

| | |
|--------------|---|
| Date | 5 November 2022 |
| Team ID | PNT2022TMID42565 |
| Project Name | Smart waste management system for metropolitan cities |
| Story Points | 15 |

Sprint 2

Develop the python code to find the GPS location using Latitude and Longitude (random values) and send it to Node red using IBM Watson platform and view location of bins on map

PYTHON CODE :

```
import
wiotp.sdk.device
import time import
random myConfig = {
    "identity": {
        "orgId": "fzv53v",
        "typeId": "Bin",
        "deviceId": "Bin_1"
    },
    "auth": {
        "token": "1234567890"
    }
}

def myCommandCallback (cmd):
```

```

print ("Message received from IBM IoT Platform: %s" % cmd.data['command'])

m=cmd.data['command']

client = wiotp.sdk.device.DeviceClient(config=myConfig, logHandlers=None)

client.connect()
def pub (data): client.publishEvent(eventId="status",
msgFormat="json", data=myData, qos=0,
onPublish=None)
print ("Published data Successfully: %s", myData)
while True:

    myData={'name': 'Bin1', 'lat': 13.092677, 'lon': 80.188314}

    pub (myData)
    time.sleep (3)

client.commandCallback = myCommandCallback

client.disconnect ()

```

Output in python IDLE :

The screenshot shows the Python IDLE environment with a script named 'bin gps.py' and its output in the Shell window.

Script Code (bin gps.py):

```

import wiotp.sdk.device
import time
import random
myConfig = {
    "identity": {
        "orgId": "fzv53v",
        "typeId": "Bin",
        "deviceId": "Bin_1"
    },
    "auth": {
        "token": "1234567890"
    }
}

def myCommandCallback (cmd):
    print ("Message received from IBM IoT Platform: %s" % cmd.data['command'])
    m=cmd.data['command']

client = wiotp.sdk.device.DeviceClient(config=myConfig, logHandlers=None)
client.connect()

def pub (data):
    client.publishEvent(eventId="status", msgFormat="json", data=myData, qos=0, onPublish=None)
    print ("Published data Successfully: %s", myData)

while True:
    myData={'name': 'Bin1', 'lat': 13.092677, 'lon': 80.188314}
    pub (myData)
    time.sleep (3)

    client.commandCallback = myCommandCallback

client.disconnect ()

```

Shell Output:

```

Python 3.7.9 (tags/v3.7.9:13c94747c7, Aug 17 2020, 16:30:00) [MS
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more info
>>>
===== RESTART: C:\Users\Akash M\Desktop\bin gps.py ==
2022-11-11 10:36:33,849 wiotp.sdk.device.client.DeviceClient: ]
d successfully: d:fzv53v:Bin:Bin_1
Published data Successfully: %s {'name': 'Bin1', 'lat': 13.09267
314}
Published data Successfully: %s {'name': 'Bin1', 'lat': 13.09267
314}

```

IBM Watson IOT platform :

The screenshot shows the IBM Watson IoT Platform dashboard. The top navigation bar includes 'Browse', 'Action', 'Device Types', and 'Interfaces'. A search bar is present with the text 'Search by Device ID'. The main content area displays a table of devices. The first device has ID 12345, status 'Disconnected', device type 'Rasp', class ID 'Device', and was added on 'Nov 12, 2022 10:39 AM'. Below the table, a detailed view for device 12345 is shown, including fields for Device ID, Device Type, Date Added, Added By, and Connection Status. The bottom status bar indicates '1 Simulation running'.

| Device ID | Status | Device Type | Class ID | Date Added | Descriptive Location |
|-----------|--------------|-------------|----------|-----------------------|----------------------|
| 12345 | Disconnected | Rasp | Device | Nov 12, 2022 10:39 AM | |

| Identity | Device Information | Recent Events | State | Logs |
|-------------------|---------------------------------|---------------|-------|------|
| Device ID | 12345 | | | |
| Device Type | Rasp | | | |
| Date Added | Nov 12, 2022 10:39 AM | | | |
| Added By | mithradhanalakshmi915@gmail.com | | | |
| Connection Status | Disconnected | | | |

Node Red Platform :

The screenshot shows the Node-RED web interface. The main workspace displays a flow diagram for 'Flow 1'. The flow starts with an 'IBM IoT' node (labeled 'connected'). This node branches into two parallel paths. The top path goes through a 'weight' function node to a 'Weight' output node. The bottom path goes through a 'location' function node to a 'Location' output node. Both output nodes are connected to a 'msg.payload' node. The left sidebar shows a 'parser' section with a 'json' node. The right sidebar shows a 'dashboard' section with 'Layout', 'Site', and 'Theme' tabs, and a 'Tabs & Links' section with a 'Smart Waste Management System' link.