**Table of Contents**

### Page No.

### [Chapter 1. Introduction](#chap2)

1. Introduction to the System 4
2. Problem Definition 4
3. Aim 4
4. Objective 4
5. Goal 5
6. Need of System 5

### [Chapter 2. Hardware and Software requirement](#chap3)

1. Introduction 7
2. System environment 7
3. Software requirement 8
4. Hardware requirements 8

### [Chapter 3. System Analysis](#chap4)

1. Purpose 9
2. Project Scope 9
3. Existing System 9
4. Proposed System 9
5. System Description 10

### [Chapter 4.Implementation issues](#chap 8)

1. Python 11
2. HTML 13
3. Cascading style sheet(CSS) 16
4. Sqlite 17

[**Chapter 5. System Design**](#chap5)

1. Introduction 21
2. Conceptual Design 22
3. ER-Diagram 23
4. Logical Design 23
5. Physical Design 24
6. Data Flow Diagram of **“Railway Reservation System** 24

### [Chapter 6.User Screens](#chap 8) 25-32

### [Chapter 7.Coding](#chap 8) 33-61

[**Chapter 8. Conclusion**](#chap 9)

1. Features of “Online Railway Reservation System” 62
2. Benefits Accrued from “Railway Reservation System” 63
3. Limitations of “Railway Reservation System” 64

[Bibliography](#bibliography)

**CHAPTER # 1**

**Introduction**

*Contents:*

* Introduction
* Problem Definition
* Aim
* Objective
* Goal
* Need of System

**Introduction to the System:**

The main aim of the project was to develop a website which would facilitate the reservation of online train tickets through an effective and yet simple GUI for a normal passenger intending to travel in trains. Apart from reserving tickets, through our system a passenger can compare online fares ‘from’ one city ‘to’ other cities.

**Problem Definition:**

Managing your online railway reservation System may seem tricky, but a RRS is part of Passenger Service System (application support direct contact with passenger).

**Aim:**

# “To manage onine booking of train tickets”.

**Objective:**

* To perform a thorough analysis of working of the whole System.
* To study the problems in the System through fact finding techniques.
* To follow SDLC to develop the system.
* To develop conceptual, logical and physical model for the system.
* To develop Graphical User Interface (GUI) as per convenience of the user.
* To implement the physical model, being tested as per the standards.
* To document our efforts and analysis in a proper comprehensible manner.

**Goal:**

The project is basically targeted at those people who would like to travel through train and have an Internet access.

Finally passengers curious in comparing the prices for various train tickets for their selected source and destination cities.

To make a database that is consistent, reliable and secure.

To provide correct, complete and ongoing information.

To develop a well-organized information storage system.

To make good documentation so as to facilitate possible future enhancements.

**Need of the System:**

There is always a need of a system that will perform to online train ticket booking and train searching and also show the train route.

Thus, there is a big need of an online train ticket booking system, which provides all the Above- mentioned facilities and many more.

**CHAPTER # 2**

**Hardware and Software Requirements**

*Contents:*

##### Introduction

* System environment
* Software requirement
* Hardware requirements

**Introduction:**

In this chapter we mentioned the software and hardware requirements, which are necessary for successfully running this system. The major element in building systems is selecting compatible hardware and software. The system analyst has to determine what software package is best for the **“Online Railway Reservation System”** and, where software is not an issue, the kind of hardware and peripherals needed for the final conversion.

**System Environment:**

After analysis, some resources are required to convert the abstract system into the real one.

The hardware and software selection begins with requirement analysis, followed by a request for proposal and vendor evaluation.

Software and real system are identified. According to the provided functional specification all the technologies and its capacities are identified. Basic functions and procedures and methodologies are prepared to implement. Some of the Basic requirements such as hardware and software are described as follows: -

**Hardware and Software Specification**

**Software Requirements:**

* Technology: Python Django
* IDE : Pycharm/Atom
* Client Side Technologies: HTML, CSS, JavaScript , Bootstrap
* Server Side Technologies: Python
* Data Base Server: Sqlite
* Operating System: Microsoft Windows/Linux

**Hardware Requirements:**

* Processor: Pentium-III (or) Higher
* Ram: 64MB (or) Higher
* Hard disk: 80GB (or) Higher

**CHAPTER # 3**

**System Analysis**

*Contents:*

##### Purpose

* Project Scope
* Existing System
* Proposed System
* System Overview

### Purpose:

To manage the online Train ticket booking. It helps to passenger to book train ticket from anywhere. Also make payment online for it. It helps to people to reserve seats at their prefer time

**Project Scope:**

The project has a wide scope, as it is not intended to a particular organization. This project is going to develop generic software, which can be applied by any businesses organization. More over it provides facility to its passenger. Also the software is going to provide a huge amount of summary data.

**Proposed System:**

The railway reservation system is available in the market that can serve customers to book online train tickets.

**System Overview:**

The key features required in the system are as follows:

1. Booking
2. Cancellation
3. Passenger Details
4. Trains
5. Search
6. Report
7. **Booking:** The system can book online tickets of the passengers,

where they want to go.

1. **Cancellation:** The passenger**s** can cancel their train tickets as

usually**.**

1. **Passenger Details:**

The System stores all the necessary information of the

passengers.

1. **Train:**

It shows the train schedule such as train time, arrival and departure time, fare, route etc.

**5)** **Search:**

This is provided the search options of the system that can search any related information of the system.

**6) Report:**

This shows the reports in different fields of the system.

**CHAPTER # 4**

**Implementation issues**

**Python**

Python is a widely used general-purpose, high level programming language. It was initially designed by Guido van Rossum in 1991 and developed by Python Software Foundation. It was mainly developed for emphasis on code readability, and its syntax allows programmers to express concepts in fewer lines of code.

Python is a programming language that lets you work quickly and integrate systems more efficiently.

Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

**HTML**

HTML (Hypertext Markup Language) is the set of markup symbols or codes inserted in a file intended for display on a World Wide Web browser page. The markup tells the Web browser how to display a Web page's words and images for the user. Each individual markup code is referred to as an element (but many people also refer to it as a tag). Some elements come in pairs that indicate when some display effect is to begin and when it is to end.

**CASCADING STYLE SHEET (CSS)**

Cascading Style Sheets (CSS) are a collection of rules we use to define and modify web pages. CSS are similar to styles in Word. CSS allow Web designers to have much more control over their pages look and layout. For instance, you could create a style that defines the body text to be Verdana, 10 point. Later on, you may easily change the body text to Times New Roman, 12 point by just changing the rule in the CSS. Instead of having to change the font on each page of your website, all you need to do is redefine the style on the style sheet, and it will instantly change on all of the pages that the style sheet has been applied to. With HTML styles, the font change would be applied to each instance of that font and have to be changed in each spot.

CSS can control the placement of text and objects on your pages as well as the look of those objects.

HTML information creates the objects (or gives objects meaning), but styles describe how the objects should appear. The HTML gives your page structure, while the CSS creates the “presentation”. An external CSS is really just a text file with a .css extension. These files can be created with Dreamweaver, a CSS editor, or even Notepad.

The best practice is to design your web page on paper first so you know where you will want to use styles on your page. Then you can create the styles and apply them to your page.

**Javascript**

JavaScript is a programming language commonly used in web development. It was originally developed by Netscape as a means to add dynamic and interactive elements to websites. While JavaScript is influenced byJava, the syntax is more similar to C and is based on ECMAScript, a scripting language developed by Sun Microsystems.

JavaScript is a client-side scripting language, which means the source code is processed by the client's web browser rather than on the web server. This means JavaScript functions can run after a webpage has loaded without COMMUNICATING with the server. For example, a JavaScript function may check a web form before it is submitted to make sure all the required fields have been filled out. The JavaScript code can produce an error message before any information is actually transmitted to the server.

Like server-side scripting languages, such as PHP and ASP, JavaScript code can be inserted anywhere within the HTML of a webpage. However, only the output of server-side code is displayed in the HTML, while JavaScript code remains fully visible in the source of the webpage. It can also be referenced in a separate .JS file, which may also be viewed in a browser.

**Django**

Django is a web application framework written in Python programming language. It is based on MVT (Model View Template) design pattern. The Django is very demanding due to its rapid development feature. It takes less time to build application after collecting client requirement.

This framework uses a famous tag line: **The web framework for perfectionists with deadlines.**

**CHAPTER** # **5**

**System Design**

*Contents:*

* Use case diagram
* Class Diagram
* Sequence Diagram
* Data flow diagram

**Use Case Diagram:**

* Use case diagram consists of use cases and actors and shows the interaction between them. The key points are:
* The main purpose is to show the interaction between the use cases and the actor.
* To represent the system requirement from user’s perspective.
* The use cases are the functions that are to be performed in the module.

**SYSTEM**

**ADMIN**

* **Use Case Diagram between ADMIN and SYSTEM:**

Fig.5.1

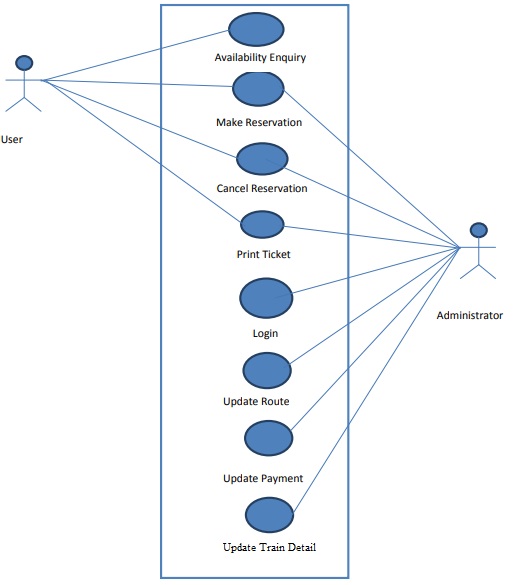
**USER**

**SYSTEM**

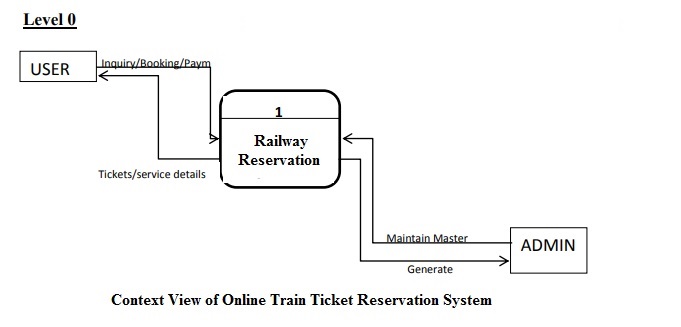
Fig.5.2

**Use Case Diagram between USER and SYSTEM:**

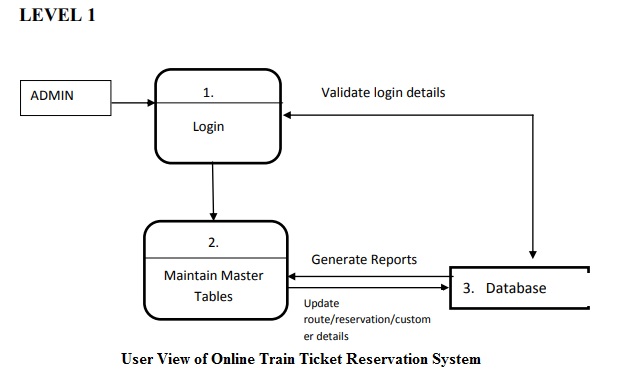
**Use Case Diagram**

****

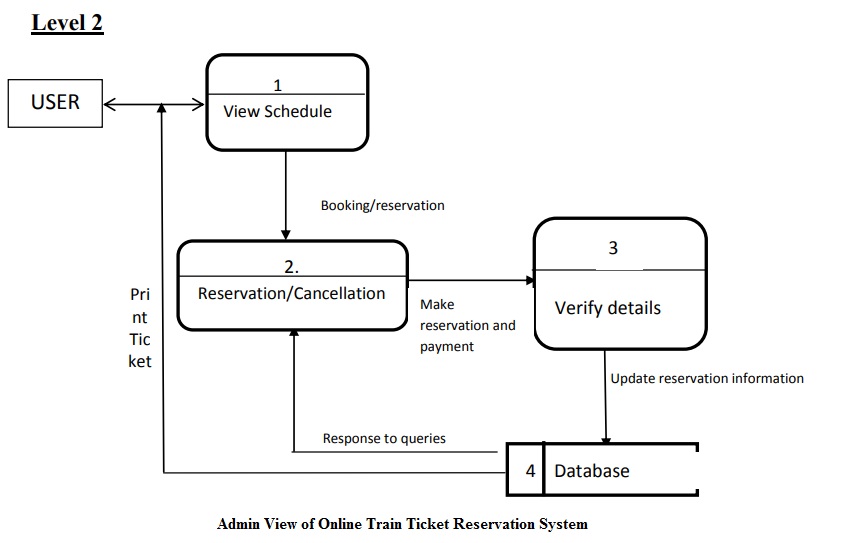
**DFD (Data Flow Diagram)**

****

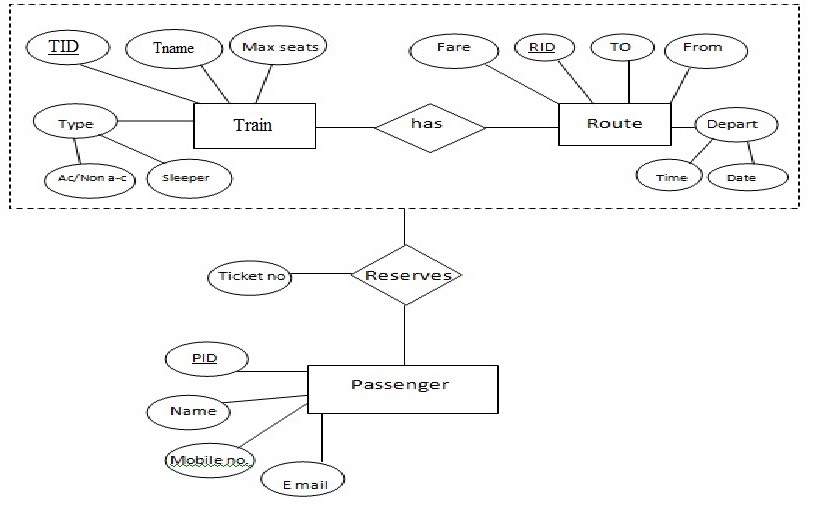
**DFD Level 1**

****

**DFD Level 2**

****

**ER DIAGRAM**

****

**Sequence Diagram For Administrator:-**

**Administrator**

**Success:hide()**

**Login**

**Application**

**Database**

**Login**

**:Request**

**:Validate()**

**:executeQuery()**

**Response**

**Show Result**

**Failed:show()**

Fig.5.4

**Sequence Diagram For User:-**

**User**

**Success:hide()**

**Login**

**Application**

**Database**

**Login**

**:Request**

**:Validate()**

**:executeQuery()**

**Response**

**Show Result**

**Failed:show()**

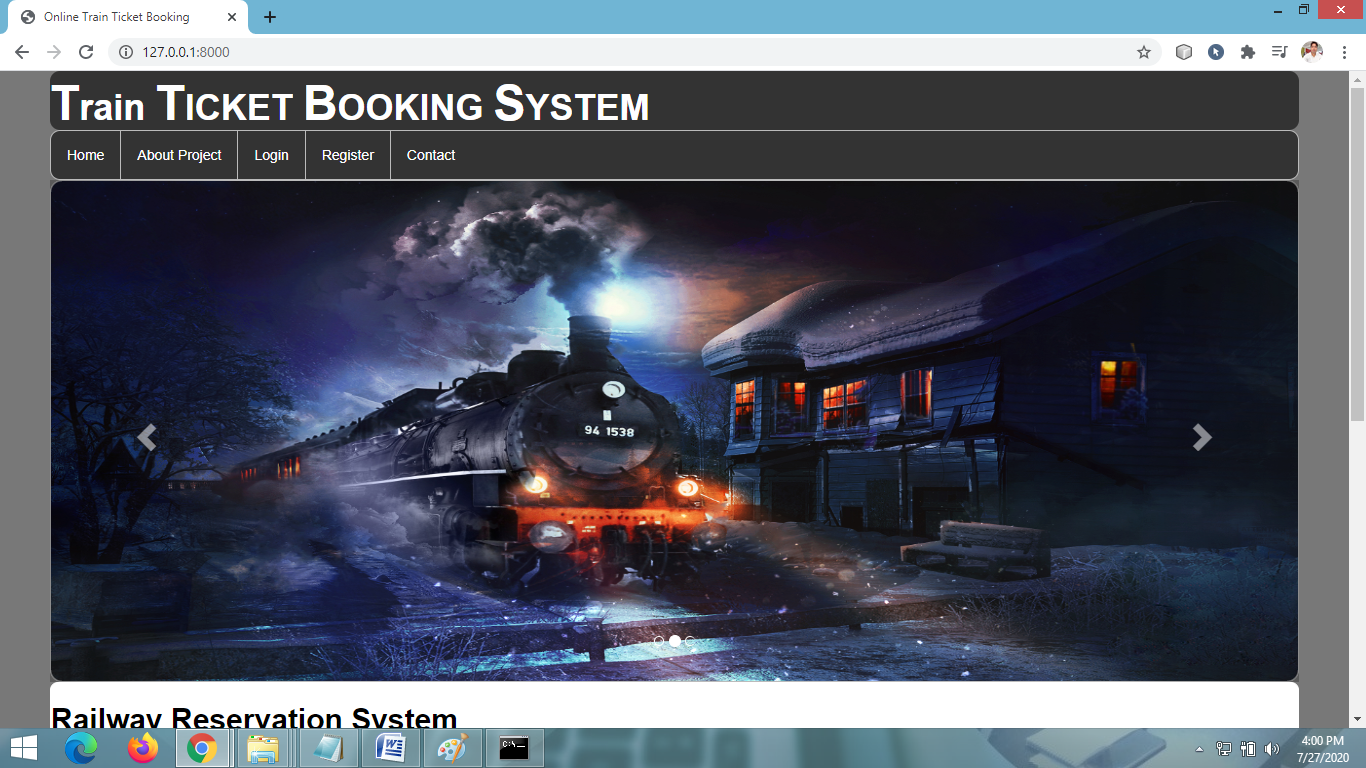
**:**

Fig.5.5

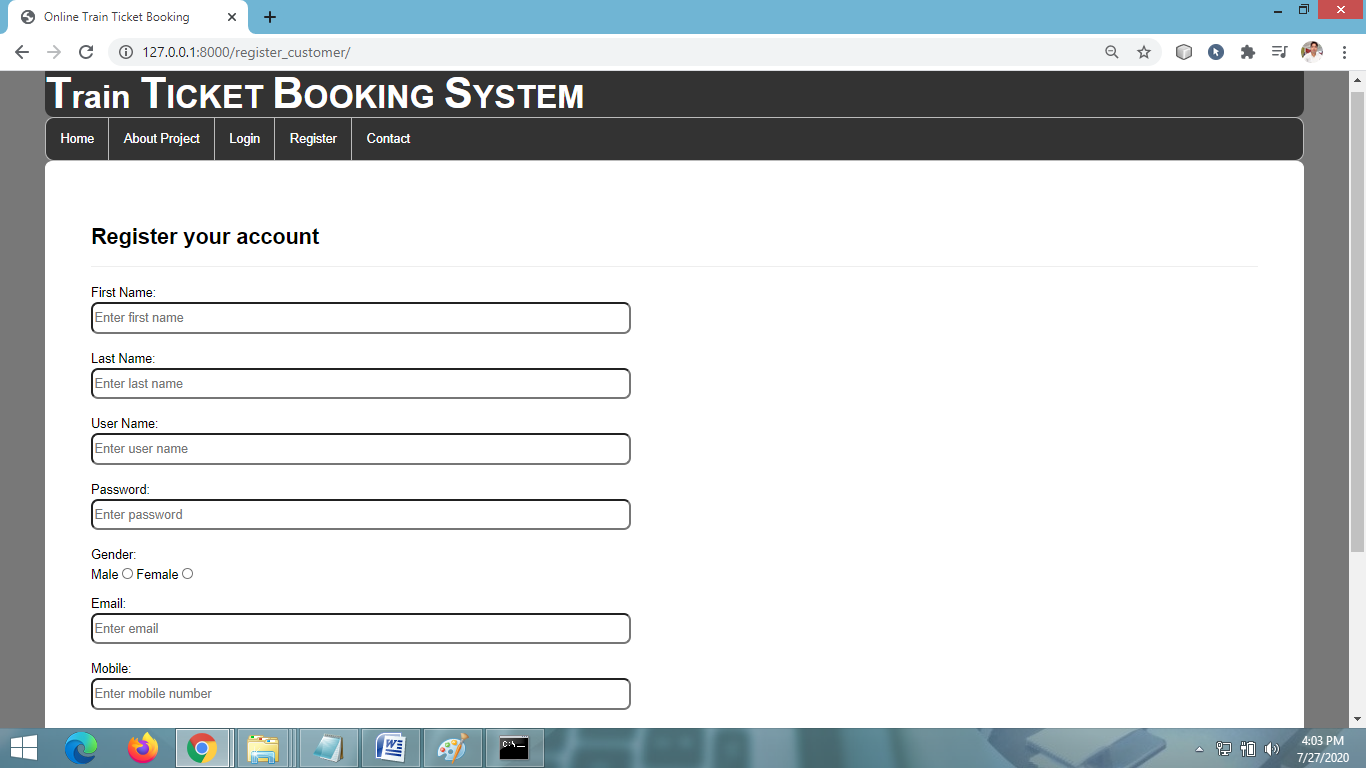
**CHAPTER # 6**

**Output screens**

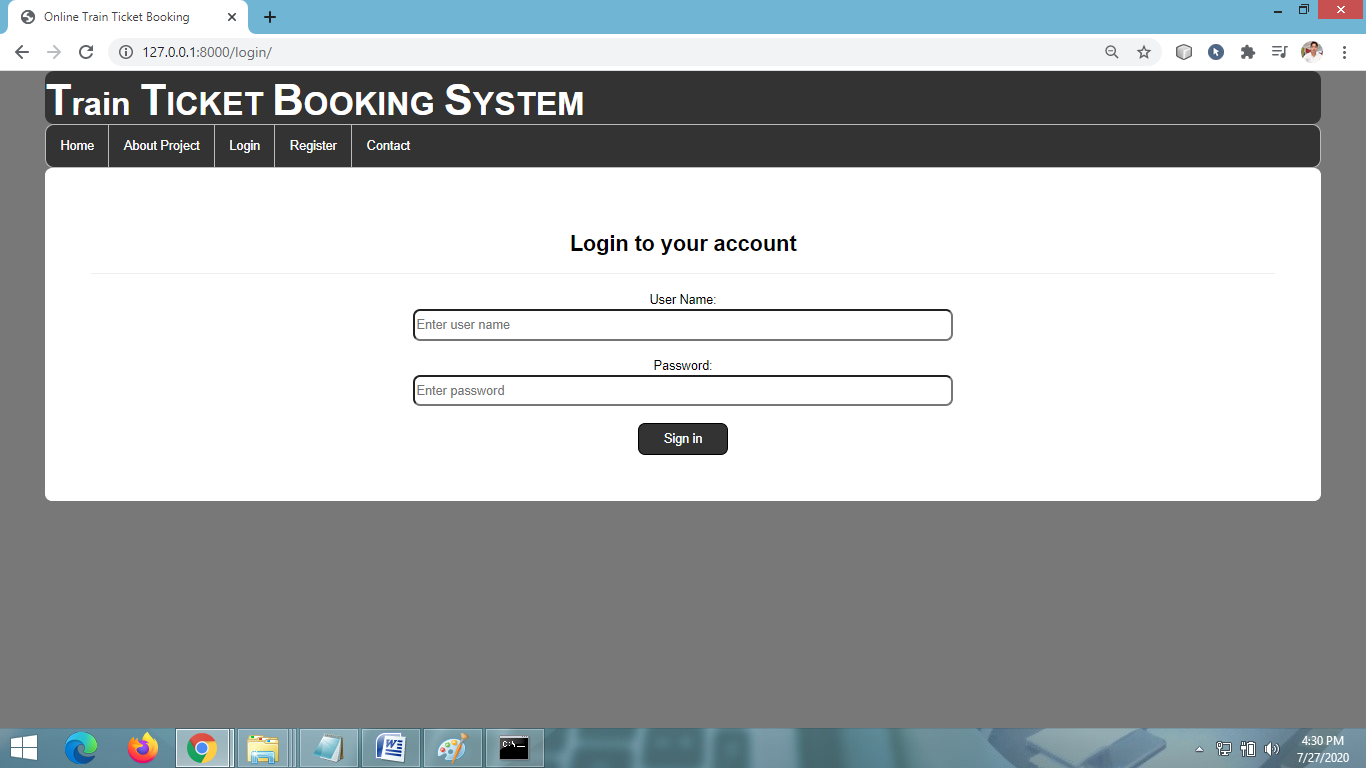
**HOME PAGE**



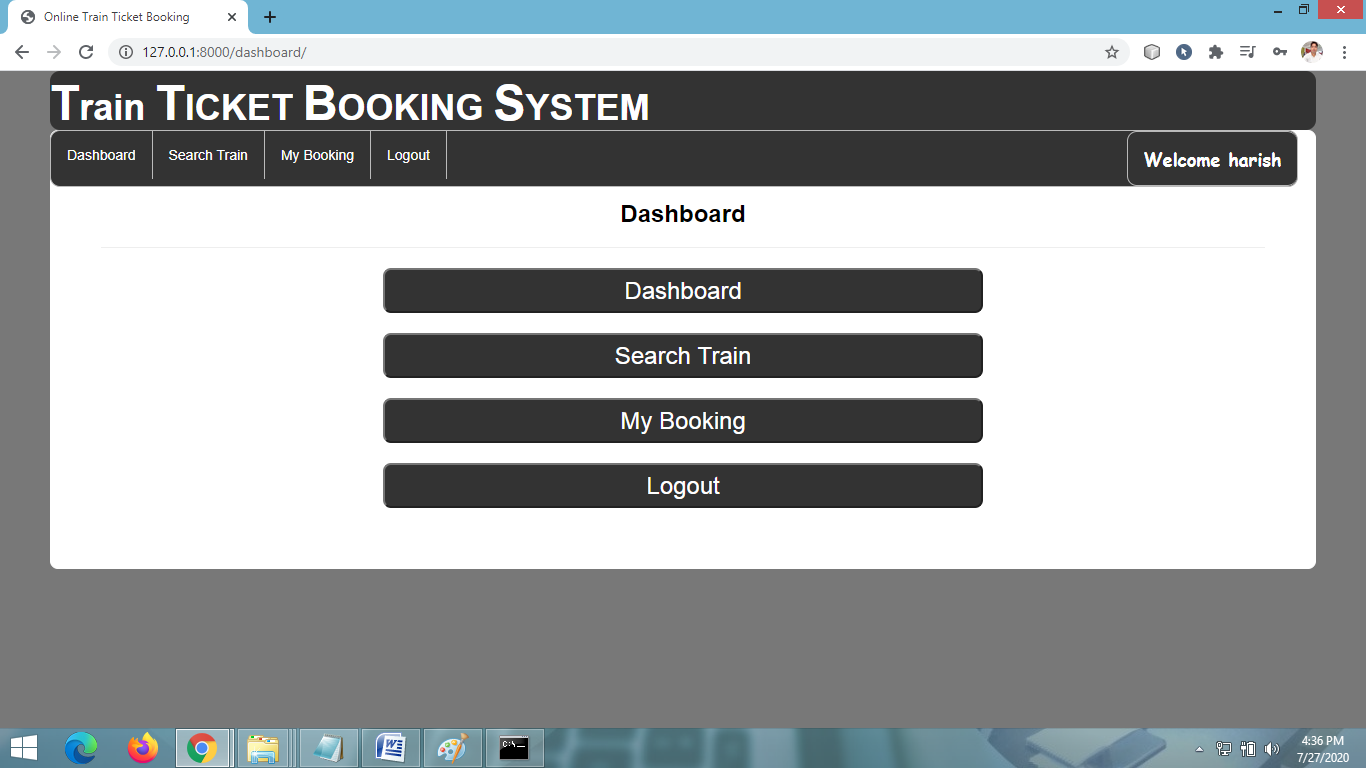
**REGISTRATION PAGE**

****

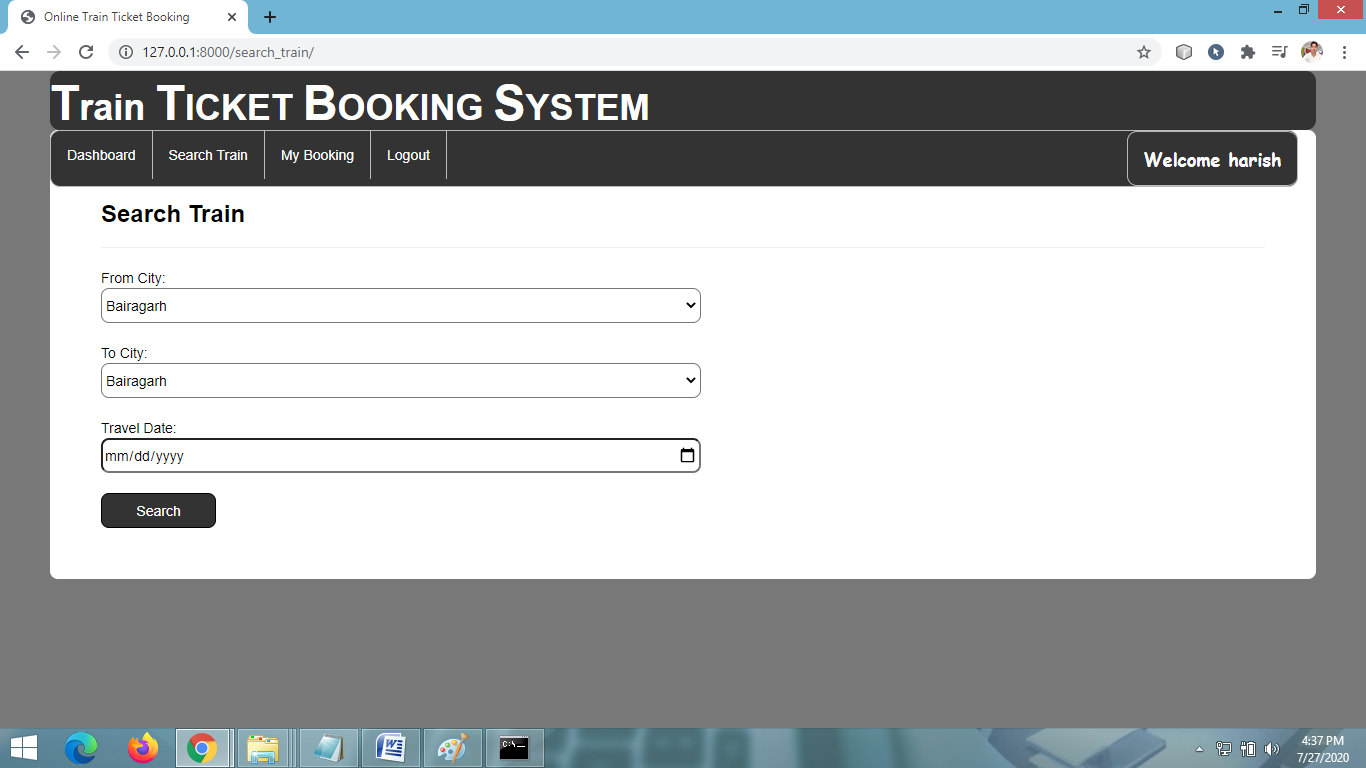
**LOGIN PAGE**

****

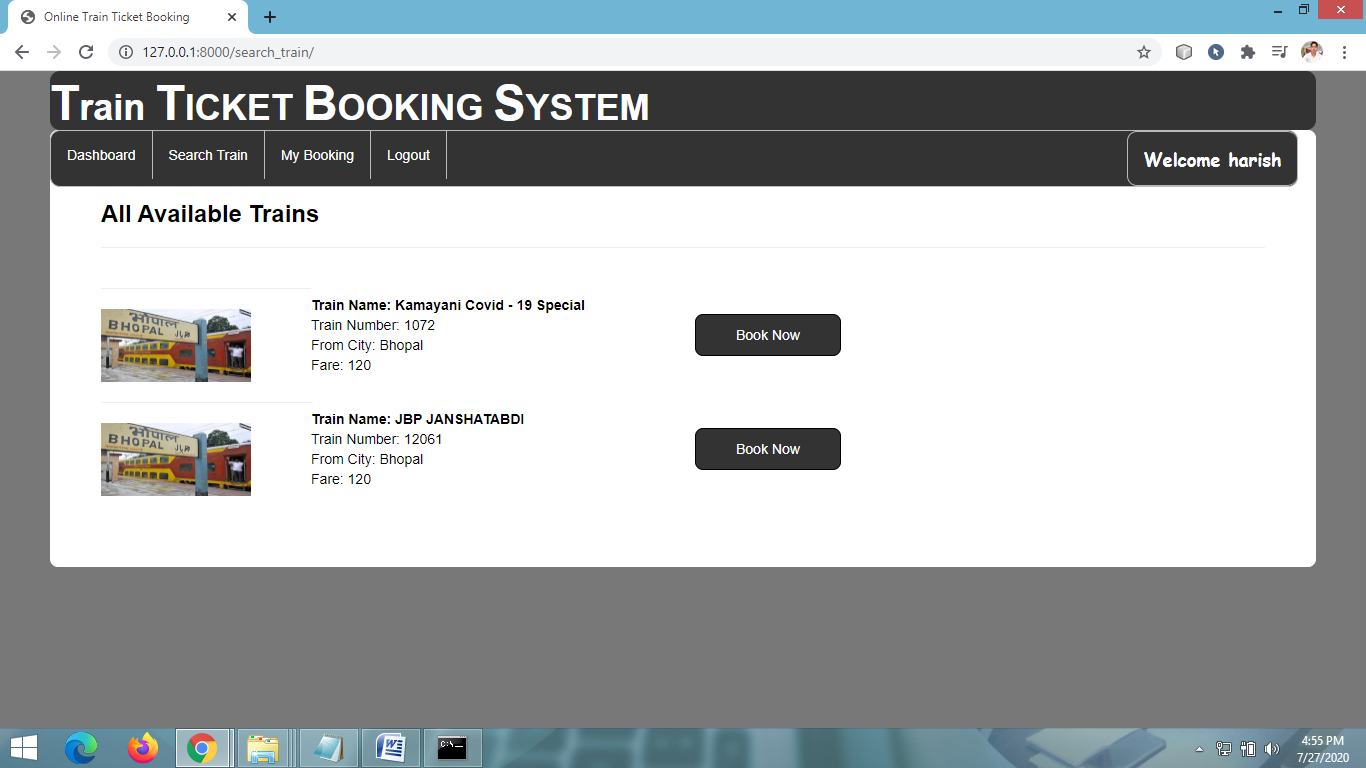
**WELCOME USER PAGE**

******

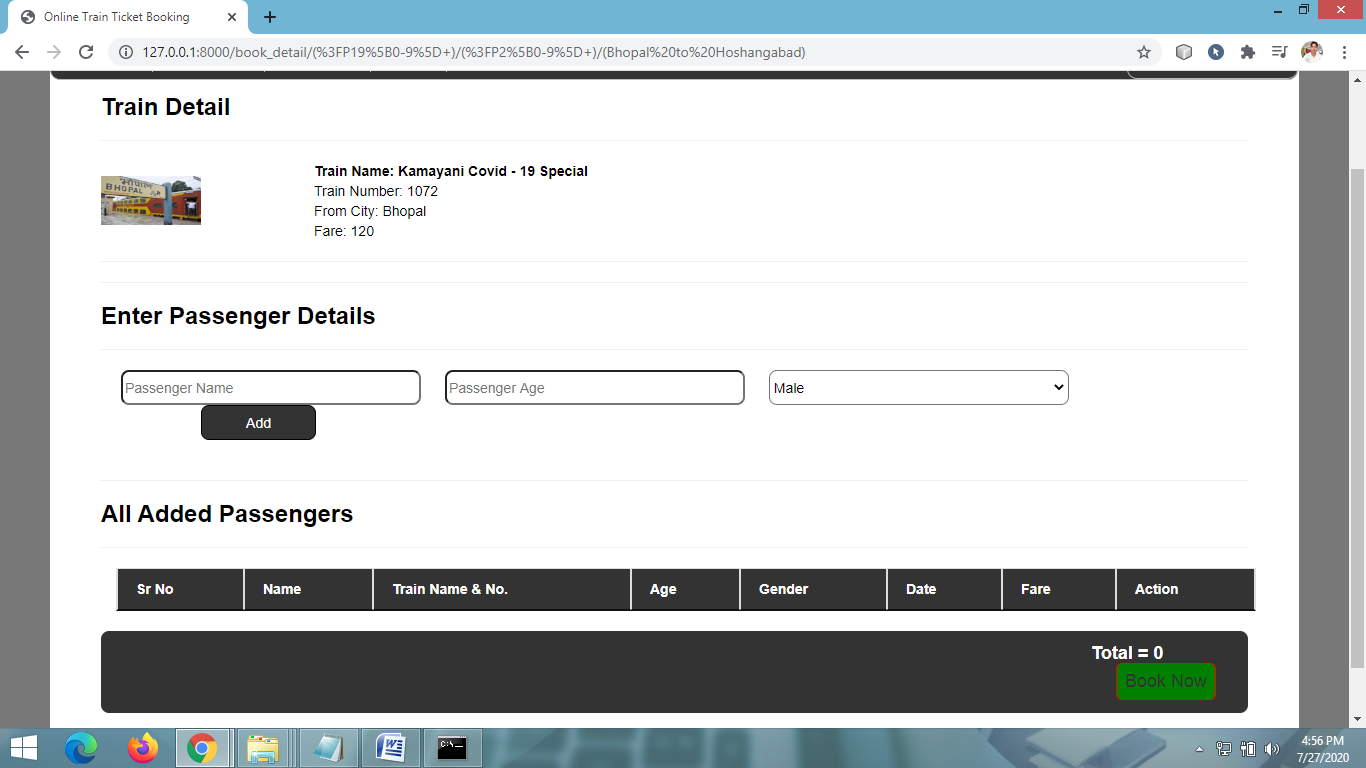
**SEARCH TRAIN PAGE**



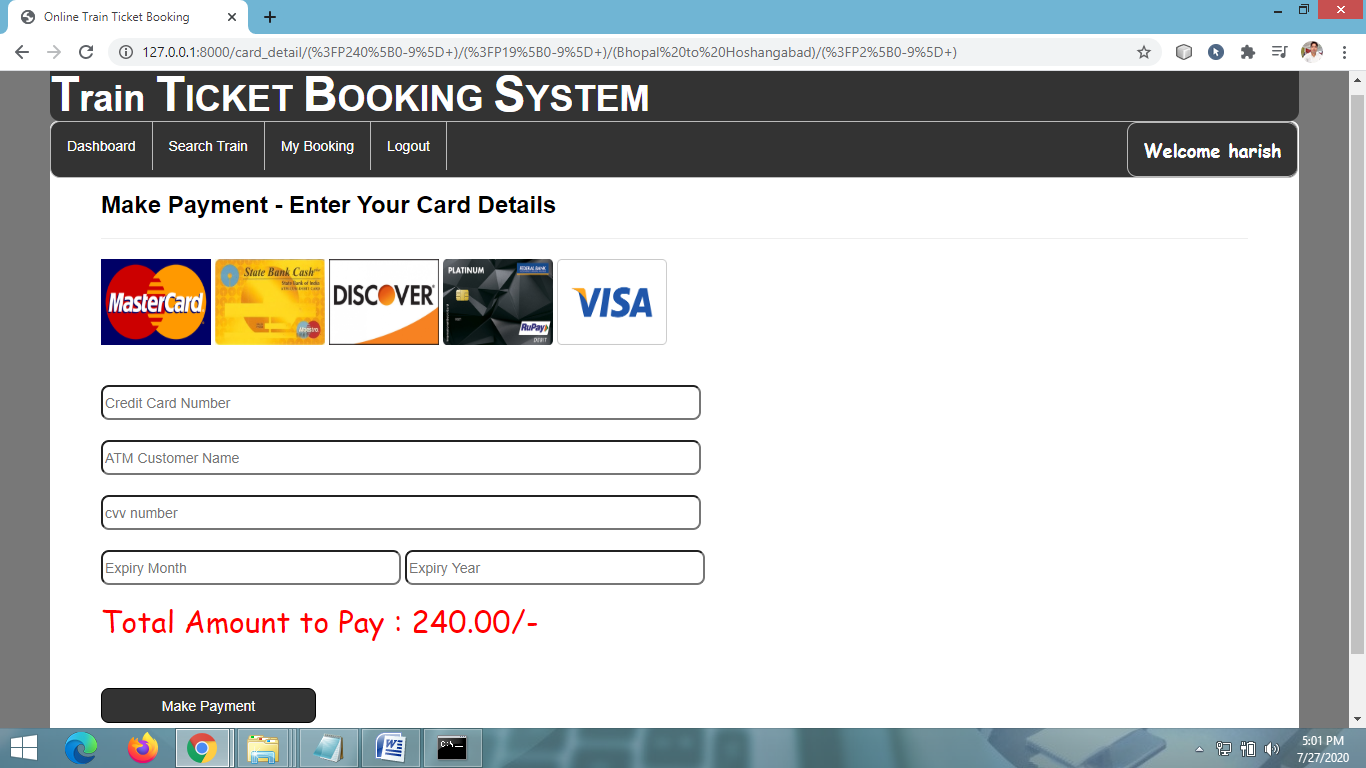
**VIEW AVAILABLE TRAINS**

****

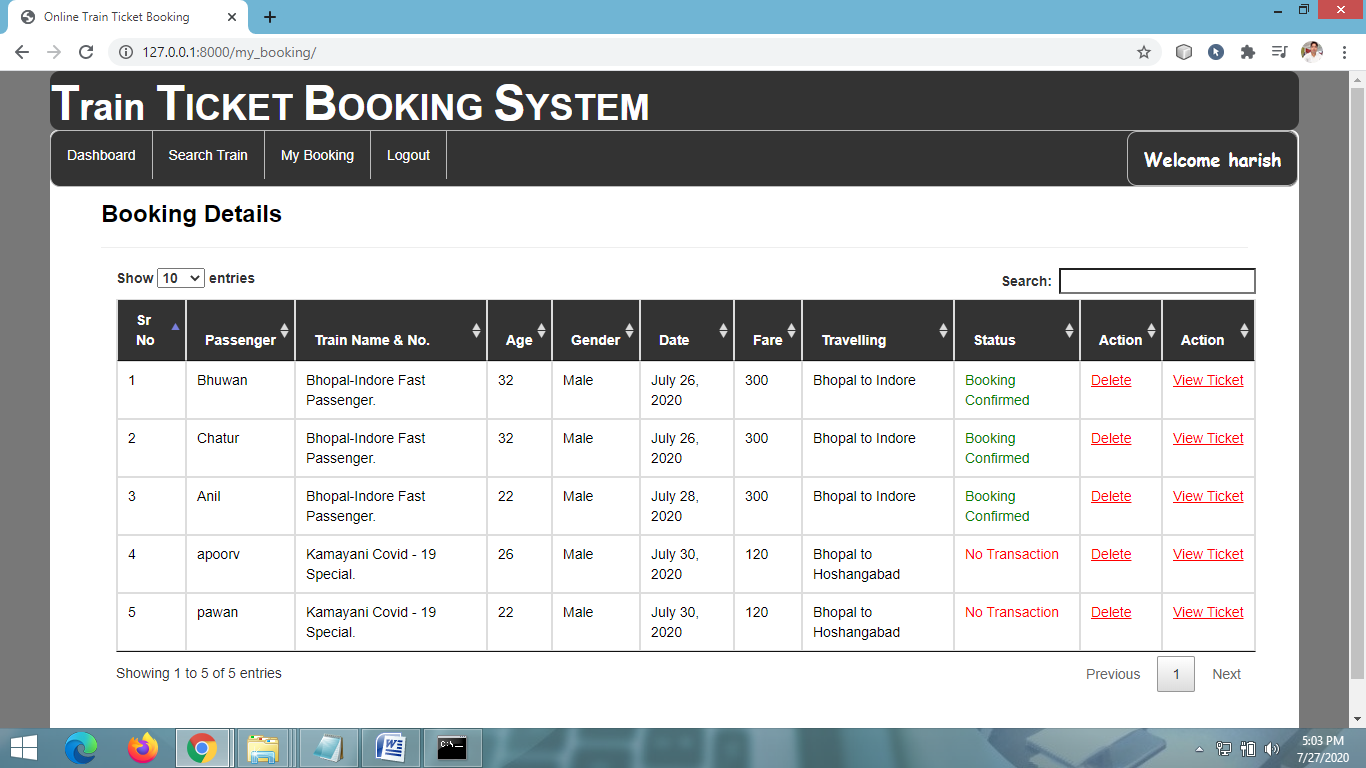
**ENTER PASSENGER DETAILS**



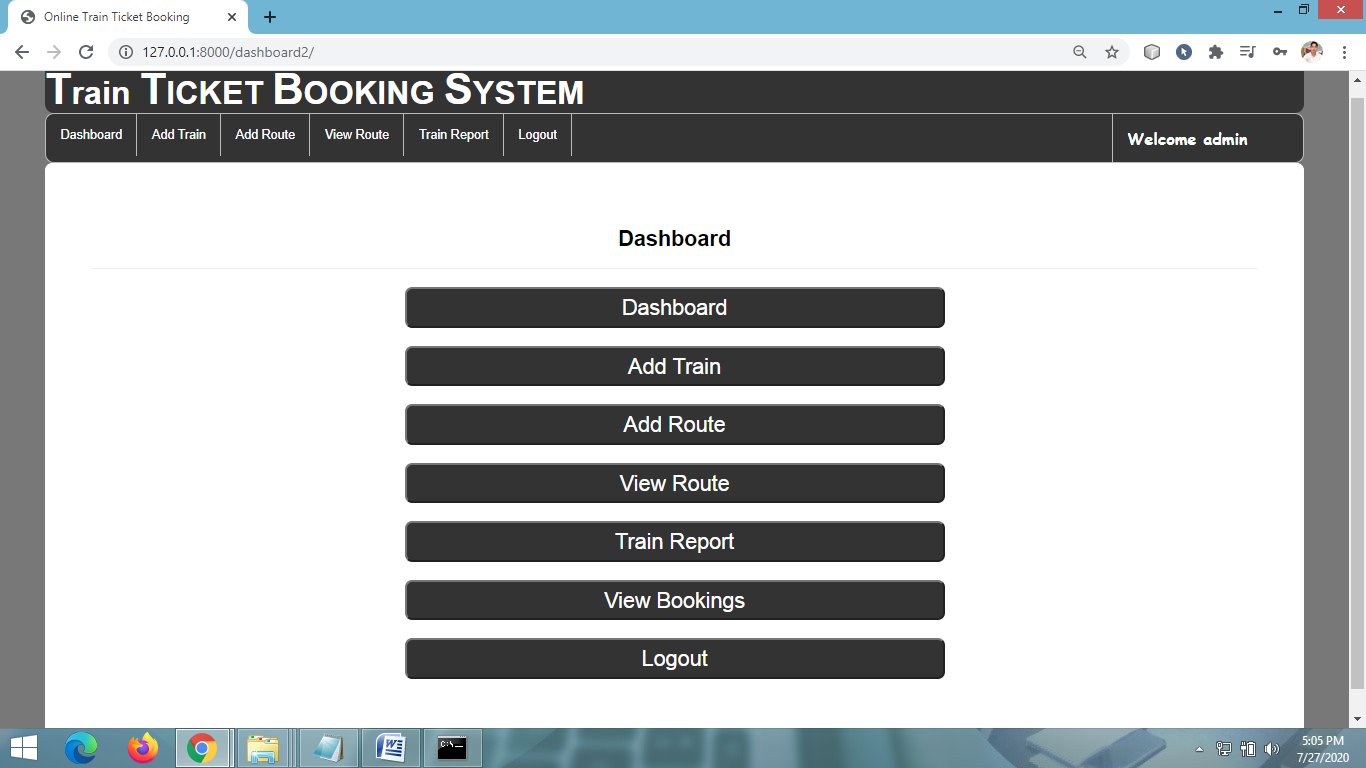
**PAYMENT PAGE**

****

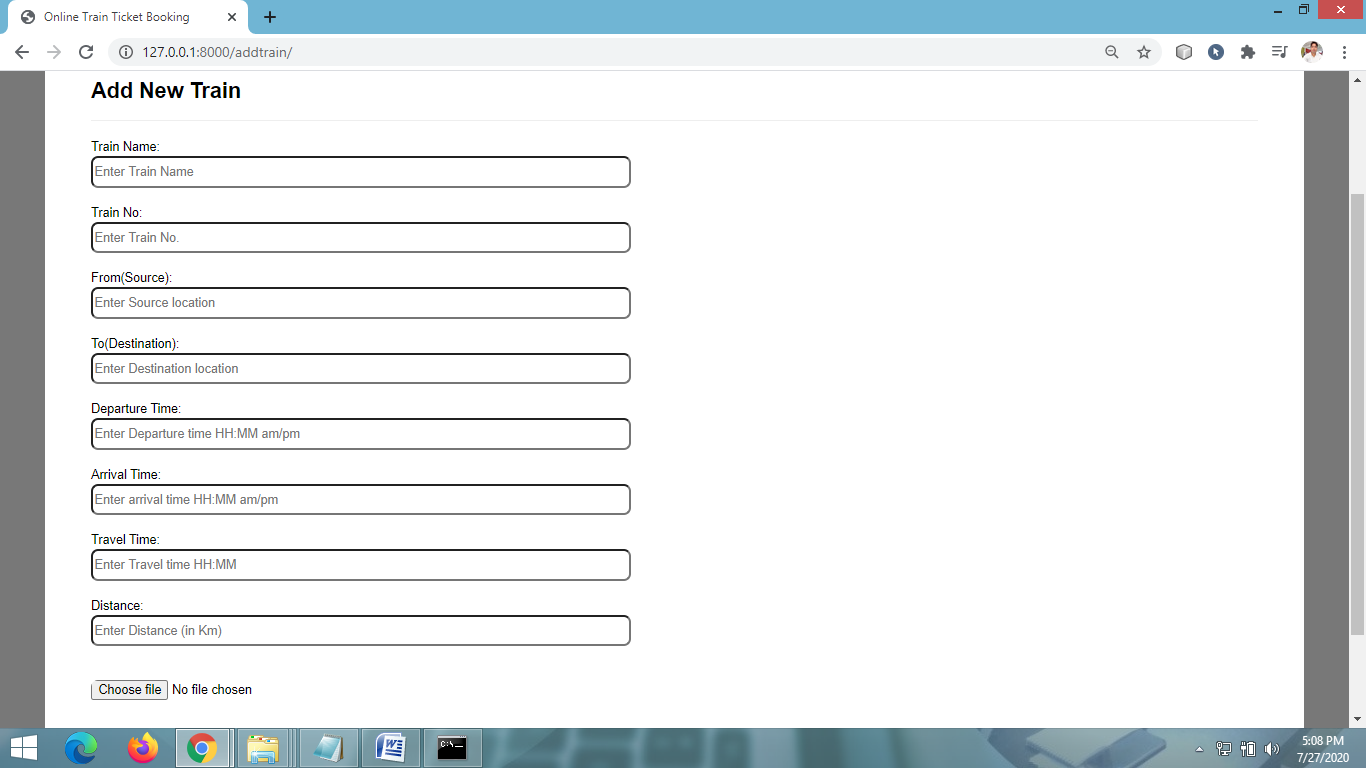
**BOOKING PAGE**



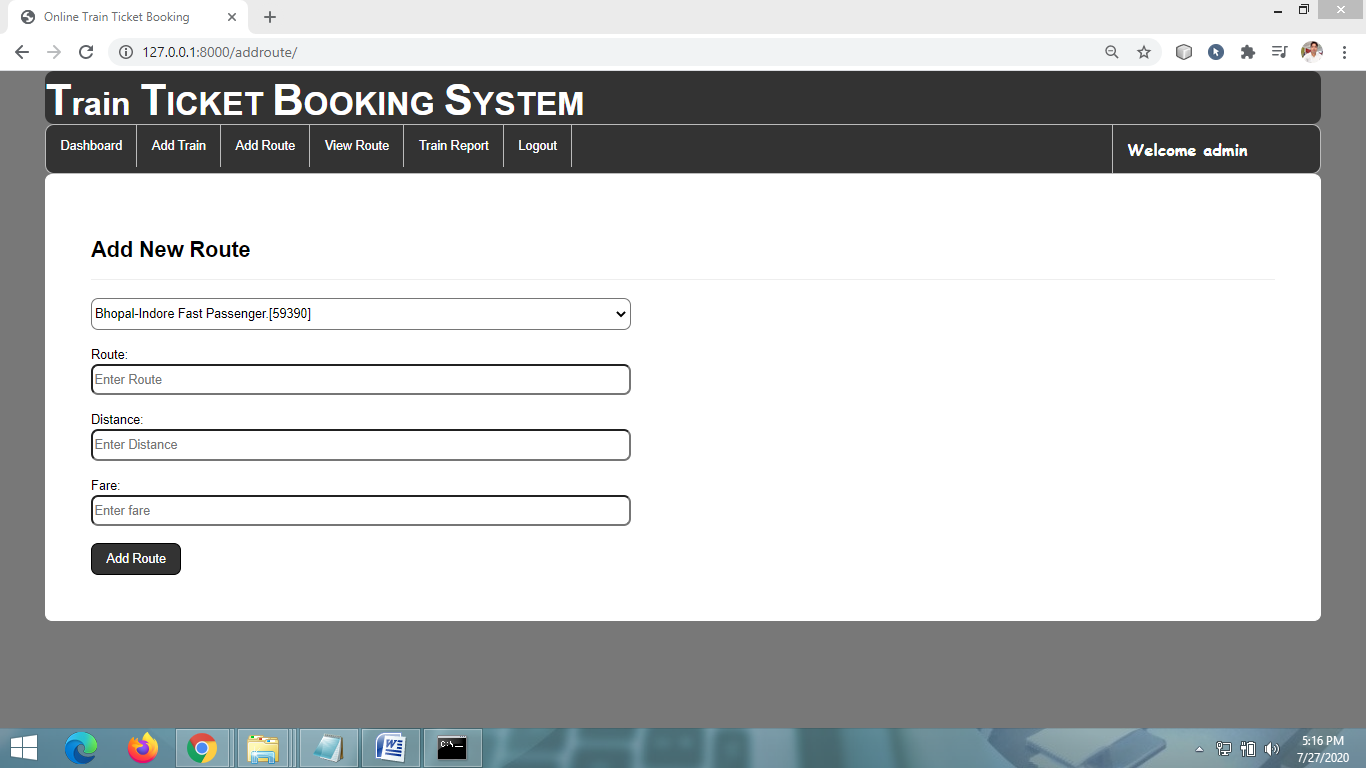
**ADMIN HOME PAGE**



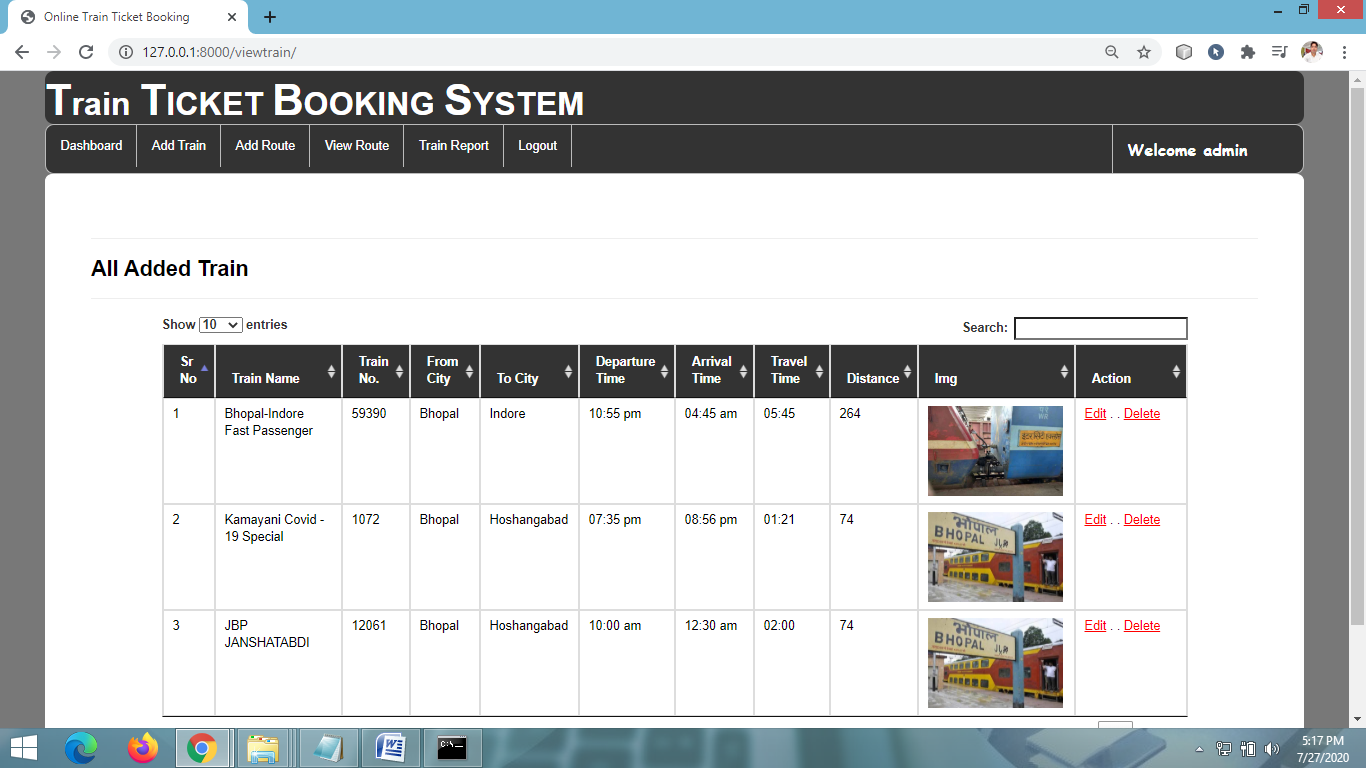
**ADD NEW TRAIN DETAILS PAGE**



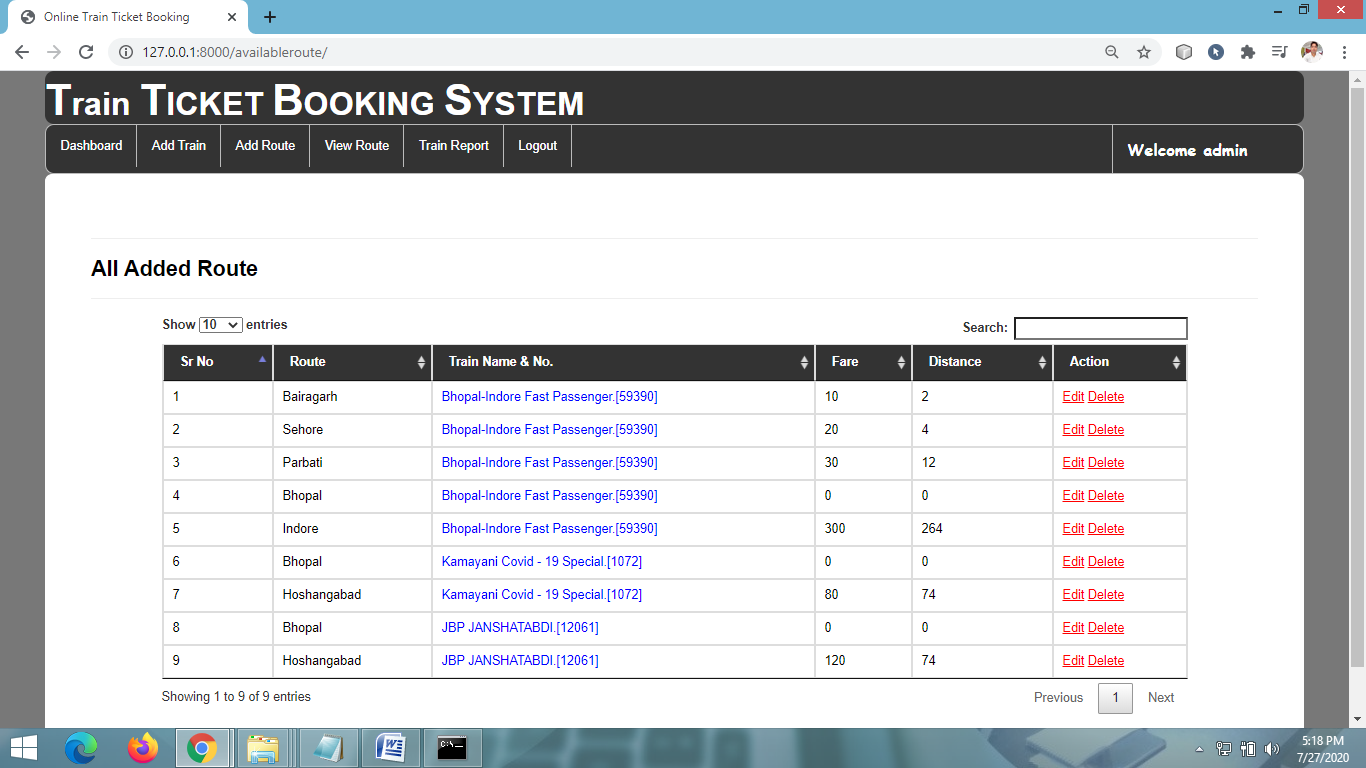
**ADD NEW TRAIN ROUTE**



**VIEW ALL TRAIN DETAILS**



**VIEW ALL ROUTES**



**CHAPTER # 7**

***Coding***

**CUSTOMER LOGIN PAGE CODING**

**{% extends 'navigation.html' %}**

**{% load static %}**

**{% block a %}**

**<style>**

**input[type=text][type=password]{**

**border-radius:8px;**

**width:600px;**

**height:16px;**

**}**

**</style>**

**{% if error %}**

**<script>**

**alert('Logged in Sucessfully');**

**window.location='{% url 'dashboard' %}';**

**</script>**

**{% endif %}**

**{% if error2 %}**

**<script>**

**alert('Logged in Sucessfully');**

**window.location='{% url 'admindashboard' %}';**

**</script>**

**{% endif %}**

**{% if error3 %}**

**<script>**

**alert('Usename & Password are not Correct');**

**window.location='{% url 'login\_customer' %}';**

**</script>**

**{% endif %}**

**<div style="color:Black;border:1px solid white;border-radius:8px;background-color:white;padding:50px">**

**<h3><strong>Login to your account</strong></h3><hr>**

**<form method="post" action="">**

**{% csrf\_token %}**

**User Name:<br><input type="text" placeholder="Enter user name" name="uname" style="border-radius:8px;width:600px;height:35px;"><br><br>**

**Password:<br><input type="password" placeholder="Enter password" name="pwd" style="border-radius:8px;width:600px;height:35px;"><br><br>**

**<input type="submit" value="Sign in" style="color:white;border-radius:8px;width:100px;height:35px;background-color:#333;border:1px solid black">**

**</form>**

**</div>**

**{% endblock %}**

**USER REGISTRATION PAGE CODING**

**{% extends 'navigation.html' %}**

**{% load static %}**

**{% block a %}**

**<style>**

**input[type=text][type=password]{**

**border-radius:8px;**

**width:600px;**

**height:16px;**

**}**

**</style>**

**{% if error %}**

**<script>**

**alert('Ragistration Sucessfull');**

**window.location='{% url 'login\_customer' %}';**

**</script>**

**{% endif %}**

**<div style="color:Black;border:1px solid white;border-radius:8px;background-color:white;padding:50px">**

**<h3><strong>Register your account</strong></h3><hr>**

**<form method="post" action="">**

**{% csrf\_token %}**

**First Name:<br><input type="text" placeholder="Enter first name" name="fname" style="border-radius:8px;width:600px;height:35px;"><br><br>**

**Last Name:<br><input type="text" placeholder="Enter last name" name="lname" style="border-radius:8px;width:600px;height:35px;"><br><br>**

**User Name:<br><input type="text" placeholder="Enter user name" name="uname" style="border-radius:8px;width:600px;height:35px;"><br><br>**

**Password:<br><input type="password" placeholder="Enter password" name="pwd" style="border-radius:8px;width:600px;height:35px;"><br><br>**

**Gender:<br><div style="border-radius:8px;width:600px;height:35px;"> Male <input type="radio" value="Male" name="male"> Female <input type="radio" value="Female" name="male"><br><br></div>**

**Email:<br><input type="email" placeholder="Enter email" name="email" style="border-radius:8px;width:600px;height:35px;"><br><br>**

**Mobile:<br><input type="text" placeholder="Enter mobile number" name="mobile" style="border-radius:8px;width:600px;height:35px;"><br><br>**

**Date of Birth:<br><input type="date" placeholder="Enter your Date of Birth" name="birth" style="border-radius:8px;width:600px;height:35px;"><br><br>**

**Full Address:<br><input type="text" placeholder="Enter Address" name="add" style="border-radius:8px;width:600px;height:35px;"><br><br>**

**<input type="submit" value="Sign in" style="color:white;border-radius:8px;width:100px;height:35px;background-color:#333;border:1px solid black">**

**</form>**

**</div>**

**{% endblock %}**

**SEARCH TRAIN PAGE CODING**

**{% extends 'navigation2.html' %}**

**{% load static %}**

**{% block b %}**

**<style>**

**input[type=text][type=password]{**

**border-radius:8px;**

**width:600px;**

**height:16px;**

**}**

**</style>**

**<div style="color:Black;border:1px solid white;border-radius:8px;background-color:white;padding:50px">**

**{% if error %}**

**<h3><strong>All Available TRAIN</strong></h3><hr>**

**<table width="800">**

**{% for i in route1 %}**

**<tr style="margin-top:10px">**

**<td><hr><img src="{{i.0.img.url}}" style="width:150px;height:135"></td>**

**<td><b><br>TRAIN Name: {{i.0.TRAINname}}</b><br>**

**TRAIN Number: {{i.0.TRAIN\_no}}<br>**

**From City: {{i.0.from\_city}}<br>**

**Fare: {{fare3}}<br></td>**

**<td><br><br><button style="background-color:#333;border:1px solid black;border-radius:8px;padding:10px"><a href="{% url 'book\_detail' coun i.0.id route %}" style="padding:30px;color:white;">Book Now</a></button><br><br></td>**

**</tr>**

**{% endfor %}**

**</table><br>**

**{% else %}**

**<h3><strong>Search TRAIN</strong></h3><hr>**

**<form method="post" action="">**

**{% csrf\_token %}**

**From City:<br><select name="fcity" style="border-radius:8px;width:600px;height:35px;">**

**{% for i in data2 %}**

**<option>{{i.route}}</option>**

**{% endfor %}**

**</select><br><br>**

**To City:<br><select name="tcity" style="border-radius:8px;width:600px;height:35px;">**

**{% for i in data2 %}**

**<option>{{i.route}}</option>**

**{% endfor %}**

**</select><br><br>**

**Travel Date:<br><input type="date" name="date" style="border-radius:8px;width:600px;height:35px;"><br><br>**

**<input type="submit" value="Search" style="color:white;border-radius:8px;width:115px;height:35px;background-color:#333;border:1px solid black">**

**</form>**

**{% endif %}**

**</div>**

**{% endblock %}**

**VIEW MY BOOKINGS PAGE**

**{% extends 'navigation2.html' %}**

**{% load static %}**

**{% block b %}**

**<style>**

**input[type=text][type=password]{**

**border-radius:8px;**

**width:600px;**

**height:16px;**

**}**

**</style>**

**{% if error %}**

**<script>**

**alert('Booking Deleted Successfully')**

**window.location='{% url 'my\_booking' %}'**

**</script>**

**{% endif %}**

**<div style="color:Black;border:1px solid white;border-radius:8px;background-color:white;padding:50px">**

**<h3><strong>Booking Details</strong></h3><hr>**

**<div class="container">**

**<table class="table table-bordered" id="myTable">**

**<thead style="color:white;background-color:#333">**

**<tr>**

**<th>Sr No</th>**

**<th>Passenger</th>**

**<th>TRAIN Name & TRAIN No.</th>**

**<th> Age</th>**

**<th>Gender</th>**

**<th>Date</th>**

**<th>Fare</th>**

**<th>Travelling</th>**

**<th>Status</th>**

**<th>Action</th>**

**</tr>**

**</thead>**

**<tbody>**

**{% for i in pro %}**

**<tr>**

**<td>{{forloop.counter}}</td>**

**<td>{{i.name}}</td>**

**<td>{{i.TRAIN.TRAINname}}. {{i.TRAIN.TRAIN\_no}}</td>**

**<td>{{i.age}}</td>**

**<td>{{i.gender}}</td>**

**<td>{{i.date1}}</td>**

**<td>{{i.fare}}</td>**

**<td>{{i.route}}</td>**

**{% if i.status == 'set' %}**

**<td style="color:green">Booking Confirmed</td>**

**{% else %}**

**<td style="color:red">No Transaction</td>**

**{% endif %}**

**<td><a href="{% url 'delte\_my\_booking' i.id %}" style="color:red"><u>Delete</u></a></td>**

**</tr>**

**{% endfor %}**

**</tbody>**

**</table>**

**</div>**

**</div>**

**{% endblock %}**

**ADMIN HOME PAGE CODING**

**{% extends 'navigation3.html' %}**

**{% load static %}**

**{% block b %}**

**<style>**

**input[type=text][type=password]{**

**border-radius:8px;**

**width:600px;**

**height:16px;**

**}**

**</style>**

**<div style="color:Black;border:1px solid white;border-radius:8px;background-color:white;padding:50px">**

**<h3><strong>Dashboard</strong></h3><hr>**

**<h3><a href="{% url 'admindashboard' %}"><button style="width:600px;color:white;border-radius:8px;height:45px;background-color:#333">Dashboard</button></a></h3>**

**<h3><a href="{% url 'add\_TRAIN' %}"><button style="width:600px;color:white;border-radius:8px;height:45px;background-color:#333">Add TRAIN</button></a></h3>**

**<h3><a href="{% url 'add\_route' %}"><button style="width:600px;color:white;border-radius:8px;height:45px;background-color:#333">Add Route</button></a></h3>**

**<h3><a href="{% url 'availableroute' %}"><button style="width:600px;color:white;border-radius:8px;height:45px;background-color:#333">View Route</button></a></h3>**

**<h3><a href="{% url 'view\_TRAIN' %}"><button style="width:600px;color:white;border-radius:8px;height:45px;background-color:#333">TRAIN Report</button></a></h3>**

**<h3><a href="{% url 'log\_out' %}"><button style="width:600px;color:white;border-radius:8px;height:45px;background-color:#333">Logout</button></a></h3>**

**</div>**

**{% endblock %}**

**<li id="id1"><a href="{% url 'availableroute' %}">View Route</a></li>**

**ADD NEW TRAIN DETAIL FORM CODING**

**{% extends 'navigation3.html' %}**

**{% load static %}**

**{% block b %}**

**<style>**

**input[type=text][type=time][type=file]{**

**border-radius:8px;**

**width:600px;**

**height:16px;**

**}**

**</style>**

**{% if error %}**

**<script>alert('Added New TRAIN Sucessfully');**

**window.location='{% url 'view\_TRAIN' %}';**

**</script>**

**{% endif %}**

**<div style="color:Black;border:1px solid white;border-radius:8px;background-color:white;padding:50px">**

**<h3><strong>Add New TRAIN</strong></h3><hr>**

**<form method="post" action="" enctype="multipart/form-data">**

**{% csrf\_token %}**

**TRAIN Name:<br><input type="text" placeholder="Enter TRAIN Name" name="TRAINname" style="border-radius:8px;width:600px;height:35px;"><br><br>**

**TRAIN No:<br><input type="text" placeholder="Enter TRAIN no" name="TRAIN\_no" style="border-radius:8px;width:600px;height:35px;"><br><br>**

**From City:<br><input type="text" placeholder="Enter pick location" name="fcity" style="border-radius:8px;width:600px;height:35px;"><br><br>**

**To City:<br><input type="text" placeholder="Enter Drop location" name="tcity" style="border-radius:8px;width:600px;height:35px;"><br><br>**

**Departure Time:<br><input type="text" placeholder="Enter Departure time HH:MM am/pm" name="dtime" style="border-radius:8px;width:600px;height:35px;"><br><br>**

**Arrival Time:<br><input type="text" placeholder="Enter arrival time HH:MM am/pm" name="atime" style="border-radius:8px;width:600px;height:35px;"><br><br>**

**Travel Time:<br><input type="text" placeholder="Enter Travel time HH:MM" name="ttime" style="border-radius:8px;width:600px;height:35px;"><br><br>**

**Distance:<br><input type="number" placeholder="Enter Distance (in Km)" name="dis" style="border-radius:8px;width:600px;height:35px;"><br><br>**

**<br><input type="file" name="img" style="border-radius:8px;width:600px;height:35px;"><br><br>**

**<input type="submit" value="Add TRAIN" style="color:white;border-radius:8px;width:100px;height:35px;background-color:#333;border:1px solid black">**

**</form>**

**</div>**

**{% endblock %}**

**ADD TRAIN ROUTE FORM CODING**

**{% extends 'navigation3.html' %}**

**{% load static %}**

**{% block b %}**

**{% if error %}**

**alert('Route is added successfully')**

**{% endif %}**

**<style>**

**input[type=text][type=time]{**

**border-radius:8px;**

**width:600px;**

**height:16px;**

**}**

**</style>**

**{% if error %}**

**<script>**

**alert('Added New Route Sucessfully');**

**window.location='{% url 'availableroute' %}';**

**</script>**

**{% endif %}**

**<div style="color:Black;border:1px solid white;border-radius:8px;background-color:white;padding:50px">**

**<h3><strong>Add New Route</strong></h3><hr>**

**<form method="post" action="">**

**{% csrf\_token %}**

**<select name="TRAIN" style="border-radius:8px;width:600px;height:35px;">**

**{% for i in data %}**

**<option value="{{i.id}}">{{i.TRAINname}}.[{{i.TRAIN\_no}}]</option>**

**{% endfor %}**

**</select><br><br>**

**Route:<br><input type="text" placeholder="Enter Route" name="route" style="border-radius:8px;width:600px;height:35px;"><br><br>**

**Distance:<br><input type="number" placeholder="Enter Distance" name="dis" style="border-radius:8px;width:600px;height:35px;"><br><br>**

**Fare:<br><input type="number" placeholder="Enter fare" name="fare" style="border-radius:8px;width:600px;height:35px;"><br><br>**

**<input type="submit" value="Add Route" style="color:white;border-radius:8px;width:100px;height:35px;background-color:#333;border:1px solid black">**

**</form>**

**</div>**

**{% endblock %}**

**EDIT TRAIN ROUTE DETAIL PAGE CODING**

{% extends 'navigation3.html' %}

{% load static %}

{% block b %}

{% if error %}

alert('Route is added successfully')

{% endif %}

<style>

input[type=text][type=time]{

border-radius:8px;

width:600px;

height:16px;

}

</style>

{% if error %}

<script>alert('Update Route Sucessfully');

window.location='{% url 'availableroute' %}';

</script>

{% endif %}

<div style="color:Black;border:1px solid white;border-radius:8px;background-color:white;padding:50px">

<h3><strong>Edit Route</strong></h3><hr>

<form method="post" action="">

{% csrf\_token %}

<select name="TRAIN" style="border-radius:8px;width:600px;height:35px;">

{% for i in data2 %}

{% if i.id == data.TRAIN.id %}

<option value="{{i.id}}" selected>{{i.TRAINname}}.[{{i.TRAIN\_no}}]</option>

{% else %}

<option value="{{i.id}}">{{i.TRAINname}}.[{{i.TRAIN\_no}}]</option>

{% endif %}

{% endfor %}

</select><br><br>

Route:<br><input type="text" placeholder="Enter Route" name="route" value="{{data.route}}" style="border-radius:8px;width:600px;height:35px;"><br><br>

Distance:<br><input type="number" placeholder="Enter Distance" name="dis" value="{{data.distance}}" style="border-radius:8px;width:600px;height:35px;"><br><br>

Fare:<br><input type="number" placeholder="Enter fare" name="fare" value="{{data.fare}}" style="border-radius:8px;width:600px;height:35px;"><br><br>

<input type="submit" value="Update" style="color:white;border-radius:8px;width:100px;height:35px;background-color:#333;border:1px solid black">

</form>

</div>

{% endblock %}

**“VIEWS.PY” FILE CODING**

from django.shortcuts import render,redirect

from django.contrib.auth import authenticate,logout,login

from django.contrib.auth.models import User

from .models import \*

from django.contrib import messages

# Create your views here.

def nav(request):

return render(request,'carousel.html')

def About(request):

return render(request,'about.html')

def Contact(request):

return render(request,'contact.html')

def Login\_customer(request):

error = False

error2 = False

error3 = False

if request.method == "POST":

n = request.POST['uname']

p = request.POST['pwd']

try:

user = authenticate(username=n,password=p)

except:

error3 = True

try:

if user.is\_staff:

login(request,user)

error2 = True

elif user:

login(request, user)

error=True

except:

error3=True

d = {'error':error,'error2':error2,'error3':error3}

return render(request,'login\_customer.html',d)

def Register\_customer(request):

error = False

if request.method == "POST":

n = request.POST['uname']

f = request.POST['fname']

l = request.POST['lname']

e = request.POST['email']

a = request.POST['add']

m = request.POST['mobile']

g = request.POST['male']

d = request.POST['birth']

p = request.POST['pwd']

user = User.objects.create\_user(first\_name=f,last\_name=l,username=n,password=p,email=e)

Register.objects.create(user=user,add=a,mobile=m,gender=g,dob=d)

error = True

d = {'error':error}

return render(request,'register\_customer.html',d)

def Search\_TRAIN(request):

if not request.user.is\_authenticated:

return redirect('login')

data = Add\_route.objects.all()

ase = Asehi.objects.all()

coun = 7

error=False

fare3=0

count = 0

count1 = 0

data1=0

data2=0

route1=[]

route=0

b\_no =[]

b\_no1 =[]

bhu=0

if request.method=="POST":

f = request.POST["fcity"]

t = request.POST["tcity"]

da = request.POST["date"]

data1 = Add\_route.objects.filter(route=f)

data2 = Add\_route.objects.filter(route=t)

for i in data1:

for j in data2:

if i.TRAIN.TRAIN\_no==j.TRAIN.TRAIN\_no:

route1.append(Add\_TRAIN.objects.filter(TRAIN\_no=i.TRAIN.TRAIN\_no))

for i in data1:

fare1=i.fare

count+=1

b\_no.append(i.TRAIN.TRAIN\_no)

for i in data2:

fare2 = i.fare

count1+=1

b\_no1.append(i.TRAIN.TRAIN\_no)

fare3 = fare2-fare1

if fare3<5 and fare3>0:

fare3 = 5

elif fare3<0:

fare3 = fare3\*(-1)

elif fare3==0:

fare3 = fare3

route = f+" to "+t

Asehi.objects.create(fare=fare3,TRAIN\_name="TRAIN2",date3=da)

for i in ase:

coun = coun + 1

error=True

d={"data2":data,'route1':route1,'fare3':fare3,"error":error,'coun':coun,'route':route}

return render(request,'search\_TRAIN.html',d)

def Dashboard(request):

if not request.user.is\_authenticated:

return redirect('login')

return render(request,'dashboard.html')

def Logout(request):

logout(request)

return redirect('nav')

def Book\_detail(request,coun,pid,route1):

if not request.user.is\_authenticated:

return redirect('login')

error = False

data = Asehi.objects.get(id=coun)

data2 = Add\_TRAIN.objects.get(id=pid)

user2 = User.objects.filter(username=request.user.username).get()

user1 = Register.objects.filter(user=user2).get()

pro = Passenger.objects.filter(user=user1)

book = Book\_ticket.objects.filter(user=user1)

total = 0

for i in pro:

if i.status!="set":

total = total + i.fare

passenger=0

if request.method=="POST":

f = request.POST["name"]

t = request.POST["age"]

da = request.POST["gender"]

passenger = Passenger.objects.create(user=user1,TRAIN=data2,route=route1,name=f,gender=da,age=t,fare=data.fare,date1=data.date3)

Book\_ticket.objects.create(user=user1, route=route1, fare=total, passenger=passenger, date2=data.date3)

if passenger:

error = True

d = {'data':data,'data2':data2,'pro':pro,'total':total,'book':book,'error':error,'route1':route1,'coun':coun,'pid':pid}

return render(request,'book\_detail.html',d)

def Delete\_passenger(request,pid,bid,route1):

if not request.user.is\_authenticated:

return redirect('login')

data = Passenger.objects.get(id=pid)

data.delete()

ase = Asehi.objects.all()

coun = 7

for i in ase:

coun = coun + 1

messages.info(request,'Passenger Deleted Successfully')

return redirect('book\_detail', coun,bid,route1)

def Card\_Detail(request,total,coun,route1,pid):

if not request.user.is\_authenticated:

return redirect('login')

error=False

data = Asehi.objects.get(id=coun)

data2 = Add\_TRAIN.objects.get(id=pid)

user2 = User.objects.filter(username=request.user.username).get()

user1 = Register.objects.filter(user=user2).get()

pro = Passenger.objects.filter(user=user1)

book = Book\_ticket.objects.filter(user=user1)

count=0

pro1 = 0

if request.method == "POST":

error=True

for i in pro:

count = i.name

if i.status != "set":

i.status="set"

i.save()

return redirect('my\_booking')

total1=total

d = {'user':user1,'data':data,'data2':data2,'pro':pro,'pro1':pro1,'total':total1,'book':book,'error':error,'route1':route1,'count':count}

return render(request,'card\_detail.html',d)

def my\_booking(request):

if not request.user.is\_authenticated:

return redirect('login')

user2 = User.objects.filter(username=request.user.username).get()

user1 = Register.objects.filter(user=user2).get()

pro = Passenger.objects.filter(user=user1)

book = Book\_ticket.objects.filter(user=user1)

d = {'user':user1,'pro':pro,'book':book}

return render(request,'my\_booking.html',d)

def delte\_my\_booking(request,pid):

if not request.user.is\_authenticated:

return redirect('login')

error=False

pro = Passenger.objects.get(id=pid)

pro.delete()

error=True

d = {'error':error}

return render(request,'my\_booking.html',d)

def Add\_TRAIN(request):

if not request.user.is\_authenticated:

return redirect('login')

error=False

if request.method == "POST":

n = request.POST['TRAINname']

no = request.POST['TRAIN\_no']

f = request.POST['fcity']

to= request.POST['tcity']

de= request.POST['dtime']

a = request.POST['atime']

t = request.POST['ttime']

d = request.POST['dis']

i = request.FILES['img']

Add\_TRAIN.objects.create(TRAINname=n,TRAIN\_no=no,from\_city=f,to\_city=to,departuretime=de,arrivaltime=a,trevaltime=t,distance=d,img=i)

error=True

d={"error":error}

return render(request,'add\_TRAIN.html',d)

def view\_TRAIN(request):

if not request.user.is\_authenticated:

return redirect('login')

data=Add\_TRAIN.objects.all()

d={"data":data}

return render(request,"view\_TRAIN.html",d)

def add\_route(request):

error=False

data=Add\_TRAIN.objects.all()

if request.method == "POST":

b = request.POST['TRAIN']

r = request.POST['route']

f= request.POST['fare']

d = request.POST['dis']

TRAIN1 = Add\_TRAIN.objects.filter(id=b).get()

Add\_route.objects.create(TRAIN=TRAIN1,route=r,distance=d,fare=f)

error = True

d={"data":data,"error":error}

return render(request,'add\_route.html',d)

def Edit\_route(request,pid):

if not request.user.is\_authenticated:

return redirect('login')

error=False

data=Add\_route.objects.get(id=pid)

data2=Add\_TRAIN.objects.all()

if request.method == "POST":

b = request.POST['TRAIN']

r = request.POST['route']

f= request.POST['fare']

d = request.POST['dis']

a = Add\_TRAIN.objects.filter(id=b).first()

data.TRAIN = a

data.route = r

data.fare = f

data.distance = d

data.save()

error=True

d={"data":data,"data2":data2,"error":error}

return render(request,'editroute.html',d)

def edit(request,pid):

if not request.user.is\_authenticated:

return redirect('login')

error = False

data1=Add\_TRAIN.objects.get(id=pid)

if request.method == "POST":

n = request.POST['TRAINname']

no = request.POST['TRAIN\_no']

de= request.POST['dtime']

a = request.POST['atime']

t = request.POST['ttime']

f = request.POST['fcity']

to= request.POST['tcity']

d = request.POST['dis']

data1.TRAINname=n

data1.TRAIN\_no=no

data1.from\_city=f

data1.to\_city=to

data1.departuretime=de

data1.arrivaltime=a

data1.traveltime=t

data1.distance=d

data1.save()

error = True

d = {'data':data1,'error':error}

return render(request,'editTRAIN.html',d)

def delete(request,pid):

if not request.user.is\_authenticated:

return redirect('login')

error2=False

data=Add\_TRAIN.objects.get(id=pid)

data.delete()

error2=True

d = {'error2':error2}

return render(request,"view\_TRAIN.html",d)

def delete\_route(request,pid):

if not request.user.is\_authenticated:

return redirect('login')

error=False

data=Add\_route.objects.get(id=pid)

data.delete()

error = True

d = {'error2':error}

return render(request,"availableroute.html",d)

def displayroute(request):

if not request.user.is\_authenticated:

return redirect('login')

data = Add\_route.objects.all()

data2 = Add\_TRAIN.objects.all()

d = {'data':data,'data2':data2}

return render(request,"availableroute.html",d)

def admindashboard(request):

if not request.user.is\_authenticated:

return redirect('login')

return render(request,'admindashboard.html')

**“MODELS.PY” FILE CODING**

from django.db import models

from django.contrib.auth.models import User

# Create your models here.

class Register(models.Model):

user = models.ForeignKey(User,on\_delete=models.CASCADE,null=True)

mobile = models.CharField(max\_length=10,null=True)

add = models.CharField(max\_length=100,null=True)

dob = models.DateField(null=True)

gender = models.CharField(max\_length=10,null=True)

def \_\_str\_\_(self):

return self.user.first\_name

class Add\_TRAIN(models.Model):

TRAINname = models.CharField(max\_length=30,null=True)

TRAIN\_no = models.IntegerField(null=True)

from\_city = models.CharField(max\_length=30,null=True)

to\_city = models.CharField(max\_length=30,null=True)

departuretime=models.CharField(max\_length=30,null=True)

arrivaltime=models.CharField(max\_length=30,null=True)

trevaltime=models.CharField(max\_length=100,null=True)

distance=models.IntegerField(null=True)

img=models.FileField(null=True)

def \_\_str\_\_(self):

return self.TRAINname+" "+str(self.TRAIN\_no)

class Add\_route(models.Model):

TRAIN = models.ForeignKey(Add\_TRAIN,on\_delete=models.CASCADE,null=True)

route = models.CharField(max\_length=100,null=True)

distance=models.IntegerField(null=True)

fare=models.IntegerField(null=True)

def \_\_str\_\_(self):

return self.route+" "+str(self.TRAIN.TRAIN\_no)

class Passenger(models.Model):

user = models.ForeignKey(Register,on\_delete=models.CASCADE,null=True)

TRAIN = models.ForeignKey(Add\_TRAIN,on\_delete=models.CASCADE,null=True)

name = models.CharField(max\_length=100,null=True)

age = models.IntegerField(null=True)

gender = models.CharField(max\_length=30,null=True)

route=models.CharField(max\_length=100,null=True)

status = models.CharField(max\_length=30,null=True)

date1 = models.DateField(null=True)

fare = models.IntegerField(null=True)

def \_\_str\_\_(self):

return self.user.user.username+" "+self.name

class Book\_ticket(models.Model):

passenger=models.ForeignKey(Passenger,on\_delete=models.CASCADE,null=True)

user=models.ForeignKey(Register,on\_delete=models.CASCADE,null=True)

route=models.CharField(max\_length=100,null=True)

date2=models.DateField(null=True)

fare=models.IntegerField(null=True)

def \_\_str\_\_(self):

return self.user.user.username+" "+self.route

class Asehi(models.Model):

fare = models.IntegerField(null=True)

TRAIN\_name = models.CharField(max\_length=30,null=True)

date3 = models.DateField(null=True)

**CHAPTER** # **8**

**Advantages & Limitations**

**Advantages of “Online Railway Reservation System”:**

“Online Railway Reservation System” provides various features, which complement the information system and increase the productivity of the system. These features make the system easily usable and convenient. Some of the important features included are listed as follows:

* Intelligent User Forms Design
  + - Data access and manipulation through same forms
    - Access to most required information
* Data Security
* Restrictive data access, as per login assigned only.
* Organized and structured storage of facts.

.

* Strategic Planning made easy.
* No decay of old Records.
* Exact financial position of the business.

**Limitations of “Online Railway Reservation System”:**

Besides the above achievements and the successful completion of the project, we still feel the project has some limitations, listed as below:

* + It is not a large scale system.
  + Only limited information provided by this system.
  + Since it is an online project, customers need internet connection to buy products.
  + People who are not familiar with computers can’t use this software.

**CHAPTER # 8**

**Future Scope**

**FUTURE SCOPE**

Thisweb application involves almost all the features of the online train ticket booking. The future implementation will be online help for the customers and chatting with website administrator.

**CONCLUSION**

The project entitled “Online Train Ticket Booking System” is developed using HTML, CSS and Bootstrap as front end and Python Django and Sqlite database in back end to computerize the process of online train ticket booking. This project covers only the basic features required.

.

**Bibliography**

**BIBLIOGRAPHY**

* Wikipedia
* <https://www.geeksforgeeks.org/python-django/>
* <https://www.javatpoint.com>
* <https://www.python.org/>
* <https://www.tutorialspoint/>
* **REFERENCE BOOKS**

### Two scoops of Django for 1.11 by Daniel Greenfeld’s and Audrey Greenfield

### Lightweight Django by Elman and Mark Lavin