Project Report

Project Name: SMART SOLUTIONS FOR RAILWAYS

Team ID: PNT2022TMID05234

TEAM LEAD:

M Kirubha

TEAM MEMBERS :

V Jana Ranchani

P Keerthika

K M Janani Bhaarathi

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 Solutions for Smart Railways for future

it can be designed to reduce the work load of the user and it is also the use of paper. Here in this project, we have all the features are like 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 personal details. A GPS module is present on 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 SURVEY:

1. Railway passenger frequently need to know about their ticket reservation status, ticket availability on a particular train or for a place, train arrival or departure details, special trains etc.
2. Customer information centers at the railway statios are unable to serve such queries at peak periods.
3. The number of the reservation counters available to the passengers and customers are very less'
4. On most of the reservation systems they are long queries, so it takes along time for any individual to book the ticket'
5. As now there are no call centers facilities available to solve the queries of the passengers.

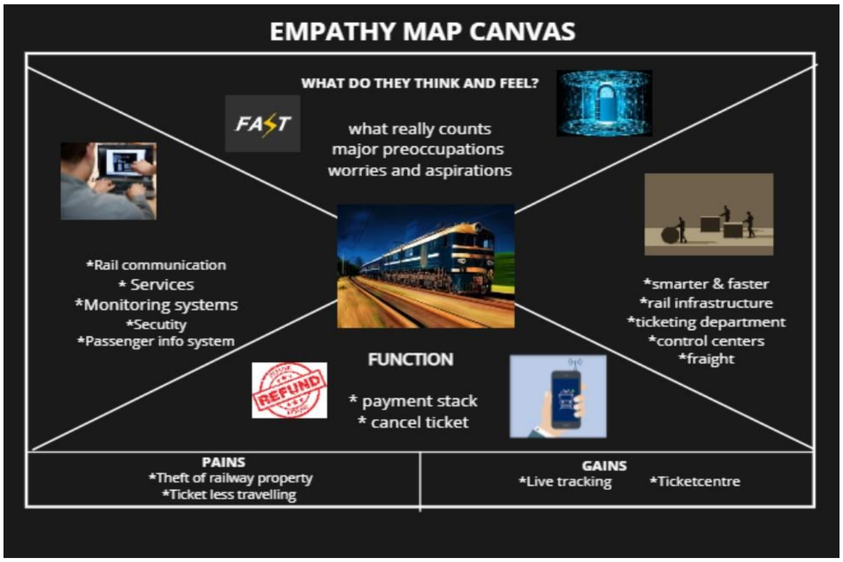
6. The online railway ticket reservation system aims to develop a web application which aims at providing trains details,train availability,as well as the facility to book ticket in online for customers,
7. So, we thought of developing the web based application which would be provide the users all these facilities from his terminal only as well as help them in booking tickets.The application was to be divided into two parts namely the user part, and the administrator part and each of these has their corresponding features.
8. Our website has various kind of information that's helps regarding booking of tickets railways.
9. It also reduce the paper work. Its totally going to be a online based.
10. In future there is going to be a ticket booking counter.

2.3 Problem Statement Definition

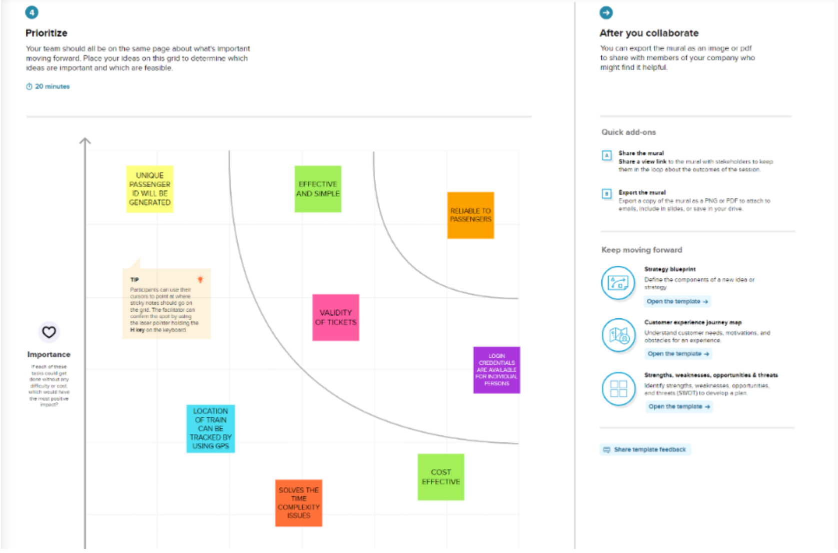
Smart Solutions for railways are designed to reduce the work load of the user and the use of paper.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



3.2 Ideation & Brainstorming



4

Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

TIP

Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the **H key** on the keyboard.

Importance

If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?

UNIQUE PASSENGER ID WILL BE GENERATED
EFFECTIVE AND SIMPLE
RELIABLE TO PASSENGERS
VALIDITY OF TICKETS
LOGIN CREDENTIALS ARE AVAILABLE FOR INDIVIDUAL PERSONS
LOCATION OF TRAIN CAN BE TRACKED BY USING GPS
COST EFFECTIVE
SOLVES THE TIME COMPLEXITY ISSUES

After you collaborate

You can export the mural as an image or pdf to share with members of your company who might find it helpful.

Quick add-ons

A **Share the mural**
Share a view link to the mural with stakeholders to keep them in the loop about the outcomes of the session.

B **Export the mural**
Export a copy of the mural as a PNG or PDF to attach to emails, include in slides, or save in your drive.

Keep moving forward

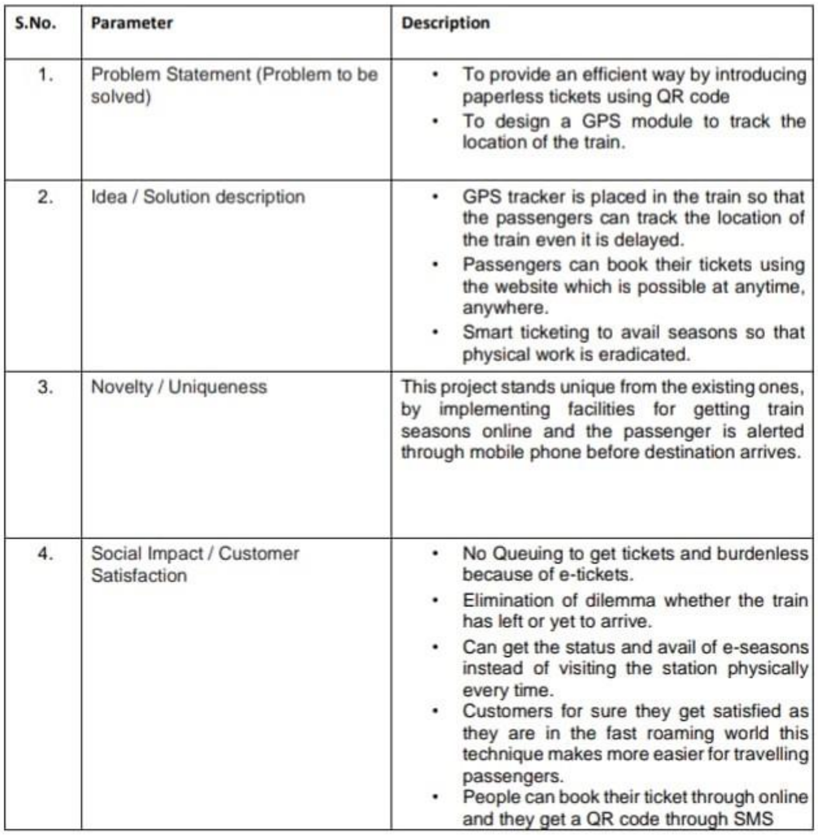
Strategy blueprint
Define the components of a new idea or strategy.
Open the template →

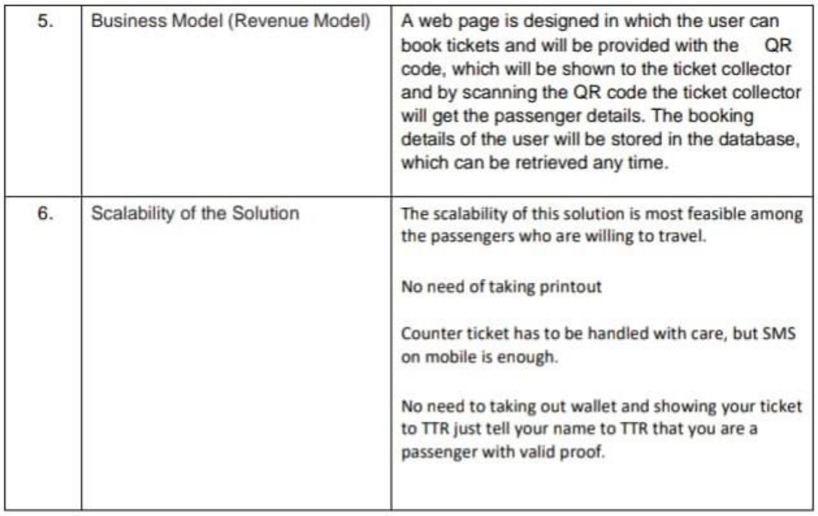
Customer experience journey map
Understand customer needs, motivations, and obstacles for an experience.
Open the template →

Strengths, weaknesses, opportunities & threats
Identify strengths, weaknesses, opportunities, and threats (SWOT) to develop a plan.
Open the template →

Share template feedback

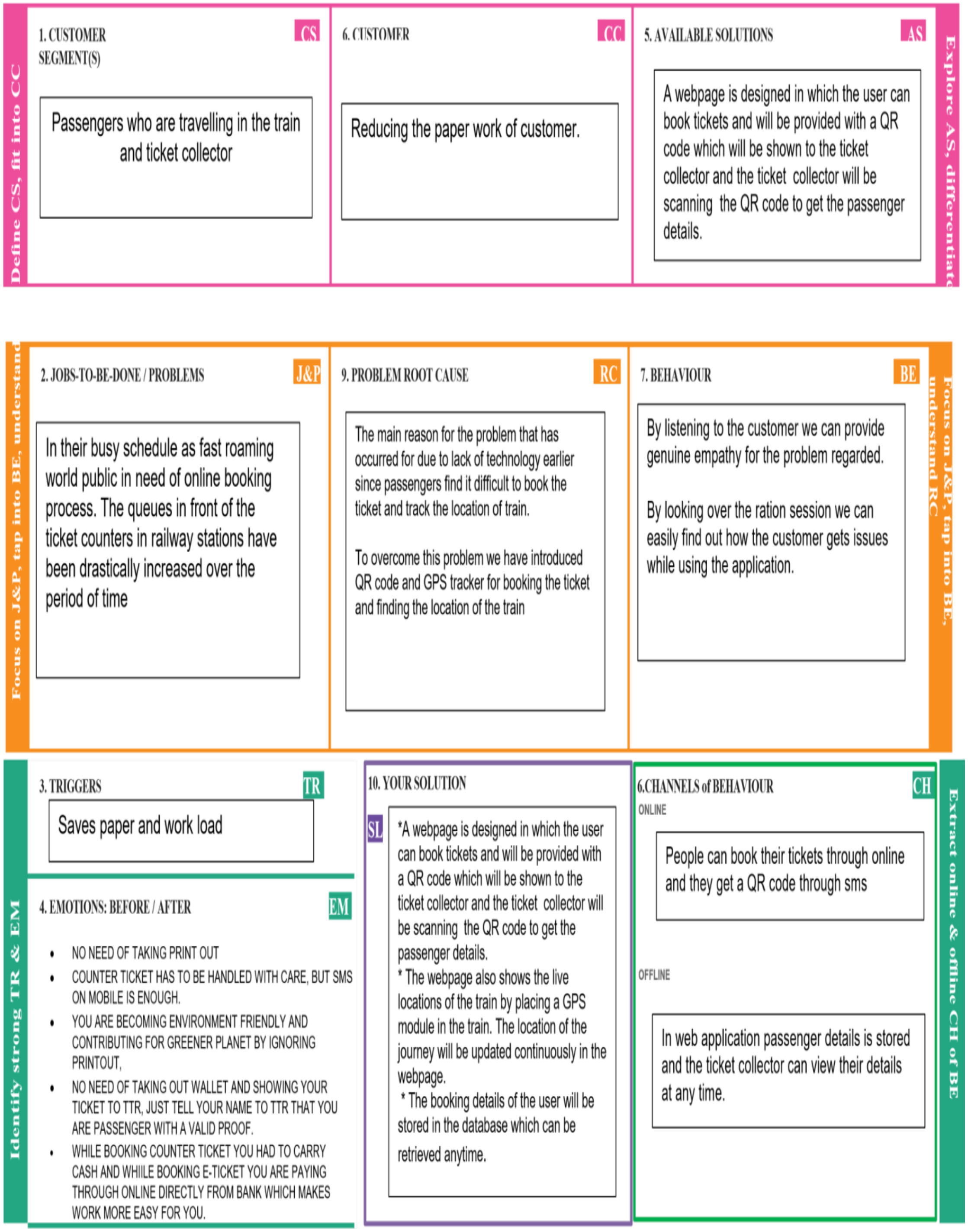
3.3 Proposed Solution





S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	• To provide an efficient way by introducing paperless tickets using QR code • To design a GPS module to track the location of the train.
2.	Idea / Solution description	• GPS tracker is placed in the train so that the passengers can track the location of the train even it is delayed. • Passengers can book their tickets using the website which is possible at anytime, anywhere. • Smart ticketing to avail seasons so that physical work is eradicated.
3.	Novelty / Uniqueness	This project stands unique from the existing ones, by implementing facilities for getting train seasons online and the passenger is alerted through mobile phone before destination arrives.
4.	Social Impact / Customer Satisfaction	• No Queuing to get tickets and burdenless because of e-tickets. • Elimination of dilemma whether the train has left or yet to arrive. • Can get the status and avail of e-seasons instead of visiting the station physically every time. • Customers for sure they get satisfied as they are in the fast roaming world this technique makes more easier for travelling passengers. • People can book their ticket through online and they get a QR code through SMS
5.	Business Model (Revenue Model)	A web page is designed in which the user can book tickets and will be provided with the QR code, which will be shown to the ticket collector and by scanning the QR code the ticket collector will get the passenger details. The booking details of the user will be stored in the database, which can be retrieved any time.
6.	Scalability of the Solution	The scalability of this solution is most feasible among the passengers who are willing to travel. No need of taking printout Counter ticket has to be handled with care, but SMS on mobile is enough. No need to taking out wallet and showing your ticket to TTR just tell your name to TTR that you are a passenger with valid proof.

3.4 Problem solution fit



Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

Passengers who are travelling in the train and ticket collector

6. CUSTOMER CC

Reducing the paper work of customer.

5. AVAILABLE SOLUTIONS AS

A webpage is designed in which the user can book tickets and will be provided with a QR code which will be shown to the ticket collector and the ticket collector will be scanning the QR code to get the passenger details.

Explore AS, differentiate

Focus on J&P, tap into BE, understand RC

2. JOBS-TO-BE-DONE / PROBLEMS J&P

In their busy schedule as fast roaming world public in need of online booking process. The queues in front of the ticket counters in railway stations have been drastically increased over the period of time

9. PROBLEM ROOT CAUSE RC

The main reason for the problem that has occurred for due to lack of technology earlier since passengers find it difficult to book the ticket and track the location of train.

To overcome this problem we have introduced QR code and GPS tracker for booking the ticket and finding the location of the train

7. BEHAVIOUR BE

By listening to the customer we can provide genuine empathy for the problem regarded.

By looking over the ration session we can easily find out how the customer gets issues while using the application.

Focus on J&P, tap into BE, understand RC

Identify strong TR & EM

3. TRIGGERS TR

Saves paper and work load

4. EMOTIONS: BEFORE / AFTER EM

- NO NEED OF TAKING PRINT OUT
- COUNTER TICKET HAS TO BE HANDLED WITH CARE, BUT SMS ON MOBILE IS ENOUGH.
- YOU ARE BECOMING ENVIRONMENT FRIENDLY AND CONTRIBUTING FOR GREENER PLANET BY IGNORING PRINTOUT,
- NO NEED OF TAKING OUT WALLET AND SHOWING YOUR TICKET TO TTR, JUST TELL YOUR NAME TO TTR THAT YOU ARE PASSENGER WITH A VALID PROOF.
- 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 MORE EASY FOR YOU.

10. YOUR SOLUTION SL

*A webpage is designed in which the user can book tickets and will be provided with a QR code which will be shown to the ticket collector and the ticket collector will be scanning the QR code to get the passenger details.
* The webpage also shows the live locations of the train by placing a GPS module in the train. The location of the journey will be updated continuously in the webpage.
* The booking details of the user will be stored in the database which can be retrieved anytime.

6.CHANNELS of BEHAVIOUR CH

ONLINE

People can book their tickets through online and they get a QR code through sms

OFFLINE

In web application passenger details is stored and the ticket collector can view their details at any time.

Extract online & offline CH of BE

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

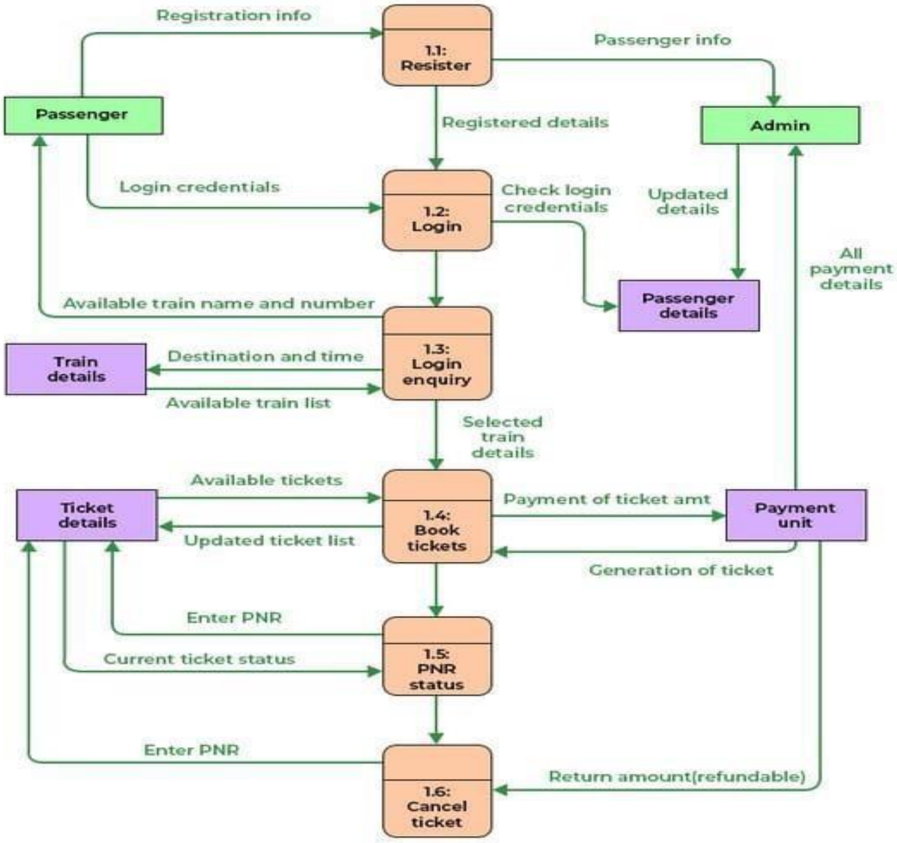
FR No.	**Functional Requirement (Epic)**	**Sub Requirement (Story / Sub-Task)**
FR-1	Passenger ticket booking	Booking through the online railway mobile app and website.
FR-2	Booking Confirmation	Booking Confirmation via Email Booking Confirmation via SMS
FR-3	Passenger objections and feedback	Through 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.	**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.

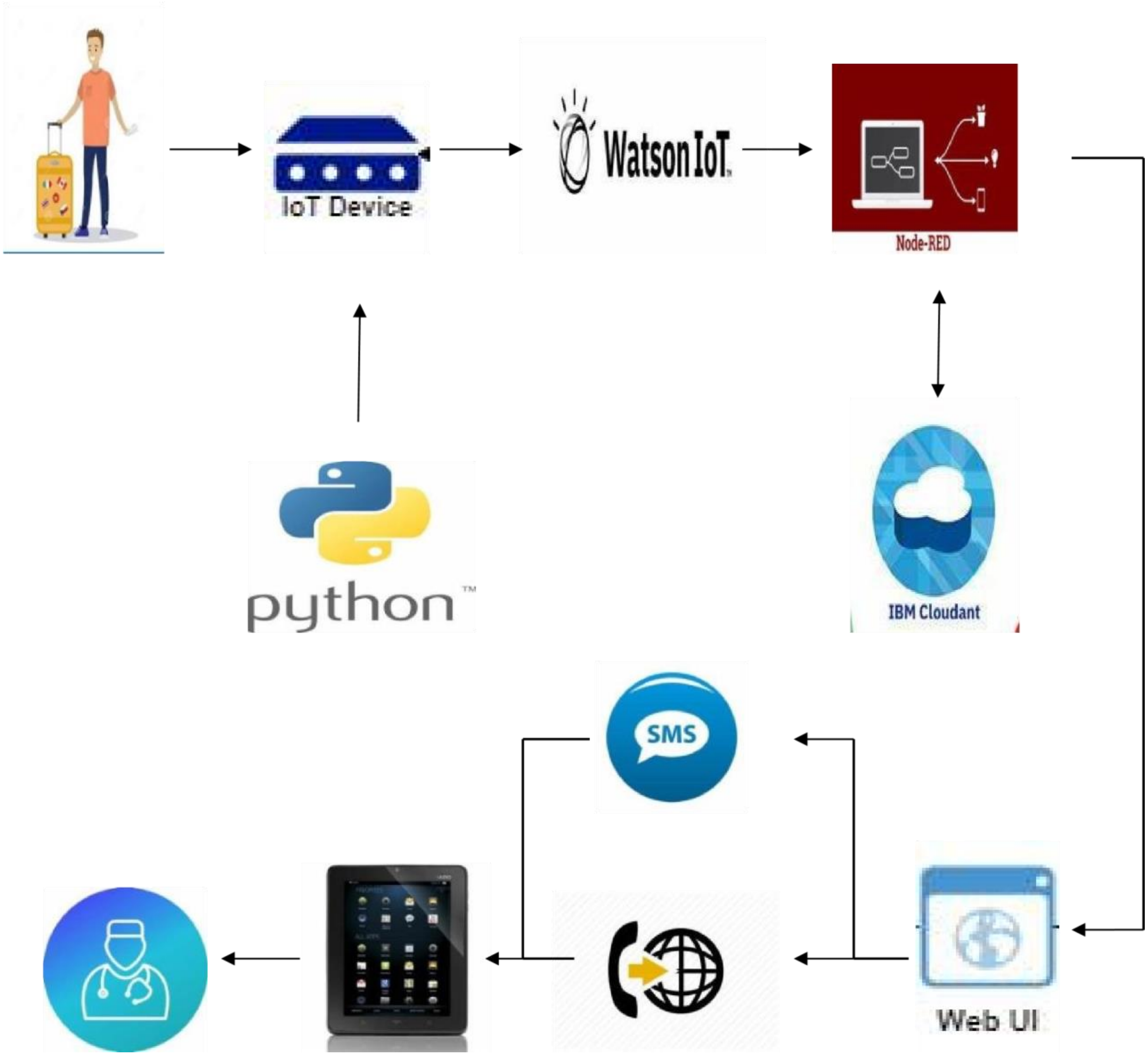
5. PROJECT DESIGN

5.1 Data Flow Diagrams



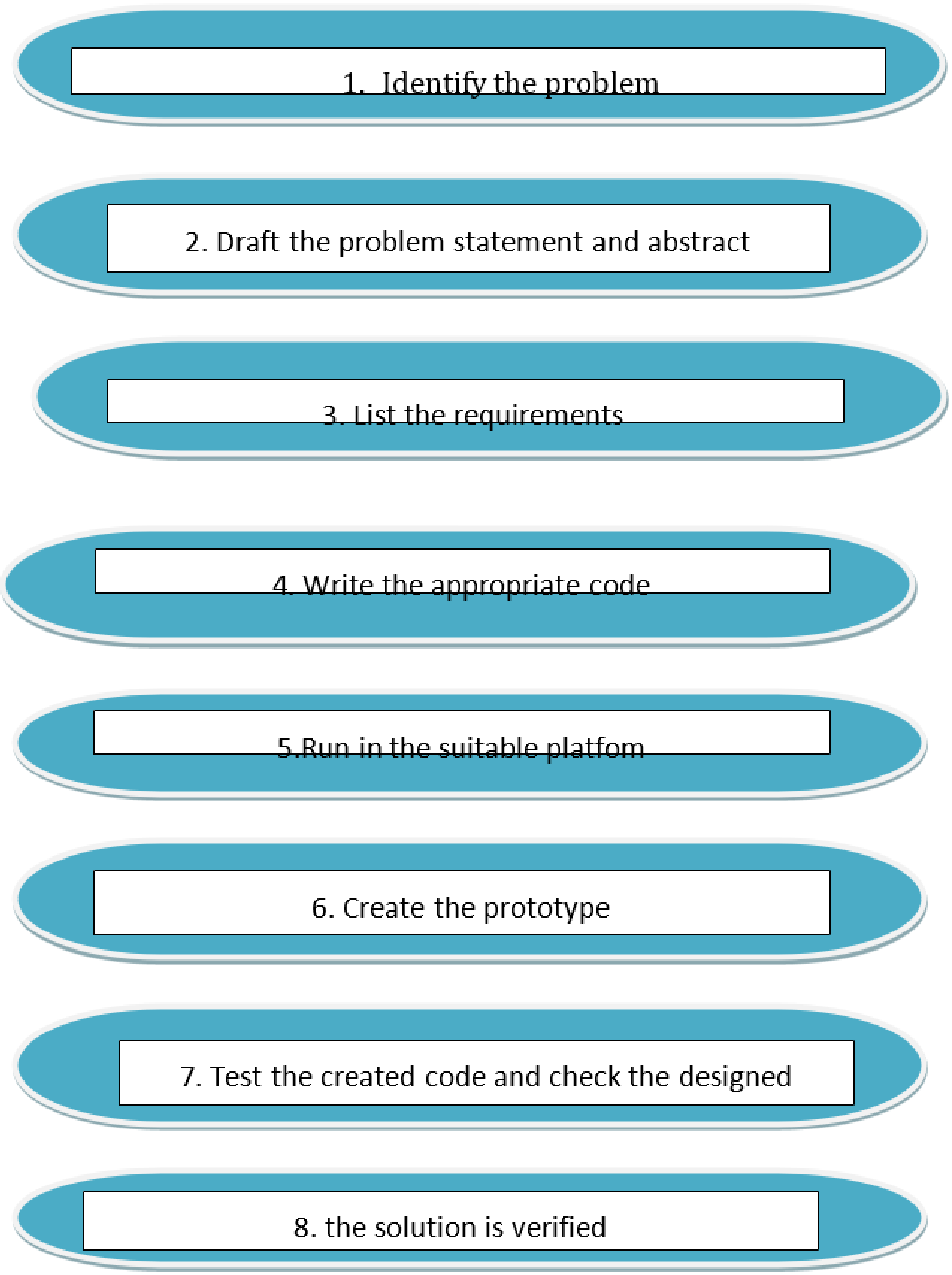
5.2 Solution Architecture

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.



6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation



1. Identify the problem

2. Draft the problem statement and abstract

3. List the requirements

4. Write the appropriate code

5.Run in the suitable platfom

6. Create the prototype

7. Test the created code and check the designed

8. the solution is verified

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. ADVANTAGES

- The passengers can use this application and book their ticket easily and no need to worry about the conformation of the ticket .
- Greater sales and marketing synergy.
- Safe and secure.

9. DISADVANTAGES

- Network issues may arise.

10. CONCLUSION

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. 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.

11. 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.

12. APPENDIX

12.1 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-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.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")
```