

SPRINT – 3

DATE	19 NOVEMBER 2022
TEAM ID	PNT2022TMID15274
PROJECT NAME	SMART WASTE MANAGEMENT FOR METROPOLITAN CITIES-IOT

PYTHON CODE : [To connect IBM WATSON]

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

```
#Provide your IBM Watson Device Credentials
organization = "s1e201"
deviceType = "lavi123"
deviceId = "12345"
authMethod = "token"
authToken = "23456789"
```

```
# Initialize GPIO
```

```
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
```

```
    level=random.randint(0,100)  
    weight=random.randint(0,100)
```

```
    data = { 'level' : level, 'weight': weight }  
    #print data
```

```
    def myOnPublishCallback():
```

```
        print ("Published level = %s C" % level, "weight = %s %% "  
% weight, "to IBM Watson")
```

```
success = deviceCli.publishEvent("IoTSensor", "json", data,
```

```
qos=0, on_publish=myOnPublishCallback)
    if not success:
        print("Not connected to IoTTF")
        time.sleep(1)
```

```
deviceCli.commandCallback = myCommandCallback
```

```
if (level>=75):
    print("Full LED ON")
```

```
# Disconnect the device and application from the cloud
deviceCli.disconnect()
```

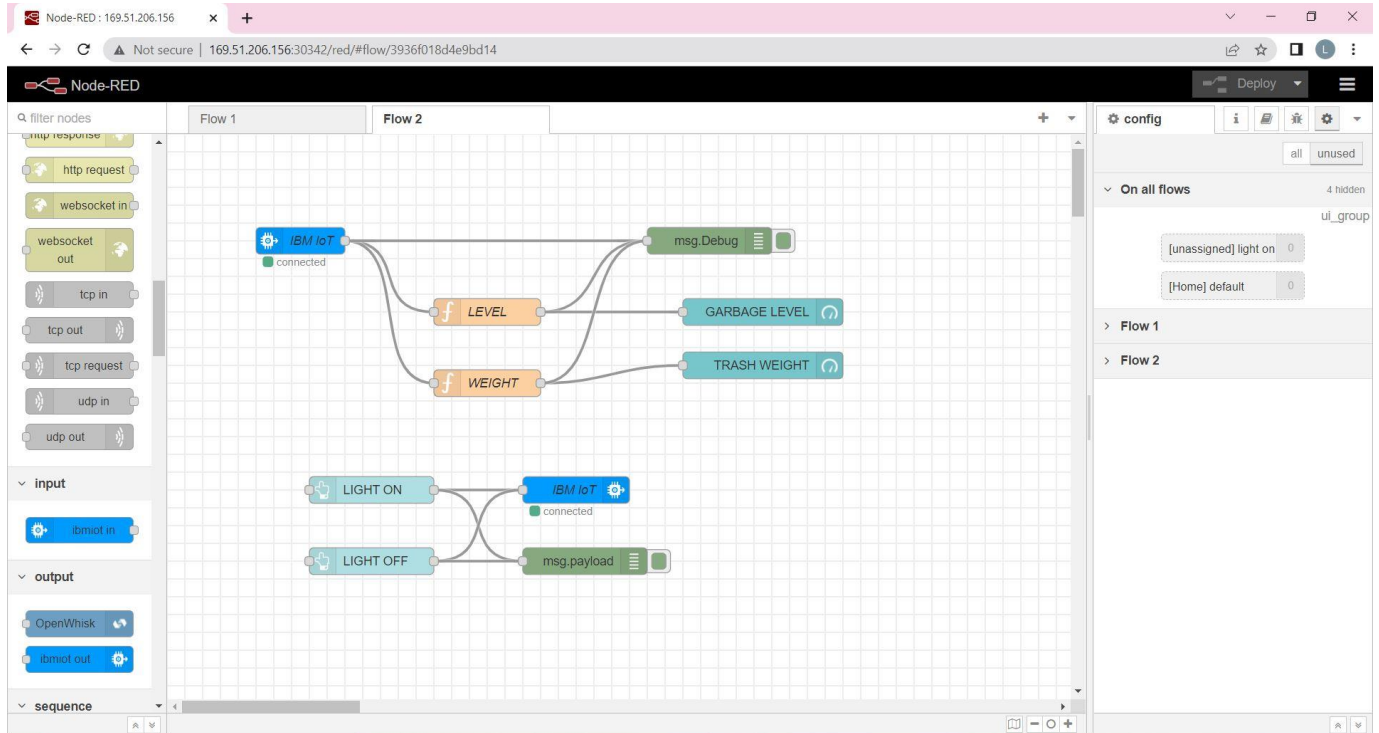
OUTPUT :

The screenshot displays the IBM Watson IoT Platform dashboard. The main interface shows a table of devices, with one device (ID: 12345, Status: Disconnected, Device Type: lavi123) selected. A modal window titled "Device Type: lavi123" is open, showing the configuration for a "Sensor" event type. The configuration includes a schedule of "20" minutes "Every Minute" and a payload editor. The payload is a JSON object with three fields: "randomNumber", "level", and "weight", each assigned a random value between 0 and 100. The payload editor shows the following JSON structure:

```
{
  "randomNumber": random(0,100),
  "level": random(0, 100),
  "weight": random(0,100)
}
```

The dashboard also shows a table of recent events for the selected device, with columns for Event, Value, Format, and Last Received. The events are listed as follows:

Event	Value	Format	Last Received
Sensor	{"randomNumber":34,"level":80,"weight":55}	json	a few seconds ago
Sensor	{"randomNumber":77,"level":62,"weight":12}	json	a few seconds ago
Sensor	{"randomNumber":32,"level":47,"weight":39}	json	a few seconds ago
Sensor	{"randomNumber":37,"level":22,"weight":74}	json	a few seconds ago
Sensor	{"randomNumber":71,"level":21,"weight":42}	json	a few seconds ago



```
File Edit Selection View Go Run Terminal Help ibm.py - Visual Studio Code

C:\Users\lavan > Desktop > Project Design and Planning > assignments > Team Member 2 > ibm.py > ...

1 import time
2 import sys
3 import ibmiotf.application
4 import ibmiotf.device
5 import random
6
7
8 #Provide your IBM Watson Device Credentials
9 organization = "s1e201"
10 deviceType = "lavi123"
11 deviceId = "12345"
12 authMethod = "token"
13 authToken = "23456789"
14
15 # Initialize GPIO
16
17
18 def myCommandCallback(cmd):
19     print("Command received: %s" % cmd.data['command'])
20     status=cmd.data['command']
21     if status=="lighton":
22         print ("led is on")
23     else :
24         print ("led is off")
25
26
27 #print(cmd)
28
29
30 try:
31     deviceOptions = {"org": organization, "type": deviceType, "id":
32     deviceId, "auth-method": authMethod, "auth-token": authToken}
33     deviceCli = ibmiotf.device.Client(deviceOptions) #.....
34
35 except Exception as e:
36     print("Caught exception connecting device: %s" % str(e))
37     sys.exit()
38
```