Smart Solutions for Railways

Category: *Internet of Things*

PROJECT REPORT

SUBMITTED BY

Team ID: PNT2022TMID14632

NAME	REGISTER NUMBER
1. Kiruthika .V	111619106065
2. Gummireddy Harshitha	111619106041
3. Kanniga Parameswari .S	111619106058
4. Leelavathi .S	111619106071

INPARTIAL FULFILLMENT FOR THE

AWARD OF THE DEGREE

Of

BACHELOR OF ENGINEERING

in

ELECTRONICS AND COMMUNICATION ENGINEERING

R M K COLLEGE OF ENGINEERING AND TECHNOLOGY

Pudhuvoyal, Gummidipoondi-601206.

Project Report Format

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)

1. INTRODUCTION

1.1 Project Overview

As trains are one of the most preferred modes of transportation amongmiddle class and impoverished people as it attracts for its amenities.

Simultaneously there is an increase at risk from thefts and accidents like chain snatching, derailment, fire accident. In order to avoid or in better words to stop allsuch brutality we came up with a solution by providing an application which can beaccessed by the user after booking their tickets. With a single click this app addresses issues by sending a text message to TC and RPF as an alert. In our project we use Node-Red service, app-development, IBM cloud platform to store passenger data.

1.2 Purpose

The purpose of this project is to report and get relived from the issues related totrains.

2. LITERATURE SURVEY

2.1 Existing problem

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

After booking the train, the person will get a QR code which has to be shown to theTicket 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 isupdated in the Web app continuously

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

2.2 References

S.NO	TITLE	AUTHOR	YEAR	KEY TECHNOLOGY
1	Main geotechnical problems of railways and roads in kriolitozone and their solutions.	Kondratiev, Valentin G	2017	Main problems in railways
2	Construction and Building Materials	Sañudo, Roberto, Marina Miranda, Carlos García, and David García-Sanchez	2019	Drainage in railways
3	Problems of Indian Railways	Benjamin	2021	Common problems in Indian railways
4	A comparative study of Indian and worldwide railways.	Sharma, Sunil Kumar, and Anil Kumar	2014	Study of Indian railways
5	Ticketing solutions for Indian railways using RFID technology	Prasanth,Venugopal, and K.P. Soman	2009	Solution for ticketing using RFID

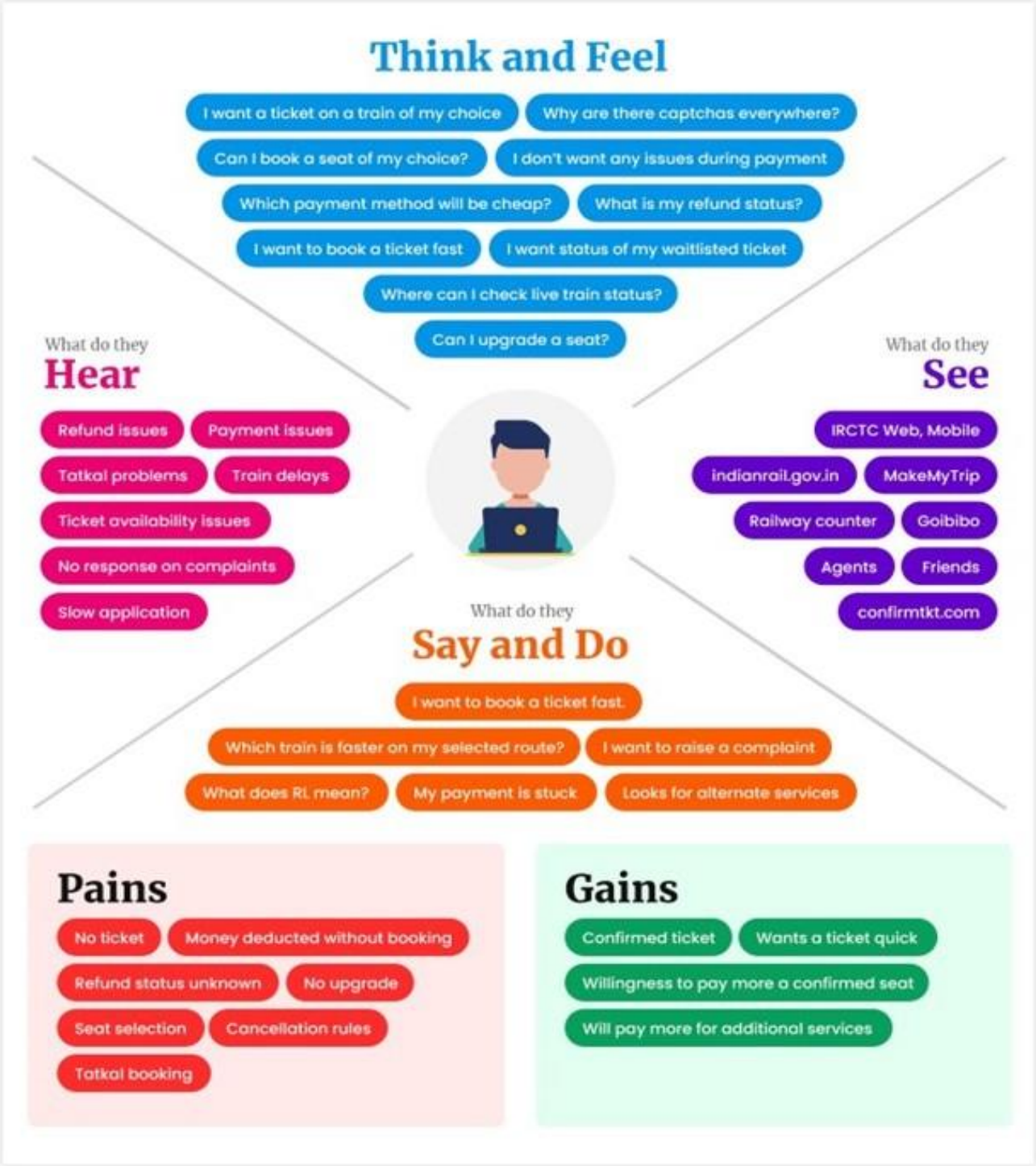
2.3 Problem Statement Definition

Smart Solutions for railways are designed to reduce the work load of the user and the use of paper.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

Online Ticket Booking:



3.2 Ideation & Brainstorming

- Creating an Application for passengers
- Digital Railway solution
- Digital Twin digital platform for Railways and Airways
- Role of sensors in predictive maintanance
- Predictive maintanace and CMMS
- The IOT connected trains
- Big Data analytics for smart Railways
- Safety is a key area of connection

Idea prioritization:

- To prect from:
- Ticket booking Jamming
- Fire accident
- Theft
- Robbery

Include Features like:

- Tracking management
- QR code

3.3 Proposed solutions

CS	CC	AS
1- CUSTOMER SEGMENT(S)	**6- CUSTOMER CONTRAINTS**	**5-AVAILABE SOLUTIONS**
Passengers are the customers.	Fewer Maintenance Delays Restructured and Optimized Passenger	A GPS tracking device will be placed in train which is helpful to find the live status of the train. Booking tickets is made

	Experience Advanced Analytics for Streamlined Operations	easier from a web page and for each ticket a unique QR will be provided.
J&P **2- JOBS TO BE DONE /PROBLEMS2-** **Ticket:** To provide a web page or web app to the customers to book their Railway tickets from anywhere at any time. **Tracking:** The live status of the train must be updated to the passengers.	**RC** **9. PROBLEM ROOTCAUSE** The Passengers it difficult to get the ticketby Standing in queue. Atthe same time they cantable to know the information about the delay of train. To overcome this problem we provide a unique QR and GPS module was installed in the train is used to trackit.	**BE** **7. BEHAVIOUR** According to the needs of the passengers we should provide a genuineempathy for the problem regarded. Looking over the rating section we can easily findout how the customer gets issue while using theapplication.

TR	SL	C
3. TRIGGERS Customer can be triggeredto the application by the usage of their neighbours. 4. EMOTIONS Before: They feel nervous because there is no option to proceed further and if they miss the train they can’ttrack it too. After: Now the passengers can track the live location ofthe train and will never losetheir confidence.	**10. YOUR SOLUTION** A web page will be provided and the passenger can sign in thepage and they can book their train ticket using it. When a ticket is booked the passenger will get a unique QR code for further verifications by the railway department. The passenger can also track the live status of the train in that web page.	**H8.CHANNELS of BEHAVIOUR** **ONLINE** Customers try to request for the problems through the application how they use and how it is favouring them using the rating option by whichwe can find the behaviourof the customer and issues or problems they face. **OFFLINE** By direct booking of ticket they need to be in a queue for receiving a ticket which seems to bea big deal for the customers.

3.3 Proposed Solution

S.No	Parameter	Description
1	Problem Statement (Problem to be solved)	To provide a smart way for booking tickets in railway department through a webpage with aunique QR for each ticket and to deliver the live status of the train to the passengers which is

		helpful in the critical situations (Stuck of train inforest areas)
2	Idea/ Solution description	Passengers can book their ticket using a web pageor web app. When the passenger is booking a ticketand successfully completed the payment for it, they will be provided with a unique QR code which contains the ticket details and passenger details. The passengers will get notified with the train timings and train's live status.
3	Novelty/ Uniqueness	Efficient booking system by verifying and validatingthe ticket as only registered users can book the tickets. Each passengers will be provided by a unique ID tothem during first login so that their data will be stored and processed securely. GPS tracking facility is provided to track the current location of the train from any place. A chat box will be provided for the passengers topost their queries or their needs and that will be fulfilled as soon as possible
4	Social Impact/ Customer Satisfaction	User friendly environment Services will be made for 24 x 7 Passenger data will be more securely maintained Reservation of tickets made easier
5	Business Model (Revenue Model)	Using chat bot we can contact user's ticket booking. The chat box can give instructions to theusers based on their location. It will store the customer's details and ticket orders in the

		database. The chat bot will send a notification tothe passenger if the booking is confirmed. Chat bot can also help in collecting passenger feedback.
6	Scalability of the solution	This model is easily adopted among online usersand it can be easily deployed. It can be used and accessed by everyone and it can handle the requests from the passengers.

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Online Registration through Gmail
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Application installation	The application is installed through the given link
FR-4	User access	Access the app requirements

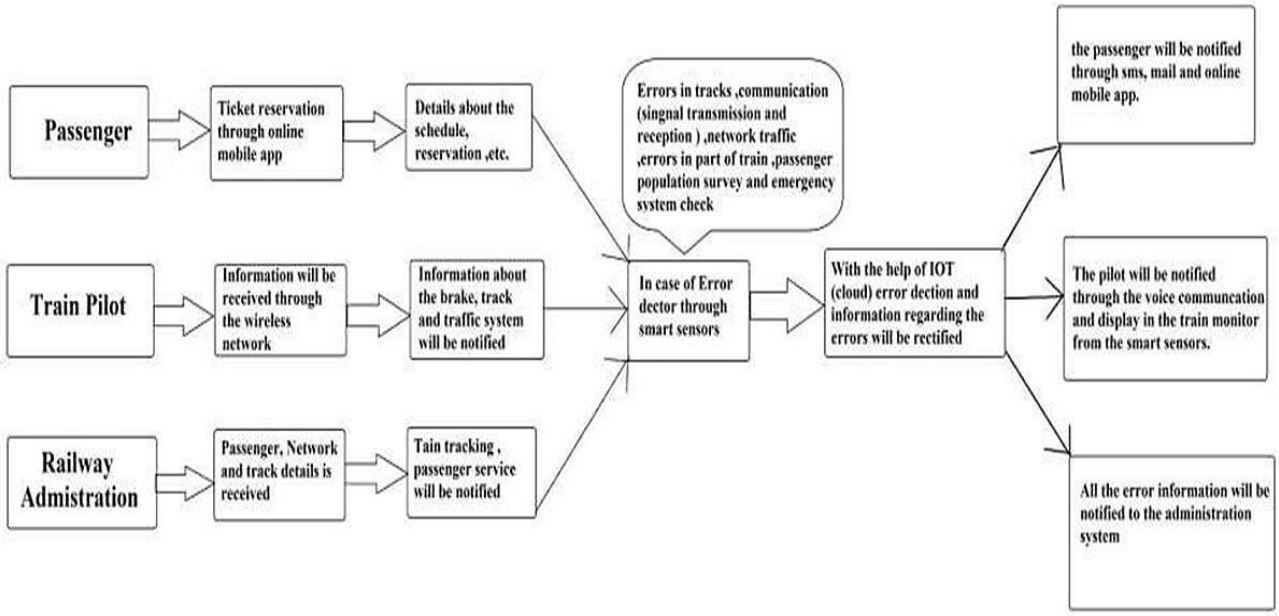
4.2 Non-Functional requirement

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	The app can be used during the travelling time Easy and simple Efficiency is high

NFR-2	Security	By clicking on the icon, the alert will be given to the respective officials
NFR-3	Reliability	Highly reliable to use
NFR-4	Performance	Low error rate
NFR-5	Availability	Free source
NFR-6	Scalability	It is scalable enough to support many users at the same time

5. PROJECT DESIGN

5.1 Data Flow Diagrams



5.2 Solution Architecture

As trains are one of the most preferred modes of transportation amongmiddle class and impoverished people as it attracts for its amenities.

Simultaneously there is an increase at risk from thefts and accidents like chain- snatching, derailment, fire accident. In order to avoid or in better words to stop allsuch brutality we came up with a solution by providing an application which can beaccessed by the user after booking their tickets. With a single click this app

addresses issues by sending a text message to TC and RPF as an alert. In our project we use Node-Red service, app-development, IBM cloud platform to store passenger data.

5.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the tickets by entering my email, and password, and confirming my password.	I can access my account/dash board	High	Sprint-1
		USN-2	As a user, I will receive a confirmation email once I have registered for the tickets.	I can receive aconfirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through the Railway application.	I can register & access the dashboard with a registration login.	Low	Sprint-1

		USN-4	As a user, I can register for the application through Online websites		Medium	Sprint-2
	Login	USN-5	As a user, I can log into the application by entering my email & password		High	Sprint-1
Train pilot	Dashboard	USN-6	To get information regarding the train system, users check the system's status through mobile applications or the dashboard display.	I can access it through the mobile app.		Sprint -1
		USN-7	While traveling the status of the track will display in the dashboard.		Medium	Sprint -2
		USN-8	other information from the admin will be displayed with an alert in the dashboard		High	Sprint -2

			display			
Administrator		USN-9	The Railway network can be monitored from the base stationof the railway	Access through the wireless network and computer system	High	Sprint -1
		USN-10	In the computer system, the railway network traffic can be analysed and easy paths can be chosen.		High	Sprint -1
		USN-11	In case of a communication signal error or problem, it will be displayed on the monitor so that the data can be sent again.		High	Sprint -1

		USN-12	The error in the tracks will be informed to the train pilot's admin and received through the mobile app or computer system.	Can be accessed through the display system ie computer system in the train	High	Sprint -1
		USN-13	The passenger details will be automatically saved on the database of the admin computer system.		Medium	Sprint -1
Customer Care Executive		USN-14	A portal is been arranged for the passenger help. the passenger can directly makea call to the respective number and ask for help	Can be accessed through telephonyitself	High	Sprint -1
		USN-15	Passengers can text the respective number through the mobile app.		Medium	Sprint -2

Customer (Web User)	Passenger objection and feedback	USN-16	Passenger call to give their feedback to the railway website.		High	Sprint -2
		USN-17	In case of any software error from the railway side, it can be reported to the inquiry desk through mail or message.	Accessed through mail or SMS	High	Sprint -2

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

STEP 1	Identify the problem
STEP 2	Prepare an abstract, problem statement
STEP 3	List required objects needed
STEP 4	Create a code and run it

STEP 5	Make a prototype
STEP 6	Test with the created code and check the designed prototype is working
STEP 7	Solution for the problem is found

6.2 Reports from

JIRASPRINT 1

```
#include <LiquidCrystal.h>

LiquidCrystal 1cd(5,6,8,9,10,11); int red1ed = 2; int green1ed = 3;int
buzzer = 4; int sensor = A0;
int sensorThresh = 400;void
setup()

{

pinMode(red1ed, OUTPUT); pinMode(green1ed,OUTPUT); pinMode(buzzer,OUTPUT);
pinMode(sensor,INPUT); serial.begin(9600); 1cd.begin(16,2);

}
Void loop()

{

int analogValue = analogRead(sensor); Serial.print(analogvalue);
```

```
if(analogValue>sensorThresh)
      {
         digitalWrite(red1ed,HIGH);  digit1Weite(green1ed,LOW);
tone(buzzer,1000,10000);
         1cd.clear();
         1cd.setCursor(0,1);
         1cd.print("RAILWAYS"); delay(1000);
         1cd.clear();
         1cd.setCursor(0,1);
         1cd.print("SMART  SOLUTION");  delay(1000);
     }

     else
    {
         digitalWrite(greenlad,HIGH); digitalWrite(red1ed,LOW);
         noTone(buzzer); 1cd.clear(); 1cd.setCursor(0,0);
         1cd.print("SAFE"); delay(1000);
           1cd.clear();
           1cd.setCursor(0,1);
         1cd.print("ALL CLEAR"); delay(1000);
       }
 }
```

SPRINT 2

Main Program:

```
importwiotp.sdk.device
importtime importrandom
myConfig={
"identity":{
"orgId":"gagtey",
"typeId":"GPS",
"deviceId":"12345"
},
"auth":{
"token":"12345678"
}
}
defmyCommandcallback(cmd):
print("messagereceivedfromIBMIOTPlatform:%s"%cmd.data['command'])
m=cmd.data['command']
client=wiotp.sdk.device.deviceclient(config=myConfig,logHandlers=None) client.connect()
defpub(data): client.publishEvent(eventId="status",msgFormat="json",data=mydata,qos=0,
print("publishedatasuccessfully:%s",mydata)
whileTrue:
mydata={'name':'Train1','lat':17.6387448,'lon':78.4754336)
pub(myData)
time.sleep(3)
#mydata={'name':'Train2','lat':17.6387448,'lon':78.4754336)
#pub(myData)
```

```
#time.sleep(3)
mydata={'name':'Train1','lat':17.6341908,'lon':78.4744722)
pub(myData)
time.sleep(3)
mydata={'name':'Train1','lat':17.6340889,'lon':78.4745052)
pub(myData)
time.sleep(3)
mydata={'name':'Train1','lat':17.6248626,'lon':78.4720259)
pub(myData)
time.sleep(3)
mydata={'name':'Train1','lat':17.6188577,'lon':78.4698726)
pub(myData)
time.sleep(3)
mydata={'name':'Train1','lat':17.6132382,'lon':78.4707318)
pub(myData)
time.sleep(3)
client.commandCallback=mycommanCallbak
client.disconnect()
```

Code:

```
importcv2 importnumpyasnp
importtime
importpyzbar.pyzbaraspuzbar
fromibmcloudant.cloudant_v1importcloudantv1
fromibmcloudantimportcouchDbsessionAuthenticator
fromibm_cloud_sdk_core.AuthenticatorsimportBasicAuhtenticator
authenticator=BasicAuthenticator('apikey-v2-
16u3crmdpkghhxefdikvpssoh5fwezrmuup5fv5g3ubz','b0ab119f45d3e6255eabb978)
service=cloudantv1(authenticator=authenticator)
```

```
service.set_service_url('https:/ apikey-v2-
16u3crmdpkghhxefdikvpssoh5fwezrmuup5fv5g3ubz:b0ab119f45d3e6255eabb978
cap=cv2.videoCapture(0)
font=cv2.FONT_HERSHEY_PLAIN
whileTrue:
_,frame=cap.read(0)
decodeObjects=pyzbar.decode(frame)
forobjindecodeObjects:
#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: responce=service.get_document(db='booking',doc_id=a).get_result()
print(response)
time.sleep(5)
exceptExceptionase:
print("NotvalidTicket")
time.sleep(5)
cap.imshow("Frame",frame)
ifcv2.waitKey{1}&0XFF==ord('q'):
break
cap.release()
cv2.destroyAllWindows()
client.disconnect()
```

SPRINT 3

- This project presents its first ever digital event dedicated to rail transport, the“Smart Mobility Experience” which will take place on March 24th. This event

will be the occasion for clients and partners of the rail ecosystem, to discovernew products and major innovations, as well as to exchange about the digitalization and future of rail.

- for improved service performance and energy efficiency, and to boost the attractiveness for users.
- It helps transporting passengers safely, and with best possible experience, supervises operations with accurate situation awareness, and optimizes transport service efficiency.
- Using digital technologies such as IoT, cloud and web IT, data analytics , it designs innovative solutions such as digital signalling, train autonomy, mobile ticketing, passenger flow analytics, data driven operation control, smart maintenance, which will drastically impact the way we all travel.
- Provide real-time passenger density insights to public transport operators
- The solution helps alleviate crowding by reducing busy times, and consequently enhances overall passenger safety, comfort, and travel experience.
- The targeted performances of density accuracy are above 90%.

In Hand's Connectivity Solution for Rail Transit:

MAIN:

```
importwiotp.sdk.device
importtime importrandom
myConfig={
"identity":{
"orgId":"gagtey",
```

```
"typeId":"GPS",
"deviceId":"12345"
},
"auth":{
"token":"12345678"
}
}
defmyCommandcallback(cmd):
print("messagereceivedfromIBMIOTPlatform:%s"%cmd.data['command'])
m=cmd.data['command']
client=wiotp.sdk.device.deviceclient(config=myConfig,logHandlers=None) client.connect()
defpub(data): client.publishEvent(eventId="status",msgFormat="json",data=mydata,qos=0,
print("publishedatasuccessfully:%s",mydata)
whileTrue:
mydata={'name':'Train1','lat':17.6387448,'lon':78.4754336)
pub(myData)
time.sleep(3)
#mydata={'name':'Train2','lat':17.6387448,'lon':78.4754336)
#pub(myData)
#time.sleep(3)
mydata={'name':'Train1','lat':17.6341908,'lon':78.4744722)
pub(myData)
time.sleep(3)
mydata={'name':'Train1','lat':17.6340889,'lon':78.4745052)
pub(myData)
time.sleep(3)
mydata={'name':'Train1','lat':17.6248626,'lon':78.4720259)
pub(myData)
```

```
time.sleep(3)
mydata={'name':'Train1','lat':17.6188577,'lon':78.4698726)
pub(myData)
time.sleep(3)
mydata={'name':'Train1','lat':17.6132382,'lon':78.4707318)
pub(myData)
time.sleep(3)
client.commandCallback=mycommanCallbak
client.disconnect()
```

PROGRAM:

```
importcv2 importnumpyasnp
importtime
importpyzbar.pyzbaraspuzbar
fromibmcloudant.cloudant_v1importcloudantv1
fromibmcloudantimportcouchDbsessionAuthenticator
fromibm_cloud_sdk_core.AuthenticatorsimportBasicAuhtenticator
authenticator=BasicAuthenticator('apikey-v2-
16u3crmdpkghhxefdikvpssoh5fwezrmuup5fv5g3ubz','b0ab119f45d3e6255eabb978)
service=cloudantv1(authenticator=authenticator)
service.set_service_url('https:/ apikey-v2-
16u3crmdpkghhxefdikvpssoh5fwezrmuup5fv5g3ubz:b0ab119f45d3e6255eabb978
cap=cv2.videoCapture(0)
font=cv2.FONT_HERSHEY_PLAIN
whileTrue:
_,frame=cap.read(0)
decodeObjects=pyzbar.decode(frame)
```

```
forobjindecodeObjects:
#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: responce=service.get_document(db='booking',doc_id=a).get_result()
print(response)
time.sleep(5) exceptExceptionase:
print("NotvalidTicket")
time.sleep(5)
cap.imshow("Frame",frame)
ifcv2.waitKey{1}&0XFF==ord('q'):
break
cap.release()
cv2.destroyAllWindows()
client.disconnect()
```

SPRINT 4

Main:

```
importwiotp.sdk.device
importtime importrandom
myConfig={
"identity":{
"orgId":"gagtey",
"typeId":"GPS",
"deviceId":"12345"
},
```

```
"auth":{
"token":"12345678"
}
}
defmyCommandcallback(cmd):
print("messagereceivedfromIBMIOTPlatform:%s"%cmd.data['command'])
m=cmd.data['command']
client=wiotp.sdk.device.deviceclient(config=myConfig,logHandlers=None) client.connect()
defpub(data): client.publishEvent(eventId="status",msgFormat="json",data=mydata,qos=0,
print("publishedatasuccessfully:%s",mydata)
whileTrue:
mydata={'name':'Train1','lat':17.6387448,'lon':78.4754336)
pub(myData)
time.sleep(3)
#mydata={'name':'Train2','lat':17.6387448,'lon':78.4754336)
#pub(myData)
#time.sleep(3)
mydata={'name':'Train1','lat':17.6341908,'lon':78.4744722)
pub(myData)
time.sleep(3)
mydata={'name':'Train1','lat':17.6340889,'lon':78.4745052)
pub(myData)
time.sleep(3)
mydata={'name':'Train1','lat':17.6248626,'lon':78.4720259)
pub(myData)
time.sleep(3)
mydata={'name':'Train1','lat':17.6188577,'lon':78.4698726)
pub(myData)
```

```
time.sleep(3)
mydata={'name':'Train1','lat':17.6132382,'lon':78.4707318)
pub(myData)
time.sleep(3)
client.commandCallback=mycommanCallbak
client.disconnect()
```

Program:

```
importcv2 importnumpyasnp
importtime
importpyzbar.pyzbaraspuzbar
fromibmcloudant.cloudant_v1importcloudantv1
fromibmcloudantimportcouchDbsessionAuthenticator
fromibm_cloud_sdk_core.AuthenticatorsimportBasicAuhtenticator
authenticator=BasicAuthenticator('apikey-v2-
16u3crmdpkghhxefdikvpssoh5fwezrmuup5fv5g3ubz','b0ab119f45d3e6255eabb978)
service=cloudantv1(authenticator=authenticator)
service.set_service_url('https:/ apikey-v2-
16u3crmdpkghhxefdikvpssoh5fwezrmuup5fv5g3ubz:b0ab119f45d3e6255eabb978
cap=cv2.videoCapture(0)
font=cv2.FONT_HERSHEY_PLAIN
whileTrue:
_,frame=cap.read(0)
decodeObjects=pyzbar.decode(frame)
forobjindecodeObjects:
#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: responce=service.get_document(db='booking',doc_id=a).get_result()
print(response)
time.sleep(5) exceptExceptionase:
print("NotvalidTicket")
time.sleep(5)
cap.imshow("Frame",frame)
ifcv2.waitKey{1}&0XFF==ord('q'):
break
cap.release()
cv2.destroyAllWindows()
client.disconnect()
```

7. CODING & SOLUTIONING

7.1 Feature 1

1. IoT device
2. IBM Watson Platform
3. Node red
4. Cloudant DB
5. Web UI
6. MIT App Inventor
7. Python code

7.2 Feature 2

1. Login
2. Verification

3. Ticket Booking
4. Adding rating

8. TESTING AND RESULTS

8.1 Test Cases

Test Case 1

Test case ID	Feature Type	Component	Test Scenario	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Executed By
1	Functional	Registratio n	Registration through the form by Filling in my details	1.Click on register 2.Fill the registration form 3.click Register		Registration form to be filled is to be displayed	Working as expected	PASS	VAISHNAVI
2	UI	Generating OTP	Generating the otp for further process	1.Generating of OTP number		user can register through phone numbers and to get otp number	Working as expected	PASS	MRITHULLA
3	Functional	OTP verification	Verify user otp using mail	1.Enter gmail id and enter password 2.click submit	Username: railways password: admin	OTP verifed is to be displayed	Working as expected	FAIL	JESLENE
4	Functional	Login page	Verify user is able to log into application with InValid credentials	1.Enter into log in page 2.Click on My Account dropdown button 3.Enter InValid username/email in Email text box 4.Enter valid password in password text box	Username: railways password: admin	Application should show 'incorrect email or password ' validation message.	Working as expected	FAIL	ABINAYA
5	Functional	Display Train details	The user can view about the available train details	1.As a user, I can enter the start and destination to get the list of trains available connecting the above	Username: railways password: admin	A user can view about the available trains to enter start and destination details	Working as expected	PASS	VAISHNAVI

Test Case 2

Test case ID	Feature Type	Componen t	Test Scenario	Pre-Requisite	Steps To Execute	Expected Result	Actual Result	Status	Executed By
1	Functional	Booking	user can provide the basic details such as a name, number, etc		1. Enter the member's details like name, number.	Tickets booked to be displayed	Working as expected	Pass	Abinaya
2	UI	Booking seats	User can choose the train, starting and ending destination, date of travel.		1. Known to which train is available	known to which the seats are available	Working as expected	fail	Jeslene
3	Functional	Payment	user, I can choose to pay through credit Card/debit card/UPI.		1.user can choose payment method 2.payment method	payment for the booked tickets to be done using payment method through either the following methods credit Card/debit	Working as expected	Fail	Mrithulla
4	Functional	Redirection	user can be redirected to the selected		1.After payment the user will be redirected to the previous page	After payment the user will be redirected to the previous page	Working as expected	pass	Vaishnavi

Test Case 3

Test case ID	Feature Type	Componen t	Test Scenario	Pre-Requisite	Steps To Execute	Expected Result	Actual Result	Status	Executed By
1	Functional	Ticket generation	a user can download the generated e ticket for my journey along with the QR code which is used for authentication during my journey.		1.Enter method of reservation 2.Enter name,age,gender 3.Enter how many tickets wants to be booked 4.Also enter the number member's details like name,age,gender	Tickets booked to be displayed	Working as expected	Pass	Abinaya
2	UI	Ticket status	a usercan see the status of my ticket Whether it's confirmed/waiting/RAC		1.known to the status of the tickets booked	known to the status of the tivkets booked	Working as expected	Fail	Mrithulla
3	Functional	Reporting issues	user can access the reporting portal once the jouney begins		1. reporting	issues have been reported	Working as expected	pass	Vaishnavi

Test Case 4

Test case ID	Feature Type	Componen t	Test Scenario	Pre-Requisite	Steps To Execute	Expected Result	Actual Result	Status	Executed By
1	Functional	Ticket cancellatio	user can cancel my tickets there's any Change of plan		1.tickets to be cancelled	Tickets booked to be cancelled	Working as expected	Fail	Jeslene
2	Functional	Rate	a user will feed rating about the train journey		1.information feeding on trains	information feeding on trains	Working as expected	pass	Vaishnavi

9. ADVANTAGES

1. The passengers can use this application, while they are travellingalone to ensure their safety.
2. It is easy to use.
3. It has minimized error rate.

10. DISADVANTAGES

Network issues may arise.

11. CONCLUSION

Almost all the countries across the globe strive to meet the demand for safe, fast, and reliable rail services. Lack of operational efficiency and reliability, safety, and security issues, besides aging railway systems and practices are haunting various countries to bring about a change in their existing rail infrastructure. The global rail industry struggles to meet the increasing demand for freight and passenger transportation due to lack of optimized use of rail network and inefficient use of rail assets. Often, they suffer from the lack in smart technologies and latest technological updates to provide the most efficient passenger services.This is expected to induce rail executives to build rail systems that are smarter and more efficient. The passenger reservation system of Indian Railways is one of the world's largest reservation models. Daily about one million passengers travel in reserved accommodation with Indian Railways. Another sixteen million travel with unreserved tickets in Indian Railways. In this vast system, it is a herculean task to efficiently handle the passenger data, which is a key point of consideration now-a-days. But the implementation of the latest technological updates in this system gradually turns inevitable due to increasing demand for providing the most efficient passenger services. Handling the passenger data efficiently backed by intelligent processing and timely retrieval would help backing up the security breaches. Here

we've explored different issues of implementing smart computing in railway systems pertaining to reservation models besides pointing out some future scopes of advancement. Most significant improvements have been evidenced by more informative and user-friendly websites, mobile applications for real-time information about vehicles in motion, and e-ticket purchases and timetable information implemented at stations and stops. With the rise of Industry, railway companies can now ensure that they are prepared to avoid the surprise of equipment downtime. Like above mentioned, the developed application of our project can lead the passenger who travel can travel safely without any fear.

12. FUTURE SCOPE

This application is ensured for safety for the passengers while they are travelling alone as well as they travel with their family or friends.

In future, this application may also be used by passengers who travel through bus. By further enhancement of the application the passengers can explore more features regarding their safety.

13. APPENDIX

13.1 Source Code

LOGIN

```
from tkinter import *
import sqlite3

root = Tk()
root.title("Python: Simple Login Application") width = 400 height = 280 screen_width =
root.winfo_screenwidth() screen_height = root.winfo_screenheight() x = (screen_width/2)
- (width/2) y = (screen_height/2) - (height/2)
```

```
  root.geometry("%dx%d+%d+%d" % (width, height, x, y))
root.resizable(0, 0)

  #==============================VARIABLES=================
=======
  ============
  USERNAME = StringVar()
  PASSWORD  =  StringVar()

  #==============================FRAMES====================
========
  ============
  Top = Frame(root, bd=2, relief=RIDGE)
  Top.pack(side=TOP, fill=X)
  Form = Frame(root, height=200)
  Form.pack(side=TOP, pady=20)

  #==============================LABELS====================
=======
  ============
    lbl_title  =  Label(Top,  text  =  "Python:  Simple  Login Application",
  font=('arial', 15)) lbl_title.pack(fill=X)
  lbl_username = Label(Form, text = "Username:", font=('arial', 14), bd=15)
lbl_username.grid(row=0, sticky="e")
  lbl_password = Label(Form, text = "Password:", font=('arial', 14), bd=15)
lbl_password.grid(row=1, sticky="e") lbl_text = Label(Form)
  lbl_text.grid(row=2,  columnspan=2)

  #==============================ENTRY
```

```
  WIDGETS==================================
  username = Entry(Form, textvariable=USERNAME, font=(14))
username.grid(row=0, column=1)
  password = Entry(Form, textvariable=PASSWORD, show="*", font=(14))
password.grid(row=1, column=1)




  #==============================METHODS==================
========
  ============= def Database():
      global conn, cursor
                                conn =
sqlite3.connect("pythontut.db") cursor =
conn.cursor()
        cursor.execute("CREATE TABLE IF NOT EXISTS `member` (mem_id INTEGER
NOT NULL PRIMARY KEY AUTOINCREMENT, username TEXT, password TEXT)")
cursor.execute("SELECT * FROM `member` WHERE `username` = 'admin' AND
  `password` =
'admin'") if
cursor.fetchone() is
None:
                    cursor.execute("INSERT INTO `member` (username, password)
VALUES('admin',
  'admin')") conn.commit() def Login(event=None): Database() if
USERNAME.get() == "" or PASSWORD.get() == "":
              lbl_text.config(text="Please complete the required
field!", fg="red")    else:
```

```
        cursor.execute("SELECT * FROM `member` WHERE `username` = ?
AND `password`
  = ?", (USERNAME.get(), PASSWORD.get()))        if cursor.fetchone() is not None:
            HomeWindow()
            USERNAME.set("")
PASSWORD.set("")
lbl_text.config(text="")          else:
            lbl_text.config(text="Invalid username or password", fg="red")
            USERNAME.set("")          PASSWORD.set("")
      cursor.close()
      conn.close()


  #==============================BUTTON
  WIDGETS==================================
  btn_login      =      Button(Form,      text="Login",      width=45,      command=Login)
btn_login.grid(pady=25, row=3, columnspan=2)
  btn_login.bind('<Return>', Login)


    def HomeWindow():       global Home       root.withdraw()       Home = Toplevel()
        Home.title("Python: Simple Login Application")          width = 600       height = 500
    screen_width = root.winfo_screenwidth()          screen_height =
    root.winfo_screenheight()         x = (screen_width/2) - (width/2)          y =
    (screen_height/2) - (height/2)
        root.resizable(0, 0)
        Home.geometry("%dx%d+%d+%d" % (width, height, x, y))
        lbl_home = Label(Home, text="Successfully Login!", font=('times new roman',20)).pack()
```

```
btn_back = Button(Home, text='Back', command=Back).pack(pady=20, fill=X)

        def Back():     Home.destroy()     root.deiconify()
```

REGISTRATION

```
from tkinter import*     base = Tk()     base.geometry("500x500")
base.title("registration form")

labl_0 = Label(base, text="Registration form",width=20,font=("bold",20))
labl_0.place(x=90,y=53)

lb1= Label(base, text="Enter Name", width=10, font=("arial",12))
lb1.place(x=20, y=120)  en1= Entry(base)
            en1.place(x=200, y=120)

lb3= Label(base, text="Enter Email", width=10, font=("arial",12))     lb3.place(x=19,
y=160)  en3= Entry(base)
en3.place(x=200, y=160)

lb4= Label(base, text="Contact Number", width=13,font=("arial",12))
lb4.place(x=19, y=200)  en4= Entry(base)
            en4.place(x=200, y=200)

lb5= Label(base, text="Select Gender", width=15, font=("arial",12)) lb5.place(x=5,y=240)
var = IntVar()
Radiobutton(base,          text="Male",          padx=5,variable=var,
value=1).place(x=180, y=240)
Radiobutton(base,          text="Female",          padx          =10,variable=var,
value=2).place(x=240,y=240) Radiobutton(base,          text="others",  padx=15,
```

```
variable=var, value=3).place(x=310,y=240)

list_of_cntry = ("United States", "India", "Nepal", "Germany") cv = StringVar()drplist=
OptionMenu(base, cv, *list_of_cntry) drplist.config(width=15) cv.set("United States")
lb2= Label(base, text="Select Country", width=13,font=("arial",12))
lb2.place(x=14,y=280)
drplist.place(x=200, y=275)

lb6= Label(base, text="Enter Password",
width=13,font=("arial",12)) lb6.place(x=19, y=320) en6=
Entry(base, show='*') en6.place(x=200, y=320)

lb7= Label(base, text="Re-Enter Password", width=15,font=("arial",12))
lb7.place(x=21, y=360) en7 =Entry(base, show='*') en7.place(x=200, y=360)

Button(base, text="Register", width=10).place(x=200,y=400)
base.mainloop()
```

START AND DESTINATION

```
# import module import requests
from bs4 import BeautifulSoup

# user define function # Scrape the data def getdata(url): r = requests.get(url)
return r.text


# input by geek from_Station_code = "GAYA"from_Station_name
= "GAYA"
```

```
To_station_code = "PNBE"
To_station_name = "PATNA"#
url
  url                          = "https:/ www.railyatri.in/booking/trains-between-
stations?from_code="+from_Station_code+"&from_name="+from_Station_name+
"+JN+&j ourney_date=+Wed&src=tbs&to_code=" + \
    To_station_code+"&to_name="+To_station_name + \
    "+JN+&user_id=-
1603228437&user_token=355740&utm_source=dwebsearch_tbs_search_trains"

# pass the url # into getdata function htmldata = getdata(url)soup =
BeautifulSoup(htmldata, 'html.parser')

# find the Html tag
# with find() # and convert into string data_str = "" for item in soup.find_all("div",
class_="col-xs-12 TrainSearchSection"): data_str = data_str + item.get_text() result =
data_str.split("\n")

print("Train between "+from_Station_name+" and "+To_station_name) print("")

# Display the result for item in result:    if item != "":       print(item)
```

TICKET BOOKING

```
print("\n\nTicket Booking System\n")
restart = ('Y')
while restart != ('N','NO','n','no'): print("1.Check PNR status") print("2.TicketReservation")
option = int(input("\nEnter your option : "))
```

```
if option == 1:  print("Your PNR status is t3")
exit(0)
elif option == 2: people = int(input("\nEnter no. of Ticket you want : ")) name_l
= [] age_l = [] sex_l = [] for p in range(people): name = str(input("\nName : "))
name_l.append(name)   age = int(input("\nAge :"))  age_l.append(age)
sex = str(input("\nMale or Female : "))
sex_l.append(sex)
restart = str(input("\nDid you forgot someone? y/n: ")) if restart in ('y','YES','yes','Yes'):
                        restart = ('Y')   else :       x = 0    print("\nTotal Ticket : ",people)       for
p in range(1,people+1):       print("Ticket : ",p)      print("Name : ", name_l[x])      print("Age
: ", age_l[x])    print("Sex : ",sex_l[x])     x += 1
```

SEATS BOOKING

```
berth_type(s):

     if s>0 and s<73:          if s % 8 == 1 or s % 8 == 4:              print (s), "is lower
berth"         elif s % 8 == 2 or s % 8 == 5:              print (s), "is middle berth"
elif s % 8 == 3 or s % 8 == 6:              print (s), "is upper berth"          elif s % 8 == 7:
print (s), "is side lower berth"           else:
print (s), "is side upper berth"         else:
print (s), "invalid seat number"

# Driver code s = 10
berth_type(s)       # fxn call for berth type

s = 7
berth_type(s)       # fxn call for berth type
```

```
s = 0
berth_type(s)        # fxn call for berth type
```

CONFIRMATION

```
  # import module import requests from bs4 import BeautifulSoup importpandas as pd

  # user define function # Scrape the data def getdata(url): r = requests.get(url)
  return r.text

  # input by geek
  train_name = "03391-rajgir-new-delhi-clone-special-rgd-to-ndls"

  # url
  url = "https:/ www.railyatri.in/live-train-status/"+train_name

  # pass the url # into getdata function htmldata = getdata(url)soup =
  BeautifulSoup(htmldata, 'html.parser')

  # traverse the live status from # this Html code data = [] for item insoup.find_all('script', type="application/ld+json"):
  data.append(item.get_text())

  # convert into dataframe df
  = pd.read_json(data[2])

  # display this column of # dataframe
  print(df["mainEntity"][0]['name'])
```

```
print(df["mainEntity"][0]['acceptedAnswer']['text'])
```

TICKET GENERATION

```
class Ticket:       counter=0
def__init_(self,passenger_name,source,destination):self._
passenger_name=passenger_name
self.____source=source        self.____destination=destination
self.Counter=Ticket.counter          Ticket.counter+=1          def
validate_source_destination(self):
if (self._source=="Delhi" and (self.__destination=="Pune" or self._
destination=="Mumbai" or self._destination=="Chennai" orself.__
destination=="Kolkata")):                    return True          else:
                        return False
def generate_ticket(self ):
 if True:
__ticket_id=self._source[0]+self._destination[0]+"0"+str(self. Counter)
                    print( "Ticket id will be:",__ticket_id)            else: return
False               def get_ticket_id(self):          return self.ticket_id       def
get_passenger_name(self):           return self.__passenger_name          def
get_source(self):          if self._source=="Delhi":
return self._sourceelse:
print("you have written invalid soure option")                return None       def
get_destination(self):          if self.__destination=="Pune":               return
self.__destination          elif self._destination=="Mumbai":
return self.__destination           elif self._destination=="Chennai":
return self.__destination           elif self._destination=="Kolkata":
return self._destination
else:
```

```
        return  None
```

OTP GENERATION

```
import os import math import randomimport
smtplib

digits  =  "0123456789"OTP
= ""

for i in range (6):
    OTP += digits[math.floor(random.random()*10)]

otp = OTP + " is your OTP" message = otps =
smtplib.SMTP('smtp.gmail.com',              587)
s.starttls()

emailid = input("Enter your email: ")
s.login("YOUR Gmail ID", "YOUR APP PASSWORD")
s.sendmail('&&&&&&',emailid,message)

a = input("Enter your OTP >>: ") if a == OTP:
    print("Verified") else:
    print("Please Check your OTP again")
```

OTP VERIFICATION

```
import os import math import randomimport
smtplib
```

```
digits = "0123456789"OTP
  = ""
  for i in range (6):
     OTP += digits[math.floor(random.random()*10)] otp =
     OTP + " is your OTP" message = otp
  s = smtplib.SMTP('smtp.gmail.com', 587)s.starttls()

    emailid = input("Enter your email: ")
    s.login("YOUR Gmail ID", "YOUR APP PASSWORD")
    s.sendmail('&&&&&&',emailid,message)

      a = input("Enter your OTP >>: ") if a == OTP:
         print("Verified") else:
         print("Please Check your OTP again")
```

13.2 GitHu

bGitHub

link:

https://github.com/IBM-EPBL/IBM-Project-37692-1660317989

Demo Video Link

https://drive.google.com/drive/folders/1jmM3gzodBKQeg3lEd6dEURjcOAZ4urk0