

# **Project Report**

## **SMART SOLUTIONS FOR RAILWAYS**

**Team ID: PNT2022TMID04222**

**Team: Chandrabrabhu C - Team Lead**

**Dinesh Kumar S**

**Mia Marriane**

**Divya V**

**Faculty Mentor: D.Thamizhselvi - Assistant professor**

**Industry Mentor: Mr. BARADWAJ**

# **1. INTRODUCTION**

## **1.1 Project Overview**

Trains are one of the most popular modes of transportation among the middle and lower classes due to their convenience. Simultaneously, the chance of thefts and disasters such as chain snatching, derailment, and fire is increasing. We designed a solution in the form of an application that consumers may use after purchasing their tickets to avoid or, more properly, to halt all such violence. This programme addresses problems by sending an alarm text message to TC and RPF with a single click. In our project, we employ the Node-Red service, app development, and the IBM cloud infrastructure to store passenger data.

## **1.2 Purpose**

The purpose of this project is to report and get relieved from the issues related to trains and make a hasslefree journey for the passengers.

# **2. LITERATURE SURVEY**

## **2.1 Existing problem**

- 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.

## 2.2References

Title	Author	Source	Year	Findings
Problems of Indian Railways	N. Benjamin	Taylor & Francis group	2021	Common problems in Indian Railway system
Development of Railway Signaling System based on Network Technology	Y. Hirano Takashi Kato T. Kuni <u>fuji</u>	IEEE	2006	Optimal way of transmitting the signals in railway system using latest technology
Automatic Fault Detection of Railway Track System Based on PLC	Naveen Bhargav Avanish Gupta Mayank Khirwar	International Journal of Recent Research Aspects ISSN	2016	Train cracks and the obstacles in front are detected using the sensors
Ticketing Solutions for Indian Railways Using RFID Technology	Venugopal Prasanth Hari Prasad R K.P. Soman	IEEE	2010	Technology for providing tickets in railways

## 2.3 Problem Statement

Railway transport is one of the cheapest, fastest, convenient and biggest mode of transport in India. Though most of the customers are satisfied by the service provided by the Indian railways, few of them face some challenges.

Problem Statement	I am	I'm trying to	But	Because	Which makes me feel
1	Passenger	Travel long distance	Cannot get preferred seat allotment	Seat availability is not known	unhappy
2	Passenger	Board the train	Cannot find the arrival time of the train	Of network issues	Frustrated

### 3. IDEATION & PROPOSED SOLUTION

#### 3.1 Empathy Map Canvas



## 3.2 Customer Journey



### 3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	<ul style="list-style-type: none"><li>➤ Unable to reserve a preferred seat, and seat availability is unknown.</li><li>➤ The train's arrival time cannot be determined due to network issues.</li></ul>
2.	Idea / Solution description	<ul style="list-style-type: none"><li>➤ Paperless tickets using qr code are generated to reduce the cost of ticket generation and to provide the tickets instantly.</li><li>➤ Live location of the train and the delay in train timings can be found in the application</li><li>➤ Seat availability of the trains and the seat preference can be managed using a simple UI.</li></ul>
3.	Novelty / Uniqueness	<ul style="list-style-type: none"><li>➤ User friendly application with more features</li><li>➤ Better security</li></ul>
4.	Social Impact / Customer Satisfaction	<ul style="list-style-type: none"><li>➤ Since the tickets are paperless the passengers need not worry about the tickets</li><li>➤ Tickets can be easily verified using the valid id proof of the customer</li></ul>
5.	Business Model (Revenue Model)	<ul style="list-style-type: none"><li>➤ The cost for generating the tickets is reduced.</li><li>➤ Revenue can also be generated using food and snacks delivery in trains.</li></ul>

6.	Scalability of the Solution	<ul style="list-style-type: none"> <li>➤ Managing the booking history and the journey history can be done easily using machine learning.</li> <li>➤ Food and snacks can be ordered using our application in future.</li> </ul>
----	-----------------------------	--

### 3.4 Problem Solution fit

Define CS, fit into CL	<b>1. CUSTOMER SEGMENT(S)</b> <span>CS</span> <ul style="list-style-type: none"><li>➤ Passengers</li><li>➤ Ticket Checkers</li></ul>	<b>6. CUSTOMER LIMITATIONS</b> <span>CL</span> <small>EG. BUDGET, DEVICES</small> <ul style="list-style-type: none"><li>➤ Minimising the paperworks for customers</li></ul>	<b>5. AVAILABLE SOLUTIONS</b> <span>AS</span> <small>PLUSES &amp; MINUSES</small> <ul style="list-style-type: none"><li>➤ Passengers could take a monthly pass</li><li>➤ They can take the tickets in alternate stations</li></ul>	Explore AS, differentiate
	<b>2. PROBLEMS / PAINS + ITS FREQUENCY</b> <span>PR</span> <ul style="list-style-type: none"><li>➤ Passengers spend lot of time in queue for booking the ticket</li><li>➤ The seat availability of the train are not known</li><li>➤ The passengers cannot find the location of the train easily</li></ul>	<b>9. PROBLEM ROOT / CAUSE</b> <span>RC</span> <ul style="list-style-type: none"><li>➤ Passengers are having difficulty purchasing tickets and tracking the location of the train.</li></ul>	<b>7. BEHAVIOR + ITS INTENSITY</b> <span>BE</span> <ul style="list-style-type: none"><li>➤ We can convey true empathy for the situation by listening to the customer.</li></ul>	Focus on PR, tap into BE, understand RC
Identify strong TR & EM	<b>3. TRIGGERS TO ACT</b> <span>TR</span> <ul style="list-style-type: none"><li>➤ Passengers should wait for a long time to book the tickets.</li><li>➤ Passengers cannot get the preferred seat location</li></ul>	<b>10. YOUR SOLUTION</b> <span>SL</span> <ul style="list-style-type: none"><li>➤ A web app where the user can book their tickets easily</li><li>➤ The train location can be found on the web app</li><li>➤ TTE's can verify the tickets easily</li></ul>	<b>8. CHANNELS of BEHAVIOR</b> <span>CH</span> <div>ONLINE<ul style="list-style-type: none"><li>➤ Passengers can book their tickets online and receive a QR code via SMS.</li></ul></div> <div>OFFLINE<ul style="list-style-type: none"><li>➤ Passenger information is maintained in the web application, and the ticket collector can access it at any time.</li></ul></div>	Extract online & offline CH of BE
	<b>4. EMOTIONS</b> <span>EM</span> <small>BEFORE / AFTER</small> <ul style="list-style-type: none"><li>➤ Booking the tickets and the verification can be done easily which saves more time</li></ul>			



## 4. REQUIREMENT ANALYSIS

### 4.1 Functional requirement

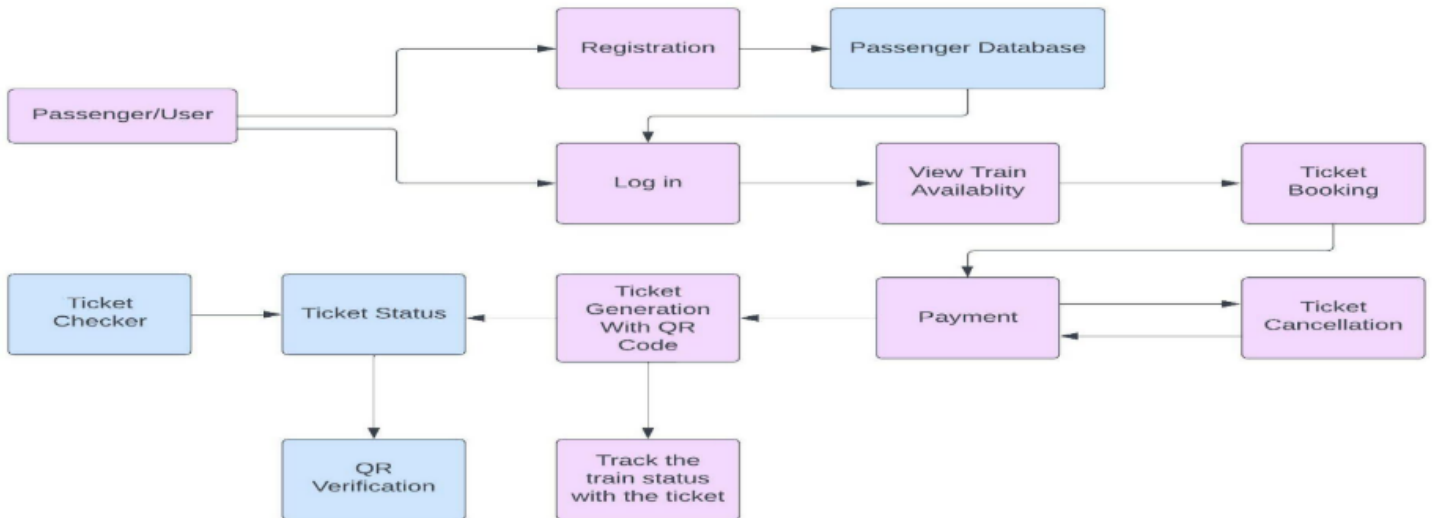
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through online web portal
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Ticket Booking	Availability of seats payment interface  Ticket generation with unique id  QR code generation
FR-4	Ticket Validation	Login page for TTE Scanning the qr code Validation of the ticket
FR-5	Train location tracking	Fetching the location of the train using gps  Sending the data to the end point

## 4.2 Non-Functional requirement

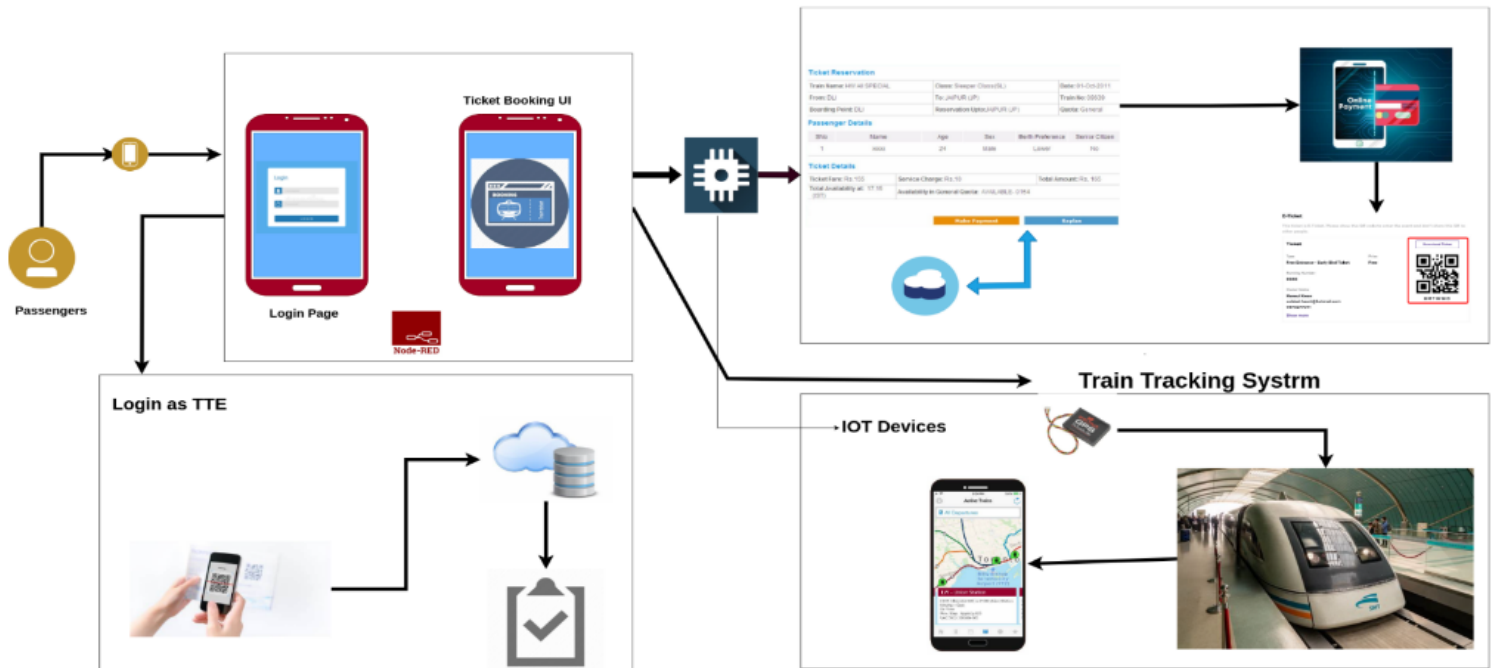
FR No.	Non-Functional Requirement	Description
NFR-1	Usability	The tickets can be booked easily using the web app by checking the seat availability, The TTE should be able to validate the tickets by scanning the qr code
NFR-2	Security	The application should be more secure by using otp confirmation and secure methods for communication
NFR-3	Reliability	The train location displayed in the web app should be accurate and the application should accept all kinds of payment methods.
NFR-4	Performance	Though many number of users try to book the ticket concurrently the server should be able to server all the users without crashing
NFR-5	Availability	The ticket booking should have almost 100% uptime for the user to book the tickets without any issues and distribute the live status of the train
NFR-6	Scalability	The system should be able to process the location of all the train every second

## 5. PROJECT DESIGN

### 5.1 Data Flow Diagrams



### 5.2 Solution Architecture



### 5.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
PASSENGER (Mobile user)	Booking registration	USN-1	As a passenger, I book the ticket for the journey by entering my personal information.	I can access the web link to install the application.	High	Sprint-1
	Confirmation	USN-2	As a passenger, I will receive confirmation of the booking once I have registered for the application	I can receive confirmation email & click confirm.	High	Sprint-1
	Application registration	USN-3	As a passenger, I can register for the application through the weblink.	I can register & access the application through google login.	Low	Sprint-2
	Application access	USN-4	As a passenger, I can access the application during my travel for resolving my issues.		Medium	Sprint-1

## 6. PROJECT PLANNING & SCHEDULING

### 6.1 Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I may register by filling out the form with my personal details.	1	High	Chandraprabhu C

Sprint-1		USN-2	As a user, I can register through phone numbers, Gmail, Facebook or other social sites	1	High	Dinesh Kumar S
Sprint-1	Confirmation	USN-3	As a user, If my registration is successful, I will receive a confirmation email or an OTP to my mobile number.	2	Low	Mia Marianne
Sprint-1	login	USN-4	As a user, I can login via login id and password or through OTP .	2	Medium	Divya M
Sprint-1	Display Train details	USN-5	As a user, I can enter a start and destination to get a list of trains that connect the above.	1	High	Divya M
Sprint-2	Booking	USN-6	As a user, I may supply basic information such as a name, age, gender, and so on.	2	High	Chandraprabhu C

Sprint-2		USN-7	<p>As a user, I have the option of selecting the class and seat/berth. If my preferred seat/berth</p> <p>becomes unavailable, I will be assigned based on availability.</p>	1	Low	Chandrababhu C
Sprint-2	Payment	USN-8	<p>As a user, I have the option of paying using a</p> <p>credit card, a debit card, or a UPI.</p>	1	High	Mia Marianne

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-2		USN-9	As a user, I will be forwarded to the chosen	2	High	Mia Marianne
Sprint-3	Ticket generation	USN-10	<p>As a user, I can download the generated</p> <p>e-ticket for my trip, as well as the QR code that will be used for authentication during my trip.</p>	1	High	Dinesh Kumar S

<b>Sprint-3</b>	<b>Ticket status</b>	<b>USN-11</b>	As a user, I am able to view the status of my ticket Whether affirmed, pending, or RAC.	<b>2</b>	<b>High</b>	<b>Dinesh Kumar S</b>
<b>Sprint-3</b>	<b>Reminders notification</b>	<b>USN-12</b>	As a user, I'm getting leftovers from my trip. A day before my departure.	<b>1</b>	<b>High</b>	<b>Divya M</b>
<b>Sprint-3</b>	<b>Ticket cancellation</b>	<b>USN-13</b>	As a user, I can use GPS to monitor the train and obtain information such as the ETA, current stop, and delay.	<b>2</b>	<b>High</b>	<b>Chandrabrabhu C</b>
<b>Sprint-4</b>		<b>USN-14</b>	As a user, I can use GPS to monitor the train and obtain information such as the ETA, current stop, and delay.	<b>1</b>	<b>High</b>	<b>Chandrabrabhu C</b>
<b>Sprint-4</b>	<b>Raise queries</b>	<b>USN-15</b>	As a user,I can send queries via email or the query box.	<b>2</b>	<b>Medium</b>	<b>Mia Marianne</b>
<b>Sprint-4</b>	<b>Answer the queries</b>	<b>USN-16</b>	As a customer care, I will respond to the customers' questions and concerns.	<b>2</b>	<b>High</b>	<b>Divya M</b>

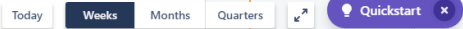
Sprint-4	Feed details	USN-17	As a user, If a new compartment is added, I will input information on train delays and add more seats..	1	High	Dinesh Kumar S
----------	--------------	--------	---	---	------	----------------

### 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	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	5 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov2022



Give feedback Share Export ...

[View settings](#)

3

•

Insights

0 0 0 Create sprint

Smart Solutions for Ra...

Software project

- PLANNING
- Roadmap

Backlog

Board
- DEVELOPMENT
- Code
- Project pages

Add shortcut




Project settings

Projects / Smart Solutions for Railways

SSFR Sprint 1

 Epic

TO DO 1 ISSUE	IN PROGRESS 1 ISSUE	DONE 2 ISSUES
<div>Sprint-4</div> <div>SSFR-8</div>	<div>Sprint-3</div> <div>SSFR-7</div>	<div>Sprint-1</div> <div>SSFR-5</div> <div>Sprint-2</div> <div>SSFR-6</div>

 10 days remaining

Complete sprint

GROUP BY: None

Insights

You're in a team-managed project

Learn more

Quickstart

## **7. CODING & SOLUTIONING**

### **7.1 Feature 1**

- . IoT device
- i. IBM Watson Platform
- ii. Node red
- iii. Cloudant DB
- iv. Web UI
- v. MIT App Inventor
- vi. Python code

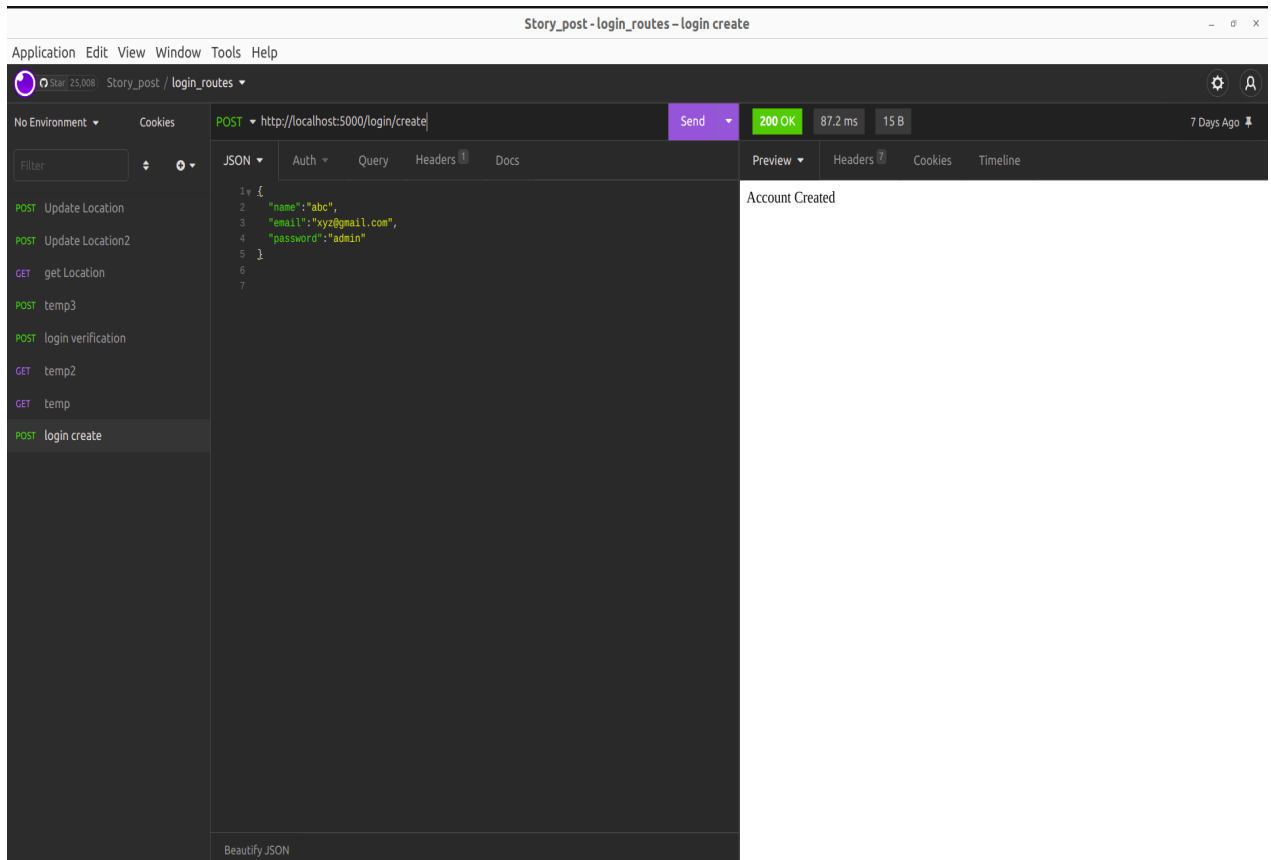
### **7.2 Feature 2**

- . Login
- i. Verification
- ii. Ticket Booking
- iii. Adding rating

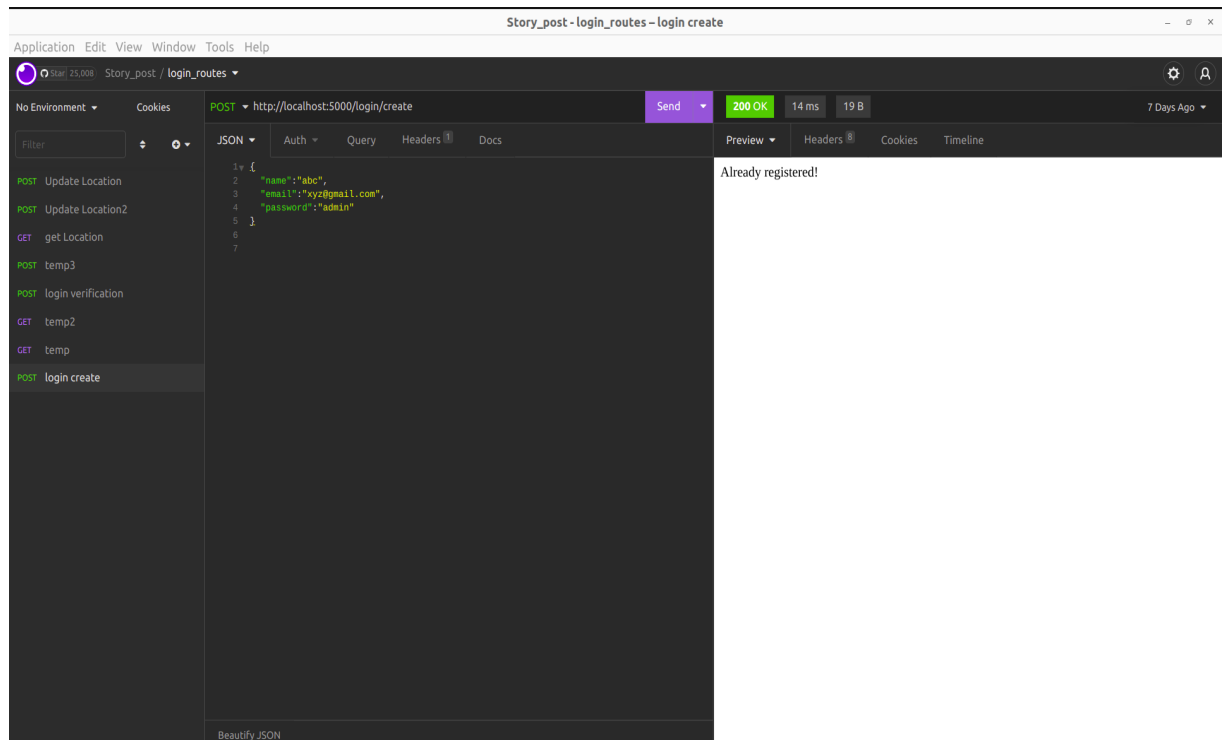
## 8. TESTING AND RESULTS

### 8.1 Test Case

#### Test case 1



#### Test case 2



## Test case 3

The screenshot shows the Postman application window titled "Story\_post - login\_routes - login verification". The interface includes a menu bar (Application, Edit, View, Window, Tools, Help) and a toolbar with a star icon, "25.008", and "Story\_post / login\_routes". On the left, a list of requests is shown, with "POST login verification" selected. The main panel displays the details of the selected request: a POST to "http://localhost:5000/login". The "Send" button is highlighted. The response status is "200 OK" with a time of "9.11 ms" and a size of "5 B". The response body is a valid JSON object: 

```
{ 1: { 2: "name": "abc", 3: "email": "xyz@gmail.com", 4: "password": "admin" 5: } 6: 7: }
```

. The "Raw" tab is selected, showing the raw response body as "Valid". The "Headers", "Cookies", and "Timeline" tabs are also visible.

## Test case 4

The screenshot shows the Postman application window titled "Story\_post - login\_routes - login verification". The interface is similar to the previous one, but the response status is "200 OK" with a time of "7.17 ms" and a size of "7 B". The response body is an invalid JSON object: 

```
{ 1: { 2: "name": "abc", 3: "email": "xyz@gmail.com", 4: "password": "admin" 5: } 6: 7: }
```

. The "Raw" tab is selected, showing the raw response body as "Invalid". The "Headers", "Cookies", and "Timeline" tabs are also visible.

## Test case 5

The screenshot shows a REST client application window titled "Story\_post - login\_routes - get Location". The interface includes a menu bar (Application, Edit, View, Window, Tools, Help), a toolbar with a "Send" button, and a status bar at the bottom.

The main area is divided into three panels:

- Left Panel:** A list of endpoints under the "login\_routes" collection. The selected endpoint is "get Location" (GET).
- Center Panel:** The "JSON" tab is active, displaying the request body as a JSON object: 

```
{ 1: { 2: "_id": "25147" 3: } }
```
- Right Panel:** The "Preview" tab is active, displaying the response body as a JSON object: 

```
{ 1: { 2: "_id": "25147", 3: "location": "Tindivanam-TMV", 4: "name": "Cholan Express" 5: } }
```

The status bar at the bottom indicates a "200 OK" response with a status of "200 OK", a response time of "8.27 ms", and a response size of "67 B". A notification at the bottom of the window states: "meet.google.com is sharing your screen. Stop sharing Hide".

## **9. ADVANTAGES**

- Passengers who are traveling alone can use this application to safeguard their safety.
- Passengers can track train location status as well as they will get the notification when the train arrives at the station.

## **10. DISADVANTAGES**

- There may be network issues.

## **11. CONCLUSION**

Almost every country in the world strives to fulfill the need for safe, rapid, and dependable train services. Lack of operating efficiency and dependability, safety and security concerns, as well as aging railway systems and procedures, are all urging various countries to modernize their rail infrastructure. Due to a lack of optimized rail network utilization and inefficient use of rail assets, the global rail sector is struggling to fulfill the rising demand for freight and passenger transportation. They frequently lack sophisticated technology and the most recent technical improvements to give the most effective passenger services. This is likely to encourage rail executives to design better, more efficient train networks. The Indian Railways' passenger reservation system is one of the largest in the world. On Indian Railways, nearly one million people travel in reserved rooms every day. Another sixteen million people commute on Indian Railways using unreserved tickets. It is a herculean challenge in this enormous system to properly manage passenger data, which is a critical factor these days. However, due to increasing demand for the most efficient passenger services, the installation of the most recent technology improvements in this system is becoming increasingly necessary. Handling passenger data effectively, supported by intelligent processing and prompt retrieval, would contribute to fewer security breaches. We've looked at many aspects of incorporating smart computing in railway systems related to reservation models, as well as some potential future directions. More informative and user-friendly websites, mobile applications providing real-time information regarding cars in motion, and e-ticket sales and schedule information adopted at stations and stops have been the most important advancements. Railroad firms may now ensure that they are prepared to prevent the surprise of equipment downtime as the sector grows. As previously said, our project's produced application may lead the passenger who travels securely and without worry.

## **12. FUTURE SCOPE**

This service ensures passenger safety while traveling alone or with family or friends. Bus passengers may potentially benefit from this application in the future. Passengers will be able to learn more about their safety once the programme is improved.

## 13. APPENDIX

### 13.1 Source Code

#### App.jsx

```
import "./App.css";
import React, { Component } from "react";
import { Routes, Route, Link, BrowserRouter, Navigate } from "react-router-dom";
import LoginPage from "./components/login.jsx";
import HomePage from "./components/home.jsx";
import Navbar from "./components/navbar";
import CreateAccount from "./components/createAccount";
import NotificationContainer from "react-notifications/lib/NotificationContainer";
import Profile from "./components/profile";
import Recent from "./components/recent";
import Reservation from "./components/reservation";
import TrainStatus from "./components/TrainStatus";
import AboutUs from "./components/about-us";

class App extends Component {
  state = {};
  render() {
    return (
      <BrowserRouter>
        <Navbar />
        <Routes>
          <Route path="/" element={<HomePage />} />
          <Route path="/login" element={<LoginPage />} />
          <Route path="/createAccount" element={<CreateAccount />} />
          <Route path="/profile" element={<Profile />} />
          <Route path="/recent" element={<Recent />} />
          <Route path="/book-ticket" element={<Reservation />} />
          <Route path="/status" element={<TrainStatus />} />
        </Routes>
        <NotificationContainer />
      </BrowserRouter>
    );
  }
}
```

export default App;



## Booking.jsx

```
import React, { Component } from "react";
import ReactDOM from "react-dom";
// import { fuzzySearch } from "react-select-search";
import SelectSearch from "react-select-search";
import "./reservationDropdownStyle.css";
import stations from "./stations.json";
import DatePicker from "react-datepicker";
import jsPDF from "jspdf";
import {
  NotificationManager,
  NotificationContainer,
} from "react-notifications";
import axios from "axios";
// const trainImg = require("./pictures/train1.jpg");
const URL = "http://localhost:5000/";

class Reservation extends Component {
  state = {
    fromStationId: "",
    currentState: 1,
    toStationId: "",
    trains: [],
    currentTrain: {},
    startDate: new Date(),
    passangers: [],
  };
  handleFromStationChange = (e) => {
    console.log("Fruit Selected!!", e);
    // this.setState({ fruit: e.target.value });
    this.setState({ fromStationId: e });
  };
  handleToStationChange = (e) => {
    console.log("Fruit Selected!!", e);
    // this.setState({ fruit: e.target.value });
    this.setState({ toStationId: e });
  };
  selectFromStation = () => {
    const countries = [];
    for (const i in stations) {
      countries.push({
        name: stations[i]["stationName"] + " - " + stations[i]["code"],
        value: stations[i]["stationName"] + " - " + stations[i]["code"],
      });
    }
    return (
      <SelectSearch
        options={countries}
        value={this.state.fromStationId}
      />
    );
  };
}
```

```

        search
        // filterOptions={fuzzySearch}
        emptyMessage="Not found"
        onChange={this.handleFromStationChange}
        placeholder="Stations"
    />
    );
};

selectToStation = () => {
    const countries = [];
    for (const i in stations) {
        countries.push({
            name: stations[i]["stationName"] + " - " + stations[i]["code"],
            value: stations[i]["stationName"] + " - " + stations[i]["code"],
        });
    }
    return (
        <SelectSearch
            options={countries}
            value={this.state.toStationId}
            search
            // filterOptions={fuzzySearch}
            emptyMessage="Not found"
            onChange={this.handleToStationChange}
            placeholder="Stations"
        />
    );
};

onDateChange = (e) => {
    console.log(e);
    if (e < Date.now() - 24 * 3600000) {
        console.log("Date range");
        NotificationManager.warning("Invalid Date");
        return;
    }
    this.setState({ startDate: new Date(e), trains: [] });
};

Example = () => {
    return (
        <DatePicker
            selected={this.state.startDate}
            onChange={this.onDateChange}
            dateFormat="d-MMMM-yyyy"
            className="rounded"
        />
    );
};

dynamicStateChange = () => {
    if (this.state.currentState === 1) {
        return this.stage1();
    }
};

```



```

stage2 = () => {
  if (this.state.trains.length === 0) {
    for (let i = 0; i < 15; i++) {
      this.state.trains.push({
        trainno: Math.floor(Math.random() * 100000),
        name: "ABC express",
        arrivalTime: this.getStage2ArrivalTime(),
        departureTime: this.getStage2departureTime(),
      });
    }
    this.state.trains.sort((a, b) => {
      return a.arrivalTime.getTime() - b.arrivalTime.getTime();
    });
    this.setState({ trains: this.state.trains });
  }
  return (
    <div className="container-fluid">
      {this.state.trains.map((i) => {
        return (
          <div onClick={e => this.selectTrain(i)} className="row ">
            <div className={"col m-2 rounded " + this.stage2DynamicColor(i)}>
              {i.trainno + " - " + i.name}
            </div>
            <div className={"col m-2 rounded " + this.stage2DynamicColor(i)}>
              {this.displayTime(i.arrivalTime)}
            </div>
            <div className={"col m-2 rounded " + this.stage2DynamicColor(i)}>
              {this.displayTime(i.departureTime)}
            </div>
          </div>
        );
      })}
    </div>
  );
};

displayTime = (curTime) => {
  return curTime.toLocaleDateString() + " " + curTime.toLocaleTimeString();
};

getStage2ArrivalTime = () => {
  const fiveRound = 5 * 60 * 1000;
  const curTime = new Date(
    Math.floor(
      (this.state.startDate.getTime() + Math.random() * 20 * 3600000) /
      fiveRound
    ) * fiveRound
  );
  console.log(this.state.startDate);
  return curTime;
};

getStage2departureTime = () => {

```

```

const fiveRound = 5 * 60 * 1000;
const curTime = new Date(
  Math.floor(
    (this.state.startDate.getTime() + (1 + Math.random()) * 20 * 3600000) /
    fiveRound
  ) * fiveRound
);
console.log(this.state.startDate);
return curTime;
};
selectTrain = (e) => {
  this.setState({ currentTrain: e });
};
stage2DynamicColor = (i) => {
  if (i === this.state.currentTrain) return "bg-primary fw-bold";
  return "bg-info";
};
stage3 = () => {
  if (this.state.passangers.length === 0) this.addPasangers();
  return (
    <div className="container-fluid">
      <div
        style={{ paddingTop: 10, paddingBottom: 10 }}
        className="row justify-content-end"
      >
        <div className="col-1">
          <button onClick={this.addPasangers} className="btn bg-info">
            <svg
              xmlns="http://www.w3.org/2000/svg"
              width="16"
              height="16"
              fill="currentColor"
              class="bi bi-plus"
              viewBox="0 0 16 16"
            >
              <path d="M8 4a.5.5 0 0 1 .5.5v3h3a.5.5 0 0 1 0 1h-3v3a.5.5 0 0 1 1 1v-3h3a.5.5 0 0 1 0 1h-3v-3A.5.5 0
0 1 8 4z"/>
            </svg>
          </button>
        </div>
        <div className="col-1">
          <button onClick={this.removePassengers} className="btn bg-info">
            <svg
              xmlns="http://www.w3.org/2000/svg"
              width="16"
              height="16"
              fill="currentColor"
              class="bi bi-dash"
              viewBox="0 0 16 16"
            >

```

```

        <path d="M4 8a.5.5 0 0 1 .5-.5h7a.5.5 0 0 1 0 1h-7A.5.5 0 0 1 4 8z" />
    </svg>
</button>
</div>
</div>
<div>{this.getFormattedPassanger()}</div>
</div>
);
};
getFormattedPassanger = () => {
    return this.state.passangers.map((i) => {
        return (
            <div className="row m-2">
                <div className="col">
                    <div className="form-floating">
                        <input
                            type="text"
                            className="form-control"
                            onChange={e => this.handleNameChange(e, i)}
                            placeholder="name"
                            name="name"
                            value={i.name}
                        />
                        <label for="floatingInput">Name</label>
                    </div>
                </div>
                <div className="col">
                    <div className="form-floating">
                        <input
                            type="number"
                            className="form-control"
                            onChange={e => this.handleAgeChange(e, i)}
                            placeholder="age"
                            name="name"
                            value={i.age}
                        />
                        <label for="floatingInput">Age</label>
                    </div>
                </div>
                <div className="col">
                    <div className="form-floating">
                        <input
                            type="number"
                            className="form-control"
                            onChange={e => this.handleADNChange(e, i)}
                            placeholder="age"
                            name="name"
                            value={i.adno}
                        />
                        <label for="floatingInput">Aadhar number</label>

```

```

        </div>
    </div>
    <div className="col flex-grow-0">
        <select
            style={{ marginTop: 20 }}
            value={i.gender}
            onChange={(e) => this.handleGenderChange(e, i)}
        >
            <option value="Male">Male</option>
            <option value="Female">Female</option>
        </select>
    </div>
</div>
);
});
};

```

```

handleADNChange = (e, i) => {
    i.adno = e.target.value;
    this.setState({ passangers: this.state.passangers });
};
handleAgeChange = (e, i) => {
    i.age = e.target.value;
    this.setState({ passangers: this.state.passangers });
};
handleNameChange = (e, i) => {
    i.name = e.target.value;
    this.setState({ passangers: this.state.passangers });
};
handleGenderChange = (e, i) => {
    // console.log(e, i);
    i.gender = e.target.value;
    this.setState({ passangers: this.state.passangers });
};
addPasangers = () => {
    this.state.passangers.push({
        name: "",
        gender: "Male",
        age: null,
        adno: null,
    });
    this.setState({ passangers: this.state.passangers });
};
removePassengers = () => {
    if (this.state.passangers.length === 1) {
        NotificationManager.warning("can't delete");
    } else {
        this.state.passangers.pop();
        this.setState({ passangers: this.state.passangers });
    }
}

```

```

};
stage4 = () => {
  var amt = this.state.passangers.length * 100;
  console.log(amt);
  if (amt > localStorage.getItem("balance")) {
    console.log(amt);
    NotificationManager.info(
      "Recharge you wallet in your profile section",
      "Insufficient ammount"
    );
    console.log(amt);
    return "Invalid";
  }
  return (
    <button onClick={this.pay} className="btn btn-primary">
      Pay
    </button>
  );
};
pay = () => {
  const data = { ...this.state };
  delete data["trains"];
  delete data["currentState"];
  data["uid"] = localStorage.getItem("email");
  data["amt"] = this.state.passangers.length * 100;
  console.log(data);
  axios.post(URL + "user/book", data).then((res) => {
    console.log(res);
    if (res.data === "debited") {
      this.setState({ currentState: this.state.currentState + 1 });
    } else {
      NotificationManager.warning("Error");
    }
  });
};
stage5 = () => {
  const tkbody = "\
<div> <h1>hi</h1> <h3>hello</h3> </div> \
";
  return (
    <div className="h1">Successfully booked</div>
    // <button onClick={this.generatePdf}>
    //   <h1>Download</h1>
    // </button>
  );
};
generatePdf = () => {
  var doc = new jsPDF("p", "pt", "a4");
  const tkbody = `
<div> <h1>hi</h1> <h3>hello</h3> </div>

```



```

`;
doc.html(tkbody, {
  callback: function (doc) {
    doc.save("tk.pdf");
  },
  x: 0,
  y: 0,
});
};

incrementState = () => {
  this.setState({ currentState: this.state.currentState + 1 });
};

decrementState = () => {
  this.setState({ currentState: this.state.currentState - 1 });
};

render() {
  return (
    <div style={{ minHeight: "85vh" }} className="container-fluid ">
      {this.dynamicStateChange()}
      {/* {this.stage4()} */}
      {/* <div className="row ">
        <div className="col-sm d-flex justify-content-center justify-content-md-end ">
          <span>From</span>
        </div>
        <div className="col-sm d-flex justify-content-center justify-content-md-start ">
          {this.selectFromStation()}
        </div>
      </div> */}
      <div className=
        <button onClick={this.decrementState}>Back</button>
        <button onClick={this.incrementState}>Next</button>
      </div>
    </div>
  );
}
}

export default Reservation;

```

## 13.2 GitHub & Demo Video

**GitHub Link:**

<https://github.com/IBM-EPBL/IBM-Project-25456-1659964203>

**Demo Link:**

<https://youtu.be/sA4zBelCOVo>