

Final Report

(Railway Management System)

Course Code: CS254

Course Title: DBMS Lab

Semester: B. Tech 4th Sem

Section: S2

Academic Year: 2020-21

Course Instructor: Dr. Annappa B and

Mr. Sharath Yaji

Team Members:

1. Praneeth G, 191CS235, 9481107349, praneethg.191cs235@nitk.edu.in
2. B Videep Kumar Reddy, 191CS215, 8431853391, videepreddy.191cs215 @nitk.edu.in

1 Abstract

Railway Management System

Course Code: CS254

Course Title: DBMS LAB

Semester: B. Tech 4th Sem

Section: S2

Academic Year: 2020-21

Course Instructor: Dr. Annappa B

and Mr. Sharath Yaji

Team Members:

1. Praneeth G, 191CS235, 9481107349, praneethg.191cs235@nitk.edu.in
2. B Videep Kumar Reddy, 191CS215, 8431853391, videepreddy.191cs215 @nitk.edu.in

Brief Description:

The main aim of this Railway Management System project is to develop a gateway which would facilitate the reservation of train tickets and a gateway for Train officials To perform respective their duties.

This project replicates the processes which happen in a Railway Station.

There will be a homepage through which a user can navigate to a booking page, security login page, train staff login page.

After which, he/she can book a ticket.

Then the user can also see his/her bookings and also can even check train details.

A user will be allowed to check train details.

Security personal and Train Staff can log in and perform their respective functionalities such as security check and confirming passengers.

Key Features:

1. Login facility
2. Ticket Booking
3. Ticket Cancellation
4. Train Details
5. Show Bookings

Software Specifications:

- Frontend: HTML ,CSS, JavaScript
- Backend: Django

References:

1. <https://www.djangoproject.com/>
2. <https://docs.google.com/viewer?url=https://www.computer-pdf.com/pdf/0911-learning-django.pdf>

2 Introduction

The main aim of this Railway Management System project is to develop a gateway which would facilitate the booking of train tickets.

This project replicates the processes which happen in a Railway Station.

- * Ticket Booking
- * Security Check
- * Train Clearance

Railway Management System is a web application, developed to maintain the details of passengers, security personnel and train staff in a Railway Station.

It maintains the information about the personal details of the passengers and their bookings.

The details of the security personnel and the train staff for each train is also stored.

They can log in and perform their respective functionalities such as security check and confirming passengers.

This software package has been developed using HTML, CSS, Bootstrap and Javascript at Front End and Django at Back End with MySQL Server database.

This application works on the following tables in Database:

Tables:

- * Passenger
- * Trains
- * Train Staff
- * Security

3 ER Diagram

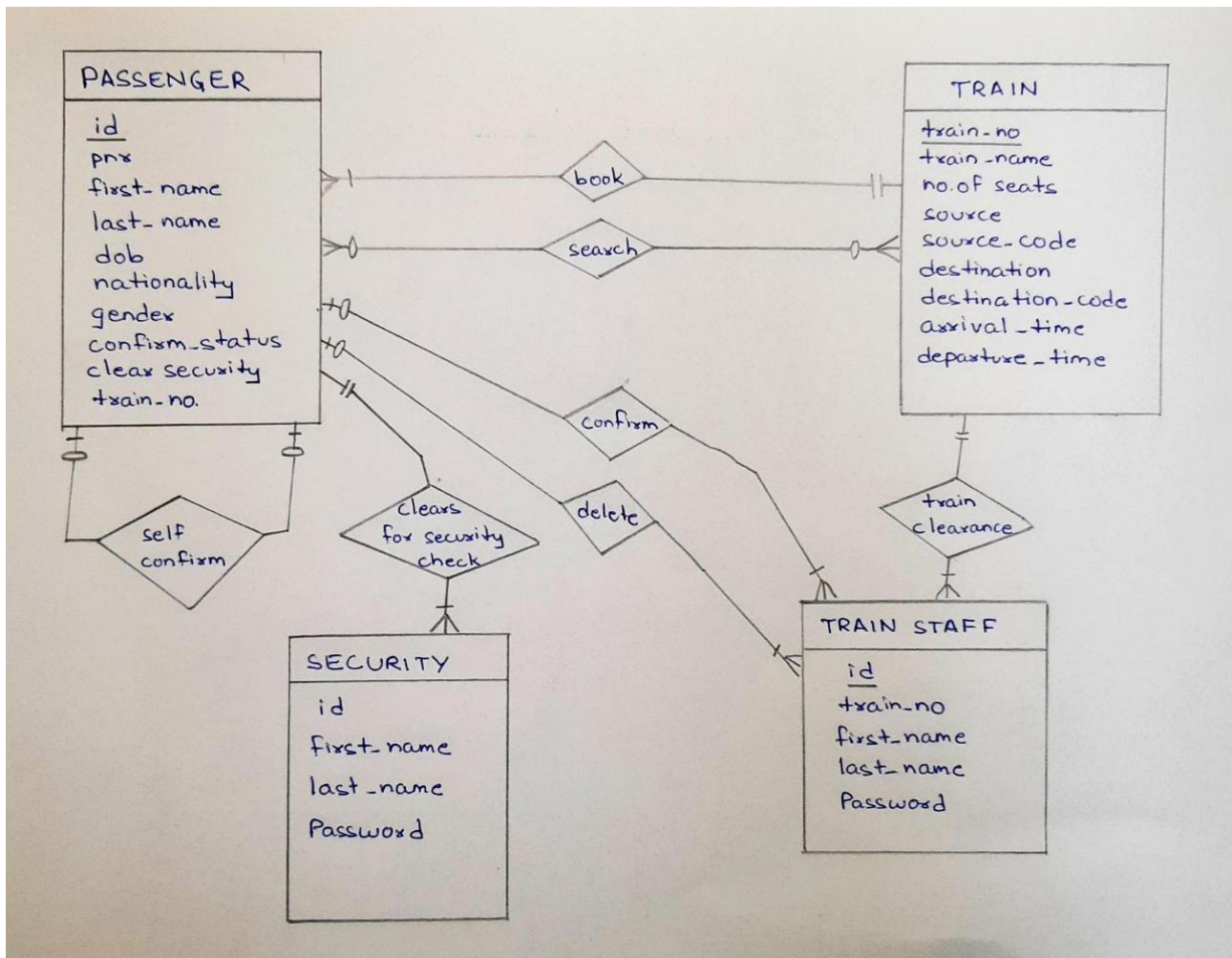


Figure 1: ER-Diagram

4 Source Code

For full code refer the following link:

<https://github.com/praneeth-github/Railway-managment-system>

Few important functions are shown.

trains.model.py

```
7
8 class Train(models.Model):
9     train_no = models.IntegerField(primary_key=True, default=1007)
10    train_name = models.CharField(max_length=50)
11    no_of_seats = models.IntegerField(default=0)
12    source = models.CharField(max_length=50)
13    source_code = models.CharField(max_length=3)
14    destination = models.CharField(max_length=50)
15    destination_code = models.CharField(max_length=3)
16    arrival_time = models.DateTimeField()
17    departure_time = models.DateTimeField()
18
19
20 class Passenger(models.Model):
21    pnr = models.CharField(max_length=10)
22    first_name = models.CharField(max_length=50)
23    last_name = models.CharField(max_length=50)
24    dob = models.DateField(default=1/1/1990)
25    nationality = models.CharField(max_length=50)
26    gender = models.CharField(max_length=1)
27    train_no = models.ForeignKey(Train, on_delete=models.CASCADE, default=1007)
28    checked_in_status = models.BooleanField(default=0)
29    cleared_security_status = models.IntegerField(default=0)
30
```

```
31
32 class User(AbstractUser):
33     USER_TYPE_CHOICES = (
34         (1, 'trainstaff'),
35         (2, 'security'),
36         (3, 'admin'),
37     )
38
39     user_type = models.PositiveSmallIntegerField(choices=USER_TYPE_CHOICES, default=3)
40
41
42 class Security(models.Model):
43     user = models.OneToOneField(User, on_delete=models.CASCADE, default=2)
44     id = models.IntegerField(primary_key=True)
45
46
47
48 class Staff(models.Model):
49     user = models.OneToOneField(User, on_delete=models.CASCADE, default=1)
50     id = models.IntegerField(primary_key=True)
51     train_no = models.ForeignKey(Train, on_delete=models.CASCADE, default=1007)
52
```

```

14 def login_security(request):
15     if request.method == "POST":
16         username = request.POST['username']
17         password = request.POST['password']
18         user = authenticate(username=username, password=password)
19         if user is not None:
20             if user.is_active:
21                 if user.user_type == 2:
22                     login(request, user)
23                     return redirect('security_clearing') #, pk=user.security.id)
24                 else:
25                     return render(request, 'security_login.html', {'error_message': 'Invalid security staff credentials'})
26             else:
27                 return render(request, 'security_login.html', {'error_message': 'Your account has been disabled'})
28         else:
29             return render(request, 'security_login.html', {'error_message': 'Invalid login'})
30     return render(request, 'security_login.html')
31
32
33 def login_staff(request):
34     if request.method == "POST":
35         username = request.POST['username']
36         password = request.POST['password']
37         user = authenticate(username=username, password=password)
38         if user is not None:
39             if user.is_active:
40                 if user.user_type == 1:
41                     login(request, user)
42                     return redirect('staff_home', train_no=user.staff.train_no.train_no)
43                 else:
44                     return render(request, 'staff_login.html', {'error_message': 'Invalid train staff credentials'})
45             else:
46                 return render(request, 'staff_login.html', {'error_message': 'Your account has been disabled'})
47         else:
48             return render(request, 'staff_login.html', {'error_message': 'Invalid login'})
49     return render(request, 'staff_login.html')

```

trains.views.py

```

52 def home(request):
53     return render(request, 'home.html')
54
55
56 def station_mgmt(request):
57     return render(request, 'station_mgmt.html')
58
59
60 def clear_security(request):
61     data = Passenger.objects.all()
62     return render(request, 'security_clearing.html', {'passengers': data})
63
64
65 def clear_for_security(request, pk):
66     passenger = get_object_or_404(Passenger, pk=pk)
67     passenger.cleared_security_status = request.user.security.id
68     passenger.save()
69
70     return redirect('security_clearing')
71
72
73 def staff_home(request, train_no):
74     # staff = get_object_or_404(Staff, pk=pk)
75     data = Passenger.objects.filter(train_no=train_no)
76     return render(request, 'staff_home.html', {'passengers': data, 'train_no': train_no})
77
78
79 def view_trains(request):
80     data = Train.objects.all()
81     return render(request, 'view_trains.html', {'trains': data})
82

```

```

84 def self_check_in(request, pk):
85     passenger = get_object_or_404(Passenger, pk=pk)
86     passenger.checked_in_status = True
87     passenger.save()
88     return redirect('passenger_home', pk=passenger.pk)
89
90
91 def search_by_source(request):
92     if request.method == "POST":
93         source = request.POST['source']
94         if source:
95             data = Train.objects.filter(source=source)
96             return render(request, 'view_trains.html', {'trains': data})
97         else:
98             return redirect('view_trains')
99     else:
100         return redirect('view_trains')
101
102
103 def search_by_destination(request):
104     if request.method == "POST":
105         destination = request.POST['destination']
106         if destination:
107             data = Train.objects.filter(destination=destination)
108             return render(request, 'view_trains.html', {'trains': data})
109         else:
110             return redirect('view_trains')
111     else:
112         return redirect('view_trains')
113

```

```

114
115 def view_available_trains(request):
116     if request.method == "POST":
117         source = request.POST['source']
118         destination = request.POST['destination']
119         if source and destination:
120             trains = Train.objects.filter(source=source, destination=destination)
121             if trains:
122                 return render(request, 'home.html', {'trains': trains})
123             else:
124                 return render(request, 'home.html', {'error_message_train': "No trains found"})
125         else:
126             return redirect('home')
127     else:
128         return redirect('home')
129
130
131 def book_train(request, pk):
132     if request.method == "POST":
133         train = Train.objects.get(train_no=pk)
134         first_name = request.POST['first_name']
135         last_name = request.POST['last_name']
136         nationality = request.POST['nationality']
137         gender = request.POST['gender']
138         dob = request.POST['dob']
139         pnr = str(train.train_no) + str(train.destination_code)
140         passenger = Passenger(pnr=pnr, first_name=first_name, last_name=last_name, nationality=nationality,
141                               train_no=train, gender=gender, dob=dob)
142         passenger.save()
143         passenger.pnr = str(train.train_no) + str(train.destination_code) + str(passenger.pk)
144         passenger.save()
145         train.no_of_seats -= 1
146         train.save()
147         return redirect('passenger_home', pk=passenger.pk)
148     else:
149         return render(request, 'book_train.html', {'train_no': pk})
150

```

```

151
152 def view_booking(request):
153     if request.method == "POST": # view existing booking
154         pnr = request.POST['pnr']
155         try:
156             passenger = Passenger.objects.get(pnr=pnr)
157         except Passenger.DoesNotExist:
158             passenger = None
159         if passenger:
160             passenger = get_object_or_404(Passenger, pnr=pnr)
161             return render(request, 'passenger_home.html', {'passenger': passenger})
162         else:
163             return render(request, 'home.html', {'error_message_booking': 'No booking found'})
164

```

```

180 def staff_check_in(request, pk):
181     passenger = get_object_or_404(Passenger, pk=pk)
182     passenger.checked_in_status = True
183     passenger.save()
184     return redirect('staff_home', train_no=passenger.train_no.train_no)
185
186
187 def delete_passengers(request, train_no):
188     train = Train.objects.get(train_no=train_no)
189     passenger = Passenger.objects.filter(train_no=train)
190     passenger.delete()
191     return redirect('staff_home', train_no=train.train_no)
192

```

base.html

```

1 {% load static %}
2 <!DOCTYPE html>
3 <html lang="en">
4 <head>
5     <meta charset="UTF-8">
6     <title>{% block title %}{% endblock %}</title>
7     {% load staticfiles %}
8     {# <link href="https://stackpath.bootstrapcdn.com/bootswatch/4.1.3/united/bootstrap.min.css" rel="stylesheet" integrity="sha384-+d4wMJSxEP3vzs2qZl
9     <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">#}
10     <link href="https://bootswatch.com/4/united/bootstrap.min.css" rel="stylesheet" >
11     {# integrity="sha384-+d4wMJSxEP3vzs2qZBElQRITZMxwGwH15Nyn2K/9XjKiHmh3sBuk1Un/IbcdMcYC4" crossorigin="anonymous">#}
12
13     <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
14     <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>
15 </head>
16 <body>
17     <nav class="navbar navbar-expand-lg navbar-dark bg-primary">
18     <a class="navbar-brand" >Railway Management System</a>
19     <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarColor01" aria-controls="navbarColor01" aria-expanded="false"
20     <span class="navbar-toggler-icon"></span>
21     </button>
22
23     <div class="collapse navbar-collapse" id="navbarColor01">
24     {% with url_name=request.resolver_match.url_name %}
25     <ul class="navbar-nav mr-auto">
26         <li class="nav-item {% if url_name == 'home' %}active{% endif %}">
27             <a class="nav-link" href="{% url 'home' %}">Home <span class="sr-only">(current)</span></a>
28         </li>
29         <li class="nav-item {% if url_name == 'view_trains' %}active{% endif %}">
30             <a class="nav-link" href="{% url 'view_trains' %}">View trains</a>
31         </li>
32         <li class="nav-item {% if url_name == 'security_login' %}active{% endif %}">
33             <a class="nav-link" href="{% url 'security_login' %}">Login as Security</a>
34         </li>
35         <li class="nav-item {% if url_name == 'staff_login' %}active{% endif %}">
36             <a class="nav-link" href="{% url 'staff_login' %}">Login as Staff</a>
37         </li>

```

```

38         {% if user.is_authenticated %}
39         <li class="nav-item">
40             <a class="nav-link" href="{% url 'logout' %}">Logout</a>
41         </li>
42         {% endif %}
43     </ul>
44     {% endwith %}
45     <!-- <form class="form-inline my-2 my-lg-0">
46         <input class="form-control mr-sm-2" placeholder="Search" type="text">
47         <button class="btn btn-secondary my-2 my-sm-0" type="submit">Search</button>
48     </form> -->
49 </div>
50 </nav>
51 <div style="margin-left:20px; margin-top:20px">
52 {% block body %}{% endblock %}
53 </div>
54
55 <footer class="footer" style="margin-left:20px; margin-top:20px">
56     <small>
57         Module made by <a href="https://github.com/praneeth-github">PRANEETH G</a> and <a href="https://github.com/videep1">B VIDEEP KUMAR REDDY</a>
58     </small>
59 </footer>
60
61 </body>
62 </html>

```


5 Results

Railway Management System [Home](#) [View trains](#) [Login as Security](#) [Login as Staff](#) [Logout](#)

Module made by PRANEETH G and B VIDEEP KUMAR REDDY

Figure 2: Home Page

Railway Management System [Home](#) [View trains](#) [Login as Security](#) [Login as Staff](#) [Logout](#)

train No.	Train	No. of seats	Source	Destination	Departure Time	Arrival Time	
1008	Shatabdi Express	10	Bangalore	Chennai	March 28, 2021, 1 p.m.	March 28, 2021, noon	<input data-bbox="1300 790 1385 824" type="button" value="Book Now!"/>
1009	Brindavan Express	10	Bangalore	Chennai	March 29, 2021, 7 a.m.	March 29, 2021, 6 a.m.	<input data-bbox="1300 846 1385 880" type="button" value="Book Now!"/>

Module made by PRANEETH G and B VIDEEP KUMAR REDDY

Figure 3: Search Trains

Railway Management System [Home](#) [View trains](#) [Login as Security](#) [Login as Staff](#)

trains

Train No.	Train Name	No. of seats	Source	Destination	Departure Time	Arrival Time	
1007	Rajdhani Express	5	(BLR) Bangalore	(DHL) Delhi	March 28, 2021, 8 p.m.	March 28, 2021, 6 p.m.	<input data-bbox="1300 1384 1385 1417" type="button" value="Book Now!"/>
1008	Shatabdi Express	10	(BLR) Bangalore	(CHN) Chennai	March 28, 2021, 1 p.m.	March 28, 2021, noon	<input data-bbox="1300 1440 1385 1473" type="button" value="Book Now!"/>
1009	Brindavan Express	10	(BLR) Bangalore	(CHN) Chennai	March 29, 2021, 7 a.m.	March 29, 2021, 6 a.m.	<input data-bbox="1300 1496 1385 1529" type="button" value="Book Now!"/>

Module made by PRANEETH G and B VIDEEP KUMAR REDDY

Figure 4: View all Trains

Railway Management System

[Home](#) [View trains](#) [Login as Security](#) [Login as Staff](#) [Logout](#)

Enter your details

First Name

Rahul

Last Name

Kumar

Gender

☒ Male ☐ Female

Date of Birth

10-06-1992

Nationality

Indian

Book

Module made by PRANEETH G and B VIDEEP KUMAR REDDY

Figure 5: Book Ticket

Railway Management System

[Home](#) [View trains](#) [Login as Security](#) [Login as Staff](#) [Logout](#)

Passenger Details of PNR: 1007DHL6

Name: Rahul Kumar

Train Name: Rajdhani Express

Departure: March 28, 2021, 8 p.m.

Arrival: March 28, 2021, 6 p.m.

From: Bangalore

To: Delhi

Perform Self Confirmation

Module made by PRANEETH G and B VIDEEP KUMAR REDDY

Figure 6: Show Booking

Railway Management System

[Home](#) [View trains](#) [Login as Security](#) [Login as Staff](#) [Logout](#)

Security Staff Login

Username

praneethg

Password

Submit

Module made by PRANEETH G and B VIDEEP KUMAR REDDY

Figure 7: Secourity Login

Railway Management System
Home
View trains
Login as Security
Login as Staff
Logout

Security

PNR	Name	train No.	Cleared Security Status
1007DHL6	Rahul Kumar	1007	0
1007DHL7	Amit Singh	1007	0

Clear for security
Clear for security

Module made by PRANEETH G and B VIDEEP KUMAR REDDY

Figure 8: Security Check

Railway Management System
Home
View trains
Login as Security
Login as Staff
Logout

train Staff Login

Username
videepb

Password

Submit

Module made by PRANEETH G and B VIDEEP KUMAR REDDY

Figure 9: Train Staff Login

Railway Management System
Home
View trains
Login as Security
Login as Staff
Logout

Staff

Train No.	PNR	Name	Date of Birth	Nationality	Gender	Confirmation Status	Cleared Security Status
1007	1007DHL6	Rahul Kumar	June 10, 1992	Indian	M	Confirm	2
1007	1007DHL7	Amit Singh	May 16, 1995	Indian	M	Confirmed	2

All Confirmed GO GREEN FLAG

Module made by PRANEETH G and B VIDEEP KUMAR REDDY

Figure 10: Confirming Tickets

6 References:

1. <https://docs.djangoproject.com/>
2. <https://www.digitalocean.com/community/tutorials/how-to-create-a-django-app-and-connect-it-to-a-database>
3. <https://youtu.be/F5mRW0jo-U4>
4. <https://docs.djangoproject.com/en/3.1/topics/auth/default/>

****** END ******