Delievery of Sprint-3

Date	15 November 2022
Team ID	PNT2022TMID42279
Project Name	SMART WASTE MANAGEMENT SYSTEM FORMETROPOLITAN CITIES
Maximum Marks	2 Marks

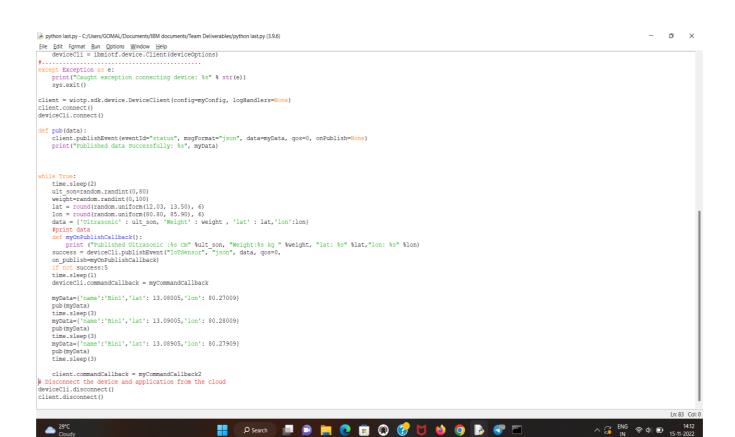
```
CODE:
import time
import wiotp.sdk.device
import sys
import ibmiotf.application
import ibmiotf.device
import random
import sys
#Provide your IBM Watson Device Credentials
organization = "udgvx5"
deviceType = "GPS"
deviceId = "1"
authMethod = "token"
authToken = "12345678"
myConfig = {
  "identity":{
     "orgld":"udgvx5",
     "typeId":"GPS",
     "deviceId":"1"
  },
   "auth":{
     "token": "12345678"
  }
def myCommandCallback2(cmd):
  print("Message received from IBM IoT Platform: %s" % cmd.data['command'])
  m=cmd.data['command']
# Initialize GPIO
def myCommandCallback(cmd):
  print("Command received: %s" % cmd.data['command'])
  status=cmd.data['command']
  if status =="lighton":
     print("led in on")
  else:
    print ("led is off")
  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()
client = wiotp.sdk.device.DeviceClient(config=myConfig, logHandlers=None)
client.connect()
deviceCli.connect()
def pub(data):
  client.publishEvent(eventId="status", msgFormat="json", data=myData, qos=0, onPublish=None)
  print("Published data Successfully: %s", myData)
while True:
  time.sleep(2)
  ult_son=random.randint(0,80)
  weight=random.randint(0,100)
  lat = round(random.uniform(12.03, 13.50), 6)
  lon = round(random.uniform(80.80, 85.90), 6)
  data = {'Ultrasonic' : ult_son, 'Weight' : weight , 'lat' : lat,'lon':lon}
  #print data
  def myOnPublishCallback():
     print ("Published Ultrasonic :%s Cm" %ult_son, "Weight:%s kg " %weight, "lat: %s" %lat, "lon: %s"
%lon)
  success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0,
  on_publish=myOnPublishCallback)
  if not success:5
  time.sleep(1)
  deviceCli.commandCallback = myCommandCallback
  myData={'name':'Bin1','lat': 13.08005,'lon': 80.27009}
  pub(myData)
  time.sleep(3)
  myData={'name':'Bin1','lat': 13.09005,'lon': 80.28009}
  pub(myData)
  time.sleep(3)
  myData={'name':'Bin1','lat': 13.08905,'lon': 80.27909}
  pub(myData)
  time.sleep(3)
  client.commandCallback = myCommandCallback2
# Disconnect the device and application from the cloud
deviceCli.disconnect()
```

client.disconnect()

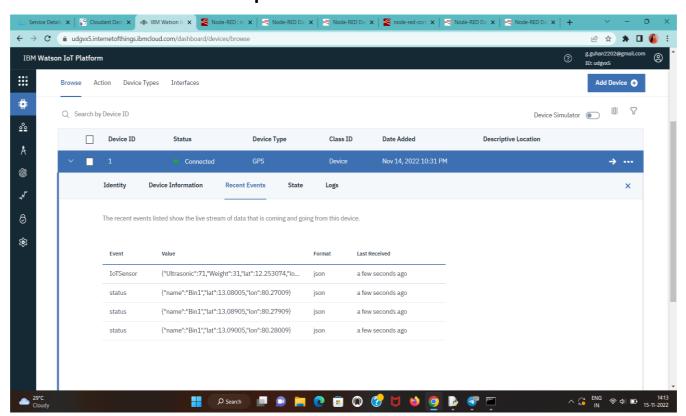
Python code:

```
🌛 python last.py - C:/Users/GOMAL/Documents/IBM documents/Team Deliverables/python last.py (3.9.6)
                                                                                                                                                                                                                                                                                                    - o ×
File Edit Format Run Options Window Help import time import wiotp.sdk.device
import wiotp.sdk.device
import sys
import immiotf.application
import immiotf.device
import random
import sys
Provide your IBM Watson Device Credentials
organization = "udgvx5"
deviceType = "GPS"
deviceTyd = "1"
authMethod = "token"
authToken = "12345678"
myConfig = {
    "identity":{
        "orgId":"udgvx5",
        "typeId":"GPS",
        "deviceId":"1"
       },
"auth":{
    "token":"12345678"
  bef myCommandCallback2(cmd):
    print("Message received from IBM IOT Platform: %s" % cmd.data['command'])
    m=cmd.data['command']
# Initialize GPIO
def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status =="lighton":
        print("led in on")
    else:
             print ("led is off")
      deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method":authMethod, "auth-token": authToken} deviceCli = ibmiotf.device.Client(deviceOptions)
                      aught exception connecting device: %s" % str(e))
       sys.exit()
client = wiotp.sdk.device.DeviceClient(config=myConfig, logHandlers=None)
client.connect()
                                                                                      △ 29°C
```



Python Output:

IBM Watson Platform Output:



NodeRed Platform:

