

PROJECT DEVELOPMENT PHASE

SPRINT-4

TEAM ID	PNT2022TMID04740
PROJECT	Smart waste management system for metropolitan cities

PYTHON SCRIPT:

```
import requests

import json

import ibmiotf.application

import ibmiotf.device

import time

import random

import sys

# watson device details

organization = "j5bxb7"

devicType = "IOT123edevicetype"

deviceId = "IOTece4"

authMethod= "token"

authToken= "e2)-17xkqIFMvm3@II"

#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

```

OUTPUT:

1.PYTHON SIMULATION:

```

spr4.py - C:/Users/ELCOT/AppData/Local/Programs/Python/Python37/spr4.py (3.7.0)
File Edit Format Run Options Window Help

import ibmiotf.device
import time
import random
import sys
# watson device details
organization = "j5bxb7"
deviceType = "IOT123edevicetype"
deviceId = "IOTece4"
authMethod= "token"
authToken= "e2)-17xkqIFMvm3@lI"
#generate random values for random 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": deviceType,"id": deviceId}
    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
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 %"

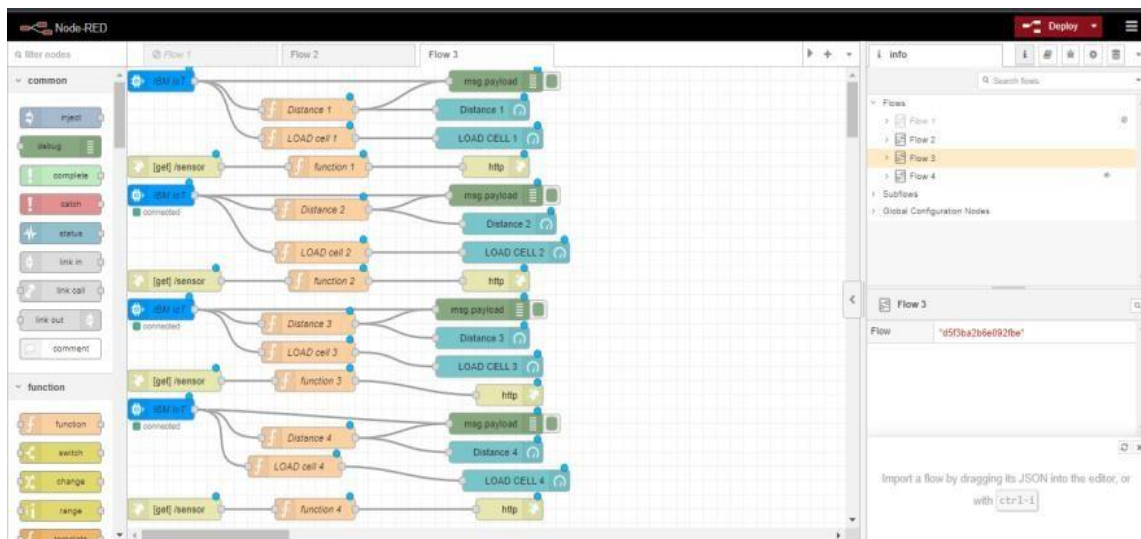
    elif loadcell < 8 and loadcell > 12:
        load = "60 %"

    elif loadcell < 4 and loadcell > 7:
        load = "40 %"
    else:

Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
>>>
=== RESTART: C:/Users/ELCOT/AppData/Local/Programs/Python/Python37/spr4.py ===
2022-11-15 20:06:50,185 ibmiotf.device.Client INFO Connected successfully: d:j5bxb7:IOT123edevicetype:IOTece4
Gandigramam, Karur
published distance = 45 loadcell:15 lon = 78.177731 lat = 10.678991
0 %
Risk warning:40 %
alert :No need to collect right now
Gandigramam, Karur
published distance = 45 loadcell:15 lon = 78.177731 lat = 10.678991
0 %
Risk warning:40 %
alert :No need to collect right now
Gandigramam, Karur
published distance = 53 loadcell:5 lon = 78.177731 lat = 10.678991
0 %
Risk warning:40 %
alert :No need to collect right now
Gandigramam, Karur
published distance = 53 loadcell:5 lon = 78.177731 lat = 10.678991
0 %
Risk warning:40 %
alert :No need to collect right now
Gandigramam, Karur
published distance = 33 loadcell:10 lon = 78.177731 lat = 10.678991
0 %
Risk warning:dumpster is above 60%
alert :No need to collect right now
Gandigramam, Karur
published distance = 33 loadcell:10 lon = 78.177731 lat = 10.678991
0 %
Risk warning:dumpster is above 60%
alert :No need to collect right now
Gandigramam, Karur
published distance = 20 loadcell:14 lon = 78.177731 lat = 10.678991

```

2. Node-RED Connection setup for data transmission from IBM Watson IOT platform to Node-RED dashboard:

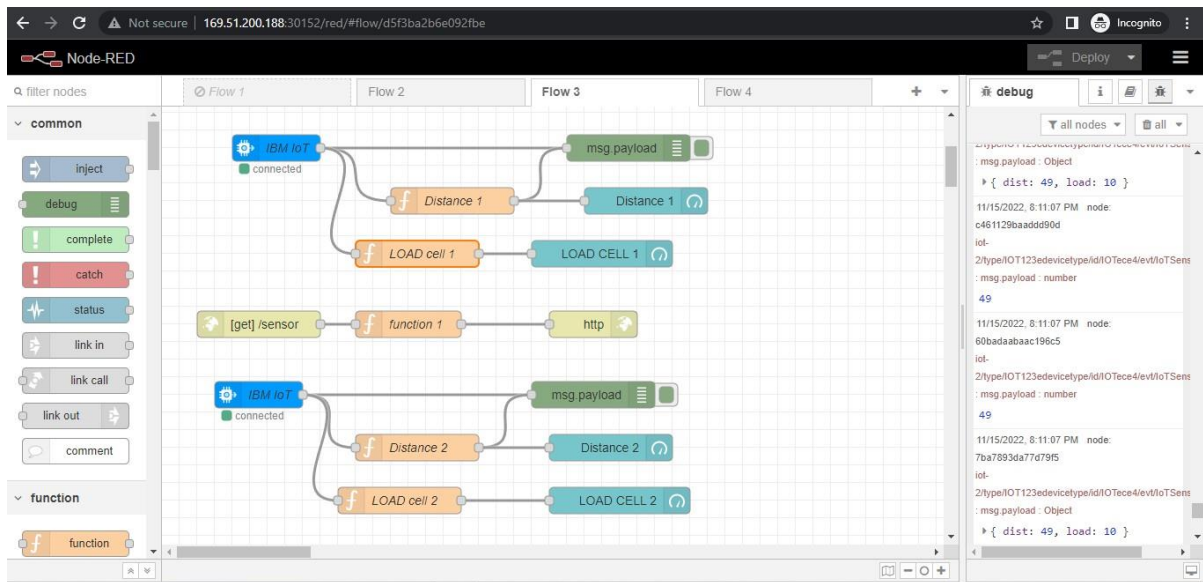


3. Data transfer to IBM Watson IOT platform:

The screenshot shows the IBM Watson IoT platform interface. The top navigation bar includes 'Browse', 'Action', 'Device Types', and 'Interfaces'. A 'Delete' button is visible. The main table lists devices, with 'IoTec4' selected. The 'Recent Events' tab is active, showing a list of events.

Event	Value	Format	Last Received
IoTSensor	{"dist":55,"load":15}	json	a few seconds ago
IoTSensor	{"type":"Buffer","data":[34,97,108,101,114,116,...]}	json	a few seconds ago
IoTSensor	{"dist":18,"load":13}	json	a few seconds ago
IoTSensor	{"type":"Buffer","data":[34,97,108,101,114,116,...]}	json	a few seconds ago

4.Data transfer from IBM Watson IOT platform and python script to Node-RED:



5.Storing database in IBM Cloudant DB:

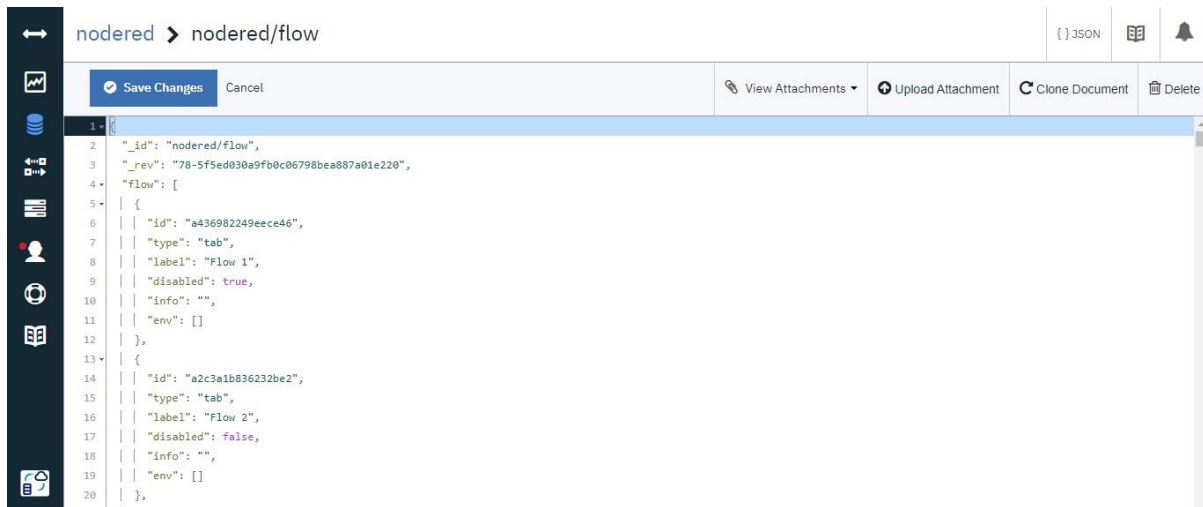
The screenshot shows the IBM Cloudant DB interface. The 'Databases' section displays a table with the following data:

Name	Size	# of Docs	Partitioned	Actions
nodered	37.9 KB	4	No	[Icons for actions]
simple	31 bytes	1	No	[Icons for actions]

The screenshot shows the 'nodered' database details in the IBM Cloudant DB interface. The 'All Documents' section displays a table with the following data:

id	key	value
._design/library	._design/library	{ "rev": "1-c93136490a0976308f8b3..." }
nodered/credential	nodered/credential	{ "rev": "7-5cfc9d4d7a92fb46121fe4..." }
nodered/flow	nodered/flow	{ "rev": "77-2dcff42c2f0b3057e15bd..." }
nodered/settings	nodered/settings	{ "rev": "16-b43ccdb036a30d73adc2f..." }

6.Data Stored in JSON format:



```
1 {
2   "_id": "nodered/flow",
3   "_rev": "78-5f5ed030a9fb0c06798bea887a01e220",
4   "flow": [
5     {
6       "id": "a436982249eece46",
7       "type": "tab",
8       "label": "Flow 1",
9       "disabled": true,
10      "info": "",
11      "env": []
12    },
13    {
14      "id": "a2c3a1b836232be2",
15      "type": "tab",
16      "label": "Flow 2",
17      "disabled": false,
18      "info": "",
19      "env": []
20    }
21  ]
22 }
```

7.Web UI:

