

SIGNS WITH SMART CONNECTIVITY FOR BETTER ROAD SAFETY

IBM PROJECT REPORT

Team Leader : B.GAYATHRI

Team Members :

1. T.S.DIVYASRI
2. R.KARISHMA
3. M.SELVI

Faculty Mentor : T.R.MUTHU

Evaluator : S. SUBHA

Industry Mentor :

1. SANTOSHI

Team ID: PNT2022TMID11493

CONTENTS

1. INTRODUCTION

- 1.1 Project Overview
- 1.2 Purpose

2. LITERATURE SURVEY

- 2.1 Existing problem
- 2.2 References
- 2.3 Problem Statement Definition

3. IDEATION & PROPOSED SOLUTION

- 3.1 Empathy Map Canvas
- 3.2 Ideation & Brainstorming
- 3.3 Proposed Solution
- 3.4 Problem Solution fit

4. REQUIREMENT ANALYSIS

- 4.1 Functional requirement
- 4.2 Non-Functional requirements

5. PROJECT DESIGN

- 5.1 Data Flow Diagrams
- 5.2 Solution & Technical Architecture
- 5.3 User Stories

6. PROJECT PLANNING & SCHEDULING

- 6.1 Sprint Planning & Estimation
- 6.2 Sprint Delivery Schedule
- 6.3 Reports from JIRA

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

- 7.1 Feature 1
- 7.2 Feature 2
- 7.3 Database Schema (if Applicable)

8. TESTING

- 8.1 Test Cases
- 8.2 User Acceptance Testing

9. RESULTS

- 9.1 Performance Metrics

10. ADVANTAGES & DISADVANTAGES

11. CONCLUSION

12. FUTURE SCOPE

13. APPENDIX

- Source Code
- GitHub & Project Demo Link

Chapter-1

1.INTRODUCTION :

1.1.PROJECT OVERVIEW :

In the present systems, the road signs and the speed limits are static. But there are certain situations, at which there may be a need to update these signs. We can consider some cases such as we need road diversions due to heavy traffic or due to accidents, or the speed limits need to be updated due to severe climatic conditions. Under such scenarios if these signs were digitized, we can change the road signs accordingly. These intelligent transportation systems offer significant opportunities to save lives. In today's life, road safety is a significant area of concern. Our project aims to provide automatic updation of the smart road signs for better road safety.

1.2.PURPOSE :

So the purpose of this project is to digitize the road signs by showing alert messages through mobile phones. This alert message will get updated by dynamic inputs. The static signs cannot be updated when needed, which when digitized can be made convenient and thus been benefitted. The alert message will show the user to drive the vehicle according to the climatic changes, traffic and scenarios. This will alert the user about the speed limits and prevent the accidents that are happening on the road. The inputs are obtained dynamically and it is processed to take decisions of speed, routes and other signs are altered accordingly.

The climatic conditions and the weather will be obtained dynamically from the OpenweatherApi. This information will be extracted by a python script which has been developed. Based on the scenarios where the weather is predicted the alert message will be sent to the user. The IOT platform will be created in IBM Watson platform and the python script that was developed will be integrated with it.

NODE-RED is a programming tool for wiring together hardware devices, API's and other online services in many ways. The IOT platform will be integrated with NODE-RED platform and the alert message will be sent to the mobile phone by integrating the NODE-RED with the user application. The vibration alert will be displayed with the speed limits in the authorized person's mobile.

Chapter-2

2.LITERATURE SURVEY :

2.1.Existing problem :

The existing methodology has certain systems in the vehicles that recognize these sign boards and allows control of vehicles and routes accordingly. This makes use of RFID transmitters and receivers in vehicles. The vehicle to vehicle transmission is also used for sharing of information. There are many solutions for automatically recognizing the road signs by the vehicle itself. The automatic recognition of traffic signs is essential to autonomous driving, assisted driving, and driving safety. Currently, convolutional neural networks are the most popular deep learning algorithm in traffic sign recognition. For the accuracy of visual inspection, a region of interest(ROI) extraction method was designed through content analysis and key information recognition. Besides, a Histogram of oriented gradients (HOG) method was developed for image detection to prevent projection distortion. Furthermore, a traffic sign recognition learning architecture was created based on CapsNet, which relies on neurons to represent target parameters like dynamic routing, path pose and direction, and effectively capture the traffic sign information from different angles or directions.

2.2.References :

- 1.Satadal Saha, Subhadip Basu, Mita Nasipuri and Dippakumar Basu, “Development of an automated Red Light Violation Detection System(RLVDS)for Indian Vehicles”, Processing of IEEE National Conference on Computing and Communicating Systems.
- 2.M.S. Aminian, A.Allamehzadeh M.Mostand, C.Olaverri-Monreal, “Cost-efficient traffic sign detection relying on smart mobile devices” In International Conference on Computer Aided Systems Theory ,pp, 419-426, Feb, 2017.

2.3.Problem Statement Definition :

This project will replace the static boards to smart signed boards that will change the speed limits according to the weather climate and show diversion messages if there are accidents in the road and alert messages if there are hospital schools or any roadworks.

Chapter-3

3.IDEATION & PROPOSED SOLUTION :

3.1.Empathy Map Canvas :

An empathy map is a visual tool for gaining insight into user's perspective. It has an outline of various aspects of one or multiple users on a chart, this can be shared and collaborated easily. The empathy map is given as follows;

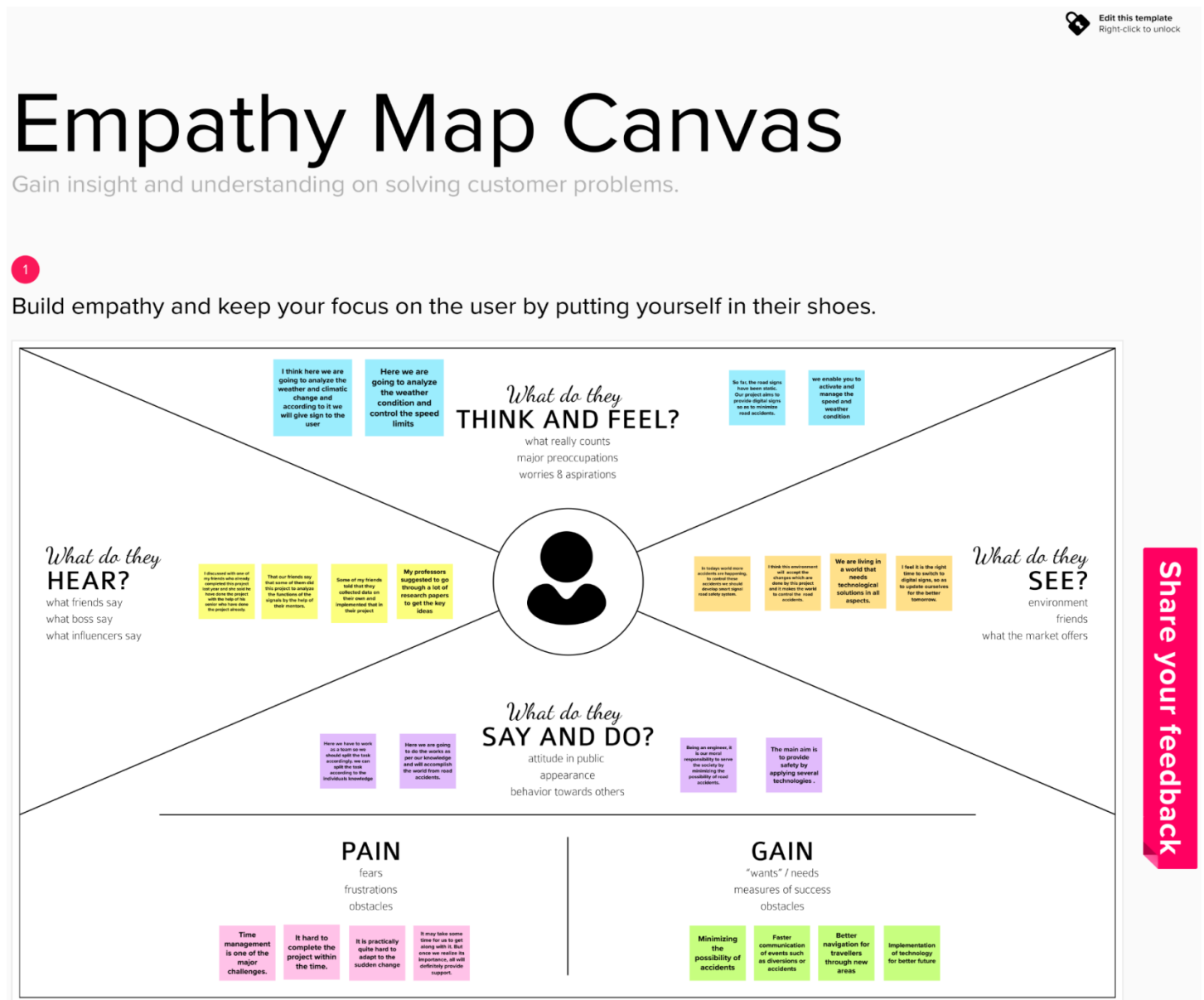


figure.1.1.Empathy Map Canvas

3.2.Ideation & Brainstorming :

Brainstorming is the method design teams used to generate ideas to solve clearly defined design problems.

In a controlled condition and a free thinking environment , teams approach a problem by such means as

“How might we”.The following figure represent a brainstorming image:

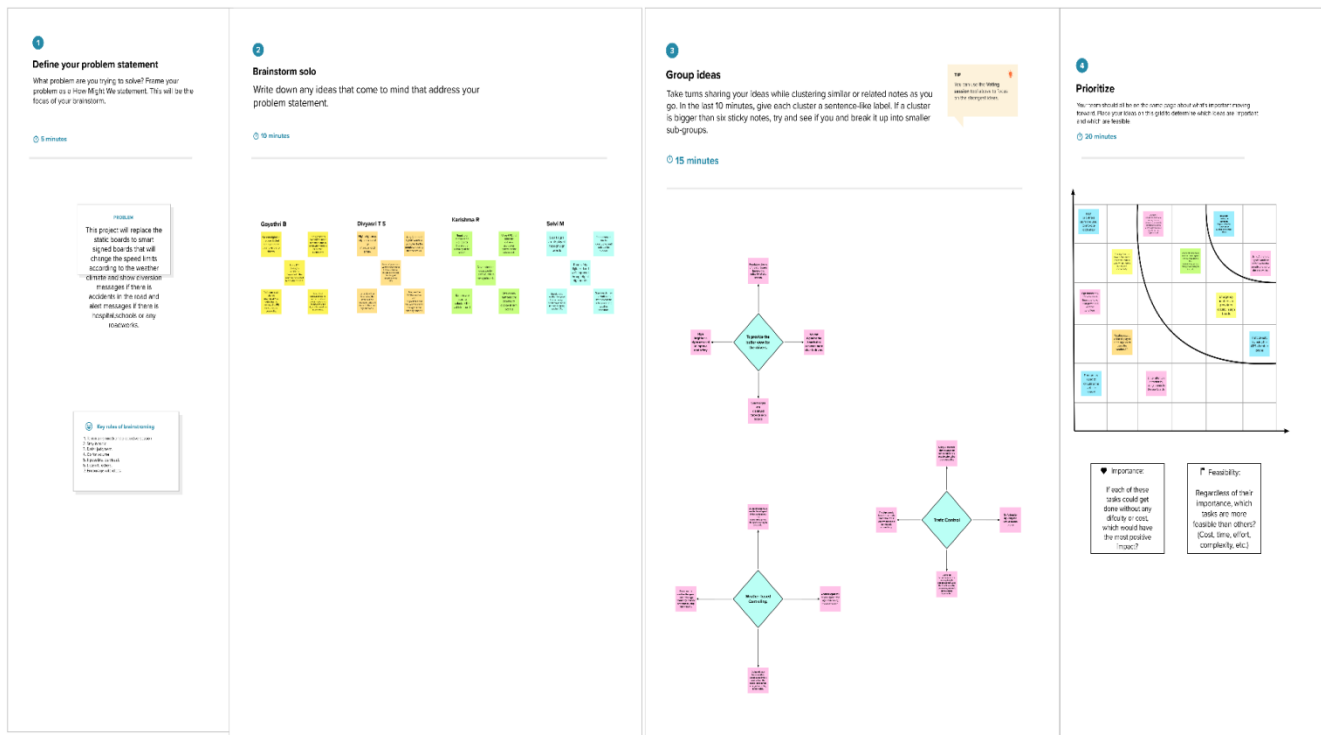


figure.1.2.Ideation and Brainstorming

3.3.Proposed Solution :

The following table consists of the parameters to be consider while developing the solution:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	This project will replace the static boards to smart signes boards that will change the speed limits according to the weather climate and show diversion messages if there is accidents in the road.

2.	Idea / Solution description	<ul style="list-style-type: none"> • The weather and temperature details are obtained from API. The speed limit will be updated automatically in accordance with the weather. • The details regarding any accidents and traffic congestion faced on the particular roads are obtained.
3.	Novelty / Uniqueness	<ul style="list-style-type: none"> • Generic sign board for all the applications that uses both buttons and web service for updation pedestrians are given the access to request the sign change of the signal change to cross the road
4.	Social Impact / Customer Satisfaction	<ul style="list-style-type: none"> • Diversion reasons will be displayed if there is no traffic, pedestrians can cross the street without waiting. Customers can reach the destination before the expected time..
5.	Business Model (Revenue Model)	<ul style="list-style-type: none"> • Since APIs are used to actively monitor the customers environment, this project employs a business strategy in which revenue will be generated on the basis of the length of time in which the customers actively interact with the product. • This product is aimed to be free of cost to the public. • The public will also gain information about the roads.
6.	Scalability of the Solution	<ul style="list-style-type: none"> • In the future, if any update is required either on the hardware or software side, it can be easily implemented. • The hardware components can be directly interfaced with the microcontroller and small modifications can be made in the programming. • The website application has to be updated with the additional functionality

Table.1.1.Proposed Solution

3.4.Problem Solution fit :

Problem solution fit is validating the problem that exists using real data and feedback.

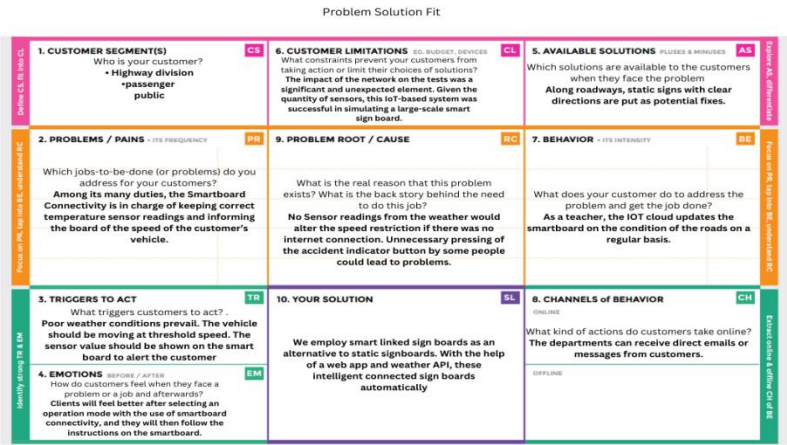


figure..3.1.Problem Solution fit

Chapter-4

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 Visibility	Sign Boards should be made of bright colored LEDs capable of attraction driver's attention not too distracting to cause accidents
FR-2	User Understanding	Should display information through means like images/illustration with text so that the user can understand the signs correctly.
FR-3	User Convenience	Display should be big enough to display all the signs correctly so that it is visible even to far away drivers..
FR-4	Accuracy analysis	The message should be accurate according to the current scenarios.
FR-5	Dynamic changes	Should not cause any buffer.

Table.4.1.Functional Requirements

4.2.Non-Functional requirements :

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

FR No.	Non-Functional Requirement	Description
NFR 1	Usability	➤Should be able to dynamically update with respective time .
NFR 2	Security	➤Should be secure enough that only the intendedmessages are displayed in the display.
NFR 3	Reliability	➤Should convey the traffic information correctly.
NFR 4	Performance	➤ Display should update dynamically whenever the weather or traffic values are updated.
NFR 5	Availability	➤Should be on service 24/7.
NFR 6	Scalability	➤Should be modular and hence able to scale observers horizontally.

Table.4.2.Non Functional Requirements

Chapter-5

5.PROJECT DESIGN :

5.1Data Flow Diagrams :

Data flow diagram is used to represent the flow of the project development, the data flow diagram for signs with Smart Connectivity for Better Road Safety is given below:

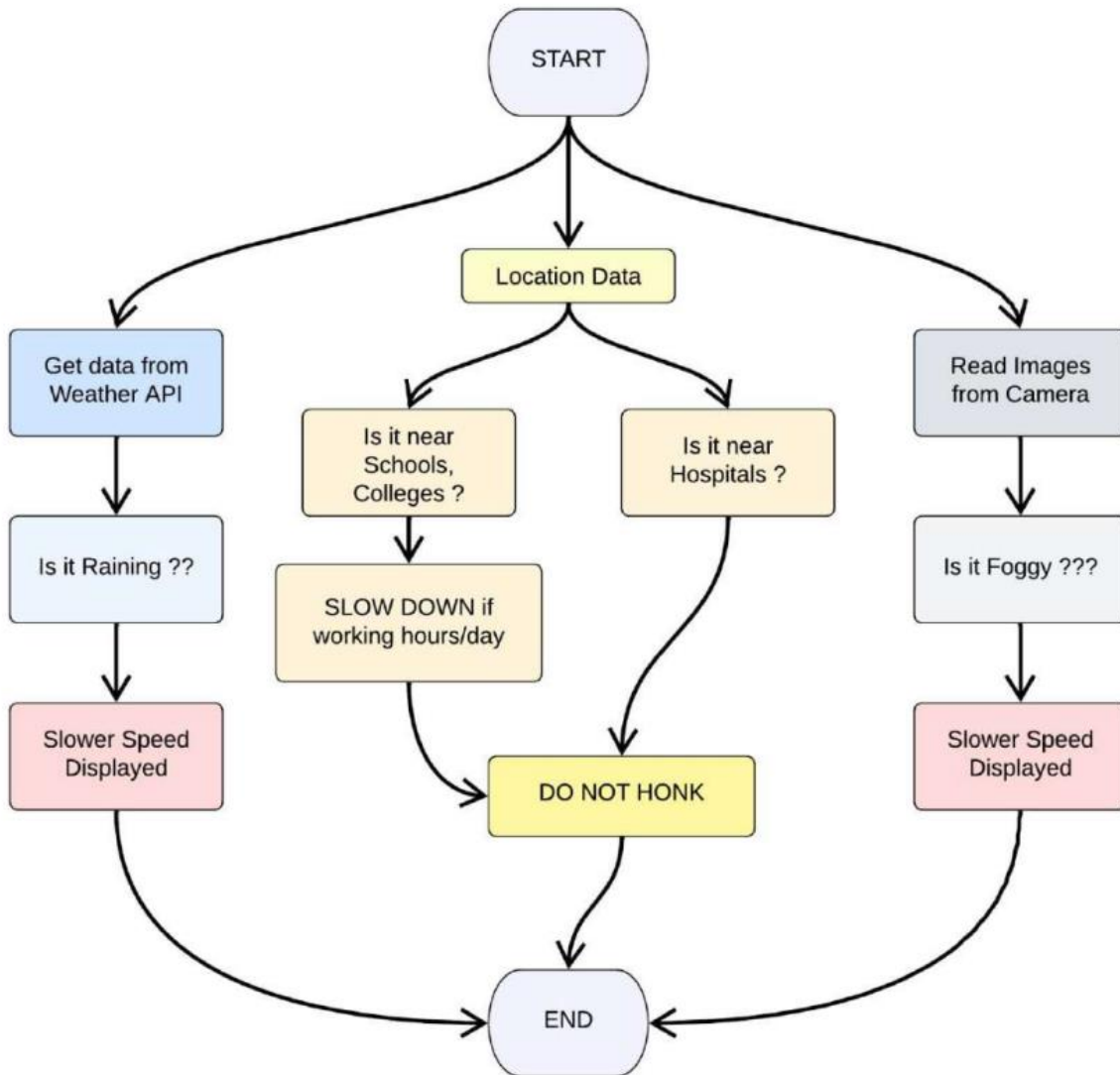


figure.5.1.Project Design Flow

5.2.Solution & Technical Architecture :

5.2.1.Solution and technical architecture diagram is given as follows:

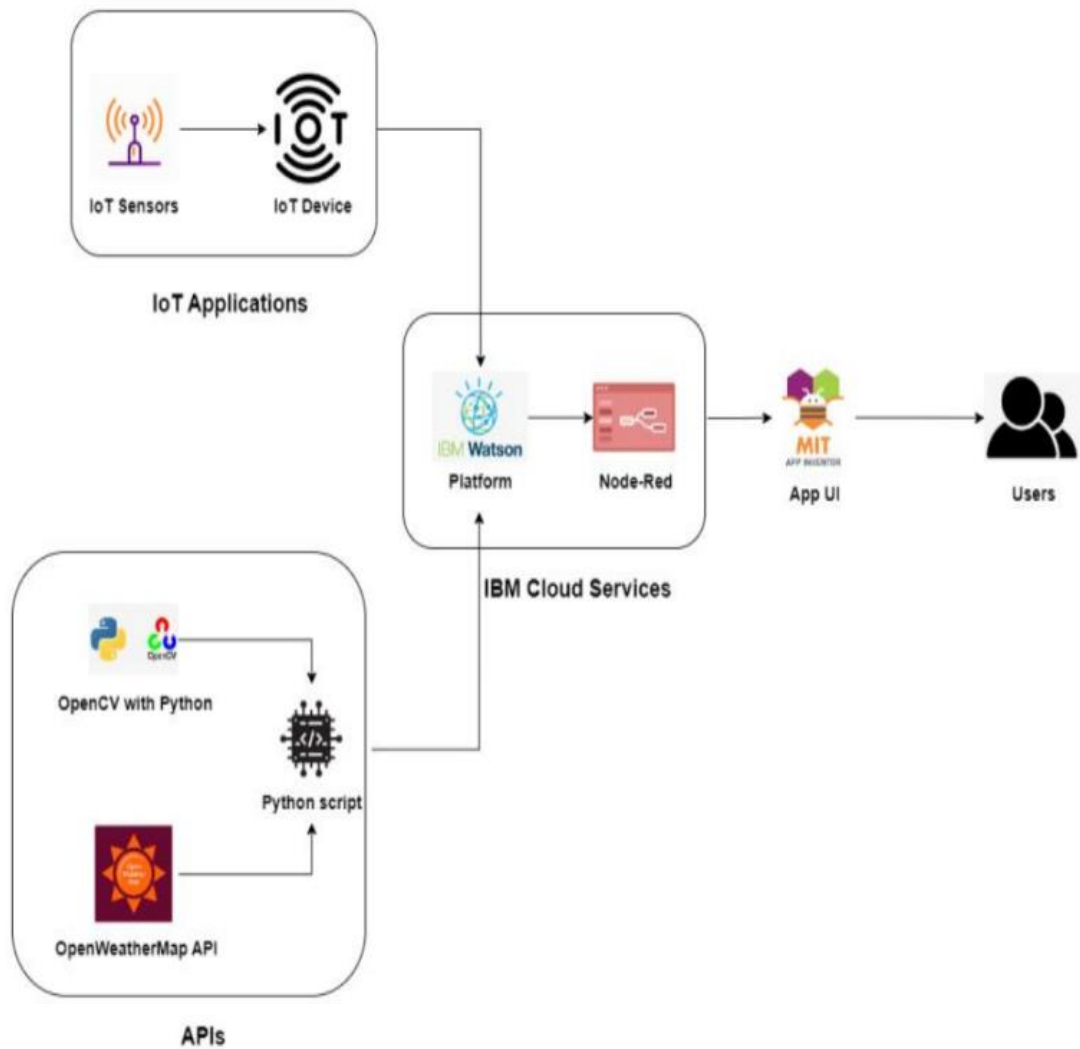


figure.5.2.Solution and Technology Architecture

5.2.2.Solution Built

Components and technologies are given below:

S.No	Component	Description	Technology
1	User Interface	User can interact with the app using MIT App	HTML, CSS, JavaScript / Angular Js /React Js
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.	IBM Cloud
6	Cloud Database	Database Service on Cloud	IBM DB2, IBM Cloudant etc.
7	File Storage	File storage requirements	IBM Block Storage or Other StorageService or Local Filesystem
8	External API-1	Purpose of External API used in the application	Open Weather Map API
9	External API-2	Purpose of External API used in the application	IBM Watson Platform, Node - Red
10.	Infrastructure (Server / Cloud)	Application Deployment on Local System / CloudLocal Server Configuration: Cloud Server Configuration:	Local, Cloud Foundry, Kubernetes

figure.5.1.Solution Built

5.2.User Stories :

User story consists of the journey of the user throughout the development of this project. The user journey map is given below:

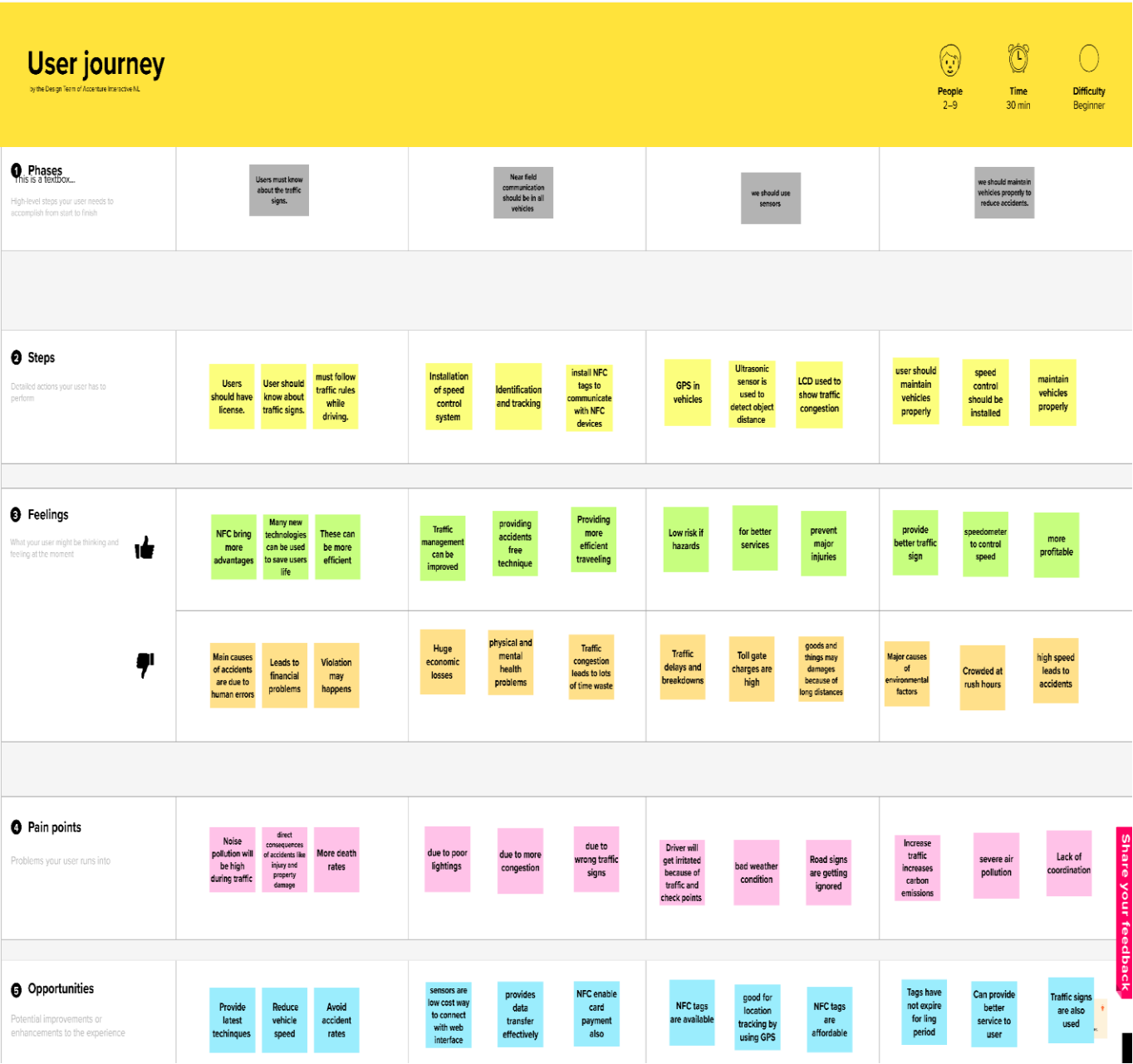


figure.5.2.User Stories

Chapter-6

6.1.PROJECT PLANNING & SCHEDULING :

6.1.Sprint Planning & Estimation :

Sprint	Functional requirements	User story or task	Story points	Priority	Team Members
Sprint-1	Resources Initialization	Create and initialise accounts in various public APIs like OpenWeatherApi.	1	Low	Gayathri Karishma DivyaSri Selvi
Sprint-1	Local Server/Software Run	Write a Python program that outputs results given the inputs like weather and location.	1	Medium	Gayathri Karishma DivyaSri Selvi
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	Gayathri Karishma DivyaSri Selvi
Sprint-3	Hardware Initialization	Integrate the hardware to be able to access the cloud functions and provide inputs to the same	2	High	Gayathri Karishma DivyaSri Selvi
Sprint-4	UI/UX Optimisation & Debugging	Optimize all the shortcomings and provide better user experience	2	High	Gayathri Karishma DivyaSri Selvi

Table.6.1.Sprint Planning

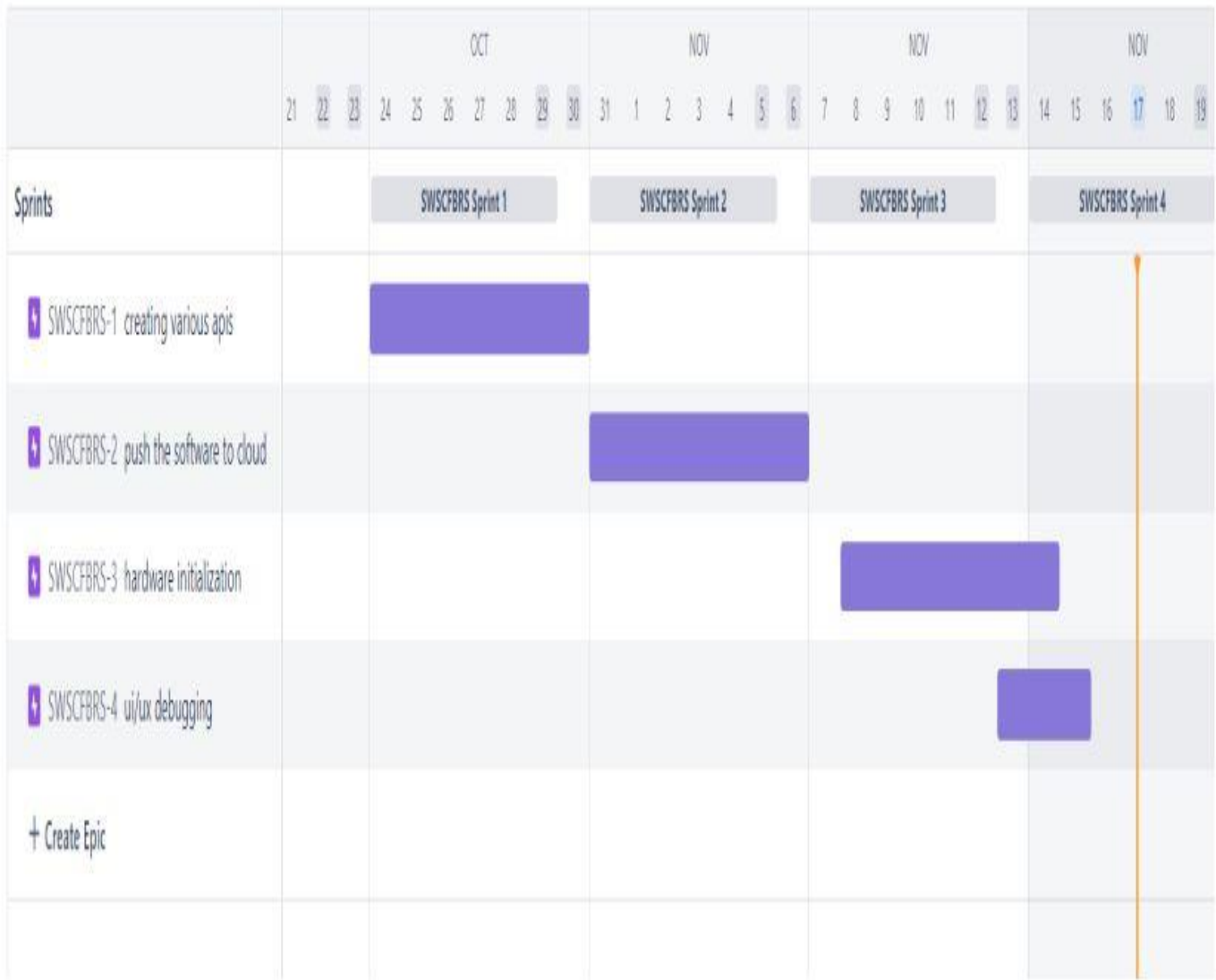
6.2.Sprint Delivery Schedule :

Project Tracker, Velocity & Burndown Chart:

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint end Date	Story Points Completed(Based 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

Table.6.2.Sprint delivery schedule

6.2.Reports from JIRA:



Chapter-7

7.CODING & SOLUTIONING :

7.1.Feature-1:

Python Code:

The following python program was written to integrate the temperature and the humidity information from OpenWeatherApi to the IBM IOT Platform.

Brain.py:

```
#Python code
# IMPORT SECTION STARTS
import weather
from datetime import datetime as dt
# IMPORT SECTION ENDS
# -----
# UTILITY LOGIC SECTION STARTS
def processConditions(myLocation,APIKEY,localityInfo):
    weatherData = weather.get(myLocation,APIKEY)
    finalSpeed = localityInfo["usualSpeedLimit"] if "rain" not in weatherData else localityInfo["usualSpeedLimit"]/2
    finalSpeed = finalSpeed if weatherData["visibility"]>35 else finalSpeed/2
    if(localityInfo["hospitalsNearby"]):
        # hospital zone
        doNotHonk = True
    else:
        if(localityInfo["schools"]["schoolZone"]==False):
            # neither school nor hospital zone
            doNotHonk = False
        else:
            # school zone
            now = [dt.now().hour,dt.now().minute]
            activeTime = [list(map(int,_split(":"))) for _ in localityInfo["schools"]["activeTime"]]
            doNotHonk = activeTime[0][0]<=now[0]<=activeTime[1][0] and activeTime[0][1]<=now[1]<=activeTime[1][1]
    return({
        "speed" : finalSpeed,
        "doNotHonk" : doNotHonk
    })
```

```

})
# UTILITY LOGIC SECTION ENDS

main.py:
# Python code
# IMPORT SECTION STARTS
import brain
# IMPORT SECTION ENDS
# -----
# USER INPUT SECTION STARTS
myLocation = "Chennai,IN"
APIKEY = "c7388b7d0d823ee0ee0be65c6fd40411"
localityInfo = {
    "schools": {
        "schoolZone": True,
        "activeTime": ["7:00", "17:30"] # schools active from 7 AM till 5:30 PM
    },
    "hospitalsNearby": False,
    "usualSpeedLimit": 40 # in km/hr
}
# USER INPUT SECTION ENDS
# -----
# MICRO-CONTROLLER CODE STARTS
print(brain.processConditions(myLocation, APIKEY, localityInfo))
'''
MICRO CONTROLLER CODE WILL BE ADDED IN SPRINT 2 AS PER OUR PLANNED SPRINT SCHEDULE
'''
# MICRO-CONTROLLER CODE END

```

Weather.py:

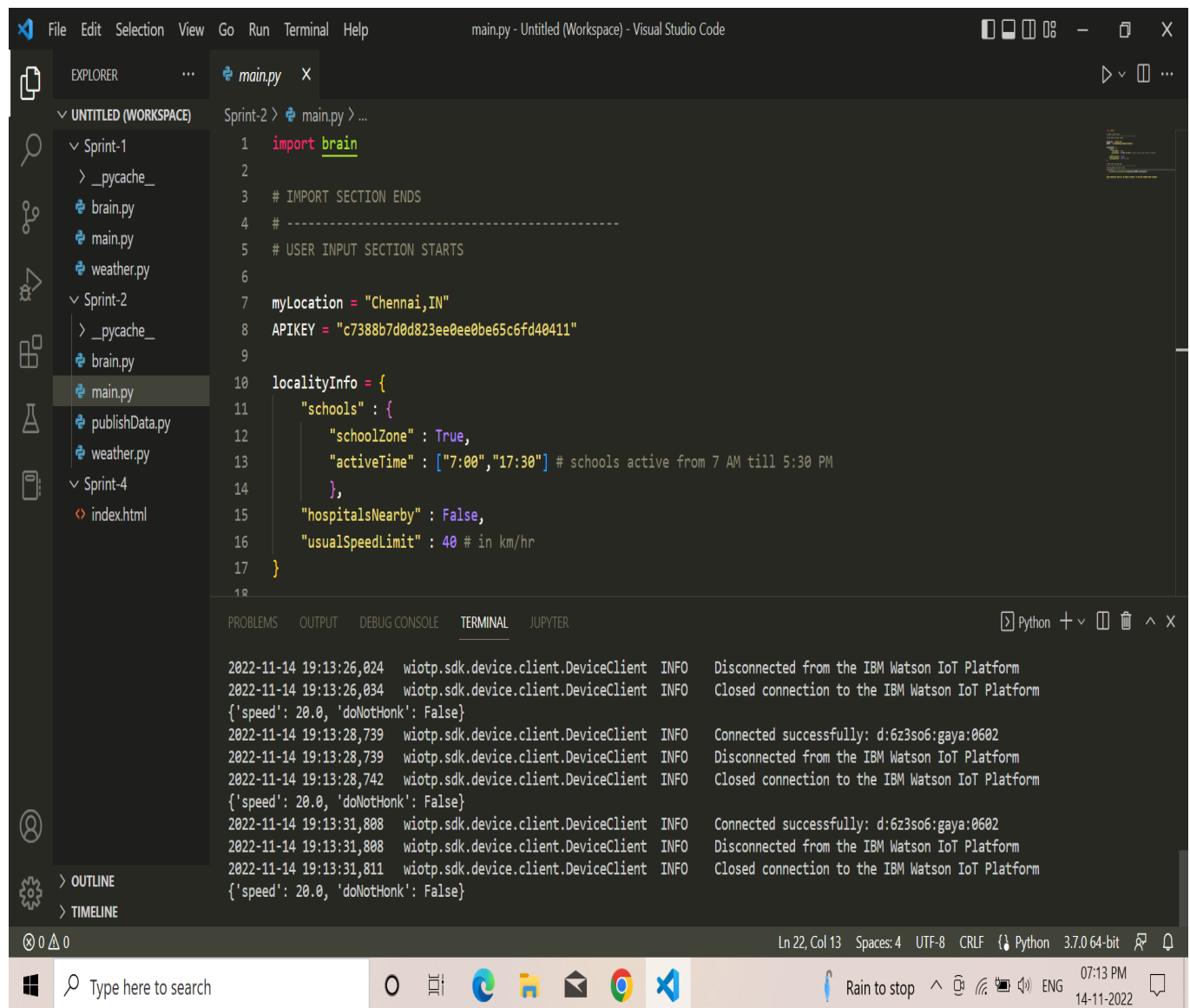
```
# Python code

import requests as reqs

def get(myLocation, APIKEY):
    apiURL = f"https://api.openweathermap.org/data/2.5/weather?q={ myLocation }&appid={ APIKEY }"
    responseJSON = (reqs.get(apiURL)).json()
    responseObject = {
        "temperature":
            responseJSON['main']['temp'] - 273.15,
        "weather": [
            responseJSON['weather'][_]['main'].lower()
            for _ in range(len(responseJSON['weather']))
        ],
        "visibility":
            responseJSON['visibility'] /
            100, # visibility in percentage where 10km is 100% and 0km is 0%
    }
    if ("rain" in responseJSON):
        responseObject["rain"] = [
            responseJSON["rain"][key] for key in responseJSON["rain"]
        ]
    return (responseObject)
```

Outputs:

The following images shows the output of the above python program:



The screenshot displays the Visual Studio Code interface with a Python file named `main.py` open. The file contains the following code:

```
1 import brain
2
3 # IMPORT SECTION ENDS
4 # -----
5 # USER INPUT SECTION STARTS
6
7 myLocation = "Chennai,IN"
8 APIKEY = "c7388b7d0d823ee0ee0be65c6fd40411"
9
10 localityInfo = {
11     "schools": {
12         "schoolZone": True,
13         "activeTime": ["7:00", "17:30"] # schools active from 7 AM till 5:30 PM
14     },
15     "hospitalsNearby": False,
16     "usualSpeedLimit": 40 # in km/hr
17 }
18
```

The terminal window at the bottom shows the output of the program, which consists of a series of log messages from the `wiotp.sdk.device.client.DeviceClient` module. The messages indicate that the program successfully connected to the IBM Watson IoT Platform and then disconnected. The output is as follows:

```
2022-11-14 19:13:26,024 wiotp.sdk.device.client.DeviceClient INFO Disconnected from the IBM Watson IoT Platform
2022-11-14 19:13:26,034 wiotp.sdk.device.client.DeviceClient INFO Closed connection to the IBM Watson IoT Platform
{'speed': 20.0, 'doNotHonk': False}
2022-11-14 19:13:28,739 wiotp.sdk.device.client.DeviceClient INFO Connected successfully: d:6z3so6:gaya:0602
2022-11-14 19:13:28,739 wiotp.sdk.device.client.DeviceClient INFO Disconnected from the IBM Watson IoT Platform
2022-11-14 19:13:28,742 wiotp.sdk.device.client.DeviceClient INFO Closed connection to the IBM Watson IoT Platform
{'speed': 20.0, 'doNotHonk': False}
2022-11-14 19:13:31,808 wiotp.sdk.device.client.DeviceClient INFO Connected successfully: d:6z3so6:gaya:0602
2022-11-14 19:13:31,808 wiotp.sdk.device.client.DeviceClient INFO Disconnected from the IBM Watson IoT Platform
2022-11-14 19:13:31,811 wiotp.sdk.device.client.DeviceClient INFO Closed connection to the IBM Watson IoT Platform
{'speed': 20.0, 'doNotHonk': False}
```

The status bar at the bottom of the window shows the current file is `main.py`, the editor is in `Ln 22, Col 13`, the encoding is `UTF-8`, the line ending is `CRLF`, the interpreter is `Python 3.7.0 64-bit`, and the system clock shows `07:13 PM 14-11-2022`.

figure.7.1.Python Output

The data will be sent to the IBM IOT Platform as shown below:

The screenshot displays the IBM Watson IoT Platform dashboard. The browser address bar shows the URL: <https://6z3so6.internetofthings.ibmcloud.com/dashboard/devices/browse>. The dashboard header includes the IBM Watson IoT Platform logo and a user profile for 910619106015@smartinternz.com with ID: 6z3so6. The main navigation bar has tabs for Browse, Action, Device Types, and Interfaces, along with an Add Device button. The central panel shows the 'Recent Events' tab for a selected device. It contains a table with the following data:

Event	Value	Format	Last Received
status	{"temperature":25.990000000000001,"visibility"...	json	a few seconds ago
status	{"temperature":25.990000000000001,"visibility"...	json	a few seconds ago
status	{"temperature":25.990000000000001,"visibility"...	json	a few seconds ago
status	{"temperature":25.990000000000001,"visibility"...	json	a few seconds ago
status	{"temperature":25.990000000000001,"visibility"...	json	a few seconds ago

Below the table, it indicates '0 Simulations running'.

figure.7.2.Ibm Output Data

7.2.Feature-2

Node RED is a programming platform which is used to used

APPENDIX:

- **GITHUB AND PROJECT DEMO LINK**

<https://github.com/IBM-EPBL/IBM-Project-30857-1660191214>

- **PROJECT DEVELOPMENT PHASE LINK**

<https://github.com/IBM-EPBL/IBM-Project-47838-1664170967/tree/main/Project%20Development%20Phase>

- **DEMO VIDEO DOWNLOAD LINK**

<https://youtu.be/sKTWlg5Rdvg>

CONCLUSION:

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

FUTURE SCOPE:

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