IBM NALAIYA THIRAN 2022-23 PROJECT REPORT

SIGNS WITH SMART CONNECTIVITY FOR BETTER ROAD SAFETY

TEAM ID - PNT2022TMID14540

1. INTRODUCTION

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 displayed.
- Messages indicating school ,hospital, police station zones are also displayed.

2. LITERATURE SURVEY

2.1 Existing problem

a) **Digital Notice Board Based on IOT**

This project presents a digital notice board using IoT module. The idea behind this project is to

provide its users with a simple, fast and reliable way to put up important notices in an LED where the user can send a message to be displayed in the LED. The message can be sent through an android application designed in this project, through the IoT module. So, notices can be put up in an LED display from any location in the world. It uses a microcontroller for system control, IoT based technology for communication and sends the message through the android

application. The project consists of Arduino UNO board, IoT module, an LED, and an android application for user interface with the hardware. This device can be used anywhere irrespective of the place of deployment provided mobile network connectivity is available. This is a project that displays messages that the user desires, on an LED Display Matrix. The Display consists of 256 LED lights, sequentially arranged in 8 rows and 32 columns (8*32). Apart from the display, the project consists of a Node MCU controller which helps the system to connect to the Wi-Fi. This system makes use of Google Assistant to accept speech inputs from user, through user's Android smartphone. User needs to login into their Google account. A USB cable acts as the power cable for the system. The speech input is converted into a text display in an alphanumeric format which is predefined. The displayed message will either scroll or remain static, based on the size of display and length of message. This project can widely use in offices, schools, educational institutions as well as government and corporate offices to display important notices and messages. This can prove to help users save a lot of time as against the use of traditional pin and paper notice display.

b) **Internet of Things Based Notifications Using Smart Notice Board**

Conventional Notice Board employs manual display and monitoring with papers and ledgers. The Target users are unaware of information displayed on the notice board. The objective of the project is to display the message on the notice board from anywhere and anytime, that even provides broadcast alerts to the target users. The system was designed and developed using the Internet of Things. Arduino board integrates the display unit, Mobile App and SMS Agent through Internet. The message to be displayed on the notice board is sent through a mobile app to the board with Arduino. As soon as the message is displayed, SMS alert is sent to the target users. A system of efficient Notice Board display controlled through the Internet is accomplished and presented in this paper.

c) **An IoT based Smart Monitoring System for Vehicles**

There is increased adoption of penalty and fine for traffic rule violators in the public sector but there is a tendency for people to evade from those imposed fines and restrictions for their own safety. Our system will completely monitor all the traffic violations namely over speeding, rash driving, drunken driving, driving without a seat belt, and so on right from the starting of the car. There is an increasing demand to develop a system to check passengers without coming out of the vehicle. A new system for the police force to check the vehicle's details with a smart device placed in the vehicle. The device is equipped with speed monitoring, Alcohol detection, Seat belt checking, etc. If any violation is detected the controller sends an emergency data to the cloud, thus the vehicle is in continuous monitoring mode, and RTO will get updates about the vehicles which are violating rules. Alcoholic breath sensor will continuously monitor the driver's breath, speed sensor will be connected with the speedometer and checks for over speeding, Seat belt sensor will warn the driver if he/she is not using the seat belt, vehicle details including license, pollution details, insurance, etc. will be uploaded to the server or cloud. If any of the above things are violated, automatically defaulter will be imposed fines and the details will be sent to the Motor vehicle department.

d)**Congestion Adaptive Traffic Light Control and Notification Architecture Using Google Maps APIs :**

Controlling of traffic signals optimally helps in avoiding traffic jams as vehicle volume density changes on temporally short and spatially small scales. Nowadays, due to embedded system

development with the rising standards of computational technology, condense electronics boards as well as software packages, system can be developed for controlling cycle time in real time. At present, the traffic control systems in India lack intelligence and act as an open-loop control system, with no feedback or sensing network, due to the high costs involved. This paper aims to improve the traffic control system by integrating different technologies to provide intelligent feedback to the existing network with congestion status adapting to the changing traffic density patterns. The system presented in this paper aims to sense real-time traffic congestion around the traffic light using Google API crowdsource data and hence avoids infrastructure cost of sensors. Subsequently, it manipulates the signal timing by triggering and conveying information to the timer control system. Generic information processing and communication hardware system designed in this paper has been tested and found to be functional for a pilot run in real time. Both simulation and hardware trials show the transmission of required information with an average time delay of 1.2 seconds that is comparatively very small considering cycle time.
Mishra, Sumit Kumar, Devanjan Bhattacharya and Ankit K. Gupta. “Congestion Adaptive Traffic Light Control and Notification Architecture Using Google Maps APIs.” Data 3 (2018): 67.

e) **An IoT based Weather Information Prototype Using WeMos :**
The Internet of Things (IOT) describes the interconnection of devices and people through the traditional internet and social networks for various day-to-day applications like weather monitoring, healthcare systems, smart cities, irrigation field, and smart lifestyle. IOT is the new revolution of today’s internet world which monitors live streaming of the entire world’s status like temperature, humidity, thunderstorm, earthquake, floods etc. that can stagger an alarm to human life. This paper proposes a low-cost weather monitoring system which retrieves the weather condition of any location from the cloud database management system and shows the output on an OLED display. The proposed system uses an ESP8266-EX microcontroller based Wemos D1 board and it is implemented on Arduino platform which is used to retrieve the data from the cloud. The main objective of this paper is to view weather conditions of any location and allows to access the current data of any station.

R. K. Kodali and A. Sahu, "An IoT based weather information prototype using WeMos," 2016 2nd International Conference on Contemporary Computing and Informatics (IC3I), 2016, pp. 612-616, doi: 10.1109/IC3I.2016.7918036.

f) **IOT Based Weather Monitoring and Reporting System Project**
The IOT based Weather Monitoring and Reporting System project is used to get Live reporting of weather conditions. It will Monitor temperature, humidity, moisture and rain level. Suppose Scientists/nature analysts want to monitor changes in a particular environment like volcano or a rain-forest. And these people are from different places in the world. In this case, SMS based weather
monitoring system has some limitations. Since it sends SMS to few numbers. And time for sending SMS increases as the number of mobile numbers increases. In order to know the information about weather of a perticular place then they have to visit that particular sites. Where everyone can see it.

Anita M. Bhagat ,Ashwini G. Thakare ,Kajal A. Molke , Neha S. Muneshwar ,Prof. V. Choudhary IOT Based Weather Monitoring and Reporting System ,2019 .

g) **Incorporating Weather Updates for Public Transportation Users of Recommendation Systems:**

This work presents a system for augmenting the functionality of Yelp-like recommendation sites by enabling users to search for places bounded by travel-time when using public transportation, and modifying recommendations based on updated weather conditions. Using public transport, although is cheaper and efficient, entails that only fixed places of boarding/exiting may be used which, in turn, implies walking to (from) a particular location from (to) a given station. Given the impact of the weather on the mood and activities, preferences for a certain type of services may need to be dynamically adjusted based on the current weather or the near-future forecast, modulo travel-routes
to preferred locations. In this work, we develop a model to predict a user's preferred mode of transport (car, or public transit) from their old check-ins and incorporate the weather context into
the recommendation process. We use event-based modeling to control the extent of walking depending on user-defined tolerance information and live weather conditions. We implemented a web application (both desktop and mobile platforms), utilizing existing tools such as Google Maps Direction API and OpenWeatherMap API for retrieving real-time information.

h) **System Natural Data is now becoming more valuable in a day to get real-time data for natural data:**

Physical monitoring of the environment allows for the identification of areas suitable for agriculture, industry, and other purposes. In this article, the Arduino-UNO microcontroller-based board is used for the data acquisition strategy and the use of analog and digital sensors. Temperature, humidity, light intensity and gas concentrations can be monitored in real-time [4] [9-12] [13-16].

i) **The impact of daily weather on daily travel trips :**

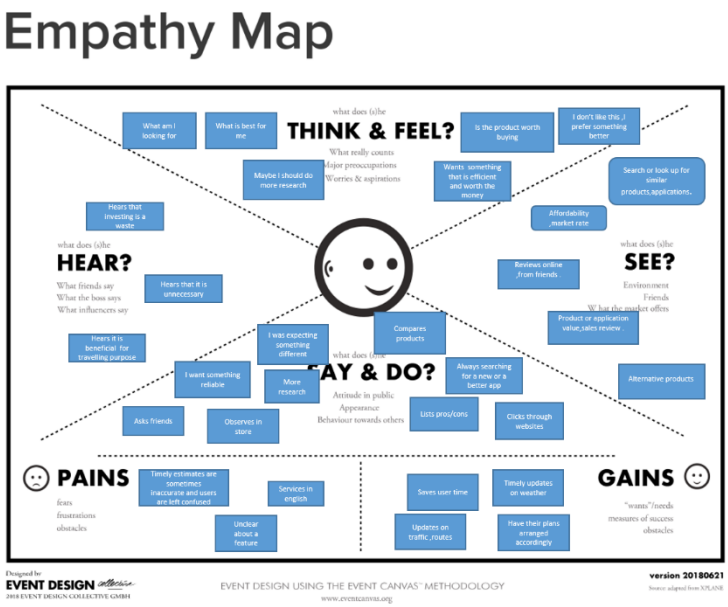
It is of increasing social interest - climate change and increasing scarcity, understanding the climatic implications of travel behavior, especially walking and biking. Recently, various courses are travel, health, and biometeorology.

2.2 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

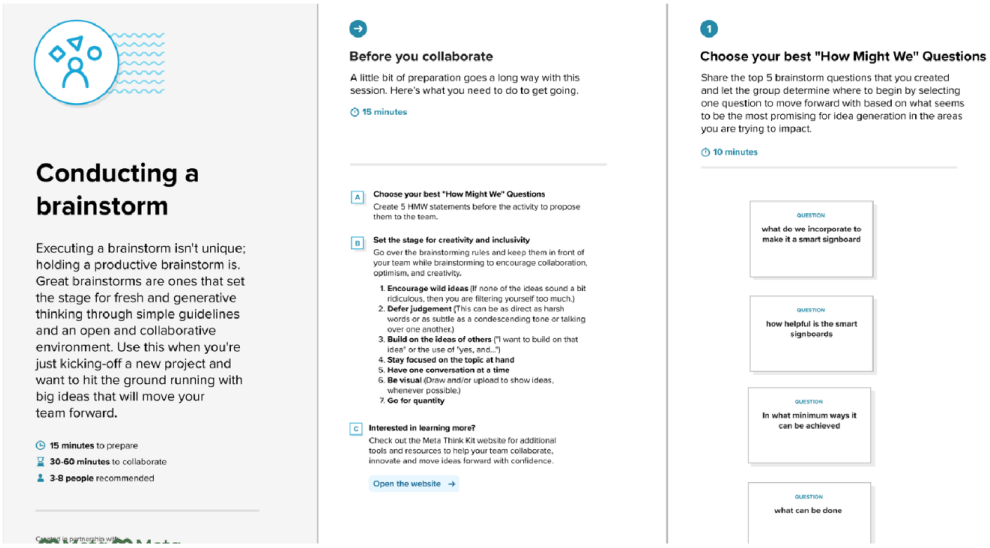


https://github.com/IBM-EPBL/IBM-Project-10904-1659243422/blob/main/Project%20Design%20and%20Planning/Ideation%20phase/Team%20Member%201-Sulana%20S%20Chandarshekar/Empathy%20Map%20Canvas%20ibm.pdf

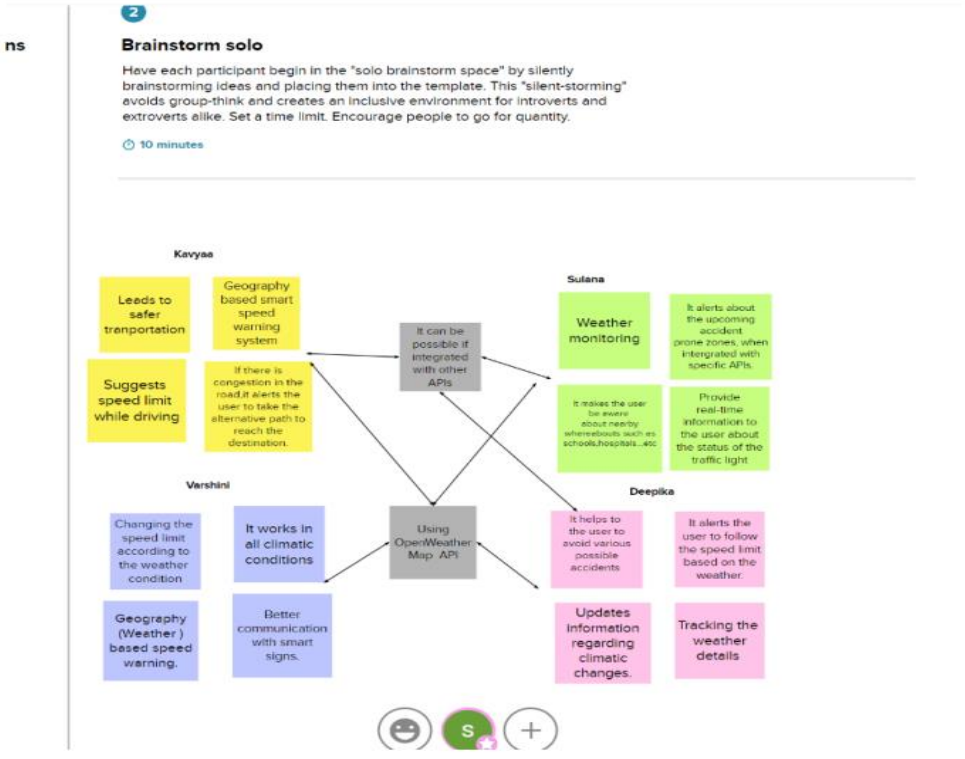
3.2 Ideation & Brainstorming

https://github.com/IBM-EPBL/IBM-Project-10904-1659243422/blob/main/Project%20Design%20and%20Planning/Ideation%20phase/Team%20Member%201-Sulana%20S%20Chandarshekar/Brainstorming%20(2).pdf

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



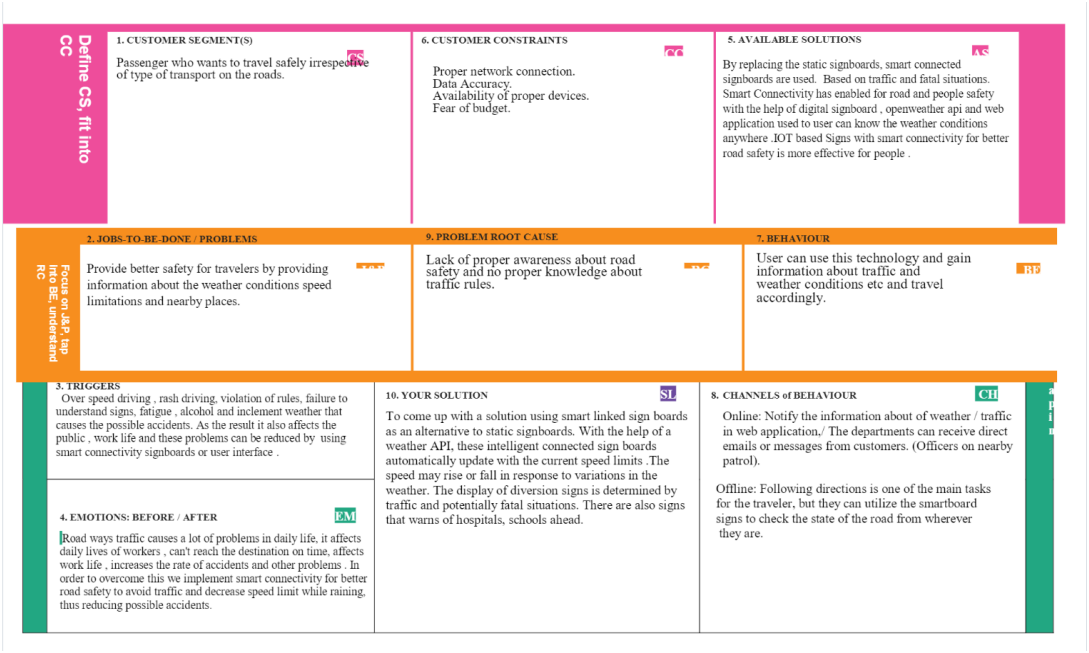
Step-2: Brainstorm, Idea Listing and Grouping



3.3 Proposed Solution

S. N0	PARAMETER	DESCRIPTION
1.	Problem Statement (Problem to be solved)	To avoid road accidents caused by collisions, over speeding of vehicles due to adverse weather conditions.
2.	Idea / Solution description	The project approach focuses on to digitalizing the already existing static signboards to smart signboards using a interface where people are able to see about whether the indications and such information. this information can be accessed from open weather map and we can display the updates on the user interface on a timely basis. The smart display gets the speed limitations from a web app using weather API and update periodically. based on the weather changes, the user is alerted on the suggested speed limits.
3.	Novelty / Uniqueness	Sign boards are converted to digital display where APIs and online services are integrated in new and interesting ways. Open Weather Map is an online service that provides global weather data, forecasts and historical weather data for any geographical location.
4.	Social Impact / Customer Satisfaction	Suggests the speed limits for the user to follow based on the weather which can be helpful in reducing any possible accident.
5.	Business Model (Revenue Model)	It doesn't cost much for implementation (unless it is for use in a largescale because we obtain the required information to display and integrate it with an interface which can be achieved by programming accordingly with the help of IOT.
6.	Scalability of the Solution	This project is highly feasible and can later on be further updated with other additional features as well.

3.4 Problem Solution fit



4. REQUIREMENT ANALYSIS

4.1 Functional requirement

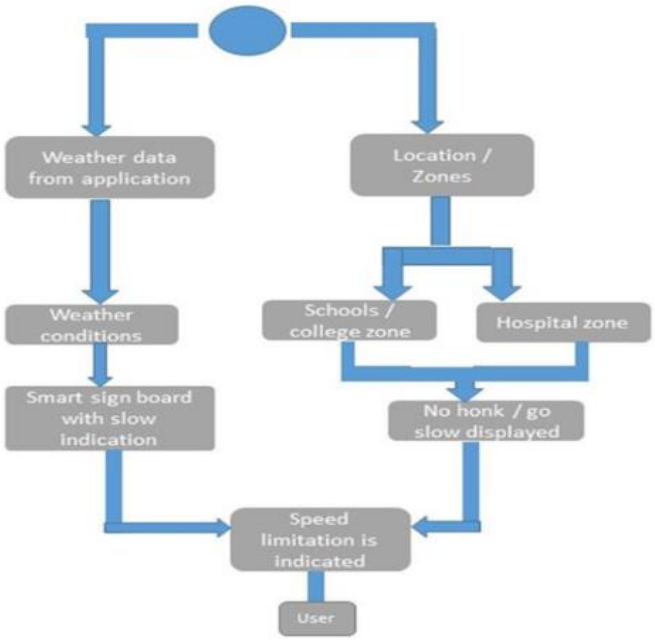
S. NO	FUNCTIONAL REQUIREMENTS (EPIC)	SUB REQUIREMETS (STORY/SUB-TASK)
1.	User Visibility	Sign Boards should be made with LED's which are bright colored and are capable of attracting the driver attention but it should also not be too distracting or blinding cause it may lead to accidents.
2.	User Need	The smart sign boards should be placed frequently in places it is needed and less in places where it is not needed much to avoid confusion for the user during travel.
3.	User Understanding	For better understanding of the driver, the signs should be big, clear and legible and it can also include illustrations which will make it easily understandable to the driver.
4.	User Convenience	The display should be big enough that it should even be visible from far distance clearly.

4.2 Non-Functional requirements

S. NO	NON-FUNCTIONAL REQUIREMENTS	DESCRIPTION
1.	Usability	It should be able to Upgrade and Update when there is a need for it.
2.	Security	It should have good security system so that no other person is able to hack and display
3.	Reliability	It should be able to display to information correctly and error-free.
4.	Performance	It should be able to automatically update itself when certain weather or traffic problem occurs.
5.	Availability	It should be available 24/7 so that it can be beneficial to the customer i.e the driver.
6.	Scalability	It should be able to easily change and upgrade according to change and need in requirement.

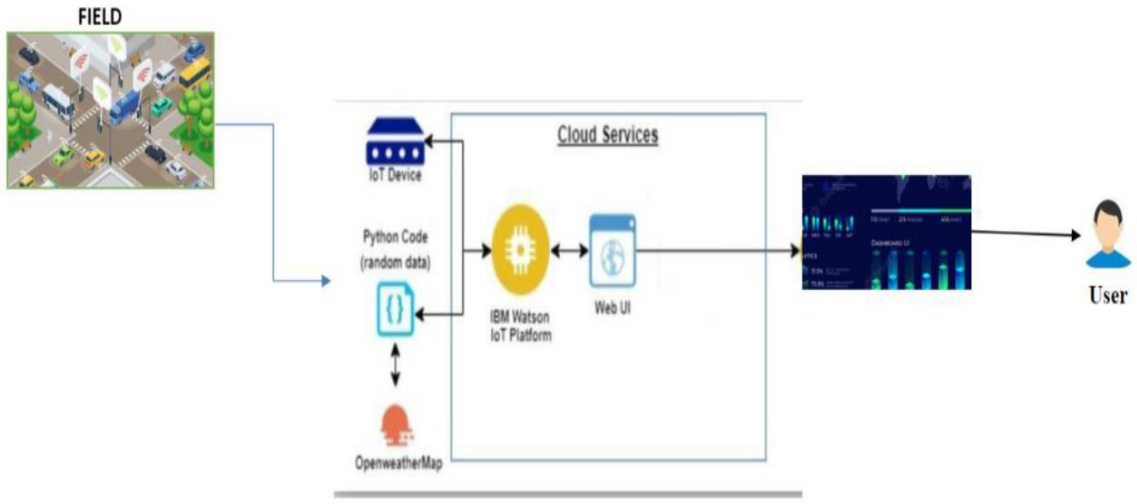
5. PROJECT DESIGN

5.1 Data Flow Diagrams

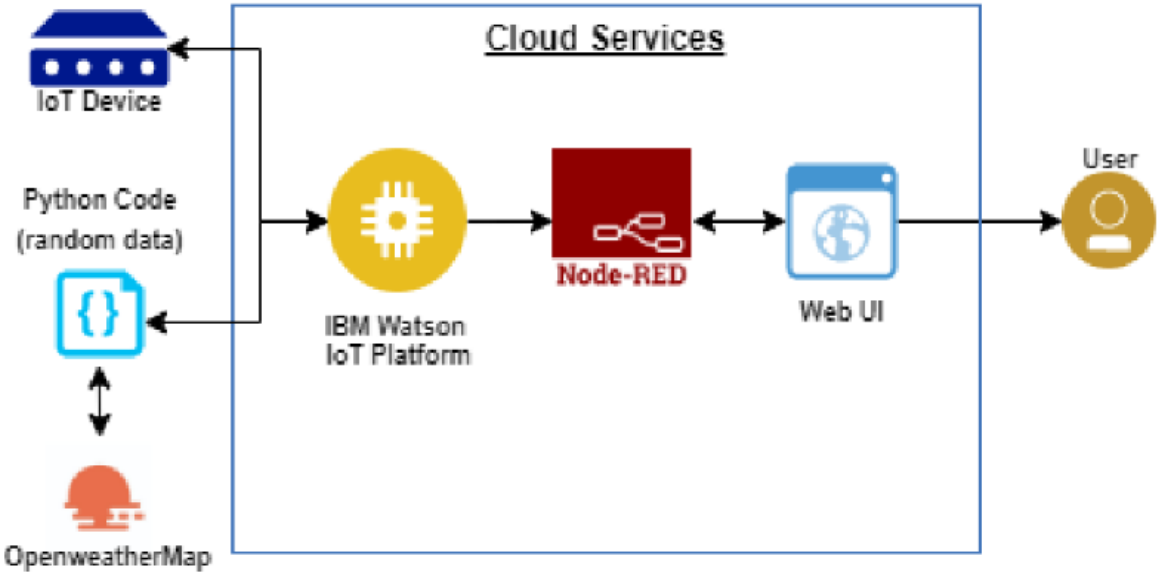


5.2 Solution & Technical Architecture

SOLUTION ARCHITECTURE:



TECHNICAL ARCHITECTURE:



5.3 User Stories

	Functional requirements(epic)	**User story number**	**User story/task**	**Acceptance criteria**	**priority**
User 1	User visibility/usability	USN-1	To display speed limitation on the smart sign boards.	Can view the speed limitations	High
User 2		USN-2	To increase or decrease the speed according to the weather conditions.	Can see the changes corresponding to the weather	High
	Interface	USN-3	As a user the interface or signboard should be simple and easily readable.	Can view the interface easily.	High
	Data	USN-4	Display the data regarding the weather changes.	Weather changes are displayed.	High
	Data	USN-5	Display the zones (sensitive) nearby such as schools, hospitals.	Nearby zones are displayed.	High
Administrator (officials)	Problem solving/fault clearance	USN-6	As an official who is in charge for the proper functioning of the sign boards have to maintain it through periodic monitoring.	Officials can monitor the sign boards for proper functioning.	Medium

6. PROJECT PLANNING & SCHEDULING

a. Sprint Planning & Estimation

sprint	Functional requirements	User story number	User story /task	Story points	priority	Team members
Sprint 1	IDE	USN-1	Installing all the software which are required like python IDE.	2	Low	Sulana. C Deepika. J Varshini. S Kavyaa. G
Sprint 1	Resources	USN-2	Initialization Create and initialize accounts in various public APIs like Open Weather API.	5	Low	Sulana. C Deepika. J Varshini. S Kavyaa. G
Sprint 1		USN-3	Write a Python program that outputs results given the inputs like weather.	13	Medium	Sulana. C Deepika. J Varshini. S Kavyaa. G
Sprint 2		USN-4	Checking the simulation with conditions and Coding.	5	Medium	Sulana. C Deepika. J Varshini. S Kavyaa. G
Sprint 2	software	USN-5	Working with IBM Watson IOT and Node Red integration.	2	High	Sulana. C Deepika. J Varshini. S Kavyaa. G
Sprint 2			Test the above created IOT device s and workflow.	13	High	Sulana. C Deepika. J Varshini. S Kavyaa. G
Sprint 3	Application development	USN-6	Using MIT App Inventor create an App.	13	Medium	Sulana. C Deepika. J Varshini. S Kavyaa. G
Sprint 3			Integrate the MIT app with node -red.	5	Medium	Sulana. C Deepika. J Varshini. S Kavyaa. G

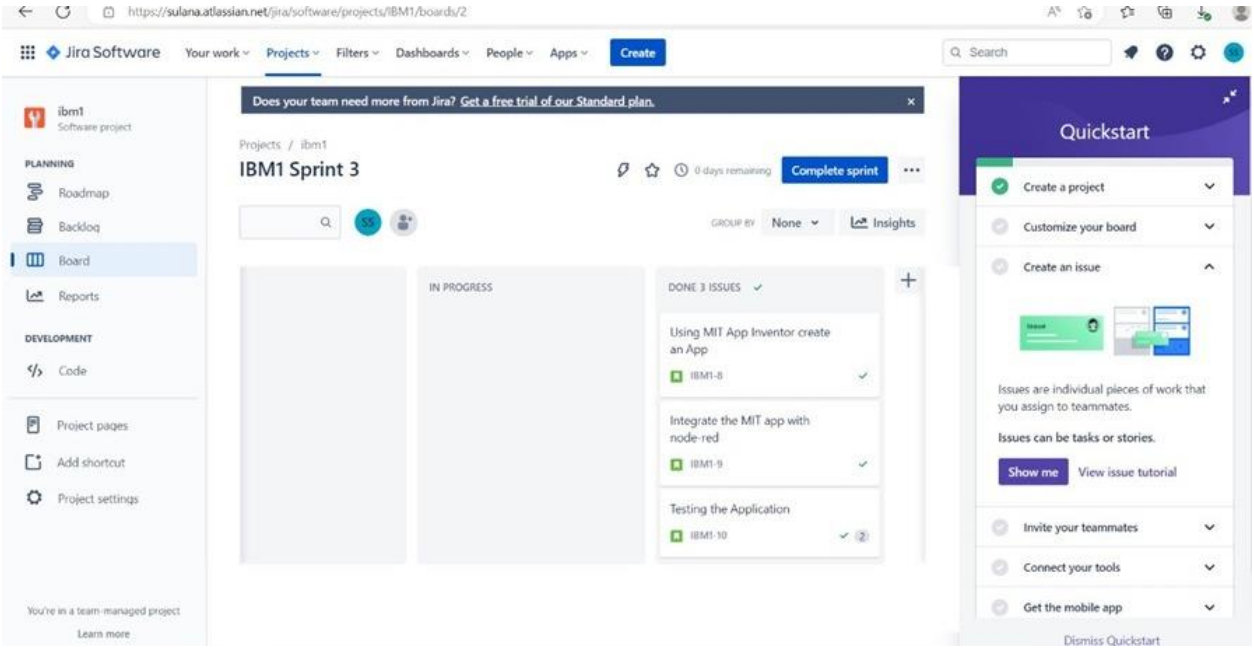
Sprint 3			Testing the Application.	2	Medium	Sulana. C Deepika. J Varshini. S Kavyaa. G
Sprint 4	Interface	USN-7	Displaying speed Limitations.	2	Medium	Sulana. C Deepika. J Varshini. S Kavyaa. G
Sprint 4			Displaying traffic diversion Signs depending on the road conditions.	5	Medium	Sulana. C Deepika. J Varshini. S Kavyaa. G
Sprint 4			Testing of the user interface with the software.	13	Medium	Sulana. C Deepika. J Varshini. S Kavyaa. G

b. Sprint Delivery Schedule

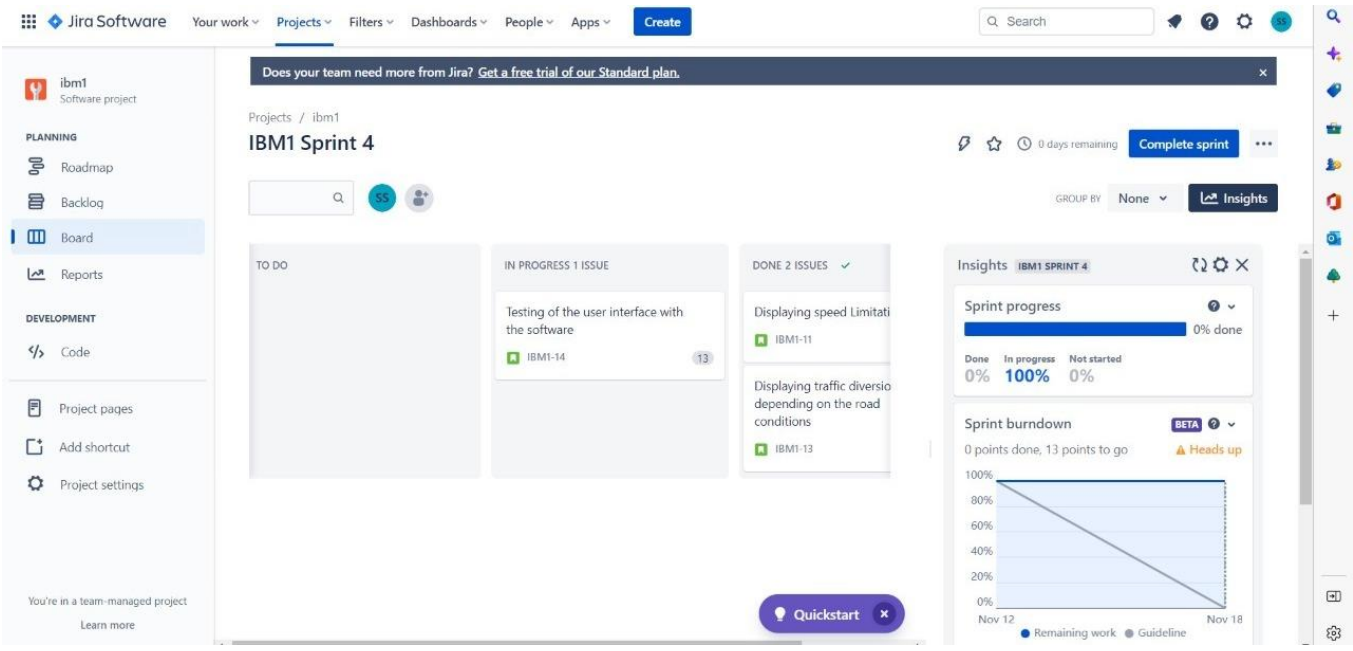
sprint	Total story points	duration	Sprint start date	Sprint end date(planned)	Story points completed (as on planned end date)	Sprint release date (actual)
Sprint-1	20	6 days	24 Oct 2022	25 Oct 2022	20	25 Oct 2022
Sprint-2	20	6 days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 days	14 Nov2022	19 Nov 2022	20	19 Nov 2022

c.) Reports from JIRA

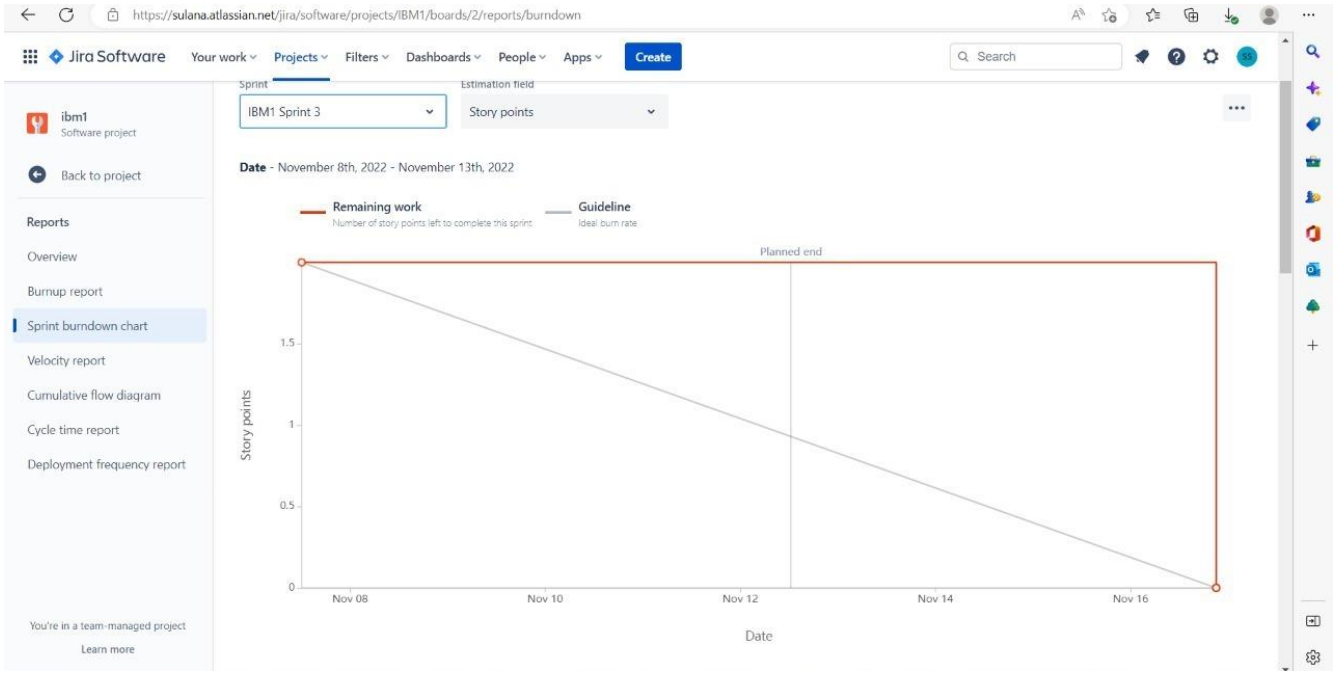
Sprint -3 :



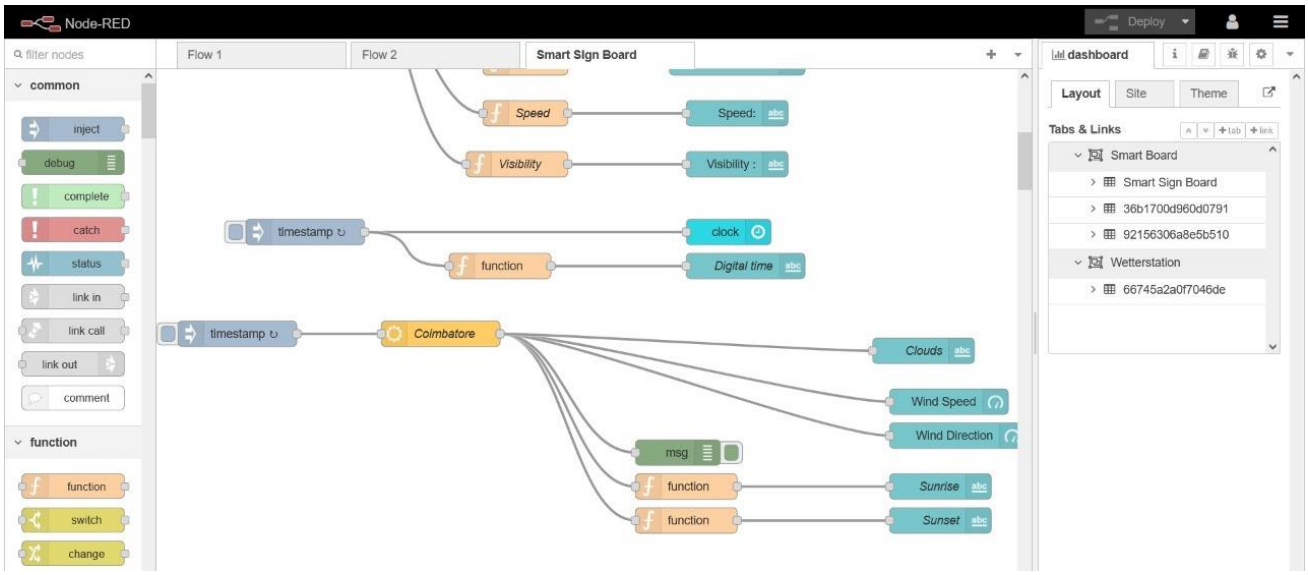
Sprint - 4 :



Burndown Chart for Sprint – 3:

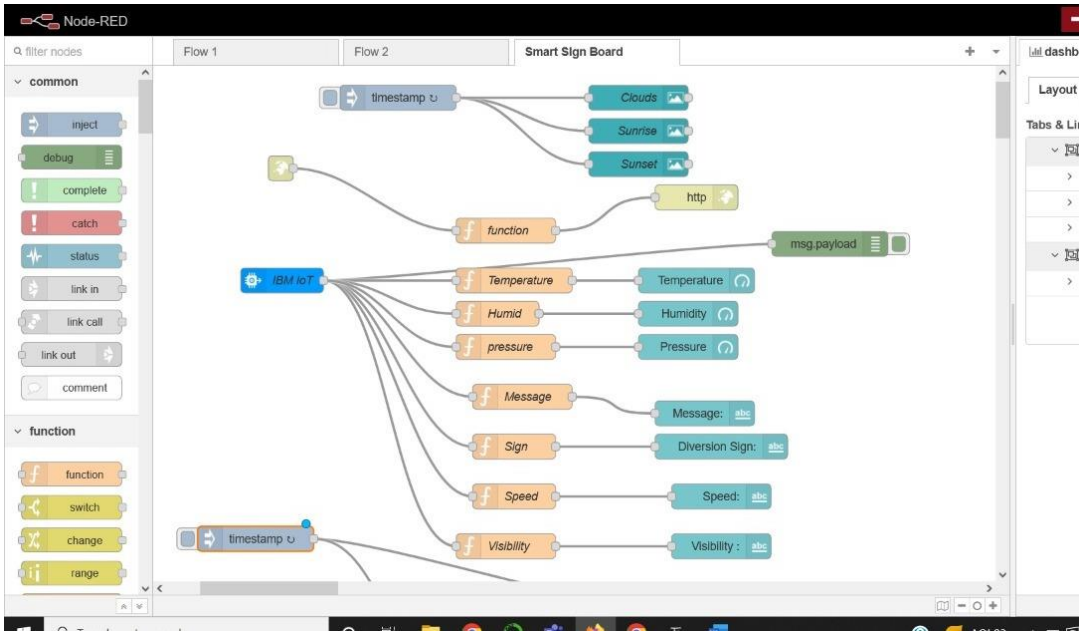


6. CODING & SOLUTIONING (Explain the features added in the project along with code)



6.1 Feature 1 -GET WEATHER DETAILS FOR GIVEN LOCATION

This part of Node RED flow accepts an http GET end point, from which the location, uid, 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 microcontroller. Thus a lot of battery is saved due to lesser processing time.



6.2 Feature 2- GET SPEED LIMITATIONS, MESSAGES, SIGNS

The Node RED flow obtains the data published to the cloud such as speed limitations, messages such as warnings about the zones (schools, hospitals, police stations), signs such as diversions, U-turns etc, and displays them in the dashboard.

7. TESTING

7.1 Test Cases

- TEST CASE 1

Temperature': 303.03, 'Humidity': 51, 'Pressure': 1010, 'Message': 'SLOW DOWN , SCHOOL IS NEAR', 'Sign': '', 'Speed': '', 'Visibility': 'Clear Weather'

- TEST CASE 2

Temperature': 303.03, 'Humidity': 51, 'Pressure': 1010, 'Message': '', 'Sign': 'Left Diversion <-', 'Speed': 'SLOW DOWN , Speed Limit Exceeded', 'Visibility': 'Clear Weather'

- TEST CASE 3

Temperature': 303.03, 'Humidity': 51, 'Pressure': 1010, 'Message': 'SLOW DOWN , HOSPITAL NEARBY', 'Sign': 'Left Diversion <-', 'Speed': '', 'Visibility': 'Clear Weather'

- TEST CASE 4

Temperature': 303.03, 'Humidity': 51, 'Pressure': 1010, 'Message': 'NEED HELP, POLICE STATION NEARBY', 'Sign': 'U Turn', 'Speed': 'Moderate Speed', 'Visibility': 'Clear Weather'.

7.2 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.

8. RESULTS

8.1 Performance Metrics

The performance of the website varies based on the software chosen for implementation . 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.

9. 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.

10. 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, traffics and maintain a peaceful environment.

11. 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 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.

12. APPENDIX

Source Code :

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

myConfig = {
   "identity": {
      "orgId": "ojfcbe",
      "typeId": "sulan",
      "deviceId":"1234"
   },
   "auth": {
      "token": "RsCA-twpue)2)c8j&r"



   }
}

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()

BASE_URL = "https://api.openweathermap.org/data/2.5/weather?"
CITY = "Coimbatore"
URL = BASE_URL + "q=" + "Coimbatore" + "&appid=" +
"fbcb52a2a6c7bbea1396de2b6b17ea8a"

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']
            repo=random.randint(0,5)
            if repo==1:
               prt="SLOW DOWN , SCHOOL IS NEAR"
            elif repo==3:
               prt="SLOW DOWN , HOSPITAL NEARBY"
            elif repo==5:
               prt="NEED HELP, POLICE STATION NEARBY"
            else:
                prt=""
            speed=random.randint(0,150)
            if speed>=100:
               prt3="SLOW DOWN , Speed Limit Exceeded"
            elif speed>=60 and speed<100:
               prt3="Moderate Speed"
            else:
                prt3="Usual speed limit"
            sign=random.randint(0,5)
            if sign==1:
               prt2="Right Diversion ->"
            elif sign==3:
               prt2="Left Diversion <-"
            elif sign==5:
               prt2="U Turn"
            else:
                prt2=""
            if temperature<=50:
               prt4="Fog Ahead, Drive Slow"
            else:
                prt4="Clear Weather"

        else:
            print("Error in the HTTP request")
        myData={'Temperature':temperature,'Humidity':humidity,'Pressure':pressure,
'Message':prt, 'Sign':prt2, 'Speed':prt3, 'Visibility':prt4}
        client.publishEvent(eventId="status", msgFormat="json", data=myData, qos=0,
onPublish=None)
```

```
        print("Published data Successfully: %s", myData)
        client.commandCallback = myCommandCallback
        time.sleep(5)
    client.disconnect()
```

GitHub & Project Demo Link

GitHub link - https://github.com/IBM-EPBL/IBM-Project-10904-1659243422

Project Demo link - https://github.com/IBM-EPBL/IBM-Project-10904-1659243422/blob/main/Final%20Deliverables/final%20code/PNT2022TMID14540%20-demo%20video.mp4