Project Report

Project Name: **SMART SOLUTIONS FOR RAILWAYS**
Team ID: **PNT2022TMID25665**

TEAM LEAD:

SANTHA KUMAR.D

TEAM MEMBERS :

SASI KUMAR.V

SANTHOSH.D

SRIRAM.A

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, appdevelopment, IBM cloud platform to store passenger data. **1.2 Purpose**

SMART SOLUTION FOR RAILWAYS

Date	1 October 2022
Domain Name	Internet Of Things (IoT)
Project Name	***Smart solution for railways***
Team ID	PNT2022TMID25665

Objective:

Paper pen works takes time and can be time consuming. People in this current fast world won't like to still stand in a queue and book tickets. And most importantly, nowadays most people have a smartphone on their own. So that implementing the solution proposed won't be a challenging task. In fact, compared to the traditional method of booking tickets standing in a queue, our method has a lot of advantages.

Our solution is to design a website where we can book ticket and receive QR Code which can be scanned during boarding. Passengers can also monitor the train status and as well as they are alerted through mobile before their destination arrives.

There won't be any loss in implementing this solution proposed by us. In fact, the truth is the **Tamil Nadu government has been already involved in developing and promoting the solution proposed by us by providing 60% discounts to those passengers who book tickets through online** in Chennai Metro Rail Service (CMRL) understanding the advantages.

And Finally, Digitizing the booking and verification process & alert passenger before their destination arrives. Digitizing the works reduces manual paper pen work and it becomes easier and time saving. Because of the smart changes that can be introduced by implementing in the railway system, **Our Indian Railways can be called as "SMART RAILWAYS".**

LITERATURE SURVEY

PAPER NAME	AUTHOR	YEAR	METHOLOGY	MERITS	DEMERITS
Passenger Monitoring Model for easily Accessible Public City Trams/Trains.	Roman Khoeblal, Teeravisit Laohapensaeng, Roungsan Chaisricharoen	2015	Passenger monitoring, passenger control RFID distance reading, ticket control, RFID ticket inspection.	It is possible to travel cross country with a single public transportation card, using transport systems of several transport operators.	Applicable only for passenger monitoring.
Application of smart computing in Indian Railway Systems.	Parag Chatterjee, Asoke Nath	2014	By Interlinking unique identification system with train ticket reservation system by using video surveillance, rail sensors, biometric input devices and multimedia displays.	Reduces manual effort in passenger data entry. Provides security verification.	Significant investment is needed. Risk of database.
Android Suburban Railway Ticketing with GPS as Ticket Checker.	Sana Khoja, Maithili Kadam	2012	Android, SQ lite, Cloud Database, ASR, QR Code.	E-Ticket facility, enabling reuse and replacement of components.	QR Codes before the user enters or leaves the station, where the user can have access which is risk in ticket booking.

Novel Approach for Smart Indian Railways.	Sujith Kumar, K.M.Yatheendra Parvan, V.Sumathy, Thejeswari C.K	2017	Digitalization, Smart Railways, Aadhar Card, Smartphone, Identity Verification.	Employ a mobile application through which passengers can access various ticketing options in user friendly and efficient manner.	Biometric database is risk of hacking.
A Review on IOT based automated seat allocation and verification using QR code.	Sarvath Saba, Sharon Philip, Shriharsha, Mukund Naik, Sudeep Sherry	2022	The system lets the passenger to have a comfortable journey by checking the temperature first for normal and then the count for avoid crowd using the QR Code.	This model proposes a radical change in train operation and passenger experience. One of the many steps towards a more digitized society as a part of the "Digital India" movement proposed in 2015 by the Prime Minister.	The system is not fool-proof and requires a dramatic change in the existing system in terms of the people allowed on platforms, etc. but baby steps matter.

Reference:

1. Roman Khoeblal, Teeravisit Laohapensaeng, Roungsan Chaisricharoen, "Passenger Monitoring Model for easily Accessible Public City Trams/Trains" (2015).
2. Parag Chatterjee, Asoke Nath, "Application of smart computing in Indian Railway Systems" (2014).

3. Sana Khoja, Maithili Kadam, “Android Suburban Railway Ticketing with GPS asTicket Checker” (2012).

4. Sujith Kumar, K.M.Yatheendra Parvan, V.Sumathy, Thejeswari C.K, “NovelApproach for Smart Indian Railways” (2017).

5. Sarvath Saba, Sharon Philip, Shriharsha, Mukund Naik, Sudeep Sherry, “A Reviewon IOT based automated seat allocation and verification using QR code”(2022).

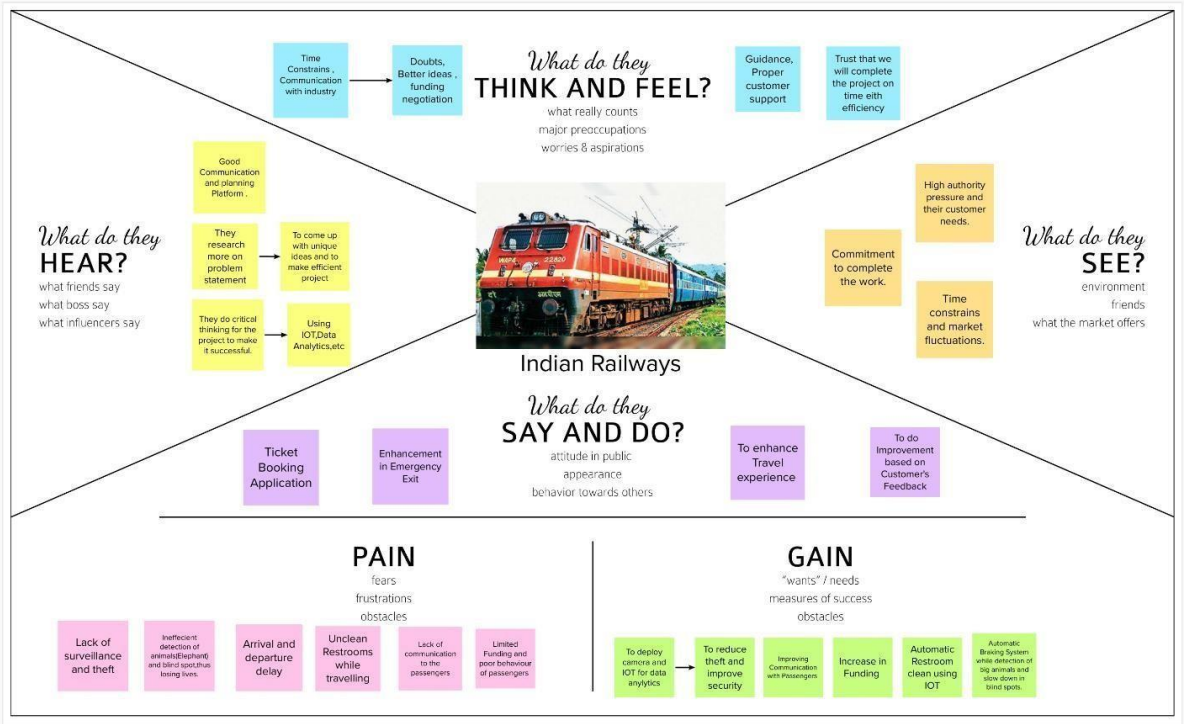
Ideation Phase Empathize & Discover

Date	19 September 2022
Team ID	PNT2022TMID25665
Project Name	Smart solution for Railways

Empathy Map Canvas:

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user'sbehaviors and attitudes.

Reference:
https://app.mural.co/invitation/mural/smartinternzibmiotsmartsolut6184/16627 90391718? sender=ub72a907284043284ab647148&key=b27221cd-3c1b-44a4-bd15-1947bfd7faff



PROJECT DESIGN PHASE-II

SOLUTION REQUIREMENTS (FUNCTIONAL & NON-FUNCTIONAL)

Team ID	PNT2022TMID25665
Project Name	Project - Smart Solutions for Railways

FUNCTIONAL REQUIREMENTS:

Following are the functional requirements of the proposed solution.

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.

NON-FUNCTIONAL REQUIREMENTS:

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

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.

Project Design Phase-II Data FlowDiagram & User Stories

Date	15 October 2022
Team ID	PNT2022TMID25665
Project Name	Project - Smart Solutions for Railways

Data Flow Diagrams:

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and wheredata is stored.

Example: **(Simplified)**

Step 1: The work here starts during the first time installation of our application where the user has to sign up. During sign up the basic customer information like first name, last name, date of birth, mobile no, city, state etc., will be gathered and it will be stored into MySQL database. So every time when the user buys the ticket this customer information is sent to the database for security purpose and also the ticket is generated accordingly. During sign up the username will be set as the user's mobile number or Email-id and the password will be as per the choice of the user. On the other hand if the user has an account then he can sign in directly. Thus the user can use different android phones and will not be restricted to only his phone. The above information will be send to server with the help of internet.

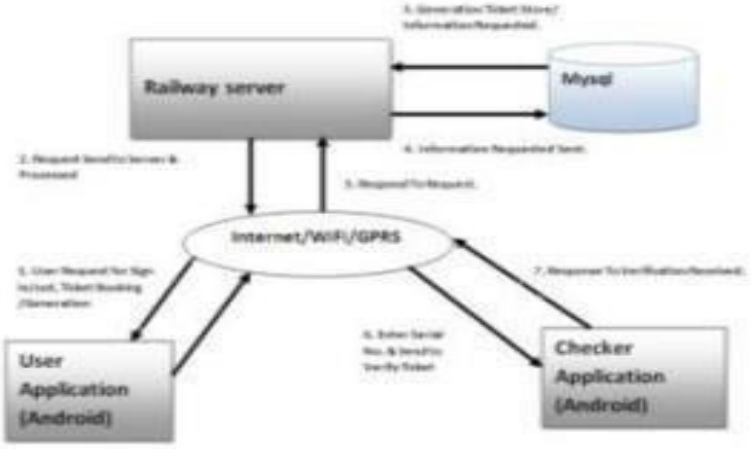


Fig 1. System Architecture

Step 2: The user scan Qr-code for source and select destination, number of tickets, single or return journey. Then the user is directed to the payment option. Payment can be done through prepaid services, i.e. the balance of the mobile no will be displayed along with the cost of the ticket and if the user agrees to proceed then the equivalent 'amount' of the ticket will be deducted from the balance of the mobile no.

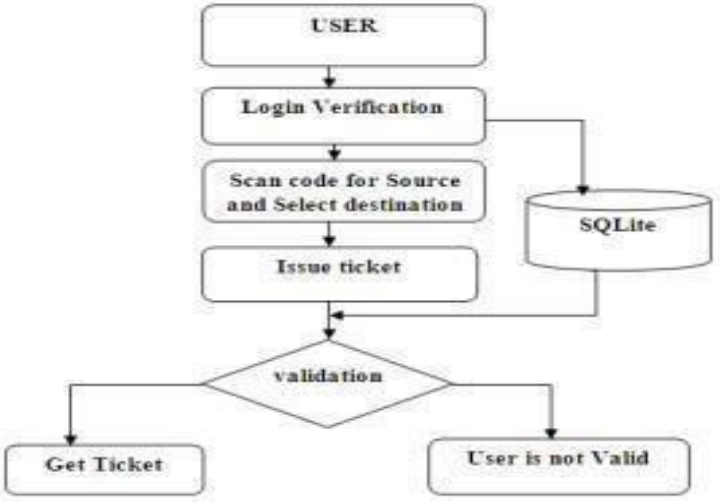


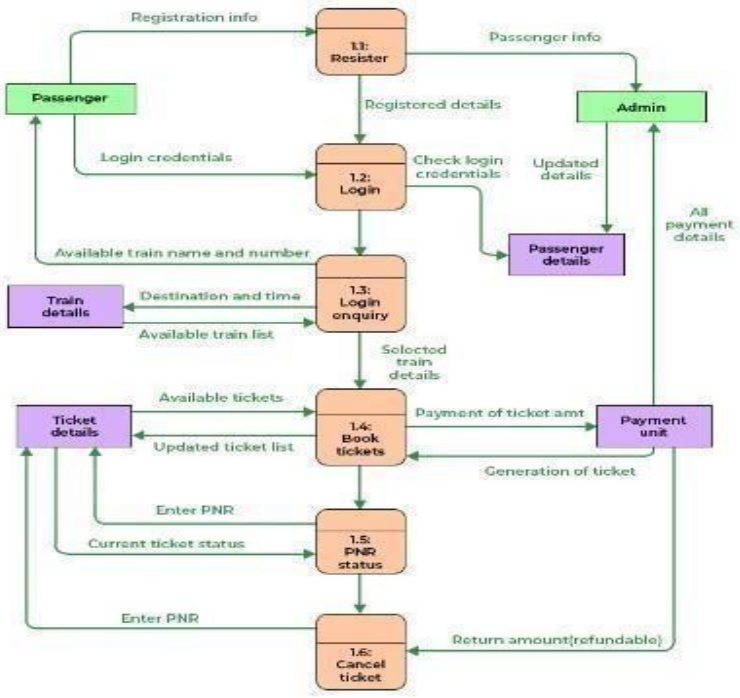
Fig 2. Flow Process of Ticket Booking.

Step 3: Once the customer click the buy button a code in the railway server validates the pin number and passwords, if it is successful it saves both the journey details and customer info in the server's MySQL database.

Step 4: The code on the server side generates the time of buy and the expiry timing of the ticket; the details are saved in the railway's MySQL database. Then Ticket no. is generated on server side, saved in the database and also sent back to the user mobile and saved in the application memory which serves as a ticket for the user.

Step 5: In this module the checker will enter the Ticket no. which will validate and verify the journey details from the railway database, especially the time and date of the ticket.

Example: Data Flow Diagram for Reserving ticket



User Stories

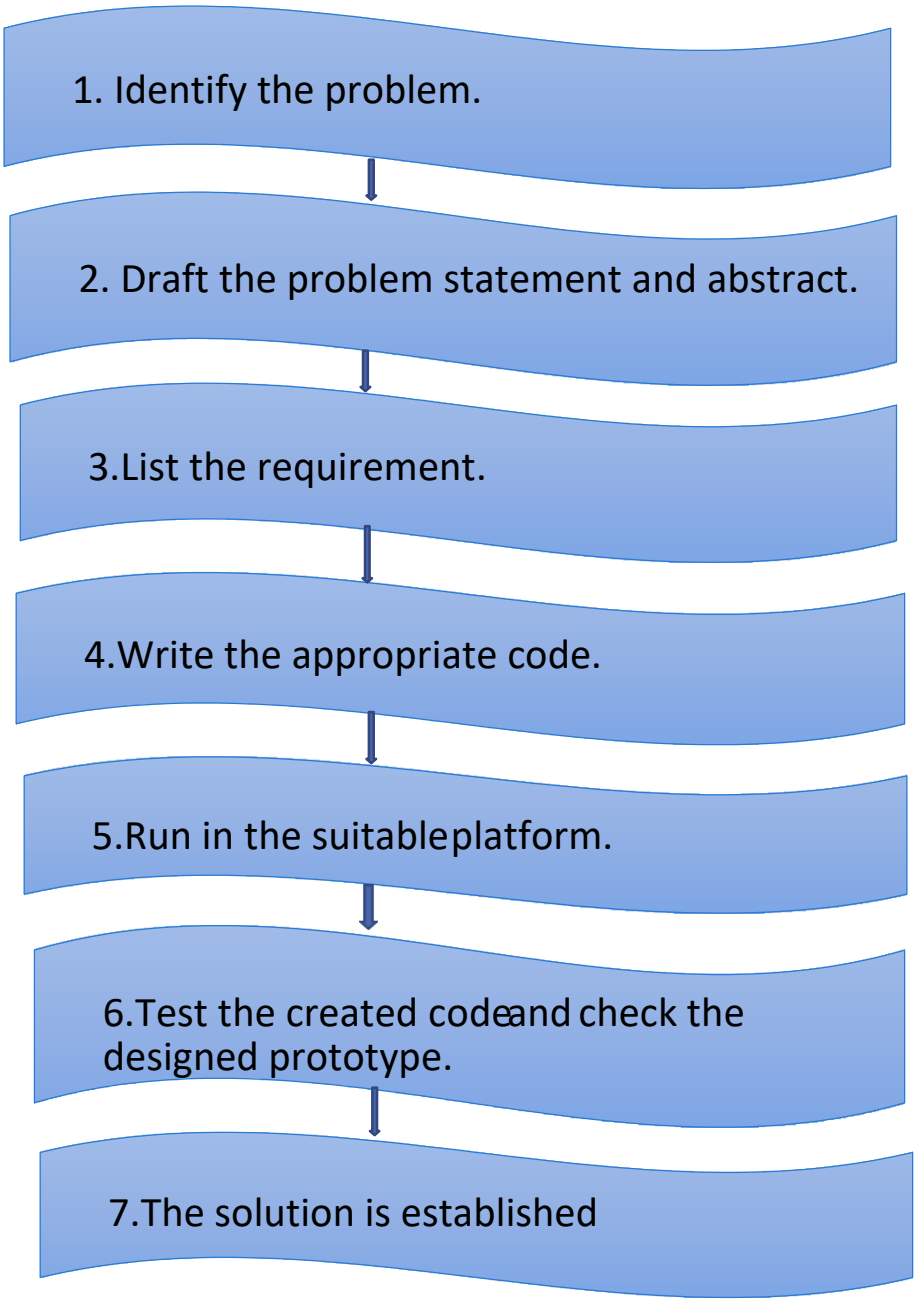
Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Reserving ticket	USN-1	As a user, I can register for the application by entering my email, password, and confirmingmy password.	I can access my account / dashboard	High	Sprint-1
Customer (Mobile user)	Reserving ticket	USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1

Customer (Mobile user)	Reserving ticket	USN-3	As a user, I can register for the application and enter the details for reserving the ticket.	I can register & dashboard wit Login
Customer (Mobile user)	Dashboard	Users	The details will be stored safely	I can access it database
Customer (Web user)	Reserving ticket	User	Enter the details and click submit button to book ticket	I can use the C which is been
Customer Care Executive	Connecting the service provider	Customer	Connects with the service by logging in	Can get conne the server
Administrator	Provides the services	Admin	The data is given by the user	Can add or up provided by th

SPRINT DELIVERY PLAN

Date	28 October 2022
Team ID	PNT2022TMID53836PNT2022TMID25665
Project Name	Smart Solutions for Railways



PROJECT DEVELOPMENT DELIVERY OF SPRINT-1

TEAM ID	PNT2022TMID25665
PROJECT NAME	SMART SOLUTION FOR RAILWAYS

SPRINT-1

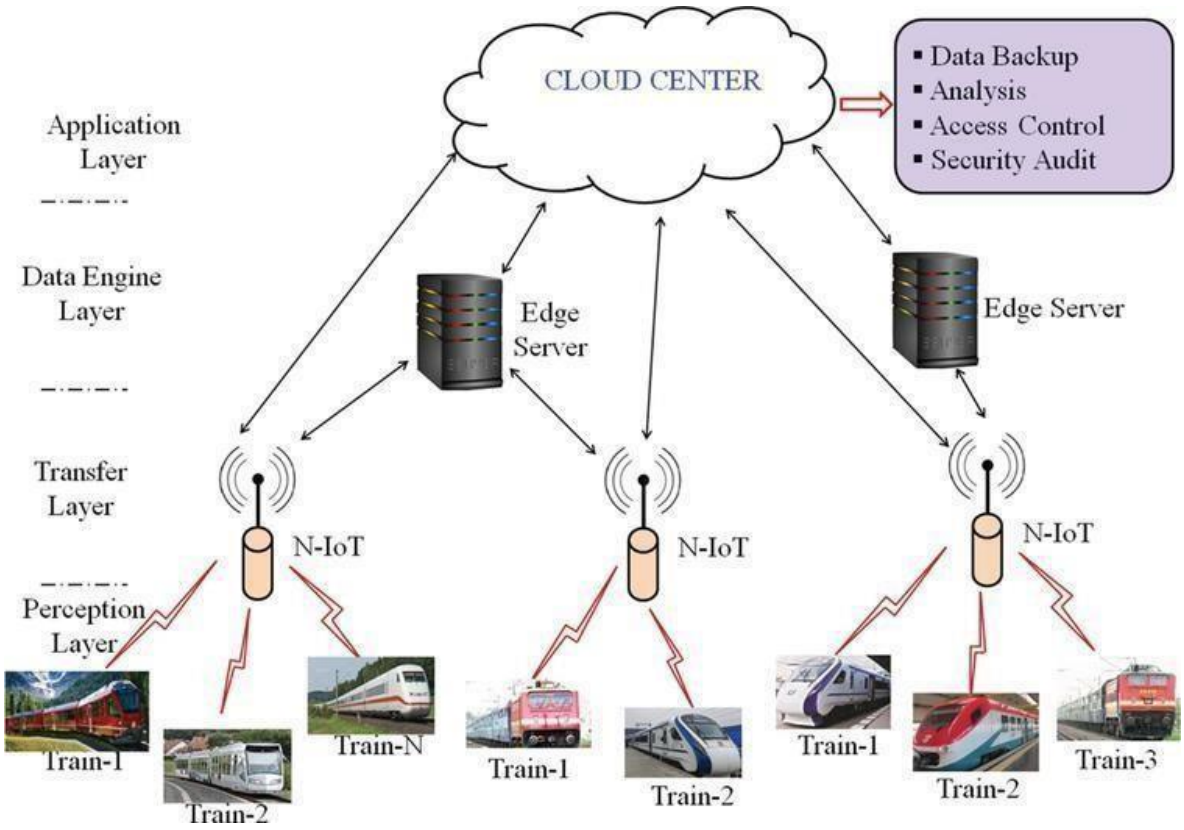
```
#include <LiquidCrystal.h>

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);
      }
}
```

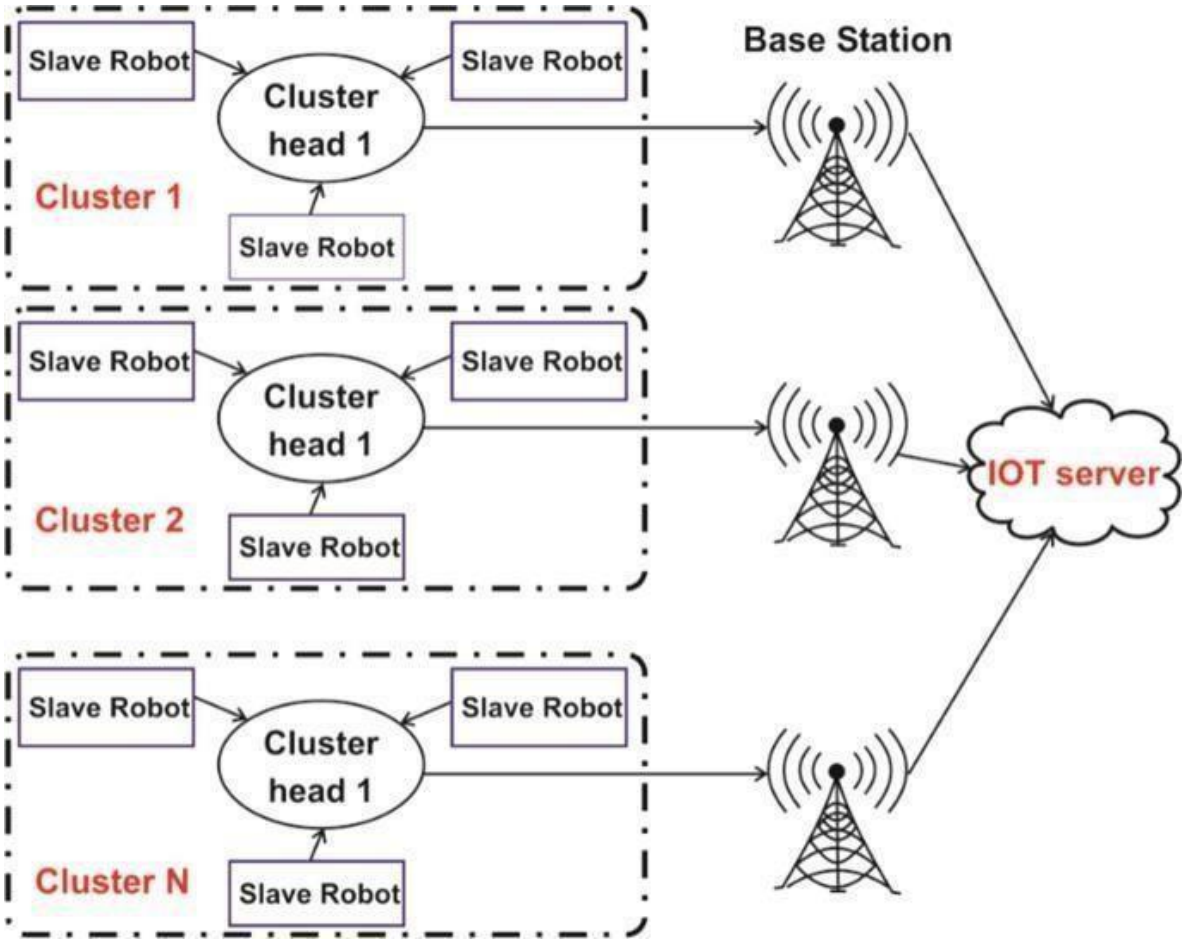
PROJECT DEVELOPMENT DELIVERY OF SPRINT-2

TEAM ID	PNT2022TMID25665
PROJECT NAME	SMART SOLUTIONS FOR RAILWAYS

Proposed architecture for smart track monitoring system.



Structural health monitoring of railway tracks using IOT-based multi robot system:



PROJECT DEVELOPMENT DELIVERY OF SPRINT-3

TEAM ID	PNT2022TMID25665
PROJECT NAME	SMART SOLUTIONS FOR RAILWAYS

Efficient and Secure Operation with Better Passenger Experience:

Trams and trains are an indispensable part of urban transportation. With continuous waves of urbanization, it is required that operation of the rail transit system be more intelligent and efficient. As those vehicles run in separate and closed tracks, railways entail a number of challenges and risks in terms of security and management, including engines, doors, fire control, as well as vibration and electromagnetic disturbance. Making sure everything is in good order can be a major challenge. Danger must be identified as soon as possible, as accidents mean not only revenue losses, but more importantly, life security.

Meanwhile, with a large amount of time spent on the way, travel is expected to be not only convenient but also pleasant – especially in the IOT era where everything can and should be connected. Better passenger experience is the key to success in the increasingly fierce market competition.

Thus, a full-functional communications system is needed for rail transportation.

In Hand's Connectivity Solution for Rail Transit:


Engine
CCTV
Door
Fire Control
Driver Panel
Ticketing
Infotainment
Wi-Fi
GNSS
IVG710
OBD-II
LTE CAT6/CAT4
Back Office
Driving Behavior
Vehicle Location
Vehicle Diagnostics
Speed
Security

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

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 userfriendly websites, mobile applications for real-time information about vehicles in motion, and eticket 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_screenwi
dth()  screen_height
=
root.winfo_screenhei
ght()  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
HomeWind
ow():
global
Home
root.withdraw()     Home
= Toplevel()
Home.title("Python:
```

```
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)
```

```
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
```

```
% 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:
```

```
 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-clonespecial-rgd-to-ndls"

# url  url = "https://www.railyatri.in/live-trainstatus/"+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.__passenger_name     def
 get_source(self):         if
 self.__source=="Delhi":
            return         self.__source
 else:
```

```
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 == 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 ==
```

```
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-26445-1660026927