FINAL DELIVERABLES

FINAL CODE

BIN PYTHON CODE SIMULATOR→_BIN1.PY_

```
import requests

import json

import ibmiotf.application

import ibmiotf.device

import time

import random

import sys

# watson device details

organization = "4yi0vc"

devicType =  "BIN1"

deviceId = "BIN1ID"

authMethod= "token"

authToken= "123456789"

#generate random values for randomo variables (temperature&humidity)

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 "temp" with value integer value into the cloud as a type of event for
every 10 seconds

deviceCli.connect()


while True:


   distance= random.randint(10,70)

   loadcell= random.randint(5,15)

   data= {'dist':distance,'load':loadcell}




   if loadcell < 13 and loadcell > 15:

      load = "90 %"


   elif loadcell < 8 and loadcell > 12:

       load = "60 %"


   elif loadcell < 4 and loadcell > 7:

       load = "40 %"

   else:

       load = "0 %"


   if distance < 15:

       dist = 'Risk warning:' 'Dumpster poundage getting high, Time to collect :) 90 %'




   elif distance < 40 and distance >16:

       dist = 'Risk warning:' 'dumpster is above 60%'


   elif distance < 60 and distance > 41:
       dist = 'Risk warning:' '40 %'
   else:
       dist = 'Risk warning:' '17 %'




   if load == "90 %" or distance == "90 %":
       warn  = 'alert :' ' Dumpster poundage getting high, Time to collect :)'

   elif load == "60 %" or distance == "60 %":

       warn = 'alert :' 'dumpster is above 60%'
   else :
       warn = 'alert :' 'No need to collect right now '
   def myOnPublishCallback(lat=10.678991,long=78.177731):
      print("Gandigramam, Karur")
      print("published distance = %s " %distance,"loadcell:%s " %loadcell,"lon = %s " %long,"lat = %s"
%lat)
      print(load)
      print(dist)
      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(30)




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