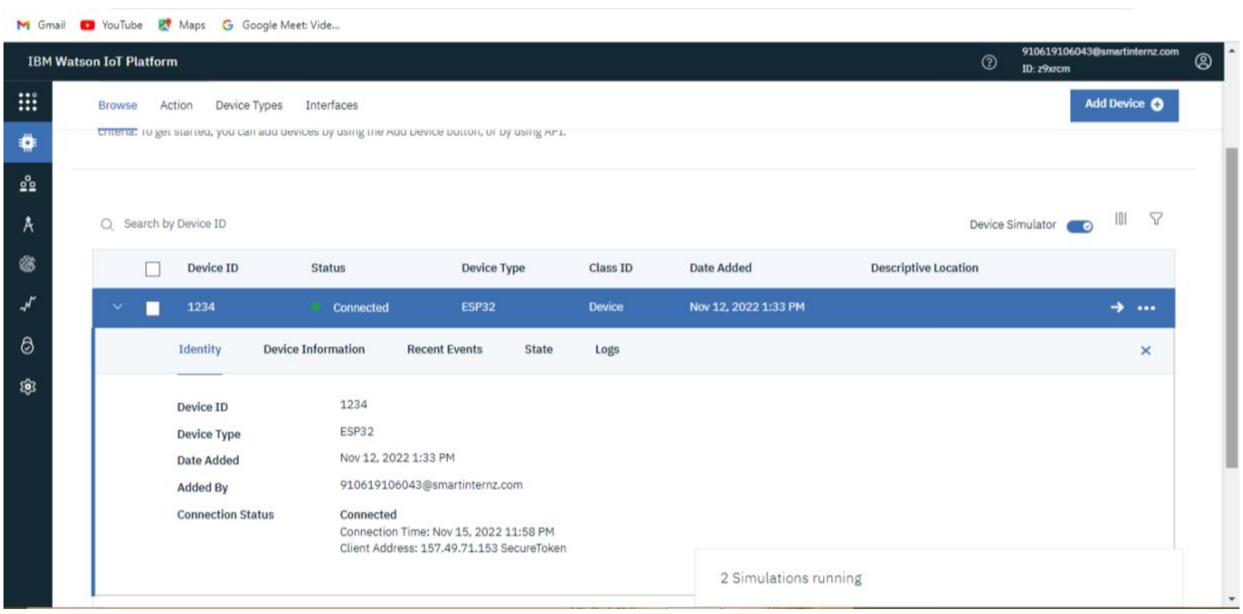
**MQTT and HTTP**
Connect to the IBM Cloud using open, lightweight MQTT or HTTP.

**IBM Watson® IoT Platform**
Manage connected devices so your apps can access live and historical data.

**REST and real-time APIs**
Use highly-secure APIs to connect your apps with data from your devices.

**Your application and analytics**
Create analytic apps in the IBM Cloud, another cloud or your own servers.

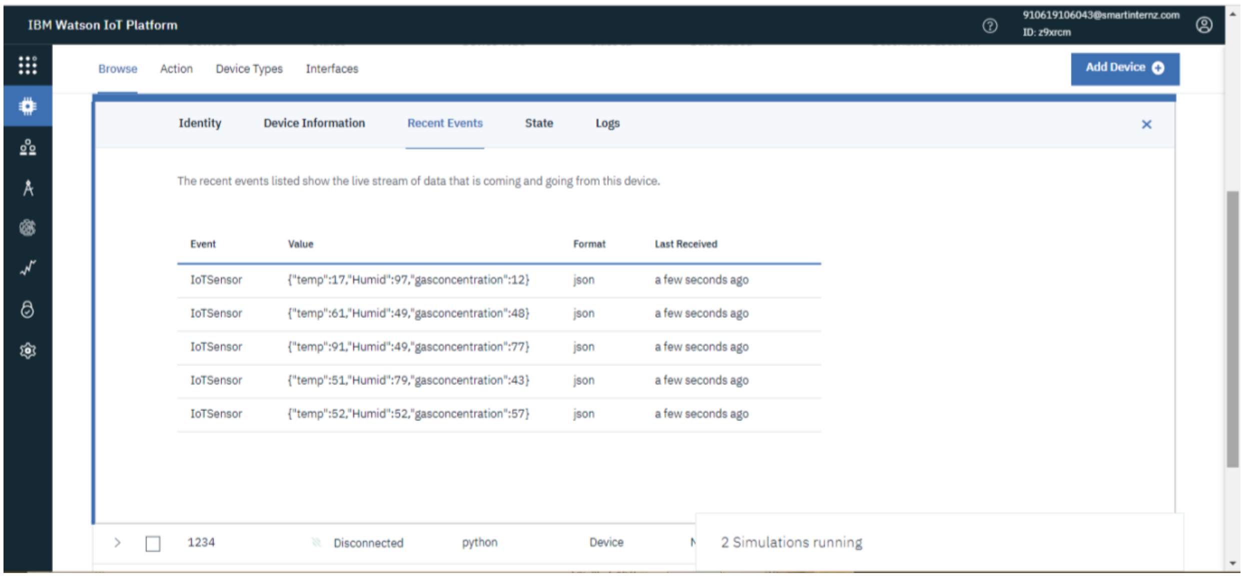
## Using the Device Created in IBM Watson:



criteria: To get started, you can add devices by using the Add Device button, or by using API.

| Device ID | Status | Device Type | Class ID | Date Added | Descriptive Location |
|---|---|---|---|---|---|
| 1234 | Connected | ESP32 | Device | Nov 12, 2022 1:33 PM | |

| | |
|---|---|
| Device ID | 1234 |
| Device Type | ESP32 |
| Date Added | Nov 12, 2022 1:33 PM |
| Added By | 910619106043@smartinternz.com |
| Connection Status | Connected<br>Connection Time: Nov 15, 2022 11:58 PM<br>Client Address: 157.49.71.153 SecureToken |



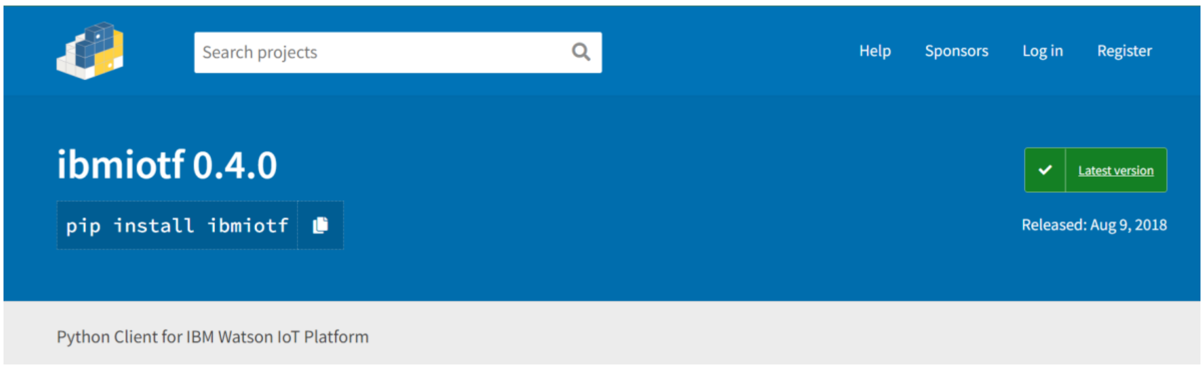
The recent events listed show the live stream of data that is coming and going from this device.

| Event | Value | Format | Last Received |
|---|---|---|---|
| IoTSensor | {"temp":17,"Humid":97,"gasconcentration":12} | json | a few seconds ago |
| IoTSensor | {"temp":61,"Humid":49,"gasconcentration":48} | json | a few seconds ago |
| IoTSensor | {"temp":91,"Humid":49,"gasconcentration":77} | json | a few seconds ago |
| IoTSensor | {"temp":51,"Humid":79,"gasconcentration":43} | json | a few seconds ago |
| IoTSensor | {"temp":52,"Humid":52,"gasconcentration":57} | json | a few seconds ago |

| | | | |
|---|---|---|---|
| 1234 | Disconnected | python | Device |

**Connected** sign shows that it is connected and live

**Python code execution:**



ibmiotf 0.4.0

```
pip install ibmiotf
```

Released: Aug 9, 2018

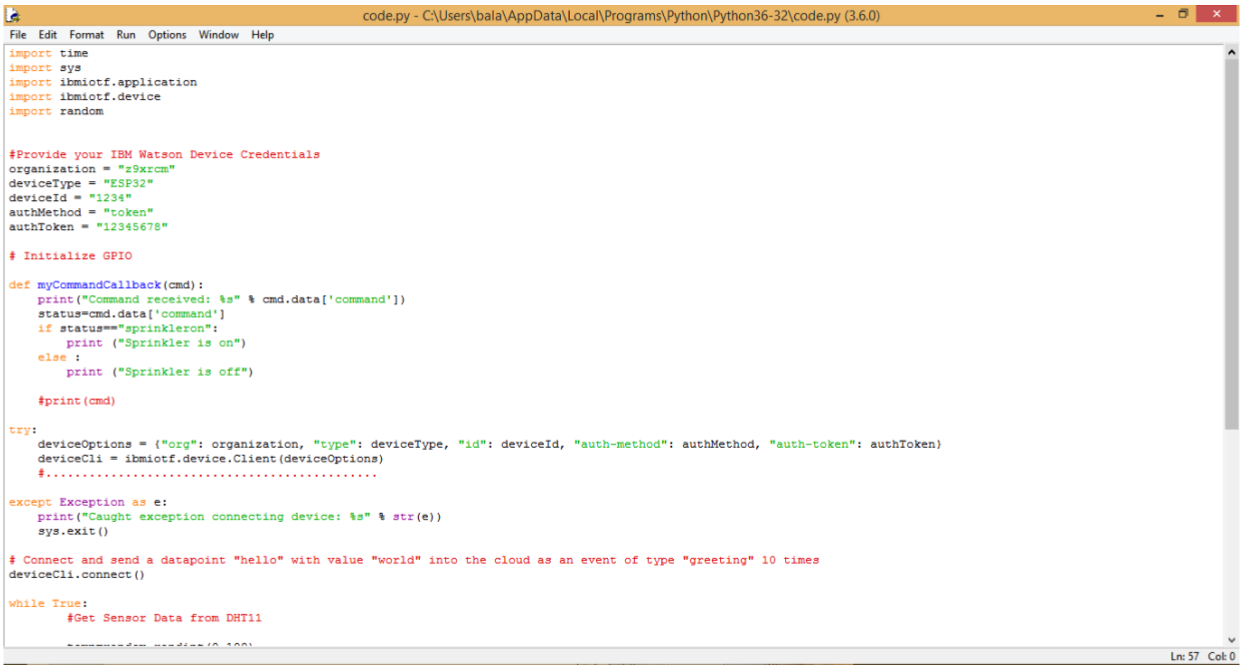
Python Client for IBM Watson IoT Platform

**Install this package :** Python Client for IBM Watson IoT Platform

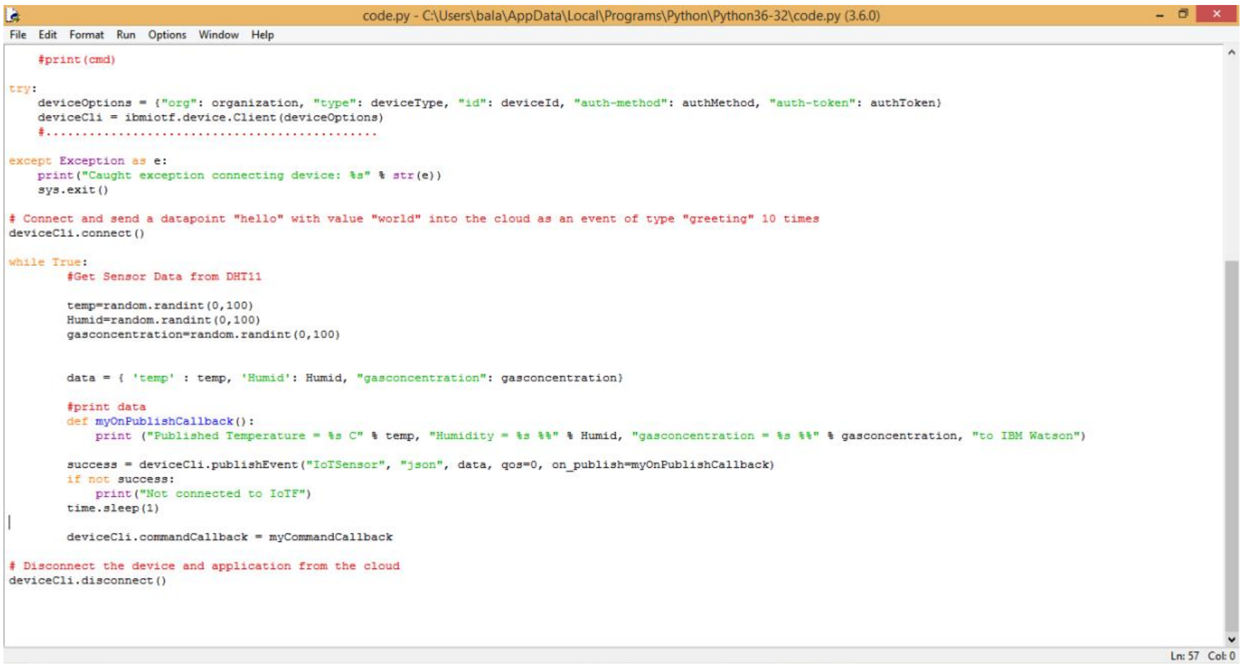
**python  code:**

```
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random


#Provide your IBM Watson Device Credentials
organization = "z9xrcm"
deviceType = "ESP32"
deviceId = "1234"
authMethod = "token"
authToken = "12345678"

# Initialize GPIO

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

    #print(cmd)

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

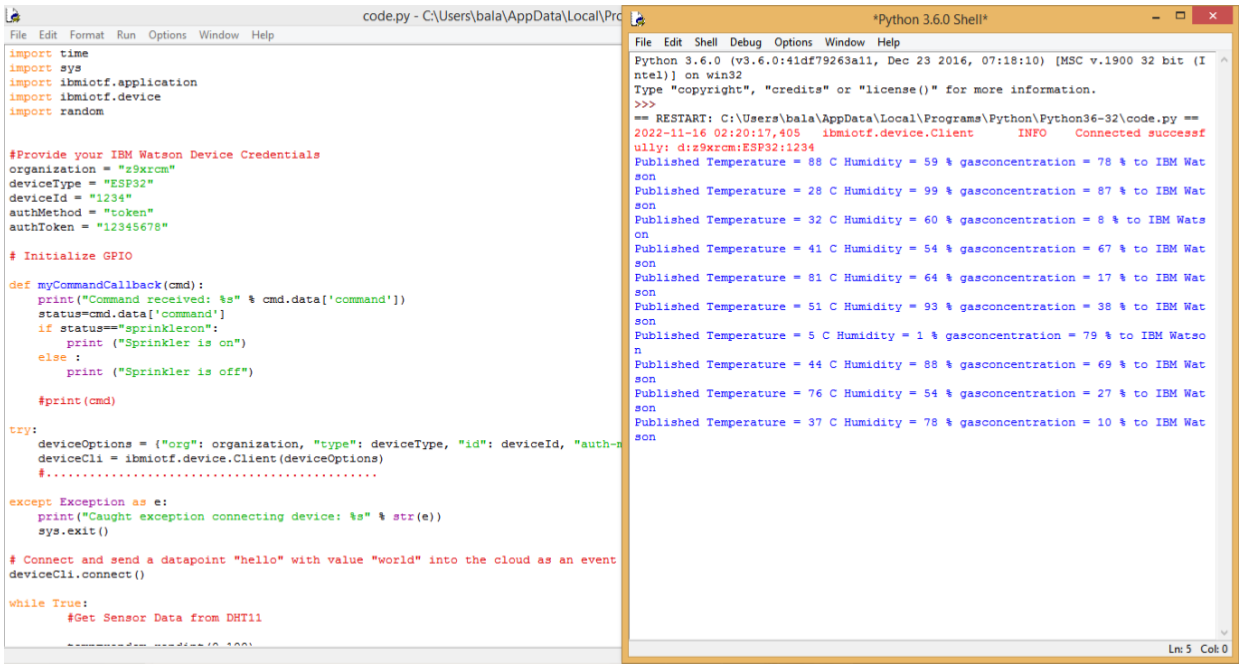
```
   #............................................

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

      temp=random.randint(0,100)
      Humid=random.randint(0,100)
      gasconcentration=random.randint(0,100)


      data = { 'temp' : temp, 'Humid': Humid, "gasconcentration": gasconcentration}

      #print data
      def myOnPublishCallback():
         print ("Published Temperature = %s C" % temp, "Humidity = %s %%" %
Humid, "gasconcentration = %s %%" % gasconcentration, "to IBM Watson")

      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()
```


code.py - C:\Users\bala\AppData\Local\Programs\Python\Python36-32\code.py (3.6.0)
File Edit Format Run Options Window Help
Ln: 57 Col: 0

```python
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random


#Provide your IBM Watson Device Credentials
organization = "z9xrcm"
deviceType = "ESP32"
deviceId = "1234"
authMethod = "token"
authToken = "12345678"

# Initialize GPIO

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


code.py - C:\Users\bala\AppData\Local\Programs\Python\Python36-32\code.py (3.6.0)
File Edit Format Run Options Window Help
Ln: 57 Col: 0

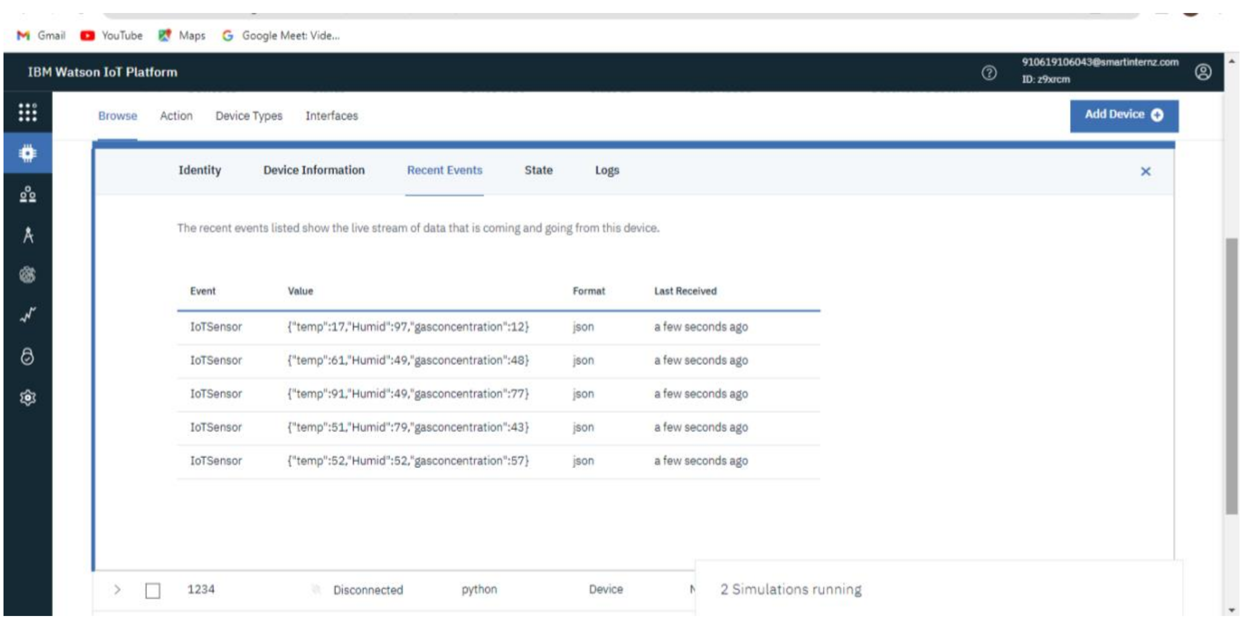
```python
    #print(cmd)

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

        temp=random.randint(0,100)
        Humid=random.randint(0,100)
        gasconcentration=random.randint(0,100)


        data = { 'temp' : temp, 'Humid': Humid, "gasconcentration": gasconcentration}

        #print data
        def myOnPublishCallback():
            print ("Published Temperature = %s C" % temp, "Humidity = %s %%" % Humid, "gasconcentration = %s %%" % gasconcentration, "to IBM Watson")

        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()
```


code.py - C:\Users\bala\AppData\Local\Pro
File Edit Format Run Options Window Help

```python
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random


#Provide your IBM Watson Device Credentials
organization = "z9xrcm"
deviceType = "ESP32"
deviceId = "1234"
authMethod = "token"
authToken = "12345678"

# Initialize GPIO

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

    #print(cmd)

try:
    deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-m
    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
deviceCli.connect()

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

```
Python 3.6.0 (v3.6.0:41df79263a11, Dec 23 2016, 07:18:10) [MSC v.1900 32 bit (I
ntel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:\Users\bala\AppData\Local\Programs\Python\Python36-32\code.py ==
2022-11-16 02:20:17,405   ibmiotf.device.Client      INFO    Connected successf
ully: d:z9xrcm:ESP32:1234
Published Temperature = 88 C Humidity = 59 % gasconcentration = 78 % to IBM Wat
son
Published Temperature = 28 C Humidity = 99 % gasconcentration = 87 % to IBM Wat
son
Published Temperature = 32 C Humidity = 60 % gasconcentration = 8 % to IBM Wats
on
Published Temperature = 41 C Humidity = 54 % gasconcentration = 67 % to IBM Wat
son
Published Temperature = 81 C Humidity = 64 % gasconcentration = 17 % to IBM Wat
son
Published Temperature = 51 C Humidity = 93 % gasconcentration = 38 % to IBM Wat
son
Published Temperature = 5 C Humidity = 1 % gasconcentration = 79 % to IBM Watso
n
Published Temperature = 44 C Humidity = 88 % gasconcentration = 69 % to IBM Wat
son
Published Temperature = 76 C Humidity = 54 % gasconcentration = 27 % to IBM Wat
son
Published Temperature = 37 C Humidity = 78 % gasconcentration = 10 % to IBM Wat
son
```

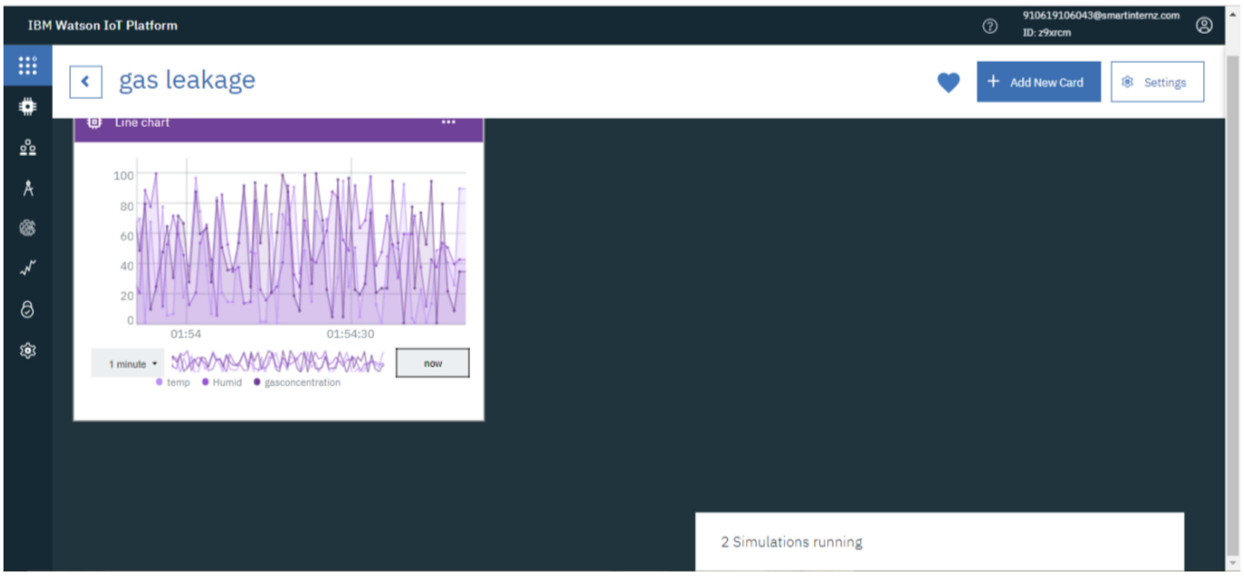
## Recent Events in IBM Watson IoT Platform:



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

| Event | Value | Format | Last Received |
|---|---|---|---|
| IoTSensor | {"temp":17,"Humid":97,"gasconcentration":12} | json | a few seconds ago |
| IoTSensor | {"temp":61,"Humid":49,"gasconcentration":48} | json | a few seconds ago |
| IoTSensor | {"temp":91,"Humid":49,"gasconcentration":77} | json | a few seconds ago |
| IoTSensor | {"temp":51,"Humid":79,"gasconcentration":43} | json | a few seconds ago |
| IoTSensor | {"temp":52,"Humid":52,"gasconcentration":57} | json | a few seconds ago |

## Boards in IBM Platform:


IBM Watson IoT Platform
910619106043@smartinternz.com
ID: z9xrcm
gas leakage
Add New Card
Settings
Line chart
100
80
60
40
20
0
01:54
01:54:30
1 minute
now
temp
Humid
gasconcentration
2 Simulations running