**IBM NALAIYA THIRAN 2022-23 PROJECT REPORT**

**SIGNS WITH SMART CONNECTIVITY FOR BETTER ROAD SAFETY TEAM ID -** PNT2022TMID23663

1. **INTRODUCTION**
   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. **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.

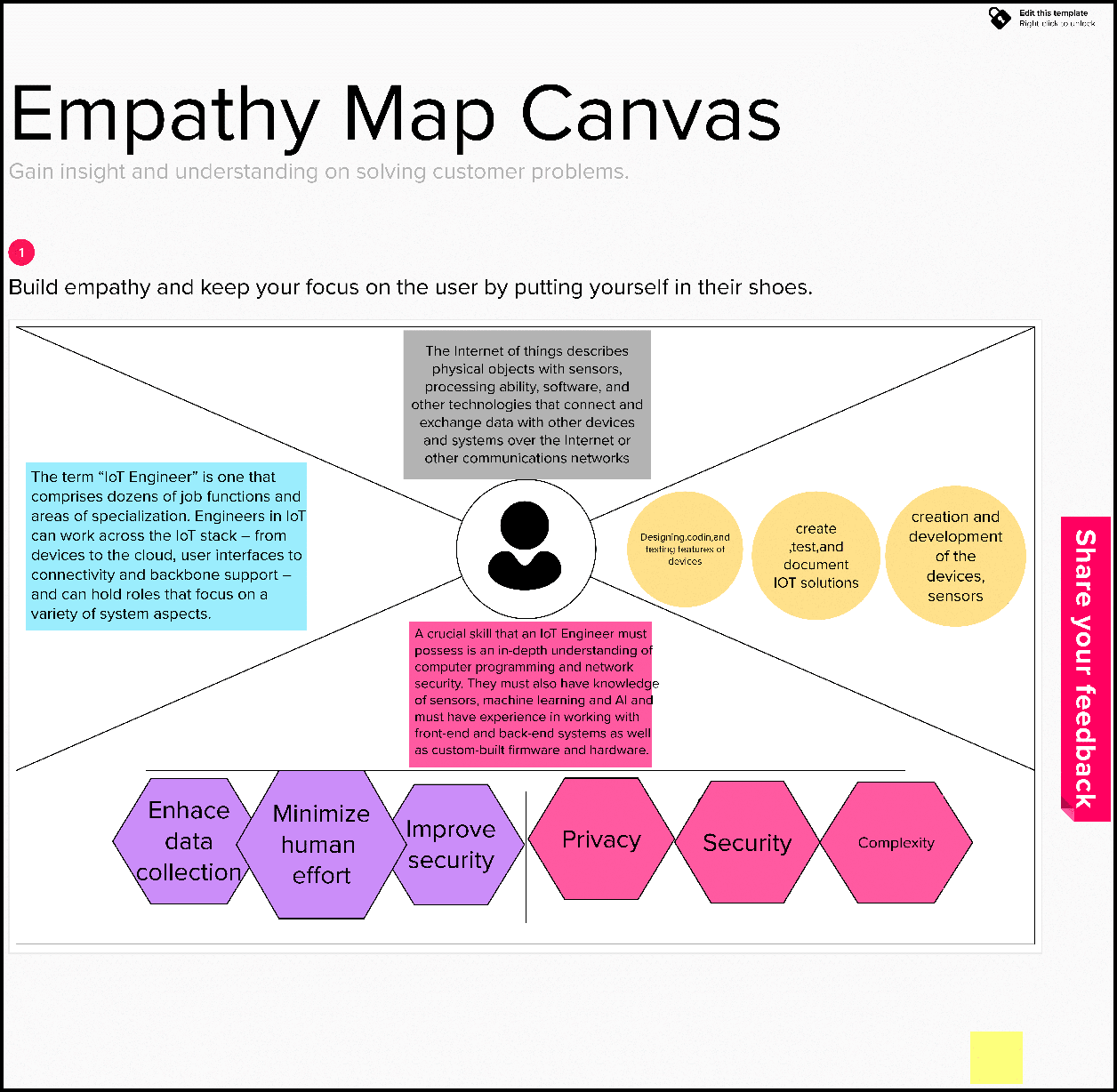
1. **LITERATURE SURVEY**
   1. **EXISTING PROBLEM**

* Rain makes brakes inefcient and leads to accidents
* Fog reduces visibility and increases the probability of accidents
* Traﬃc diversion requires human intervention
  1. **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 neighbouring 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.
  1. **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.

1. **IDEATION AND PROPOSED SOLUTION**
   1. **EMPATHY MAP CANVAS**

[https://github.com/IBM-EPBL/IBM-Project-20178-](https://github.com/IBM-EPBL/IBM-Project-20178-1659714220/blob/main/IDEATION%20PHASE/Empathy%20map%20canvas.pdf)

[1659714220/blob/main/IDEATION%20PHASE/Empathy%20map%20canvas.pdf](https://github.com/IBM-EPBL/IBM-Project-20178-1659714220/blob/main/IDEATION%20PHASE/Empathy%20map%20canvas.pdf)

* 1. **IDEATION & BRAINSTORMING**

[https://github.com/IBM-EPBL/IBM-Project-20178-](https://github.com/IBM-EPBL/IBM-Project-20178-1659714220/blob/main/IDEATION%20PHASE/Brainstorm%20and%20idea.pdf)

[1659714220/blob/main/IDEATION%20PHASE/Brainstorm%20and%20idea.pdf](https://github.com/IBM-EPBL/IBM-Project-20178-1659714220/blob/main/IDEATION%20PHASE/Brainstorm%20and%20idea.pdf)

* 1. **PROPOSED SOLUTION**
* Use a ESP32 to drive a display as a replacement for static sign boards.
* Conﬁgure IBM cloud server such that upon making a single http request with location, unique id, usual speed limit & hospital/school zone info, it returns processes the data at cloud and returns only the message to be displayed at the sign board display.
* Another http end point is conﬁgured to set the direction to be displayed. Upon accessing this http end point, the direction is set remotely for a display using it's unique id.
  1. **PROBLEM SOLUTION FIT**
     + The display replaces the static signs
     + Processing requirement of microcontroller is reduced since all the processing is done in the cloud servers.
     + Direction can be remotely set by the concerned authorities without needing to personally attend the site.

1. **REQUIREMENT ANALYSIS**
   1. **FUNCTIONAL REQUIREMENTS**

[https://github.com/IBM-EPBL/IBM-Project-19485-](https://github.com/IBM-EPBL/IBM-Project-19485-1659698659/blob/master/Project%20Design%20%26%20Planning/Project%20Design%20Phase%202/Solution%20Requirements.pdf)

[1659698659/blob/master/Project%20Design%20%26%20Planning/Project%20Design%](https://github.com/IBM-EPBL/IBM-Project-19485-1659698659/blob/master/Project%20Design%20%26%20Planning/Project%20Design%20Phase%202/Solution%20Requirements.pdf) [20Phase%202/Solution%20Requirements.pdf](https://github.com/IBM-EPBL/IBM-Project-19485-1659698659/blob/master/Project%20Design%20%26%20Planning/Project%20Design%20Phase%202/Solution%20Requirements.pdf)

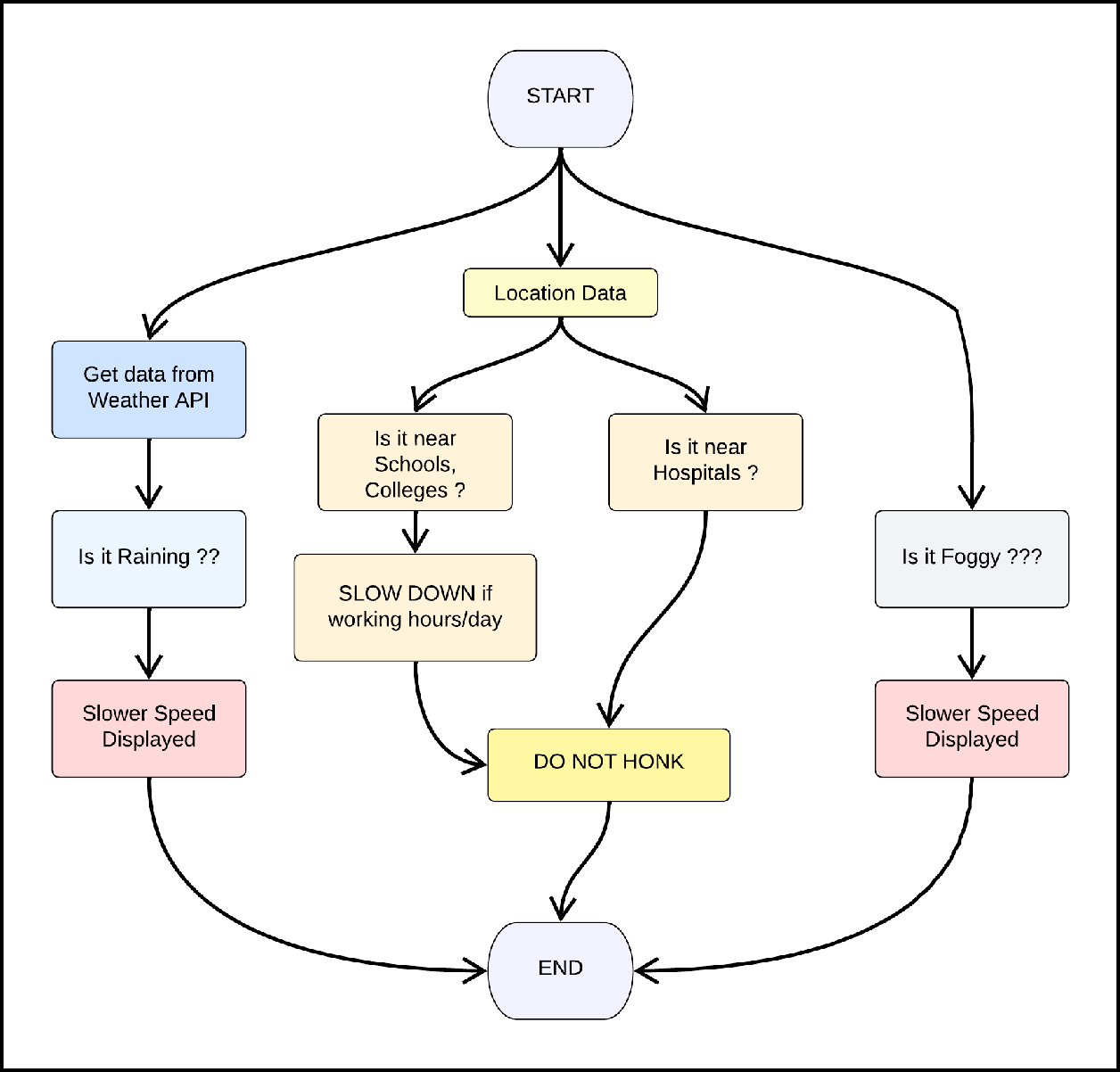
|  |  |  |
| --- | --- | --- |
| **FR No. Functional Requirement Sub Requirement** | | |
| FR-1 | User Visibility | Displays must be made with bright colored LEDs  just enough for User attention |
| FR-2 | User Understanding | Display images/text for easier user understanding. |
| FR-3 | User Convenience | Text must be big enough for users to grasp  message with ease |

* 1. **NON-FUNCTIONAL REQUIREMENTS**

[https://github.com/IBM-EPBL/IBM-Project-19485-](https://github.com/IBM-EPBL/IBM-Project-19485-1659698659/blob/master/Project%20Design%20%26%20Planning/Project%20Design%20Phase%202/Solution%20Requirements.pdf)

[1659698659/blob/master/Project%20Design%20%26%20Planning/Project%20Design%](https://github.com/IBM-EPBL/IBM-Project-19485-1659698659/blob/master/Project%20Design%20%26%20Planning/Project%20Design%20Phase%202/Solution%20Requirements.pdf) [20Phase%202/Solution%20Requirements.pdf](https://github.com/IBM-EPBL/IBM-Project-19485-1659698659/blob/master/Project%20Design%20%26%20Planning/Project%20Design%20Phase%202/Solution%20Requirements.pdf)

|  |  |  |  |
| --- | --- | --- | --- |
| **NFR No.** | **Non Functional**  **Requirement** | **Desciption** |  |
| NFR-1 | Usability | Should be able to dynamically update with time and  weather. | |
| NFR-2 | Security | Should be secure enough that only intended messages  are displayed. | |
| NFR-3 | Reliability | Should convey the traﬃc sign correctly. | |
| NFR-4 | Performance | Display should be updated as soon as traﬃc signs are  updated. | |
| NFR-5 | Availability | Should be working 24/7 | |
| NFR-6 | Scalability | Should be modular and hence  horizontally on servers. | be able to scale |

1. **PROJECT DESIGN**
   1. **DATA FLOW DIAGRAMS**
   2. **SOLUTION & TECHNICAL ARCHITECTURE**

[https://github.com/IBM-EPBL/IBM-Project-19485-](https://github.com/IBM-EPBL/IBM-Project-19485-1659698659/blob/master/Project%20Design%20%26%20Planning/Project%20Design%20Phase%201/Solution%20Architecture%20Technical.pdf)

[1659698659/blob/master/Project%20Design%20%26%20Planning/Project%20Design%](https://github.com/IBM-EPBL/IBM-Project-19485-1659698659/blob/master/Project%20Design%20%26%20Planning/Project%20Design%20Phase%201/Solution%20Architecture%20Technical.pdf) [20Phase%201/Solution%20Architecture%20Technical.pdf](https://github.com/IBM-EPBL/IBM-Project-19485-1659698659/blob/master/Project%20Design%20%26%20Planning/Project%20Design%20Phase%201/Solution%20Architecture%20Technical.pdf)

* 1. **USER STORIES**

[https://github.com/IBM-EPBL/IBM-Project-19485-](https://github.com/IBM-EPBL/IBM-Project-19485-1659698659/blob/master/Project%20Design%20%26%20Planning/Project%20Design%20Phase%202/CustomerJourneyMap.pdf)

[1659698659/blob/master/Project%20Design%20%26%20Planning/Project%20Design%](https://github.com/IBM-EPBL/IBM-Project-19485-1659698659/blob/master/Project%20Design%20%26%20Planning/Project%20Design%20Phase%202/CustomerJourneyMap.pdf) [20Phase%202/CustomerJourneyMap.pdf](https://github.com/IBM-EPBL/IBM-Project-19485-1659698659/blob/master/Project%20Design%20%26%20Planning/Project%20Design%20Phase%202/CustomerJourneyMap.pdf)

1. **PROJECT PLANNING AND SCHEDULING PHASE**
   1. **SPRINT PLANNING & ESTIMATION**

[https://github.com/IBM-EPBL/IBM-Project-19485-](https://github.com/IBM-EPBL/IBM-Project-19485-1659698659/blob/master/Project%20Design%20%26%20Planning/Project%20Planning/ProjectPlanning.pdf)

[1659698659/blob/master/Project%20Design%20%26%20Planning/Project%20Planning](https://github.com/IBM-EPBL/IBM-Project-19485-1659698659/blob/master/Project%20Design%20%26%20Planning/Project%20Planning/ProjectPlanning.pdf)

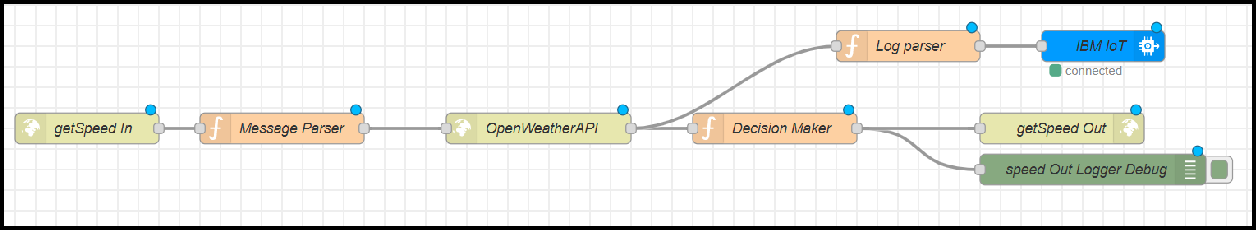
[/ProjectPlanning.pdf](https://github.com/IBM-EPBL/IBM-Project-19485-1659698659/blob/master/Project%20Design%20%26%20Planning/Project%20Planning/ProjectPlanning.pdf)

* 1. **SPRINT DELIVERY SCHEDULE**

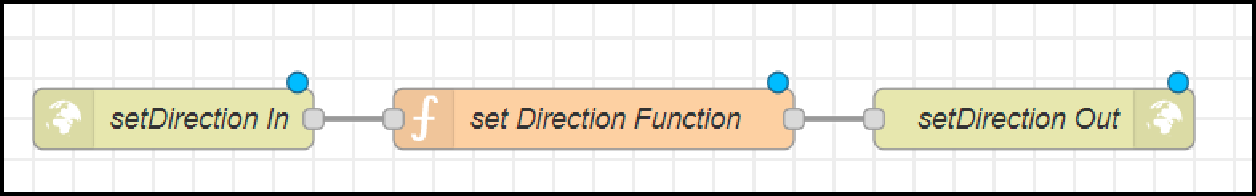
[https://github.com/IBM-EPBL/IBM-Project-19485-](https://github.com/IBM-EPBL/IBM-Project-19485-1659698659/blob/master/Project%20Design%20%26%20Planning/Project%20Planning/ProjectPlanning.pdf)

[1659698659/blob/master/Project%20Design%20%26%20Planning/Project%20Planning](https://github.com/IBM-EPBL/IBM-Project-19485-1659698659/blob/master/Project%20Design%20%26%20Planning/Project%20Planning/ProjectPlanning.pdf)

[/ProjectPlanning.pdf](https://github.com/IBM-EPBL/IBM-Project-19485-1659698659/blob/master/Project%20Design%20%26%20Planning/Project%20Planning/ProjectPlanning.pdf)

1. **CODING & SOLUTIONING**
   1. **FEATURE 1 - GET SPEED FOR GIVEN LOCATION & CLIMATE**

This part of Node RED ﬂow accepts an http GET end point at **"/getSpeed"** from which the location, uid, hospital/school zone info are passed. Message parser sets the required APIKEY for OpenWeatherAPI for the next block. This data is then passed onto Decision Maker which makes all the decisions regarding the message to be output at the display and sends it as a http response. This data is displayed at the micro- controller. Thus a lot of battery is saved due to lesser processing time.

* 1. **FEATURE 2 - SET DIRECTION REMOTELY FOR A GIVEN SIGN BOARD**

This part of Node RED ﬂow accepts an http GET end point at **"/setDirection"** from which the uid and direction information are passed by the respective authorities. Set Direction Function block adds the direction information to the database and returns the same as an http response. This data is sent to the microcontroller along with the **"/getSpeed"** path and the microcontroller displays it.

A detailed documentation of all the workﬂows is available at the following link : [https://github.com/IBM-EPBL/IBM-Project-19485-](https://github.com/IBM-EPBL/IBM-Project-19485-1659698659/blob/master/Project%20Development%20Phase/Sprint%204/Readme.md)

[1659698659/blob/master/Project%20Development%20Phase/Sprint%204/Readme.md](https://github.com/IBM-EPBL/IBM-Project-19485-1659698659/blob/master/Project%20Development%20Phase/Sprint%204/Readme.md)

1. **TESTING**
   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/Hosipital Zone - Do not Honk sign is displayed.

* 1. **USER ACCEPTANCE TESTING**

Dynamic speed & divertion variations based on the weather and traﬃc helps user to avoid traﬃc and have a safe journey home. The users would welcome this idea to be implemented everywhere.

1. **RESULTS**
   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.

1. **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.

1. **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.

1. **FUTURE SCOPE**

**1**

**2**

**3**

**4**

**5**

**6**

**7**

**8**

**9**

**#include <WiFi.h> #include <HTTPClient.h> #include <Adafruit\_GFX.h>**

**#include <Adafruit\_ILI9341.h>**

**#include <string.h>**

**const char\* ssid = "Wokwi-GUEST";**

**const char\* password = "";**

1. **#define TFT\_DC 2**
2. **#define TFT\_CS 15**
3. **Adafruit\_ILI9341 tft = Adafruit\_ILI9341(TFT\_CS, TFT\_DC); 13**

**14 String myLocation = "Chennai,IN";**

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.

1. **APPENDIX**

* **GITHUB AND PROJECT DEMO LINK**

[**https://github.com/IBM-EPBL/IBM-Project-19485-1659698659**](https://github.com/IBM-EPBL/IBM-Project-19485-1659698659)

* **DEMO VIDEO DOWNLOAD LINK**

[**https://github.com/IBM-EPBL/IBM-Project-19485-**](https://github.com/IBM-EPBL/IBM-Project-19485-1659698659/raw/master/Final%20Deliverables/demoVideo.mp4)[**1659698659/raw/master/Final%20Deliverables/demoVideo.mp4**](https://github.com/IBM-EPBL/IBM-Project-19485-1659698659/raw/master/Final%20Deliverables/demoVideo.mp4)

* **SOURCE CODE - ESP 32**

**15 String usualSpeedLimit = "70"; // kmph 16**

1. **int schoolZone = 32;**
2. **int hospitalZone = 26;**

**19**

**20 int uid = 2504; // ID Unique to this Micro Contoller 21**

**22 String getString(char x) 23 {**

1. **String s(1, x);**
2. **return s; 26 }**

**27**

**28 String stringSplitter1(String fullString,char delimiter='$') 29 {**

1. **String returnString = "";**
2. **for(int i = 0; i<fullString.length();i++) {**
3. **char c = fullString[i];**
4. **if(delimiter==c)**
5. **break;**
6. **returnString+=String(c); 36 }**

**37 return(returnString); 38 }**

**39**

**40 String stringSplitter2(String fullString,char delimiter='$') 41 {**

1. **String returnString = "";**
2. **bool flag = false;**
3. **for(int i = 0; i<fullString.length();i++) {**
4. **char c = fullString[i];**
5. **if(flag)**
6. **returnString+=String(c);**
7. **if(delimiter==c)**
8. **flag = true; 50 }**

**51 return(returnString); 52 }**

**53**

**54 void rightArrow()**

**55 {**

1. **int refX = 50;**
2. **int refY = tft.getCursorY() + 40; 58**
3. **tft.fillRect(refX,refY,100,20,ILI9341\_RED);**
4. **tft.fillTriangle(refX+100,refY- 30,refX+100,refY+50,refX+40+100,refY+10,ILI9341\_RED);**

**61 }**

**62**

**63 void leftArrow()**

**64 {**

1. **int refX = 50;**
2. **int refY = tft.getCursorY() + 40; 67**
3. **tft.fillRect(refX+40,refY,100,20,ILI9341\_RED);**
4. **tft.fillTriangle(refX+40,refY- 30,refX+40,refY+50,refX,refY+10,ILI9341\_RED);**

**70 }**

**71**

**72 void upArrow()**

**73 {**

1. **int refX = 125;**
2. **int refY = tft.getCursorY() + 30; 76**
3. **tft.fillTriangle(refX-**

**40,refY+40,refX+40,refY+40,refX,refY,ILI9341\_RED);**

1. **tft.fillRect(refX-15,refY+40,30,20,ILI9341\_RED); 79 }**

**80**

1. **String APICall() {**
2. **HTTPClient http; 83**
3. **String url = "https://node-red-grseb-2022-11-05-test.eu- gb.mybluemix.net/getSpeed?";**
4. **url += "location="+myLocation+"&";**
5. **url +=**

**"schoolZone="+(String)digitalRead(schoolZone)+(String)"&";**

1. **url +=**

**"hospitalZone="+(String)digitalRead(hospitalZone)+(String)"& ";**

1. **url +=**

**"usualSpeedLimit="+(String)usualSpeedLimit+(String)"&";**

1. **url += "uid="+(String)uid;**
2. **http.begin(url.c\_str());**
3. **int httpResponseCode = http.GET(); 92**
4. **if (httpResponseCode>0) {**
5. **String payload = http.getString();**
6. **http.end();**
7. **return(payload); 97 }**
8. **else {**
9. **Serial.print("Error code: ");**
10. **Serial.println(httpResponseCode);**

**101 }**

**102 http.end();**

**103 } 104**

1. **void myPrint(String contents) {**
2. **tft.fillScreen(ILI9341\_BLACK);**
3. **tft.setCursor(0, 20);**
4. **tft.setTextSize(4);**
5. **tft.setTextColor(ILI9341\_RED);**
6. **//tft.println(contents); 111**
7. **tft.println(stringSplitter1(contents));**
8. **String c2 = stringSplitter2(contents);**
9. **if(c2=="s") // represents Straight**

**115 {**

**116 upArrow();**

**117 }**

**118 if(c2=="l") // represents left**

**119 {**

**120 leftArrow();**

**121 }**

**122 if(c2=="r") // represents right**

**123 {**

**124 rightArrow();**

**125 }**

**126 } 127**

1. **void setup() {**
2. **WiFi.begin(ssid, password, 6); 130**
3. **tft.begin();**
4. **tft.setRotation(1); 133**
5. **tft.setTextColor(ILI9341\_WHITE);**
6. **tft.setTextSize(2);**
7. **tft.print("Connecting to WiFi"); 137**

**138 while (WiFi.status() != WL\_CONNECTED) {**

**139 delay(100);**

**140 tft.print(".");**

**141 } 142**

1. **tft.print("\nOK! IP=");**
2. **tft.println(WiFi.localIP());**

**145 } 146**

1. **void loop() {**
2. **myPrint(APICall());**

**149 delay(100);**

**150 }**