**SPRINT 3**

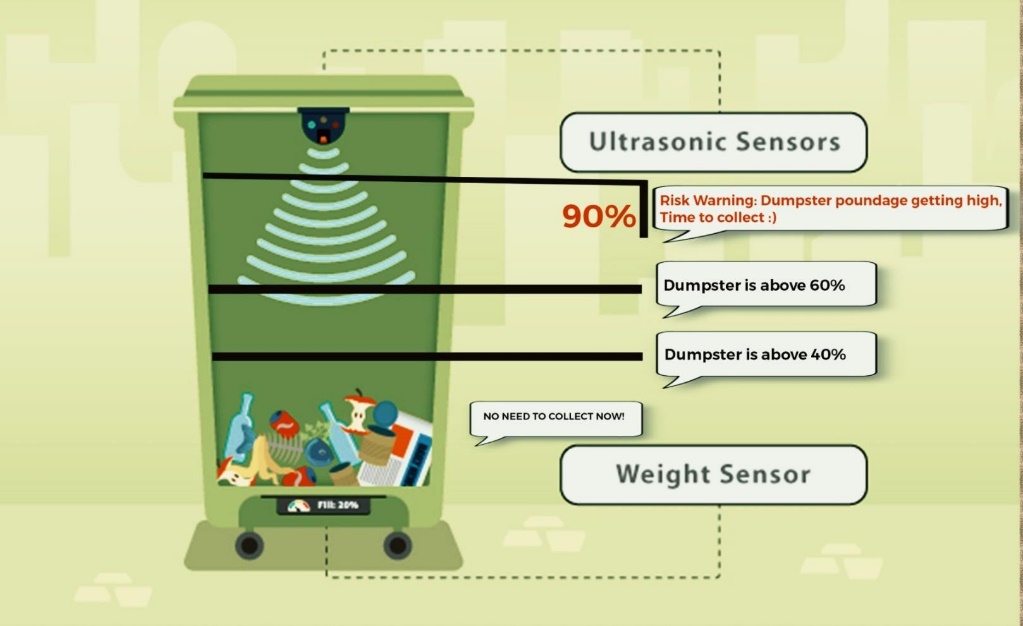
|  |  |
| --- | --- |
| TEAM ID | PNT2022TMID50747 |
| PROJECT | Smart Waste Management System For Metropolitan Cities |

**DESIGN AN WEB PAGE :**

**PARAMETERS ARE:**

1. **DISTANCE**
2. **WEIGHT**

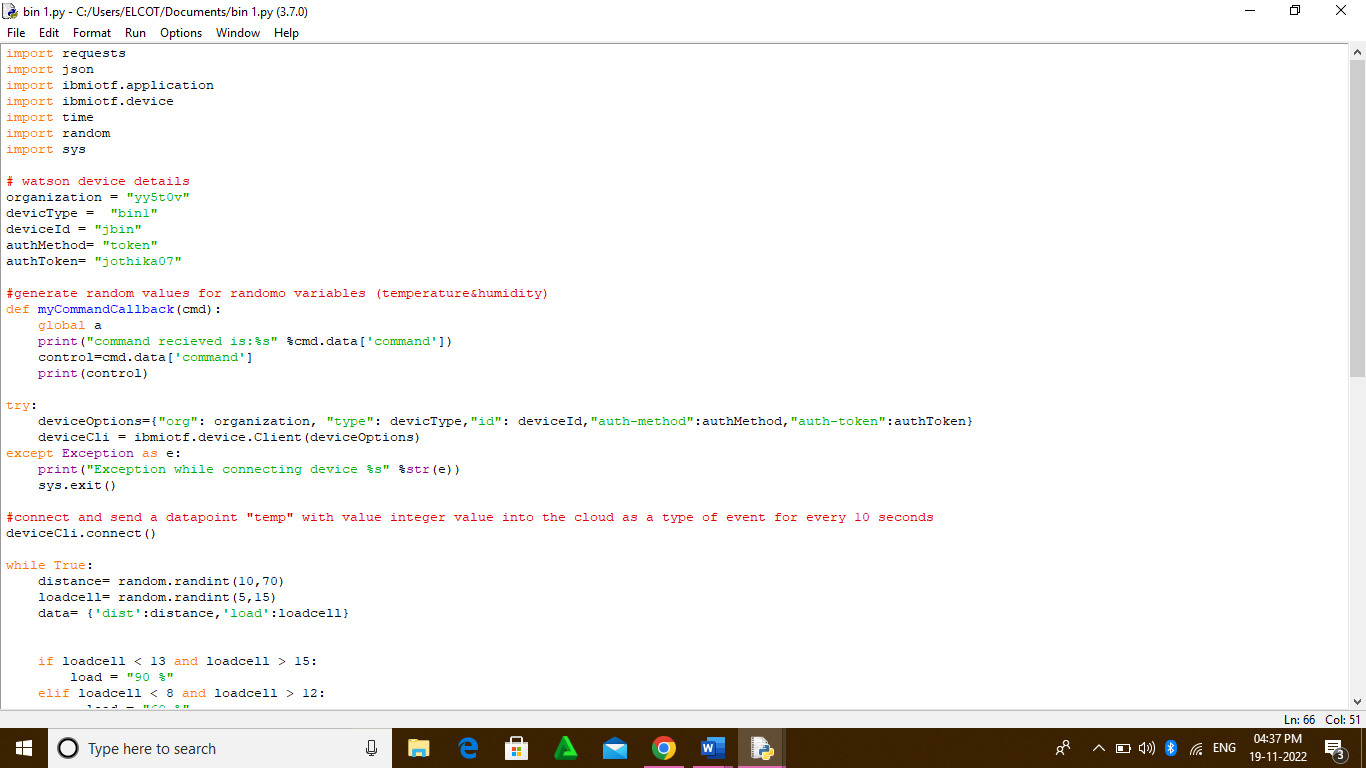
PROJECT MODEL :



The ultrasonic sonic sensor and the weight sensor which are used to calculate the garbage distance and the weight respectively.

There are certain assumptions assumed by us, They are

* The length of the trash can is assumed to be 200 cm.
* The maximum weight of the can is assumed to be 2 Kg.
* If the garbage distance goes more than 180cm i.e more than 90% of the trash can , the sensor is has to send to send an alert to the garbage collector.
* If the alert is received , then the garbage collector has to come and collect the garbage.
* The current weight and the garbage distance is to be updated periodically, i.e for 5 minutes.





**PYTHON CODE:**

import requests

import json

import ibmiotf.application

import ibmiotf.device

import time

import random

import sys

# watson device details

organization = "yy5t0v"

devicType = "bin2"

deviceId = "kbin"

authMethod= "token"

authToken= "kiruba05"

generate random values for randomo variables (temperature&humidity)

def myCommandCallback(cmd):

global a

print("command recieved is:%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("Exception while 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:' 'Garbage level is high, collection time :) 90 %'

elif distance < 40 and distance >16:

dist = 'Risk warning:' 'garbage 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 :' ' Garbage level is high, collection time :)'

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

warn = 'alert :' 'garbage is above 60%'

else :

warn = 'alert :' 'Levels are low, collection not needed '

def myOnPublishCallback(lat=8.7450,long=77.7046):

print("Thatchanallur, Tiruneveli")

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()

