Final Report
(Railway Management System)

Course Code: CS254 **Course Title:** DBMS Lab

Semester: B. Tech 4^{th} 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 4^{th} 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

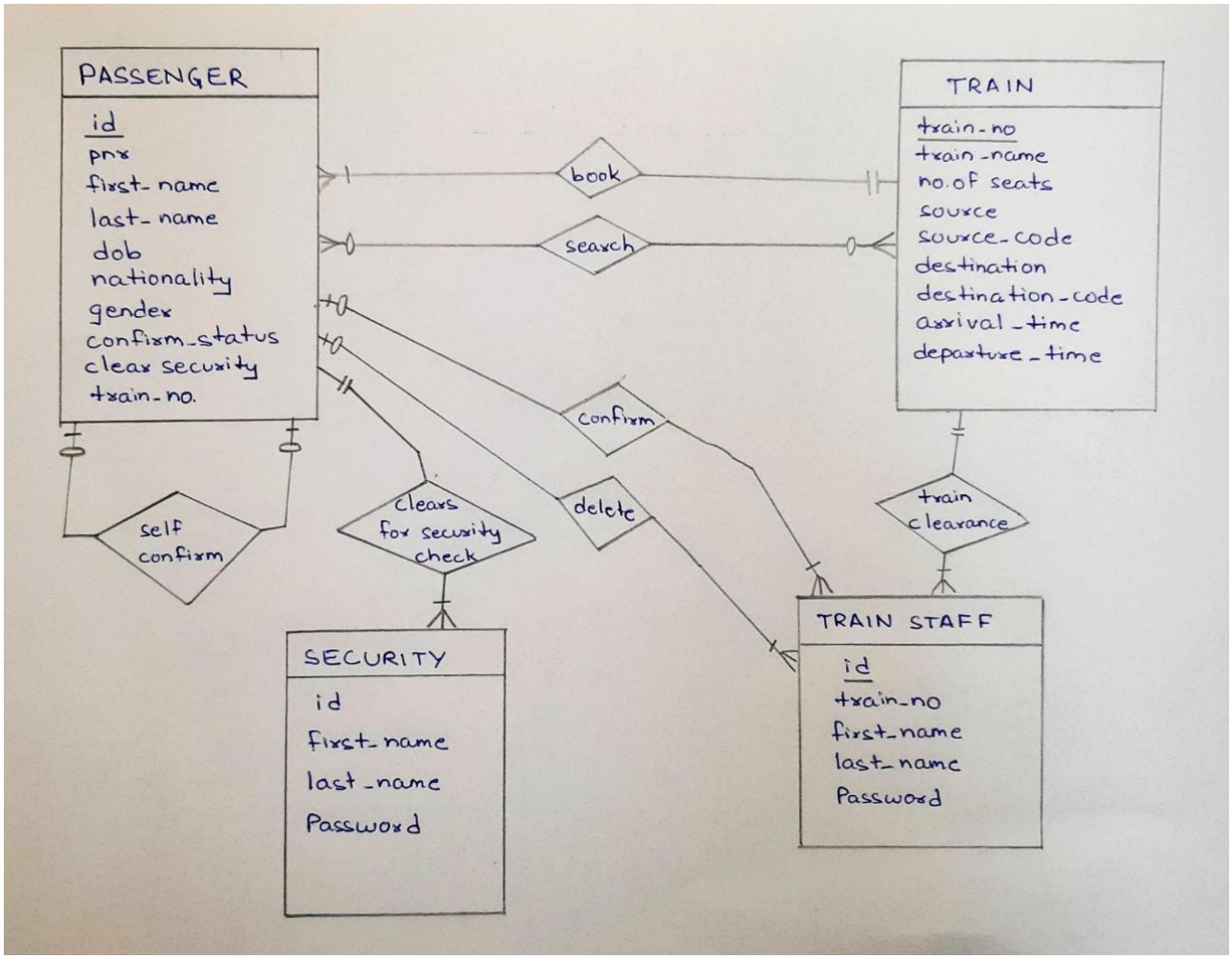

Figure 1: ER-Diagram

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

```
class Train(models.Model):
    train_no = models.IntegerField(primary_key=True, default=1007)
    train_name = models.CharField(max_length=50)
    no_of_seats = models.IntegerField(default=0)
    source = models.CharField(max_length=50)
    source_code = models.CharField(max_length=3)
    destination = models.CharField(max_length=50)
    destination_code = models.CharField(max_length=3)
    arrival_time = models.DateTimeField()
    departure_time = models.DateTimeField()


class Passenger(models.Model):
    pnr = models.CharField(max_length=10)
    first_name = models.CharField(max_length=50)
    last_name = models.CharField(max_length=50)
    dob = models.DateField(default=1/1/1990)
    nationality = models.CharField(max_length=50)
    gender = models.CharField(max_length=1)
    train_no = models.ForeignKey(Train, on_delete=models.CASCADE, default=1007)
    checked_in_status = models.BooleanField(default=0)
    cleared_security_status = models.IntegerField(default=0)
```

```
class User(AbstractUser):
    USER_TYPE_CHOICES = (
      (1, 'trainstaff'),
      (2, 'security'),
      (3, 'admin'),
    )

    user_type = models.PositiveSmallIntegerField(choices=USER_TYPE_CHOICES, default=3)


class Security(models.Model):
    user = models.OneToOneField(User, on_delete=models.CASCADE, default=2)
    id = models.IntegerField(primary_key=True)



class Staff(models.Model):
    user = models.OneToOneField(User, on_delete=models.CASCADE, default=1)
    id = models.IntegerField(primary_key=True)
    train_no = models.ForeignKey(Train, on_delete=models.CASCADE, default=1007)
```

```python
def login_security(request):
    if request.method == "POST":
        username = request.POST['username']
        password = request.POST['password']
        user = authenticate(username=username, password=password)
        if user is not None:
            if user.is_active:
                if user.user_type == 2:
                    login(request, user)
                    return redirect('security_clearing') #, pk=user.security.id)
                else:
                    return render(request, 'security_login.html', {'error_message': 'Invalid security staff credentials'})
            else:
                return render(request, 'security_login.html', {'error_message': 'Your account has been disabled'})
        else:
            return render(request, 'security_login.html', {'error_message': 'Invalid login'})
    return render(request, 'security_login.html')


def login_staff(request):
    if request.method == "POST":
        username = request.POST['username']
        password = request.POST['password']
        user = authenticate(username=username, password=password)
        if user is not None:
            if user.is_active:
                if user.user_type == 1:
                    login(request, user)
                    return redirect('staff_home', train_no=user.staff.train_no.train_no)
                else:
                    return render(request, 'staff_login.html', {'error_message': 'Invalid train staff credentials'})
            else:
                return render(request, 'staff_login.html', {'error_message': 'Your account has been disabled'})
        else:
            return render(request, 'staff_login.html', {'error_message': 'Invalid login'})
    return render(request, 'staff_login.html')
```

trains.views.py

```python
def home(request):
    return render(request, 'home.html')


def station_mgmt(request):
    return render(request, 'station_mgmt.html')


def clear_security(request):
    data = Passenger.objects.all()
    return render(request, 'security_clearing.html', {'passengers': data})


def clear_for_security(request, pk):
    passenger = get_object_or_404(Passenger, pk=pk)
    passenger.cleared_security_status = request.user.security.id
    passenger.save()

    return redirect('security_clearing')


def staff_home(request, train_no):
    # staff = get_object_or_404(Staff, pk=pk)
    data = Passenger.objects.filter(train_no=train_no)
    return render(request, 'staff_home.html', {'passengers': data, 'train_no': train_no})


def view_trains(request):
    data = Train.objects.all()
    return render(request, 'view_trains.html', {'trains': data})

```

```python
def self_check_in(request, pk):
    passenger = get_object_or_404(Passenger, pk=pk)
    passenger.checked_in_status = True
    passenger.save()
    return redirect('passenger_home', pk=passenger.pk)


def search_by_source(request):
    if request.method == "POST":
        source = request.POST['source']
        if source:
            data = Train.objects.filter(source=source)
            return render(request, 'view_trains.html', {'trains': data})
        else:
            return redirect('view_trains')
    else:
        return redirect('view_trains')


def search_by_destination(request):
    if request.method == "POST":
        destination = request.POST['destination']
        if destination:
            data = Train.objects.filter(destination=destination)
            return render(request, 'view_trains.html', {'trains': data})
        else:
            return redirect('view_trains')
    else:
        return redirect('view_trains')
```

```python
def view_available_trains(request):
    if request.method == "POST":
        source = request.POST['source']
        destination = request.POST['destination']
        if source and destination:
            trains = Train.objects.filter(source=source, destination=destination)
            if trains:
                return render(request, 'home.html', {'trains': trains})
            else:
                return render(request, 'home.html', {'error_message_train': "No trains found"})
        else:
            return redirect('home')
    else:
        return redirect('home')


def book_train(request, pk):
    if request.method == "POST":
        train = Train.objects.get(train_no=pk)
        first_name = request.POST['first_name']
        last_name = request.POST['last_name']
        nationality = request.POST['nationality']
        gender = request.POST['gender']
        dob = request.POST['dob']
        pnr = str(train.train_no) + str(train.destination_code)
        passenger = Passenger(pnr=pnr, first_name=first_name, last_name=last_name, nationality=nationality,
                              train_no=train, gender=gender, dob=dob)
        passenger.save()
        passenger.pnr = str(train.train_no) + str(train.destination_code) + str(passenger.pk)
        passenger.save()
        train.no_of_seats -= 1
        train.save()
        return redirect('passenger_home', pk=passenger.pk)
    else:
        return render(request, 'book_train.html', {'train_no': pk})
```

```python
def view_booking(request):
    if request.method == "POST":  # view existing booking
        pnr = request.POST['pnr']
        try:
            passenger = Passenger.objects.get(pnr=pnr)
        except Passenger.DoesNotExist:
            passenger = None
        if passenger:
            passenger = get_object_or_404(Passenger, pnr=pnr)
            return render(request, 'passenger_home.html', {'passenger': passenger})
        else:
            return render(request, 'home.html', {'error_message_booking': 'No booking found'})
```

```
def staff_check_in(request, pk):
    passenger = get_object_or_404(Passenger, pk=pk)
    passenger.checked_in_status = True
    passenger.save()
    return redirect('staff_home', train_no=passenger.train_no.train_no)


def delete_passengers(request, train_no):
    train = Train.objects.get(train_no=train_no)
    passenger = Passenger.objects.filter(train_no=train)
    passenger.delete()
    return redirect('staff_home', train_no=train.train_no)
```

base.html

```
{% load static %}
<!DOCTYPE html5>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>{% block title %}{% endblock %}</title>
    {% load staticfiles %}
{#    <link href="https://stackpath.bootstrapcdn.com/bootswatch/4.1.3/united/bootstrap.min.css" rel="stylesheet" integrity="sha384-+d4wMJSxEP3vzs2qZ
{#     <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">#}
      <link href="https://bootswatch.com/4/united/bootstrap.min.css" rel="stylesheet" >
{#    integrity="sha384-+d4wMJSxEP3vzs2qZBElQRTZMXwgWH15Nyn2K/9XjKiHmh3sBuk1Un/IbcdMcYC4" crossorigin="anonymous">#}

    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>
</head>
<body>
    <nav class="navbar navbar-expand-lg navbar-dark bg-primary">
  <a class="navbar-brand" >Railway Management System</a>
  <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarColor01" aria-controls="navbarColor01" aria-expanded="fals
    <span class="navbar-toggler-icon"></span>
  </button>

  <div class="collapse navbar-collapse" id="navbarColor01">
    {% with url_name=request.resolver_match.url_name %}
    <ul class="navbar-nav mr-auto">
      <li class="nav-item {% if url_name == 'home' %}active{% endif %}">
        <a class="nav-link" href="{% url 'home' %}">Home <span class="sr-only">(current)</span></a>
      </li>
      <li class="nav-item {% if url_name == 'view_trains' %}active{% endif %}">
        <a class="nav-link" href="{% url 'view_trains' %}">View trains</a>
      </li>
      <li class="nav-item {% if url_name == 'security_login' %}active{% endif %}">
        <a class="nav-link" href="{% url 'security_login' %}">Login as Security</a>
      </li>
      <li class="nav-item {% if url_name == 'staff_login' %}active{% endif %}">
        <a class="nav-link" href="{% url 'staff_login' %}">Login as Staff</a>
      </li>
      {% if user.is_authenticated %}
      <li class="nav-item">
        <a class="nav-link" href="{% url 'logout' %}">Logout</a>
      </li>
      {% endif %}
    </ul>
    {% endwith %}
    <!-- <form class="form-inline my-2 my-lg-0">
      <input class="form-control mr-sm-2" placeholder="Search" type="text">
      <button class="btn btn-secondary my-2 my-sm-0" type="submit">Search</button>
    </form> -->
  </div>
</nav>
<div style="margin-left:20px; margin-top:20px">
{% block body %}{% endblock %}
</div>

<footer class="footer" style="margin-left:20px; margin-top:20px">
  <small>
    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>
</small>
</footer>

</body>
</html>
```

5 Results

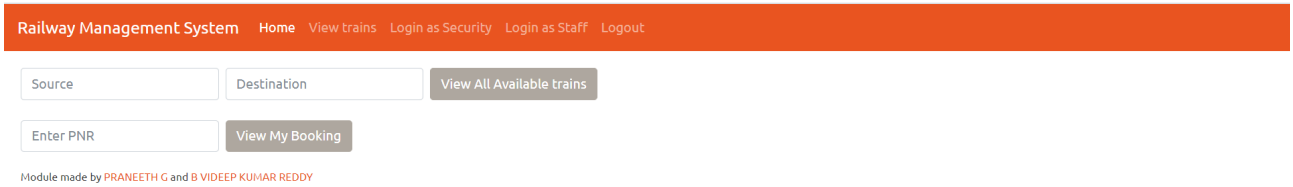

Figure 2: Home Page

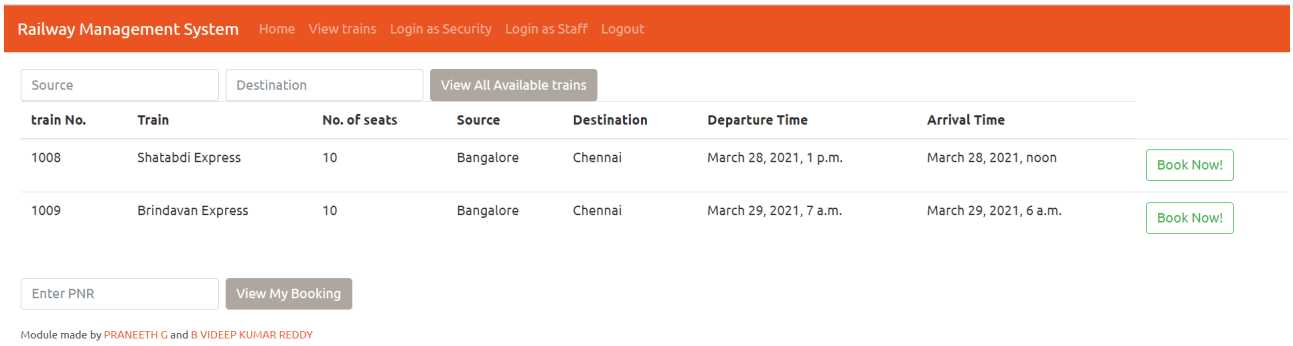

Figure 3: Search Trains

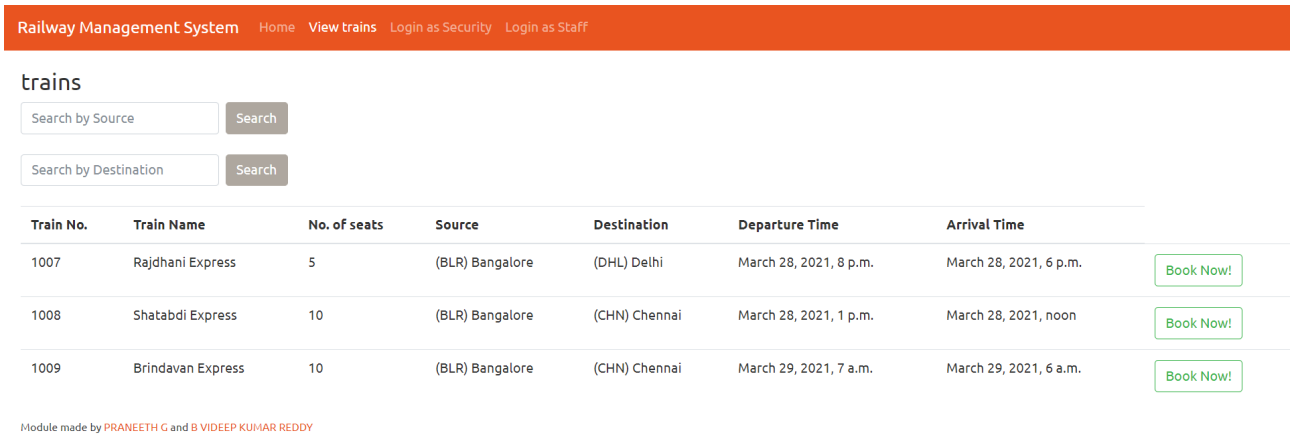

Figure 4: View all Trains

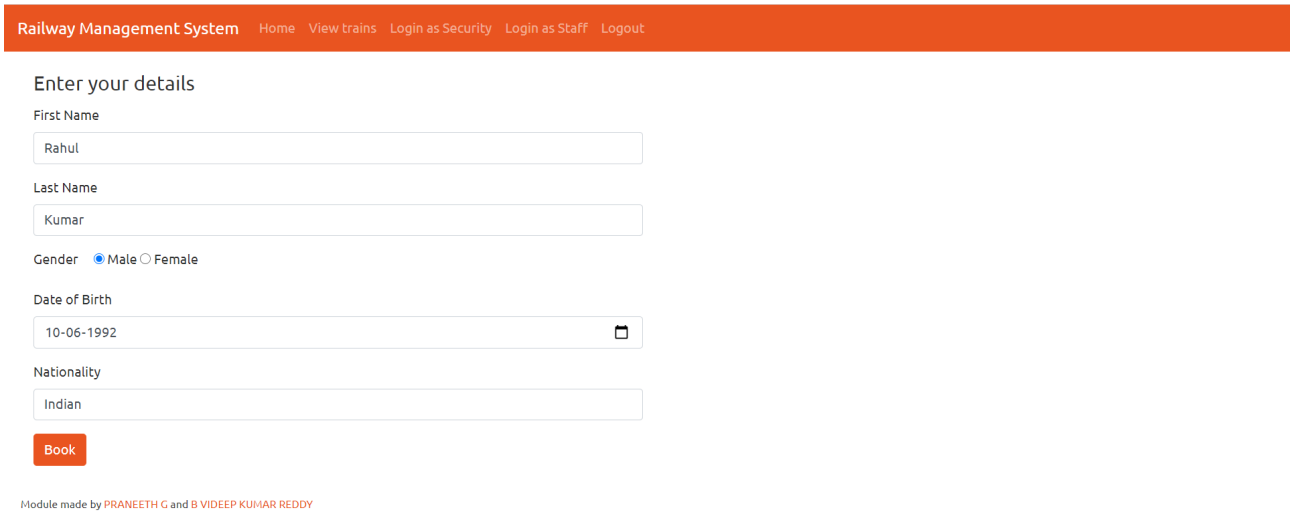

Figure 5: Book Ticket

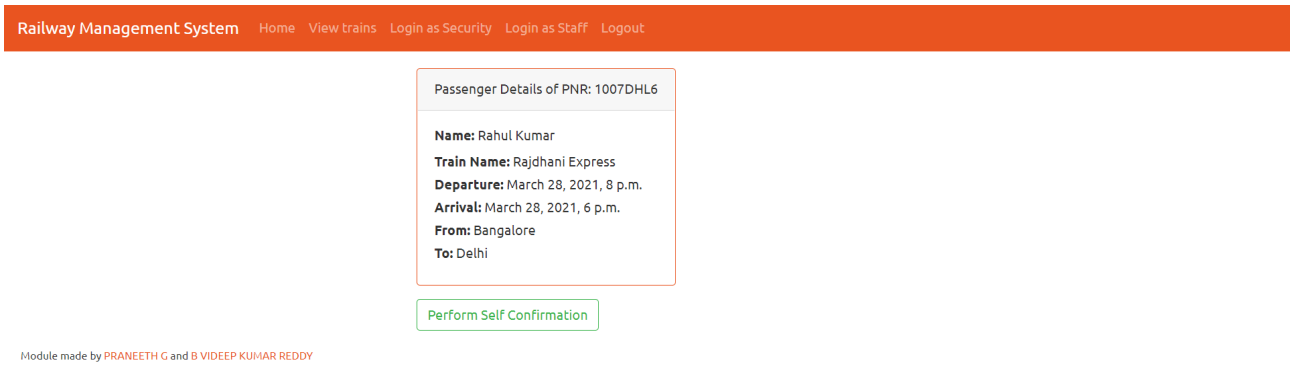

Figure 6: Show Booking

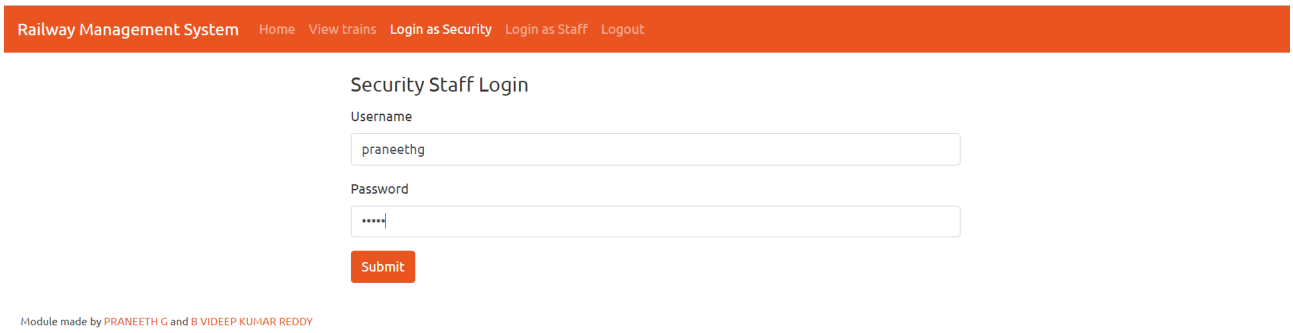

Figure 7: Secourity Login

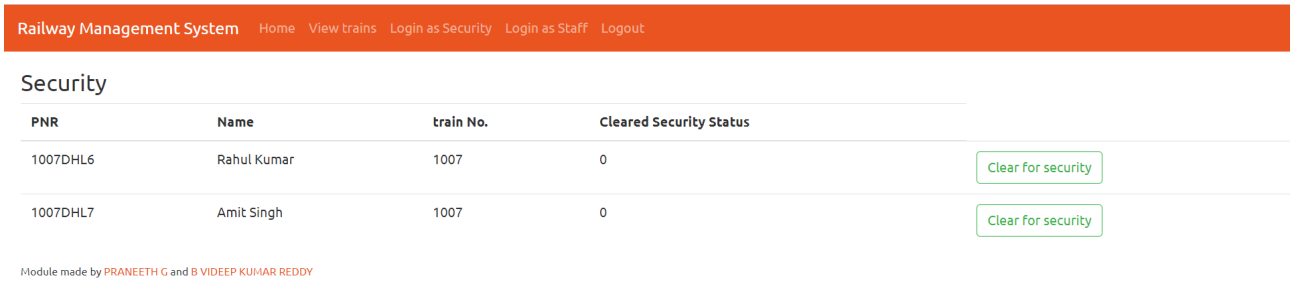

Figure 8: Security Check

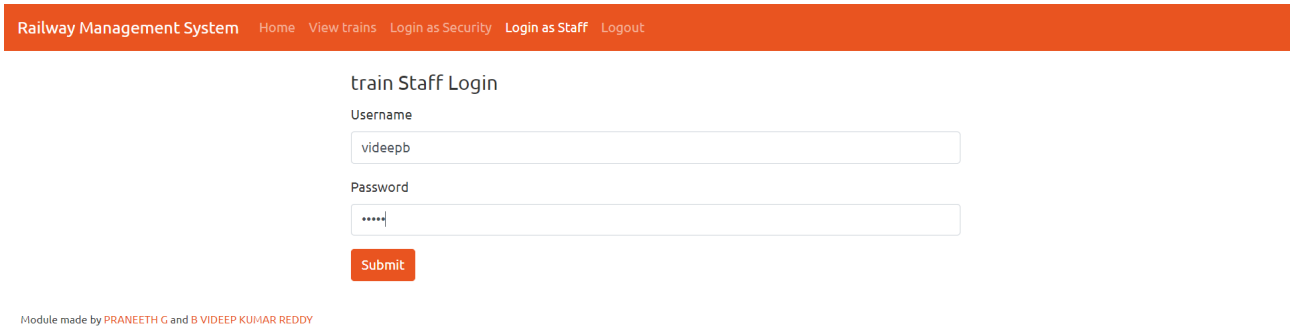

Figure 9: Train Staff Login

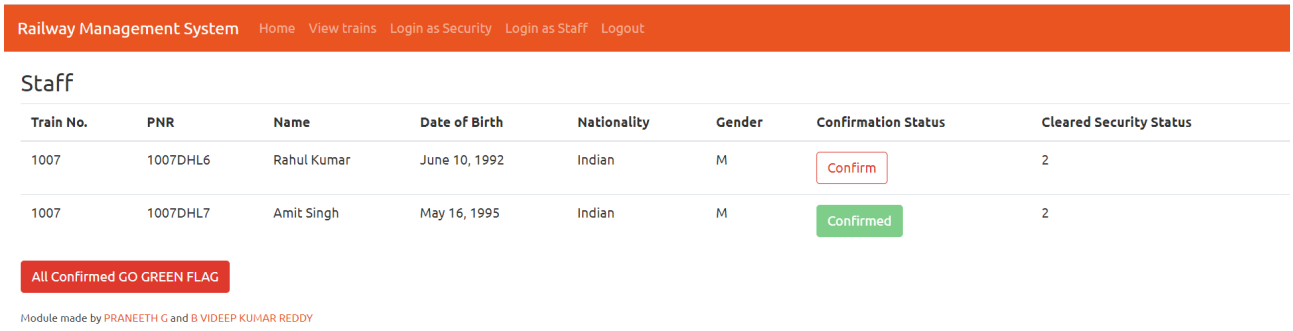

Figure 10: Confirming Tickets

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 ******