

SIGNS WITH SMART CONNECTIVITY FOR BETTER ROAD SAFETY

SUBMITTED BY(PNT2022TMID23778):

SUJITHA LAKSHMI R.

MOHANAPRIYA V.

PRIYANGA S.

SNEHA A.

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

VIVEKANANDHA COLLEGE OF ENGINEERING FOR WOMEN

NOV 2022

Project Report

1. INTRODUCTION

1. Project Overview
2. Purpose

2. LITERATURE SURVEY

1. Existing problem
2. References
3. Problem Statement Definition

3. IDEATION & PROPOSED SOLUTION

1. Empathy Map Canvas
2. Ideation & Brainstorming
3. Proposed Solution
4. Problem Solution fit

4. REQUIREMENT ANALYSIS

1. Functional requirement
2. Non-Functional requirements

5. PROJECT DESIGN

1. Data Flow Diagrams
2. Solution & Technical Architecture
3. User Stories

6. PROJECT PLANNING & SCHEDULING

1. Sprint Planning & Estimation
2. Sprint Delivery Schedule
3. Reports from JIRA

7. CODING & SOLUTIONING (Explain the features added in the project along with code)

1. Feature 1
2. Feature 2
3. Database Schema (if Applicable)

8. TESTING

1. Test Cases
2. User Acceptance Testing

9. RESULTS

1. Performance Metrics

10. ADVANTAGES & DISADVANTAGES

11. CONCLUSION

12. FUTURE SCOPE

13. APPENDIX

Source Code

GitHub & Project Demo Link

1. INTRODUCTION

Roads are the foremost source of linking between cities and villages. Due to the ease of traveling by road, vehicles have become the main way people travel. The chances of vehicular accidents (Vas) have increased with the growing number of vehicles on the roads. During a journey, one does not know what will happen on the next road, particularly during bad weather conditions (BWC). In such a situation, driving can be difficult due to bad visibility, which can lead to an accident. It was also noticed that in BWC, multiple vehicle collisions (MVCs) can occur owing to delays in receiving information about an incident. According to one study by the Islamabad police, there were 9582 accidents from 2016 to 2017 all over Pakistan, involving 11,317 vehicles, leading to 5047 fatalities and 12,696 persons injured.

1.1. Project Overview:

The goal of this project is to replace the static signboards with smart connected sign boards to get the speed limitations from a web app using weather API and update it automatically based on the weather conditions, set diversions through API and warn drivers for school zones and hospital zones.

1.2. Purpose:

- To replace the static signboards, smart connected sign boards are used.
- These smart connected sign boards get the speed limitations from a web app using weather API and update automatically.
- Based on the weather changes the speed may increase or decrease.
- Traffic diversion signs are remotely controlled using APIs.
- "DO NOT HONK" message displayed at School and Hospital Zones which can we set using buttons.

2. LITERATURE SURVEY

2.1.Existing problem

- Rain makes brakes inefficient and leads to accidents
- Fog reduces visibility and increases the probability of accidents
- Traffic diversion requires human intervention

2.2. References

- Andrzej Czyżewski in his paper titled "Development of Intelligent Road Signs with V2X Interface for Adaptive Traffic Controlling", IEEE 2019, developed IOT based intelligent road signs capable of interacting with both the vehicles and other neighboring sign boards using LORA. These sign boards

were capable of communicating with one another and changing the speed limit based on traffic and weather.

- Muhammed O. Sayin, Chung-Wei Lin, Eunsuk Kang, Shinichi Shiraishi & Tamer Basar in their paper titled "Reliable Smart Road Signs", IEEE 2019, proposed a game theoretical adversarial intervention detection mechanism for reliable smart road signs. A future trend in intelligent transportation systems is "smart road signs" that incorporate smart codes (e.g., visible at infrared) on their surface to provide more detailed information to smart vehicles.
- L.F.P. Oliveira, L.T. Manera, P.D.G. Luz in their paper titled "Smart Traffic Light Controller System", IEEE 2019, developed smart traffic lights capable of traffic accident detection enabling the enhancement of traffic light management systems, blocking and creating alternative routes to not only avoid the traffic jams, but also avoid new accidents.
- Dariusz Grabowski & Andrzej Czyzewski in their paper titled "System for monitoring road slippery based on CCTV cameras and convolutional neural networks", Springer Publications 2020, made use of Convolutional Neural Networks to identify slippery roads using CCTV cameras.

2.3. Problem Statement Definition

To replace the static signboards with smart connected sign boards to get the speed limitations from a web app using weather API and update it automatically based on the weather conditions, set diversions through API and warn drivers for school zones and hospital zones.

3. IDEATION & PROPOSED SOLUTION

3.1. Empathy Map Canvas

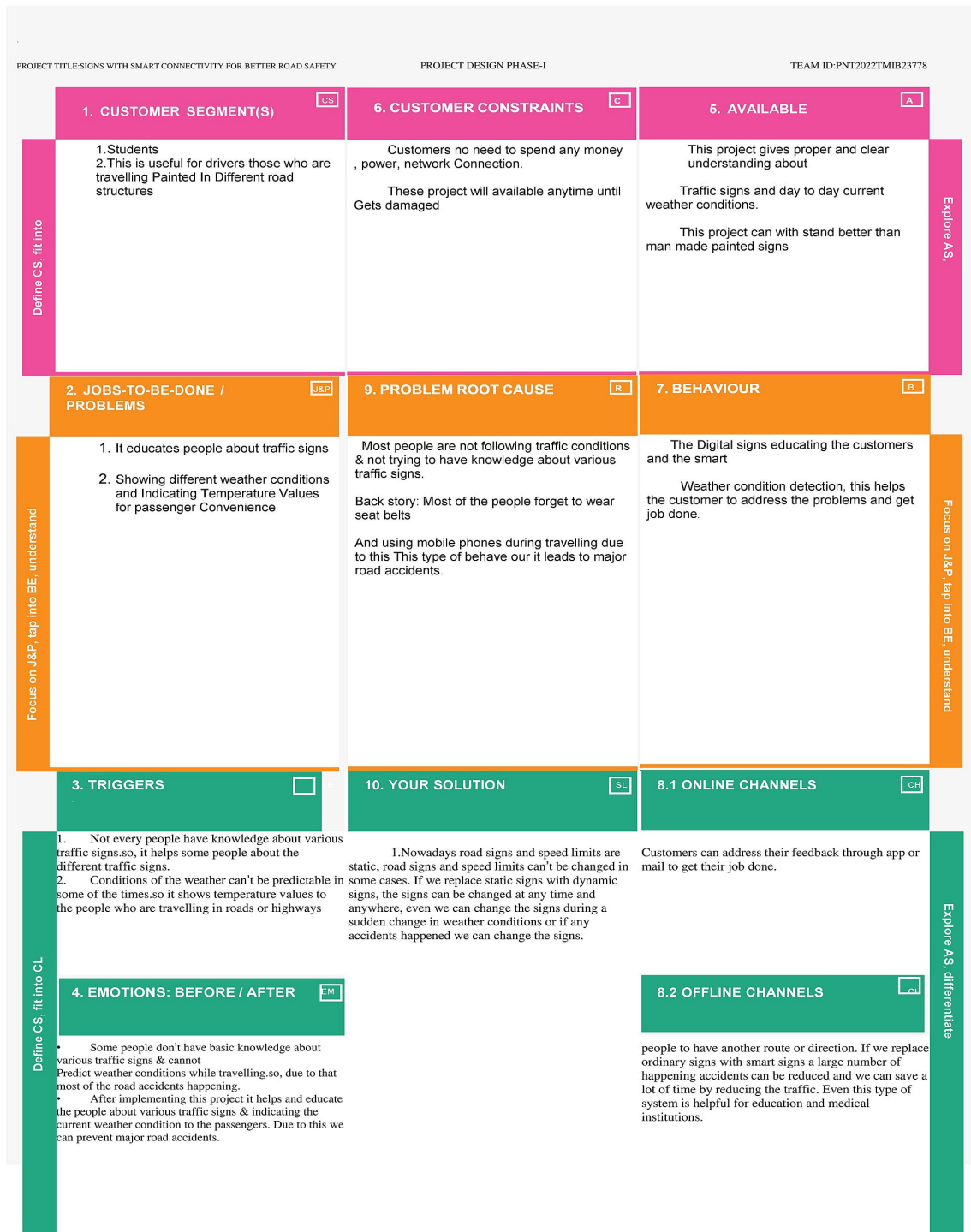


S.No.	Parameter	Description
-------	-----------	-------------

1.	Problem Statement (Problem to be solved)	<p>To replace the static signboard, smart connected sign board are used.</p> <p>These smart connected sign boards get the speed limitation from a web app using weather API and update automatically.</p> <p>Based on the weather changes the speed may increase or decrease. Based on the traffic and fatal situations the diversion signs are displayed.</p> <p>Different modes of operations can be selected with the help of buttons</p>
2.	Idea / Solution description	<p>The Weather and temperature details are obtained from the open weather Map API. Using these details, the speed limits will be updated automatically in accordance in accordance with weather conditions.</p> <p>Also, details regarding any accidents and traffic congestion faced on the particular road are obtained .</p> <p>Based on this, the traffic is diverted followed by a change in map path and the traffic is cleared. So in the traffic sign board, some buttons will be placed which will be used to make it generic; where each button will be given a functionaliaty such as changing the warning signs, Which are predefined and separate signs will be present for both school and hospital zones.</p>
		<p>By activating this button, either through the web application, and the speed limit will also be set depending upon the zones.</p> <p>Also, the pedestrians are given an option to change the traffic signs if they want to cross the road. If the pedestrian presses the button that is present on the post at the end of the road, then the traffic will be analyzed immediately. Accordingly, the signs of the traffic signals will be changed.</p> <p>This inturn reduces the freduent changing of the traffic signs even if the pedestrains are not present.</p>
3.	Novelty	<p>Generic Signs board for all applications that uses both buttons and web service for updation.</p> <p>Pedestrains are given tne access to request the signs change of the signal to cross the roads.</p>

4.	Customer Satisfaction	<p>Diversion reasons will be displayed.</p> <p>If there is no traffic, Pedestrains can cross the street without waiting. Customer can reach the destination before the expected time.</p>
5.	Business Model	<p>Since APIs are used to actively monitor the customer's environment, this project employs a business strategy in which revenue will be generated on the basis of the length of time in which the customer actively interact with the product.</p> <p>This product is aimed to be free of cost to the public, but the revenue will be generated by selling this product to the government at a low cost, So there will be less accidents and the public will be aware of the discrepancies or accidents in a particular road.</p> <p>The public will also gain all the information about the road, even if they are checking for an alternate path because of some mishaps that happens on the roads and these functionalities will increase the value of the product in the global market.</p>
6.	Scalability of the Solution	<p>In the future , if any update is required either on the hardware or software side, it can be easily implemented .</p> <p>The hardware components can be directly interfaced with the microcontroller and small MODIFI IONS CAN be directly interfaced with the product.</p> <p>So this will not affect the existing functionality of the project and new functionality can be easily integrated.</p> <p>In addition, a separate circuit will be kept along with the hardware to detect ant problem which inform the web applications. Also a notification will be send to the product service department</p>

3.4. Problem Solution fit



4. REQUIREMENT ANALYSIS

4.1. Functional requirement

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIn
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Loan type	Personal Loan Education Loan
FR-4	User Details	Name, Address, Income, Occupation
FR-6	Assets Proof	Agricultural land, Gold
	Verification	Verification of user Detail Which are Provided above

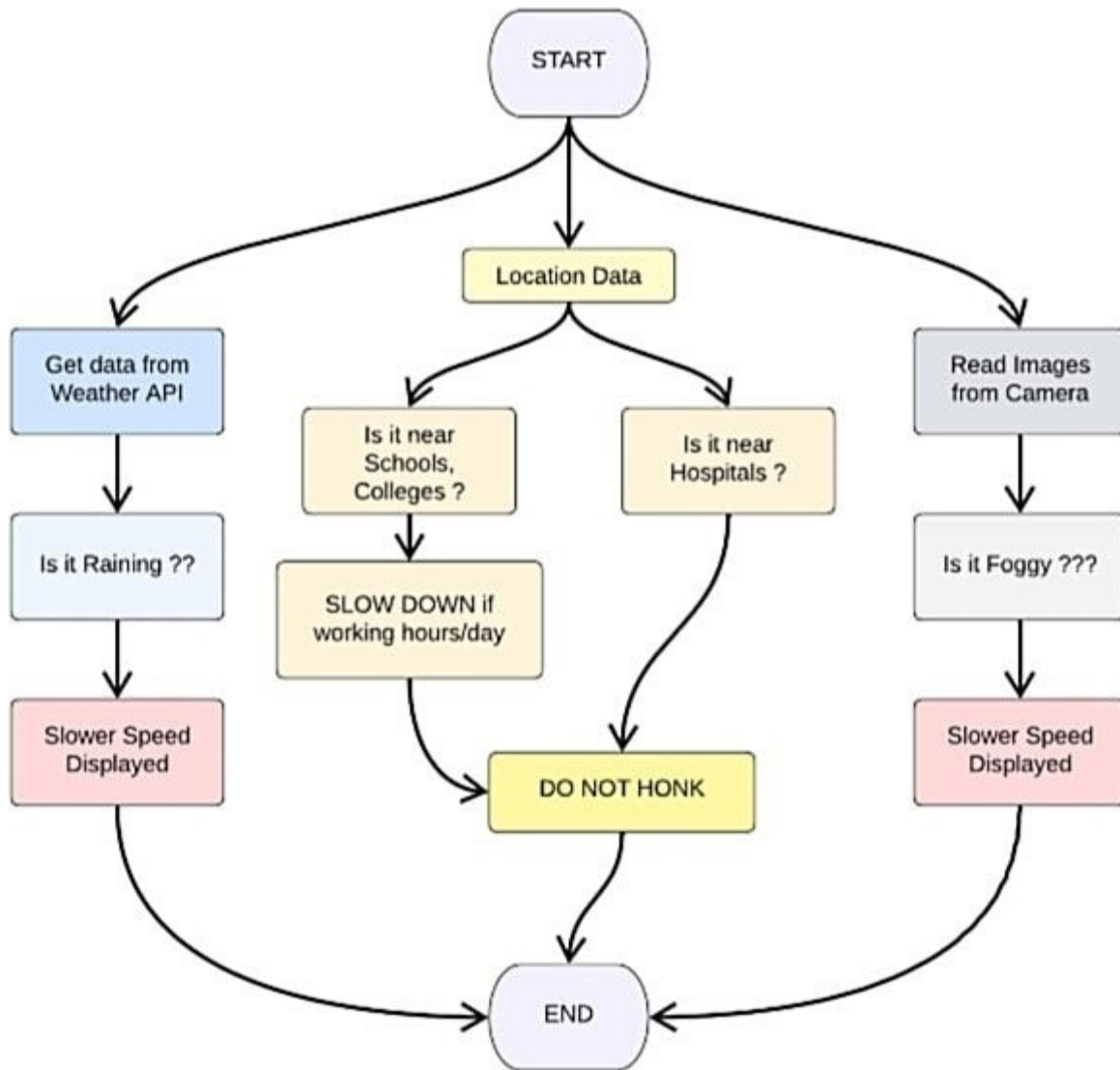
4.2. Non-Functional requirements

Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Easy to access
NFR-2	Security	User proofs
NFR-3	Reliability	Based on the customer Income
NFR-4	Performance	Previous history of the user bank account
NFR-5	Availability	Based on the customer Address
NFR-6	Scalability	Based on the customer Assets proofs

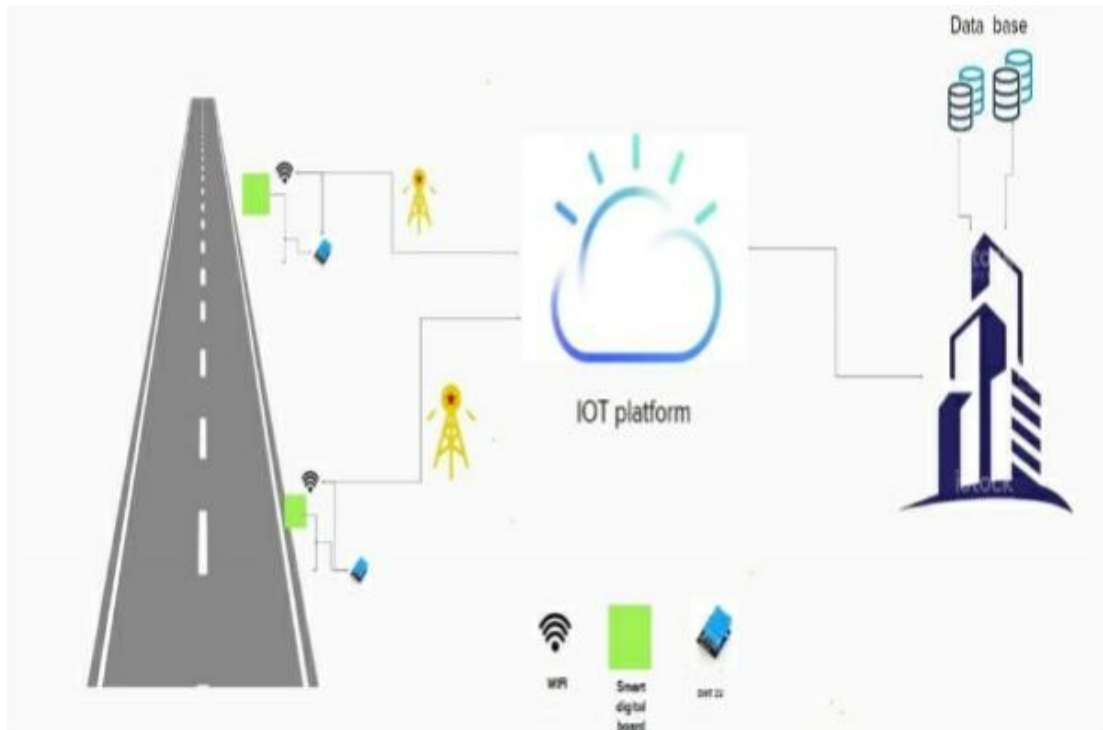
5. PROJECT DESIGN

5.1. Data Flow Diagrams



5.2. Solution & Technical Architecture

The weather and temperature details are obtained from the OpenWeatherMap API. Using these details, the speed limit will be updated automatically in accordance with the weather conditions. Also, the details regarding any accidents and traffic congestion faced on the particular road are obtained. Based on this, the traffic is diverted followed by a change in map path and the traffic is cleared.



5.3. User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password	High	Sprint-1
	Dashboard	USN-6	As a user, I can use the dashboard it will	medium	Sprint-1

			Display the summary of the total loan process.		
Customer (Web user)	Registration	USN-7	As a user, I can register for loan Website by entering my email, password, and confirming my password.	high	Sprint-1
Customer Care Executive		Doubts	As a new user how can I create my account. As a old user how can I resolve the issues.	Medium	Sprint-1
Administrator		Holding all Details	Giving approval to the particular user ID.	High	Sprint-1

6. PROJECT PLANNING & SCHEDULING

6.1. Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High	Sujitha Lakshmi R. Mohana Priya V. Priyanga S. Sneha A.
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application	1	High	Sujitha Lakshmi R. Mohana Priya V. Priyanga S. Sneha A.
Sprint-2		USN-3	As a user, I can register for the application through Facebook	2	Low	Sujitha Lakshmi R. Mohana Priya V. Priyanga S. Sneha A.
Sprint-1		USN-4	As a user, I can register for the application through Gmail	2	Medium	Sujitha Lakshmi R. Mohana Priya V. Priyanga S. Sneha A.
Sprint-1	Login	USN-5	As a user, I can log into the application by entering email & password	1	High	Sujitha Lakshmi R. Mohana Priya V. Priyanga S. Sneha A.

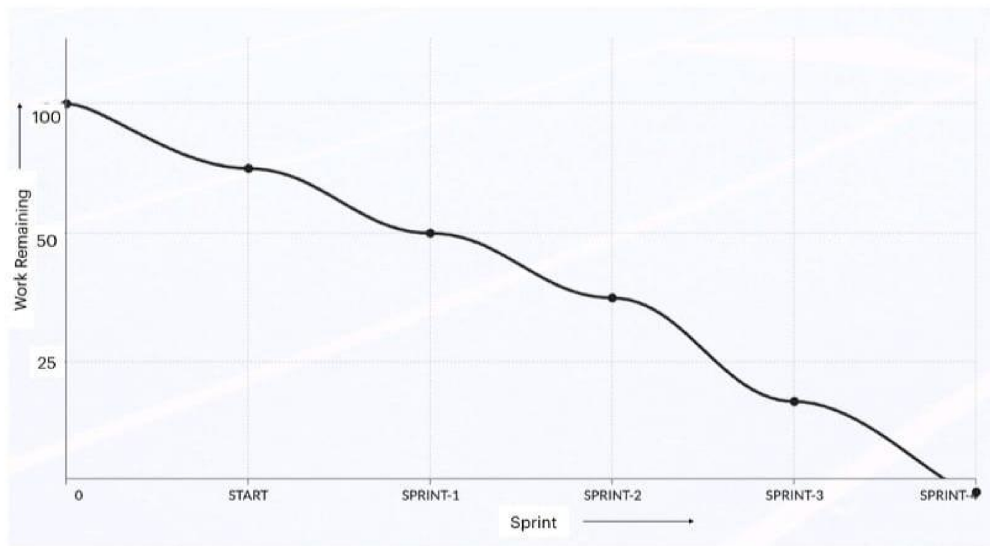
Sprint-1	Dashboard	US-1	Create the IBM Cloud services which are being used in this project.	6	High	Sujitha Lakshmi R. Mohana Priya V. Priyanga S. Sneha A.
Sprint-1		US-2	Configure the IBM Cloud services which are being used in completing this project.	4	Medium	Sujitha Lakshmi R. Mohana Priya V. Priyanga S. Sneha A.
Sprint-1		US-3	IBM Watson IoT platform acts as the mediator to connect the web application to IoT devices, so create the IBM Watson IoT platform.	5	Medium	Sujitha Lakshmi R. Mohana Priya V. Priyanga S. Sneha A.
Sprint-1		US-4	In order to connect the IoT device to the IBM cloud, create a device in the IBM Watson IoT platform and get the device credentials.	5	High	Sujitha Lakshmi R. Mohana Priya V. Priyanga S. Sneha A.
Sprint-2		US-1	Configure the connection security and create API keys that are used in the Node-RED service for accessing the IBM IoT Platform.	10	High	Sujitha Lakshmi R. Mohana Priya V. Priyanga S. Sneha A.
Sprint-2		US-2	Create a Node-RED service.	10	High	Sujitha Lakshmi R. Mohana Priya V. Priyanga S. Sneha A.
Sprint-3		US-1	Develop a python script to publish random sensor data such as temperature, humidity,rain to the IBM IoT platform	7	High	Sujitha Lakshmi R. Mohana Priya V. Priyanga S. Sneha A.
Sprint-3		US-2	After developing python code, commands are received just print the statements which represent the control of the devices.	5	Medium	Sujitha Lakshmi R. Mohana Priya V. Priyanga S. Sneha A.

Sprint-3		US-3	Publish Data to The IBM Cloud	8	High	Sujitha Lakshmi R. Mohana Priya V. Priyanga S. Sneha A.
		US-1	Create Web UI in Node-Red	10	High	Sujitha Lakshmi R. Mohana Priya V. Priyanga S. Sneha A.
Sprint-4		US-2	Configure the Node-RED flow to receive data from the IBM IoT platform and also use Cloudant DB nodes to store the received sensor data in the cloudant DB	10	High	Sujitha Lakshmi R. Mohana Priya V. Priyanga S. Sneha A.

6.2. Sprint Delivery Schedule

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

6.3. Reports from JIRA

Burndown chart:

7. CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1. Feature 1(coding and result):

```
import wiotp.sdk.device import time
import random
import ibmiotf.application
import ibmiotf.device
import requests, json
```

```
myConfig = { #Configuration "identity": {
"orgId": "pjj3fl",
"typeId": "Sign_Board", "deviceId":"Board_1"},
#API Key
"auth": {
"token": "1234567890"
}
}
```

```
#Receiving callbacks from IBM IOT
platform
defmyCommandCallback(cmd):
print("Message received from IBM IoT Platform: %s" % cmd.data['command'])
m=cmd.data['command']
```

```
client = wiotp.sdk.dev
#OpenWeatherMap Credentials
BASE_URL ="https://api.openweathermap.org/data/2.5/weather?"
CITY = "Nagercoil"
URL = BASE_URL + "q=" + CITY + "&units=metric"+"&appid=" +
"01df65417ab3968e3fc2a38c4aee27bb"
```

```
while True:
    response =
    requests.get(URL) if
    response.status_code
    ==200:
    data =
    response.json()
    main = data['main']
    temperature
    =main['temp']
    humidity =
    main['humidity']
    pressure =
    main['pressure']
    report =
    data['visibility']
```

```
#messge part
msg=random.rand
int(0,5) if
msg==1:
message="SLOW DOWN, SCHOOL IS NEAR"
elifmsg==2:
message="NEED HELP, POLICE STATION AHED"
elifmsg==3:
message="EMERGENCY, HOSPITAL NEARBY"
elifmsg==4:
message="DINE IN, RESTAURENT AVAILABLE"
else:
message="" #Speed Limit part
```

```
speed=random.randint(0,150) if speed>=100:
    speedMsg=" Limit Exceeded" elif speed>=60 and speed<100:
```

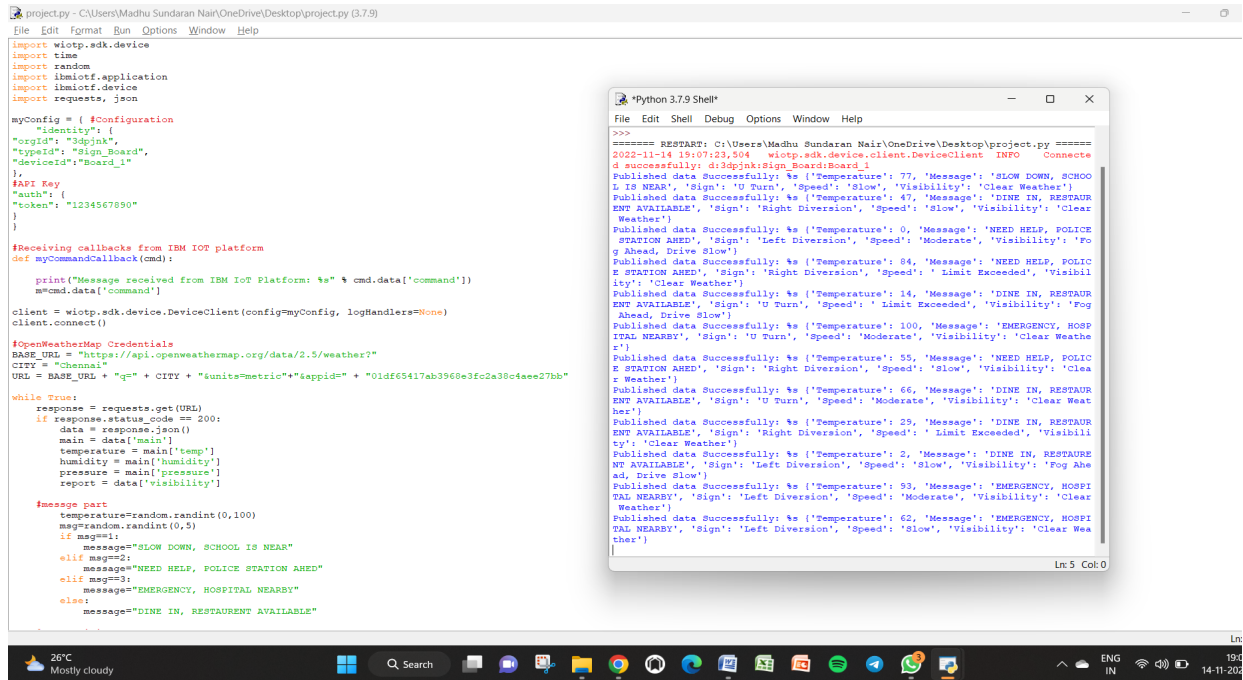
```
        speedMsg="Moderate" else:
        speedMsg="Slow"

#Diversions part sign=random.randint(0,5) if sign==1:
    signMsg="Right Diversion" elifsign==3:
    signMsg="Left Diversion" elifsign==5:
    signMsg="U Turn" else:
signMsg=""

#Visibility
if temperature < 24:
    visibility="Fog Ahead, Drive Slow" elif temperature < 20:
    visibility="Bad
Weather" else:
visibility="Clear Weather"

else:
print("Error in the HTTP request")
    myData={'Temperature':temperature, 'Message':message, 'Sign':signMsg, 'Speed':speedMsg,
'Visibility':visibility}
    client.publish(eventId="status", msgFormat="json", data=myData, qos=0, onPublish=None)
#PUBLISHING TO IOT WATSON
print("Published data Successfully: %s", myData) client.commandCallback =
myCommandCallbacktime.sleep(5)
client.disconnect()
```

Output:



The image shows a Python script in a text editor and its execution output in a terminal window. The script is a Python 3.7.9 application that interacts with an IBM IoT platform and a weather API.

```
project.py - C:\Users\Madhu Sundaran Nair\OneDrive\Desktop\project.py (3.7.9)
File Edit Format Run Options Window Help

import wiotp.sdk.device
import time
import random
import ibmiotf.application
import ibmiotf.device
import requests, json

myConfig = { #Configuration
    "identity": {
        "orgId": "3d3pjnk",
        "typeId": "Sign_Board",
        "deviceId": "Board_1"
    }
}
#API Key
"auth": {
    "token": "1234567890"
}

#Receiving callbacks from IBM IoT platform
def myCommandCallback(cmd):
    print("Message received from IBM IoT Platform: %s" % cmd.data['command'])
    m=cmd.data['command']

client = wiotp.sdk.device.DeviceClient(config=myConfig, logHandlers=None)
client.connect()

#OpenWeatherMap Credentials
BASE_URL = "https://api.openweathermap.org/data/2.5/weather?"
CITY = "Chennai"
URL = BASE_URL + "q=" + CITY + "&units=metric"&appid=" + "01d665417ab3960e3fc2a38c4ee27bb"

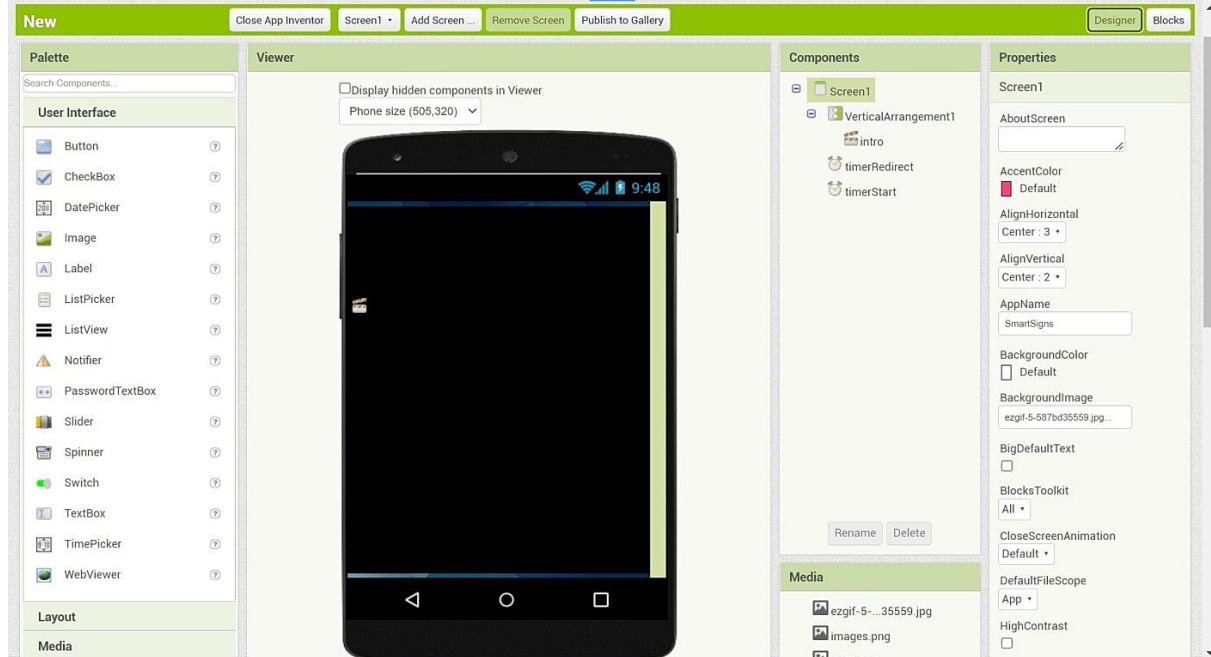
while True:
    response = requests.get(URL)
    if response.status_code == 200:
        data = response.json()
        main = data['main']
        temperature = main['temp']
        humidity = main['humidity']
        pressure = main['pressure']
        report = data['visibility']

    #message part
    temperature=random.randint(0,100)
    msg=random.randint(0,5)
    if msg==1:
        message="SLOW DOWN, SCHOOL IS NEAR"
    elif msg==2:
        message="NEED HELP, POLICE STATION AHEAD"
    elif msg==3:
        message="EMERGENCY, HOSPITAL NEARBY"
    else:
        message="DINE IN, RESTAURANT AVAILABLE"

===== RESTART: C:\Users\Madhu Sundaran Nair\OneDrive\Desktop\project.py =====
2022-11-14 19:07:23.594 wiotp.sdk.device.client.DeviceClient INFO Connecte
d successfully: d13dpjnk:Sign_Board:Board_1
Published data Successfully: %s {'Temperature': 77, 'Message': 'SLOW DOWN, SCHOO
L IS NEAR', 'Sign': 'U Turn', 'Speed': 'Slow', 'Visibility': 'Clear Weather'}
Published data Successfully: %s {'Temperature': 47, 'Message': 'DINE IN, RESTAUR
ANT AVAILABLE', 'Sign': 'Right Diversion', 'Speed': 'Slow', 'Visibility': 'Clear
Weather'}
Published data Successfully: %s {'Temperature': 0, 'Message': 'NEED HELP, POLICE
STATION AHEAD', 'Sign': 'Left Diversion', 'Speed': 'Moderate', 'Visibility': 'Fo
g Ahead, Drive Slow'}
Published data Successfully: %s {'Temperature': 84, 'Message': 'NEED HELP, POLIC
E STATION AHEAD', 'Sign': 'Right Diversion', 'Speed': 'Limit Exceeded', 'Visibil
ity': 'Clear Weather'}
Published data Successfully: %s {'Temperature': 14, 'Message': 'DINE IN, RESTAUR
ANT AVAILABLE', 'Sign': 'U Turn', 'Speed': 'Limit Exceeded', 'Visibility': 'Fog
Ahead, Drive Slow'}
Published data Successfully: %s {'Temperature': 100, 'Message': 'EMERGENCY, HOSP
ITAL NEARBY', 'Sign': 'U Turn', 'Speed': 'Moderate', 'Visibility': 'Clear Weathe
r'}
Published data Successfully: %s {'Temperature': 55, 'Message': 'NEED HELP, POLIC
E STATION AHEAD', 'Sign': 'Right Diversion', 'Speed': 'Slow', 'Visibility': 'Clea
r Weather'}
Published data Successfully: %s {'Temperature': 66, 'Message': 'DINE IN, RESTAUR
ANT AVAILABLE', 'Sign': 'U Turn', 'Speed': 'Moderate', 'Visibility': 'Clear Weat
her'}
Published data Successfully: %s {'Temperature': 29, 'Message': 'DINE IN, RESTAUR
ANT AVAILABLE', 'Sign': 'Right Diversion', 'Speed': 'Limit Exceeded', 'Visibili
ty': 'Clear Weather'}
Published data Successfully: %s {'Temperature': 2, 'Message': 'DINE IN, RESTAUR
ANT AVAILABLE', 'Sign': 'Left Diversion', 'Speed': 'Slow', 'Visibility': 'Fog Ahe
ad, Drive Slow'}
Published data Successfully: %s {'Temperature': 93, 'Message': 'EMERGENCY, HOSPI
TAL NEARBY', 'Sign': 'Left Diversion', 'Speed': 'Moderate', 'Visibility': 'Clear
Weather'}
Published data Successfully: %s {'Temperature': 62, 'Message': 'EMERGENCY, HOSPI
TAL NEARBY', 'Sign': 'Left Diversion', 'Speed': 'Slow', 'Visibility': 'Clear Wea
ther'}
Ln 5 Col 0
```

7.2. Feature 2(MIT APP INVENTER):

MIT APP INVENTOR: ICON PAGE:



8. TESTING

8.1. Test Cases

- TEST CASE 1

Clear weather - Usual Speed Limit.

- TEST CASE 2

Foggy Weather - Reduced Speed Limit.

- TEST CASE 3

Rainy Weather - Further Reduced Speed Limit.

- TEST CASE 4

School/Hospital Zone - Do not Honk sign is displayed.

8.2. User Acceptance Testing

UAT consists, in practice, of people from the target audience using the application. The defects they find are then reported and fixed. This scenario is what most closely resembles “the real world.” The process allows users to “get their hands dirty” with the application. They can see if things work as intended.

The main purpose of UAT is to validate end-to-end business flow. It does not focus on cosmetic errors, spelling mistakes, or system testing. User Acceptance Testing is carried out in a separate testing environment with a production-like data setup. It is a kind of black box testing where two or more end-users will be involved.

9. RESULTS

9.1. Performance Metrics

Based on the IBM pack we chose, the performance of the website varies. Built upon NodeJS, a light and high performance engine, NodeRED is capable of handling upto 10,000 requests per second. Moreover, since the system is horizontally scalable, a even higher demand of customers can be served.

10. ADVANTAGES & DISADVANTAGES

- **ADVANTAGES**

- Lower battery consumption since processing is done mostly by Node RED servers in the cloud.
- Cheaper and low requirement micro controllers can be used since processing requirements are reduced.
- Longer lasting systems.
- Dynamic Sign updation.
- School/Hospital Zone alerts

- **DISADVANTAGES**

- The size of the display determines the requirement of the micro controller
- Dependent on OpenWeatherAPI and hence the speed reduction is same for a large area in the scale of cities.

11. CONCLUSION

Our project is capable of serving as a replacement for static signs for a comparatively lower cost and can be implemented in the very near future. This will help reduce a lot of accidents and maintain a more peaceful traffic atmosphere in the country.

12. FUTURE SCOPE

Introduction of intelligent road sign groups in real life scenarios could have great impact on increasing the driving safety by providing the end-user (car driver) with the most accurate information regarding the current road and traffic conditions. Even displaying the information of a suggested driving speed and road surface condition (temperature, icy, wet or dry surface) could result in smoother traffic flows and, what is more important, in increasing a driver's awareness of the road situation.

13. APPENDIX

Source Code

```
import wiotp.sdk.device import time
import random
import ibmiotf.application
import ibmiotf.device
import requests, json

myConfig = { #Configuration "identity": {
"orgId": "pjj3fl",
"typeId": "Sign_Board", "deviceId":"Board_1"},
#API Key
"auth": {
"token": "1234567890"
}
}

#Receiving callbacks from IBM IOT
platform
defmyCommandCallback(cmd):
print("Message received from IBM IoT Platform: %s" % cmd.data['command'])
m=cmd.data['command']

client = wiotp.sdk.dev
#OpenWeatherMap Credentials
BASE_URL ="https://api.openweathermap.org/data/2.5/weather?"
CITY = "Nagercoil"
URL = BASE_URL + "q=" + CITY + "&units=metric"+"&appid=" +
"01df65417ab3968e3fc2a38c4aee27bb"

while True:
response =
requests.get(URL) if
response.status_code
==200:
data =
response.json()
main = data['main']
temperature
=main['temp']
humidity =
main['humidity']
```

```
pressure =
main['pressure']
report =
data['visibility']

#messge part
msg=random.rand
int(0,5) if
msg==1:
message="SLOW DOWN, SCHOOL IS NEAR"
elifmsg==2:
message="NEED HELP, POLICE STATION AHED"
elifmsg==3:
message="EMERGENCY, HOSPITAL NEARBY"
elifmsg==4:
message="DINE IN, RESTAURENT AVAILABLE"
else:
message="" #Speed Limit part

speed=random.randint(0,150) if speed>=100:
    speedMsg=" Limit Exceeded" elif speed>=60 and speed<100:
    speedMsg="M
oderate" else:
speedMsg="Slow"

#Diversion part sign=random.randint(0,5) if sign==1:
    signMsg="Right Diversion" elifsign==3:
    signMsg="Left Diversion" elifsign==5:
    signmsg="U Turn" else:
signMsg=""

#Visibility
if temperature < 24:
    visibility="Fog Ahead, Drive Slow" elif temperature < 20:
    visibility="Bad
Weather" else:
visibility="Clear Weather"

else:
print("Error in the HTTP request")
myData={'Temperature':temperature, 'Message':message, 'Sign':signMsg, 'Speed':speedMsg,
```



```
'Visibility':visibility}  
    client.publishEvent(eventId="status", msgFormat="json", data=myData, qos=0, onPublish=None)  
#PUBLISHING TO IOT WATSON  
print("Published data Successfully: %s", myData)    client.commandCallback    =  
myCommandCallbacktime.sleep(5)  
client.disconnect()
```

GitHub Link:

<https://github.com/IBM-EPBL/IBM-Project-34504-1664442153>

Project Demo Link:

<https://github.com/IBM-EPBL/IBM-Project-34504-1664442153/blob/main/Final%20Deliverables/Project%20Demo%20Link.mp4>