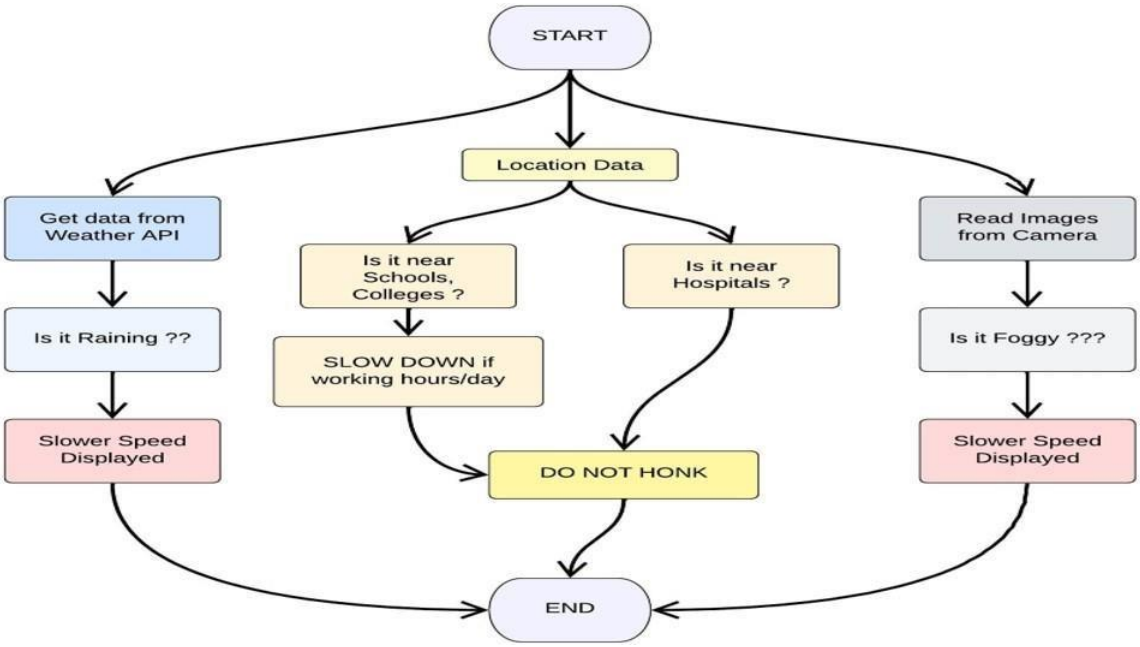
SPRINT 01

Date	27 October 2022
Team ID	PNT2022TMID31413
Project Name	Project – Signs with Smart Connectivity for Better Road Safety

SPRINT GOALS:

1. Create and initialize accounts in various public APIs like OpenWeather API.
2. Write a Python program that outputs results given the inputs like weather and location.

CODE FLOW:



PROGRAM CODE:

Weather.py

This file is a utility function that fetches the weather from

OpenWeatherAPI. It returns only certain required parameters of the API response.

```
# Python code import
requests as reqs def
  get(myLocation,APIKE
  Y): apiURL =
f"https://api.openweathermap.org/data/2.5/weather?q={myLocation}&ap
pi d={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
```

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.

```
# 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

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

```
# Python code

# 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
'''
```

MICRO-CONTROLLER CODE ENDS

OUTPUT:

Code Output

{'speed': 40, 'doNotHonk': False}

IMAGES: