IOT BASED SMART SOLUTION FOR RAILWAYS

MEMBERS: Reju Kannan, Ragavi.K, Sowmiya.N.S, Sneha.S.L

1.INTRODUCTION

PROJECT OBJECTIVE:

The Indian rail way reservation system the passengers face many difficulties to book the tickets, cancelling tickets, checking Passenger Name Record number(PNR). The developed Countries IIOT had a major impact on the smart transportation industry, with advent of autonomous vehicles and improved cargo management . Handling the passengers reservation data has been a key point of consideration in most railway service. The smart railways research report also providers an in-depth analysis of proposed and ongoing projects by various countries. The Aadhaar number in India UIDAI would serve as a major backbone for this entire Smart Passenger Reservation System (SPRS). The services offered by the Passenger Reservation System (PRS) would be provided keeping this UIDAI as the primary identification key.

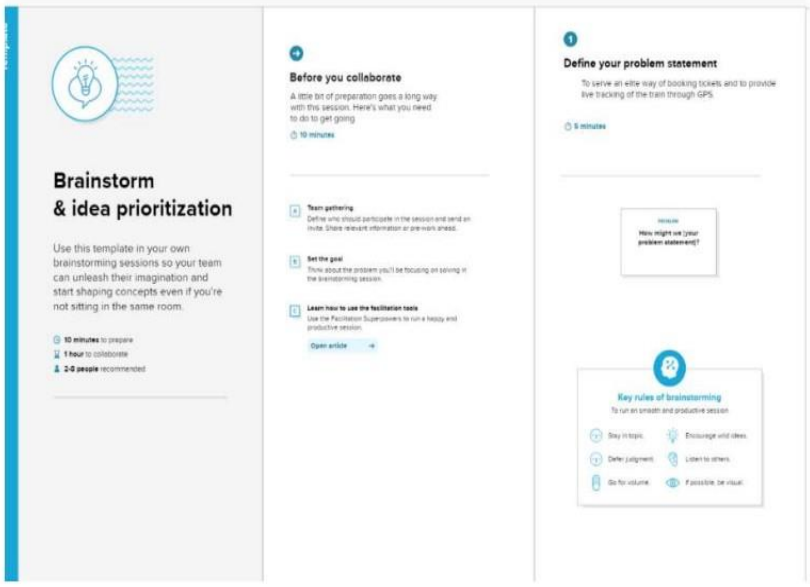
PURPOSE:

The purpose of this project is to provide ticketing using IOT(Internet Of Things) to make the booking of ticket easy. The railway reservation system facilitates the passengers to enquiry about the train available on the basics of source and destination, booking and cancellation of tickets , enquiry about the status of the booked ticket etc.., The aim of case study is to design and develop a data base maintaining records of different trains, train status and passengers .This project contains introduction to the railways reservation system. It is the computerized system of reserving the seat of the train in advance. Online reservation has made the process for the reservation of seats very much easier than ever before.

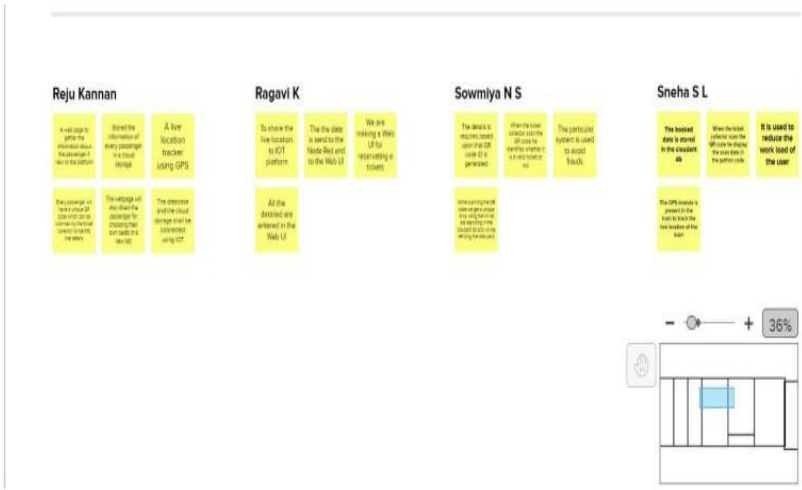
2.IDEATION PHASE

BRAIN STORM &IDEA PRIOTIZATION TEMPLATE:

Step-1:



Step-2: Brainstorm, Idea Listing and Grouping

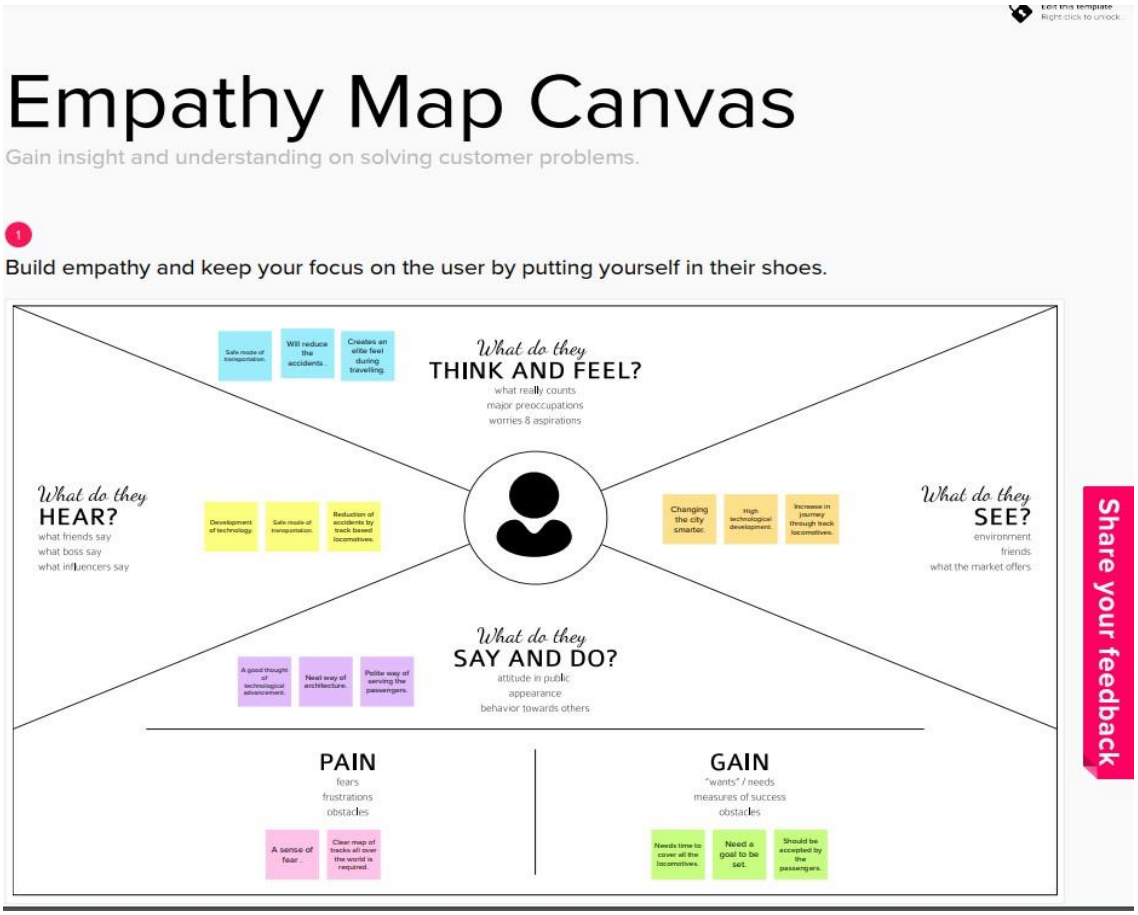


LITERATURE SURVEY:

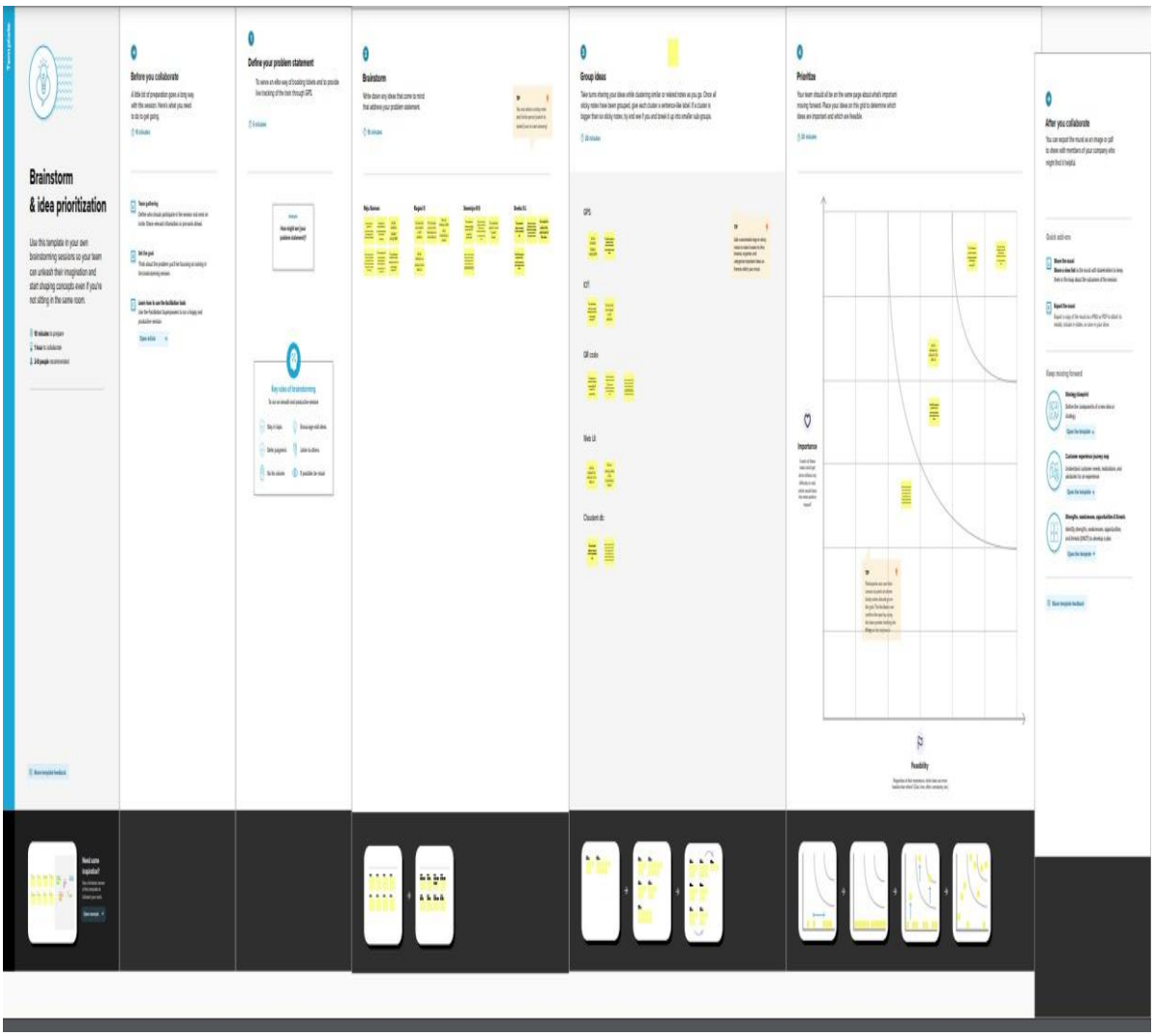
- Smart metro rail ticketing system-Year:2019,Nair,M Abishek and Tunk, Smit and Reddy, Panyam Gangadhar and Sultana, H Parveen
- Local train ticketing system using web services-Year:2022, Sheeja, Raghavan and Umaeswari, P and Bibin, Chidambaranathan and Nishanth, R and Chandana, S Hari
- Ticketing- booking patterns and policy implications-Year:2022,Xie,Jiemin and Zhan, Shuguang and Wong, SC and Wen, Keyu and Qiang, Lixia and Lo, SM

- 5G key technologies for smart railways – Year:2020,Ai,Bo and Molisch,Andreas F and Rupp, Markus and Zhong, Zhang dui
- The Internet of smart trains-YEAR:2017 Fraga-Lamas, Paula and Ferns, Tiago M and Castedo, Luis
- Smart ticketing system for railways in smart cities using software-YEAR:2017 D'silva, Godson Michael and Scariah, Anoop Kunjumon and Pannapara, Lukose Roy and Joseph
- Smart Train Accident Detection And Prevention System Using Iot Technology-Year:2021,Devi, R Lakshmi and Saravanan, G and Sangeetha, K and Pavithra,and Thiyagarajan.
- Enhanceme nt of Railways Resevation System using Intenet Of Things-YEAR:2018 Mallikarjuna, B and Reddy, D Arun Kumar and Sailaja, G
- Effectivenes s of the ETicket System Using QR Codes For Smart Transportati on Systems-YEAR:2021 Kuncara, Tommy and Putra, Arman Syah and Aisyah, Nurul and Valentino, VH

EMPATHY MAP:



IDEATION:



3.PROJECT DESIGN PHASE- I

PROPOSED SOLUTION:

- Problem Statement (Problem to be solved) :

  To reduce the workload of the user and also the use of paper.

- Idea / Solution description:

1. A Web page is designed for the public where they can book tickets by seeing the available seats.

2. After booking the train, the person will get a QR code which has to be shown to the Ticket Collector while boarding the train.

3. The ticket collectors can scan the QR code to identify the personal details.

4. A GPS module is present in the train to track it. The live status of the journey is updated in the Web app continuously.

5. All the booking details of the customers will be stored in the database with a unique ID and they can be retrieved back when the Ticket Collector scans the QR code.

- Novelty / Uniqueness :

  This project provides a live location tracking of the train to every passenger.

- Social Impact / Customer Satisfaction :

  This will bring an elite feel and this project will reduce the paperwork .It will bring in a choice for the passengers to choose their own seats and can keep a live track of the route of the train.

- Scalability of the Solution :

  This will take the technology level to the next stage and will reduce the hypertension of passengers during the last minute of booking.This can also bring in the awareness of trains being delayed well in advance due to live tracking.

PROBLEM SOLUTION FIT:

PROPOSED SOLUTION:

1. A Web page is designed for the public where they can book tickets by seeing the available seats.

2. After booking the train, the person will get a QR code which has to be shown to the ticket collector while boarding the train.

3.The tickets collectors can scan the QR code to identify the personal details.

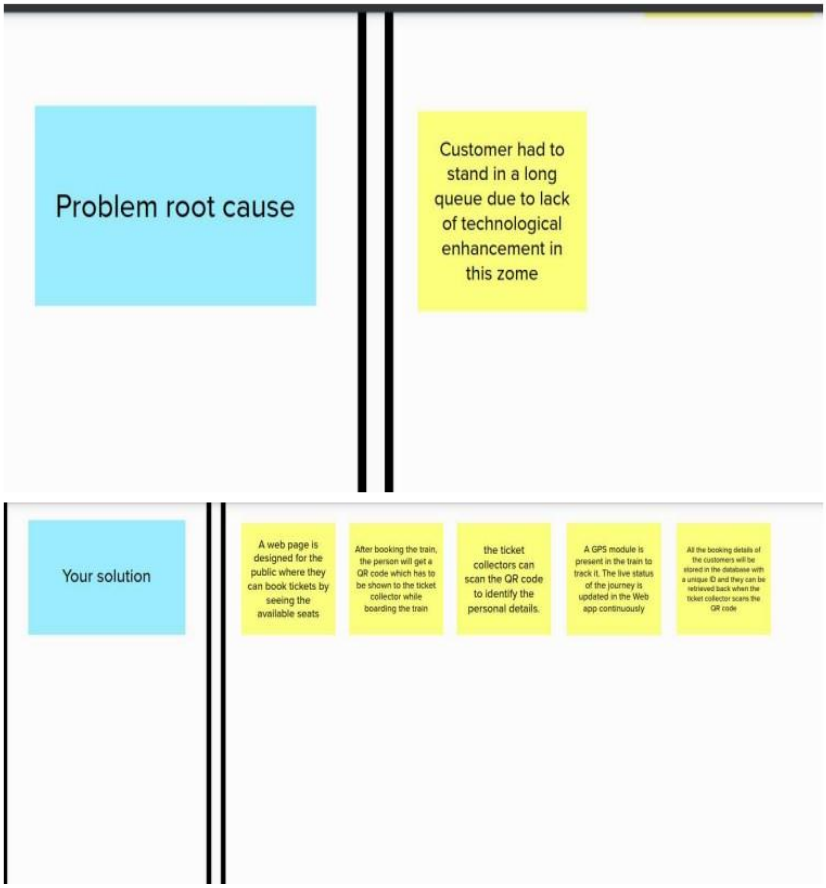
4.A GPS module is present in the train to track it. The live status of the journey is updated in the Web app continuously.

5.All the booking details of the customers will be stored in the database with a unique ID and they can be retrieved back when the Ticket Collector scans the QR

Code.

Solution fit:

Customer segment	The customer who can use this platform lies in the age group of 18 and above
Jobs to be done	To get themselves registered in the platform and then navigate to the booking platform
Triggers	Saving time than standing in long queue for ticket
Emotional Before/ After	irritated, wasting> time saving> easier and happy
Available solutions	The webpage will navigate the customer to the search brain which the destination could be entered and hence the tickets can easily be booked by clicking in the train that is listed down
Cistomer constrains	1) Budget 2) network connection 3) lack of planning for the trip
Behaviour	Ask opinion among friends and gather awards and get registered in the online booking platform
Channels and Behaviour	8.1,Online 1) Get registered 2) book tickets 3) Get confirmation mail often payment 8.2, Offline Pack the luggages if any and to take with them the adhar card and covid vaccine certificate in hand



SOLUTION ARCHITECTURE:

The customer interacts with the system through the front-end by making requests which are processed through the PHP, which is the middle tier. The system is executed on a central server and all clients communicate with it. A client handles customer interface while server handles function and operations of relevant components. All data is resident on the system server, which has the ability to interact with several clients at the same time by running several processes concurrently. Some of the processing undertaken includes verification, validations, manipulations, request processing.

Login:

After registration, the customer goes through the process of logging in else, access would be denied. The user logs in with a username and password. The system will verify the username and password to check its validity before the customer is granted entry upon the validity of the information provided, else the user is denied access to the system.

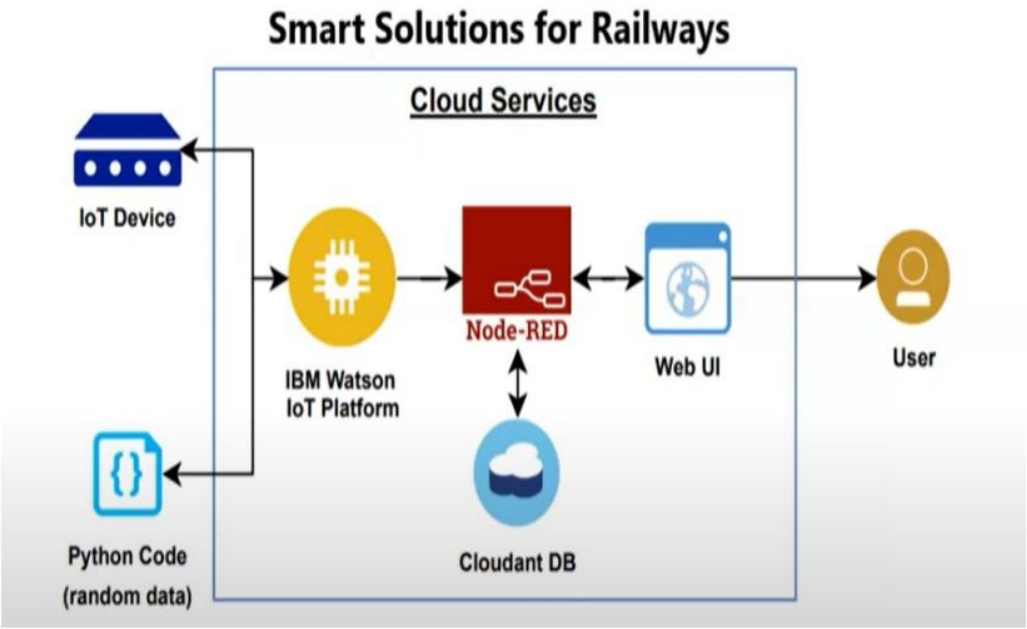
Booking:

The system offers a seat reservation system for the user where the user can choose the particular section or seat where he wants to sit after logging in. Payment for booking: in order to preserve reservation after booking, the customer pays.

Ticket print:

The customer prints the ticket after payment. Register: If a customer is new, he is mandated to register. If he wants to gain access to the system. The customer supplies some basic information in the registration page. This is mainly to know the number of customers using the system.

Solution Architecture Diagram:



4.PROJECT PHASE – II

CUSTOMER JOURNEY:

Proposed Solution:

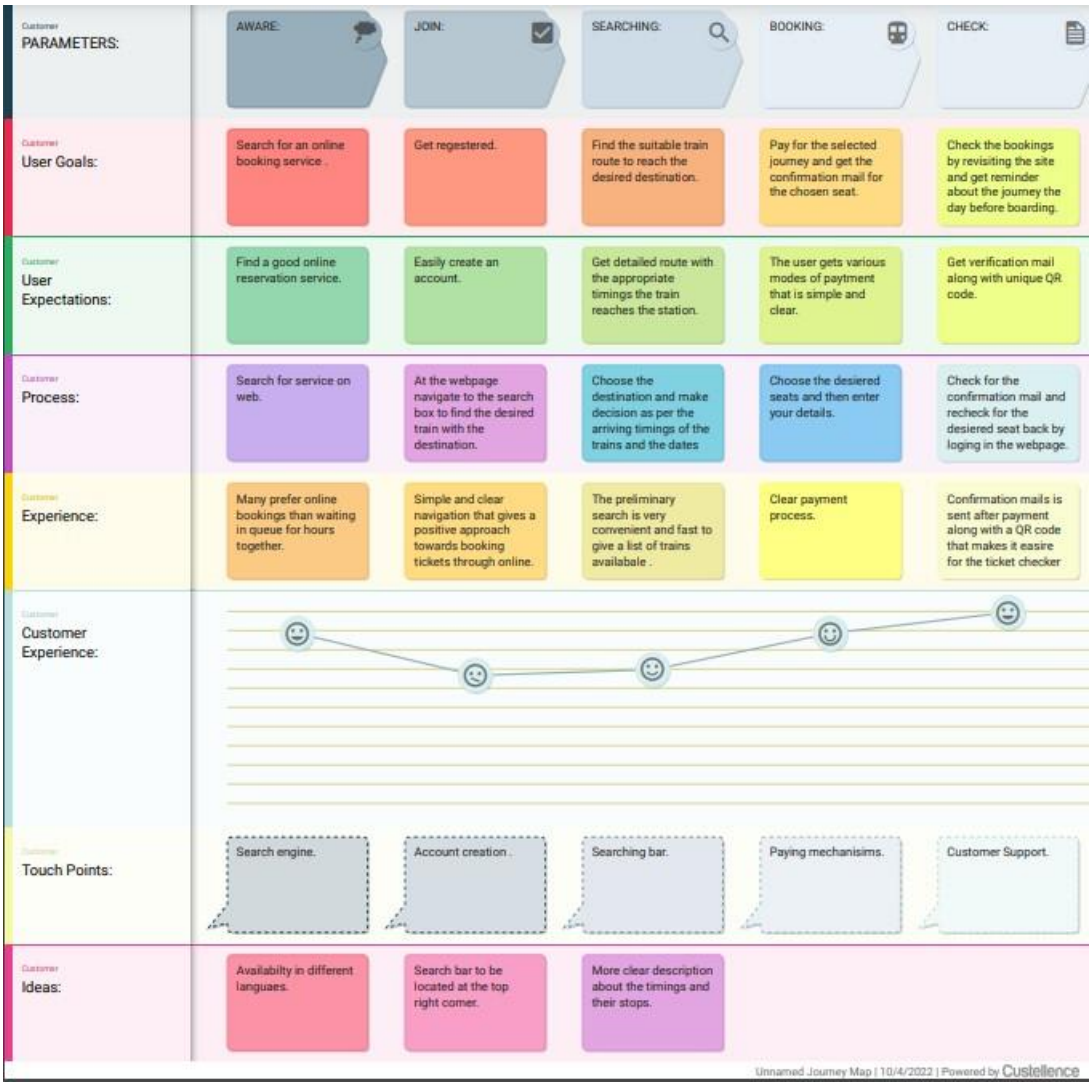
● A Web page is designed for the public where they can book tickets by seeing the available seats.

● After booking the train, the person will get a QR code which has to be shown to the Ticket Collector while boarding the train.

● The ticket collectors can scan the QR code to identify the personal details.

● A GPS module is present in the train to track it. The live status of the journey is updated in the Web app continuously. All the booking details of the customers will

be stored in the database with a unique ID and they can be retrieved back when the Ticket Collector scans the QR Code.



FUNCTIONAL REQUIREMENTS:

1.FR-1 User Registration: Registration through Form Registration through Gmail Registration through LinkedIN.

2.FR-2 User Confirmation : Confirmation via Email Confirmation via OTP.

3.FR-3 User Confirmation: Confirmation with QR code. FR-4 User Confirmation Gets a QR code through mail.

NON- FUNCTIONAL REQUIREMENTS:

1.NFR-1 Usability: You can book the tickets by entering and seeing the sheet available.

2.NFR-2 Security : The safe and secure as the train timing will not be delayed.

3.NFR-3 Reliability: It is highly reliable as no errors will take place.

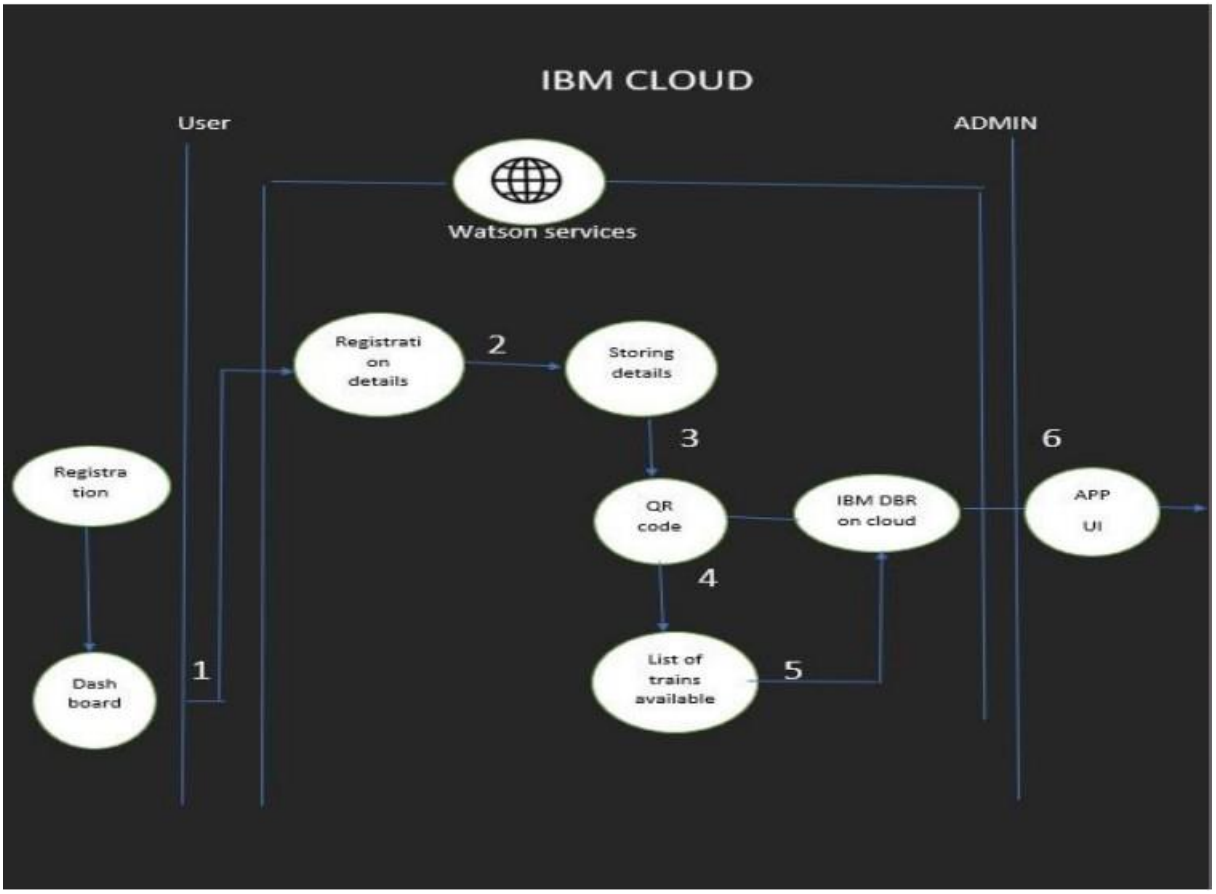
4.NFR-4 Performance : It is high performance and greater satisfaction to customers as they do not have to stand in the long queue for booking a tickets.

5.NFR-5 Availability : Highly available. NFR-6 Scalability High scalability

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 thr 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.

Example: (Simplified)



1. Login in to web page.

2. After Registering you will receive a OTP.

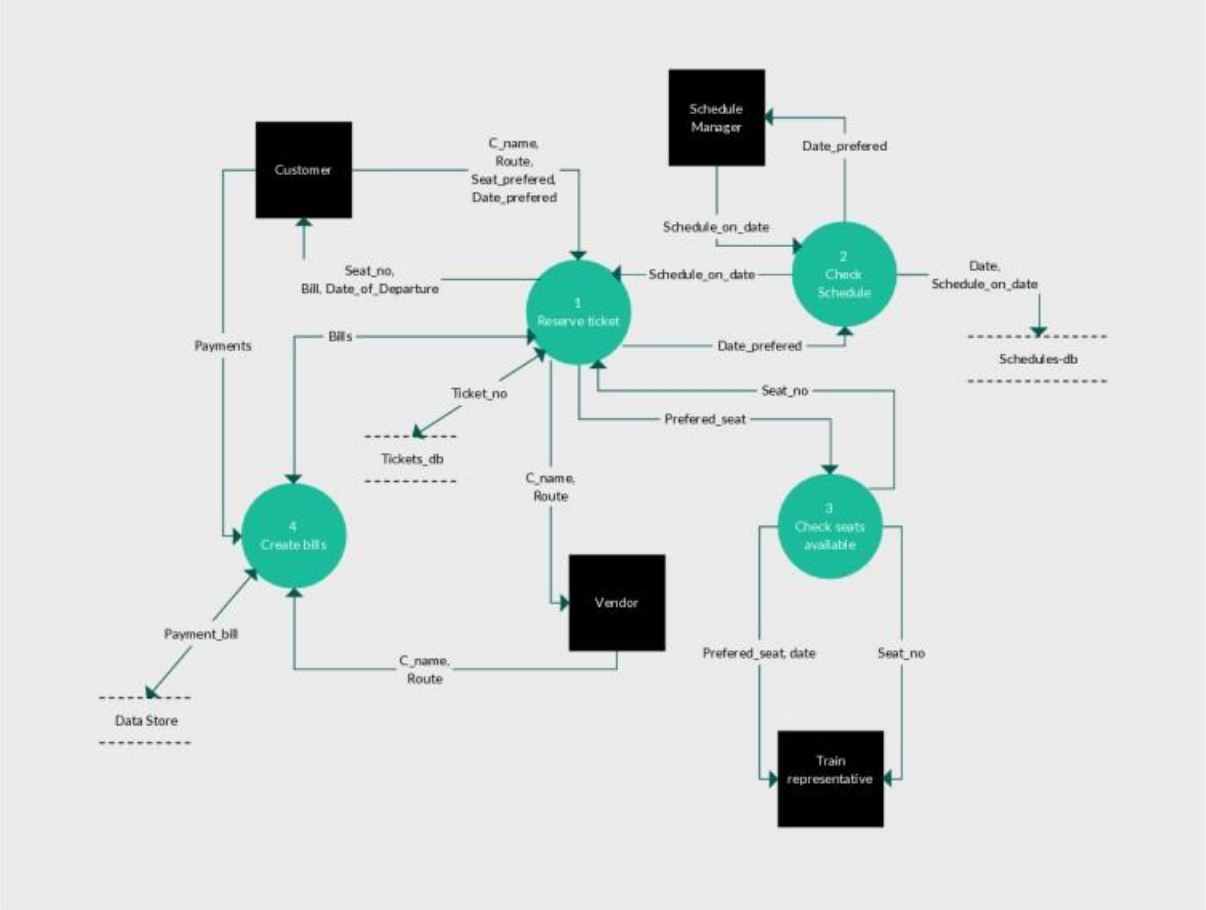
3. Next search the trin which you want in the search tab.

4. The list of tarins will be shown on basics of your searching.

5. After Selecting the train to tavelled you will be shown the list of seat available.

6. Now you can select your seating which is unreserved

Example: DFD (Industry Standard)



TECHNOLOGY ARCHITECTURE:

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table

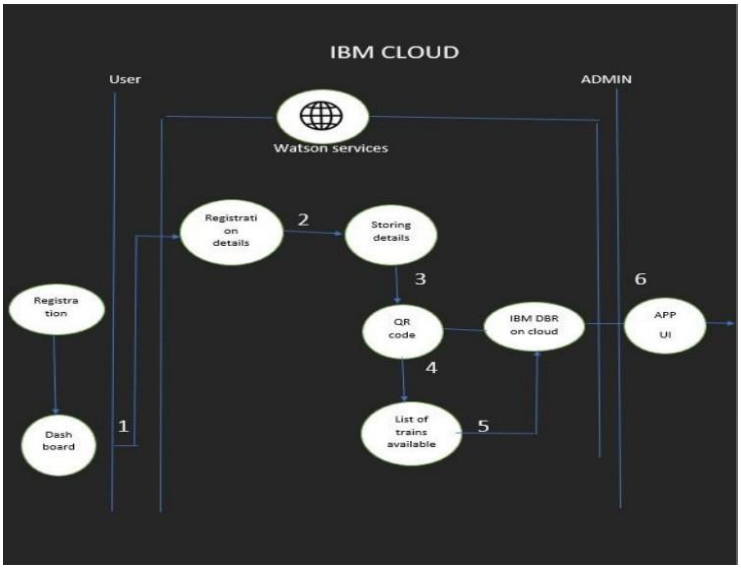


Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	Register your details	Web UI, Mobile app
2.	Application Logic-1	Login to the Web page	Java / Python
3.	Application Logic-2	Get registered	IBM Watson STT service
4.	Application Logic-3	Enter the OTP after registering	IBM Watson Assistant
5.	Database	Type your sex, current location etc...	MySQL, NoSQL, etc.
6.	Cloud Database	Type your sex, current location etc...	Adhaar card
7.	File Storage	Train timing with available trains	IBM Block Storage or Other Storage Service or Local Filesystem
8.	Machine Learning Model	It can the available train to the customer	Object Recognition Model, etc.

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Registration dashboard	IBM Cloud, data base, html
2.	Security Implementations	Security control of firewall	SHA-256, Encryptions, IAM Controls, OWASP etc.

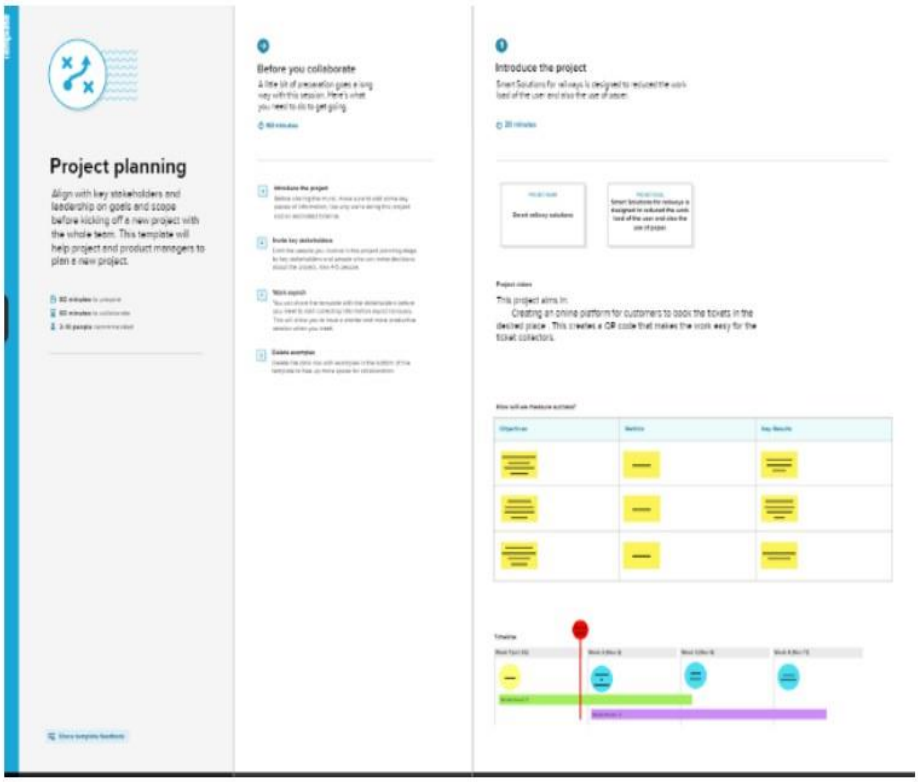
S.No	Characteristics	Description	Technology
3.	Scalable Architecture	The information of the customer and register goes hand-to-hand and hence it is highly scalable	App UI
4.	Availability	Available and data sheet of train and timing should be displayed	App UI
5.	Performance	It can use 50 request per second and it can provide 100 cache that is 1 cache is real and 1 cache is duplicate	App UI

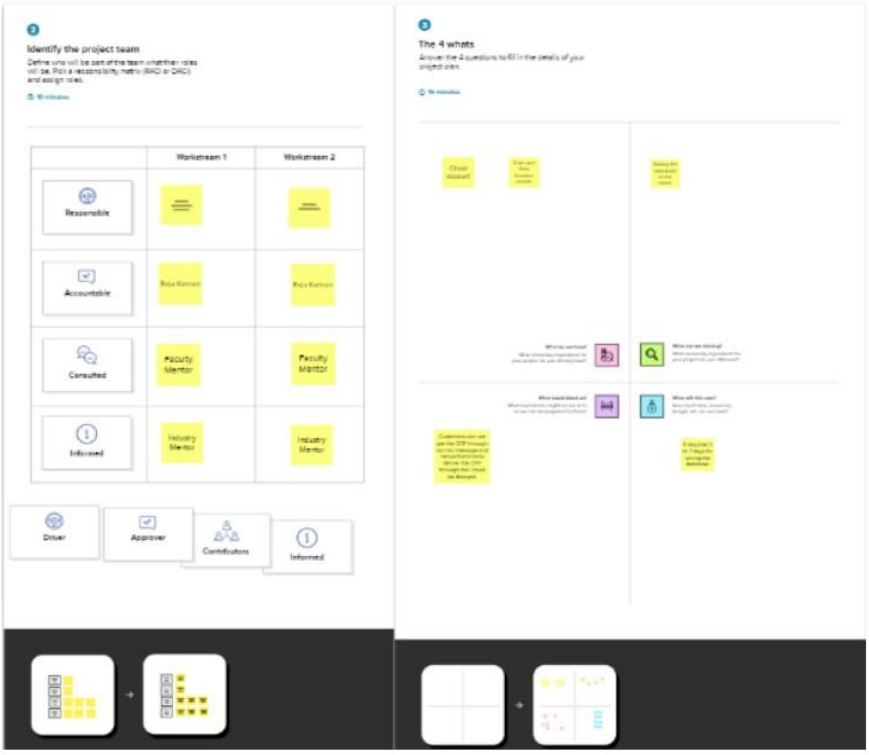
5. PROJECT PLANING PHASE

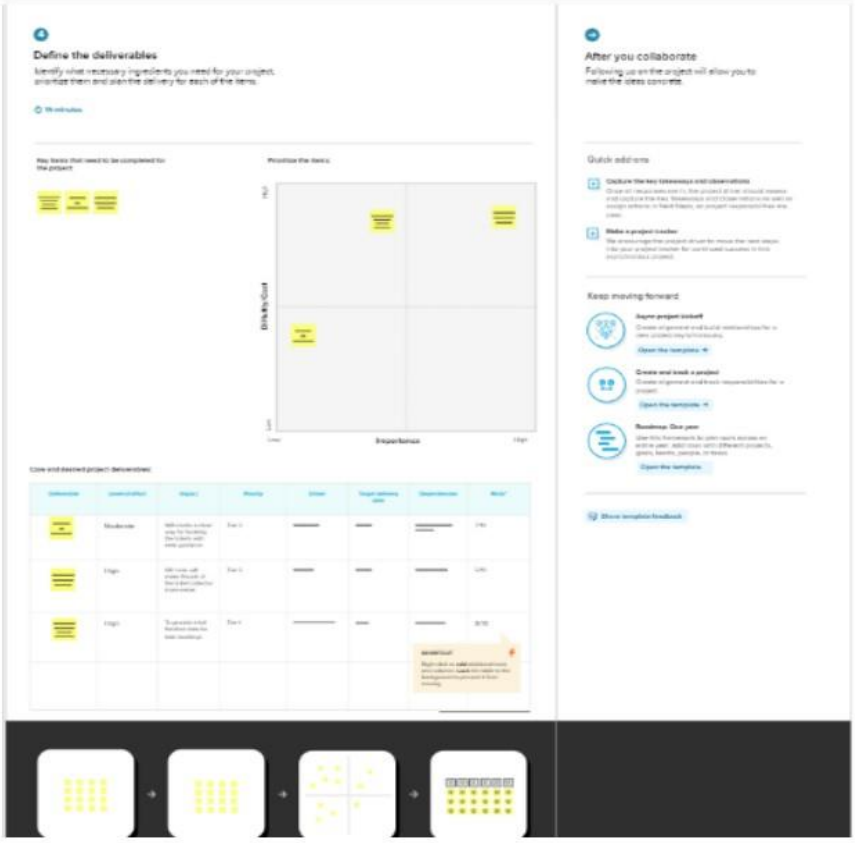
MILISTONE & ACTIVITY LIST:

Planning phase:

Project planning is a discipline addressing how to complete a project in a certain timeframe, usually with defined stages and designated resources.


Project planning
Align with key stakeholders and leadership on goals and scope before kicking off a new project with the whole team. This template will help project and product managers to plan a new project.
Before you collaborate
Introduce the project
Smart solutions for railways
This project aims in:
Creating an online platform for customers to book the tickets in the desired place. This creates a QR code that makes the work easy for the ticket collectors.


Identify the project team
Workstream 1
Workstream 2
Responsible
Accountable
Consulted
Informed
Faculty Mentor
Industry Mentor
Driver
Approver
Contributors
Informed
The 4 whats



SPRINT DELIVERY PLAN:

Sprint Schedule:

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High	Reju Kannan
Sprint-1	Registration	USN-2	As a user, I will receive confirmation email once I have registered for the application	1	High	
Sprint-1		USN-3	As a user, I can register for the application through Gmail	2	Medium	
Sprint- 2	Login	USN-4	As a user, I can log into the application by entering email & password	1	High	Ragavi K
Sprint- 3	Dashboard	USN-5	Rechecking	2	Medium	Sowmiya N S
Sprint- 4	Booking	USN-6	To book the desire train	1	High	Sneha S L

Project Tracker, Velocity & Burndown Chart:

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	31 Oct 2022	1 Nov 2022	20	02 Nov 2022
Sprint-2	20	6 Days	07 Oct 2022	05 Nov 2022	19	8 Nov 2022
Sprint-3	20	6 Days	14 Nov 2022	12 Nov 2022	19	12 Nov 2022
Sprint-4	20	6 Days	24 Nov 2022	14 Nov 2022	18	15 Nov 2022

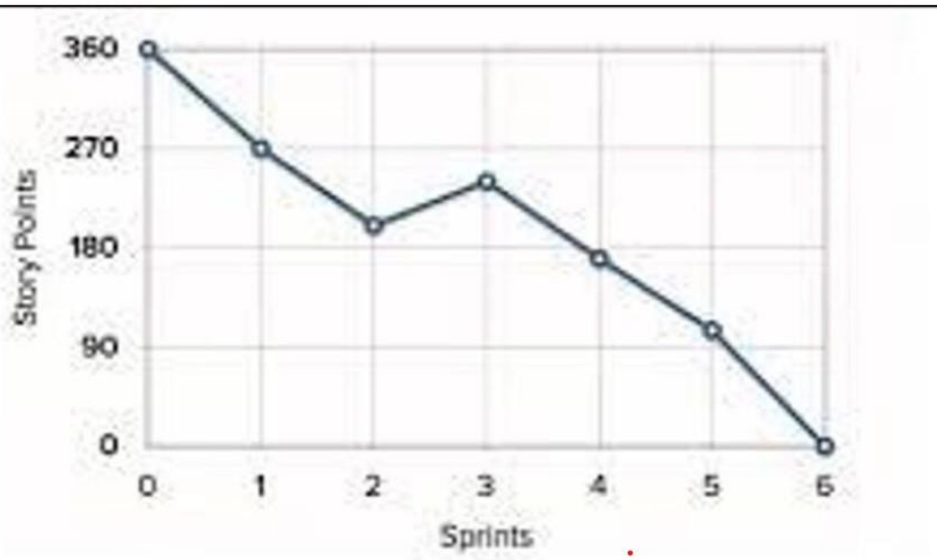
Velocity:

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day).

$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{10} = 2$$

Burndown Chart:

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.



Burndown Chart

6.PROJECT DEVELOPMENT PHASE

DELIVERY OF SPRINT-1:

PROCEDURE:

Step1: Develop node red application.

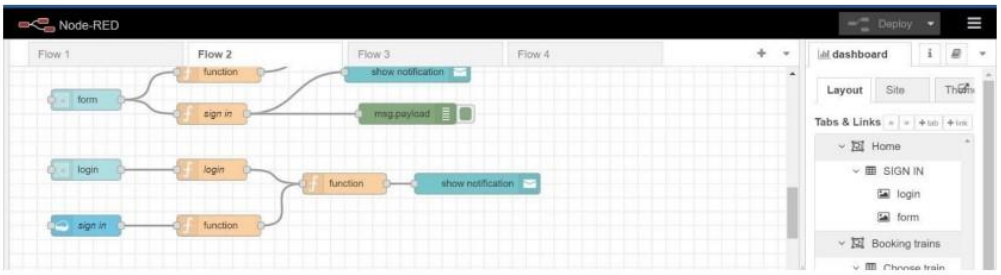
Step2: Install the required nodes from manage palette option.

Step3: Connect the node flow.

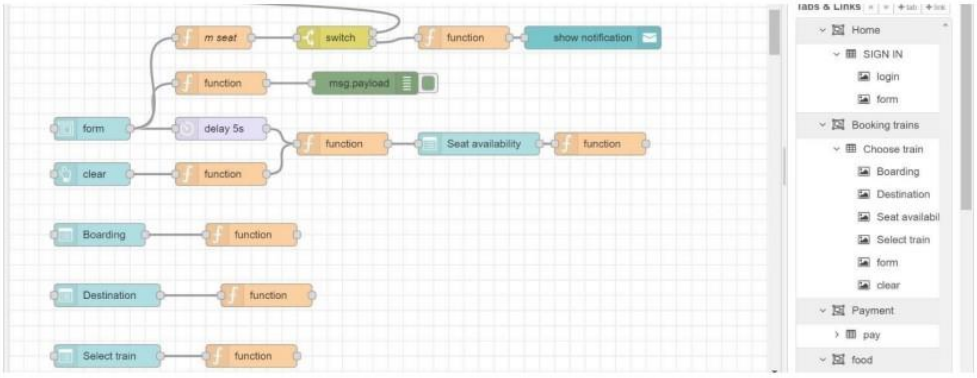
Step4: Deploy the flow.

WEB APPLICATION :

WEB APPLICATION :



NODES TO BOOK TRAIN:



FUNCTION NODE COMMAND TO INDICATE THE AVAILABLE SEATS:

```
var a=global.get('a') var
s= []
for(let i=0;i <a.length==0;i++){
s.push(a[i]) }
if(s.length==0){
   msg.options=[{"No seats available":0}]
}
else{
msg.options= s
}
msg.payload= s
return msg;
```

FUNCTION NODE COMMAND TO CHOOSE THE AVAILABLE SEATS:

```
var s=global.get('s') var
a=global.get('a') function
 reg(x){
for(let i=0;i<a.length;i++)[
if(a[i]==x){
            a.splice(i,1)
         }
     }
 }
if(s==1){
global.set('s1',s)
reg(s)
}
else if(s==2){
global.set('s2',s)
reg(s)
 }
elseif(s==3){
global.set('s3',s)
reg(s)
}
else if(s==4){
global.set('s4',s)
```

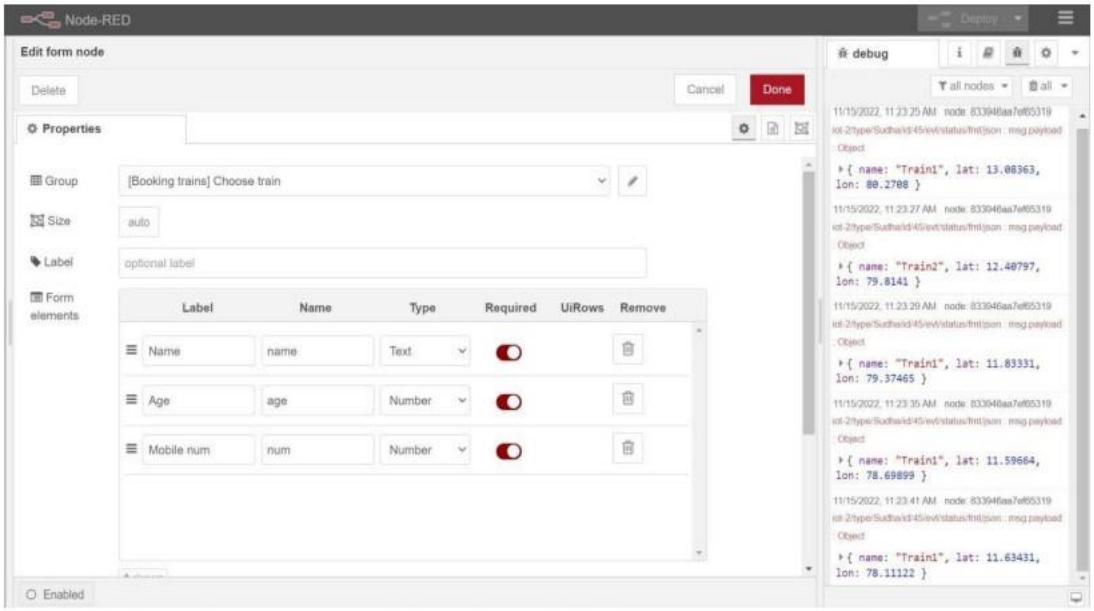
```
 reg(s)
 }
else if(s==4){
 global.set('s4',s)
reg(s)
}
return msg;
```

FUNCTION NODE COMMAND TO STORE DATA IN DATABASE:

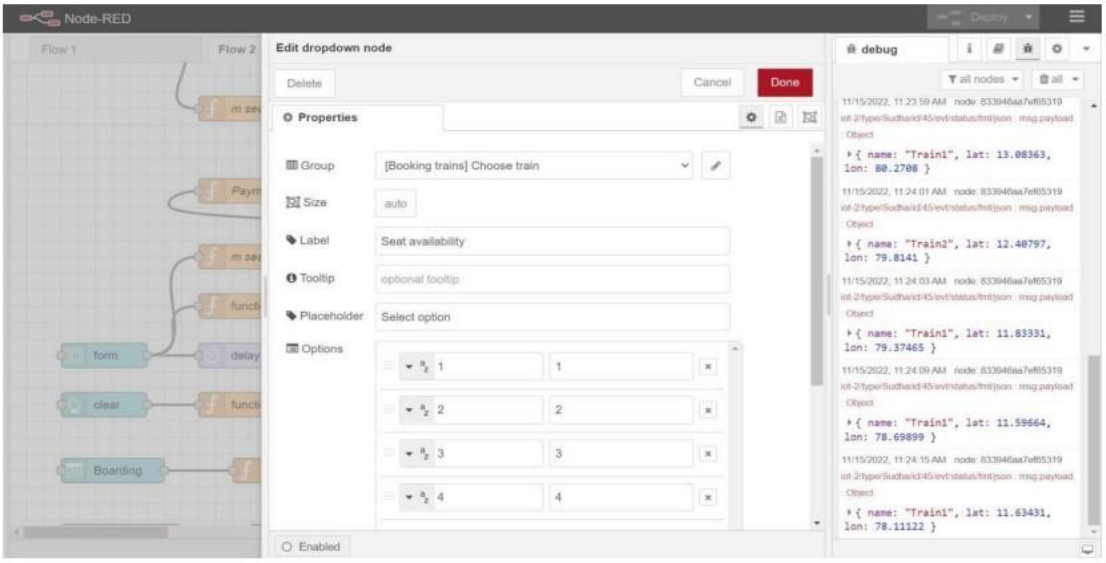
```
Var m=global.get('m')
 var d=new Date();
var utc=d.getTime()+(d.getTimezoneOffset()*60000);
var offset=5.5;
newDate=new Date(utc+(3600000*offset));
var n=newDate.toISOString() var
 date=n.slice(0,10) var time=n.slice(11,19)
var d1=date+','+time msg.payload={
    "_id":d1,
    "Name":m.Name,
 "Age":m.Age,
     "Mobile":m.Num,
    "boarding":global.get('b'),
   "destination":global.get('d'),
  "Seat":global.get('s')
}
```

return msg;

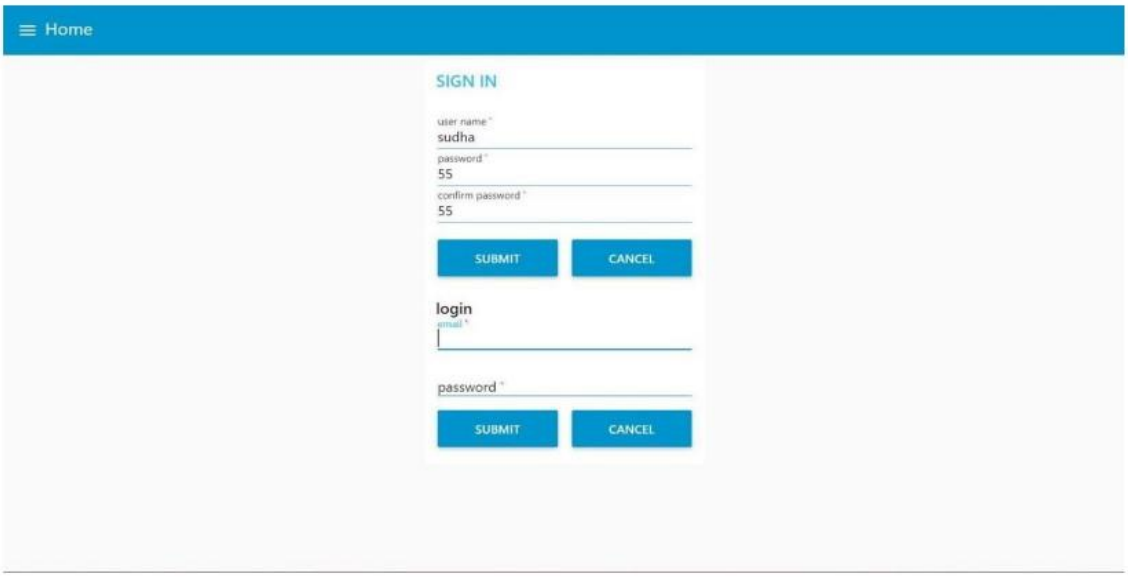
FORM DETAILS:



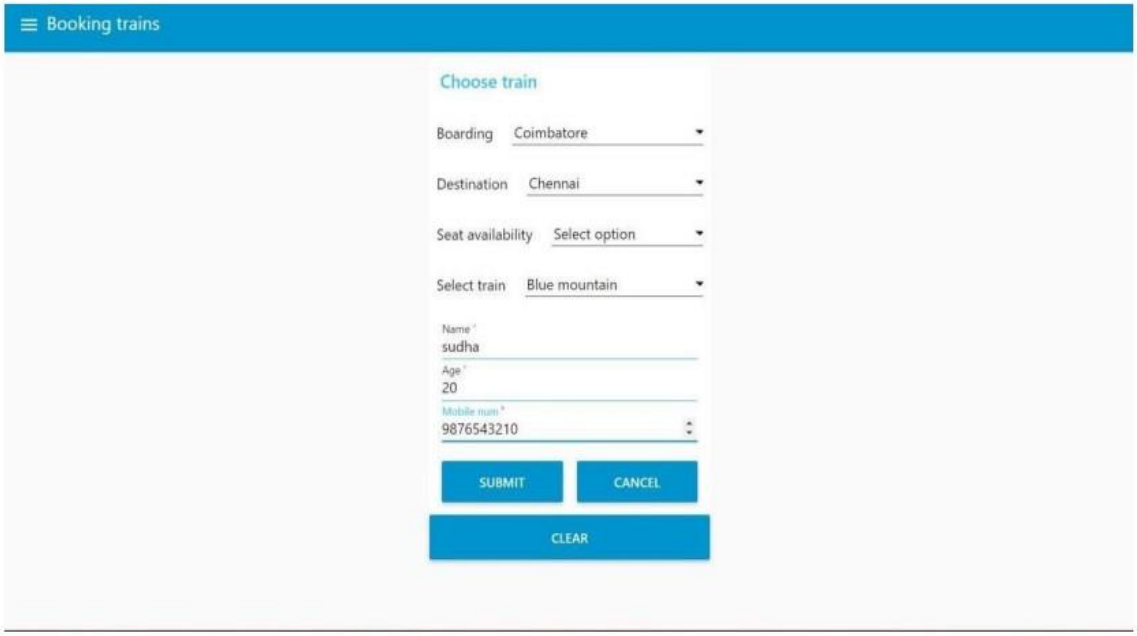
SEAT DROPDOWN BOX:



WEB UI OUTPUT:



WEB UI FOR TRAIN BOOKING:



DELIVERY OF SPRINT-2:

PROCEDURE:

Step1: Develop node red web application for train ticket booking

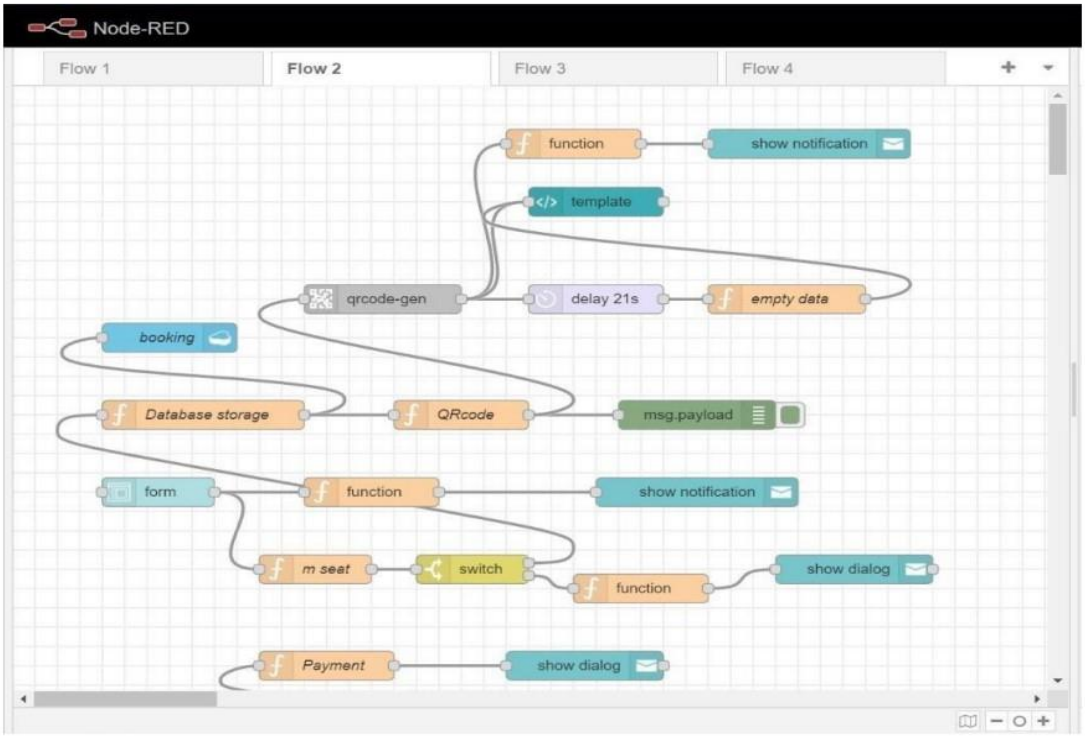
Step2: Copy the node red link and add /ui to the same link and browse it

Step3: Fill the details Step4: Click on submit

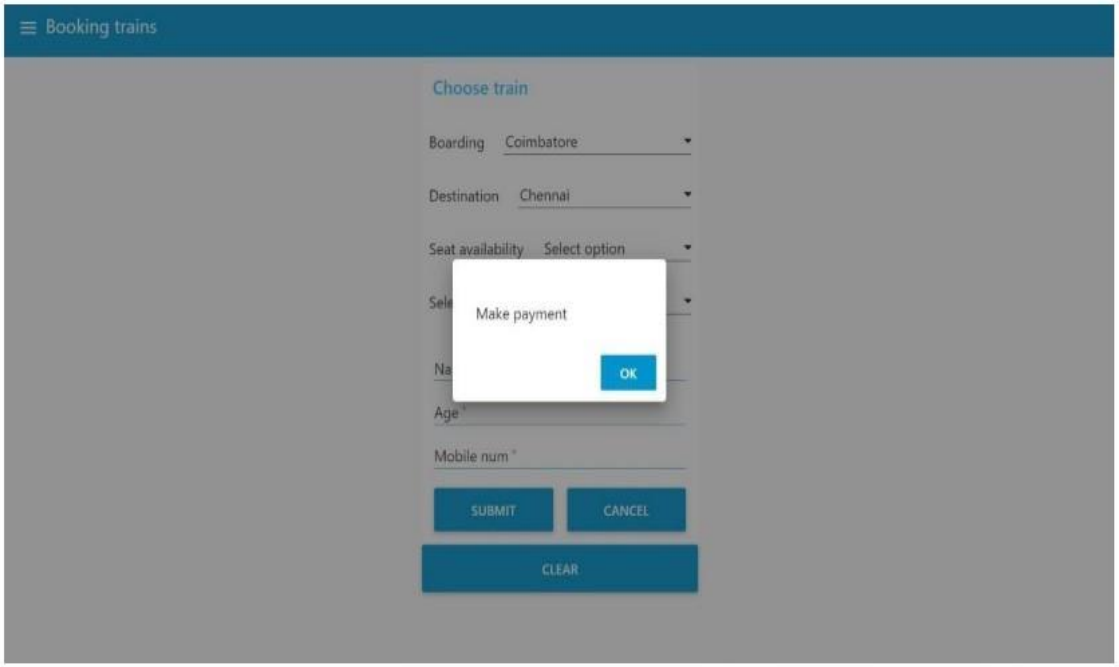
Step4: QR code will be submitted

Step5: Ticket is generated

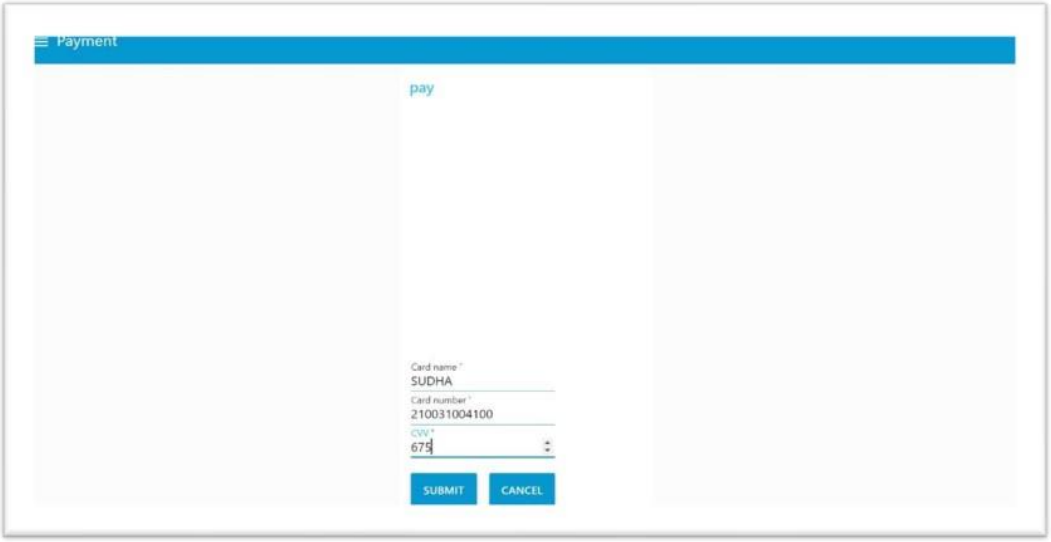
NODE RED FLOW CONNECTION:



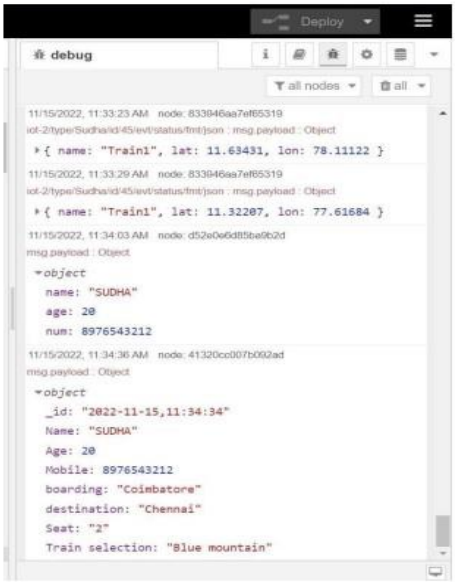
NODE RED FLOW FOR PAYMENT:



PAYMENT PAGE:



NODE RED OUTPUT:



DELIVERY OF SPRINT-3:

PROCEDURE:

Step1: Develop a python script to scan the QR code .

Step2: Connect the python code to IBM Cloudant using the credentials .

Step3: Run the program.

PYTHON SCRIPT TO SCAN QR CODE:

```
import cv2 import numpy as np import time import pyzbar.pyzbar

as pyzbar from pyzbar.pyzbar import decode from

ibmcloudant.cloudant_v1 import CloudantV1 from ibmcloudant

import CouchDbSessionAuthenticator from

ibm_cloud_sdk_core.authenticators import BasicAuthenticator

authenticator=BasicAuthenticator('apikey-v2-
125rwcp4ifi6zz2ly1cq0kakyjn98du2ysgc72h53lzi',
'af693938842290ec2c254461754447b5')                service                =
CloudantV1(authenticator=authenticator)

service.set_service_url('https://apikey-v2-
125rwcp4ifi6zz2ly1cq0kakyjn98du2ysgc72h53lzi:af693938842290ec2c2544617
54447b5@82d874994395-

4f46-a190-6a186bee5051-bluemix.cloudantnosqldb.appdomain.cloud')

cap= cv2.VideoCapture(0) font = cv2.FONT_HERSHEY_PLAIN

 while True:

_, frame = cap.read() decodedObjects = pyzbar.decode(frame)

for obj in decodedObjects:      #print ("Data", obj.data)

a=obj.data.decode('UTF-8')      cv2.putText(frame, "Ticket", (50, 50), font, 2, (255,
0, 0), 3)

#print (a)

try:

response = service.get_document(db='booking',doc_id = a).get_result()

print(response) time.sleep(5) except Exception as e:

print("NOT A VALID TICKER")

time.sleep(5)
```

```
cv2.imshow("Frame",frame)
if cv2.waitKey(1) & 0xFF ==ord('q'):
break
cap.release()
cv2.destroyAllWindows()
 client.disconnect()
```

QR CODE DETAILS:

```
*IDLE Shell 3.9.6*
File  Edit  Shell  Debug  Options  Window  Help
Python 3.9.6 (tags/v3.9.6:db3ff76, Jun 28 2021, 15:26:21) [MSC v.1929 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\Nishanth G\AppData\Local\Programs\Python\Python39\pythoncam.
py
{'_id': '2022-11-15,11:34:34', '_rev': '1-dc28604ce92b95da395e5e9a40018fef', 'Na
me': 'SUDHA', 'Age': 20, 'Mobile': 8976543212, 'boarding': 'Coimbatore', 'destin
ation': 'Chennai', 'Seat': '2', 'Train selection': 'Blue mountain'}
```

DATA STORED IN CLOUDANT:

```
booking > 2022-11-15,11:34:34
{}JSON
Save Changes   Cancel   Upload Attachment   Clone Document   Delete
1 {
2   "_id": "2022-11-15,11:34:34",
3   "_rev": "1-dc28604ce92b95da395e5e9a40018fef",
4   "Name": "SUDHA",
5   "Age": 20,
6   "Mobile": 8976543212,
7   "boarding": "Coimbatore",
8   "destination": "Chennai",
9   "Seat": "2",
10  "Train selection": "Blue mountain"
11 }
Log Out
```

DELIVERY OF SPRINT-4:

PROCEDURE:

Step1: Develop a node red application for GPS.

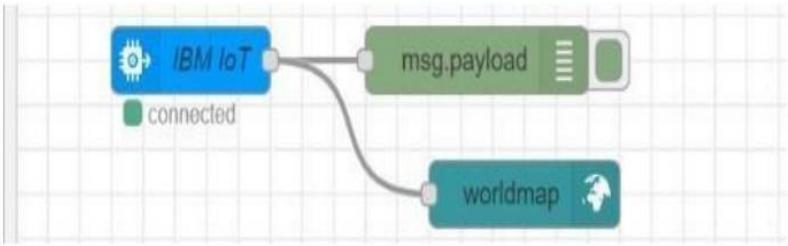
Step2: Develop a python code for GPS.

Step3: Run the program.

Step4: Train location will be displayed.

Step5: Create a node red for wakeup call and E-catering service.

NODE RED FLOW:



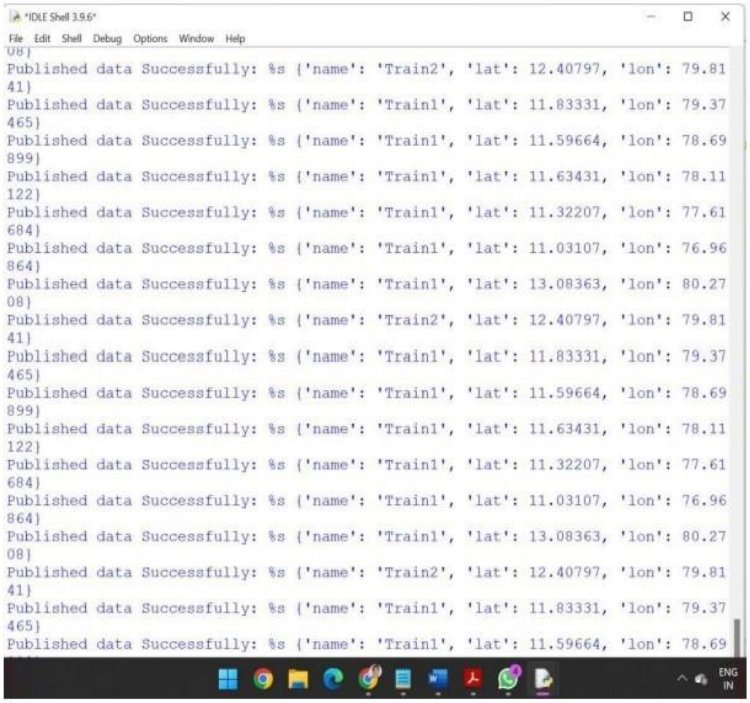
PYTHON CODE FOR GPS:

```
import wiotp.sdk.device
 import time import
random myConfig = {
        "identity": {
                "orgId": "dks66l",
                "typeId": "Sudha",
                "deviceId":"45"
        },
        "auth": {
```

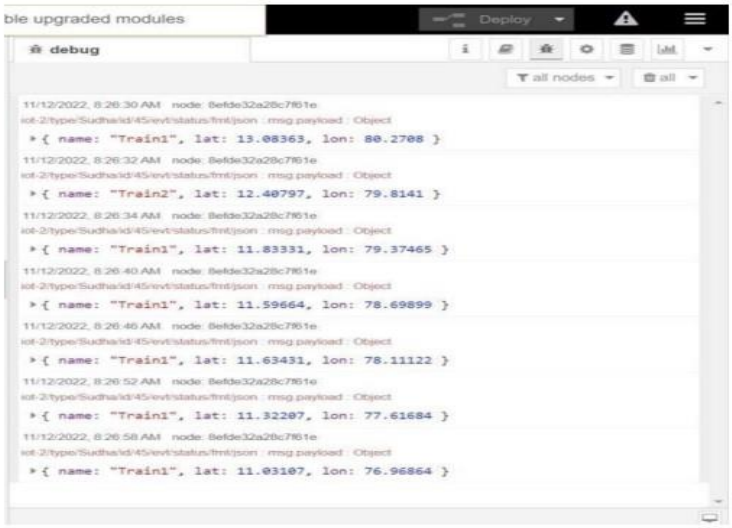
```
                "token": "sudha2002@"
        }
}
 def myCommandCallback (cmd):  print ("Message received from IBM IoT Platform:
%s" % cmd.data['command'])
        m=cmd.data['command']
 client = wiotp.sdk.device.DeviceClient(config=myConfig, logHandlers=None)
client.connect()
def pub (data):
        client.publishEvent(eventId="status",  msgFormat="json",  data=myData, qos=0, onPublish=None)
        print ("Published data Successfully: %s", myData)
while True:
        myData={'name': 'Train1', 'lat':13.08363 , 'lon': 80.27080}
        pub (myData)
        time.sleep (2)
        myData={'name': 'Train2', 'lat': 12.40797, 'lon': 79.81410}
        pub (myData)
        time.sleep (2)
        myData={'name': 'Train1', 'lat': 11.83331, 'lon': 79.37465}
        pub(myData)
        time.sleep(6)
        myData={'name': 'Train1', 'lat': 11.59664, 'lon': 78.69899}
        pub (myData)
        time.sleep (6)
        myData={'name': 'Train1', 'lat': 11.63431, 'lon': 78.11122}
```

```
pub(myData)
time.sleep (6)
myData={'name': 'Train1', 'lat': 11.32207, 'lon': 77.61684}
pub (myData)
time.sleep (6)
myData={'name': 'Train1', 'lat': 11.03107, 'lon': 76.96864}
pub (myData) time.sleep (6)
client.commandCallback = myCommandCallback
client.disconnect ()
```

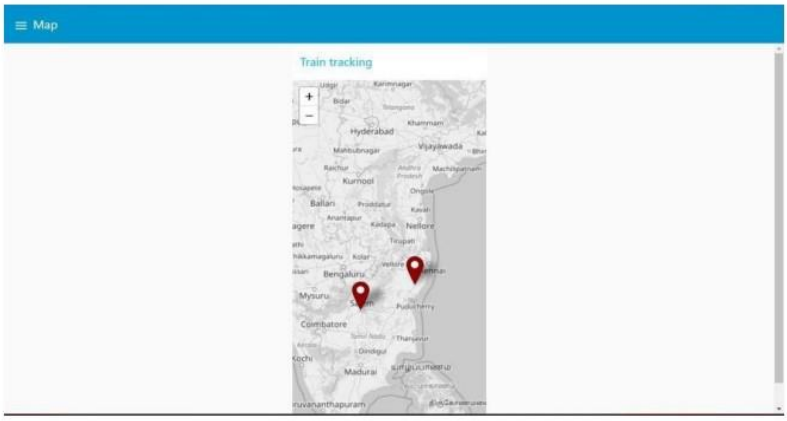
PYTHON CODE OUTPUT:



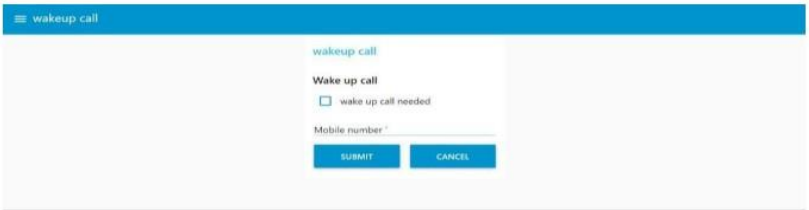
NODE RED OUTPUT:

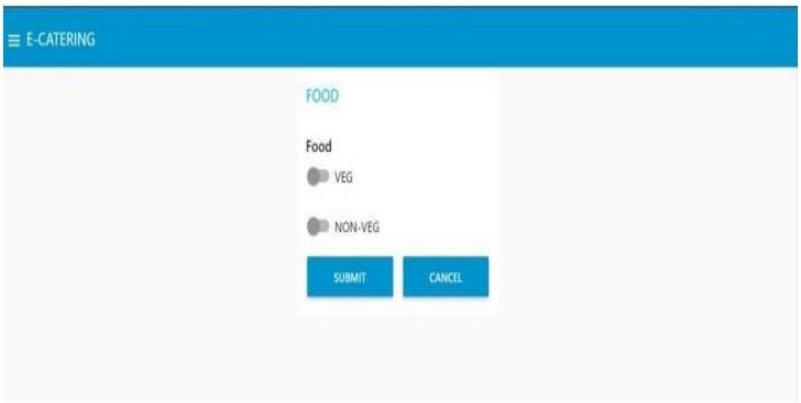


TRAIN TRACKING :



NODE RED CONNECTION FOR WAKEUP CALL AND E-CATERING SERVICE:

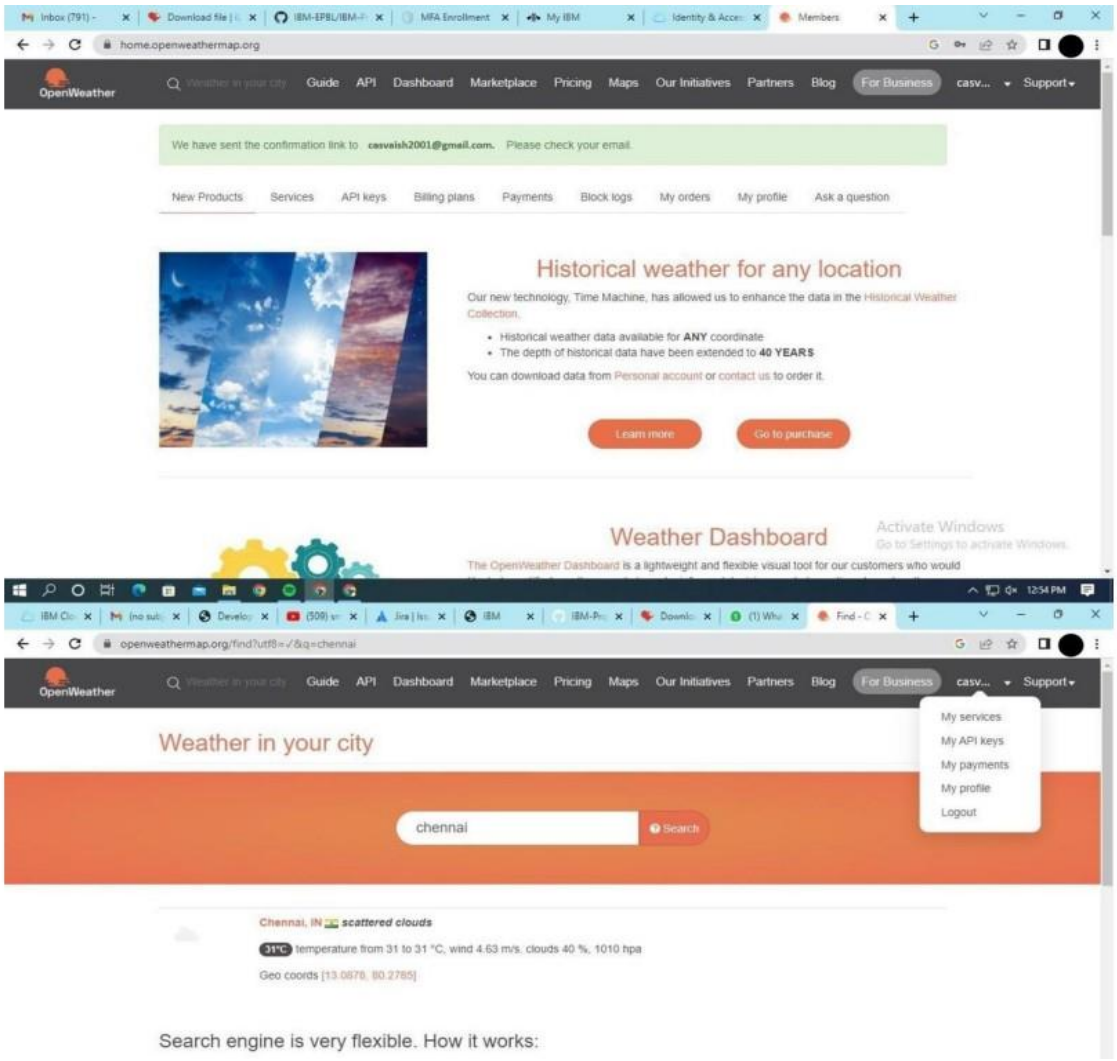




7.PYTHON SCRIPT

Create a code snippet using python to:

1. Extract weather data from Open Weather Map using APIs.
2. Send the extracted data to the cloud.
3. Receive data from the cloud and view it in the python compiler.



```
import requests
a = "https://api.openweathermap.org/data/2.5/weather?q=Chennai,IN&appid=6d13d12f9cd34a07871a5795d01e2c47"
r = requests.get(url = a)
data = r.json()
print(r)
print(data)
temp = data["main"]["temp"]
hum = data["main"]["humidity"]
print("Temperature is : ",temp)
print("Humidity is : ",hum)
```

```
<Response [200]>
Temperature is :  298.14
>>>
==================== RESTART: E:/IBM/pre/weatherMap.py ================
====
<Response [200]>
{'coord': {'lon': 80.2785, 'lat': 13.0878}, 'weather': [{'id': 701, 'main': 'Mist', 'description':
'mist', 'icon': '50n'}, {'id': 500, 'main': 'Rain', 'description': 'light rain', 'icon': '10n'}], 'bas
e': 'stations', 'main': {'temp': 298.14, 'feels_like': 299.15, 'temp_min': 298.14, 'temp_ma
x': 298.14, 'pressure': 1012, 'humidity': 94}, 'visibility': 2500, 'wind': {'speed': 1.54, 'deg
': 350}, 'rain': {'1h': 0.12}, 'clouds': {'all': 75}, 'dt': 1667317416, 'sys': {'type': 1, 'id': 9218,
'country': 'IN', 'sunrise': 1667262751, 'sunset': 1667304738}, 'timezone': 19800, 'id': 1
264527, 'name': 'Chennai', 'cod': 200}
Temperature is :  298.14
Humidity is :  94
>>>
```