Internet Of Things

In partial fulfillment for the completion of

Signs with Smart Connectivity For Better Road Safety

PROJECT REPORT

Submitted by

SNEHA R	**(19CEIT033)**
NIVETHA L	**(19CEIT022)**
JEYASHRI S	**(19CEIT012)**
PAWANA PRUDHVI TSV	**(19CEIT024)**
NAVEEN V	**(19CEIT021)**

Date	**18 November 2022**
Team ID	**PNT2022TMID07843**

BACHELOR OF TECHNOLOGY

IN OF

INFORMATION TECHNOLOGY

SNS COLLEGE OF ENGINEERING ,COIMBATORE.

During the academic year
2022-2023 (Odd Semester)

TABLE OF CONTENTS

1. INTRODUCTION:

Traffic has recently become a big issue for the people of India. As a result, it wastes valuable time, fuel, and electricity. The Internet of Things (IOT) is a networkof electrical appliances, cars, physical devices, and other items that are integrated with electronics, actuators, sensors, software, and connectivity, allowing these objects to connect and share data. Each object is uniquely identified by its embedded computing system, but it may interact with the existing Internet infrastructure.

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.
By using the Weather API we can get the weather reports based on which we can set the speed limit to particular area. 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. There are three switches through which you can switch the display to different modes.

1.2 Purpose:

Due to this heavy traffic, the number of road accidents are increased which is a major issue. Our project helps to decrease the number of road accidents usingsmart connected sign boards using Internet Of things (IOT).

2. LITERATURE SURVEY:

2.1 Existing System:

The individual traffic signals are connected with traffic control system to perform network wide traffic operation .These control systems contain a central computer, a communication network, and intersection traffic signals. Coordination of controlsystem can be implemented through different techniques like time-base, hardwired interconnection method. Coordination between traffic signals and agencies requires the development of data sharing and traffic signal control agreements. A traffic-signal system has only one purpose i.e. to deliver signal timings to the driver. The system provides features that improve the traffic engineer's ability to achieve this goal. These are primarily access features. They provide access to the intersection signal controller for maintenance and operations. The more complete and convenient the access, the more efficient the operator will be and the more effective the system. In addition to control the traffic signals, modern technology also provide surveillance capabilities, including different kinds of video surveillance and traffic detection.

2.2 References :

1. https:/ www.hindawi.com/journals/jat/2022/5829607/ 2.https:/

en.wikipedia.org/wiki/Automotive_safety

3.https:/ www.powerbulbs.com/us/blog/2020/01/yellow-or-whiter-light

2.3 Problem Statement Definition :

This project will replace static signs with smart signs that can adjust speed restrictions based on the weather and climate, display detour instructions in the event of an accident, and display alert messages in the event of hospitals, schools, or roadworks.

3. IDEATION & PROPOSED SOLUTION :

3.1 Empathy Map Canvas :

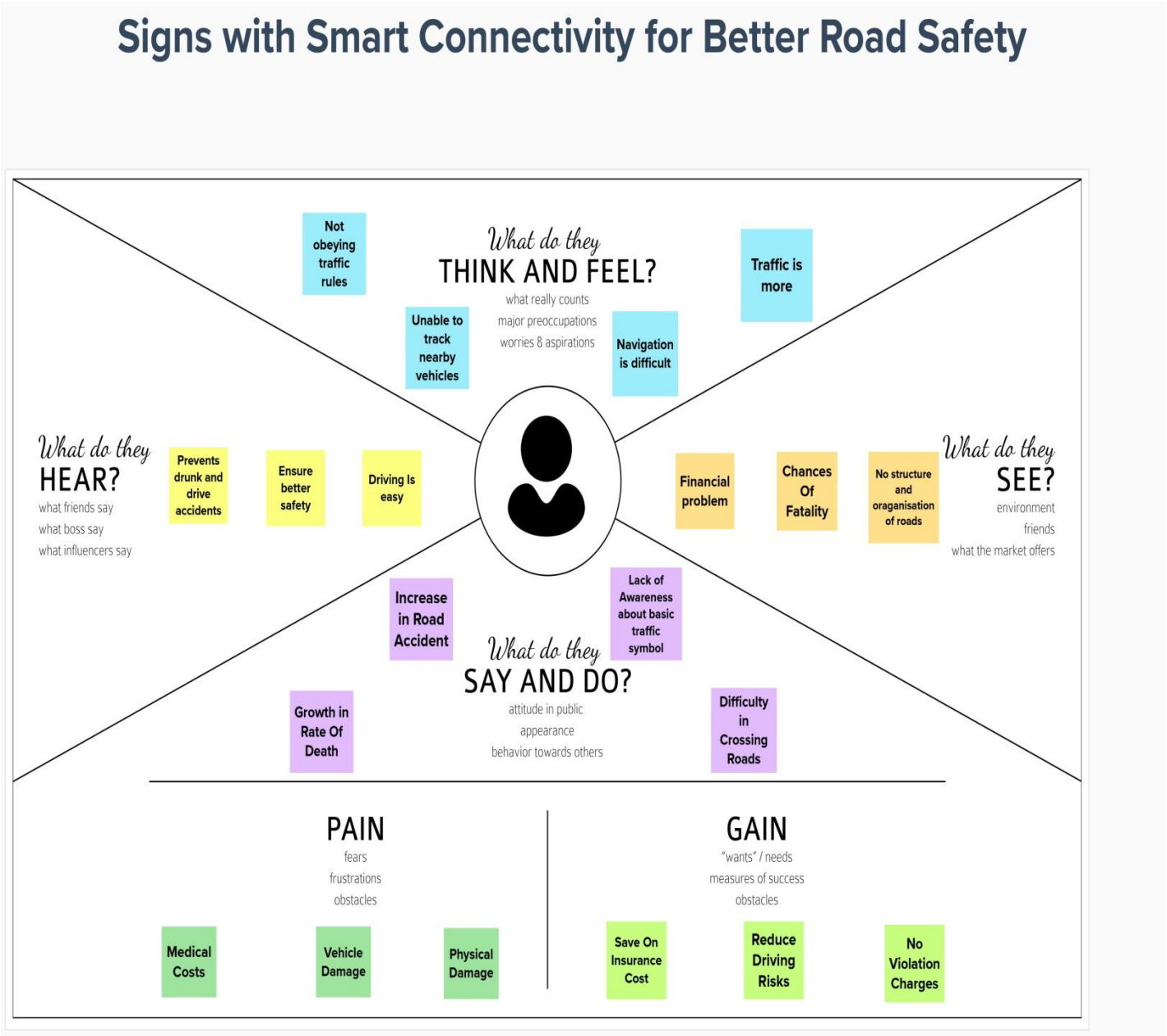
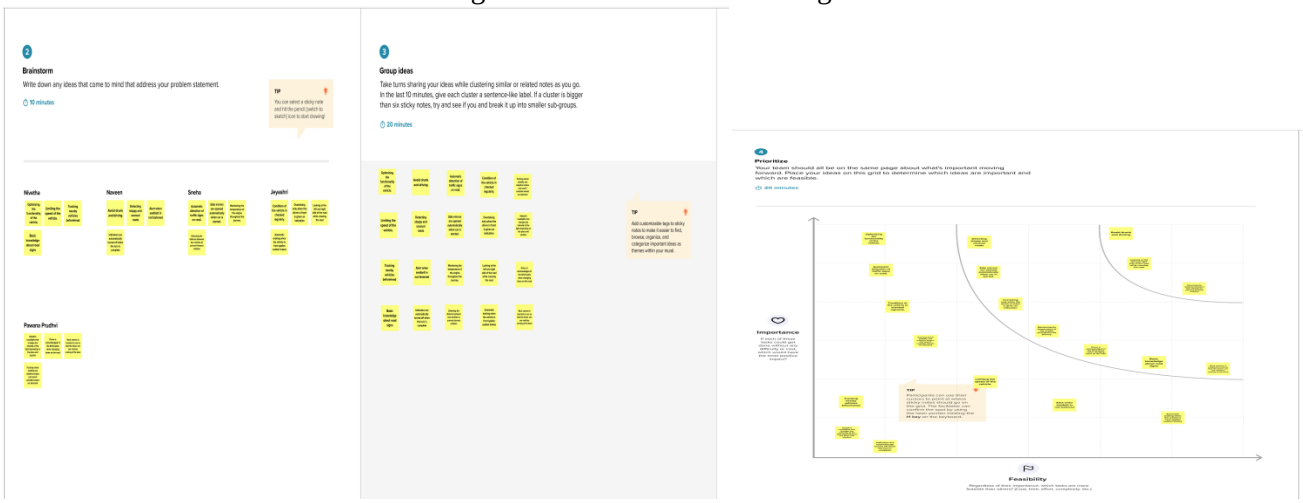


Fig 3.1 Empathy Map

3.2 Ideation & Brainstorming :

Fig 3.2 Ideation & Brainstroming



3.3 Proposed Solution :

S. No	Parameter	Description
1.	Problem Statement (Problemto be solved)	Signs with smart connectivity for Better Road Safety are intended to control speed,increase safety, and display the latest weather information.
2.	Idea / Solution description	Replacing traditional roadside signage with IOT-enabled smart ones. Smart signs are built using LED and the Internet of Things.
3.	Novelty / Uniqueness	Due to the use of LEDs, they can be seen from a distance. It is possible to view them from a distance since LEDs are used. These specifics were obtainedfrom a weather-tracking app. Additionally, it gives information about nearby places like hospitals, schools, etc. so that customers may make decisionsbased on that knowledge, such as speeding.

4.	Social Impact / Customer Satisfaction	On the department of road safety, these are clearlyfelt. The avoidance of accidents can be achieved byimposing a user-set speed limit.
5.	Business Model (Revenue Model)	The government's implementation of these for common citizens is an excellent endeavor to increase public awareness. This can be funded separately by the government, which lays the groundwork for a safer environment.
6.	Scalability of the Solution	Because it is more visible than conventional signals, it has a better chance of reducing danger and could even save many lives.

3.4 Problem Solution Fit :

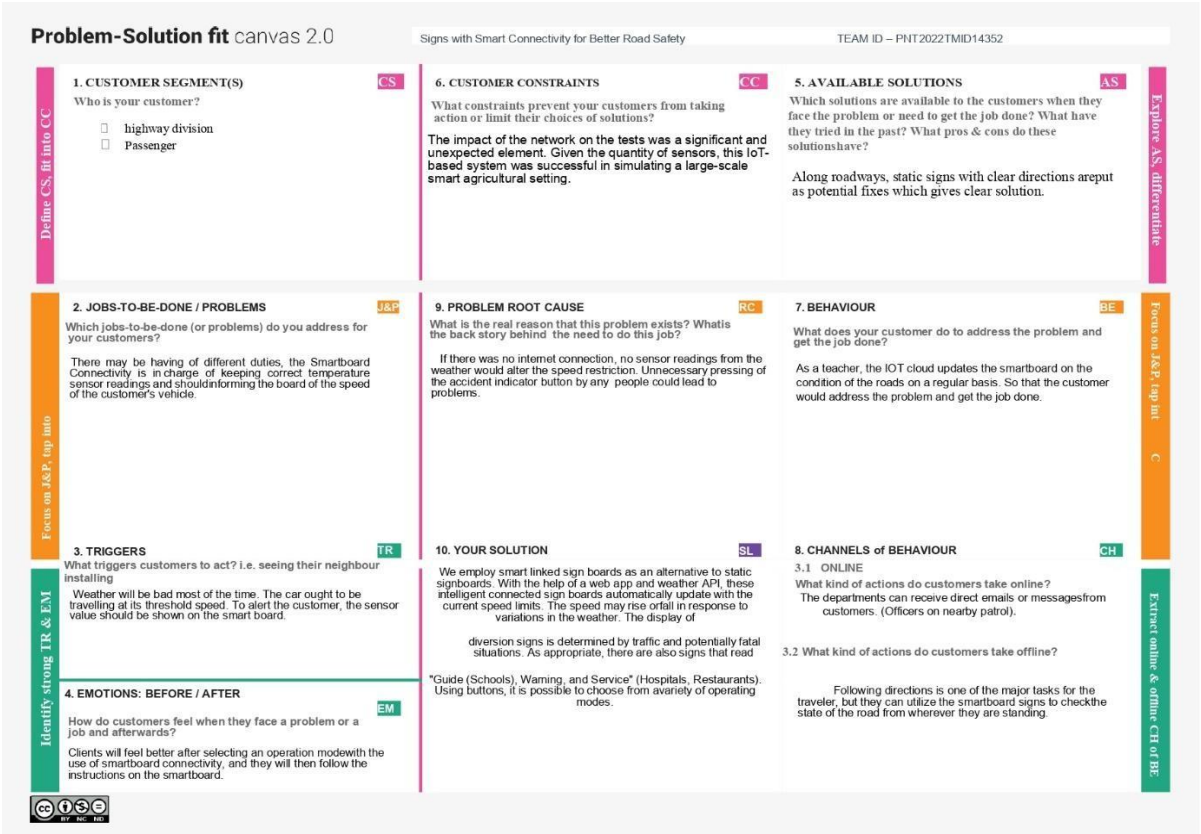


Fig 3.4 Problem Solution Fit

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 Requirements	Static signboards will be replaced with smart linkedsign 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 lengthof the road.
FR-6	Product Feedback	Will be shared through a website via Gmail

4.2 Non-Functional requirement:

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 withinthe 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, dependupon the customer needs.
NFR-6	**Scalability**	The product is based on road safety andshould cover the entire highway system.

5. PROJECT DESIGN :

5.1 Data Flow Diagram :

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information,

and where data is stored.

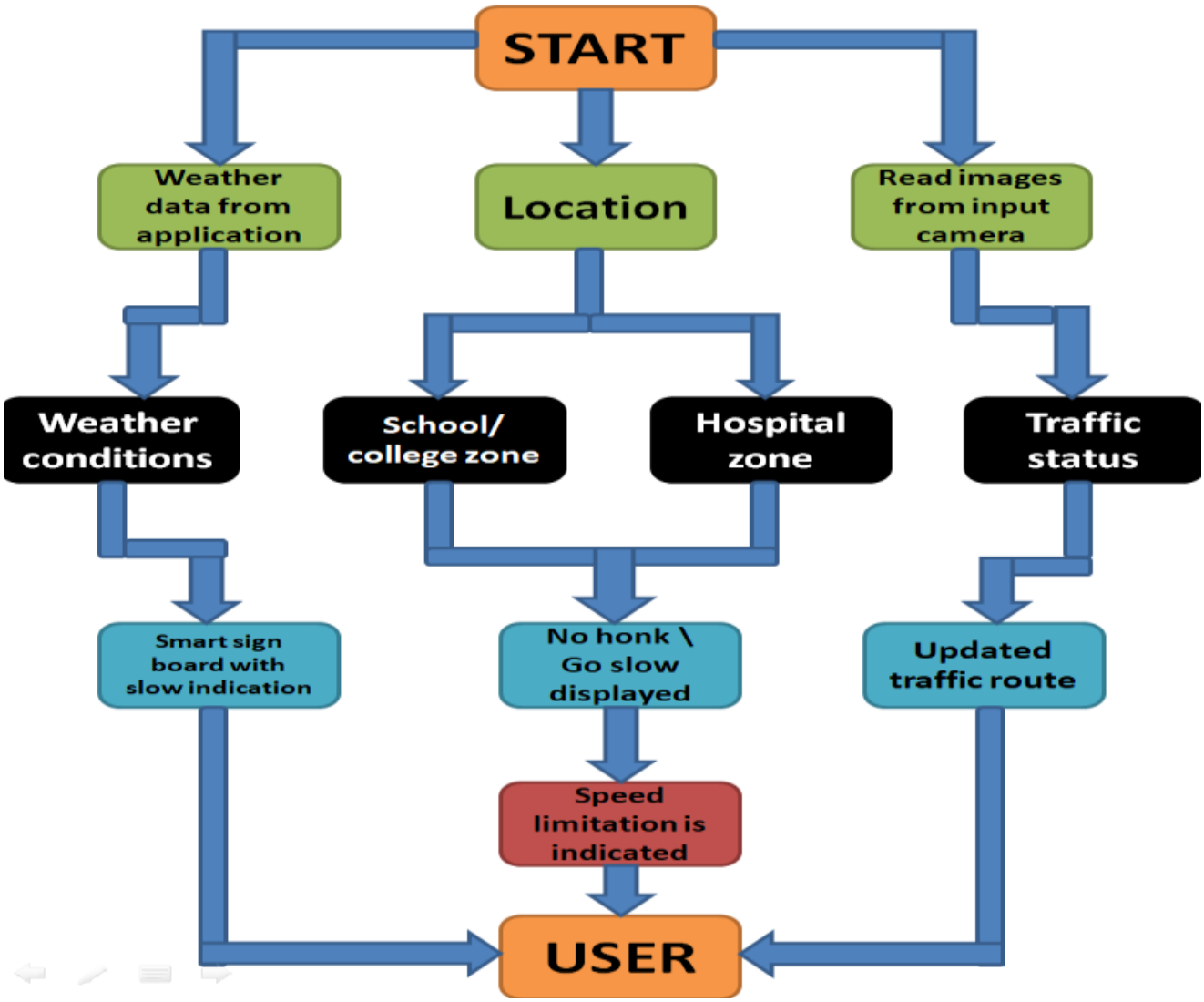


Fig 5.1 Data Flow Diagram - Signs with smart connectivity for better road safety

5.2 Solution & Technical Architecture :

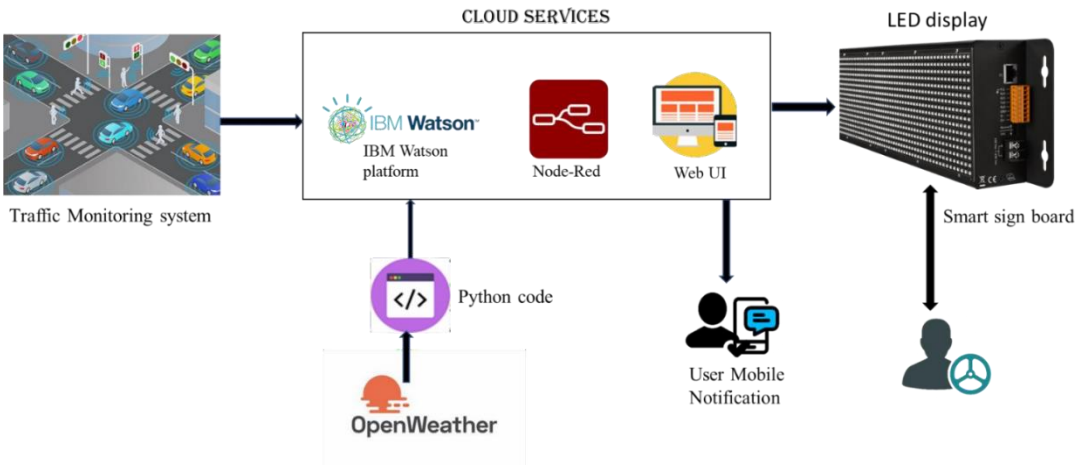


Fig 5.2 Solution & Technical Architecture - Signs with smart connectivity for better road safety

GUIDELINES:

• 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.

• Based on the traffic and fatal situations the diversion signs are displayed.

• Guide(Schools), Warning and Service(Hospitals, Restaurant) signs are also displayed accordingly.

• Different modes of operations can be selected with the help of buttons.

Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g. Web UI, Mobile App, Chatbot etc.	HTML, CSS, JavaScript / Angular Js / React Jsetc.
2.	Application Logic-1	Logic for a process in the application	Java / Python
3.	Application Logic-2	Logic for a process in the application	IBM Watson STT service
4.	Application Logic-3	Logic for a process in the application	IBM Watson Assistant
5.	Database	Data Type, Configurations etc.	MySQL, NoSQL, etc.
6.	Cloud Database	Database Service on Cloud	IBM DB2, IBM Cloudant etc.
7.	File Storage	File storage requirements	IBM Block Storage or Other Storage Service or Local Filesystem
8.	External API-1	Purpose of External API used in the application	IBM Weather API, etc.

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Security Implementations	Strong security system that anyone without login credentials and hackersare not allowed to enter the network.	Firewall,Firebase, cyber resiliency strategy
2.	Scalable Architecture	Easy to expand the operating rangeby increasing the bandwidth of the network.	IoT, internet.
3.	Availability	Available anytime and everywhere 24/7 as long as the user is signed intothe network.	IBM Cloud
4.	Performance	Supports a large number of users toaccess the technology simultaneously.	IBM Cloud

5.3 User stories :

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	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.	Access my account / dashboard	High	Sprint-1
Weather	openweathermap	USN-2	As a user, I want to check the weather of that location	Get the weather of that location	High	Sprint-1
IoT devices	Automation	USN-3	As a user, I want to use IoT devices for automation purposes	Get the work done without manual effort	High	Sprint-2
Python code	Random data	USN-4	As a user, I want to give some input to the devices for performing some action to complete the tasks very easily	Get the data workflow	Medium	Sprint-1
IBM Cloud	Cloud services	USN-5	As a user, I want to deploy these application for public version	Useful for all domain users	High	Sprint-1
Node-Red	Integration	USN-6	As a user, I want to integrate the applications with hardware	To precise for linear workflow	Medium	Sprint-3
Web UI	Interaction	USN-7	As a user, I want to interact with the digital products	To interact with the users	Medium	Sprint-2
Data validation	Checking accuracy	USN-8	As a user, I can check the ability and accuracy of the model in obtaining the required information	Check the capability of the model	High	Sprint-2
Data extraction	Obtaining the data	USN-9	As a user, I can retrieve the result data from the application for data storage for further uses	Download the result in the form of data	High	Sprint-3

6. PROJECT PLANNING & SCHEDULING :

6.1 Sprint Planning & Estimation:

Sprint	Functional Requirement (Epic)	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Resources Initialization	Create and initialize accounts in various public APIs like OpenWeatherMap API.	1	LOW	Sneha Nivetha Jeyashri Naveen Pawana prudhvi
Sprint-1	Local Server/Software Run	Write a Python program that outputs results given the inputs like weather and location.	1	MEDIUM	Sneha Nivetha Jeyashri Naveen Pawana prudhvi
Sprint-2	Push the server/software to cloud	Push the code from Sprint 1 to cloud so it can be accessed from anywhere	2	MEDIUM	Sneha Nivetha Jeyashri Naveen Pawana prudhvi
Sprint-3	Hardware initialization	Integrate the hardware to be able to access the cloud functions and provide inputs to the same.	2	HIGH	Sneha Nivetha Jeyashri Naveen Pawana prudhvi
Sprint-4	UI/UX Optimization & Debugging	Optimize all the shortcomings and provide better user experience.	2	LOW	Sneha Nivetha Jeyashri Naveen Pawana prudhvi

6.2 Sprint Delivery Schedule:

Sprint	Total Story Point	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	31 Oct 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	07 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	14 Nov 2022

7. REQUIREMENTS :

The hardware and software requirements of the project are:

7.1 Software Requirements :

- Arduino IDE
- Weather API
- IBM Cloud Platform

7.2 Hardware Requirements :

1. NODE MCU(12-E):

New NodeMcu Lua ESP8266 CH340G ESP-12E Wireless WIFI Internet Development Board ESP12E is a WIFI enabled Arduino-alike development board, Which can dramatically reduce the redundant work for configuring and manipulating hardware. Code like arduino, but interactively in Lua scipt.

Fig 7.2.1 NODE MCU(12-E)

2. PUSH BUTTON:

A Pushbutton switch is a switch featuring a button you push to open and close a circuit. Depending on the series, Pushbutton switches could perform momentary or maintainedfunctions.

E-Switch carries numerous momentary and maintained pushbutton switches that can be non-illuminated or illuminated, with multiple LED colors and lens colors, with up to IP67 ratings for protection against dust and moisture.

Fig 7.2.2 Push Button

3. OLED DISPLAY:

OLED displays are made by placing a series of organic thin films between two conductors.When an electrical current is applied, a bright light is emitted. A simple design - which bringswith it many advantages over other display technologies.

OLEDs enable emissive displays - which means that each pixel is controlled individually and emits its own light (unlike LCDs in which the light comes from a backlighting unit). OLED displays feature great image quality - bright colors, fast motion and most importantly - very high contrast.

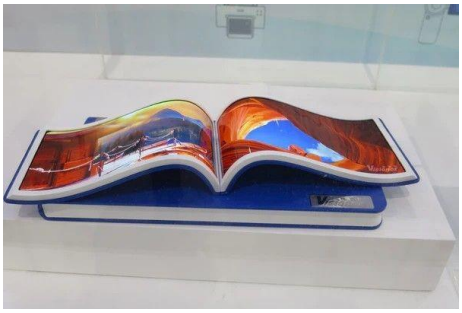


Fig 7.2.3 OLED Display

8. CODING & SOLUTIONING :

8.1 Feature 1:

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h> const
char* ssid = "SB-IOT1";
const char* password = "sb@iot11";String
command1,command2; #define ORG
"bhip5y"
#define DEVICE_TYPE "Vamsi"
#define DEVICE_ID "8500"
#define TOKEN "8500913778"
String command;
char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; char
topic[] = "iot-2/cmd/home/fmt/String";
char authMethod[] = "use-token-auth";char
token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
/ Serial.println(clientID);#include
<Wire.h>
#include <Adafruit_SSD1306.h>
#include <Adafruit_GFX.h> #define
SSD1306_LCDHEIGHT 64
/ OLED display TWI address
#define OLED_ADDR 0x3C
Adafruit_SSD1306   display(-1); #if
(SSD1306_LCDHEIGHT != 64)
#error("Height incorrect, please fix Adafruit_SSD1306.h!");
```

```
#endif
void callback(char* topic, byte* payload, unsigned int payloadLength); WiFiClient
wifiClient;
PubSubClient client(server, 1883, callback, wifiClient); void
setup() {
   display.begin(SSD1306_SWITCHCAPVCC, OLED_ADDR);
  Serial.begin(115200);
  Serial.println();
  pinMode(D1,OUTPUT);
  wifiConnect();
  mqttConnect();
}
void loop() {
  if (!client.loop()) {
    mqttConnect();
  }
delay(100);
}
void wifiConnect() {
  Serial.print("Connecting to "); Serial.print(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {delay(500);
    Serial.print(".");
  }
  Serial.print("nWiFi connected, IP address: "); Serial.println(WiFi.localIP());
}
void mqttConnect() {
```

```
  if (!client.connected()) {
    Serial.print("Reconnecting MQTT client to "); Serial.println(server); while
    (!client.connect(clientId, authMethod, token)) { Serial.print(".");
      delay(500);
    }
    initManagedDevice();
    Serial.println();
  }
}
void initManagedDevice() { if
  (client.subscribe(topic)) {
    Serial.println("subscribe to cmd OK");
  } else {
    Serial.println("subscribe to cmd FAILED");
  }
}
 void callback(char* topic, byte* payload, unsigned int payloadLength) {
  Serial.print("callback invoked for topic: "); Serial.println(topic);
  for (int i = 0; i < payloadLength; i++) {
    / Serial.println((char)payload[i]);
    command += (char)payload[i];
  }
Serial.println(command);
command1=getValue(command,',',0);
command2=getValue(command,',',1);
if(command1=="1"){
display.clearDisplay();
```

```
/ display a line of text
  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(0,10);
  display.print(command);
  /  update display with all of the above graphicsdisplay.display();
}
command ="";
command1 ="";
command2="";
}
String getValue(String data, char separator, int index)
{
    int found = 0;
    int strIndex[] = { 0, -1 };
    int maxIndex = data.length() - 1;
    for (int i = 0; i <= maxIndex && found <= index; i++) {if
        (data.charAt(i) == separator || i == maxIndex) {
            found++;
            strIndex[0] = strIndex[1] + 1; strIndex[1]
            = (i == maxIndex) ? i+1 : i;
       }
     }
    return found > index ? data.substring(strIndex[0], strIndex[1]) : "";
}
```

8.2 Feature 2:

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h> const
char* ssid = "SB-IOT1";
const char* password = "sb@iot11";String
command1,command2; #define ORG
"bhip5y"
#define DEVICE_TYPE "Vamsi"
#define DEVICE_ID "8500"
#define TOKEN "8500913778"
String command;
char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; char
topic[] = "iot-2/cmd/home/fmt/String";
char authMethod[] = "use-token-auth";char
token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
/ Serial.println(clientID);#include
<Wire.h>
#include <Adafruit_SSD1306.h>
#include <Adafruit_GFX.h> #define
SSD1306_LCDHEIGHT 64
/ OLED display TWI address
#define OLED_ADDR 0x3C
Adafruit_SSD1306   display(-1); #if
(SSD1306_LCDHEIGHT != 64)
#error("Height incorrect, please fix Adafruit_SSD1306.h!"); #endif
```

```
void callback(char* topic, byte* payload, unsigned int payloadLength); WiFiClient
wifiClient;
PubSubClient client(server, 1883, callback, wifiClient); void
setup() {
   display.begin(SSD1306_SWITCHCAPVCC, OLED_ADDR);
  Serial.begin(115200);
  Serial.println();
  pinMode(D1,OUTPUT);
  wifiConnect();
  mqttConnect();
}
void loop() {
  if (!client.loop()) {
    mqttConnect();
  }
delay(100);
}
void wifiConnect() {
  Serial.print("Connecting to "); Serial.print(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {delay(500);
    Serial.print(".");
  }
  Serial.print("nWiFi connected, IP address: "); Serial.println(WiFi.localIP());
}
void mqttConnect() {
  if (!client.connected()) {
    Serial.print("Reconnecting MQTT client to "); Serial.println(server);
```

```
    while (!client.connect(clientId, authMethod, token)) {
      Serial.print(".");
      delay(500);
    }
    initManagedDevice();
    Serial.println();
  }
}
void initManagedDevice() { if
  (client.subscribe(topic)) {
    Serial.println("subscribe to cmd OK");
  } else {
    Serial.println("subscribe to cmd FAILED");
  }
}
void callback(char* topic, byte* payload, unsigned int payloadLength) {Serial.print("callback
  invoked for topic: "); Serial.println(topic);
  for (int i = 0; i < payloadLength; i++) {
    / Serial.println((char)payload[i]);
    command += (char)payload[i];
  }
Serial.println(command);
command1=getValue(command,',',0);
command2=getValue(command,',',1);
if(command1=="2"){
display.clearDisplay();
/ display a line of text
  display.setTextSize(1);
  display.setTextColor(WHITE);
```

```
  display.setCursor(0,10);
  display.print(command2);
  /  update display with all of the above graphicsdisplay.display();
}
command ="";
command1 ="";
command2="";
}
String getValue(String data, char separator, int index)
{
    int found = 0;
    int strIndex[] = { 0, -1 };
    int maxIndex = data.length() - 1;
    for (int i = 0; i <= maxIndex && found <= index; i++) {if
        (data.charAt(i) == separator || i == maxIndex) {
            found++;
            strIndex[0] = strIndex[1] + 1; strIndex[1]
            = (i == maxIndex) ? i+1 : i;
        }
    }
    return found > index ? data.substring(strIndex[0], strIndex[1]) : "";
}
```

8.3 Feature 3 :

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h> const
char* ssid = "SB-IOT1";
const char* password = "sb@iot11";String
command1,command2; #define ORG
"bhip5y"
#define DEVICE_TYPE "Vamsi"
#define DEVICE_ID "8500"
#define TOKEN "8500913778"
String command;
char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; char
topic[] = "iot-2/cmd/home/fmt/String";
char authMethod[] = "use-token-auth";char
token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
/ Serial.println(clientID);#include
<Wire.h>
#include <Adafruit_SSD1306.h>
#include <Adafruit_GFX.h> #define
SSD1306_LCDHEIGHT 64
/ OLED display TWI address
#define OLED_ADDR 0x3C
Adafruit_SSD1306   display(-1); #if
(SSD1306_LCDHEIGHT != 64)
#error("Height incorrect, please fix Adafruit_SSD1306.h!");
```

```
#endif


void callback(char* topic, byte* payload, unsigned int payloadLength); WiFiClient
wifiClient;
PubSubClient client(server, 1883, callback, wifiClient); void
setup() {
   display.begin(SSD1306_SWITCHCAPVCC, OLED_ADDR);
  Serial.begin(115200);
  Serial.println();
  pinMode(D1,OUTPUT);
  wifiConnect();
  mqttConnect();
}
void loop() {
  if (!client.loop()) {
    mqttConnect();
  }
delay(100);
}
void wifiConnect() {
  Serial.print("Connecting to "); Serial.print(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {delay(500);
    Serial.print(".");
  }
  Serial.print("nWiFi connected, IP address: "); Serial.println(WiFi.localIP());
}
```

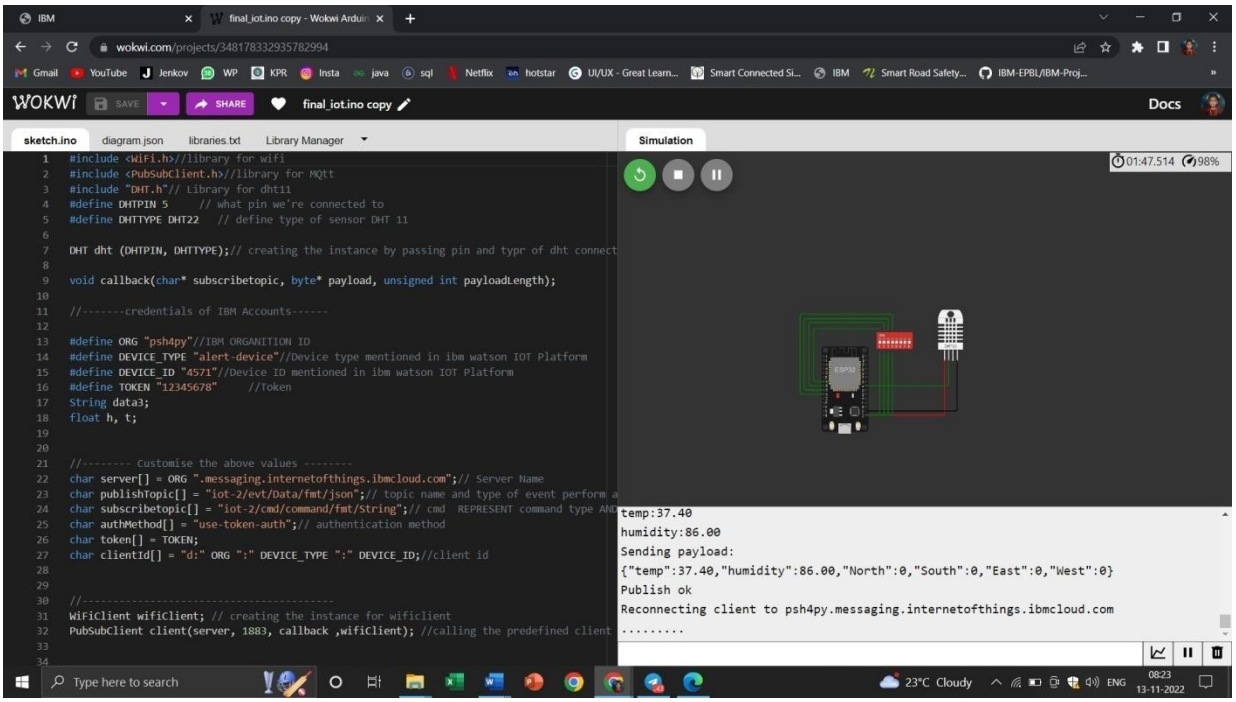
```
void mqttConnect() {
  if (!client.connected()) {
    Serial.print("Reconnecting MQTT client to "); Serial.println(server); while
    (!client.connect(clientId, authMethod, token)) { Serial.print(".");
      delay(500);
    }
    initManagedDevice();
    Serial.println();
  }
}
void initManagedDevice() { if
  (client.subscribe(topic)) {
    Serial.println("subscribe to cmd OK");
  } else {
    Serial.println("subscribe to cmd FAILED");
  }
}
void callback(char* topic, byte* payload, unsigned int payloadLength) {Serial.print("callback
  invoked for topic: "); Serial.println(topic);
  for (int i = 0; i < payloadLength; i++) {
    / Serial.println((char)payload[i]);
    command += (char)payload[i];
  }
Serial.println(command);
command1=getValue(command,',',0);
command2=getValue(command,',',1);
if(command1=="3"){
display.clearDisplay();
```

```
/ display a line of text
  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(0,10);
  display.print(command2);
  /  update display with all of the above graphicsdisplay.display();
}
command ="";
command1 ="";
command2="";
}
String getValue(String data, char separator, int index)
{
    int found = 0;
    int strIndex[] = { 0, -1 };
    int maxIndex = data.length() - 1;
    for (int i = 0; i <= maxIndex && found <= index; i++) {if
        (data.charAt(i) == separator || i == maxIndex) {
            found++;
            strIndex[0] = strIndex[1] + 1; strIndex[1]
            = (i == maxIndex) ? i+1 : i;
        }
    }
    return found > index ? data.substring(strIndex[0], strIndex[1]) : "";
```

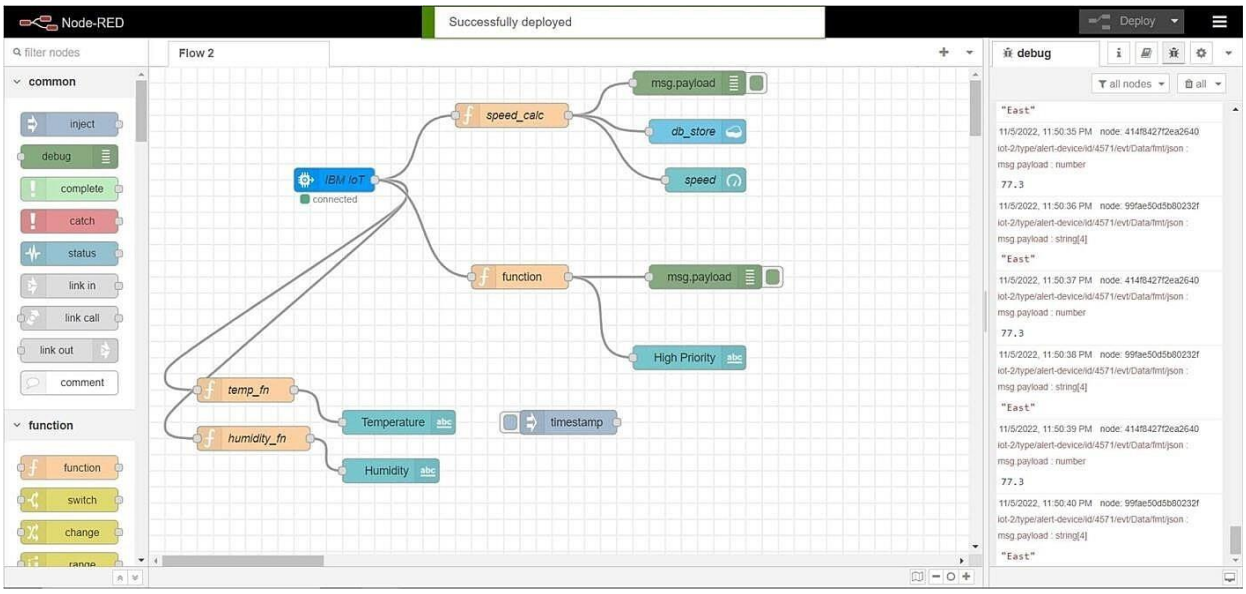
9. TESTING :

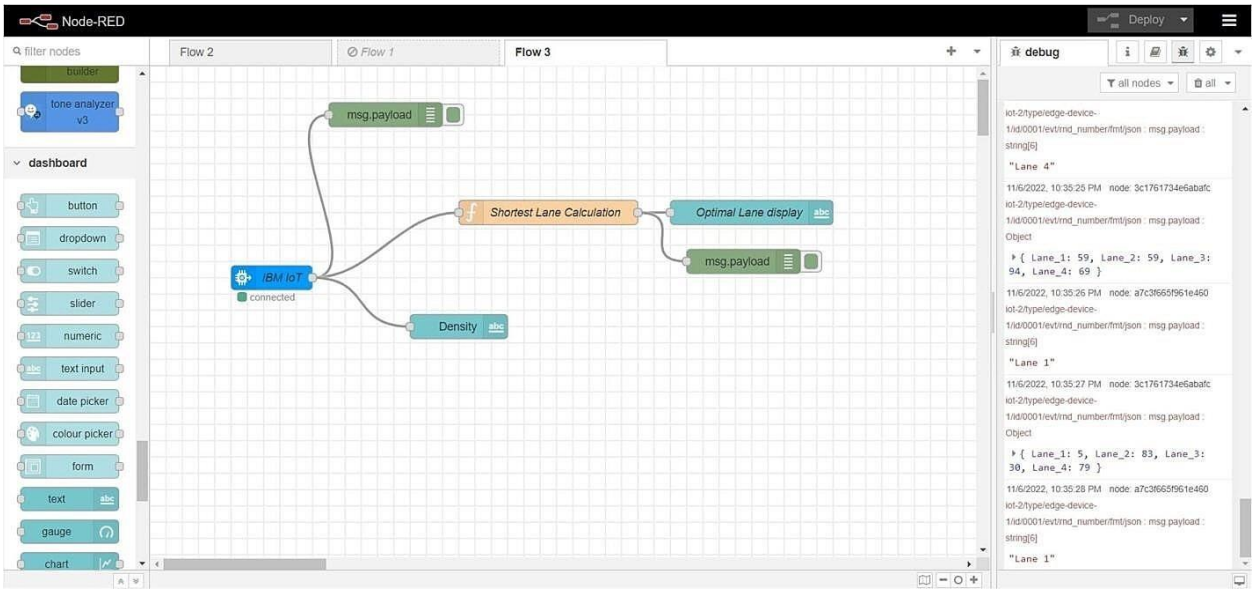
9.1 Test cases :

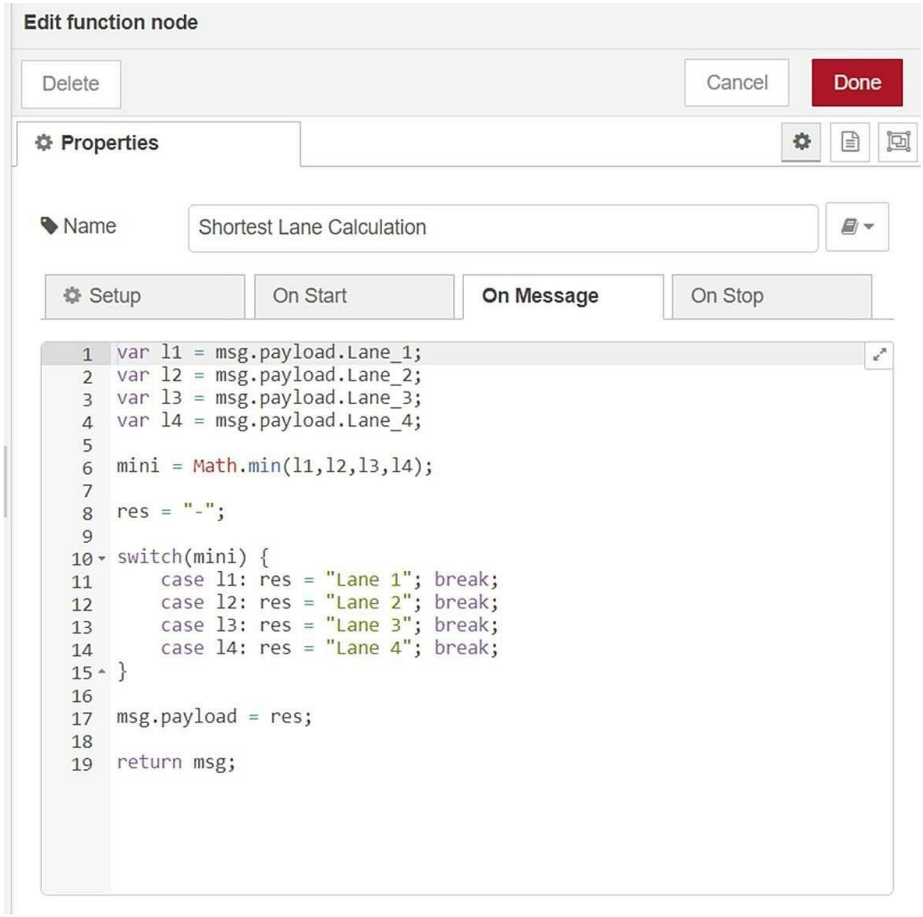
Wokwi Simulation : https://wokwi.com/projects/348178332935782994



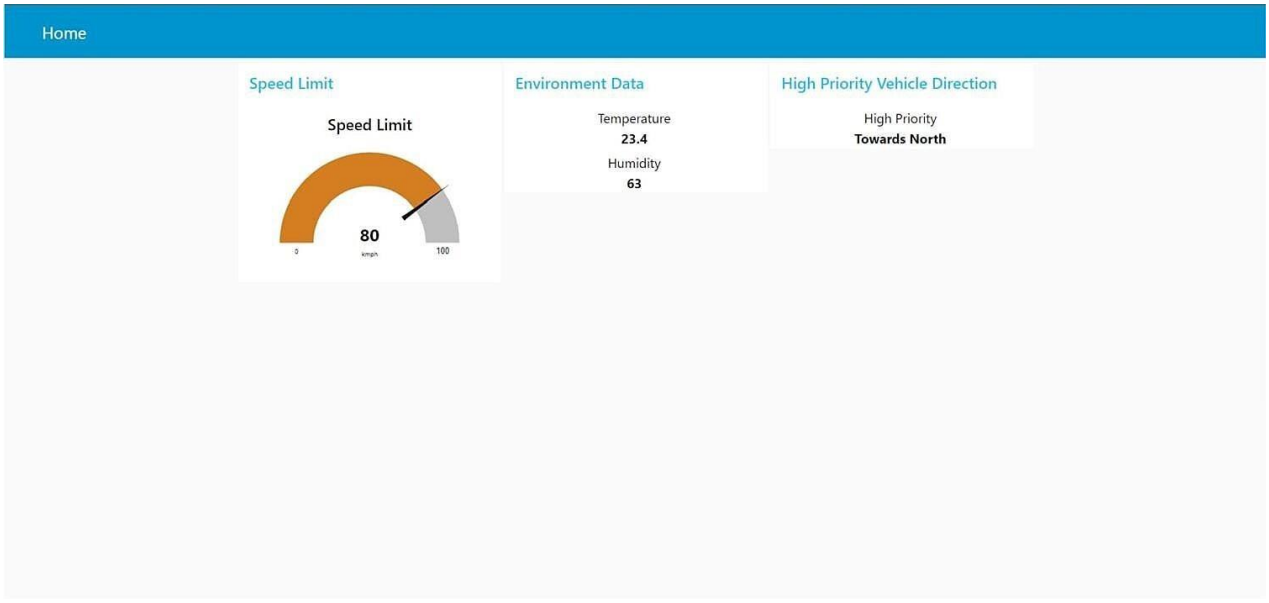
Node Red :

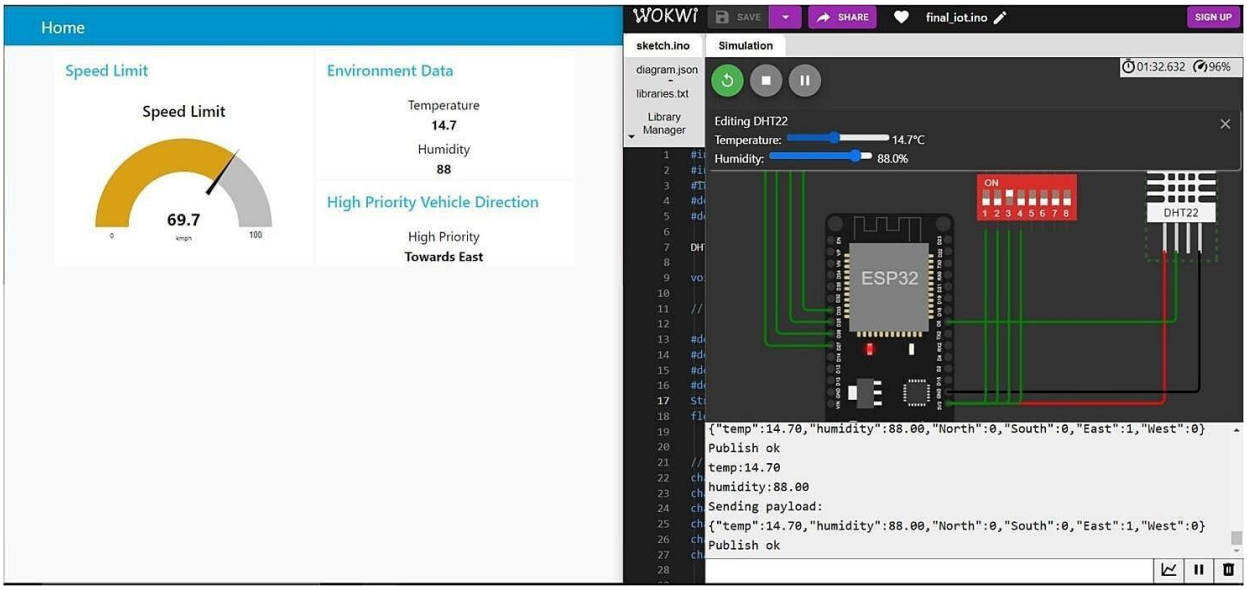



Node-RED
Deploy
filter nodes
Flow 2
Flow 1
Flow 3
debug
all nodes
builder
tone analyzer v3
dashboard
button
dropdown
switch
slider
numeric
text input
date picker
colour picker
form
text
gauge
chart
msg.payload
Shortest Lane Calculation
Optimal Lane display
msg.payload
IBM IoT
connected
Density
iot-2/type/edge-device-1/id/0001/evt/rnd_number/fmt/json : msg.payload : string[6]
"Lane 4"
11/6/2022, 10:35:25 PM node: 3c1761734e6abafc
iot-2/type/edge-device-1/id/0001/evt/rnd_number/fmt/json : msg.payload : Object
{ Lane_1: 59, Lane_2: 59, Lane_3: 94, Lane_4: 69 }
11/6/2022, 10:35:26 PM node: a7c3f665f961e460
iot-2/type/edge-device-1/id/0001/evt/rnd_number/fmt/json : msg.payload : string[6]
"Lane 1"
11/6/2022, 10:35:27 PM node: 3c1761734e6abafc
iot-2/type/edge-device-1/id/0001/evt/rnd_number/fmt/json : msg.payload : Object
{ Lane_1: 5, Lane_2: 83, Lane_3: 30, Lane_4: 79 }
11/6/2022, 10:35:28 PM node: a7c3f665f961e460
iot-2/type/edge-device-1/id/0001/evt/rnd_number/fmt/json : msg.payload : string[6]
"Lane 1"

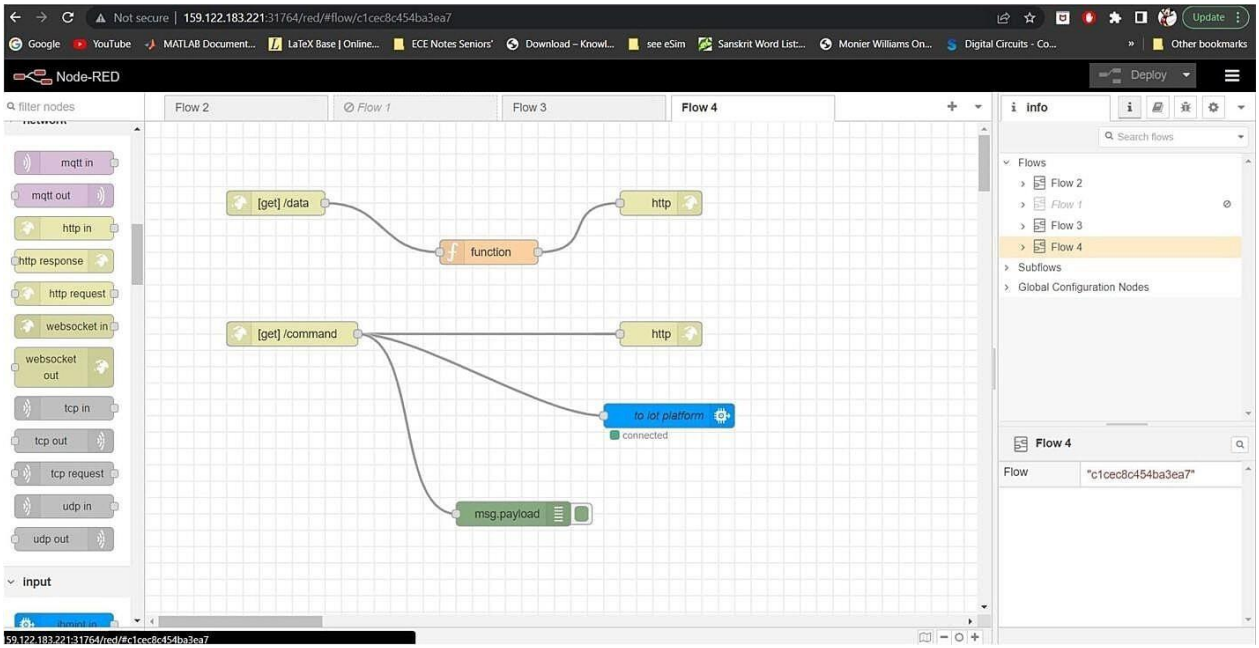

Edit function node
Delete
Cancel
Done
Properties
Name
Shortest Lane Calculation
Setup
On Start
On Message
On Stop
var l1 = msg.payload.Lane_1;
var l2 = msg.payload.Lane_2;
var l3 = msg.payload.Lane_3;
var l4 = msg.payload.Lane_4;
mini = Math.min(l1,l2,l3,l4);
res = "-";
switch(mini) {
case l1: res = "Lane 1"; break;
case l2: res = "Lane 2"; break;
case l3: res = "Lane 3"; break;
case l4: res = "Lane 4"; break;
}
msg.payload = res;
return msg;

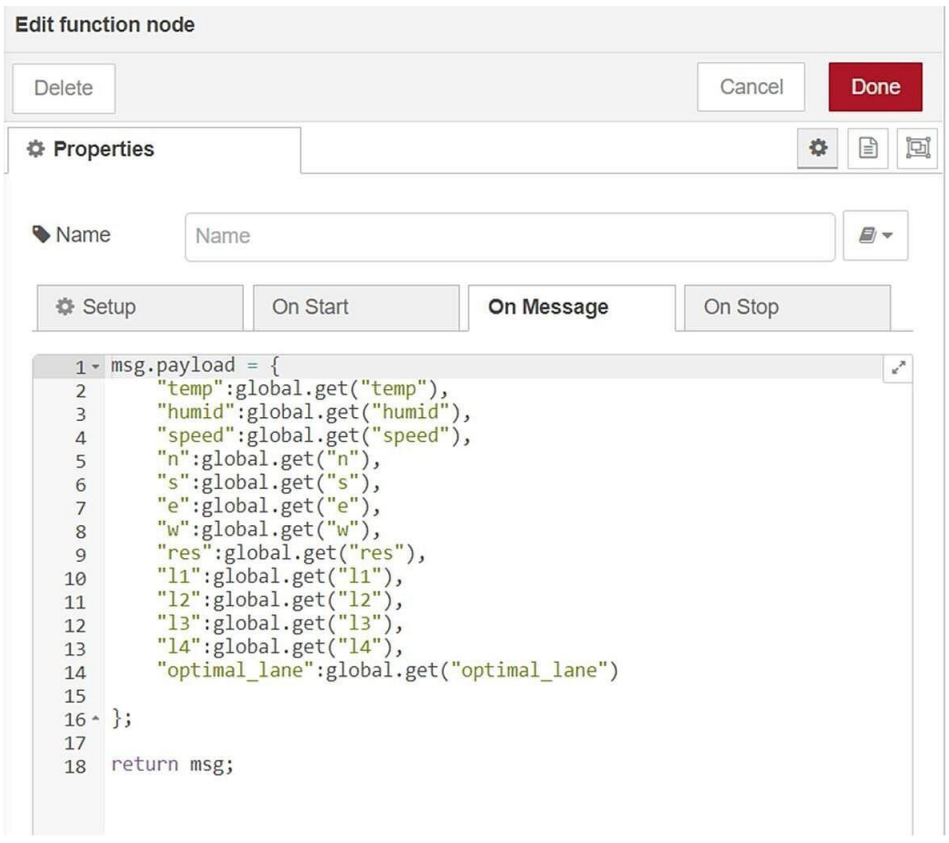
Node Red Web UI :



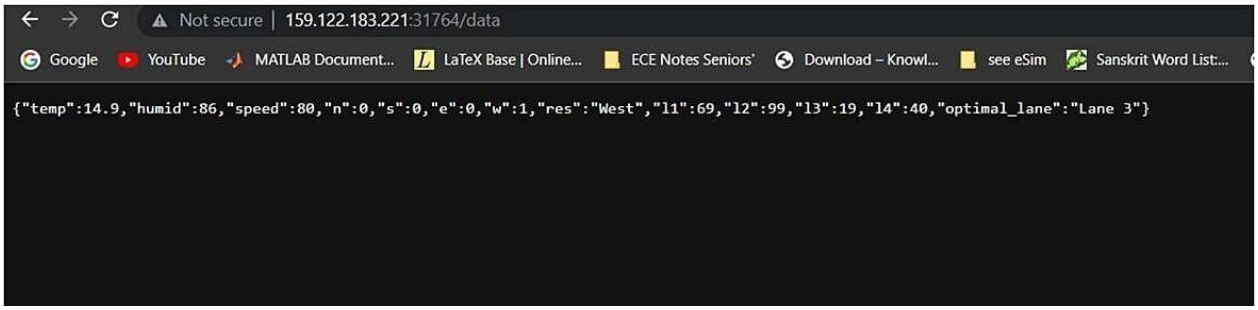


Node Red - Connect with MIT APP Inventor :

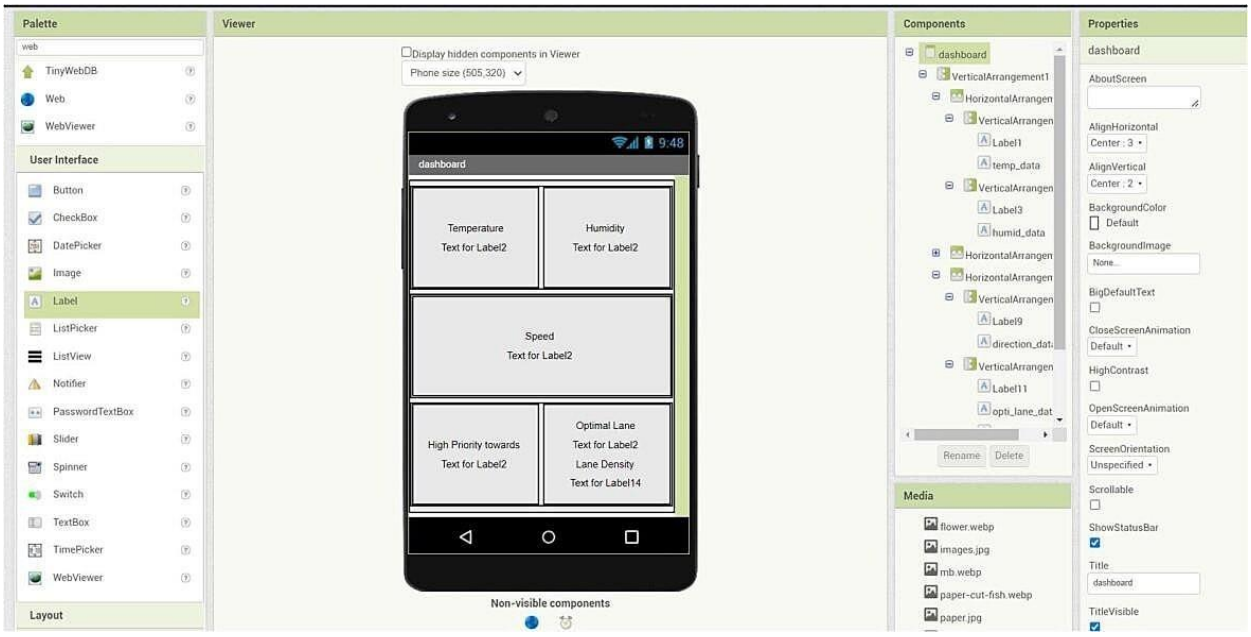




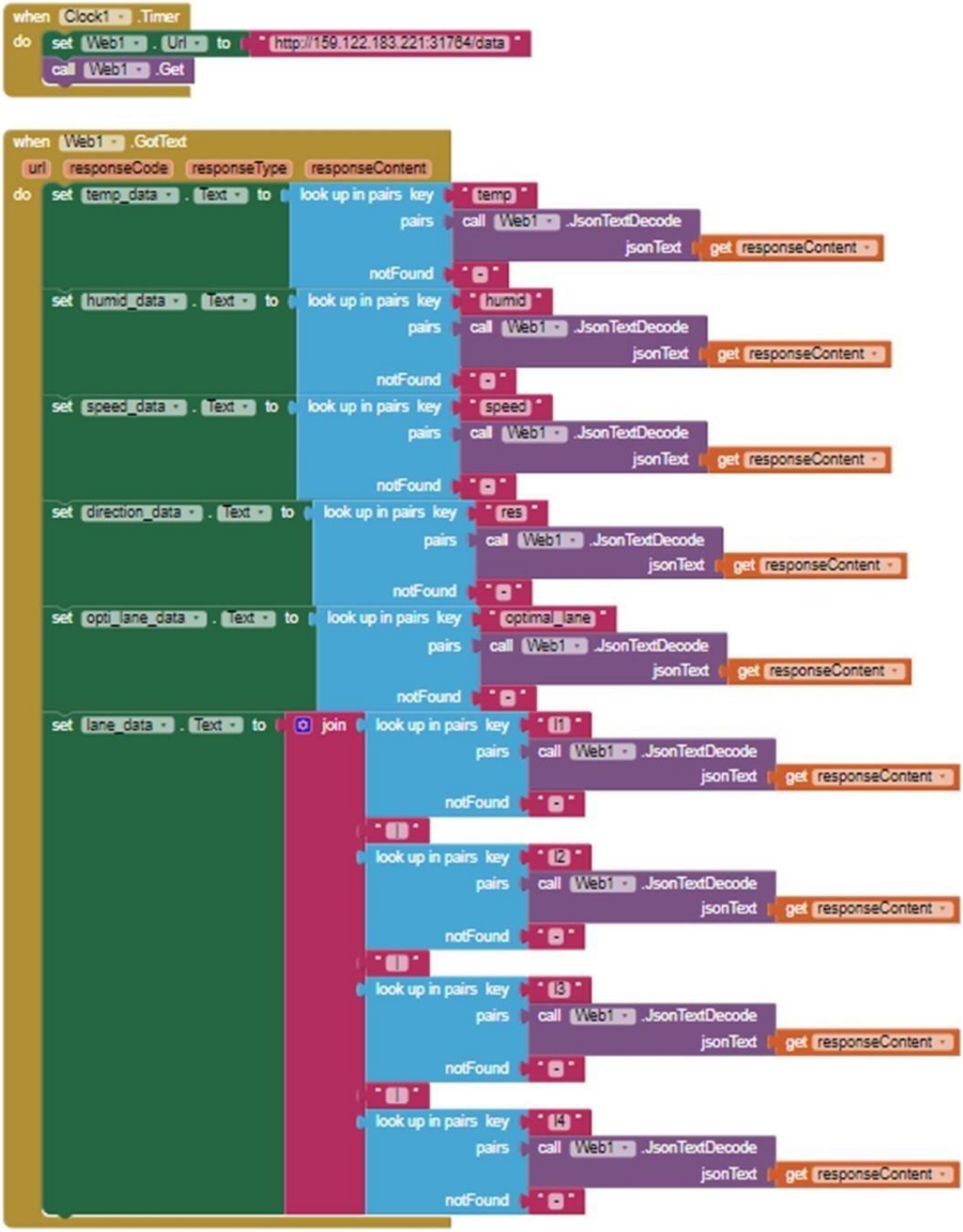
Output from Node red :



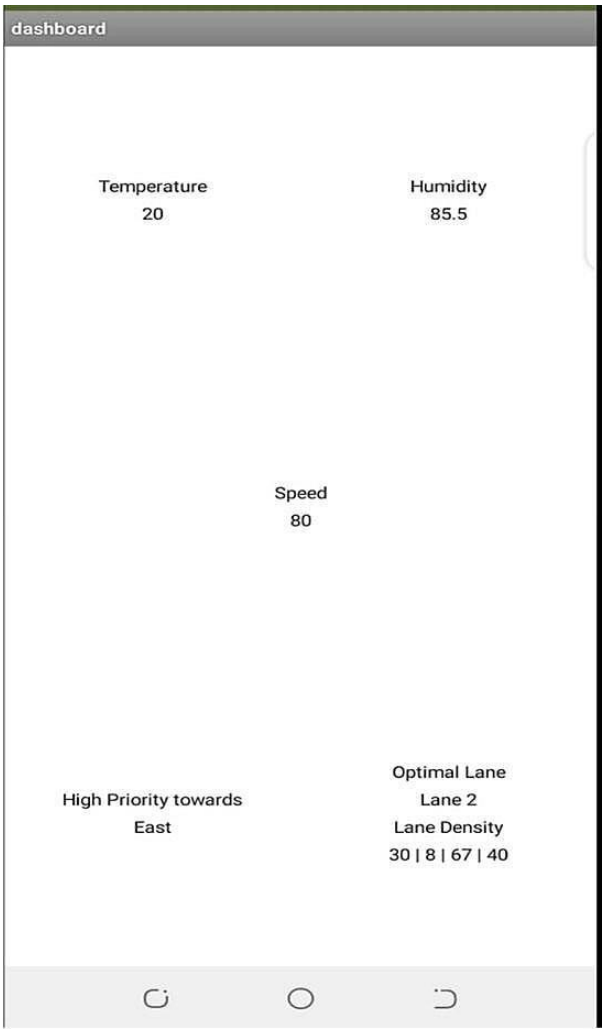
MIT App Inventor UI design:



MIT App Inventor Backend design:

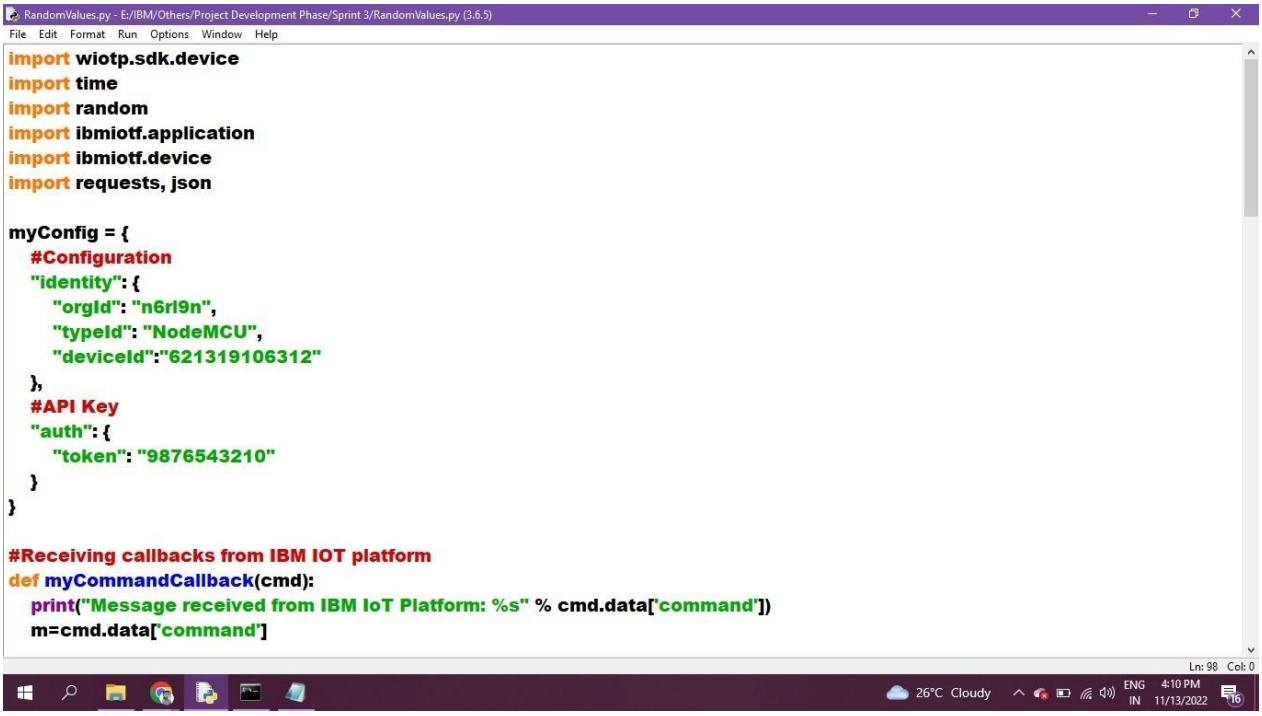


(OUTPUT) Display from MIT App:

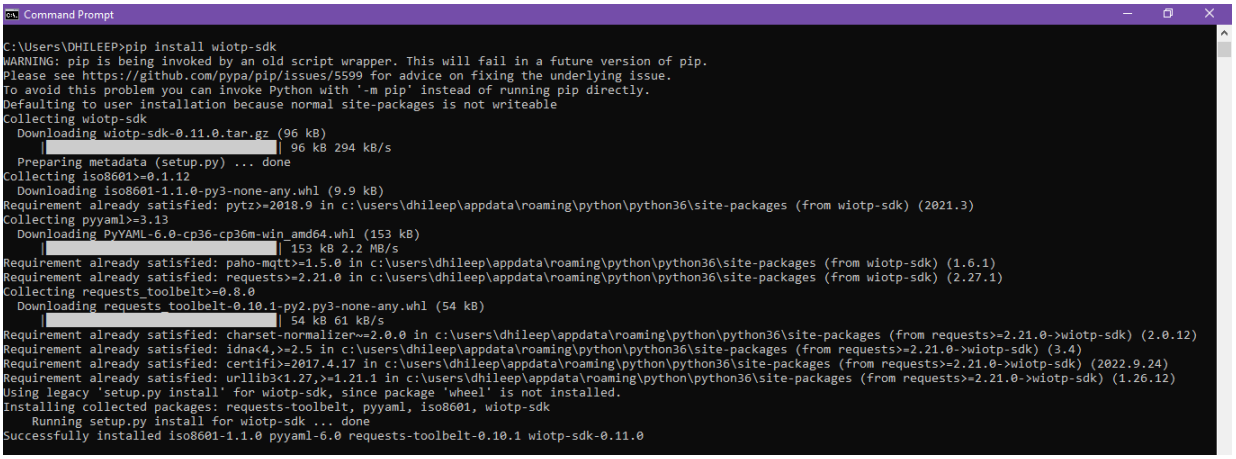


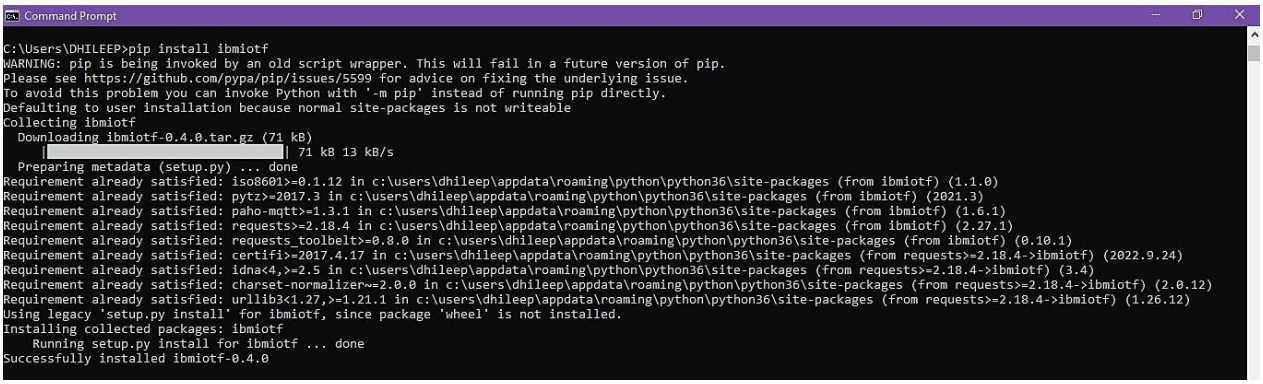
User Acceptance Testing :

Python Simulation :

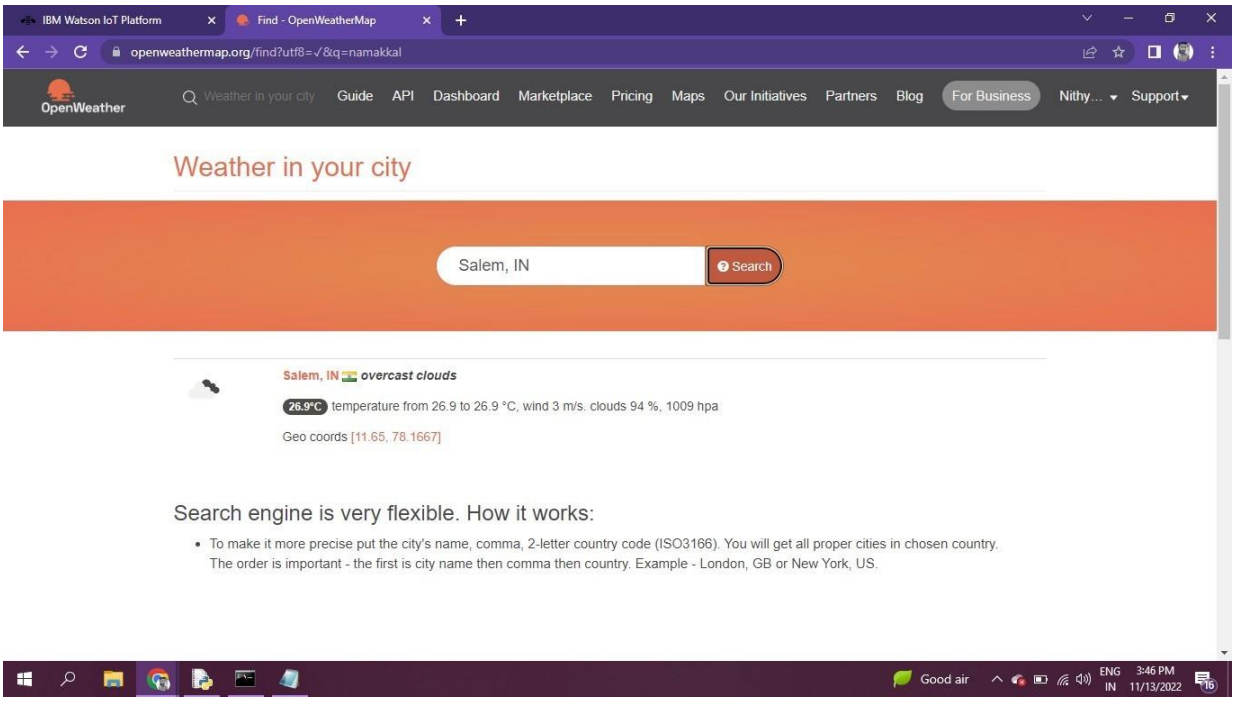


Import wiotp-sdk & ibmiotf :

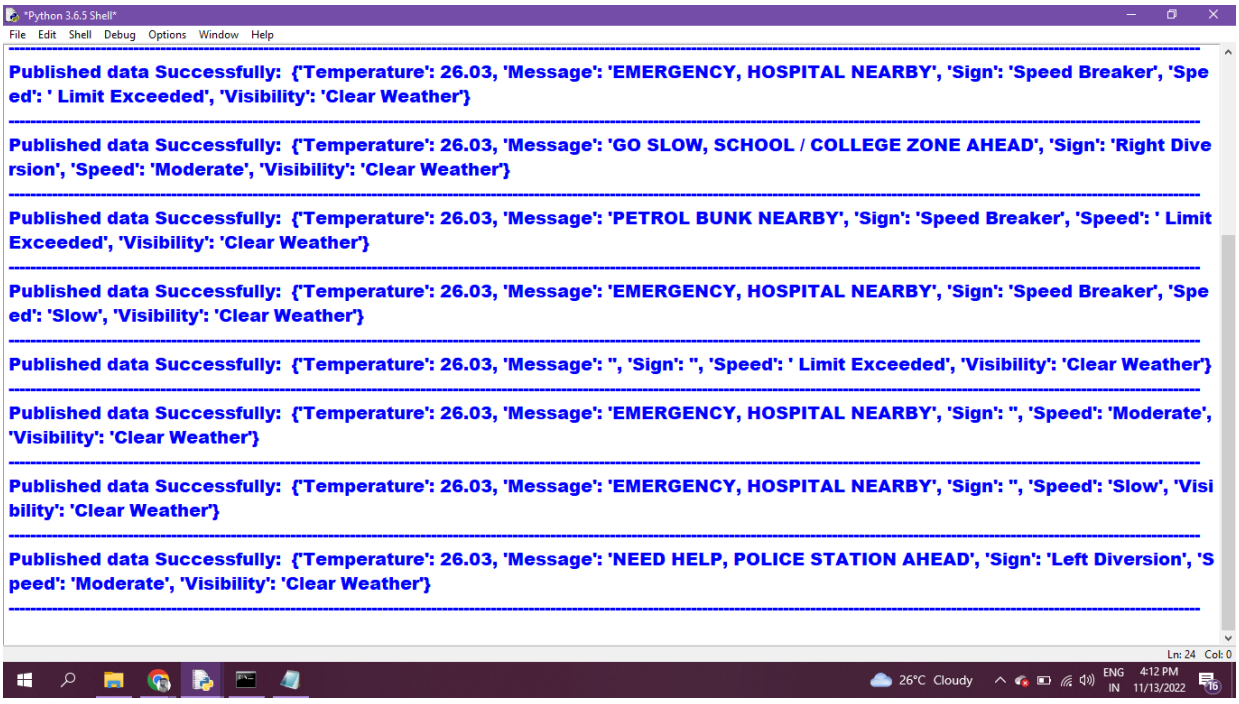




OpenWeatherMap - (Ex., Salem, IN) :



Python IDLE Output :



10. RESULT:

10.1 Performance Metrics:

The result shows three switches through which you can switch the display to different modes.Mode1 : Displaying Speed Limit
Mode2 : Display of Diversions, Alerts of Accident prone areaMode3 : Information sign boards

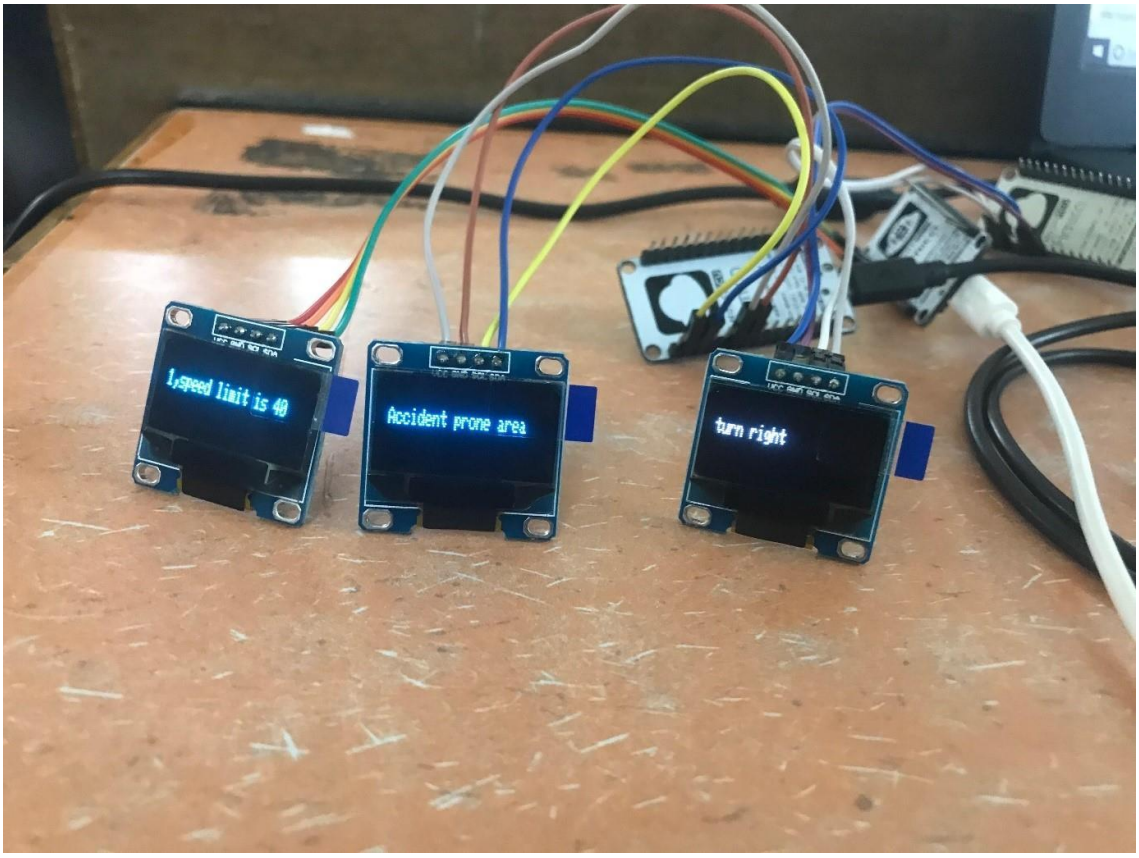


Fig 10 - This figure shows three OLEDs that we are displaying.

(1) The first OLED display shows speed limit using weather API.

(2) The second OLED display shows about the alerts of the accident prone area.(3)The third OLED display shows the information sign boards

11. ADVANTAGES & DISADVANTAGES:

Advantages :

- Efficient Traffic Management
- Automated Toll and Ticketing
- Self-driving Cars
- Advanced Vehicle Tracking or Transportation Monitoring
- Enhanced Security of the Public Transport

Disadvantages :

- Property Damage
- Bodily Injury
- Cyber Risk

12. CONCLUSION:

Roads were previously only functional in nature. Highways are now built to be safe, long- lasting, and easily accessible. The thought of a roadway being a vector for IOT networks or any other communication system was unthinkable and impractical. However, recent advances, such as the installation of digital sign boards along roadside, have provided a gateway that allows highways to function as data conveyors. Data such as road conditions and traffic patterns arenow shown on sign boards. Wireless networks can use sensor technology to enable more detailed communications at higher levels. IoT systems could be used by state and local transportation departments to target road maintenance needs, traffic utilisation, weather conditions, and accident records.

13. FUTURE SCOPE:

1. **Solar powered roadways**

Photovoltaic cells are embedded within hexagonal panels made of tempered glass, which are used to pave roads. These panels contain LEDs, microprocessors, snow-melting heating devices and inductive charging capability for electric vehicles when driving. Glass is renewable and can be engineered to be stronger than steel, and to allow cars to stop safely even when traveling athigh speeds. While this idea has gained widespread support, scalability is a challenge as it remains expensive.

2. **Smart Roads**

Specially engineered roadways fitted with smart features, including sensors that monitor and report changing road conditions, and WiFi transmitters that provide broadband services to vehicles, homes and businesses. The smart road can also charge electric cars as they drive.

3. **Glow in the dark roads**

Glowing markers painted onto existing roadway surfaces use a photo-luminescent powder thatabsorbs and stores daylight. The 500m long strips glow for 8 hours after dark. This technologyis still in the testing phase, and the glow is not yet consistent, but it could be more cost-effective than traditional road lighting technologies.

4. **Interactive lights**

Road lights activated by motion sensors to illuminate a particular section of the road as cars approach. The lights dim once the car passes. Suited for roads with less traffic, interactive lights provide night visibility as needed and reduce energy wastage when there are no cars. One design, developed in Holland, uses the wind generated by passing vehicles to power lights.

5. **Electric priority lane for charging electric vehicles**

Embedded cables generate magnetic fields that charge electric vehicles while driving. A receiver coil in the vehicle picks up electromagnetic oscillations from a transmitter coil embedded in the

road and converts them to AC, which can then power the car. Inductive charging technology already exists for static cars, but future wireless technology could charge batteries while in motion, providing distance range solutions for electric vehicles which travel longer journeys.

6. Weather detection

Networks of AI-integrated sensors detect weather conditions that impact road safety. Road Weather Information Systems (RWIS) in use today are limited because they only collect data from a small set of weather stations. A larger future network could use automated weather stations to collect atmospheric and weather data and instantly upload it to the cloud. Dynamic temperature-sensitive paint could be used to highlight invisible roadway conditions like blackice.

7. Traffic detection

Data that helps travelers plan their routes. Sensors lining highways monitor traffic flow and weight load, warn drivers of traffic jams, and automatically alert the authorities about accidents. Fiber-optic cables embedded in the road detect wear and tear, and communication between vehicles and roads can improve traffic management. For example, rapid flow technologies useartificial intelligence (AI) to manage traffic lights, which respond to each other and to cars.

Traditional systems were pre-programmed to optimize flow around peak journey times, new technologies are able to process and optimize flows in real time.

14. APPENDIX :

Source code :

Code to print the random temperature, Road signs, Speed limit, Message:(

RandomValues.py)

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

 myConfig = {
          #Configuration
          "identity": {

              "orgId": "n6rl9n",
              "typeId": "NodeMCU",
              "deviceId":"621319106312"
},
#API Key
"auth": {
              "token":  "9876543210"
          }
}

#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 =
"Salem, IN"
URL = BASE_URL + "q=" + CITY + "&units=metric"+"&appid=" +
"f58e4720c739a54c439aba9b05176839"

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.randint(0,5)if
msg==1:
      message="GO SLOW, SCHOOL ZONE AHEAD"
elif msg==2:
      message="NEED HELP, POLICE STATION AHEAD"
elif msg==3:
      message="EMERGENCY, HOSPITAL NEARBY"
elif msg==4:
      message="DINE IN, RESTAURENT AVAILABLE"
elif msg==5:
      message="PETROL BUNK NEARBY"
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" #Diversion
part
```

```
sign=random.randint(0,5) if
sign==1:
      signMsg="Right Diversion"
elif sign==2:
      signMsg="Speed Breaker"elif
sign==3:
      signMsg="Left Diversion"elif
sign==4:
      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: ", myData)
    client.commandCallback = myCommandCallback time.sleep(5)
client.disconnect()
```

Github link :

https://github.com/IBM-EPBL/IBM-Project-32650-1660211160