Sprint-01

Signs with Smart Connectivity for Better Road Safety

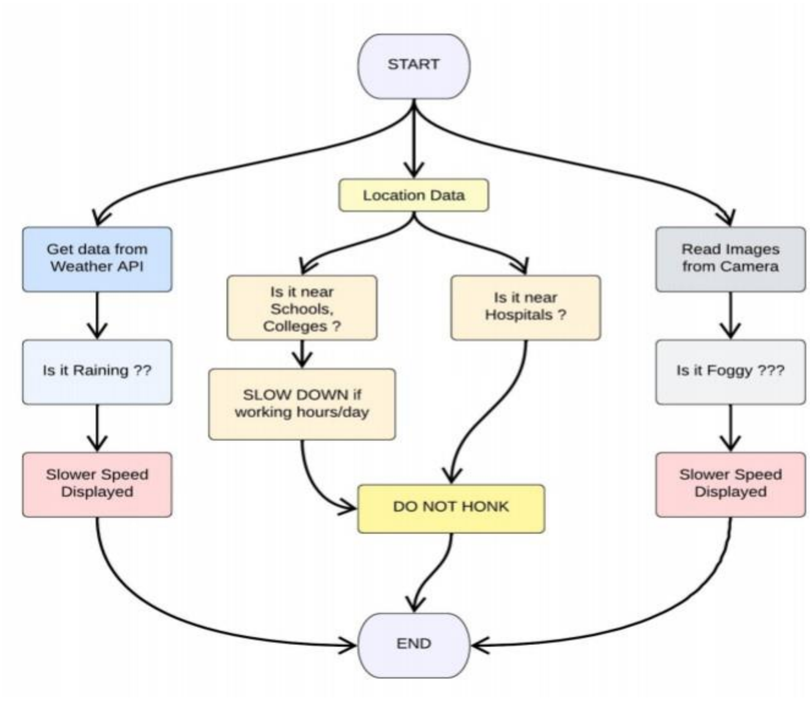
TeamID-PNT2022TMID09351

###SprintGoals:

1. Create and initialize accounts in various public
APIs like Open Weather API.

2. Write a Python program that outputs results
given the inputs like weather and location.

###CodeFlow :



Program Code

:(./weather.py)

This file is a utility function that fetches the weather from OpenWeatherAPI. It returns only certainrequiredparametersof theAPI response.

```python
#Pythoncode


importrequestsasreqs


defget(myLocation,APIKEY):
   apiURL =
   f"https://api.openweathermap.org/data/2.5/weather?q={myLocation}&appid={APIKEY}"response
   JSON= (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%
and0kmis0%
   }
   if("rain"inresponseJSON):
      returnObject["rain"] = [responseJSON["rain"][key] for key in
   responseJSON["rain"]]return(returnObject)
```

(./brain.py)

> This file is a utility function that returns only essential information to be displayed at the hardwareside andabstracts alltheunnecessarydetails.Thisiswherethecodeflow logicisimplemented.

```python
#Pythoncode


#IMPORTSECTIONSTARTS


importweather
fromdatetime importdatetimeasdt
```

```
#IMPORTSECTIONENDS

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

#UTILITYLOGICSECTIONSTARTS

defprocessConditions(myLocation,APIKEY,localityInfo):

    weatherData=weather.get(myLocation,APIKEY)


    finalSpeed = localityInfo["usualSpeedLimit"] if "rain" not in weatherData
elselocalityInfo["usualSpeedLimit"]/2

    finalSpeed=finalSpeedifweatherData["visibility"]>35elsefinalSpeed/2


    if(localityInfo["hospitalsNearby"]):

        # hospital

        zonedoNotHonk=True

    else:

        if(localityInfo["schools"]["schoolZone"]==False):

            # neither school nor hospital

            zonedoNotHonk= False

        else:

            # schoolzone

            now=[dt.now().hour,dt.now().minute]

            activeTime = [list(map(int,_.split(":"))) for _ in

            localityInfo["schools"]["activeTime"]]doNotHonk=activeTime[0][0]<=now[0]<=activ

            eTime[1][0] and
activeTime[0][1]<=now[1]<=activeTime[1][1]


    return({

        "speed" :

        finalSpeed,"doNotHonk":doNo

        tHonk

    })


#UTILITYLOGICSECTIONENDS

```

(./main.py)

> The code that runs in a forever loop in the micro-controller. This calls all the util functions fromotherpythonfilesandbasedonthereturnvaluetransduceschangesintheoutput hardwaredisplay.

```python
#Pythoncode


#IMPORTSECTIONSTARTS


importbrain


#IMPORTSECTIONENDS
#-----------------------------------------------------------------
#USERINPUTSECTIONSTARTS


myLocation="Chennai,IN"
APIKEY="bf4a8d480ee05c00952bf65b78ae826b"


localityInfo =
  {"schools":{
     "schoolZone":True,
     "activeTime":["7:00","17:30"]#schoolsactivefrom7AMtill5:30PM
     },
  "hospitalsNearby" :
  False,"usualSpeedLimit":40#inkm
  /hr
}


#USERINPUTSECTIONENDS
#-----------------------------------------------------------------
#MICRO-CONTROLLERCODESTARTS
```

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

'''

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

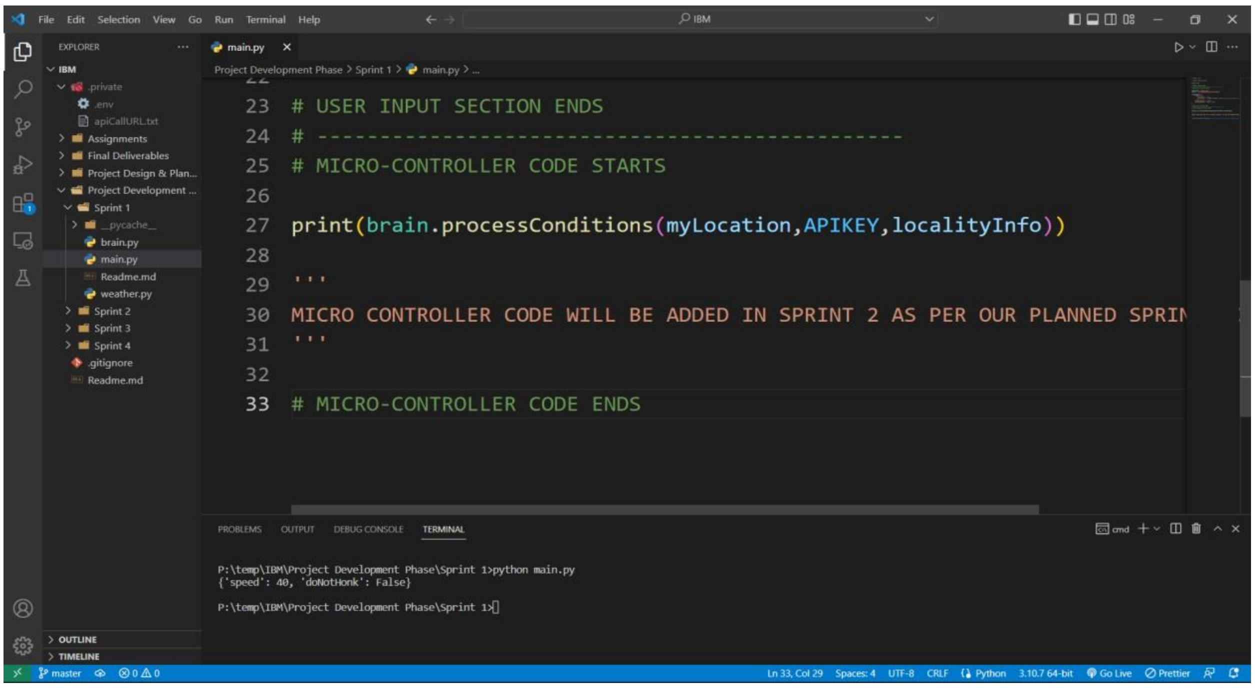
SCHEDULE'''

#MICRO-CONTROLLERCODEENDS

Output:

```python
#CodeOutput
{'speed':40,'doNotHonk':False}
```

Images:



```
# 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 SPRIN
'''

# MICRO-CONTROLLER CODE ENDS
```

```
P:\temp\IBM\Project Development Phase\Sprint 1>python main.py
{'speed': 40, 'doNotHonk': False}

P:\temp\IBM\Project Development Phase\Sprint 1>
```

ThankYou