PROJECT DOCUMENTATION

Date	17 November 2022
Team ID	PNT2022TMID53946
Project Name	Signs with Smart Connectivity for Better Road Safety

1. INTRODUCTION

1.1 Project Overview

The road signs and the speed limits are Static. But the road signs can be changed in some cases. We can consider some cases when there are some road diversions due to heavy traffic or due to accidents then we can change the road signs accordingly if they are digitalized. This project proposes a system which has digital sign boards on which the signs can be changed dynamically. If there is rainfall then the roads will be slippery and the speed limit would be decreased. These smart connected sign boards get the speed limitations from a web app using weather API and update automatically. There is a web app through which you can enter the data of the road diversions, accident prone areas and the information sign boards can be entered through web app. This data is retrieved and displayed on the sign boards accordingly.

1.2 Purpose

- To improve vehicle safety by providing real-time traffic information to the driver. Road signs play an important role in road safety.
- To be effective, road signs must be visible at a distance that enables drivers to take the necessary actions.
- To avoid the accidents, due to the increasing vehicle speed.
- To provides a voice alert to a needed action that enhances driver's attention.

2. LITERATURE SURVEY

2.1 **Existing Problem**

- Increased traffic can increase carbon emissions and other pollution.
- Land use for roads can damage built and natural environment, impose mortality on wildlife if habitats are severed.
- construction has associated environmental costs.

2.2 **Reference**

W.E.Marshall, “Understanding international road safety disparities: Why is Australia so much safer than the United States?” Accident Analysis & Prevention, vol. 111, pp. 251–265, 2018

Digital signage is evolving to smart signage which provides personalized service by adaptively changing contents according to the user context. However, previous smart signage services have difficulty to expend their service because it is not easy to connect additional sensing devices. Furthermore, previous smart signage systems only consider single signage for a service. In this paper, we propose IoT based smart signage platform. The proposed platform provides IoT based connectivity between sensors and signage platform for flexible service extension. Also, we suggest IoT based signage connection, status sensing, and controlling. Therefore, our platform can make a service group of signages dynamically and enables signages to collaborate for a service in wide area. To show the performance of proposed platform, we implemented smart nursing home service. The service shows that IoT devices and signages can be connected to the platform dynamically and collaborate together for a service in wide area.

This paper presents an approach to detect traffic signs using You Only Look Once version 4 (YOLOv4) model. The traffic sign detection and recognition system (TSDR) play an essential role in the intelligent transportation system (ITS). TSDR can be utilized for driver assistance and, eventually, driverless cars to reduce

accidents. When driving an automobile, the driver's attention is usually drawn to the road. On the other hand, most traffic signs are situated on the side of the road, which may have contributed to the collision. TSDR allows drivers to view traffic sign information without having to divert their attention. Due to the existence of a large background, clutter, fluctuating degrees of illumination, varying sizes of traffic signs, and changing weather conditions, TSDR is an important but difficult process in intelligent transport systems. Many efforts have been made to find answers to the major issues that they face. The objective of this study addresses road traffic sign detection and recognition using a technique that initially detects the bounding box of a traffic sign. Then the detected traffic sign will be recognized for usage in a speeded-up process. Since safe driving necessitates real-time traffic sign detection, the YOLOv4 network was employed in this research. YOLOv4 was evaluated on our dataset, which consisted of manual annotations to identify 43 distinctive traffic signs classes. It was able to achieve an average recognition accuracy of 84.7%. Overall, the work adds by presenting a basic yet effective model for real-time detection and recognition of traffic signs.

Ubiquitous nature of smart cities requires multiple technologies to be implemented in this area. To develop the smart cities in practice, there is huge need of “Smart Traffic Management”. Smart Traffic Management is a system to monitor and control the traffic signals using sensors to regulate the flow of traffic and to avoid the congestion for smooth flow of traffic. Prioritizing the traffic like ambulance, police etc. is also one application comes under smart traffic management. Traffic sign board plays important role to make the traffic in shape and to control and manage the traffic on roads. Many at times the driver misses the sign boards while driving due to various reasons like insufficient light, fog, rain, traffic etc. In this paper, a framework of the Smart Traffic Sign Boards (STSB) is proposed, which can communicate with the system deployed in all the vehicles to make the drivers of those vehicles aware of speed breakers, speed

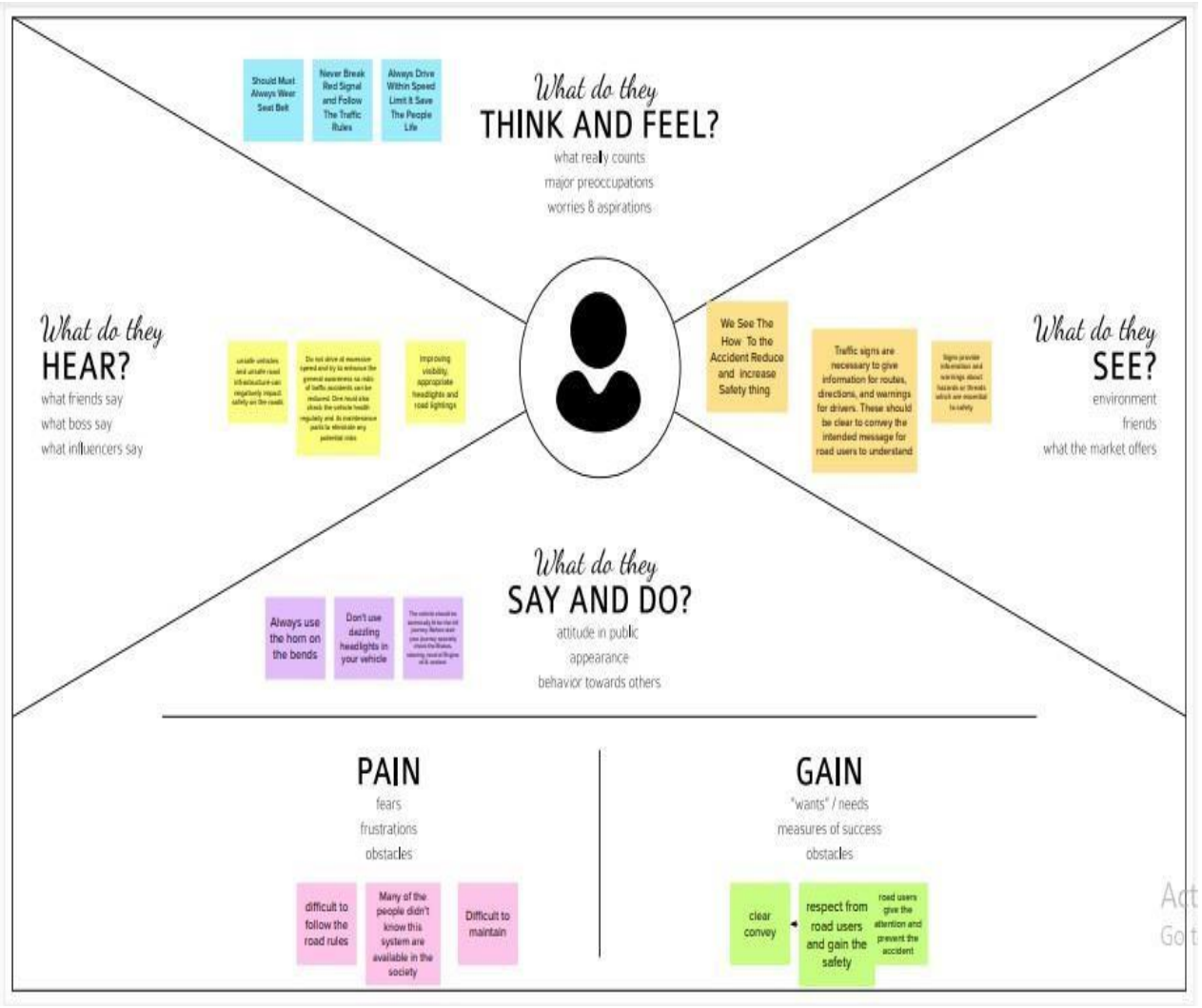
limits, schools, or ‘U’ turn ahead, etc. beforehand, to avoid the mishap due to sudden appearing of such unusual features of the road during the road journey.

2.3 **Problem Statement Definition:**

- A driver who wants to drive safely on road but there are many obstacles because of heavy traffic, weather condition, etc..,
- A driver who wants to avoid the heavy traffic roads but they are unpredictable because they change from time to time.
- A passenger who wants to travel safely but there are many road accidents because of some drivers who drive very fast and carelessly.
- A driver who wants to reach the destination but unable to choose the route and turn in wrong direction because there are no navigation instructions.

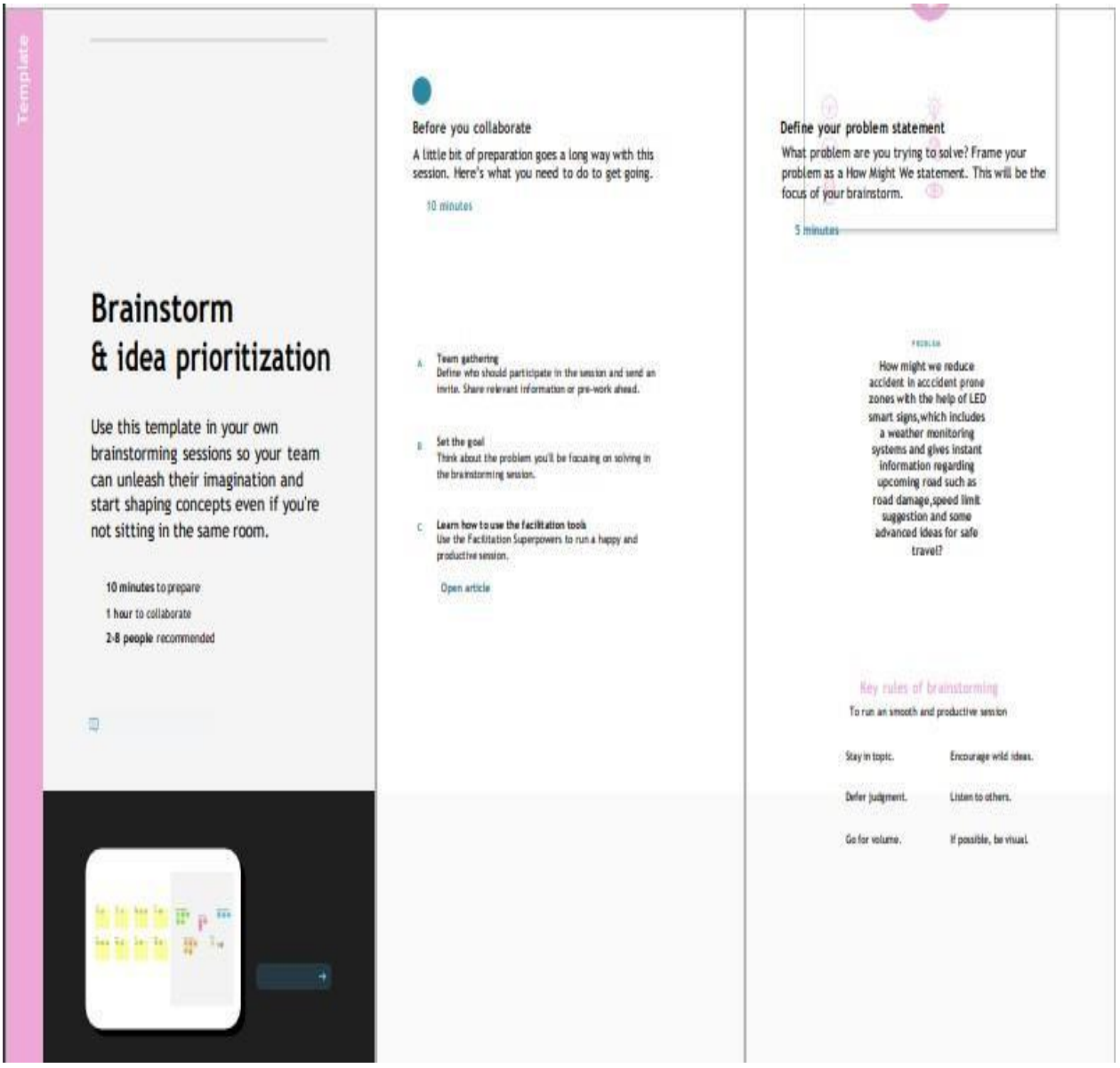
3. **IDEATION AND PROPOSED SOLUTION:**

3.1 **Empathy Map Canvas:**

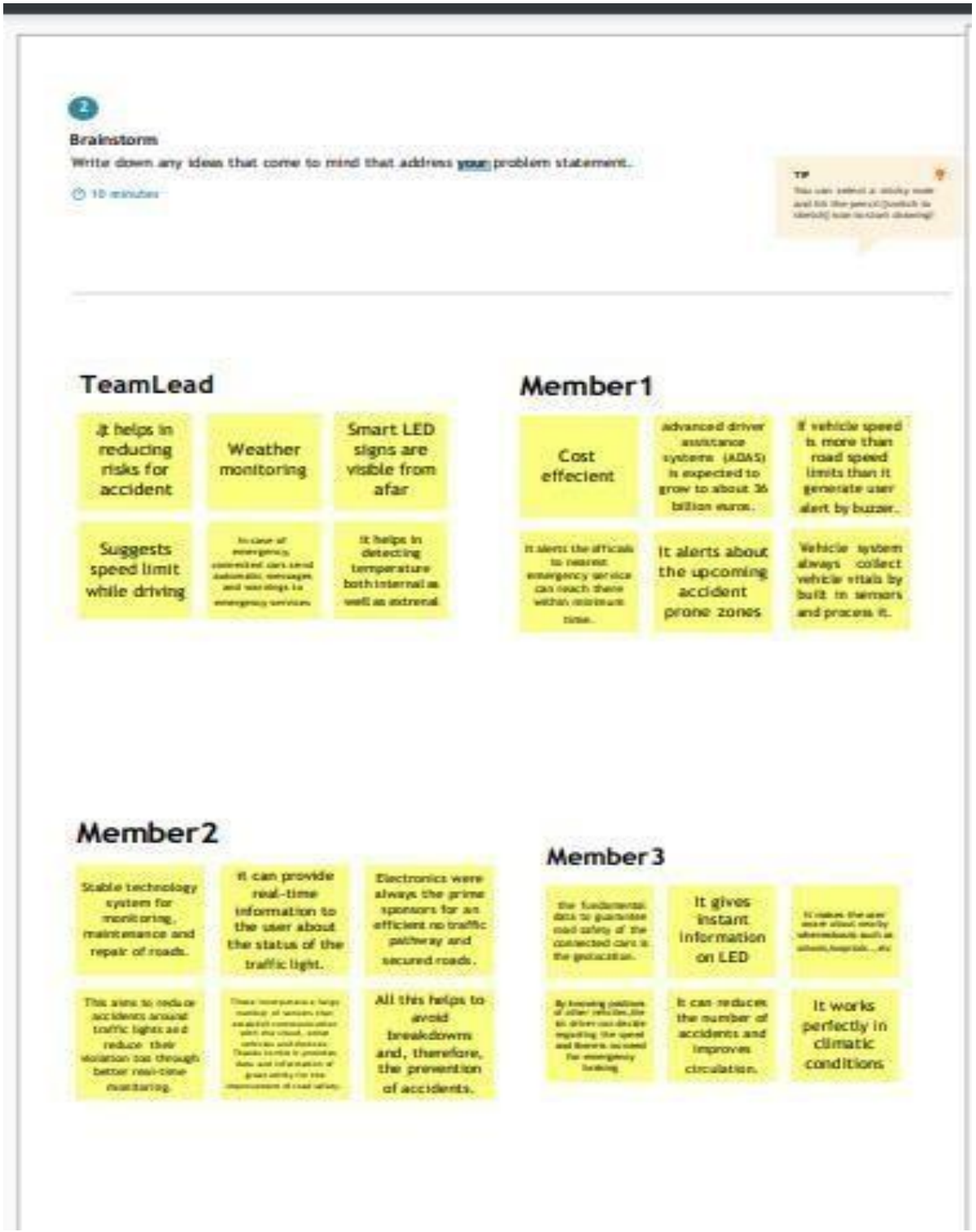


3.2 **Ideation & Brainstorming:**

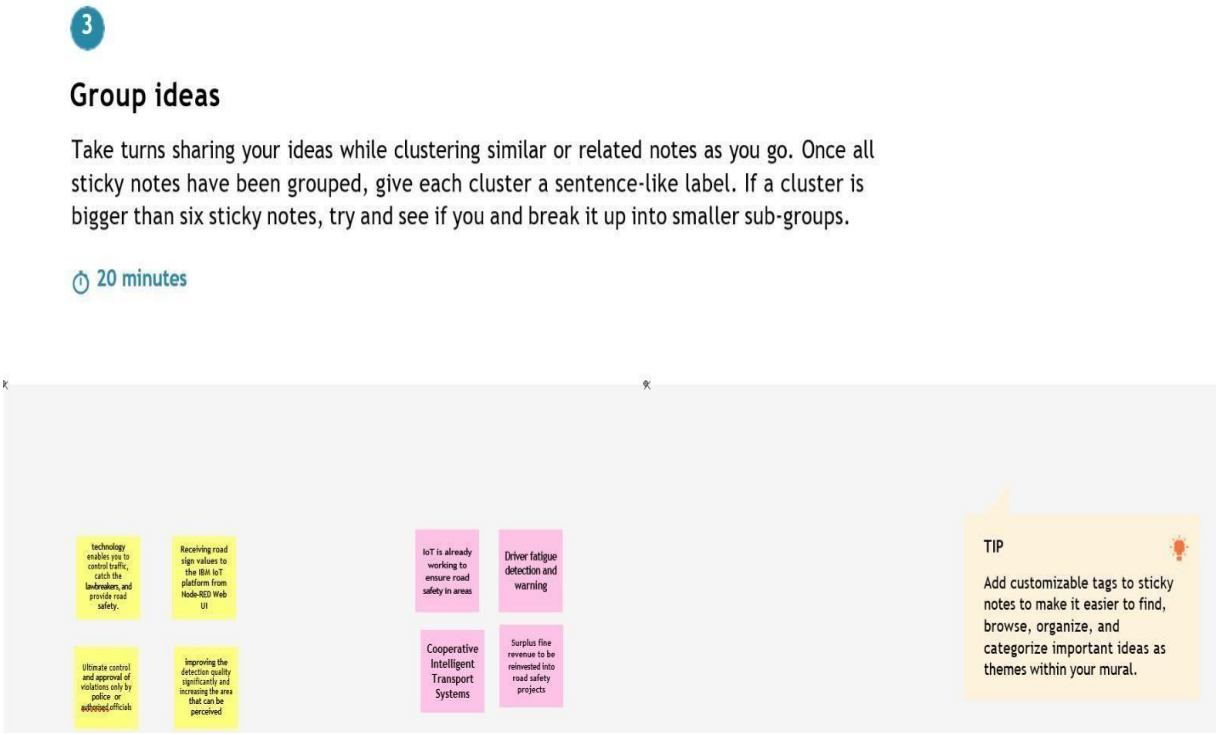
Step-1: Team Gathering, Collaboration and Select the Problem Statement.



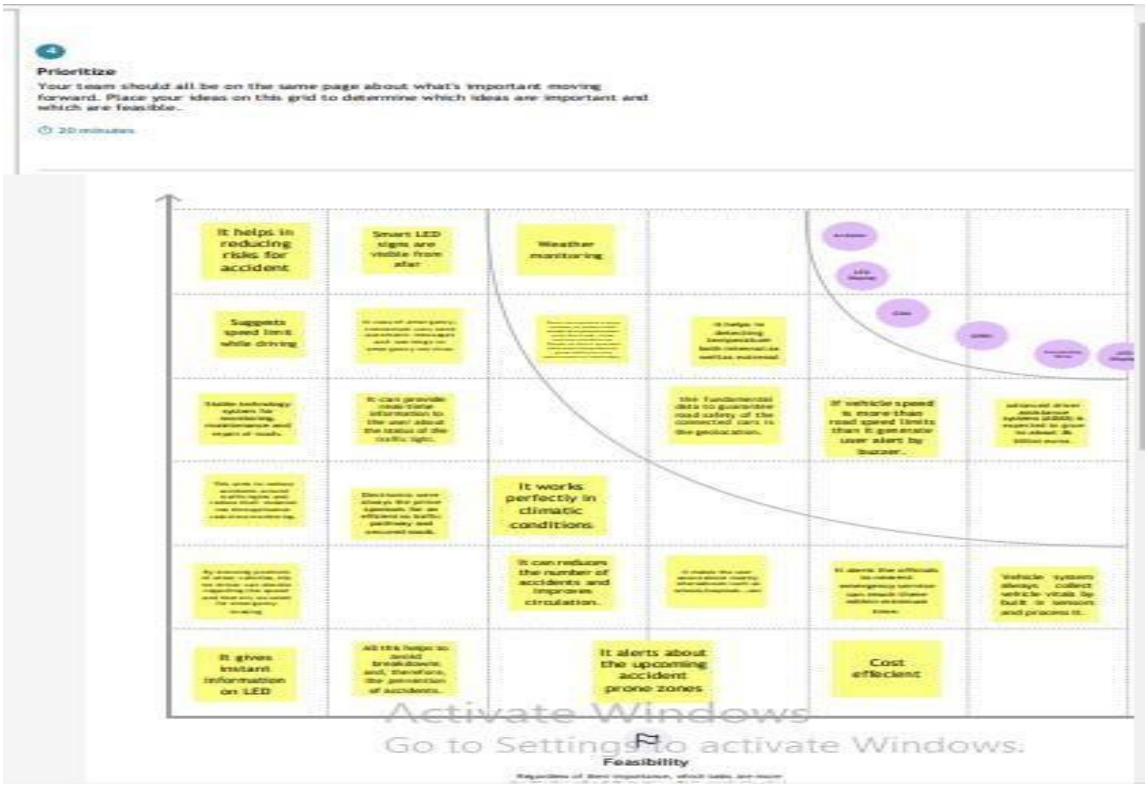
Step-2: Brainstorm, Idea Listing and Grouping



Step-3: Idea Prioritization



Prioritize:

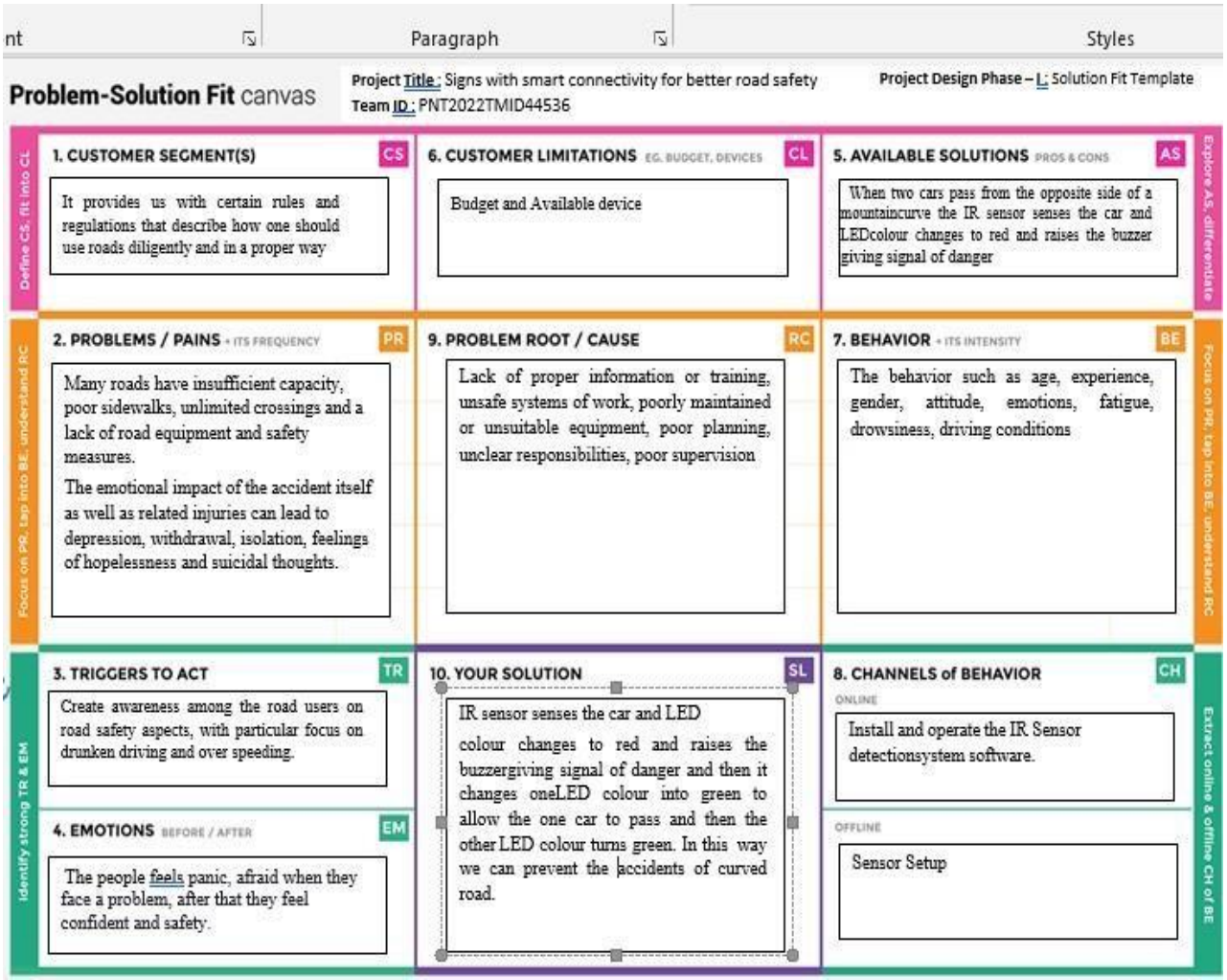


3.3 Proposed Solution:

S.No	Parameter	Description
1.	Problem Statement (Problem to besolved)	The problems in these curve roads are that the drivers are not able to see the vehicle or obstacles coming from another end of the curve. If the vehicle is in great speed, then it is difficult to control and there are chances of falling off a cliff. Hence there is a need of many road safety systems.
2.	Idea / Solution description	To avoid these problems in curve roads of mountain areas, Nevon projects has proposed this vehicle accident prevention system. This accident prevention system using sensors is powered by Arduino board, it consists of IR sensors, LED lights, and buzzer. When two cars pass from the opposite side of a mountain curve the IR sensor senses the car and LED colour changes to red and raises the buzzer giving signal of danger and then it changes oneLED colour into green to allow the one car to pass and then the other LED colour turns green. In this way we can prevent the accidents of curved road.

3.	Novelty / Uniqueness	One or more of the fundamental data to guarantee road safety of the connected cars is the geolocation. The connected cars can communicate with each other so that, depending on the speed and position of each vehicle, collisions are avoided like maneuvers involving emergency braking.
4.	Social Impact / Customer Satisfaction	Ensuring safe driving experience with real-time assistance, navigation, and even monitoring driving patterns and any emergency situation. Additionally, along with the state of the traffic, IoT drivers can receive updated information on the state of the roads potholes, grade changes, black spots, etc..
5.	Business Model (Revenue Model)	We can introduce product-based approach to earn a good revenue . Adding precise, low-cost, alwaysconnected IoT sensors and monitoring devices to the products that you sell and install enhances the types of services.
6.	Scalability of the Solution	The IOT applications must have the ability to support an increasing number of connected devices, users, application features, and analytics.

3.4 Problem Solution Fit:



4. REQUIREMENT ANALYSIS:

4.1 Functional Requirements:

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)

FR-1	USER REGISTRATION	• Through google forms • Through mail • Through linked in • Through Facebook
FR-2	USER CONFIRMATION	• Through verification mails • Through OTP
FR-3	USER APPROVAL	• Through mails • Through phone calls • Through SMS
FR-4	USER TRANSACTION	• Through net banking • Through UPI
FR-5	TESTING	• Testing through components • Testing through API and UI
FR-6	END RESULT	• End result through product features • By using the technology

4.2 Non-Functional Requirements:

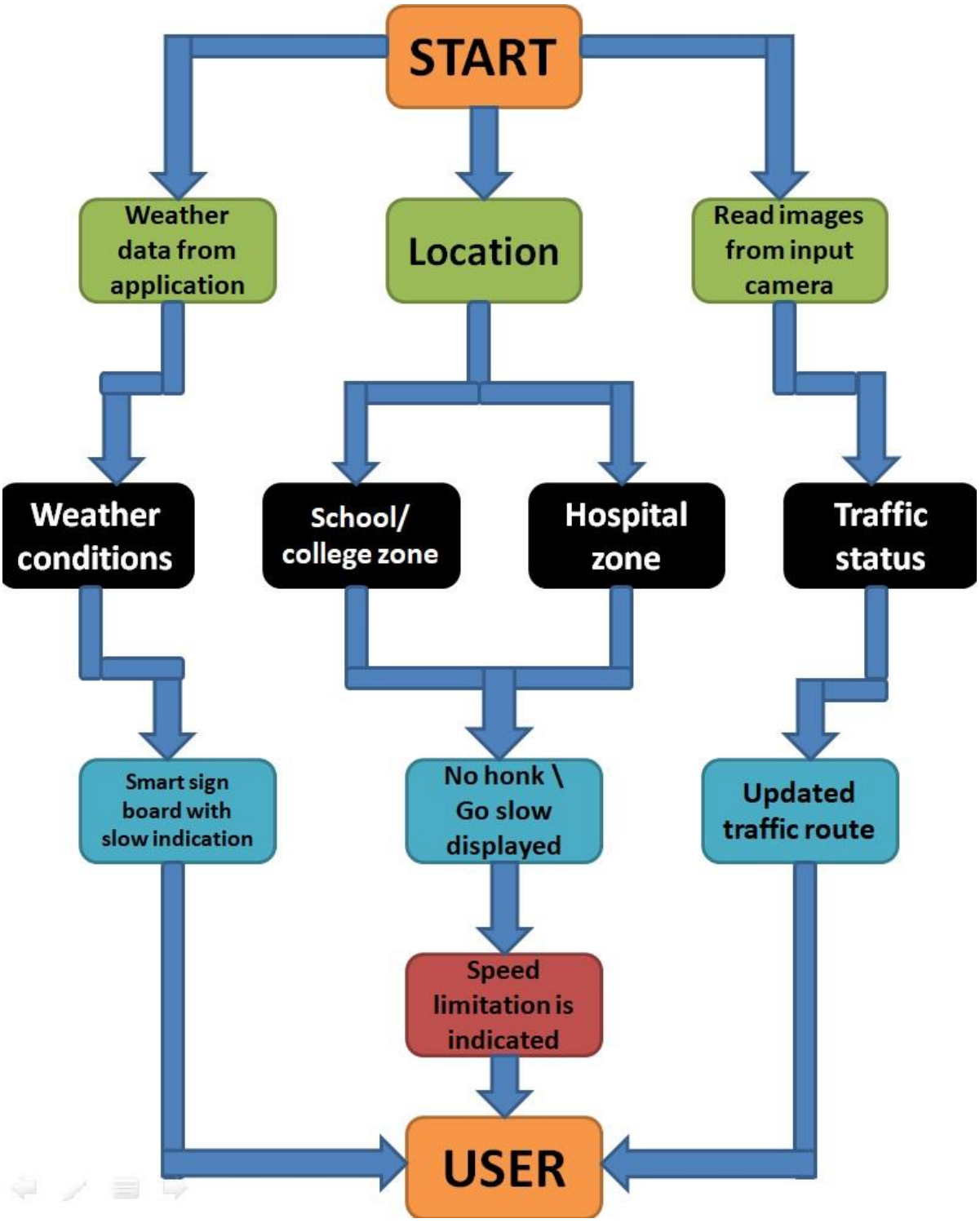
FR No.	**Non Functional Requirement**	**Description**

NFR-1	Usability	• Situations never remain the same. Therefore, there must be a constant check of the conditions prevailing and accordingly there must be changes made in these boards. • Sign boards with caution or alerts must be placed well in advance so that the drivers could be more alert with the journey. • The text content must be available in different languages to help the drivers. • Boards must be large and clear for better visibility. • Sign boards should be bright coloured so that it catches the drivers' sight. • The illustrations or the symbols used in the boards must be easily understandable.
NFR-2	Security	• The security system should be strong enough that no one can modify it other than the authority. • No one should be able to enter into the network to change, delete or manage the intimations or messages delivered through the sign boards.
NFR-3	Reliability	• There should not be any miscommunications or confusions regarding the messages displayed. • Maximumaccuracy must be ensured.

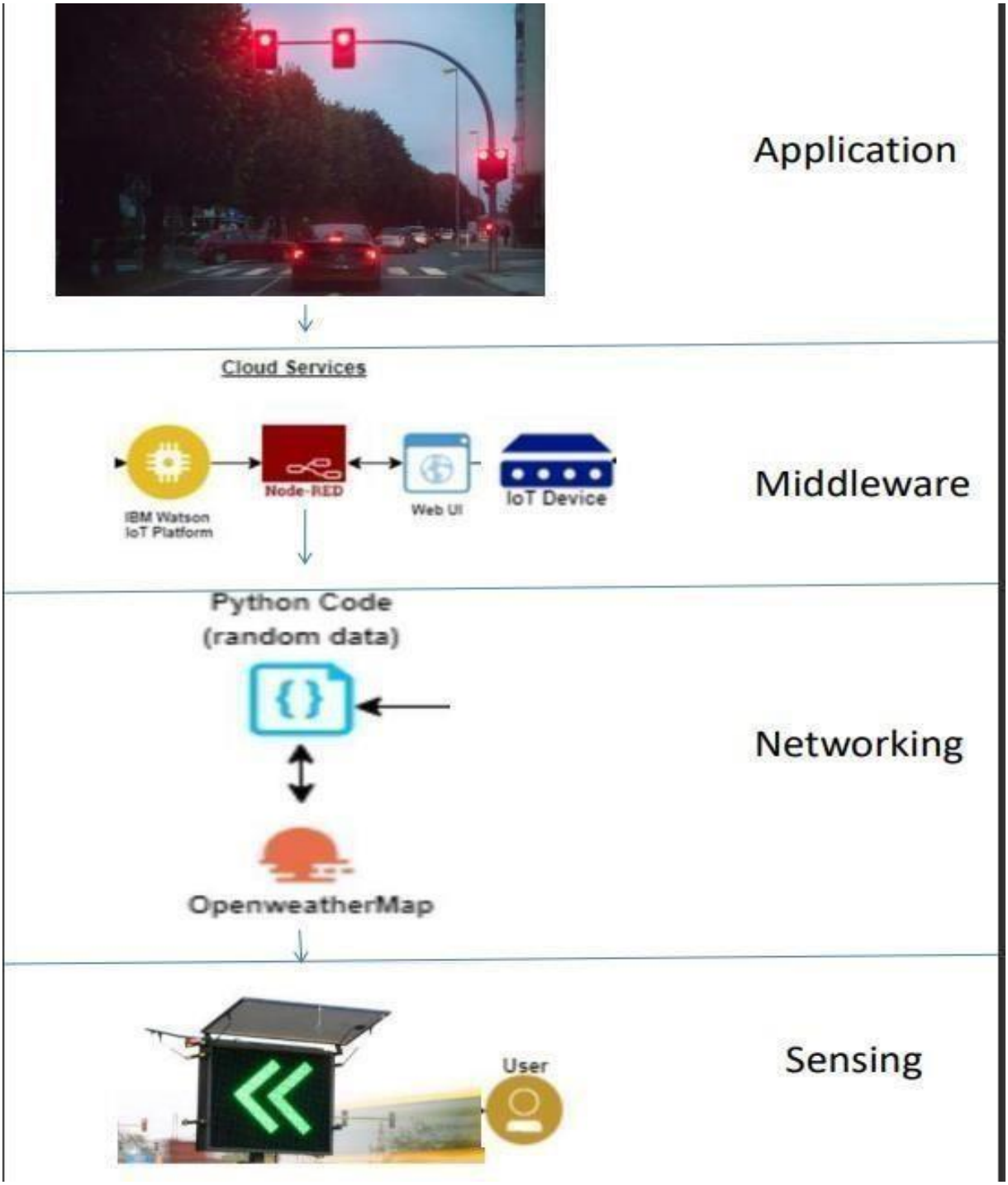
		• All the information displayed must be checked periodically and updated if any changes are needed for error-free intimation.
NFR-4	Performance	• The efficiency and the accuracy of the information hence calculated should be maximum • It should be ensured that minimum amount of energy, time and cost is required for the operation.
NFR-5	Availability	• These should be available anytime and everywhere that is 24/7. • Sign boards should be located in places which has direct view from the road. It should not present amidst bushes, trees, building etc • It should be properly monitored that no sign boards are damaged, repaired or malfunctioning at any time. • The sign boards should be made available only in places where they are required the most. Frequent availability of boards may lead to confusion and mistakes.
NFR-6	Scalability	• It should be easy to scale according to the requirement. • It should be in such a way that the network at any time of period should be ready to beexpanded and implemented on a wider scale.

5. PROJECT DESIGN:

5.1 Data Flow Diagrams:



5.2 Solution & Technical Architecture:



5.3 User Stories:

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)

FR-1	User Visibility	Sign Boards should be made with LED's which are bright coloured and are capable of attracting the drivers attention but it should also not be too distracting or blinding cause it may lead to accidents.
FR-2	User Need	The smart sign boards should be placed frequently in places it is needed and less in places where it is not needed much to avoid confusion for the user during travel.
FR-3	User Understanding	For better understanding of the driver, the signs should be big, clear and legible and it can also include illustrations which will make it easily understandable to the driver.
FR-4	User Convenience	The display should be big enough that it should even be visible from far distance clearly.

6. CODING & SOLUTIONING:

Code Explanation:

Libraries:

Including all libraries like json, random, time, sys, ibmiotf etc.

PROJECTFINALDND.py - D:/1ibm/PROJECTFINALDND.py (3.7.0)

File Edit Format Run Options Window Help

```
import requests #importing a library
import json
import ibmiotf.application
import ibmiotf.device
import time
import random
import sys
```

Credentials:

Entering all the credentials corresponding to IoT Watson device in order to publish data to it.

```
# watson device details

organization = "2s7yy7"
devicType =  "project"
deviceId = "projectid"
authMethod= "token"
authToken= "projecttoken"
```

MIT Inventor Interruption:

Receiving commands as inputs when buttons are pressed in MIT inventor in order to perform separate functions.

```
def myCommandCallback(cmd):
    global a
    #print("command recieved:%s" %cmd.data['command'])
    #status=cmd.data['command']
    print("command recieved:%s" %cmd.data['command'])
    control=cmd.data['command']
    print(control)


try:
        deviceOptions={"org": organization, "type": devicType,"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()
```

Exception Handling:

To handle exception if occurs while connecting with IBM IOT WATSON device

```
try:
        deviceOptions={"org": organization, "type": devicType,"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()
```

Main Body:

- Connecting to IBM IoT device.
- Getting temperature and humidity values in json format from open weather map as inputs.
- Accessing the values using their corresponding keys.
- Generating random values for distance since hardware sensors are not implemented.
- Passing a warning “stating please slow down” when humidity is less than 100 in order to promote safe driving experience.
- Passing instruction when distance is less than 20 in order to avoid accidents and clashes.

PROJECTFINALDND.py - D:/1ibm/PROJECTFINALDND.py (3.7.0)

File Edit Format Run Options Window Help

```
#connect and send a datapoint "temp" with value integer value into the cloud as a type of event for every 10 seconds
deviceCli.connect()

while True:

#get sensor data from DHT11



    a = "https://api.openweathermap.org/data/2.5/weather?q=Chennai,%20IN&appid=e2bea247ed9ad643a04d9a8e55499d5f"
    r=requests.get(url=a)
    data=r.json()

    Temp= data['main']['temp']
    Humd= data['main']['humidity']
    data= {'temp':Temp,'humid':Humd}
    dist=random.randint(0,50)
    dis={'dista':dist}

    if(Humd<100):
        warn={'alert':'PLEASE SLOW DOWN!!!!!!'}
    if(dist<20):
        insta={'inst':'stop'}
```

Publish Data To IBM IOT WATSON Platform:

Passing all the data(temperature, humidity, warning, instruction) to ibm iot watson.

Disconnecting the connection established with IoT Watson device.

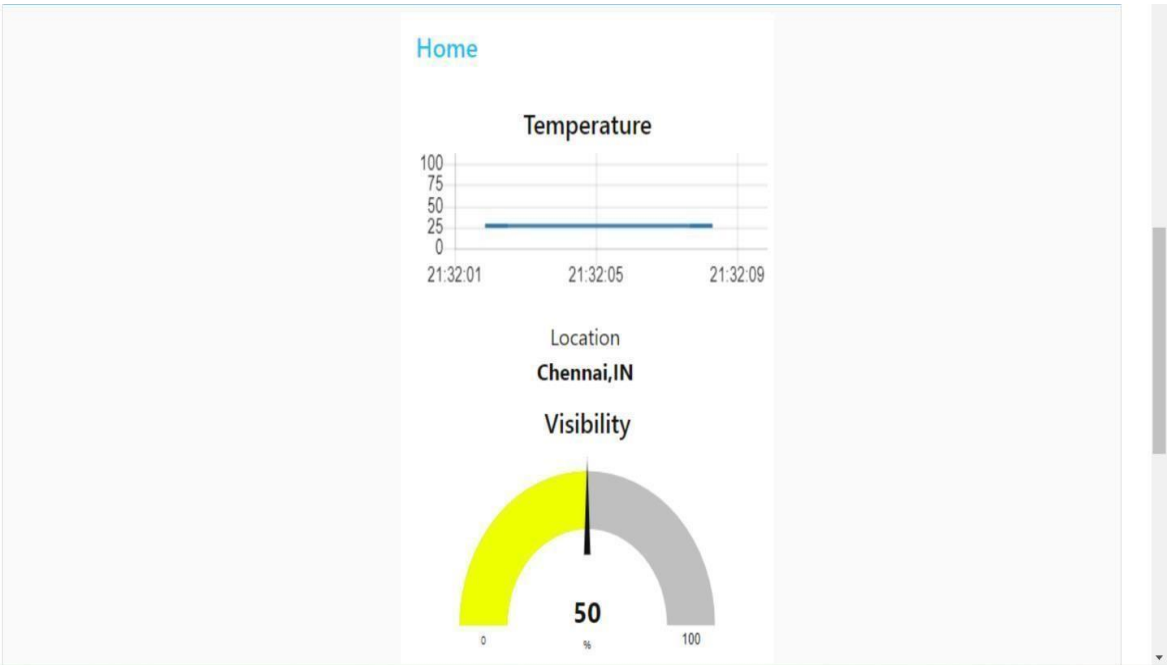
PROJECTFINALDND.py - D:/1ibm/PROJECTFINALDND.py (3.7.0)

File Edit Format Run Options Window Help

```
    data= {'temp':Temp,'humid':Humd}
    dist=random.randint(0,20)
    dis={'dista':dist}

    if(Humd<100):
        warn={'alert':'PLEASE SLOW DOWN!!!!!!'}
    if(dist<20):
        insta={'inst':'stop'}


    def myOnPublishCallback():

        print("published Temperature = %s c" %Temp,"humidity:%s %%" %Humd)
        print(warn)
        print(dis)
        print(insta)







    success=deviceCli.publishEvent ("IoTSensor","json",insta,qos=0,on_publish= myOnPublishCallback)
    success=deviceCli.publishEvent ("IoTSensor","json",data,qos=0,on_publish= myOnPublishCallback)
    success=deviceCli.publishEvent ("IoTSensor","json",warn,qos=0,on_publish= myOnPublishCallback)
    success=deviceCli.publishEvent ("IoTSensor","json",dis,qos=0,on_publish= myOnPublishCallback)

    if not success:
        print("not connected to ibmiot")
    time.sleep(5)



    deviceCli.commandCallback=myCommandCallback
#disconnect the device
deviceCli.disconnect()
```

7.1 **Feature 1:** WEATHER UPDATE AND CORRESPONDING COMMAND:

Getting temperature and humidity from Open Weather Map for a particular city and displaying warning regarding the speed when humidity is below 100.

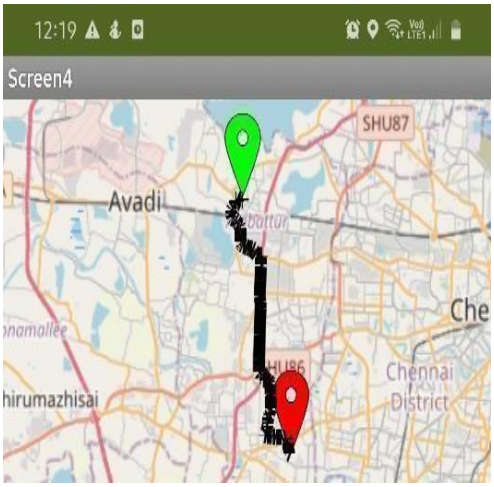


7.2 **Feature 2:** SPEED DETECTION

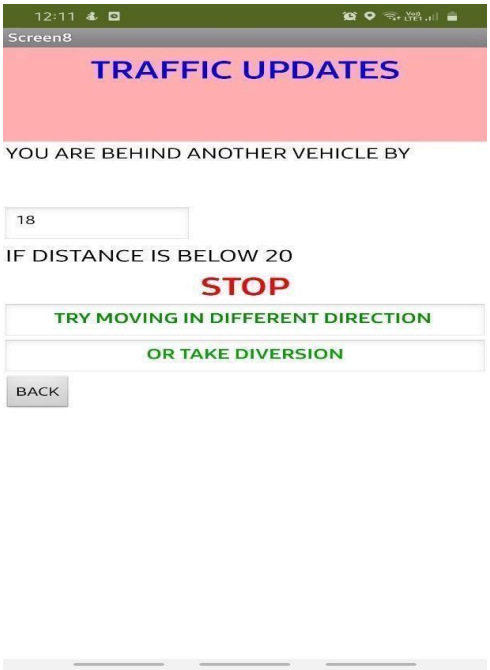
- By implementing a location sensor in MIT APP INVENTOR, with changes in the location with respect to time, speed can easily be detected and displayed in the app to the user.
- This requires location settings from user's phone to be active.
- An image of normal speed limit is also displayed which means that, travelling within that range would be safe.

7.3 **Feature 3:** MAP AND NAVIGATION

- By implementing same location sensor,current location can be detected. This also requires location settings in user's phone to be active.
- By dragging the green marker to start location and red marker to the destination location to be reached and clicking on the navigate button, displays the street path that connects the start and end point specified.
- In addition to this, it also displays the directions to be followed to reach the destination.



7.4 **Feature 4:** ZONAL CLASSIFICATION:



- Here, displays few sign boards indicating different zones like school zone, hospital zone, railway track etc. By clicking on the button below the sign displaysthe meaning and instruction to befollowed in the particular region.
- This provides the user with better understanding about the sign boards and to act accordingly.

7.4 Feature 5: DETERMINING TRAFFIC:

- Since hardware sensors are not implemented, we have used random function to generate values for the distance between the user and the vehicle ahead.
- If the distance is below 20, it instructs the driver or the user to stop immediately and try moving forward with different direction or to take diversion.

```
dist=random.randint(0,50)
dis={'dista':dist}

if(dist<20):
   insta={'inst':'stop'}
```

8. TESTING

8.1 Test Cases

A test case documents strategy that will be used to verify and ensure that a product or system meets its design specification and other requirements. A test

case is usually prepared by or with significant input from the engineer. This document describes the plans for testing the architectural prototype of System. In my Project the system has to be tested to get the Desired Output. I use different speed for testing the system.

8.2 User Acceptance Testing

In engineering and its various sub disciplines, acceptance testing is black-box testing performed on a system (e.g. software, lots of manufactured mechanical parts, or batches of chemical products) prior to its delivery. It is also known as functional testing, black-box testing, release acceptance, QA testing, application testing, confidence testing, final testing, validation testing, or factory acceptance testing.

In software development, acceptance testing by the system provider is often distinguished from acceptance testing by the customer (the user or client) prior to accepting transfer of ownership. In such environments, acceptance testing performed by the customer is known as user acceptance testing (UAT). This is also known as end-user testing, site (acceptance) testing, or field (acceptance) testing.

A smoke test is used as an acceptance test prior to introducing a build to the main testing process. Acceptance test cards are ideally created during sprint planning or iteration planning meeting, before development begins so that the developers have a clear idea of what to develop. Sometimes (due to bad planning!) acceptance tests may span multiple stories (that are not implemented in the same sprint) and there are different ways to test them out during actual sprints.

One popular technique is to mock external interfaces or data to mimick other stories which might not be played out during an iteration (as those stories may have been relatively lower business priority). A user story is not considered complete until the acceptance tests have passed.

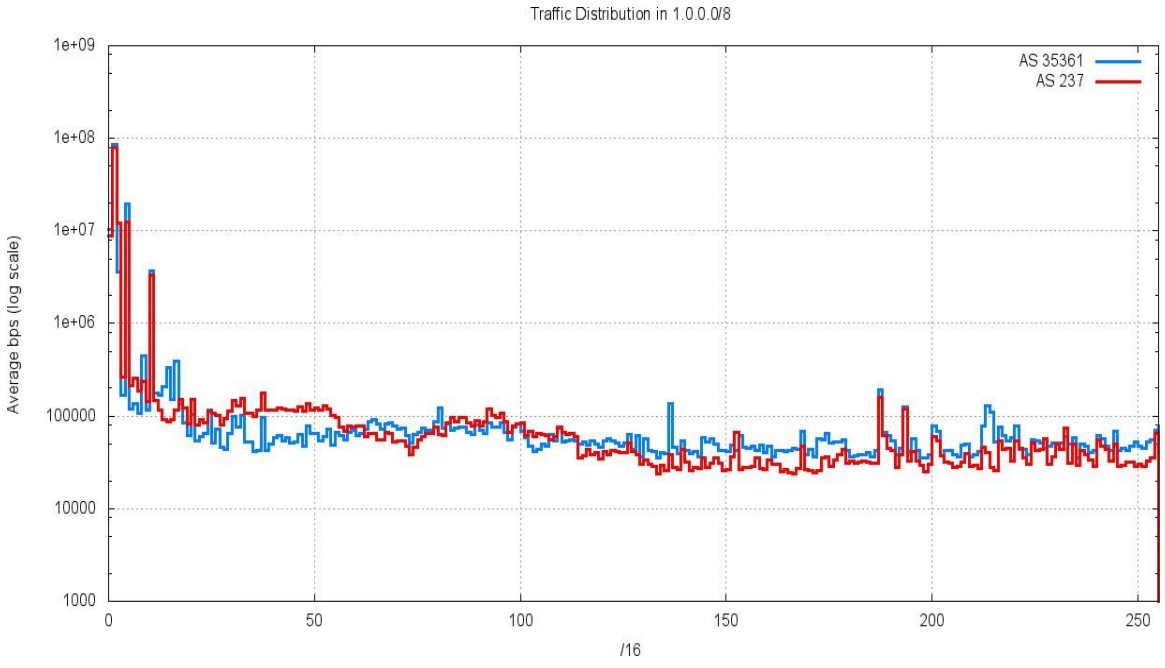
The acceptance test suite is run against the supplied input data or using an acceptance test script to direct the testers. Then the results obtained are compared with the expected results. If there is a correct match for every case, the test suite is said to pass. If not, the system may either be rejected or accepted on conditions previously agreed between the sponsor and the manufacturer.

The objective is to provide confidence that the delivered system meets the business requirements of both sponsors and users. The acceptance phase may also act as the final quality gateway, where any quality defects not previously detected may be uncovered.

In these testing procedures the project is given to the customer to test whether all requirements have been fulfilled and after the user is fully satisfied. The project is perfectly ready. If the user makes request for any change and if they found any errors those all errors has to be taken into consideration and to be correct it to make a project a perfect project.

9. Results:

9.1 performance matrics



10. ADVANTAGES:

- Signs with smart connectivity are an inexpensive and flexible medium that can help transmit information according to particular situation and entertain passengers.
- The digital signboards helps in reducing the air pollution due the emission of vehicles in heavy traffic area.
- The drivers can able to know about the weather condition and accordingly follow the speed limit displayed on the sign boards.

- The increased flexibility of these digital sign boards makes it easy for any private or government department to change the message as per the need of the hour.
- The driver can easily find the route and navigation instructions to reach the destination.
- The speed of the vehicle can be identified using location sensor.
- The digitals sign boards and the app are user-friendly.

DISADVANTAGES:

- The digital signboards involves high Installation Costs.
- Getting digital signboards up and running is a far more involved process than print media.
- If the people managing the screens are not graphic designers, it can be difficult to update the content regularly on the screen.
- The digital sign boards are still new and developing technology in the road safety sector.
- While digital sign boards require power and therefore can’t claim to be green, there is high energy use in the printing, erecting and replacement of traditional print media.

11. CONCLUSION:

Digital road signs are an important part of modern infrastructure and are becoming increasingly common. Digital road signs are becoming more common as technology improves and more states adopt them. The use of digital road signs is expected to continue to grow in the future as it would be observed user-friendly, economic, environment friendly, profitable promoting road safety. Digital road signs are designed to improve road safety and efficiency by providing real-time information to drivers. These signs can display a variety of information, including speed limits, traffic conditions, and weather warnings. Digital road signs can help drivers by providing information that is not always available from traditional signs.

12. FUTURE SCOPE:

One of the benefits of digital road signs is that they can be updated in real-time, which means that they can be used to provide motorists with up-to-the-minute information about conditions on the road ahead. This can be particularly useful in the case of accidents or other incidents that might cause delays. In the future, digital road signs could also be used to provide information about alternative routes that might be available in the event of a problem on the road. This could be particularly useful in the case of major incidents, such as road closures due to bad weather. Finally, digital road signs could be used to provide motorists with information about the best times to travel in order to avoid traffic congestion. This could be particularly useful in areas where there is a lot of traffic.

13. APPENDIX:

Source Code:

```
import requests #importing a library

import json import

ibmiotf.application import

ibmiotf.device import time import

random import sys

# watson device details

organization = "2s7yy7"

devicType = "project"

deviceId = "projectid"

authMethod= "token"

authToken= "projecttoken"

#generate random values for randomo variables (temperature&humidity) def

myCommandCallback(cmd):

global a

#print("command recieved:%s" %cmd.data['command'])

#status=cmd.data['command'] print("command

recieved:%s" %cmd.data['command'])

control=cmd.data['command'] print(control)

try:
```

```
deviceOptions={"org": organization, "type": devicType,"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 "temp" with value integer value into the cloud as
a type of event for every 10 seconds deviceCli.connect() while True:
#get sensor data from DHT11 a
=
"https://api.openweathermap.org/data/2.5/weather?q=Chennai,%20IN&appid=e
2bea247ed9ad643a04d9a8e55499d5f"
r=requests.get(url=a) data=r.json()
Temp= data['main']['temp'] Humd=
data['main']['humidity'] data=
{'temp':Temp,'humid':Humd}
dist=random.randint(0,20)
dis={'dista':dist} if(Humd<100):
warn={'alert':'PLEASE SLOW DOWN!!!!!!'}
if(dist<20):
insta={'inst':'stop'}
def myOnPublishCallback():
print("published Temperature = %s c" %Temp,"humidity:%s %%" %Humd)
print(warn) print(dis) print(insta) success=deviceCli.publishEvent
```

("IoTSensor","json",insta,qos=0,on_publish= myOnPublishCallback)

success=deviceCli.publishEvent ("IoTSensor","json",data,qos=0,on_publish=

myOnPublishCallback) success=deviceCli.publishEvent

("IoTSensor","json",warn,qos=0,on_publish= myOnPublishCallback)

success=deviceCli.publishEvent ("IoTSensor","json",dis,qos=0,on_publish=

myOnPublishCallback)

if not success:

print("not connected to ibmiot") time.sleep(5)

deviceCli.commandCallback=myCommandC

allback

#disconnect the device deviceCli.disconnect()