PROECT REPORT

SMART SOLUTIONS FOR RAILWAYS

A PROJECT REPORT

Submitted by

Team ID: PNT2022TMID39027

K. BAKKIYALAKSHMI (422119104002)
R. KOWSALYA (422119104011)
R. KRISHNAVENI (422119104012)
A. PRINCY DIANA (422119104017)
A. PRIYA ANTONY (422119104018)

in partial fulfilment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

ST. ANNE'S COLLEGE OF ENGINEERING AND TECHNOLOGY

1. INTRODUCTION

1.1 PROJECT OVERVIEW

As trains are one of the most preferred modes of transportation among middle 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 all such brutality we came up with a solution by providing an application which can be accessed 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 to trains.

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.

2.2 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

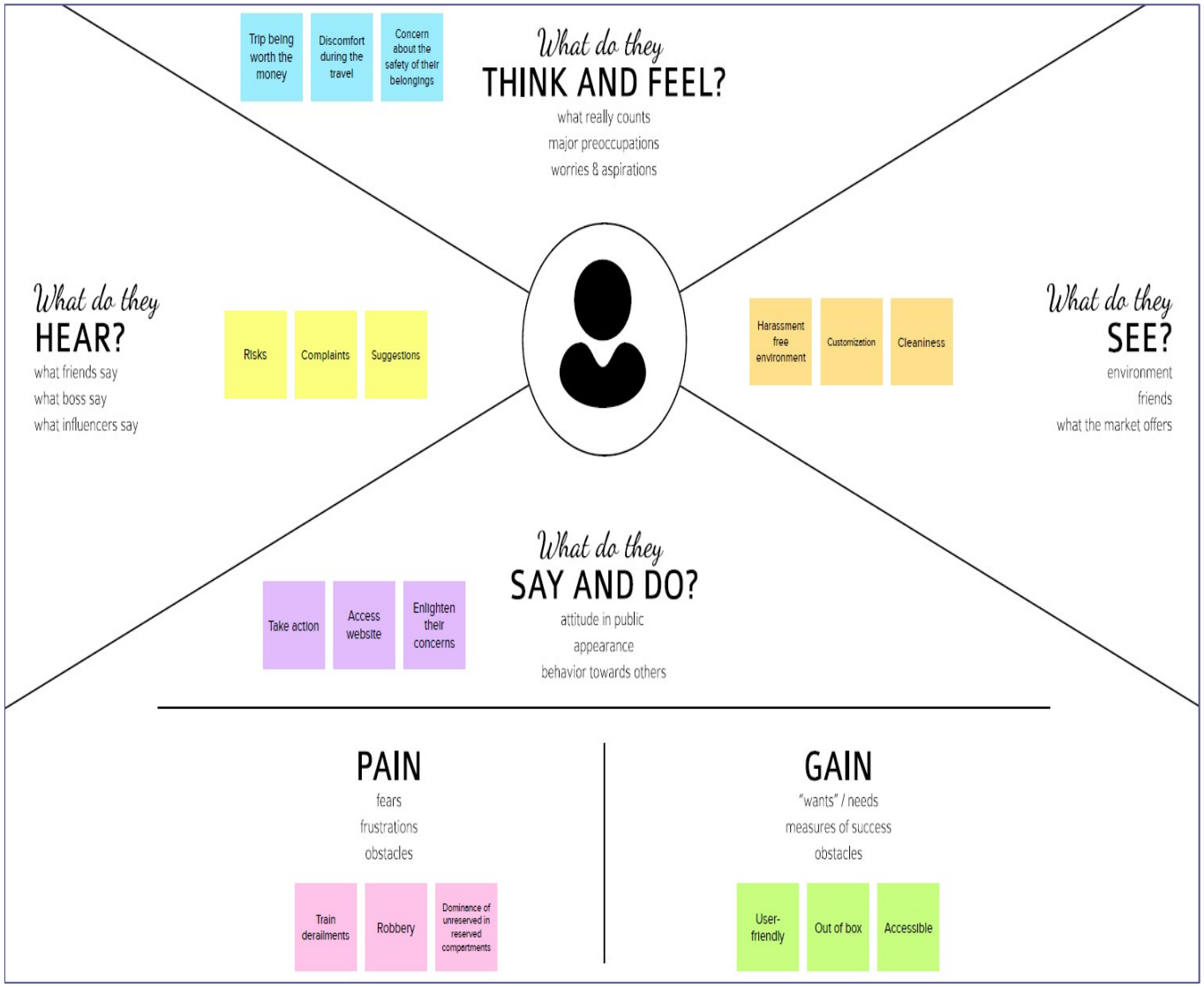
2	Construction and Building Materials	Sañudo, Roberto, Marina Miranda, Carlos García, and David García-Sanchez	2019	Drainage in railways
3	Problems of Indian Railways	Benjamin	2021	Common problems in Indian railways
4	A comparative study of Indian and worldwide railways.	Sharma, Sunil Kumar, and Anil 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.

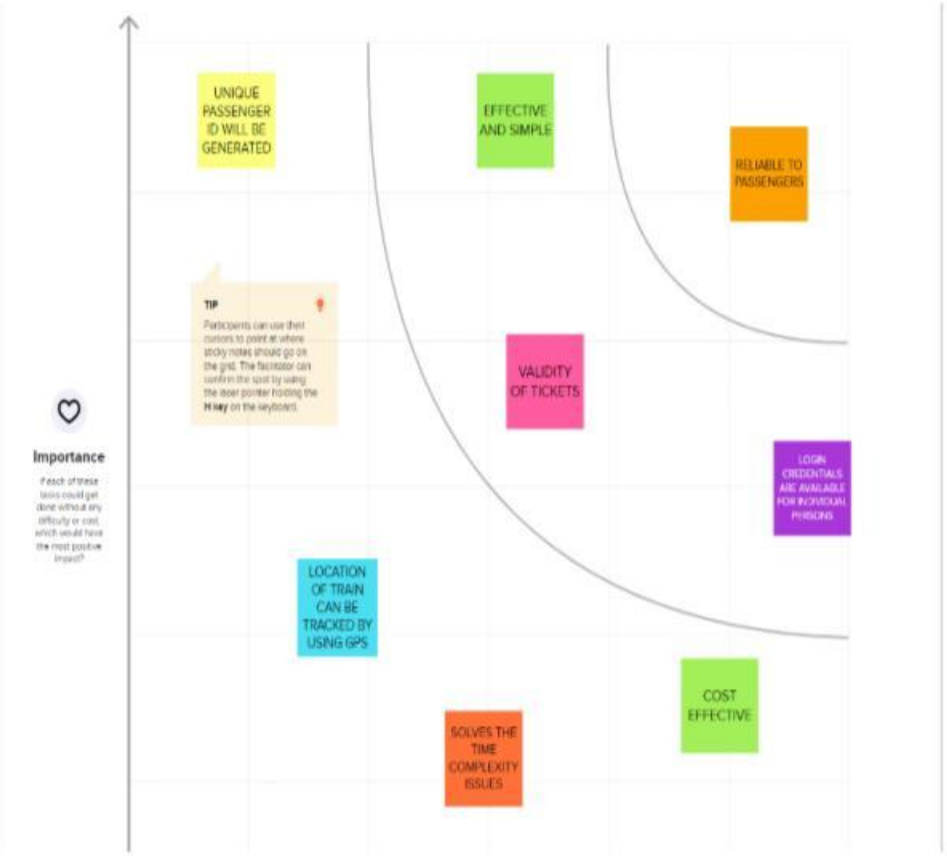
3. IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS



3.2 IDEATION & BRAINSTORMING

IDEA PRIORITIZATION:



GROUP IDEAS:

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

TICKET

EFFICIENT BOOKING SYSTEM	VALIDITY OF TICKETS	INTEGRATED TICKETING	ONLY REGISTERED USER CAN BOOK TICKETS

DATA SECURITY

ARE DATAS SECURED	DATAS ARE CONFIDENTIAL	DATA WILL BE SAFE

GPS LOCATION TRACKING

GPS FACILITIES	GPS FACILITY IS USED FOR VALIDATION OF TICKETS	LOCATION OF TRAIN CAN BE TRACKED BY USING GPS MODULE

UNIQUE ID FOR PASSENGERS

UNIQUE ID IS ISSUED	UNIQUE PASSENGER ID WILL BE GENERATED	LOGIN CREDENTIALS ARE AVAILABLE FOR INDIVIDUAL PERSONS

Write down any ideas that come to mind that address your problem statement:

R. KRISHNAVENI

CASE OF ACCESSIBILTY	UNIQUE ID IS ISSUED
DATAS ARE SECURED	VALIDITY OF TICKETS

K. BAKKIYALAKSHMI

REDUCES LABOUR WORK	EFFICIENT BOOKING SYSTEM
QR CODE CAN BE BOUGHT EASILY ANYTIME	GREATER RELIABILITY AND SAFETY

A. PRIYA ANTONY

DATAS ARE CONFIDENTIAL	INTEGRATED TICKETING
USER FRIENDLY	TICKET AVAILABILITY CAN BE ACCESSABLE

R. KOWSALYA

LOCATION OF TRAIN CAN BE VIEWED USINGGPS MODULE	EFFICIENT AND SIMPLE
QR CODE ACCESSIBILITY	COST EFFICIENT

A. PRINCY DIANA

BIOMETRIC SCANNING	VERIFICATION USING AADHAR CARD
FACE DETECTION SYSTEM TO RECOGNIZE TRAVELLER	SUPPLYING A CARD WITH SENSOR HAVING PASSENGER'S INFORMATION

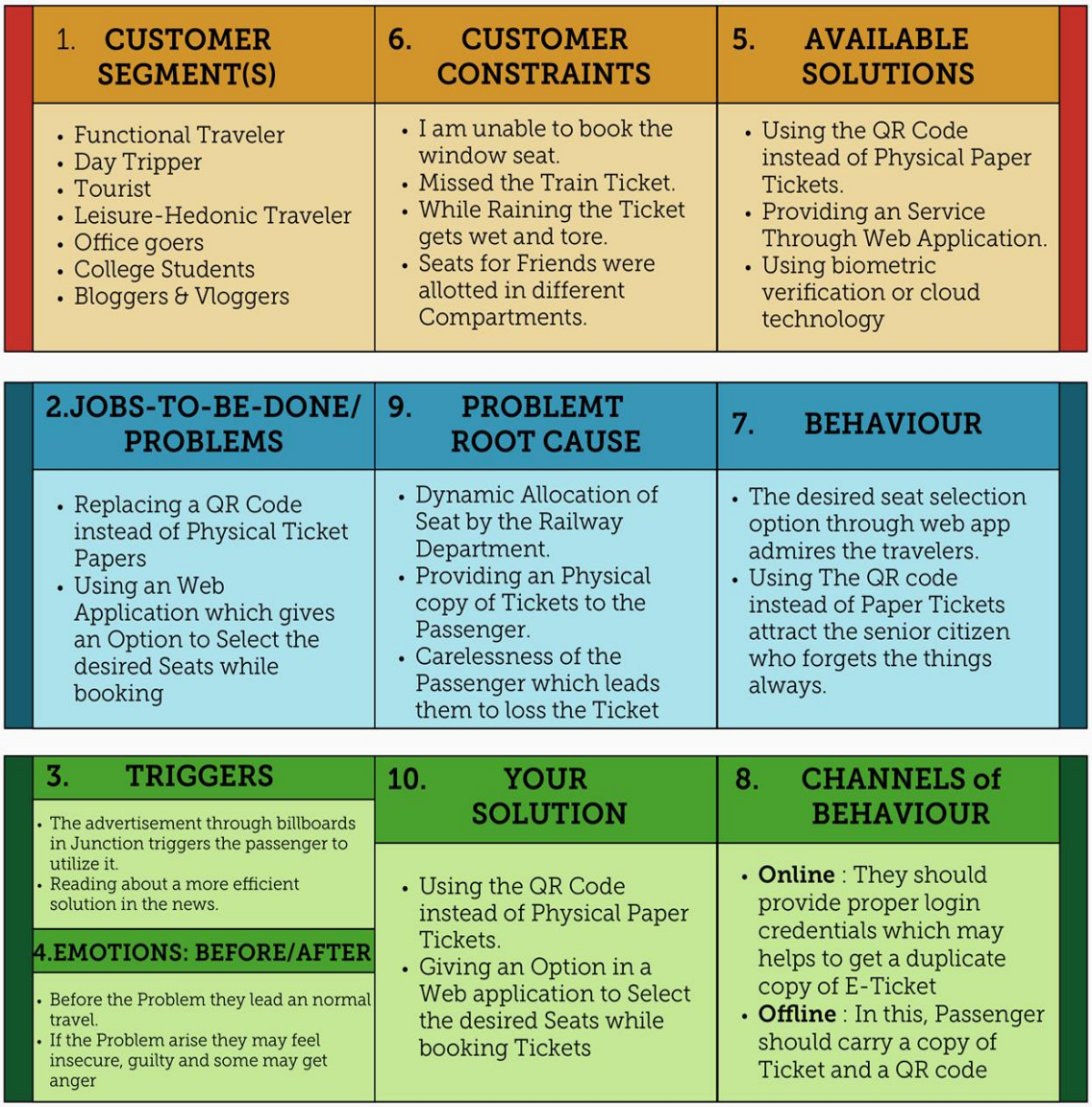
3.3 PROPOSED SOLUTION

S. No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	➢ The passenger experience is not sufficient or convenient with ticket reservations made at the counter. The travellers struggle to obtain tickets from the ticket desks in a timely manner. ➢ The printed tickets may be misplaced or overlooked in the crowd, and they may also be erased by moisture, which is a problem for the traveller. The usage of paper tickets was to blame for this. ➢ Additionally, passengers encounter the problem of being unable to reserve the preferred seat. Family members and friends Travelers were divided and assigned to separate compartments because the seats were distributed dynamically.
2.	Idea / Solution description	➢ The user can book tickets on a website, where they will also receive a QR code that they can display to the ticket collector so that the ticket collector can scan it to retrieve the user's information. ➢ By installing a GPS module inside the train, the website also displays the train's real-time positions. The journey's location will be regularly updated on the website. ➢ Additionally, the website enables users to reserve the desired seat.

3.	Novelty / Uniqueness	➢ The webpage will offer the customer a QR code, which will cut down on paperwork. ➢ It allows the user to reserve the preferred seat. ➢ All of the client booking information will be saved in the database with a special ID and may be retrieved when the ticket collector scans the QR Code.
4.	Social Impact / Customer Satisfaction	➢ There is no need to go to the station to book tickets because they can be booked online, and the transaction process is also made simple. ➢ All confirmations and cancellations will be sent to the consumer by email or mobile phone.
5.	Business Model (Revenue Model)	➢ The user of this application can plan their trip, check seat availability, browse an interactive seat map, and select a seat at their convenience. Additionally, it makes it simple for your clients to schedule daily shuttles and journeys, and it eliminates carrying around tickets. The customer can also view the train's current location. ➢ For using the abovementioned facility, a specific amount of fees may be charged, particularly if a customer wants to reserve their preferred seat, they must pay extra for a ticket.
6.	Scalability of the Solution	➢ Elimination of physical paper tickets ➢ Although counter tickets must be handled carefully, text messages on a phone are more than sufficient. ➢ You are becoming environment friendly and contributing for

		greener planet by eliminating paper copy. ➢ While booking counter ticket you had to carry cash and while booking E- ticket you are paying through online directly from bank which makes work easier for you.

3.4 PROBLEM SOLUTION FIT:



4. REQUIREMENT ANALYSIS :

4.1 Functional requirement

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Passenger ticket booking	Booking through the online railway mobile app and website.
FR-2	Booking Confirmation	Booking Confirmation via Email Booking Confirmation via SMS
FR-3	Passenger objections and feedback	Through the online application, SMS, and email to the respective authority.
FR-4	Passenger schedule	Passenger can see their train timing through the mobile app
FR-5	Passenger Emergency	Passengers in an Emergency, in case of accidents, natural disasters, or theft during the journey can complain through online applications, emergency calls, SMS, and email.

4.2 Non-Functional requirement

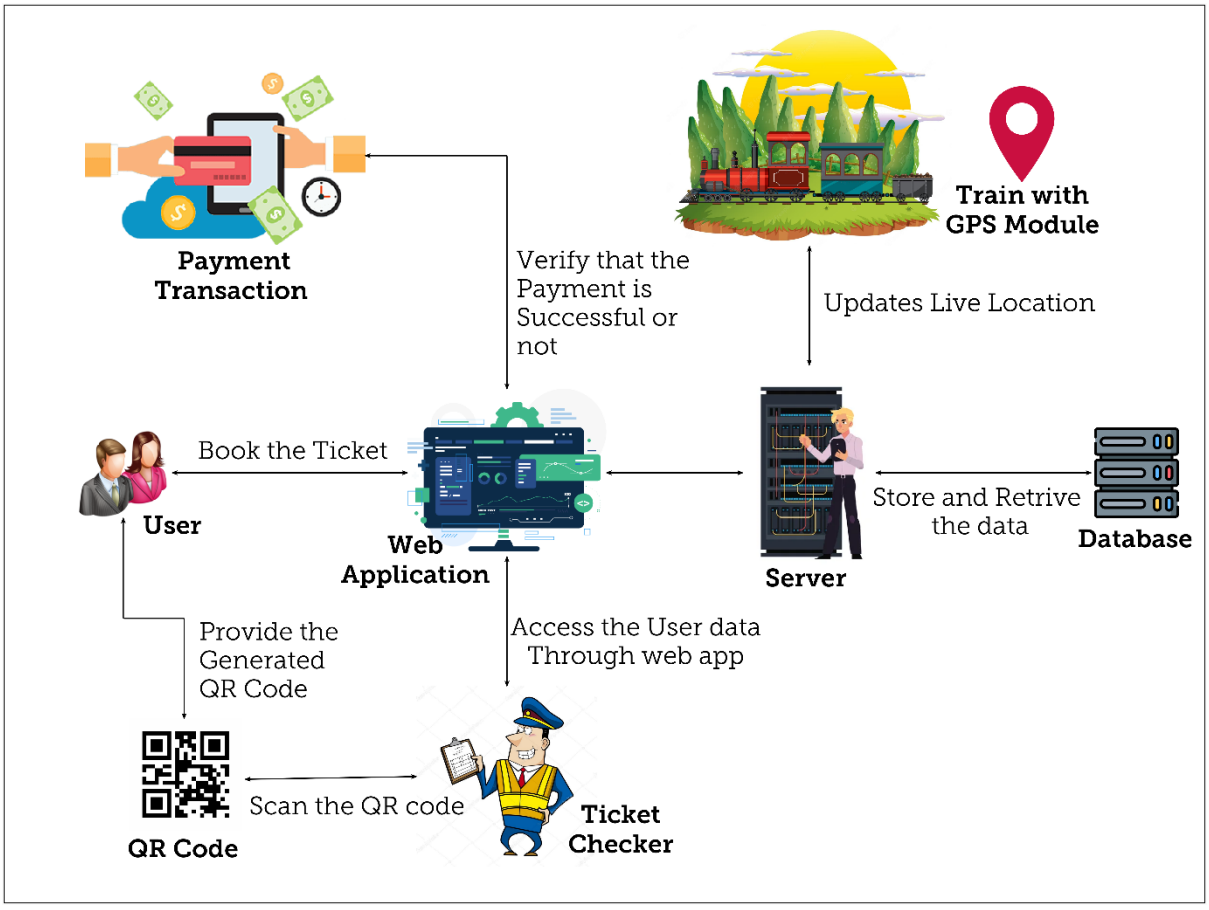
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Within periodic maintenance, we can detect cracks in the railway track. which will be highly usable on remote railway tracks.

NFR-2	**Security**	Accidents and property damage can be prevented with the help of our smart sensors which immediately send the fault to the pilot and administration.
NFR-3	**Reliability**	Traffic lights and signalling can be made accurately with the help of sensors. so it is more reliable.
NFR-4	**Performance**	Communication plays a vital role in transferring the crack-detected signal to the responsible authority so that they can take appropriate measures within a short span.
NFR-5	**Availability**	Our idea is to make the crack alert to all the trains passing through that fault-prone area.

NFR-6	**Scalability**	Our project is based on IoT & cloud, which makes the pilot and authority updated every single sec. Adhoc is easy to handle.

5. PROJECT DESIGN

5.1 DATA FLOW DIAGRAMS



5.2 Solution Architecture

As trains are one of the most preferred modes of transportation among middle 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 all such brutality we came up with a solution by providing an application which can be accessed 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
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

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

7. CODING & SOLUTIONING

7.1 Feature 1

- IoT device
- IBM Watson Platform
- Node red
- Cloudant DB
- Web UI
- MIT App Inventor
- Python code

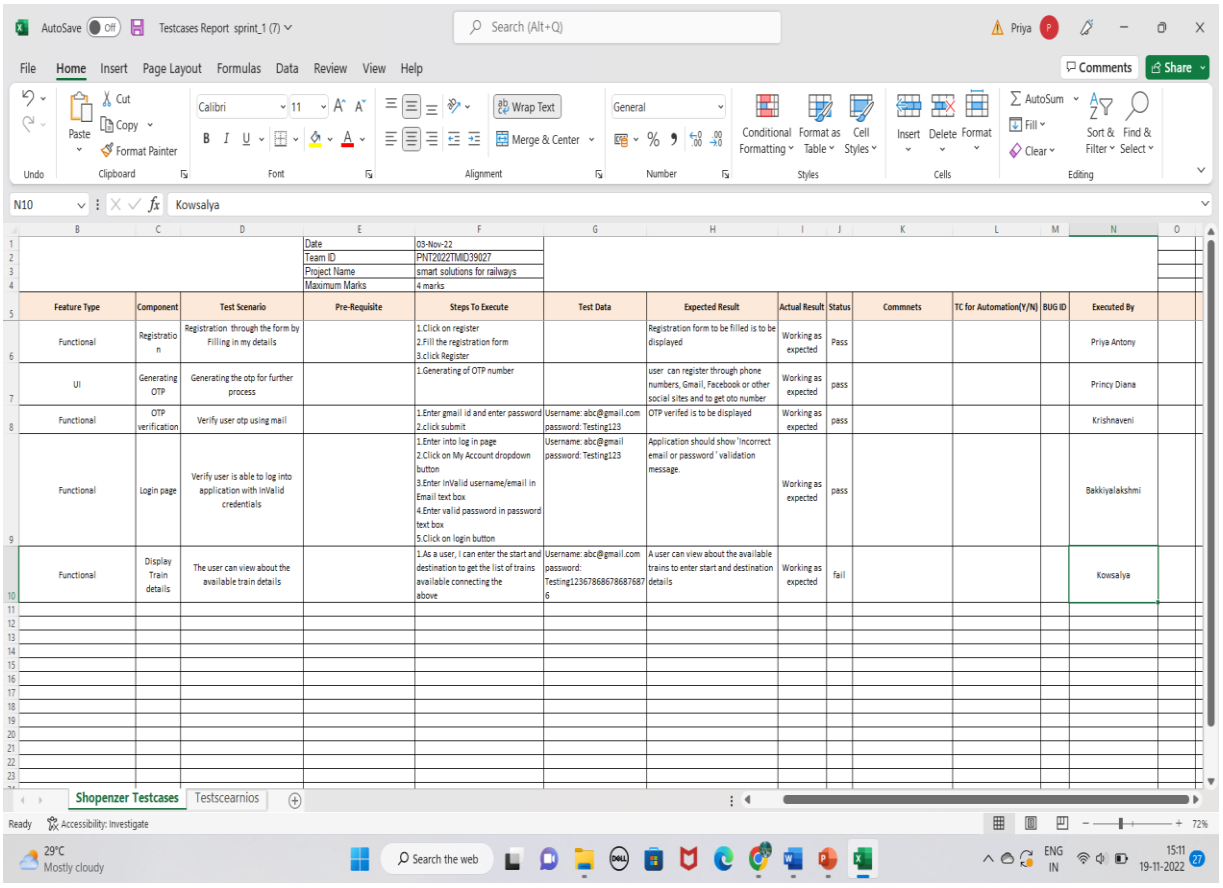
7.2 Feature 2

- Login
- Verification
- Ticket Booking
- Adding rating

8. TESTING AND RESULTS

8.1 Test Cases

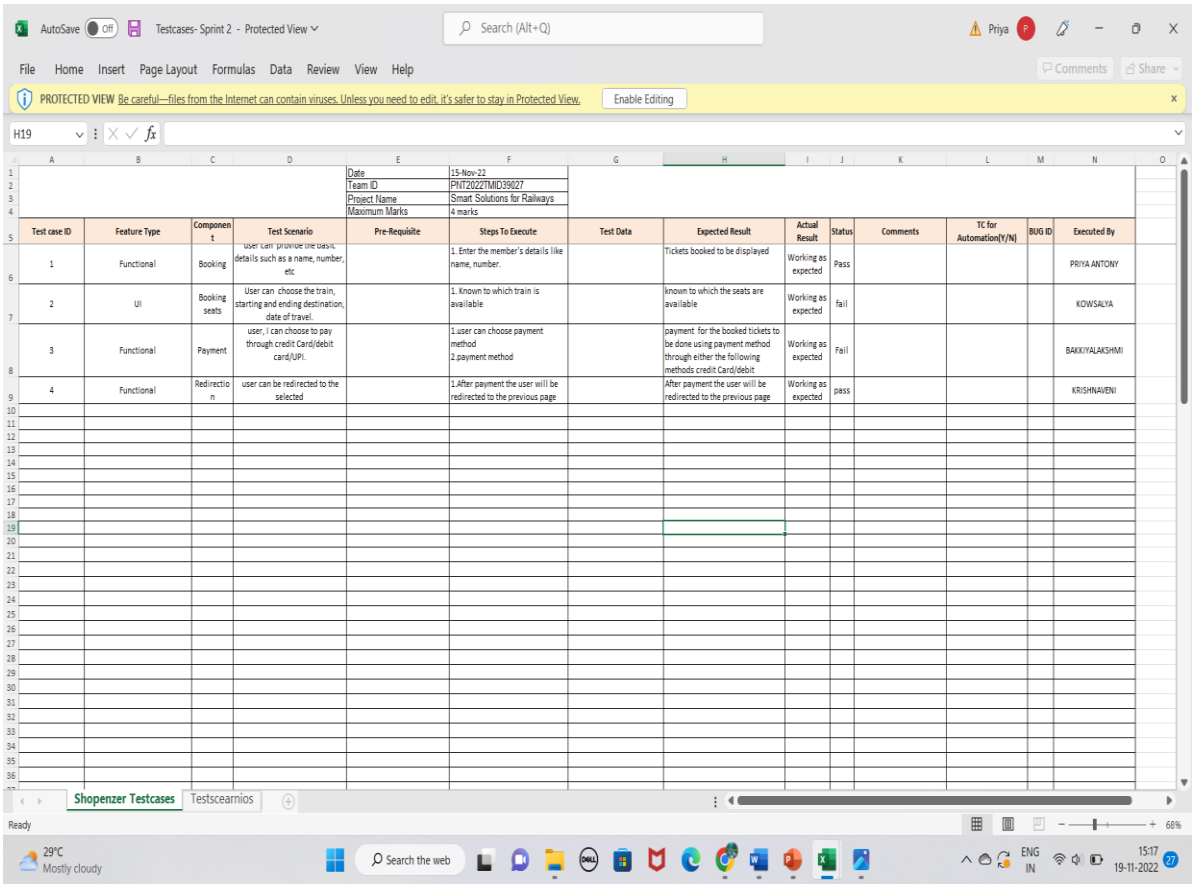
Test case 1



Date	03-Nov-22
Team ID	PNT2022TMID39027
Project Name	smart solutions for railways
Maximum Marks	4 marks

Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Commnets	TC for Automation(Y/N)	BUG ID	Executed By
Functional	Registration	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				Priya Antony
UI	Generating OTP	Generating the otp for further process		1.Generating of OTP number		user can register through phone numbers, Gmail, Facebook or other social sites and to get oto number	Working as expected	pass				Princy Diana
Functional	OTP verification	Verify user otp using mail		1.Enter gmail id and enter password 2.click submit	Username: abc@gmail.com password: Testing123	OTP verifed is to be displayed	Working as expected	pass				Krishnaveni
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 5.Click on login button	Username: abc@gmail password: Testing123	Application should show 'Incorrect email or password ' validation message.	Working as expected	pass				Bakkiyalakshmi
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: abc@gmail.com password: Testing1236786878687687 6	A user can view about the available trains to enter start and destination details	Working as expected	fail				Kowsalya

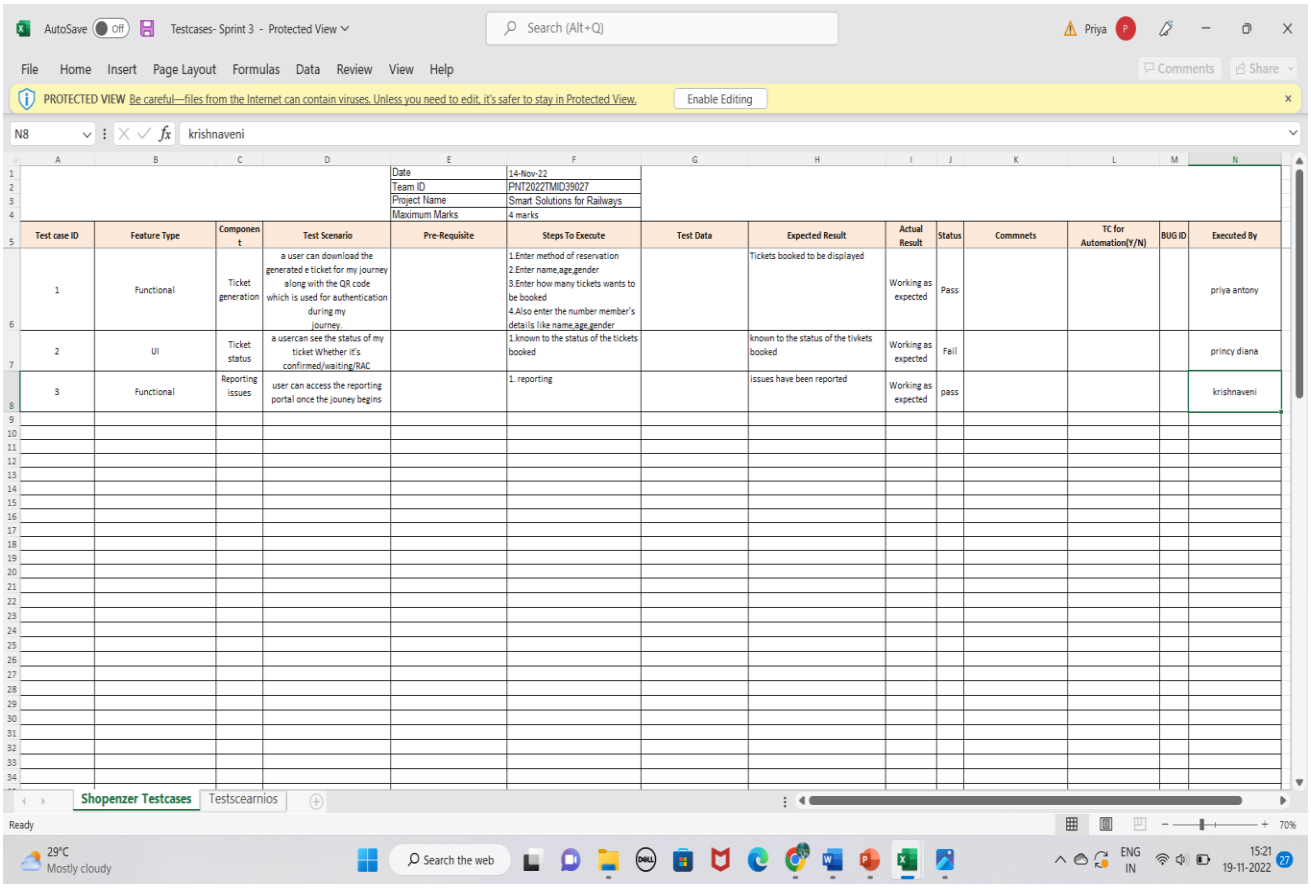
TEST CASE 2



Date	15-Nov-22
Team ID	PNT2022TMID39027
Project Name	Smart Solutions for Railways
Maximum Marks	4 marks

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Comments	TC for Automation(Y/N)	BUG ID	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				PRIYA ANTONY
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				KOWSALYA
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				BAKKIYALAKSHMI
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				KRISHNAVENI

Test case 3



Date	14-Nov-22
Team ID	PNT2022TMID39027
Project Name	Smart Solutions for Railways
Maximum Marks	4 marks

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Commnets	TC for Automation(Y/N)	BUG ID	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				priya antony
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				princy diana
3	Functional	Reporting issues	user can access the reporting portal once the jouney begins		1. reporting		issues have been reported	Working as expected	pass				krishnaveni

9. ADVANTAGES

- The passengers can use this application, while they are travelling alone to ensure their safety.
- It is easy to use.
- 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.Ticket Reservation")
 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 def

```
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 import pandas
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 in soup.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"              or
self.__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.__source
else:
           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 random
import 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")
```

OTP VERIFICATION

```
import        os
import     math
import random
import 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 GitHub

GitHub link:

https://github.com/IBM-EPBL/IBM-Project-14728-1659589144