Develop a Python Script

Team ID	**PNT2022TMID04607**
Project Name	Smart Waste Management System in Metropolitan Cities.

Step 1 : Open Python IDLEStep

2 :Type the program

Step 3 : Then Click on file & Save the document.

Step 4 : Then click on Run, Click run module.

Step 5 : Output will be appeared in the IDLE Window.

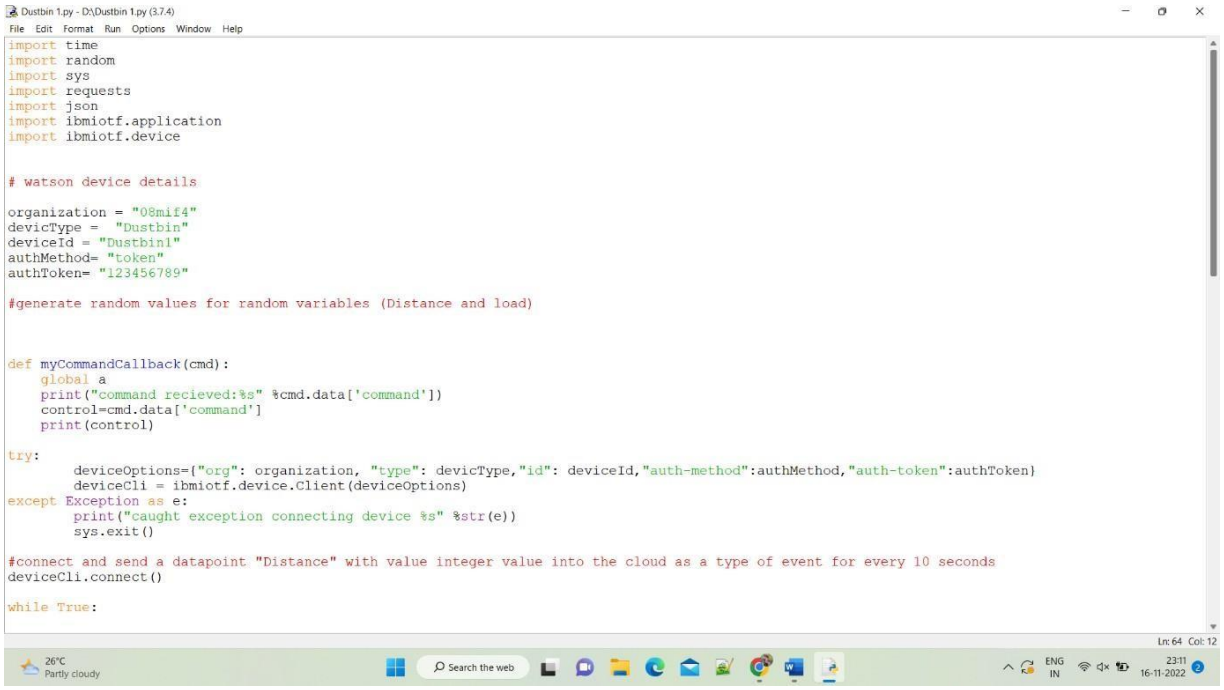
Python Script :

```
import time
import random
import sys import
requestsimport
json
import ibmiotf.applicationimport
ibmiotf.device
# watson device details
OrganizationID="kz2her"
DeviceType="INIAN"
DeviceID="9360"
AuthenticationMethod="use-token-auth"
AuthenticationToken="zw?q1U3ycJr_gLFDJ5"
#generate random values for random variables (Distance and load)def
myCommandCallback(cmd):
  global a
  print("command recieved:%s" %cmd.data['command'])
  control=cmd.data['command']
 print(control)try:
 deviceOptions={"org": organization, "type": devicType,"id":
deviceId,"authmethod":authMethod,"auth-token":authToken}
deviceCli = ibmiotf.device.Client(deviceOptions)
except Exception as e:
  print("caught exception connecting device %s" %str(e))
  sys.exit()
```

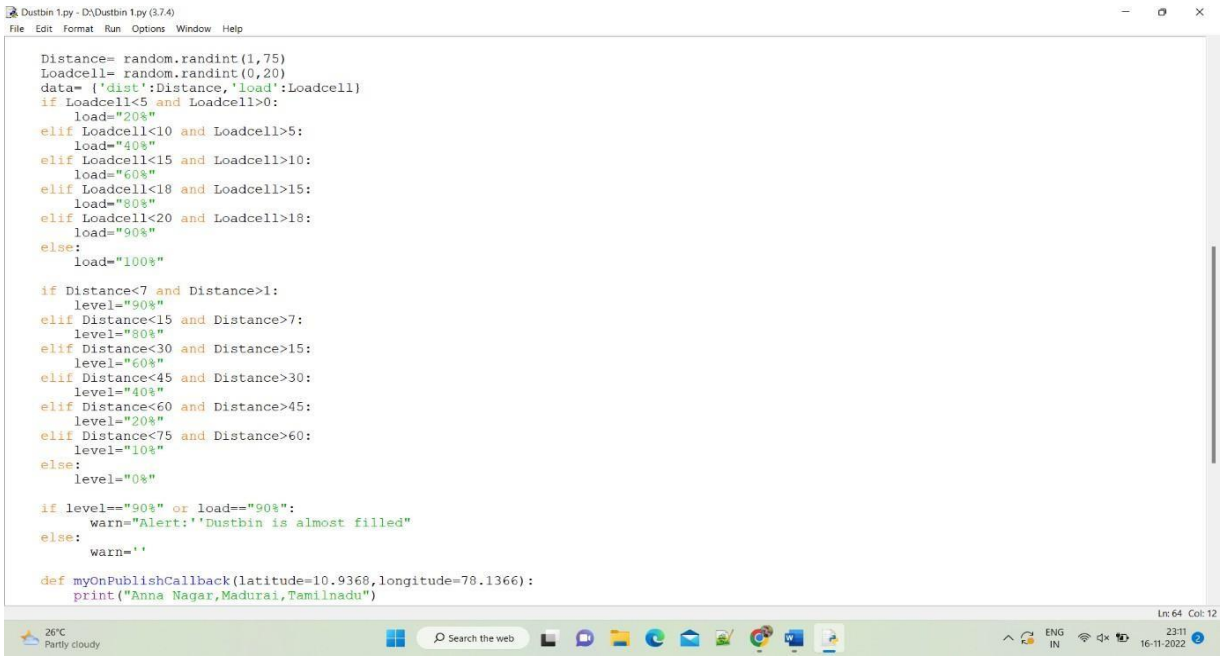
```
#connect and send a datapoint "Distance" with value integer value into the cloud as atype
of event for every 10 seconds
deviceCli.connect()
while True:
 Distance= random.randint(1,75)
Loadcell= random.randint(0,20) data=
{'dist':Distance,'load':Loadcell} if
Loadcell<5 and Loadcell>0:
   load="20%"
elif Loadcell<10 and Loadcell>5:
   load="40%"
elif Loadcell<15 and Loadcell>10:
   load="60%"
elif Loadcell<18 and Loadcell>15:
   load="80%"
elif Loadcell<20 and Loadcell>18:
 load="90%"else:
   load="100%"
if Distance<7 and Distance>1:
   level="90%"
elif Distance<15 and Distance>7:
   level="80%"
elif Distance<30 and Distance>15:
   level="60%"
elif Distance<45 and Distance>30:
   level="40%"
elif Distance<60 and Distance>45:
   level="20%"
elif Distance<75 and Distance>60:
 level="10%"else:
   level="0%"
if level=="90%" or load=="90%":
warn="Alert:''Dustbin is almost filled"else:
warn=''
  def myOnPublishCallback(latitude=10.9368,longitude=78.1366):print("Anna
   Nagar,Madurai,Tamilnadu")
print("published Level of bin = %s " %level,"Load = %s " %load, "Latitude = %s "
  %latitude,"Longitude = %s " %longitude)print(load)
print(level)
```

```
print(warn)
time.sleep(10)
success=deviceCli.publishEvent
("IoTSensor","json",warn,qos=0,on_publish=myOnPublishCallback)
success=deviceCli.publishEvent
("IoTSensor","json",data,qos=0,on_publish=myOnPublishCallback)  if not
success:
 print("not connected to ibmiot")
time.sleep(20)
deviceCli.commandCallback=myCommandCallback
#disconnect the device
deviceCli.disconnect()
```

Code Snap :



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


# watson device details

organization = "08mif4"
devicType =  "Dustbin"
deviceId = "Dustbin1"
authMethod= "token"
authToken= "123456789"

#generate random values for random variables (Distance and load)



def myCommandCallback(cmd):
    global a
    print("command recieved:%s" %cmd.data['command'])
    control=cmd.data['command']
    print(control)

try:
        deviceOptions={"org": organization, "type": devicType,"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 "Distance" with value integer value into the cloud as a type of event for every 10 seconds
deviceCli.connect()

while True:
```



```
    Distance= random.randint(1,75)
    Loadcell= random.randint(0,20)
    data= {'dist':Distance,'load':Loadcell}
    if Loadcell<5 and Loadcell>0:
        load="20%"
    elif Loadcell<10 and Loadcell>5:
        load="40%"
    elif Loadcell<15 and Loadcell>10:
        load="60%"
    elif Loadcell<18 and Loadcell>15:
        load="80%"
    elif Loadcell<20 and Loadcell>18:
        load="90%"
    else:
        load="100%"

    if Distance<7 and Distance>1:
        level="90%"
    elif Distance<15 and Distance>7:
        level="80%"
    elif Distance<30 and Distance>15:
        level="60%"
    elif Distance<45 and Distance>30:
        level="40%"
    elif Distance<60 and Distance>45:
        level="20%"
    elif Distance<75 and Distance>60:
        level="10%"
    else:
        level="0%"

    if level=="90%" or load=="90%":
          warn="Alert:''Dustbin is almost filled"
    else:
          warn=''

    def myOnPublishCallback(latitude=10.9368,longitude=78.1366):
        print("Anna Nagar,Madurai,Tamilnadu")
```

```
Dustbin 1.py - D:\Dustbin 1.py (3.7.4)
File  Edit  Format  Run  Options  Window  Help
    elif Distance<60 and Distance>45:
        level="20%"
    elif Distance<75 and Distance>60:
        level="10%"
    else:
        level="0%"

    if level=="90%" or load=="90%":
         warn="Alert:''Dustbin is almost filled"
    else:
         warn=''

    def myOnPublishCallback(latitude=10.9368,longitude=78.1366):
        print("Anna Nagar,Madurai,Tamilnadu")
        print("published Level of bin = %s " %level,"Load = %s " %load, "Latitude = %s " %latitude,"Longitude = %s " %longitude)
        print(load)
        print(level)
        print(warn)

    time.sleep(10)


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

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



    if not success:
        print("not connected to ibmiot")
    time.sleep(20)



    deviceCli.commandCallback=myCommandCallback
#disconnect the device
deviceCli.disconnect()
```

OUTPUT :