Sprint-2

Date	13 november 2022
Team ID	PNT2022TMID42771
Project Name	Signs with Smart Connectivity for Better Road Safety.

Sprint Goals :

Push data from local code to cloud

Program Code:

> weather.py
This file is a utility function that fetches the weather from OpenWeatherMap. It returns only certain required parameters of the API response.

Python code

```
import requests as reqs
def get(myLocation,APIKEY): apiURL =
f"https://api.openweathermap.org/data/2.5/weather?q={myLocation}&appid={API KEY}"
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)
```

> **publishData.py**

This code pushes data to the cloud and logs data. IBM Cloud is configured such that the data is displayed in the following website: CLICK TO OPEN NODE RED DASHBOARD

Python code

```
# IMPORT SECTION STARTS
import wiotp.sdk.device # python -m pip install wiotp import time
# IMPORT SECTION ENDS
#
# API CONFIG SECTION STARTS
 import wiotp.sdk.device # python -m pip install wiotp
import time

# IMPORT SECTION ENDS
# ---------------------------------------------
# API CONFIG SECTION STARTS

myConfig = {
    "identity" : {
        "orgId" : "gsqz5f",
        "typeId" : "NANDY",
        "deviceId" : "12345"
    },
    "auth" : {
        "token" : "9876543210"
    }
}

# API CONFIG SECTION ENDS
# ---------------------------------------------
# FUNCTIONS SECTION STARTS

def myCommandCallback(cmd):
    print("recieved cmd : ",cmd)


def logData2Cloud(location,temperature,visibility):
    client = wiotp.sdk.device.DeviceClient(config=myConfig,logHandlers=None)
    client.connect()
    client.publishEvent(eventId="status",msgFormat="json",data={
        "temperature" : temperature,
        "visibility" : visibility,
        "location" : location
    },qos=0,onPublish=None)
    client.commandCallback = myCommandCallback
    client.disconnect()
    time.sleep(1)
# FUNCTIONS SECTION ENDS
```

brain.py >

This file is a utility function that returns only essential information to be displayed at the hardware side and abstracts all the unnecessary details. This is where the code flow logic is implemented.

```
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

})
```

main.py

The code that runs in a forever loop in the micro-controller. This calls all the until functions from other python files and based on the return value transduces changes in the output hardware display.

```
# IMPORT SECTION STARTS

import brain

# IMPORT SECTION ENDS

# ---------------------------------------------

# USER INPUT SECTION STARTS

myLocation = "Chennai,IN"

APIKEY = "bf4a8d480ee05c00952bf65b78ae826b"

localityInfo = {

 "schools" : {

 "schoolZone" : True,

 "activeTime" : ["7:00","17:30"] # schools active from 7 AM till 5:30 PM

 },

 "hospitalsNearby" : False,
```

"usualSpeedLimit" : 40 # in km/hr

}

USER INPUT SECTION ENDS

--

MICRO-CONTROLLER CODE STARTS

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

'''

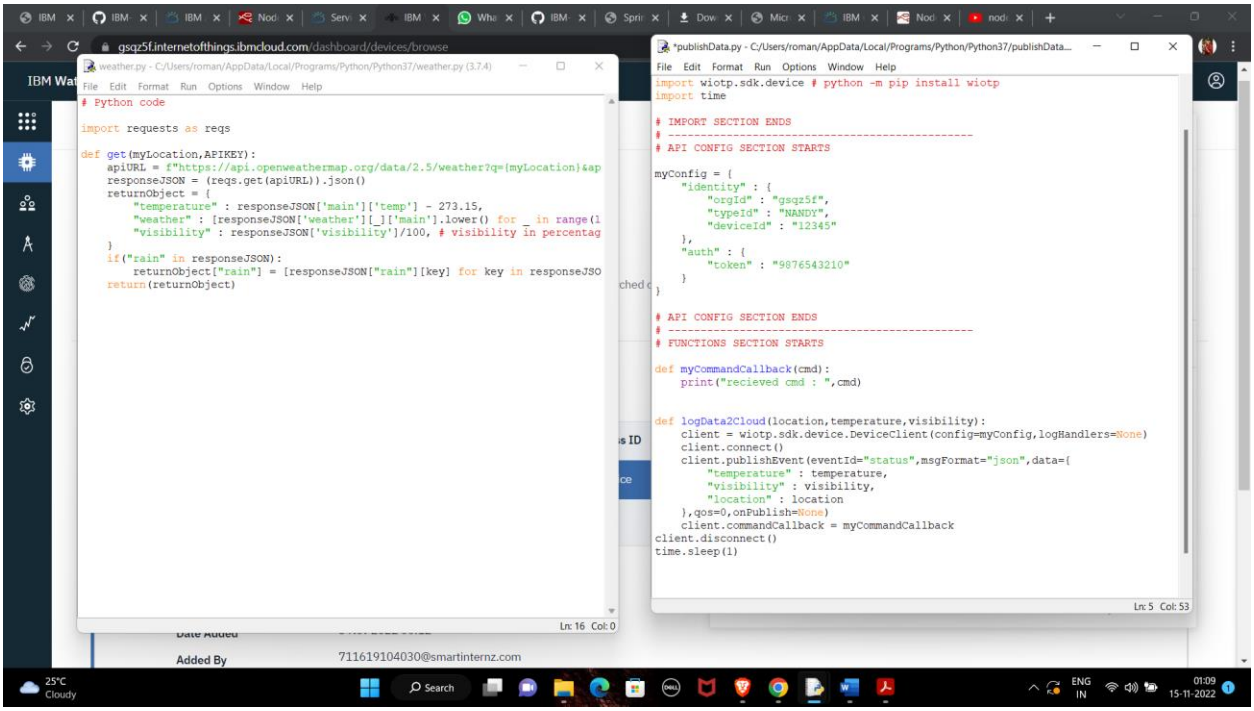
MICRO CONTROLLER CODE WILL BE ADDED IN SPRINT 2 AS PER OUR PLANNED SPRINT SCHEDULE

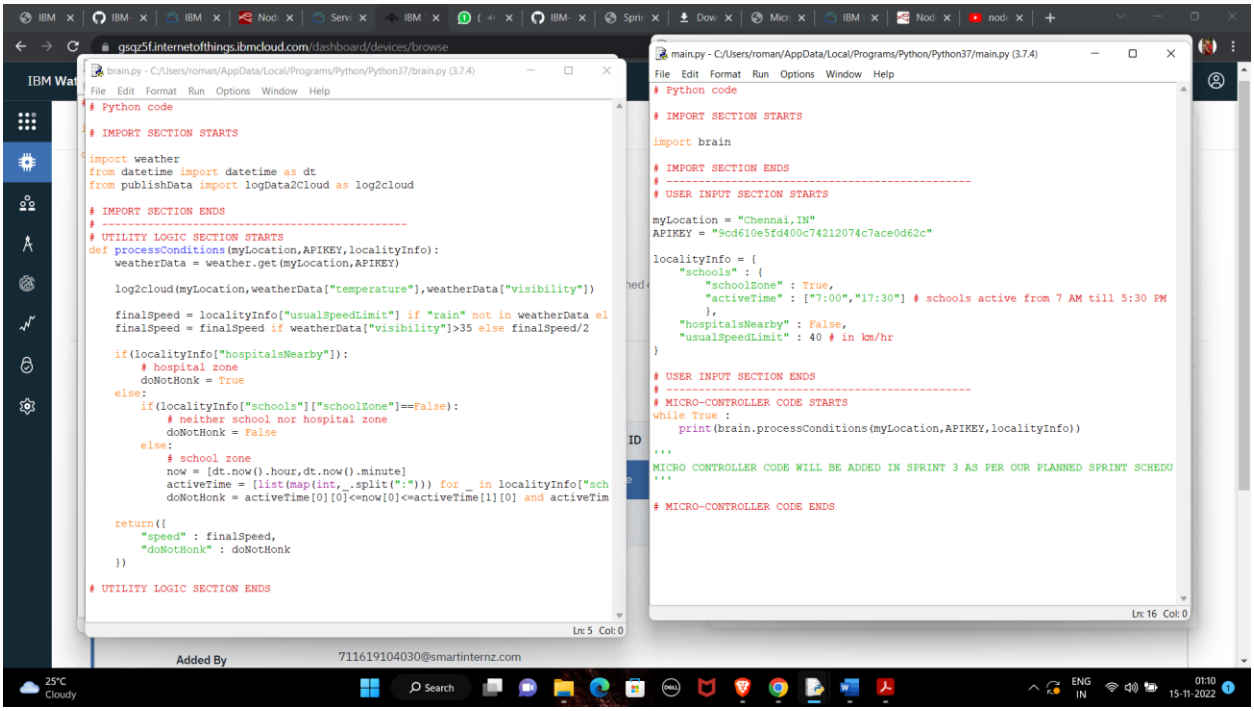
'''

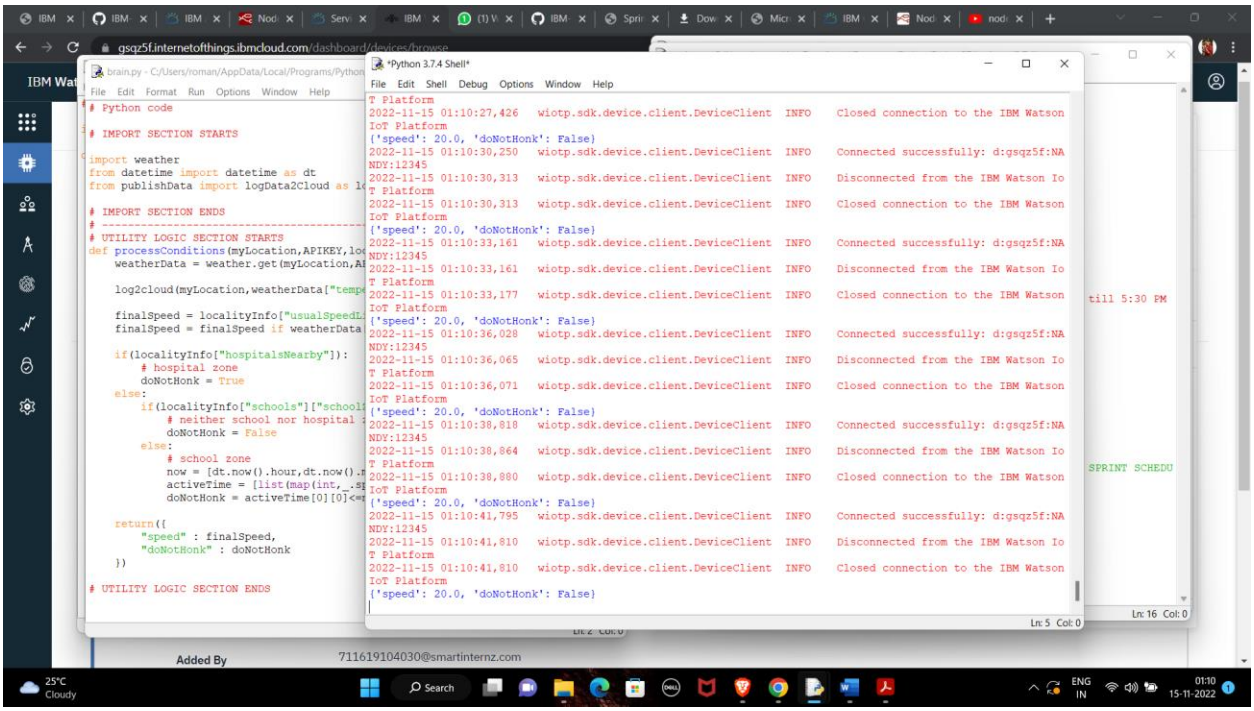
Output :

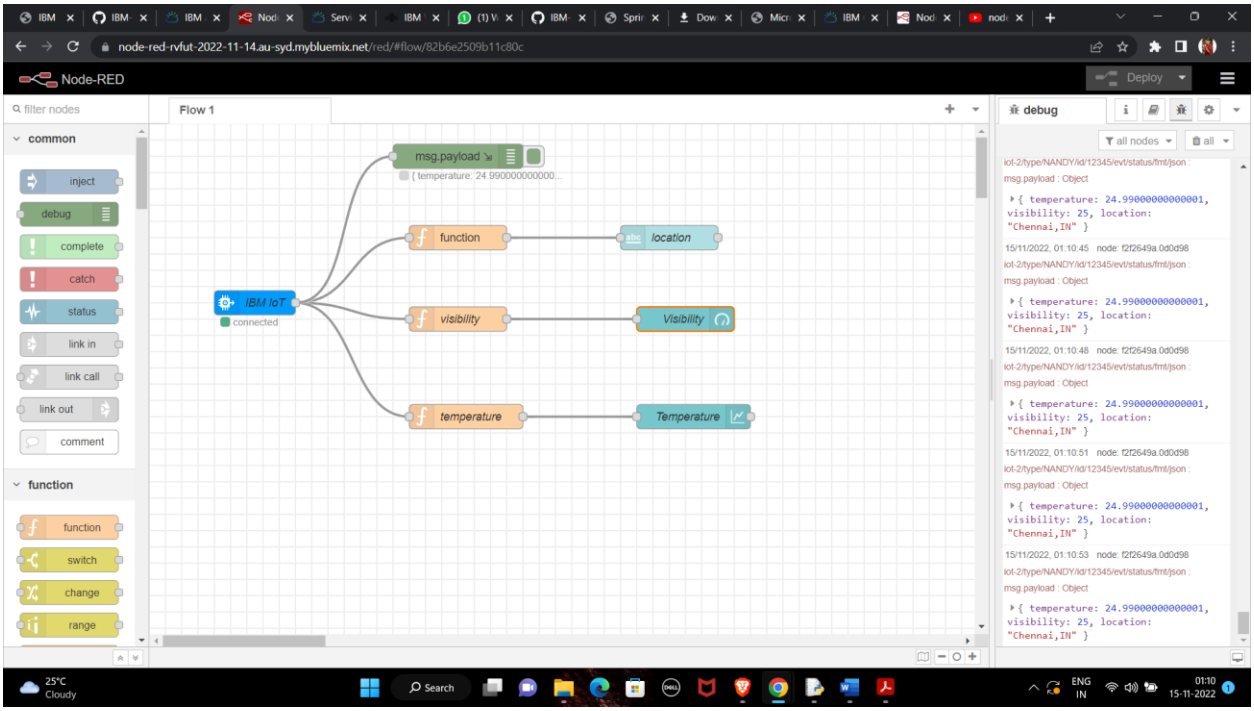
```python

# Code Output {'speed':20.0 , 'doNotHonk': False}

```

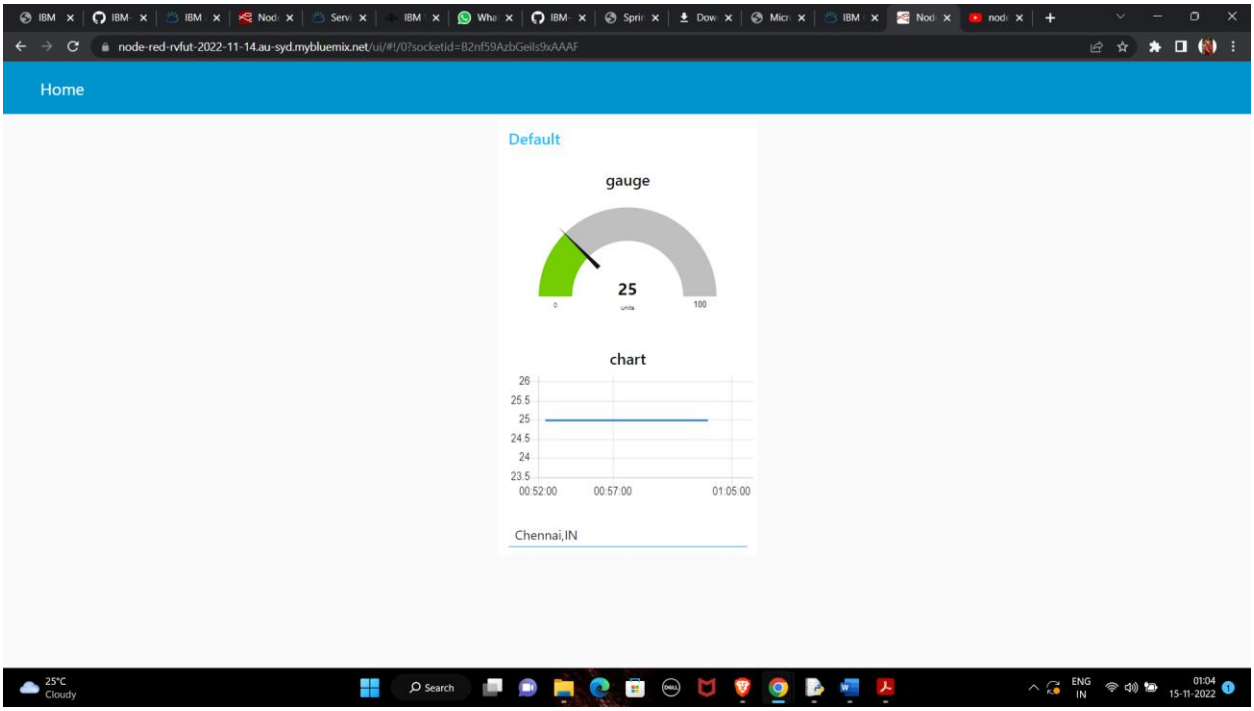
Images:

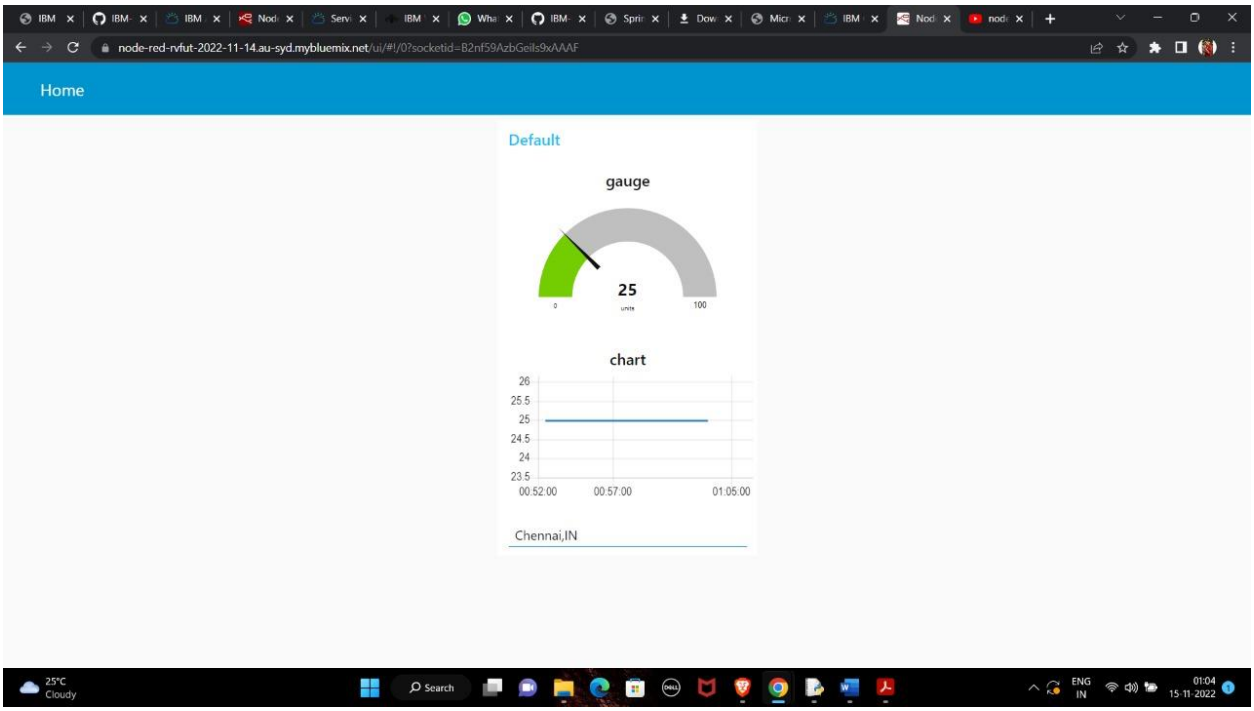

gsqz5f.internetofthings.ibmcloud.com/dashboard/devices/browse
weather.py - C:/Users/roman/AppData/Local/Programs/Python/Python37/weather.py (3.7.4)
# Python code
import requests as reqs
def get(myLocation,APIKEY):
apiURL = f"https://api.openweathermap.org/data/2.5/weather?q={myLocation}&ap
responseJSON = (reqs.get(apiURL)).json()
returnObject = {
"temperature" : responseJSON['main']['temp'] - 273.15,
"weather" : [responseJSON['weather'][_]['main'].lower() for _ in range(l
"visibility" : responseJSON['visibility']/100, # visibility in percentag
}
if("rain" in responseJSON):
returnObject["rain"] = [responseJSON["rain"][key] for key in responseJSO
return(returnObject)
*publishData.py - C:/Users/roman/AppData/Local/Programs/Python/Python37/publishData...
import wiotp.sdk.device # python -m pip install wiotp
import time
# IMPORT SECTION ENDS
# API CONFIG SECTION STARTS
myConfig = {
"identity" : {
"orgId" : "gsqz5f",
"typeId" : "NANDY",
"deviceId" : "12345"
},
"auth" : {
"token" : "98765432l0"
}
}
# API CONFIG SECTION ENDS
# FUNCTIONS SECTION STARTS
def myCommandCallback(cmd):
print("recieved cmd : ",cmd)
def logData2Cloud(location,temperature,visibility):
client = wiotp.sdk.device.DeviceClient(config=myConfig,logHandlers=None)
client.connect()
client.publishEvent(eventId="status",msgFormat="json",data={
"temperature" : temperature,
"visibility" : visibility,
"location" : location
},qos=0,onPublish=None)
client.commandCallback = myCommandCallback
client.disconnect()
time.sleep(1)
Added By
711619104030@smartinternz.com


gsqz5f.internetofthings.ibmcloud.com/dashboard/devices/browse
brain.py - C:/Users/roman/AppData/Local/Programs/Python/Python37/brain.py (3.7.4)
# Python code
# IMPORT SECTION STARTS
import weather
from datetime import datetime as dt
from publishData import logData2Cloud as log2cloud
# IMPORT SECTION ENDS
# UTILITY LOGIC SECTION STARTS
def processConditions(myLocation,APIKEY,localityInfo):
weatherData = weather.get(myLocation,APIKEY)
log2cloud(myLocation,weatherData["temperature"],weatherData["visibility"])
finalSpeed = localityInfo["usualSpeedLimit"] if "rain" not in weatherData el
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["sch
doNotHonk = activeTime[0][0]<=now[0]<=activeTime[1][0] and activeTim
return({
"speed" : finalSpeed,
"doNotHonk" : doNotHonk
})
UTILITY LOGIC SECTION ENDS
main.py - C:/Users/roman/AppData/Local/Programs/Python/Python37/main.py (3.7.4)
# Python code
# IMPORT SECTION STARTS
import brain
# IMPORT SECTION ENDS
# USER INPUT SECTION STARTS
myLocation = "Chennai,IN"
APIKEY = "9cd610e5fd400c74212074c7ace0d62c"
localityInfo = {
"schools" : {
"schoolZone" : True,
"activeTime" : ["7:00","17:30"] # schools active from 7 AM till 5:30 PM
},
"hospitalsNearby" : False,
"usualSpeedLimit" : 40 # in km/hr
}
# USER INPUT SECTION ENDS
# MICRO-CONTROLLER CODE STARTS
while True :
print(brain.processConditions(myLocation,APIKEY,localityInfo))
'''
MICRO CONTROLLER CODE WILL BE ADDED IN SPRINT 3 AS PER OUR PLANNED SPRINT SCHEDU
'''
MICRO-CONTROLLER CODE ENDS
Added By
711619104030@smartinternz.com


Python 3.7.4 Shell
File Edit Shell Debug Options Window Help
T Platform
2022-11-15 01:10:27,426 wiotp.sdk.device.client.DeviceClient INFO Closed connection to the IBM Watson
IoT Platform
{'speed': 20.0, 'doNotHonk': False}
2022-11-15 01:10:30,250 wiotp.sdk.device.client.DeviceClient INFO Connected successfully: d:gsqz5f:NA
NDY:12345
2022-11-15 01:10:30,313 wiotp.sdk.device.client.DeviceClient INFO Disconnected from the IBM Watson Io
T Platform
2022-11-15 01:10:30,313 wiotp.sdk.device.client.DeviceClient INFO Closed connection to the IBM Watson
IoT Platform
{'speed': 20.0, 'doNotHonk': False}
2022-11-15 01:10:33,161 wiotp.sdk.device.client.DeviceClient INFO Connected successfully: d:gsqz5f:NA
NDY:12345
2022-11-15 01:10:33,161 wiotp.sdk.device.client.DeviceClient INFO Disconnected from the IBM Watson Io
T Platform
2022-11-15 01:10:33,177 wiotp.sdk.device.client.DeviceClient INFO Closed connection to the IBM Watson
IoT Platform
{'speed': 20.0, 'doNotHonk': False}
2022-11-15 01:10:36,028 wiotp.sdk.device.client.DeviceClient INFO Connected successfully: d:gsqz5f:NA
NDY:12345
2022-11-15 01:10:36,065 wiotp.sdk.device.client.DeviceClient INFO Disconnected from the IBM Watson Io
T Platform
2022-11-15 01:10:36,071 wiotp.sdk.device.client.DeviceClient INFO Closed connection to the IBM Watson
IoT Platform
{'speed': 20.0, 'doNotHonk': False}
2022-11-15 01:10:38,818 wiotp.sdk.device.client.DeviceClient INFO Connected successfully: d:gsqz5f:NA
NDY:12345
2022-11-15 01:10:38,864 wiotp.sdk.device.client.DeviceClient INFO Disconnected from the IBM Watson Io
T Platform
2022-11-15 01:10:38,880 wiotp.sdk.device.client.DeviceClient INFO Closed connection to the IBM Watson
IoT Platform
{'speed': 20.0, 'doNotHonk': False}
2022-11-15 01:10:41,795 wiotp.sdk.device.client.DeviceClient INFO Connected successfully: d:gsqz5f:NA
NDY:12345
2022-11-15 01:10:41,810 wiotp.sdk.device.client.DeviceClient INFO Disconnected from the IBM Watson Io
T Platform
2022-11-15 01:10:41,810 wiotp.sdk.device.client.DeviceClient INFO Closed connection to the IBM Watson
IoT Platform
{'speed': 20.0, 'doNotHonk': False}
Ln: 5 Col: 0
Added By
711619104030@smartinternz.com


Node-RED
Deploy
Flow 1
common
inject
debug
complete
catch
status
link in
link call
link out
comment
function
function
switch
change
range
msg.payload
{ temperature: 24.990000000000...
function
location
IBM IoT
connected
visibility
Visibility
temperature
Temperature
debug
all nodes
msg.payload : Object
{ temperature: 24.99000000000001, visibility: 25, location: "Chennai,IN" }
15/11/2022, 01:10:45 node: f2f2649a.0d0d98
15/11/2022, 01:10:48 node: f2f2649a.0d0d98
15/11/2022, 01:10:51 node: f2f2649a.0d0d98
15/11/2022, 01:10:53 node: f2f2649a.0d0d98


node-red-rvfut-2022-11-14.au-syd.mybluemix.net/ui/#!/0?socketid=B2nf59AzbGeils9xAAAF
Home
Default
gauge
25
0
100
chart
26
25.5
25
24.5
24
23.5
00:52:00
00:57:00
01:05:00
Chennai,IN
25°C
Cloudy
Search
ENG
IN
01:04
15-11-2022


node-red-rvfut-2022-11-14.au-syd.mybluemix.net/ui/#!/0?socketid=B2nf59AzbGeils9xAAAF
Home
Default
gauge
25
0
100
chart
26
25.5
25
24.5
24
23.5
00:52:00
00:57:00
01:05:00
Chennai,IN
25°C
Cloudy
Search
ENG
IN
01:04
15-11-2022