# SPRINT-4

| | |
|---|---|
| Assignment Date | 16th November 2022 |
| Team ID | PNT2022TMID39421 |
| Project Name | Smart Solution For Railways Using IOT |

## Project Description:-

Smart solution for railways is designed to reduce the work load of the user and also the use of paper.

## Project Feature:-

A GPS module is present in the train to track it.The live status of the journey is updated in the web application.

## Task:-

Design the modules and test Apps To make the user to interact with software

# Python code for GPS tracking

model.py - C:\Users\yokes\AppData\Local\Programs\Python\Python37\model.py (3.7.0)

File Edit Format Run Options Window Help

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

#Provide your IBM Watson Device Credentials
organization = "vptt54"
deviceType = "GPS"          #Credentials of Watson IoT sensor simulator
deviceId = "1234"
authMethod = "token"
authToken = "12345678"


# Initialize the device client.
L=0

try:
        deviceOptions = {"org": organization, "type": deviceType, "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 "hello" with value "world" into the cloud as an event of type "greeting" 10 times
deviceCli.connect()

while True:
    overpass_url = "http://overpass-api.de/api/interpreter"
    overpass_query = """
    [out:json];area[name="India"];(node[place="village"](area););out;
    """

    response = requests.get(
    overpass_url,
    params={'data': overpass_query}
    )

    coords = []
    if response.status_code == 200:
        data = response.json()
        places = data.get('elements', [])
        for place in places:
            coords.append((place['lat'], place['lon']))
        print ("Got %s village coordinates!" % len(coords))
        print (coords[0])
    else:
        print("Error")

    i = random.randint(1,100)
    L = coords[i]
    #Send random gprs data to node-red to IBM Watson
    data = {"d":{ 'Latitude' : L[0], 'Longitude' : L[1]}}
    #print data
    def myOnPublishCallback():
        print("Published gprs location = ", L, "to IBM Watson")
```

Ln: 48 Col: 0


# Python code output

1stcode.py - C:\Users\yokes\AppData\Local\Programs\Python\Python37\1stcode.py (3.7.0)

File Edit Format Run Options Window Help

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

#Provide your IBM
organization = "
deviceType = "ab
deviceId = "1234
authMethod = "to
authToken = "123

# Initialize GPI


def myCommandCal
    print("Comma
    status=cmd.da
    if status=="
        print ("
    else :
        print ("
    #print(cmd)



try:
        deviceOp
        deviceCl
        #.......

except Exception
        print("C
        sys.exit

# Connect and se
deviceCli.connect

while True:
        #Get Sen
        temp=ran
        Humid=ra

        data = {
        #print da
        def myOnPublishCallback():
            print ("Published Temperature = %s C" % temp, "Humidity = %s %%" % Humid, "to IBM Watson")

        success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0, on_publish=myOnPublishCallback)
        if not success:
            print("Not connected to IoTF")
        time.sleep(1)

        deviceCli.commandCallback = myCommandCallback

# Disconnect the device and application from the cloud
```

"Python 3.7.0 Shell"

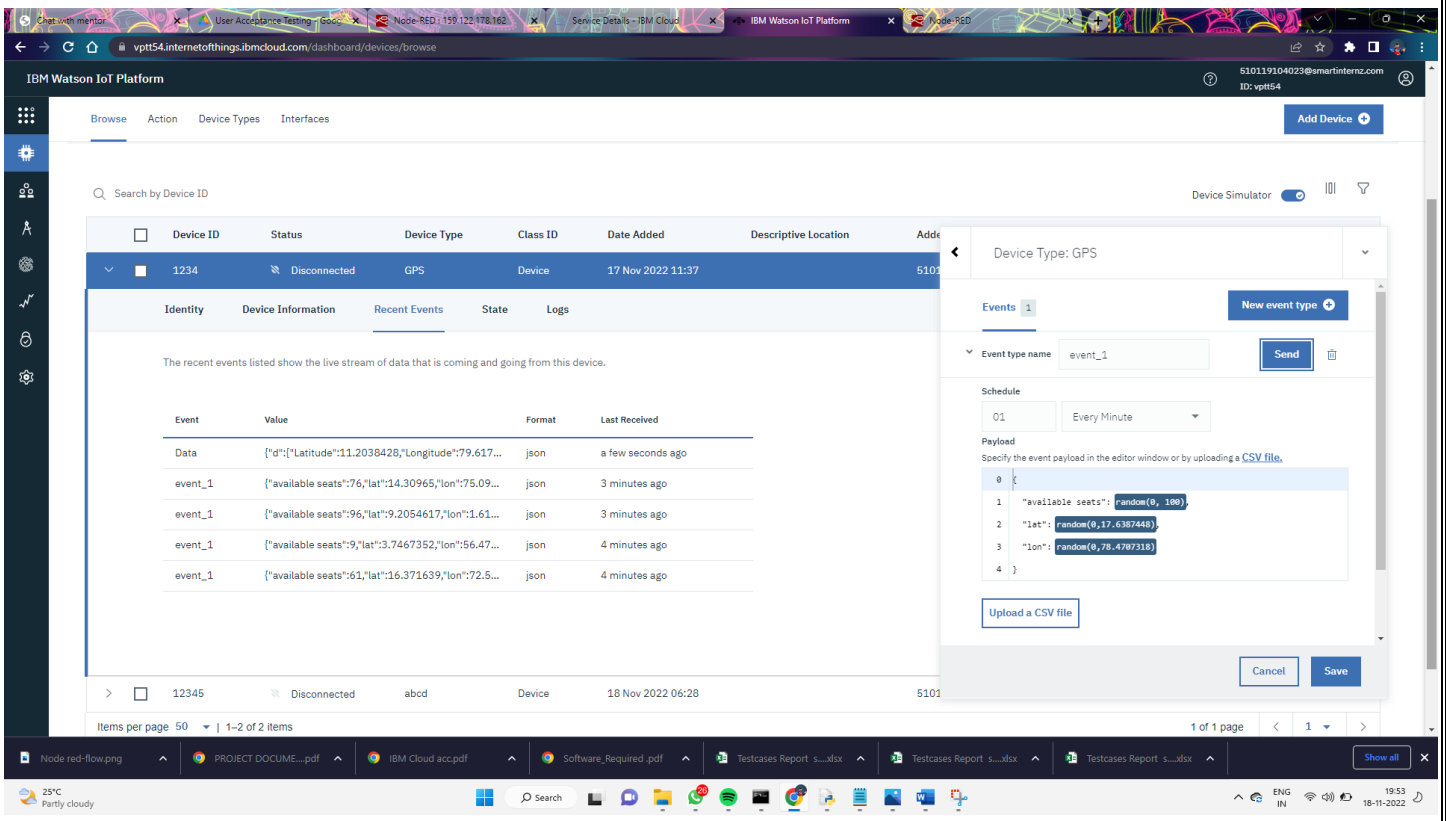
File Edit Shell Debug Options Window Help

```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD6
4)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\yokes\AppData\Local\Programs\Python\Python37\1stcode.py =
2022-11-18 06:33:45,470    ibmiotf.device.Client      INFO    Connected successfu
lly: d:vptt54:abcd:12345
Published Temperature = 7 C Humidity = 55 % to IBM Watson
Published Temperature = 92 C Humidity = 55 % to IBM Watson
Published Temperature = 20 C Humidity = 29 % to IBM Watson
Published Temperature = 11 C Humidity = 46 % to IBM Watson
Published Temperature = 90 C Humidity = 94 % to IBM Watson
Published Temperature = 19 C Humidity = 53 % to IBM Watson
Published Temperature = 45 C Humidity = 49 % to IBM Watson
Published Temperature = 37 C Humidity = 56 % to IBM Watson
Published Temperature = 73 C Humidity = 87 % to IBM Watson
Published Temperature = 98 C Humidity = 42 % to IBM Watson
Published Temperature = 32 C Humidity = 19 % to IBM Watson
Published Temperature = 83 C Humidity = 41 % to IBM Watson
Published Temperature = 53 C Humidity = 5 % to IBM Watson
Published Temperature = 97 C Humidity = 98 % to IBM Watson
Published Temperature = 99 C Humidity = 62 % to IBM Watson
Published Temperature = 77 C Humidity = 5 % to IBM Watson
Published Temperature = 56 C Humidity = 48 % to IBM Watson
Published Temperature = 48 C Humidity = 17 % to IBM Watson
Published Temperature = 57 C Humidity = 60 % to IBM Watson
Published Temperature = 25 C Humidity = 29 % to IBM Watson
Published Temperature = 61 C Humidity = 57 % to IBM Watson
Published Temperature = 35 C Humidity = 8 % to IBM Watson
Published Temperature = 2 C Humidity = 82 % to IBM Watson
Published Temperature = 1 C Humidity = 47 % to IBM Watson
Published Temperature = 97 C Humidity = 14 % to IBM Watson
Published Temperature = 25 C Humidity = 91 % to IBM Watson
Published Temperature = 40 C Humidity = 21 % to IBM Watson
Published Temperature = 82 C Humidity = 47 % to IBM Watson
Published Temperature = 56 C Humidity = 51 % to IBM Watson
Published Temperature = 40 C Humidity = 15 % to IBM Watson
Published Temperature = 35 C Humidity = 85 % to IBM Watson
Published Temperature = 81 C Humidity = 73 % to IBM Watson
```

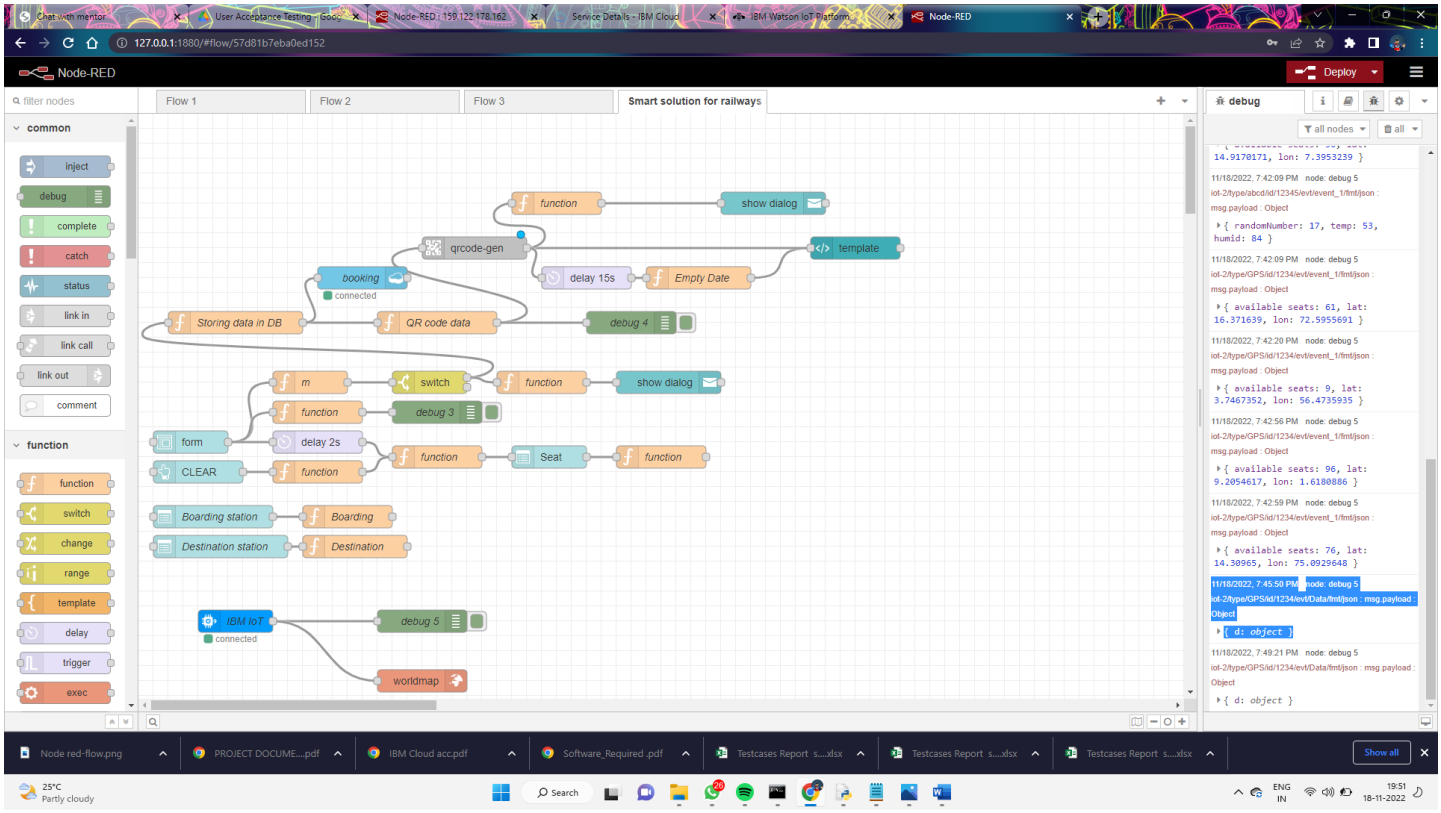
Ln: 25 Col: 0

Ln: 13 Col: 0

# IBM Watson platform code



# Node-Red flow for GPS tracking



# Dashboard Or the Web UI for Ticket confirmation