

Project Report

Project Name: SIGNS WITH SMART CONNECTIVITY FOR BETTER ROAD SAFE

Team ID: PNT2022TMID51479

TEAM LEADER : MUTHULEKSHMI P

TEAM MEMBER : VINI SHALINI M

SARAL N

ANNS A

I INTRODUCTION

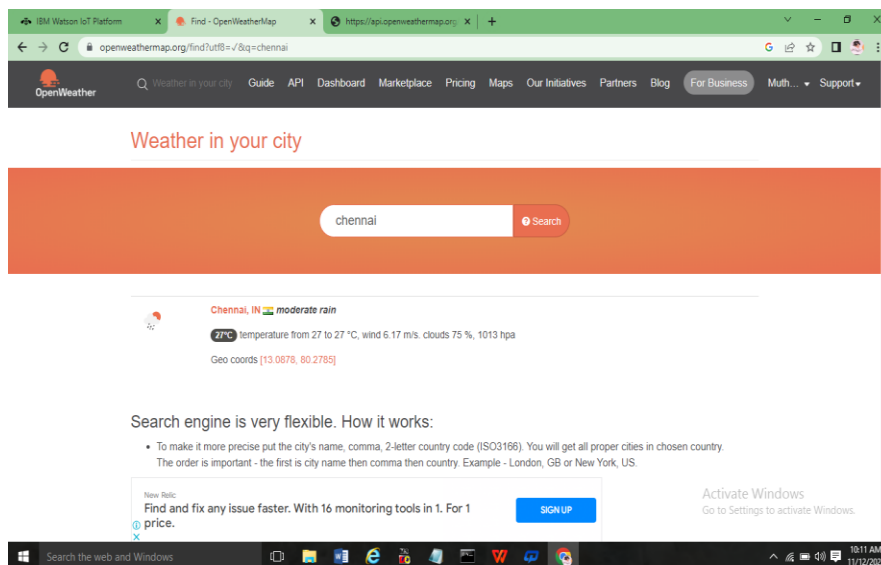
1.1 Project Overview

In present Systems 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. 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

The purpose of this project is to report and get relieved from the issues related to road safety.

1.OPEN WEATHER MAP



2. LITERATURE SURVEY

2.1 Existing problem

- Climate changes
- the non-use of motorcycle helmets, seat-belts, and child restraint,
- not so clear vision of sign boards
- distraction, including the use of mobile phones, leading to impaired driving,
- unsafe vehicles and unsafe road infrastructure can negatively impact safety on the roads,
- inadequate post-crash care,

2.2 References

- 1.Karnadi, F. K., Zhi, H. M., &Kun-chan, L., "Rapid Generation of Realistic Mobility Models for VANET", Wireless Communications and Networking Conference, WCNC2007.
- 2.General safety feathers by Transport Canada, 2007.
- 3.Shulman, M., &Deering, R., Vehicle Safety Communications in theUnited States, 2007.
- 4.Yang, X., Liu, J., Zhao, F., &Vaidya, N (2004), Vehicle-to-Vehicle Communication Protocol for Cooperative Collision Warning.
- 5.Kurt Dresner &Peter Stone (2008), Replacing the Stop Sign: Unmanaged Intersection Control, pp.94-101, Estoril, Portugal.
- 6.ARB STD-T75, 2001 Zing Zhu, Sumit Roy. Dedicated Short Range Communication System.

2.3 Problem Statement Definition

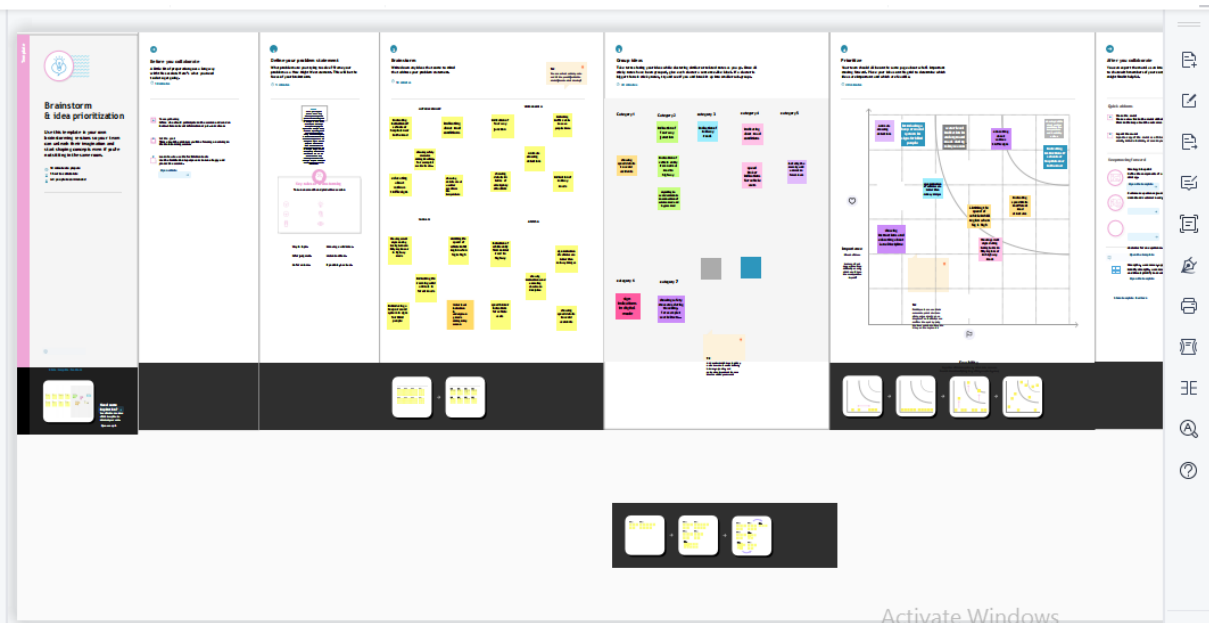
Smart Solutions for road safety are designed to reduce the risk for the user.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



3.2 Ideation & Brainstorming



3.3 Proposed Solution

Project Design Phase-I
Proposed Solution

Date	19 September 2022
Team ID	PNT2022TMD51479
Project Name	SIGNS WITH SMART CONNECTIVITY FOR BETTER ROAD SAFTY
Maximum Marks	2 Marks

Proposed Solution Template:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	<ul style="list-style-type: none"> The actual problem is that drivers are unable to know whether the road condition is safe to travel or not. Hence there will be a need of guidance to provide the safety and to avoid the inconvenience to reach the destination.
2.	Idea / Solution description	<ul style="list-style-type: none"> This problem can be overcome by using rain drop sensor to indicate there is rain is occurs or not. Using GPRS and IR sensor with camera to sense the traffic in dark areas. Collecting information from the local

		peoples and the dedslons are made by the controller who control the display manually.
3.	Novelty / Uniqueness	<ul style="list-style-type: none"> Digital sign boards are convey the information to the driver using Embedded and IOT technology. Speed limit changes according to the weather condition using rain drop sensor.
4.	Social Impact / Customer Satisfaction	<ul style="list-style-type: none"> The proposed system provides many facilities which helps the drivers to maintain the safety. Signs change based on upcoming events.
5.	Business Model (Revenue Model)	<ul style="list-style-type: none"> This prototype can be developed as product with minimum cost with high performance. Reduces manpower.
6.	Scalability of the Solution	<ul style="list-style-type: none"> User friendly. Easy to access the data from the source. Information in sign boards can be easily captured.

3.4 Problem Solution fit

Problem-Solution Fit canvas			Purpose / Vision	Version
1. CUSTOMER SEGMENT(S) CS Define CS, MS into CS. ➤ Automobiles	6. CUSTOMER LIMITATIONS CL EG. BUDGET, DEVICES Lack of knowledge, budget, no cash, network connection, lack of available devices.	5. AVAILABLE SOLUTIONS AS PROS & CONS ➤ Limited speed ➤ Follow rule and regulations ➤ Vehicles maintenance	Focus on CS, MS into CS, MS into CS, MS into CS.	
2. PROBLEMS / PAINS + ITS FREQUENCY PR ➤ Geo fencing (for younger generation). ➤ Vehicle to vehicle communication. ➤ Remote parking.	9. PROBLEM ROOT / CAUSE RC ➤ High speed ➤ Lack of time management	7. BEHAVIOR + ITS INTENSITY BE ➤ Efficient ➤ Low cost ➤ High performance		
3. TRIGGERS TO ACT TR ➤ Excessive accidents. ➤ Reading about a more efficient solution in the news. 4. EMOTIONS BEFORE / AFTER EM ➤ Time delay because of traffic and suffered a lot. ➤ Time management and update and secure life.	10. YOUR SOLUTION SL ➤ Our solution is based on IOT we where plan to use GPS tracking with digital display and voice display. ➤ It use us to solve our required problem.	8. CHANNELS of BEHAVIOR CH ONLINE ➤ The program should be error free OFFLINE ➤ The apparatus we require should be affordable		

4. REQUIREMENT ANALYSIS

4.1 Functional requirement & non functional requirement

Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Requirements	Static signboards will be replaced with smart linked sign boards that meet all criteria.
FR-2	User Registration	User Registration can be done through a Website or Gmail
FR-3	User Confirmation	Phone Confirmation Email confirmation OTP authentication
FR-4	Payments options	Bank Transfers
FR-5	Product Delivery and installation	The installation fee will be depend upon the length of the road.
FR-6	Product Feedback	Will be shared through a website via Gmail

Non-functional Requirements:

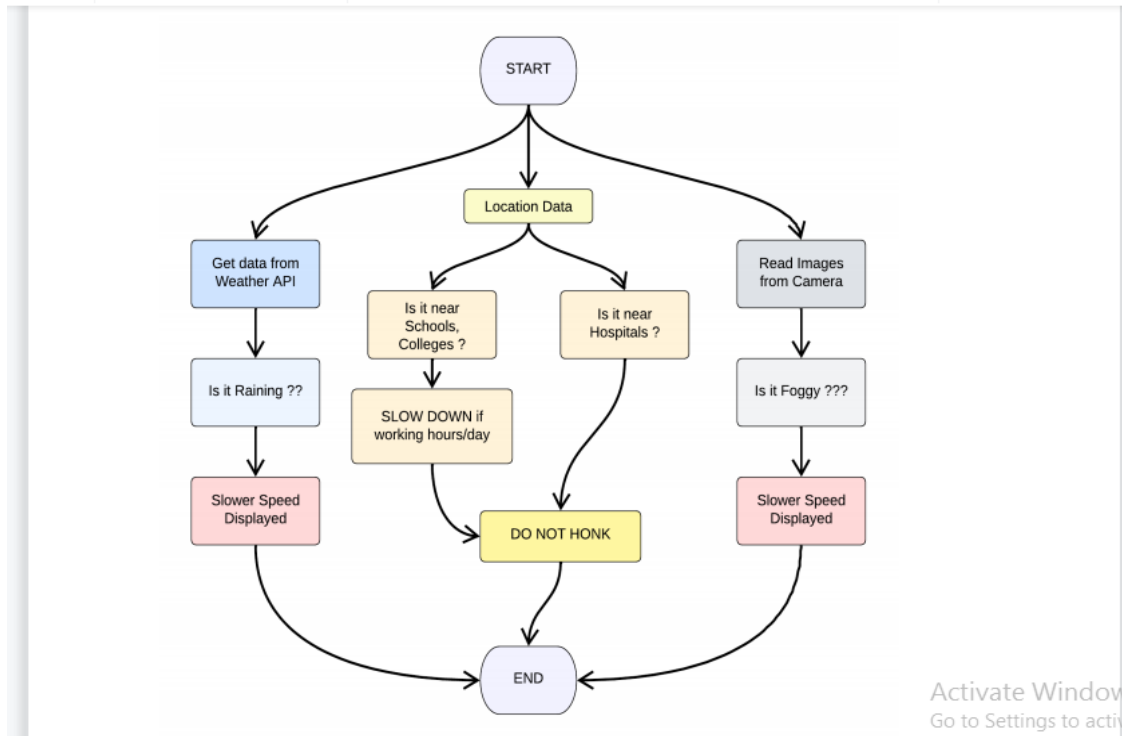
Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Will provide the clear product instructions and a self-explanatory product which is simple to use.
NFR-2	Security	Cloud data must be contained within the network, collapsing to be the real-time avoidance should be avoided, and the board will be monitored constantly.
NFR-3	Reliability	Hardware will be frequently tested.
NFR-4	Performance	The smart board must provide a better user experience and deliver the accuracy output.
NFR-5	Availability	All of the functions and the user demands will be provided, depend upon the customer needs.
NFR-6	Scalability	The product is based on road safety and should cover the entire highway system.

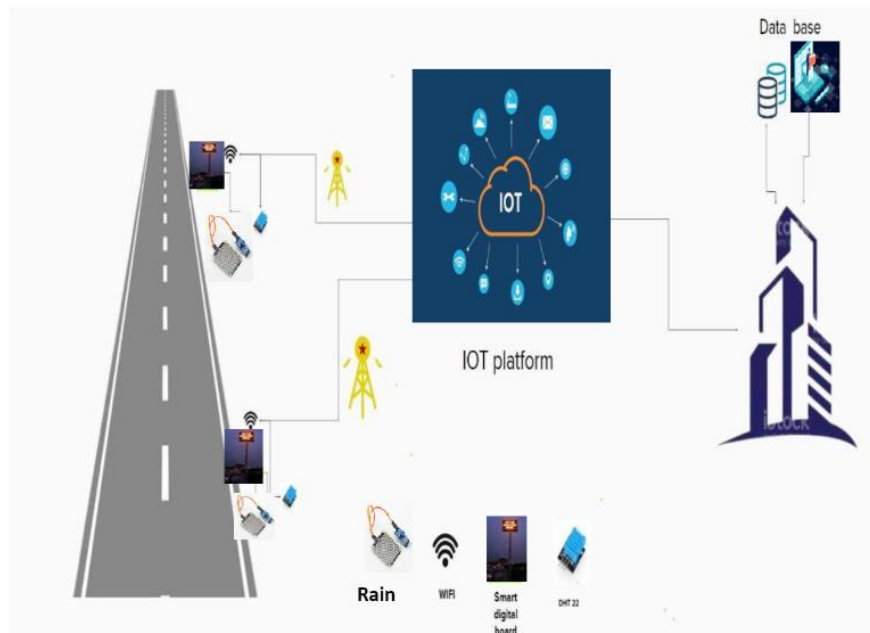
Activ
Go to 5

5. PROJECT DESIGN

5.1 Data Flow Diagrams



5.2 Solution Architecture



5.3 User Stories

Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1		US-1	Create the IBM Cloud services which are being used in this project.	6	High	Muthulekshmi, Vinishalini
Sprint-1		US-2	Configure the IBM Cloud services which are being used in completing this project.	4	Medium	Saral ,Anns
Sprint-2		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	Muthulekshmi, Vinishalini
Sprint-2		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	Muthulekshmi, Vinishalini
Sprint-3		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	Saral ,Anns

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team members
Sprint-3		US-2	Create a Node-RED service.	10	High	Saral ,Anns
Sprint-3		US-1	Develop a python script to publish random sensor data such as temperature, moisture, soil and humidity to the IBM IoT platform	7	High	Muthulekshmi, Vinishalini
Sprint-3		US-2	After developing python code, commands are received just print the statements which represent the control of the devices.	5	Medium	Muthulekshmi, Vinishalini
Sprint-4		US-3	Publish Data to The IBM Cloud	8	High	Saral ,Anns
Sprint-4		US-1	Create Web UI in Node- Red	10	High	Saral ,Anns
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	Muthulekshmi, Vinishalini

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

STEP 1	Identify the problem
---------------	-----------------------------

STEP 2	Prepare an abstract, problem statement
STEP 3	List required objects needed
STEP 4	Create a code and run it
STEP 5	Make a prototype
STEP 6	Test with the created code and check the designed prototype is working
STEP 7	Solution for the problem is found

6.2 Reports

SPRINT 1

WOKWI

SAVE SHARE final_iot.ino copy Docs

sketch.ino diagram.json libraries.txt Library Manager

```

1 #include <WiFi.h> //library for wifi
2 #include <PubSubClient.h> //library for MQTT
3 #include "DHT.h" // Library for dht11
4 #define DHTPIN 5 // what pin we're connected to
5 #define DHTTYPE DHT22 // define type of sensor DHT 11
6
7 DHT dht (DHTPIN, DHTTYPE); // creating the instance by passing pin and type of
8
9 void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
10
11 //-----credentials of IBM Accounts-----
12
13 #define ORG "psh4py" //IBM ORGANIZATION ID
14 #define DEVICE_TYPE "alert-device" //Device type mentioned in ibm watson IOT Platform
15 #define DEVICE_ID "4571" //Device ID mentioned in ibm watson IOT Platform
16 #define TOKEN "12345678" //Token
17 String data3;
18 float h, t;
19
20
21 //----- Customise the above values -----
22 char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; // Server Name
23 char publishTopic[] = "iot-2/evt/Data/fmt/json"; // topic name and type of event
24 char subscribetopic[] = "iot-2/cmd/command/fmt/json"; // cmd REPRESENT command
25 char authMethod[] = "use-token-auth"; // authentication method
26 char token[] = TOKEN;
27 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID; //client id
28
29
30 //-----

```

Simulation

00:09.316 76%

```

{"temp":1.30,"humidity":86.00,"North":0,"South":0,"East":0,"West":0}
Publish ok
temp:1.30
humidity:86.00
Sending payload:
{"temp":1.30,"humidity":86.00,"North":0,"South":0,"East":0,"West":0}
Publish ok

```

Go to Settings to activate Windows.

9:25 AM 11/17/2022

SPRINT 2

IBM Watson IoT Platform

muthuroc07@gmail.com ID: ebp7p7

Browse Action Device Types Interfaces

Add Device +

12345 Disconnected NodeMCU Device Nov 7, 2022 7:39 PM

Identity Device Information Recent Events State Logs

The recent events listed show the live stream of data that is coming and going from this device.

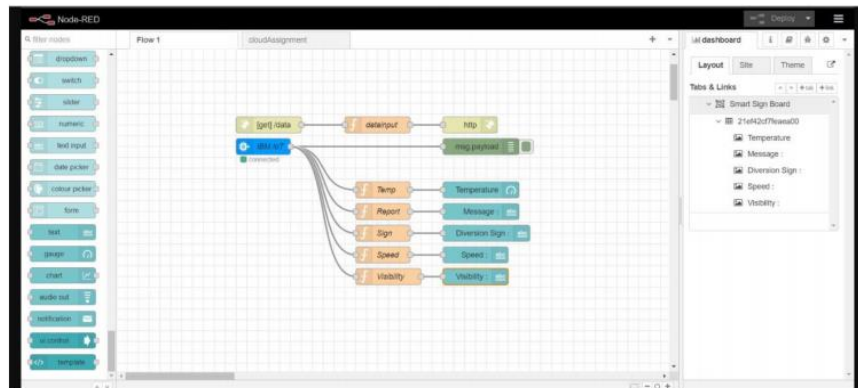
Event	Value	Format	Last Received
eventflow	{"randomNumber":100,"temp":95,"HUM":84}	json	a few seconds ago
eventflow	{"randomNumber":37,"temp":94,"HUM":81}	json	a minute ago
eventflow	{"randomNumber":7,"temp":98,"HUM":65}	json	2 minutes ago
eventflow	{"randomNumber":35,"temp":92,"HUM":91}	json	3 minutes ago
eventflow	{"randomNumber":70,"temp":98,"HUM":73}	json	4 minutes ago

1 Simulation running

Activate Windows Go to Settings to activate Windows.

Step – 3:

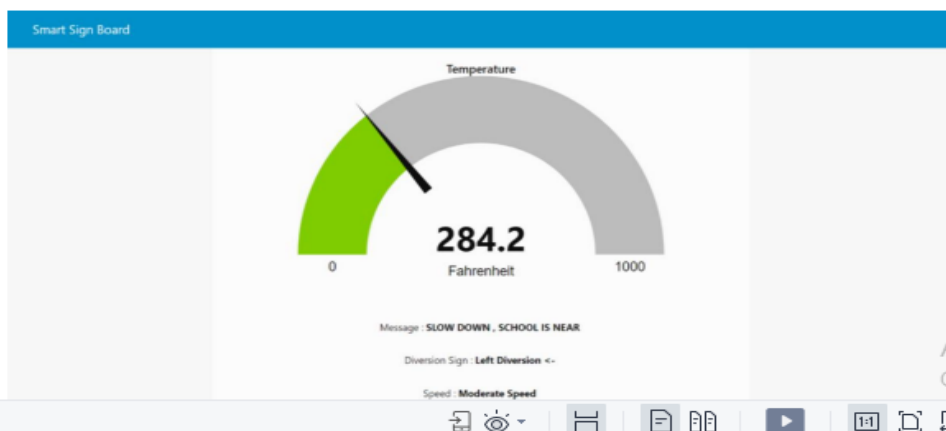
Creating Node-Red:



Find and Replace

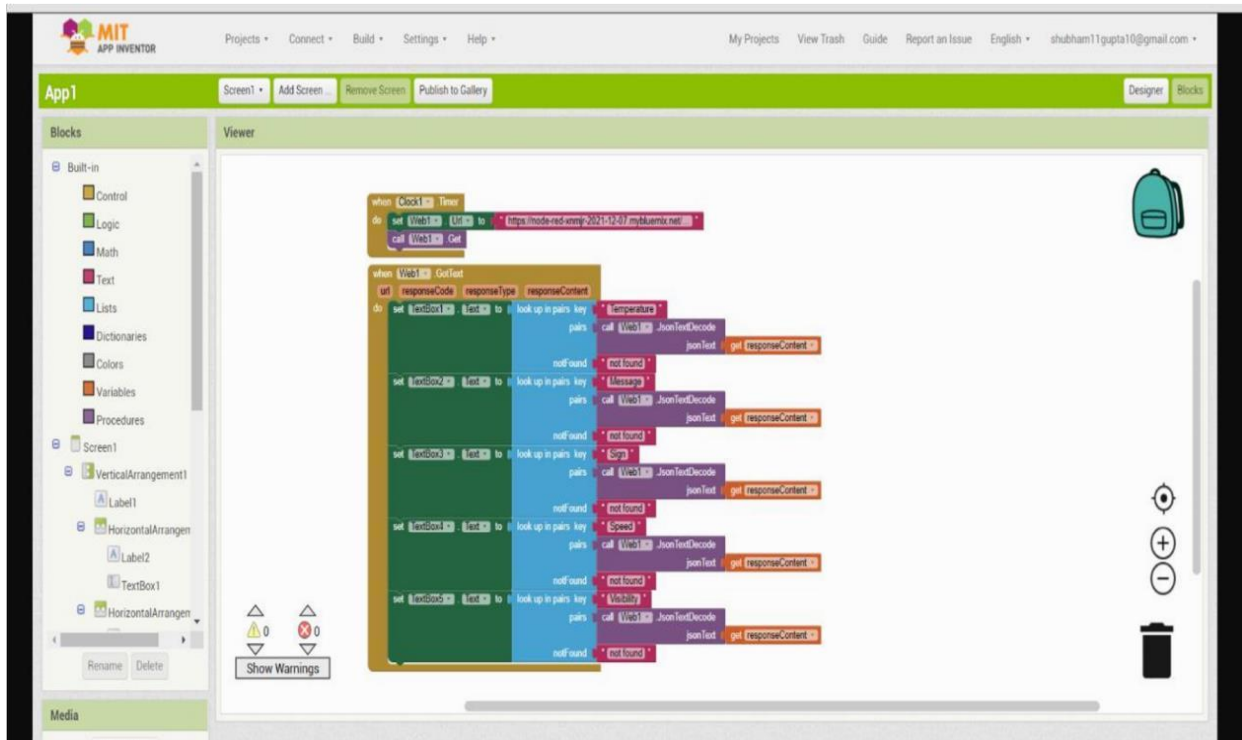
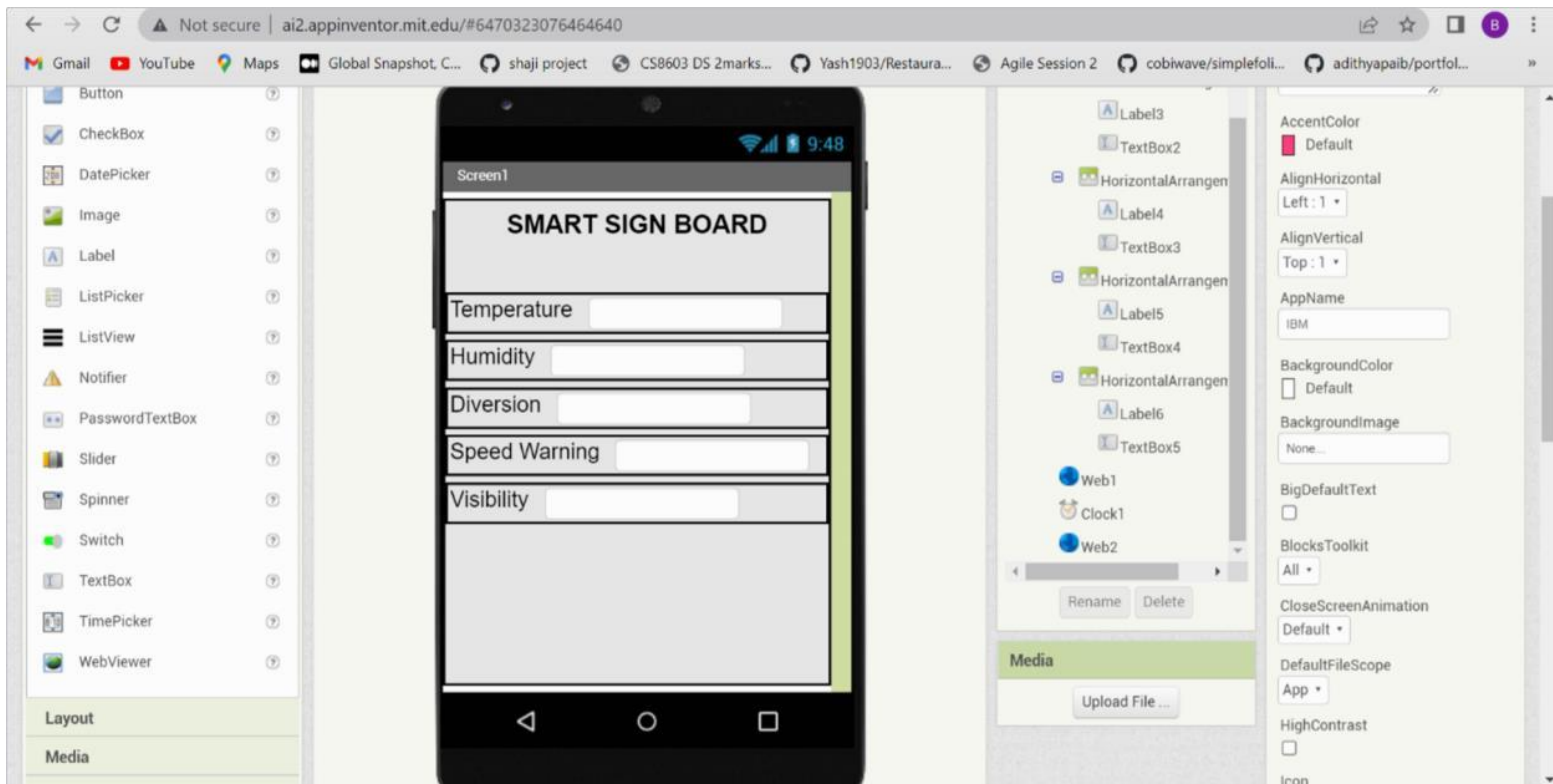
Activate Windows
Go to Settings to activate Windows

Altering Link using ui to get the Graphical output :



Activate Windows
Go to Settings to activate Windows

SPRINT 3



SPRINT 4

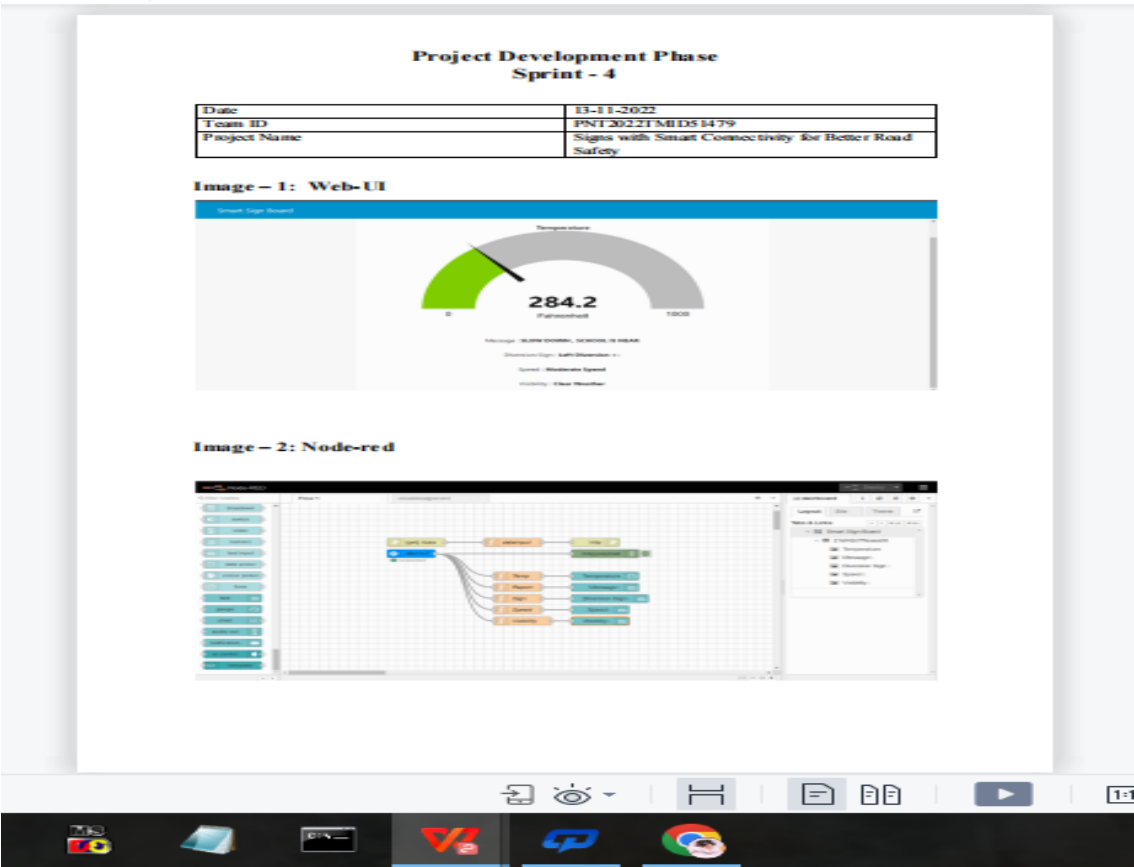
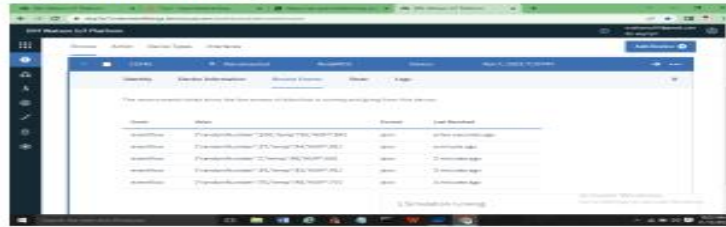


Image – 3: IOT Device output



Name	Model	Device	Last Received
device1	IBM Watson IoT Platform	device1	2018-10-10 10:10:10
device2	IBM Watson IoT Platform	device2	2018-10-10 10:10:10
device3	IBM Watson IoT Platform	device3	2018-10-10 10:10:10
device4	IBM Watson IoT Platform	device4	2018-10-10 10:10:10
device5	IBM Watson IoT Platform	device5	2018-10-10 10:10:10

Image – 4: MIT App inventor Front end



7. CODING & SOLUTIONING

7.1 Feature 1

- IoT device
- IBM Watson Platform
- Node red
- Web UI
- MIT App Inventor
- Python code

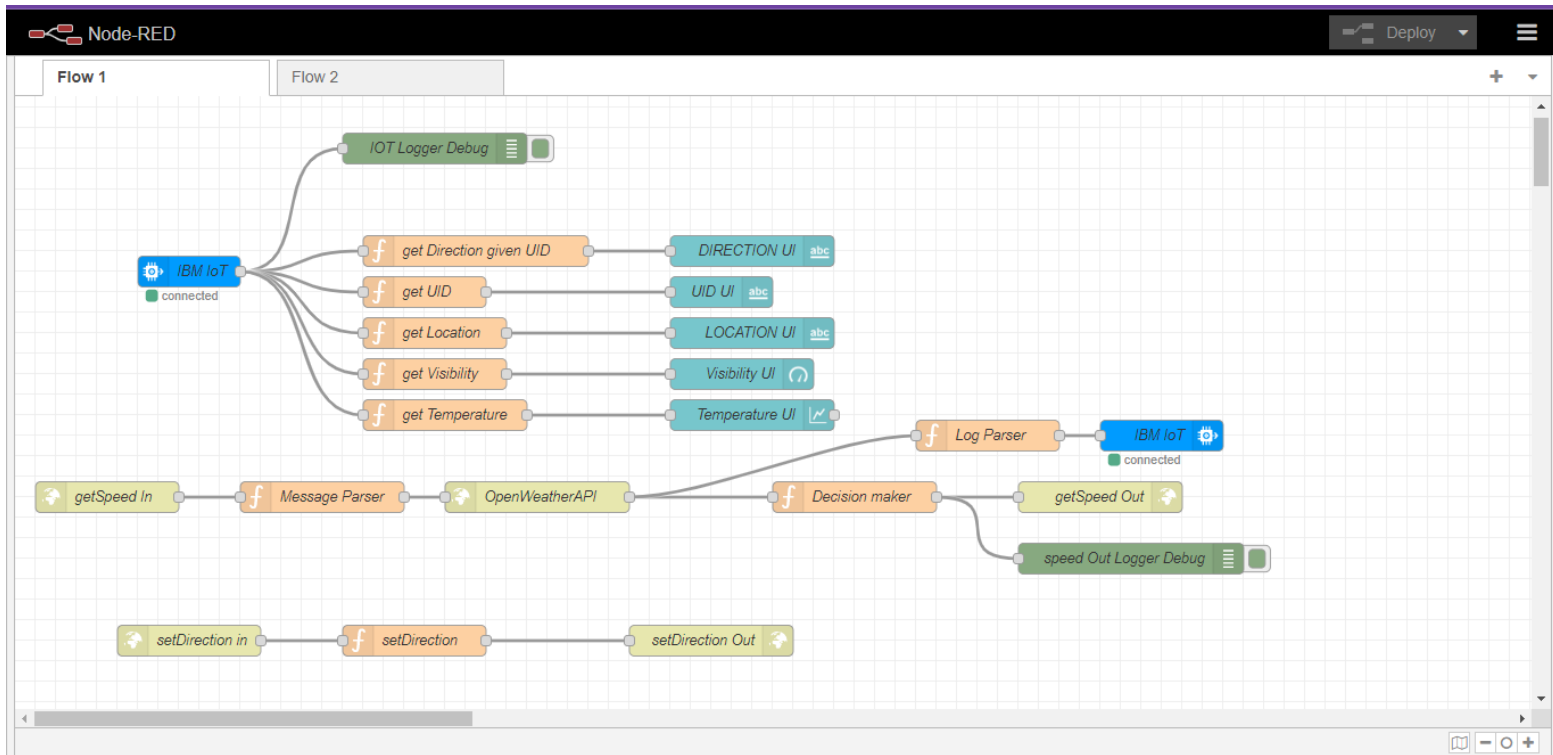
7.2 Feature 2

- Login
- Verification
- Showing the features
- Adding rating

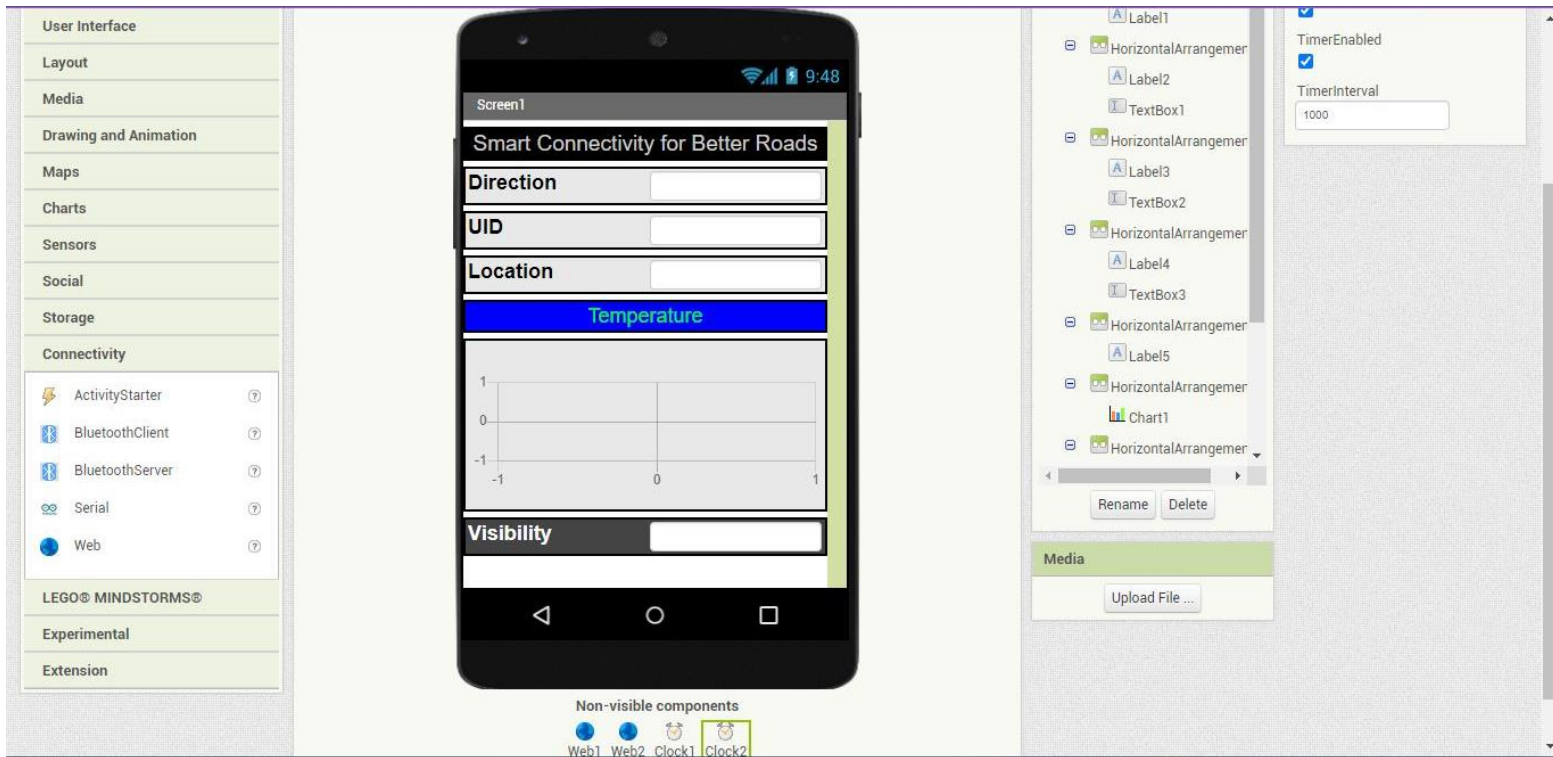
8. TESTING AND RESULTS

8.1 Test Cases

Test case 1



Test case 2



Test case 3

The screenshot shows the 'Smart_Connectivity' app interface. The top bar includes 'Screen1', 'Add Screen ...', 'Remove Screen', and 'Publish to Gallery'. The left sidebar lists 'Blocks' (Built-in, Control, Logic, Math, Text, Lists, Dictionaries, Colors, Variables, Procedures) and 'Screen1' components (HorizontalArranger, Label1, Label2, TextBox1). The main 'Viewer' area displays a code editor with the following logic:

```
when Clock1 - Timer
do
  set Web1 - Url to http://169.51.194.120:31149/setDirection?uid=250...
  call Web1 - Get

when Web1 - GotText
url responseCode responseType responseContent
do
  set TextBox2 - Text to look up in pairs key uid
  pairs call Web1 - JsonTextDecode jsonText get responseContent
  notFound not found
  set TextBox1 - Text to look up in pairs key dir
  pairs call Web1 - JsonTextDecode jsonText get responseContent
  notFound not found
```

Below the code editor are two warning icons (yellow triangle and red X) with a 'Show Warnings' button. The right sidebar contains a backpack icon, a target icon, and zoom in/out buttons.

Test case 4

The screenshot shows the 'Smart Connectivity for Better Roads' app interface. The left sidebar lists 'User Interface' components: Layout, Media, Drawing and Animation, Maps, Charts, Sensors, Social, Storage, and Connectivity. The 'Connectivity' section is expanded, showing 'ActivityStarter', 'BluetoothClient', 'BluetoothServer', 'Serial', and 'Web'. The main 'Viewer' area displays a mobile app preview with the following components: 'IOT Road', 'Smart Connectivity for Better Roads', 'Direction', 'UID', 'Location', 'Temperature', 'Temp' (a line graph showing temperature over time), and 'Visibility'. The right sidebar lists 'Non-visible components' (Web1, Clock1, Clock2, Web2) and 'Media' (Light_blue.png, download.jpg, istockph...x1024.jpg). The bottom bar contains 'Web1', 'Clock1', 'Clock2', and 'Web2'.

IOT Road

Smart Connectivity for Better Roads

Direction

L

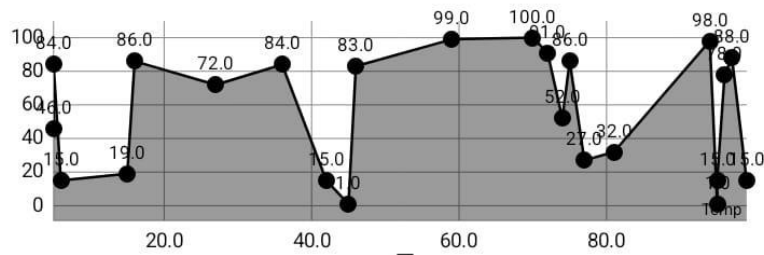
UID

2504

Location

Chennai, IND

Temperature



Visibility

89



III

O

<

8.ADVANTAGES

1. Traffic signals help for movement of traffic securely without any collision.
2. They can reduce the number of accidents on roads like pedestrian accident and right-angle collision of two cars
3. Signals can increase the capacity of traffic handling at the intersection.
4. The traffic signals help for the safe movement of slow-moving traffic by interrupting heavy traffic at regular intervals.

9. DISADVANTAGES

- Network issues may arise.

10. CONCLUSION

Almost all the countries across the globe strive to meet the demand for safe, fast, and reliable road services. Lack of operational efficiency and reliability, safety, and security issues, besides aging road systems and practices are haunting various countries to bring about a change in their existing road system infrastructure. Often, they suffer from the lack in smart technologies and latest technological updates to provide the most efficient services. This is expected to induce to build traffic systems that are smarter and more efficient. Most significant improvements have been evidenced by more informative and user-friendly websites, mobile applications for real-time information about vehicles in motion, and weather condition, construction works, schools hospitals nearby. With the rise of Industry, companies can now ensure that they are prepared to avoid the surprise of equipment downtime. This also help to avoid accidents like death, Like above mentioned, the developed application of our project can lead the passenger who travel can travel safely without any fear.

11. FUTURE SCOPE

This application is ensured for safety for the peoples while they are travelling alone as well as they travel with their family or friends.

In future, this application may also be used improve the safety the peoples who travel through it . By further enhancement of the application the peoples can explore more features regarding their safety.

12.APPENDIX

There are 3 flows in the above Node RED flow. They are

Node RED UI flow

/getSpeed API flow

/setDirection API flow

1. Node RED UI flow :

UI FLOW

"IBM IOT" node connects the backend to Node RED UI

The function nodes such as "get Direction given UID", "get UID", "get Location", "get Visibility" & "get Temperature" extract the respective data out and provides them to the UI nodes "Direction UI", "UID UI", "Location UI", "Visibility UI" & "Temperature UI".

```
// get Direction given UID
msg.payload = global.get(String(msg.payload.uid));
return msg;
```

```
// get UID
msg.payload = msg.payload.uid;
return msg;
```

```
// get Location
msg.payload = msg.payload.location;
return msg;
```

```
// get Visibility
msg.payload = msg.payload.visibility;
return msg;
```

```
// get Temperature
msg.payload = msg.payload.temperature;
return msg;
```

"IOT Logger Debug" node logs the data at debugger.

2. /getSpeed API flow :

getSpeedAPI

"getSpeed In" node is an http end point. It accepts parameters like microcontroller UID, location, school & hospital zones info.

"Message Parser" node parses the data and passes on only required information to the next node

```
global.set("data",msg.payload);
```

```
msg.payload.q = msg.payload.location;  
msg.payload.appid = "bf4a8d480ee05c00952bf65b78ae826b";  
return msg;
```

"OpenWeatherAPI" node is a http request node which calls the OpenWeather API and send the data to the next node.

"Log Parser" node extracts specific parameters from the weather data and and sends it to the next node.

```
weatherObj = JSON.parse(JSON.stringify(msg.payload));  
localityObj = global.get("data");
```

```
var suggestedSpeedPercentage = 100;
```

```
var preciseObject = {  
  temperature : weatherObj.main.temp - 273.15,  
  location : localityObj.location,  
  visibility : weatherObj.visibility/100,  
  uid : localityObj.uid,  
  direction : global.get("direction")  
};
```

```
msg.payload = preciseObject;
```

```
return msg;
```

"IBM IoT" node here (IBM IoT OUT) connects the "IBM IoT" node (IBM IoT IN) metioned in the Node RED UI flow which enables UI updation and logging.

"Decision Maker" node processes the weather data and other information from the micro controller to form the string that is to be displayed at the Sign Board

```
weatherObj = JSON.parse(JSON.stringify(msg.payload));  
localityObj = global.get("data");
```

```
var suggestedSpeedPercentage = 100;
```

```
var preciseObject = {
```

```

temperature : weatherObj.main.temp - 273.15,
weather : weatherObj.weather.map(x=>x.id).filter(code => code<700),
visibility : weatherObj.visibility/100
};

if(preciseObject.visibility<=40)
    suggestedSpeedPercentage -=30

switch(String(preciseObject.weather)[-1]) // https://openweathermap.org/weather-conditions refer
weather codes meaning here
{
    case "0" : suggestedSpeedPercentage -=10;break;
    case "1" : suggestedSpeedPercentage -=20;break;
    case "2" : suggestedSpeedPercentage -=30;break;
}

msg.payload = preciseObject;

var doNotHonk = 0;
if(localityObj.hospitalZone=="1"||localityObj.schoolZone=="1")
    doNotHonk = 1;

var returnObject = {
    suggestedSpeed : localityObj.usualSpeedLimit*(suggestedSpeedPercentage/100),
    doNotHonk : doNotHonk
}

msg.payload = String(returnObject.suggestedSpeed) + " kmph \n\n" +
(returnObject.doNotHonk==1?"Do Not Honk":"" ) + "$" + global.get(String(localityObj.uid));

return msg;

```

"getSpeed Out" node returns a http response for the request at node "getSpeed In".

"speed Out Logger Debug" logs the data for debugging.

3. /setDirection API flow :

UI FLOW

"setDirection In" node is an http end point. It accepts parameters like microcontroller UID & direction.

"set Direction Function" node sets the direction for the given UID.

```
global.set(String(msg.payload.uid),msg.payload.dir);  
return msg;
```

"setDirection Out" node returns a http response for the request at node "setDirection In".

Circuit Diagram:

ESP 32 CODE:

```
#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 = "";  
  
#define TFT_DC 2  
#define TFT_CS 15  
  
Adafruit_ILI9341 tft = Adafruit_ILI9341(TFT_CS, TFT_DC);  
  
String myLocation = "Chennai,IN";  
String usualSpeedLimit = "70"; // kmph  
  
int schoolZone = 32;  
int hospitalZone = 26;  
  
int uid = 2504; // ID Unique to this Micro Contoller  
String getString(char x)  
{  
  String s(1, x);  
  return s;  
}
```

```

String stringSplitter1(String fullString,char delimiter='$')
{
    String returnString = "";
    for(int i = 0; i<fullString.length();i++) {
        char c = fullString[i];
        if(delimiter==c)
            break;
        returnString+=String(c);
    }
    return(returnString);
}

String stringSplitter2(String fullString,char delimiter='$')
{
    String returnString = "";
    bool flag = false;
    for(int i = 0; i<fullString.length();i++) {
        char c = fullString[i];
        if(flag)
            returnString+=String(c);
        if(delimiter==c)
            flag = true;
    }
    return(returnString);
}

void rightArrow()
{
    int refX = 50;
    int refY = tft.setCursorY() + 40;
    tft.fillRect(refX,refY,100,20,ILI9341_RED);
    tft.fillTriangle(refX+100,refY-30,refX+100,refY+50,refX+40+100,refY+10,ILI9341_RED);
}

void leftArrow()
{
    int refX = 50;
    int refY = tft.setCursorY() + 40;
    tft.fillRect(refX+40,refY,100,20,ILI9341_RED);
    tft.fillTriangle(refX+40,refY-30,refX+40,refY+50,refX,refY+10,ILI9341_RED);
}

void upArrow()

```

```

{
int refX = 125;
int refY = tft.setCursorY() + 30;
tft.fillTriangle(refX-40,refY+40,refX+40,refY+40,refX,refY,ILI9341_RED);
tft.fillRect(refX-15,refY+40,30,20,ILI9341_RED);
}
String APICall() {
  HTTPClient http;
  String url = "https:// 169.51.194.120:31149/getSpeed?";
  url += "location="+myLocation+"&";
  url += "schoolZone="+((String)digitalRead(schoolZone)).toString()+"&";
  url += "hospitalZone="+((String)digitalRead(hospitalZone)).toString()+"&";
  url += "usualSpeedLimit="+((String)usualSpeedLimit).toString()+"&";
  url += "uid="+((String)uid).toString();
  http.begin(url.c_str());
  int httpResponseCode = http.GET();

  if (httpResponseCode>0) {
    String payload = http.getString();
    http.end();
    return(payload);
  }
  else {
    Serial.print("Error code: ");
    Serial.println(httpResponseCode);
  }
  http.end();
}

void myPrint(String contents) {
  tft.fillScreen(ILI9341_BLACK);
  tft.setCursor(0, 20);
  tft.setTextSize(4);
  tft.setTextColor(ILI9341_RED);
  //tft.println(contents);
  tft.println(stringSplitter1(contents));
  String c2 = stringSplitter2(contents);
  if(c2=="s") // represents Straight
  {
    upArrow();
  }
}

```



```

}
if(c2=="l") // represents left
{
leftArrow();
}
if(c2=="r") // represents right
{
rightArrow();
}
}

void setup() {
  WiFi.begin(ssid, password, 6);
  tft.begin();
  tft.setRotation(1);
  tft.setTextColor(IL19341_WHITE);
  tft.setTextSize(2);
  tft.print("Connecting to WiFi");
  while (WiFi.status() != WL_CONNECTED) {
    delay(100);
    tft.print(".");
  }
  tft.print("\nOK! IP=");
  tft.println(WiFi.localIP());
}

void loop() {
  myPrint(APICall());
  delay(100);
}

```

Output :

Node RED Dashboard :

LINK TO NODE RED DASHBOARD

13.2 GitHub

GitHub link:

<https://github.com/IBM-EPBL/IBM-Project-42700-1660707213>