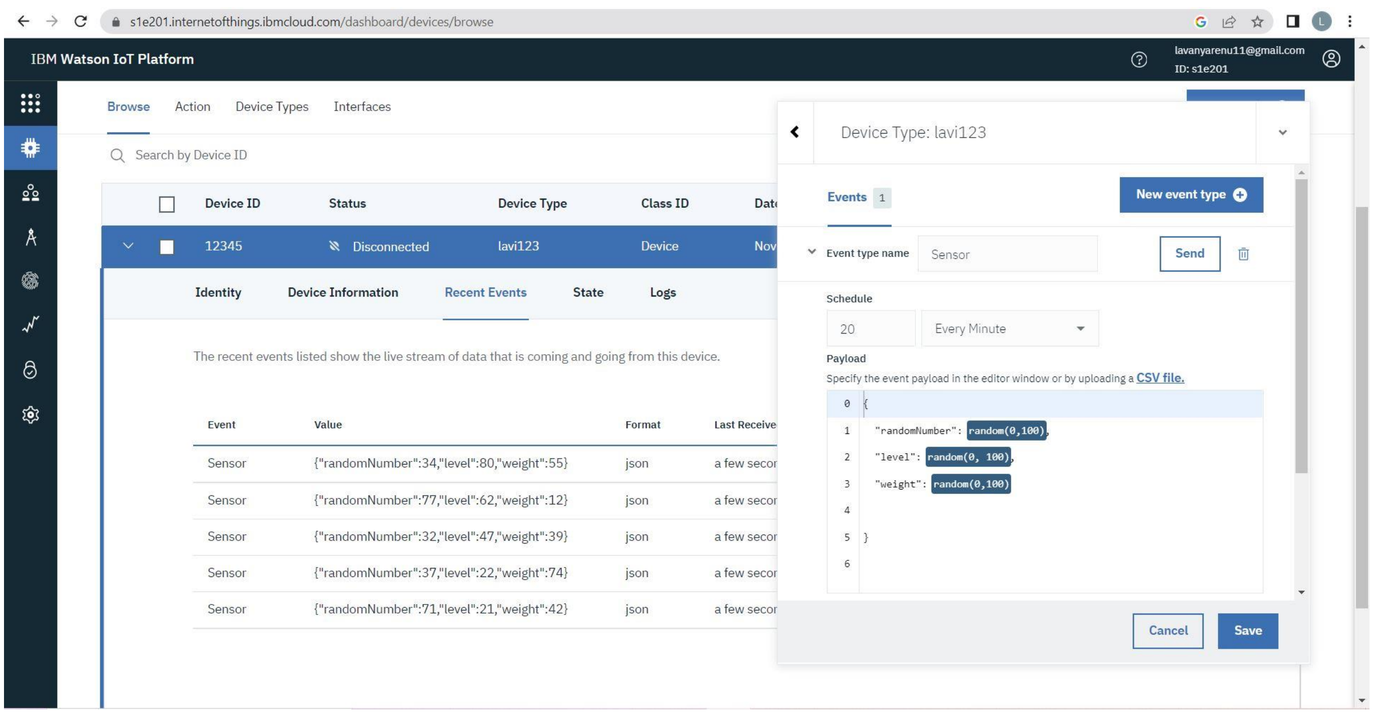
SPRINT – 3

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 IoTF")
        time.sleep(1)

        deviceCli.commandCallback = myCommandCallback

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

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

OUTPUT :



Device ID	Status	Device Type	Class ID	Dat
12345	Disconnected	lavi123	Device	Nov

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

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

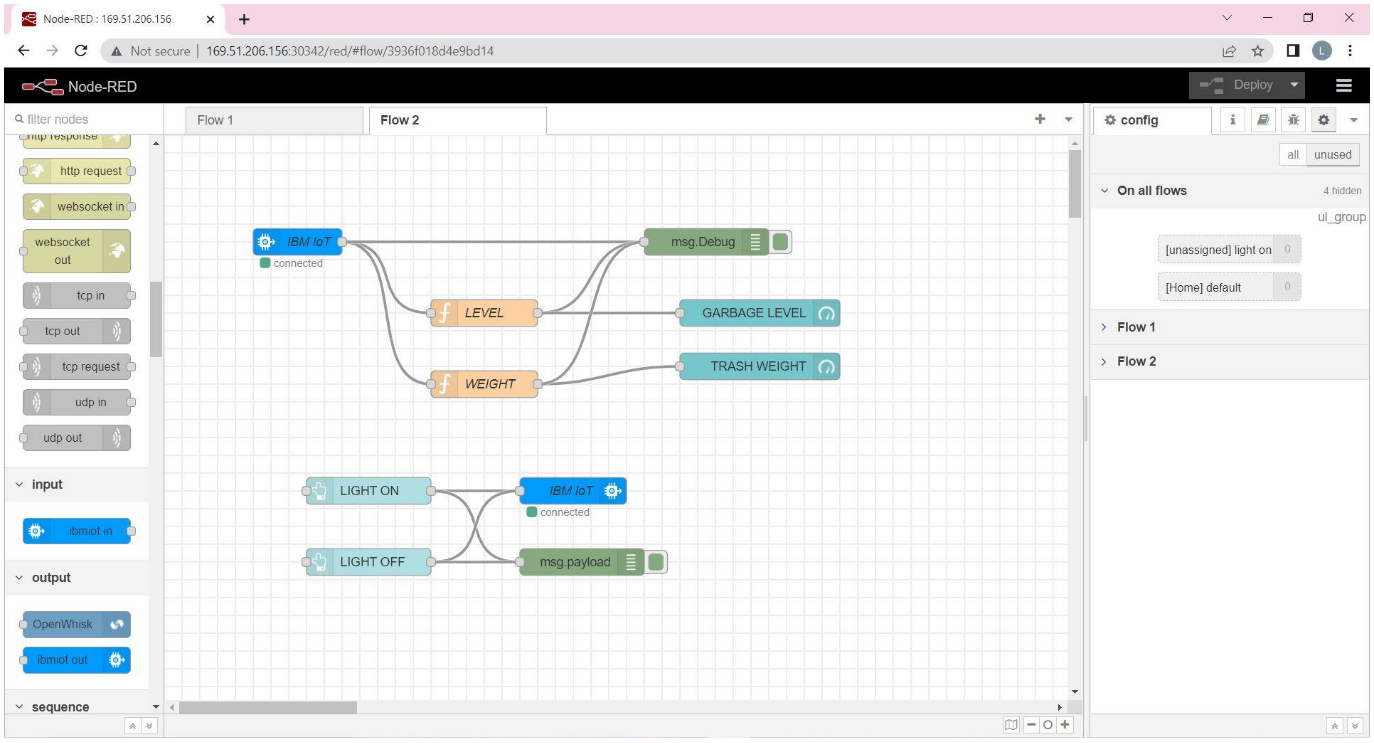
Event type name: Sensor

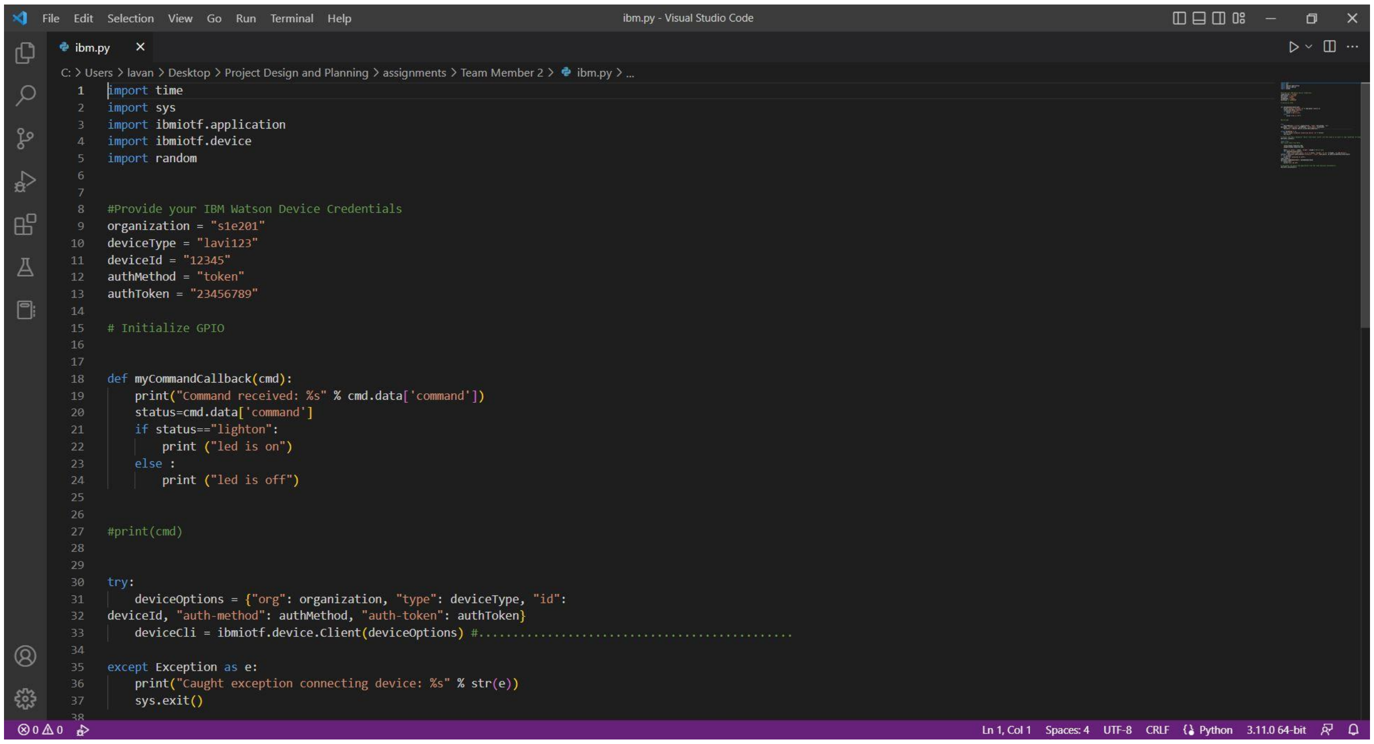
Schedule: 20 Every Minute

Payload

Specify the event payload in the editor window or by uploading a CSV file.

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

}
```




File Edit Selection View Go Run Terminal Help
ibm.py - Visual Studio Code
ibm.py
C: > Users > lavan > Desktop > Project Design and Planning > assignments > Team Member 2 > ibm.py > ...
⊗ 0 ⚠ 0
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python 3.11.0 64-bit

```python
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()
```