

## PROJECT DEVELOPMENT PHASE

### DELIVERY OF SPRINT – 2

Date	16 November 2022
Team ID	PNT2022TMID32979
Project Name	Signs with Smart Connectivity for Better Road Safety

#### Sprint goals :

- ✓ Push data from local code to cloud.

#### Program code:

>**brain.py**

#Python code

# IMPORT SECTION STARTS

import weather

from datetime import datetime as dt

# IMPORT SECTION ENDS

```
# -----
```

```
# UTILITY LOGIC SECTION STARTS
```

```
def processConditions(myLocation,APIKEY,localityInfo):  
    weatherData = weather.get(myLocation,APIKEY)  
  
    finalSpeed = localityInfo["usualSpeedLimit"] if "rain" not in weatherData else  
localityInfo["usualSpeedLimit"]/2  
  
    finalSpeed = finalSpeed if weatherData["visibility"]>35 else finalSpeed/2  
  
    if(localityInfo["hospitalsNearby"]):  
        # hospital zone  
  
        doNotHonk = True  
  
    else:  
  
        if(localityInfo["schools"]["schoolZone"]==False):  
            # neither school nor hospital zone  
  
            doNotHonk = False  
  
        else:  
            # school zone  
  
            now = [dt.now().hour,dt.now().minute]  
  
            activeTime = [list(map(int,_.split(":"))) for _ in  
localityInfo["schools"]["activeTime"]]  
  
            doNotHonk = activeTime[0][0]<= now[0]<=activeTime[1][0] and  
activeTime[0][1]<=now[1]<=activeTime[1][1]  
  
            return ({  
"speed" : finalSpeed,
```

```
"doNotHonk" : doNotHonk
})
# UTILITY LOGIC SECTION ENDS
```

**>main.py**

```
# Python code
```

```
# IMPORT SECTION STARTS
```

```
import brain
```

```
# IMPORT SECTION ENDS
```

```
#
```

```
# USER INPUT SECTION STARTS
```

```
myLocation = "Thiruvarur,IN"
```

```
APIKEY = "22aa01c42f0c7e0d38ecb57f4fc65226"
```

```
localityInfo = { "schools" : {  
  "schoolZone" : True,  
  "activeTime" : ["7:00","17:30"] # schools active from 7 AM till 5:30 PM  
  
},  
  "hospitalsNearby" : False, "usualSpeedLimit" : 50 # in km/hr  
}
```

```
# USER INPUT SECTION ENDS
```

```
#
```

```
# MICRO-CONTROLLER CODE STARTS
```

```
print(brain.processConditions(myLocation,APIKEY,localityInfo))
```

```
'''
```

```
MICRO CONTROLLER CODE WILL BE ADDED IN SPRINT 2 AS PER OUR PLANNED  
SPRINT SCHEDULE '''
```

```
# MICRO-CONTROLLER CODE ENDS
```

**>weather.py**

# Python code

import requests as reqs

def get(myLocation,APIKEY):

    apiURL =

    f"https://api.openweathermap.org/data/2.5/weather?q={myLocation}&appid={APIKEY}"

    responseJSON = (reqs.get(apiURL)).json()

    returnObject = {

        "temperature" :

        responseJSON['main']['temp'] - 273.15,

        "weather" : [

            responseJSON['weather'][\_]['main'].lower()

            for \_ in range(len(responseJSON['weather']))

        ],

        "visibility" :

        responseJSON['visibility'] /

        100, # visibility in percentage where 10km is 100% and 0km is 0%

    }

    if("rain" in responseJSON):

        returnObject["rain"] = [

            responseJSON["rain"][key] for key in responseJSON["rain"]

        ]

```
return(returnObject)
```

```
>pushdata.py
```

```
import time
```

```
import sys
```

```
import ibmiotf
```

```
import ibmiotf.device
```

```
import random
```

```
#Provide your IBM Watson Device Credentials
```

```
organization = "q9m3l6"
```

```
deviceType = "mdhar"
```

```
deviceId = "8602"
```

```
authMethod = "token"
```

```
authToken = "12345678"
```

```
# Initialize GPIO
```

```
def myCommandCallback(cmd):
```

```
    print("Command received: %s" % cmd.data['command'])
```

```
    status=cmd.data['command']
```

```
    if status=="switchon":
```

```
        print ("Switch is on")
```

```
    else :
```

```
print ("Switch is off")
```

```
#print(cmd)
```

```
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:
```

```
    #Get Sensor Data from DHT11
```

```
    temperature=28.990000000000001
```

```
    visibility=40
```

```

data = { 'temperature' : temperature, 'visibility': visibility}

#print data

def myOnPublishCallback():

    print ("Published Temperature = %s C" % temperature, "visibility = %s %"
% visibility,"to IBM Watson")

    success = deviceCli.publishEvent("ibmiot", "json", data, qos=0,
on_publish=myOnPublishCallback)

    if not success:

        print("Not connected to IoT")

        time.sleep(1)

deviceCli.commandCallback = myCommandCallback

# Disconnect the device and application from the cloud
deviceCli.disconnect()

```

## **IBM Watson and Python Integration:**

By using Watson IoT Platform, you can collect connected device data and perform analytics on real-time data. The IBM Watson IoT Platform is a fully managed, Cloud-hosted service that provides device management capabilities as well as data collection and management in a time series format.



## # output :

Connected successfully: d:q9m3l6:mdhar:8602

[illegible]