Dharmsinh Desai University, Nadiad

Faculty of Technology

Department of Computer Engineering

B.Tech CE Semester - IV

Subject :- Software Project

Project Title :- Flight Booking System

By:

1) Vivek Sonani

(CE126) (19CEUOS048)

2) Aum Thacker

(CE129) (19CEUOS134)

Guided by: Prof. Jigar M. Pandya, Prof. Pinkal C. Chauhan, Prof. Brijesh S. Bhatt

DHARMSINH DESAI UNIVERSITY
NADIAD-387001, GUJARAT

CERTIFICATE

This is to certify that the project entitled as "Flight Booking System" is a bona-fide report of the work carried out by

- **Sonani Vivek Alpeshbhai,** Student Id: **19CEUOS048**
- **Thacker Aum Pareshbhai,** Student Id: **19CEUOS134**

of Department of Computer Engineering, semester **IV**, under the guidance and supervision of **Prof. Pinkal C. Chauhan, Prof. Jigar M. Pandya** and **Prof. Brijesh S. Bhatt** for the subject **Software Project** during the academic year 2020-2021.

Project Guide	Head of the Department
Assistance Professor	Prof. & Head,
Prof. Jigar M. Pandya	**Dr. C.K Bhensdadia**
Department of Computer Engineering,	Department of Computer Engineering,
Faculty of Technology,	Faculty of Technology,
Dharmsinh Desai University,	Dharmsinh Desai University,
Nadiad Nadiad	

Contents:

Introduction:

The purpose of the project is to build an application program to reduce the manual work for managing the Passenger Reservation, Airline Enquiry, Booking Enquiry, Airlines Booking.

When commuters will enter in the website, he/she should have an account. If commuter does not have an account, commuter has to create a new account to book the flight ticket. To create a new account commuter should enter the new id/mobile no. with password.

Once commuter login the system, he/she can see the page in which he may put the information related to book the flight ticket. There is one option to select the source city which may be nearby airport of commuter. Another option is there to select destination city where commuter wants to reach. If commuter press on the search area of source and destination city, commuter can see the possible routes like Bombay to Delhi, Ahmedabad to Kolkata, Chennai to Bangalore and many more. Commuter can see departure date option in which commuter can select date on which commuter wants to go. If commuter wants to book ticket for return flight, also commuter has to select return date otherwise commuter can ignore it. Commuter needs to mention number of Travelers according to category of travelers' age. If all needed information has been entered by commuter, he/she can press on search button so system will provide information regarding flights which are available on the date specified by the commuter.

If flight has been decided by commuter, he/she can proceed to pay. Commuter will have options for payment by credit or by debit and many more methods. When payment method is selected, commuter can enter the bank account details with mobile number. If account details have been entered, commuter can press on the confirm button. Now commuters will be able to view details with unique transaction id related to his/her reservation if payment is done successfully. User may also log out from the system, if user want. Commuter can also cancel the ticket by transaction id which is provided. Commuter will get 80% refund in seven days.

Software Requirement Specification:

1. Account Management

R.1.1: Creation of user profile

Description: - If user have not registered in the system, user will have to create a user profile in the system, details such as full name, mobile number, gender, age, password are entered. This is stored in database and unique id is generated.

Input: - User profile details

Output: - Unique id

R.1.2: User login

Description: - If user wants to use any features of the system, user must have logged in to the system.

Input: Id and Password

Output: User is logged in

Processing: Password validation

R.1.3: View and Update user profile

Description: - The system provides an option to view the user profile. When user profile's information changes that must be updated in the database.

R.1.3.1: Select view profile option

State: The user has logged in and the main menu has been displayed.

Input: "View profile" option selection

Output: User profile

Next function: R.1.3.2 if user is willing to update the profile.

R.1.3.2: Select update profile option

Input: "Update profile" option selection.

Output: Edit profile

R.1.3.3: Change profile

Input: change the details and save option selection.

Output: "Your profile is updated"

R.1.4: Admin login

Description: Admin will be able to login in the system.

Input: Id and Password

Output: Admin is logged in

2. Manage Flights

R.2.1: Search available flight details

Description: Flight details are entered by user and available flight details should be displayed.

Input: Source and destination city, departure and return (optional) date, Number of travelers and “search” option selection

Output: Available flights with details

R.2.2: View and Update flight details

Description: Admin will be able to view, update, insert and delete of flight details.

R.2.2.1: View flight details

Input: Selection

Output: Flight details

R.2.2.2: Change flight details

Input: Selection

Output: Confirmation message

3. Manage Transaction

R.3.1: Making reservation

Description: User should be able to confirm the reservation by making payment.

R.3.1.1: Select the flight

State: List of available flight details has been displayed.

Input: Flight selection

Output: Details about selected flight with grand total and “payment” option selection

Processing: Number of available seats will be deducted for the selected flight

R.3.1.2: Make a payment

Input: Phone number, First name, Middle name, Last name, Credit/debit card number

Output: Payment confirmation message with unique generated reservation id

R.3.1.3: Send notification

State: Ticket has confirmed

Input: Selection

Output: Payment confirmation message

R.3.2: Ticket cancellation

State: The user has logged in and the main menu has been displayed.

Description: Upon cancellation of ticket, partial amount will be refunded and cancelled seat will be available to another users.

Input: Reservation id

Output: Ticket cancellation message

Processing: Cancelled seat will be available to another users

R.3.3: View payment history

State: The user has logged in and the main menu has been displayed.

Description: User will be able to see details of all the payment which user had made.

Input: “View payment details” option selection

Output: Payment details

4. Manage Details

R.4.1: View user details

Description: Admin will be able to see the confirmed ticket of all the user who has booked the flight ticket. Admin is allowed to see the user’s profile who has registered in the system. Admin can also view the banking details of the user who have made the reservation in the system.

R.4.1.1: View user profile

Input: Selection

Output: User profile

R.4.1.2: View all confirmed ticket details

Input: Selection

Output: Ticket details

R.4.1.3: View banking details

Input: Selection

Output: Banking details

R.4.2: View ticket details

State: The user has logged in and the main menu has been displayed.

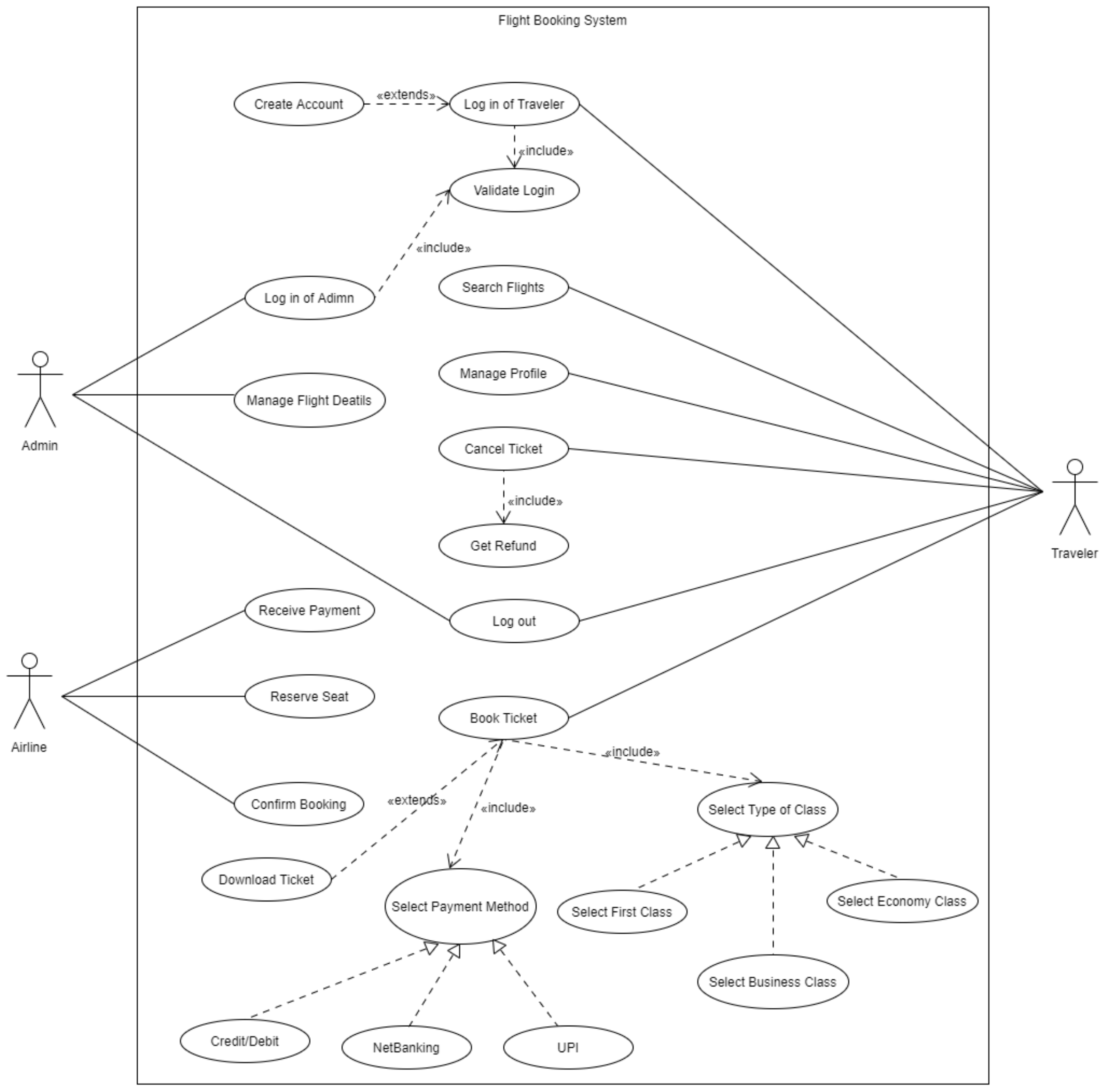
Description: Confirmed ticket details should be displayed.

Input: “View Ticket” option selection

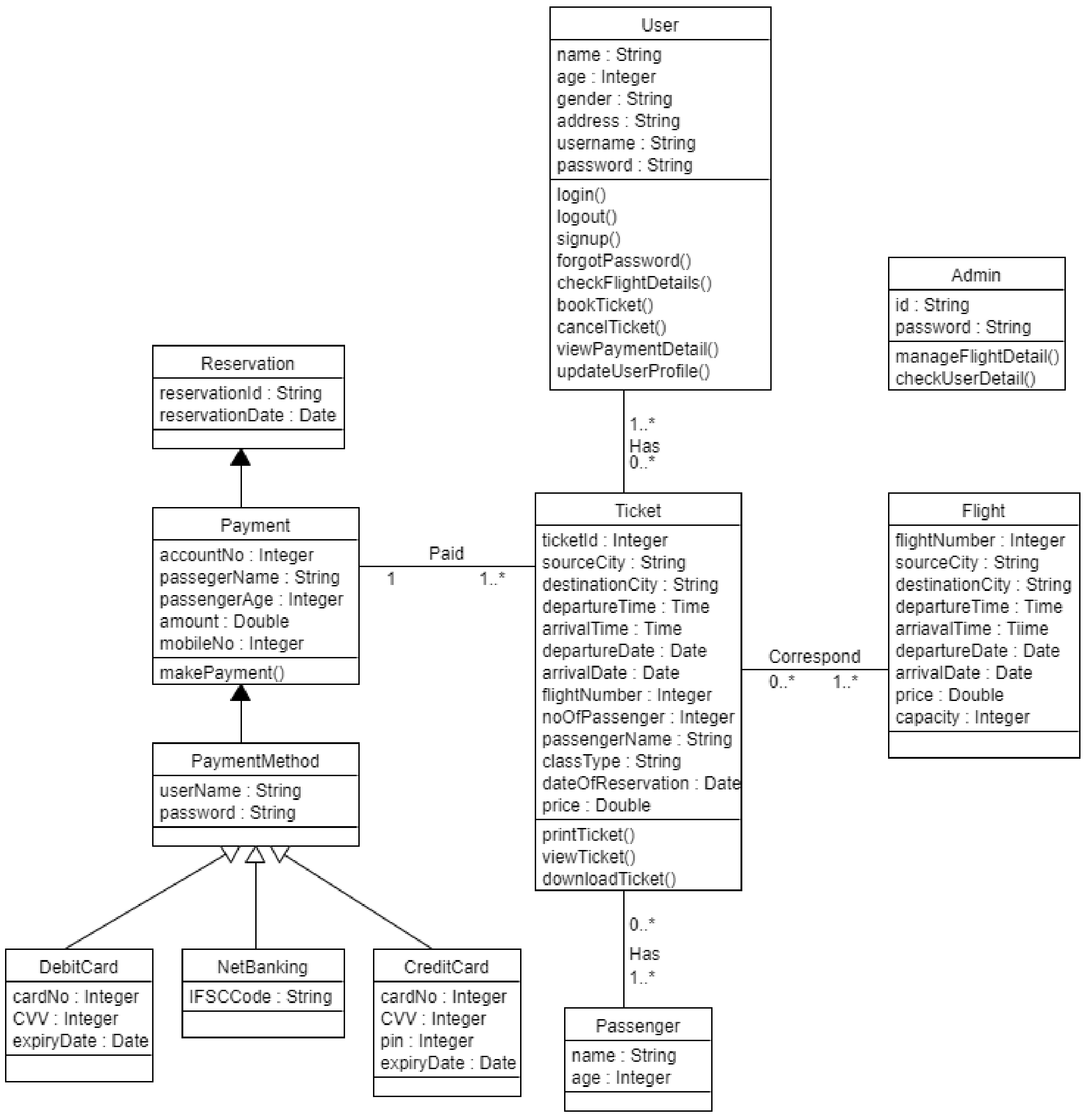
Output: Confirmed ticket details

Design Documents:

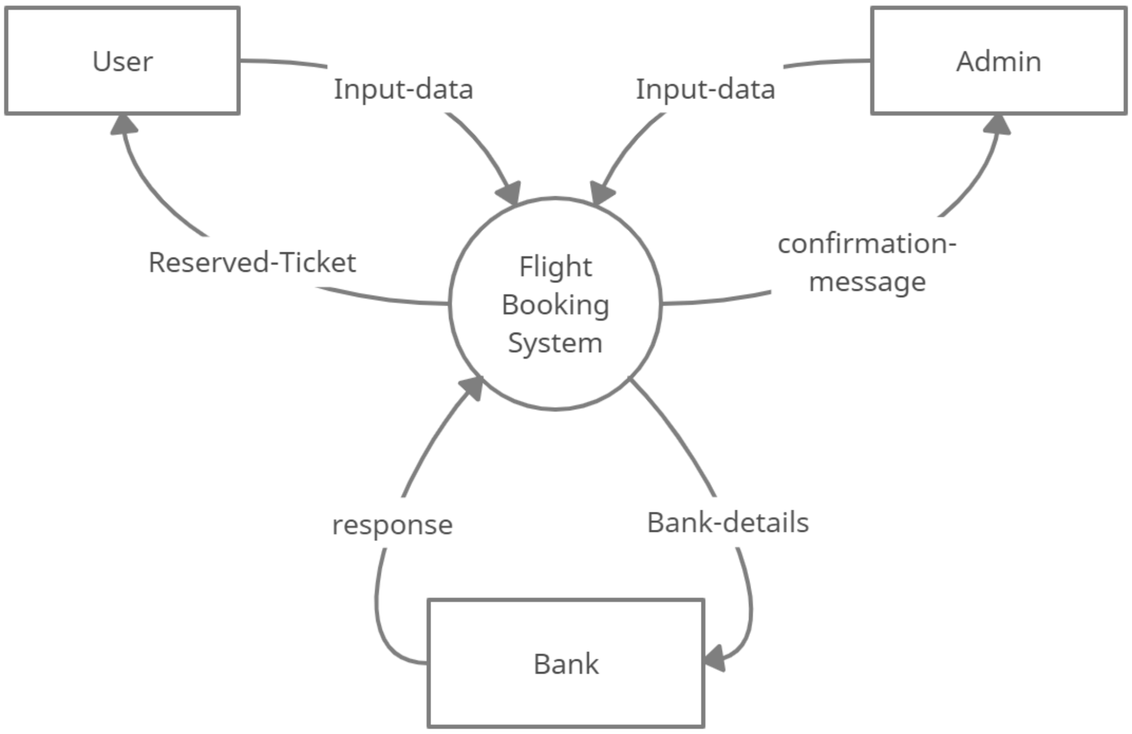
I) Use Case Diagram:



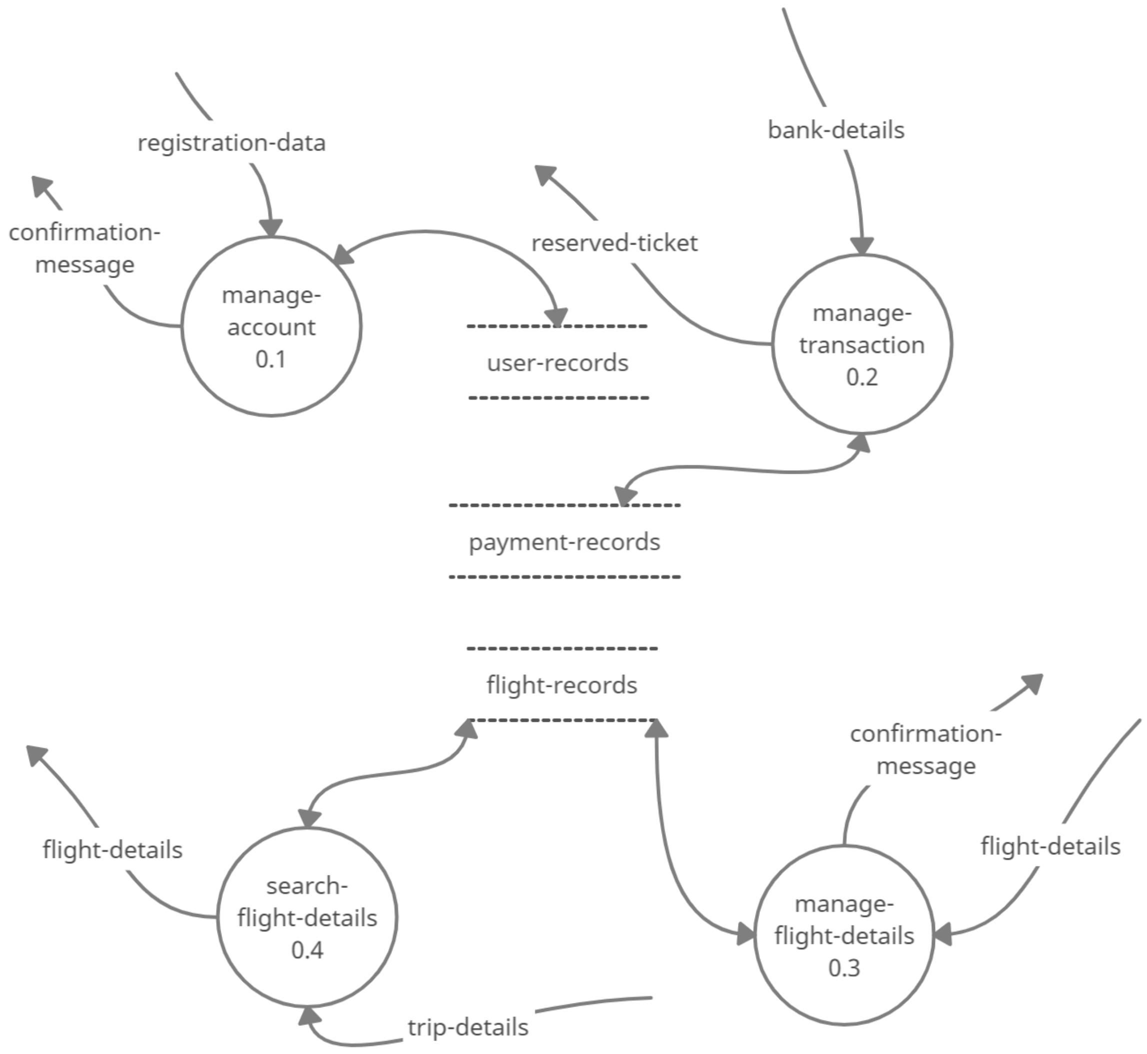
II) Class Diagram:



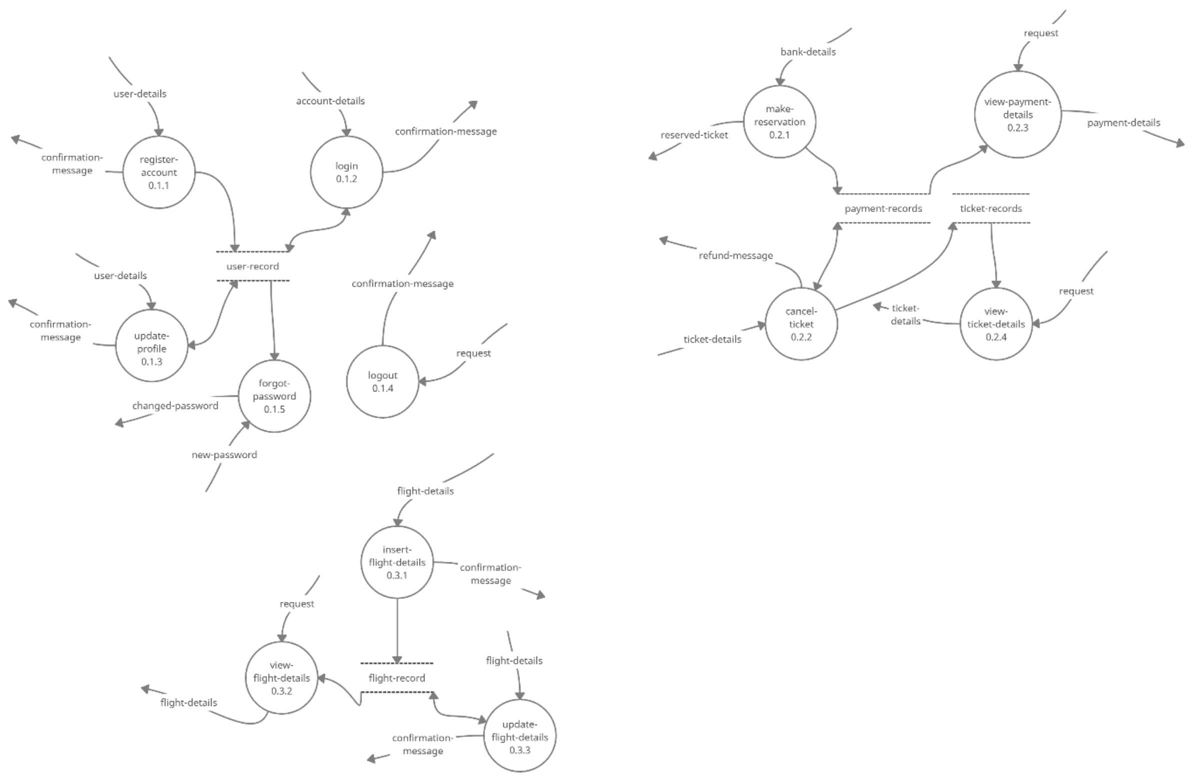
III) Data Flow Diagram:



Level 0 diagram

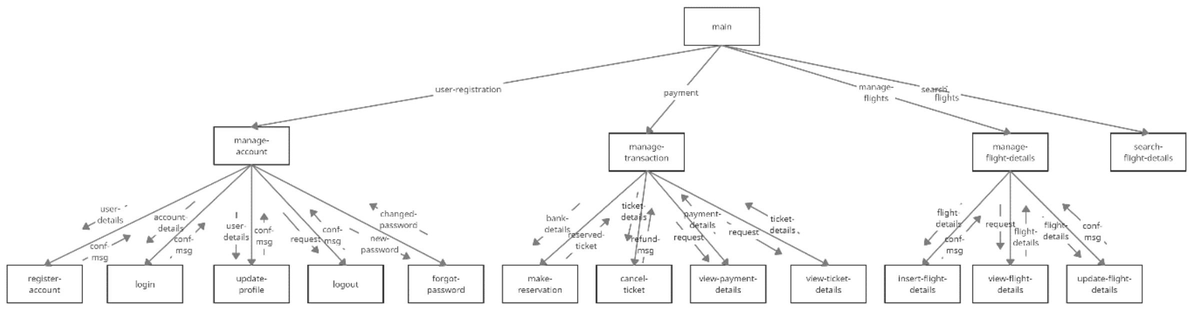


Level 1 diagram



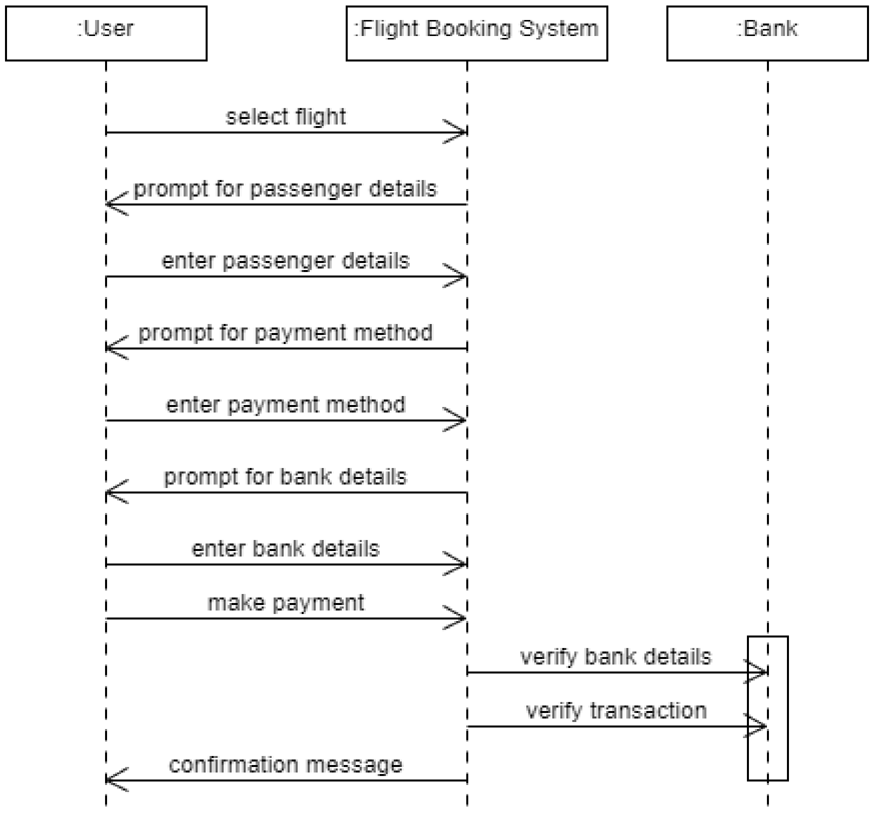
Level 2 diagram

IV) Structure Chart:

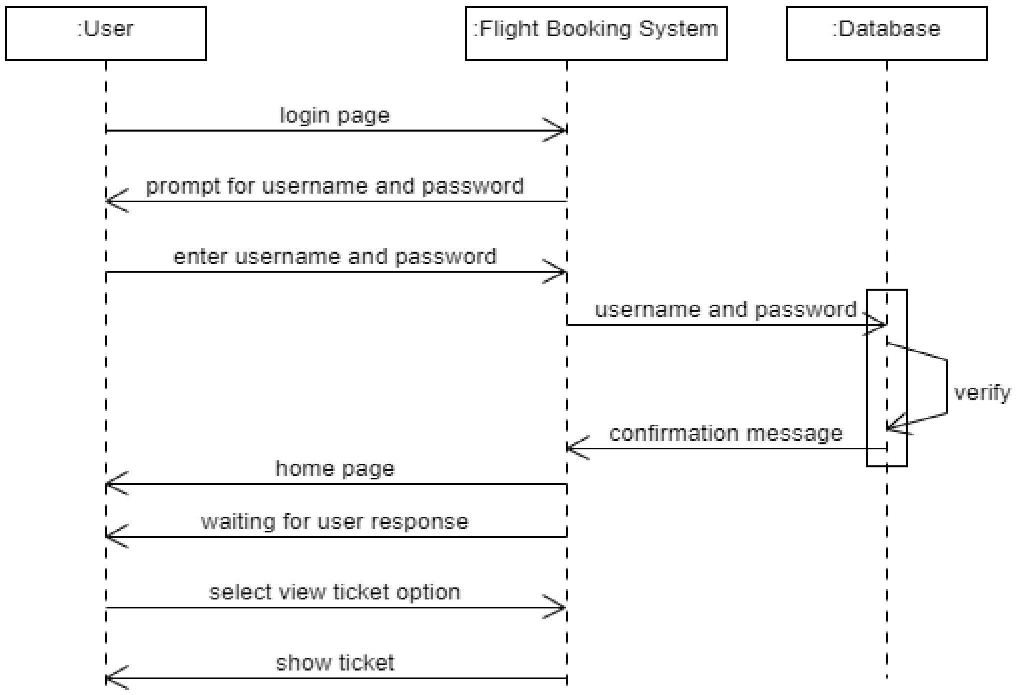


V) Sequence Diagrams:

→Book Ticket

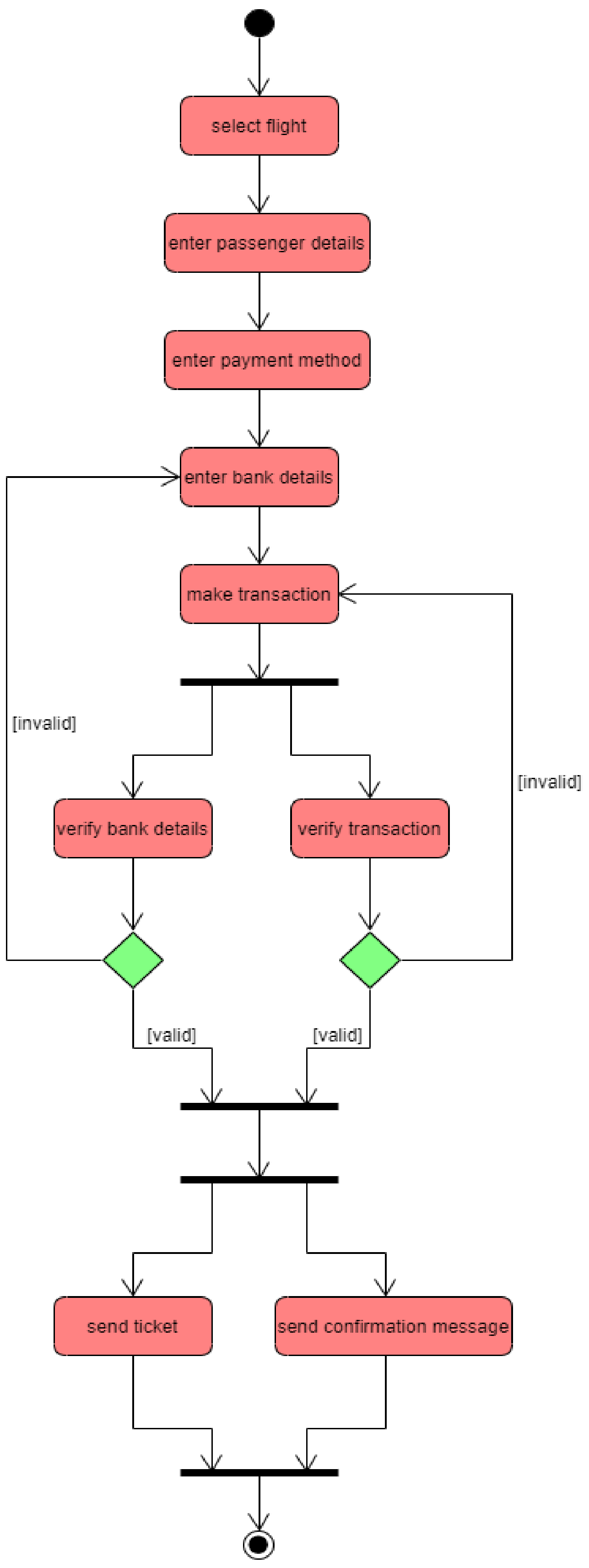


→View Ticket Detail

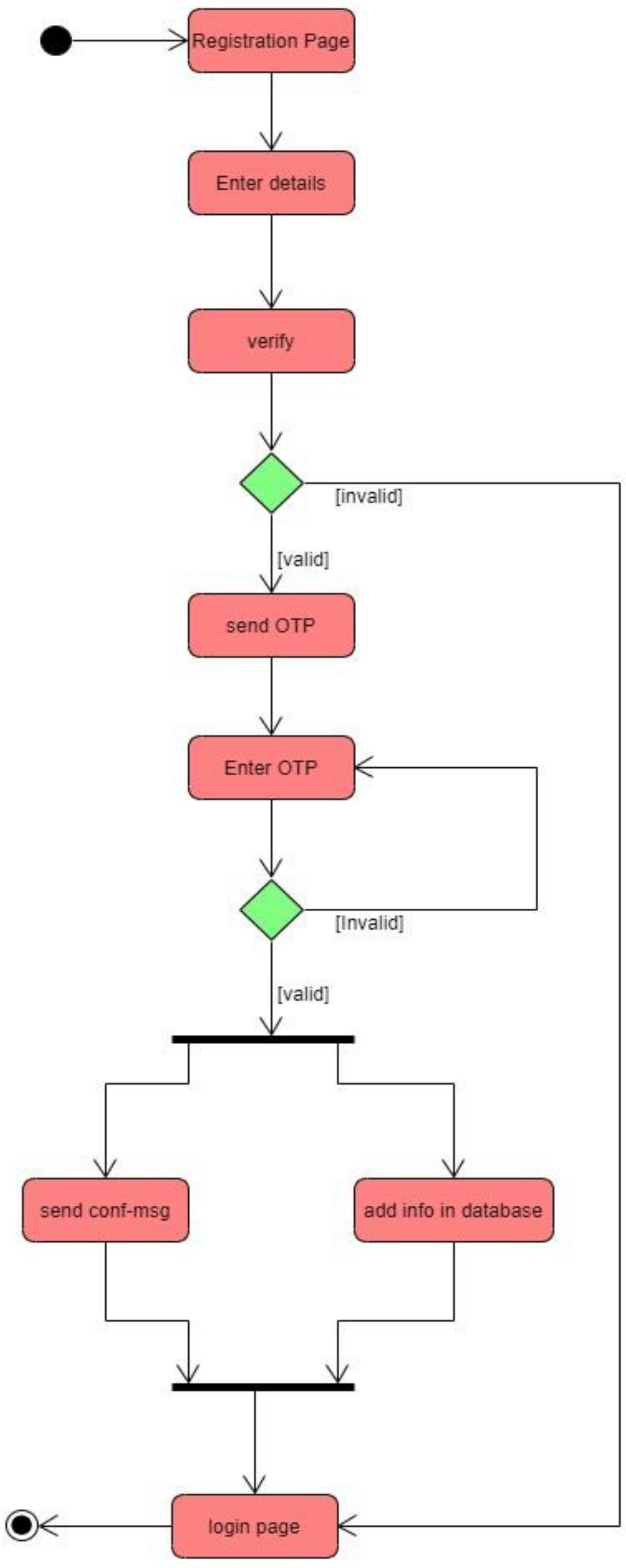


VI) Activity Diagrams:

→Book Ticket



→Register Account



Implements Details:

1. Modules:

In the following section a brief description of each module is given.

Manage Accounts:

This module allows user to register to the system. Admin/User must have to login to the system for using any of the functionality.

Search Flight:

Search Flight is the main module in our system because main functionality is performed here. User can search the flights for either one-way trip or round trip and select the flight.

Manage Payment:

User can book the flight ticket by making payment. This module provides two payment method. User can also see all payment details.

Manage Ticket:

This module allows user to view all ticket details. User can also cancel the previously booked ticket by providing authentication details.

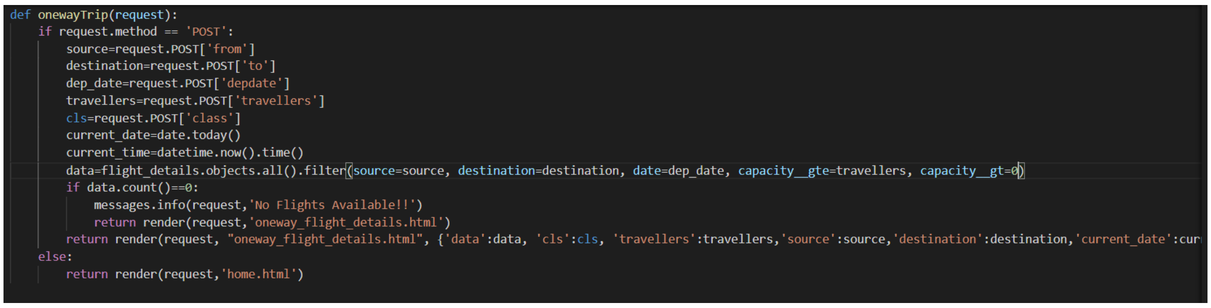
Manage Flight Details:

This module appears in the admin side. Admin have all access to do crud operations on flight details.

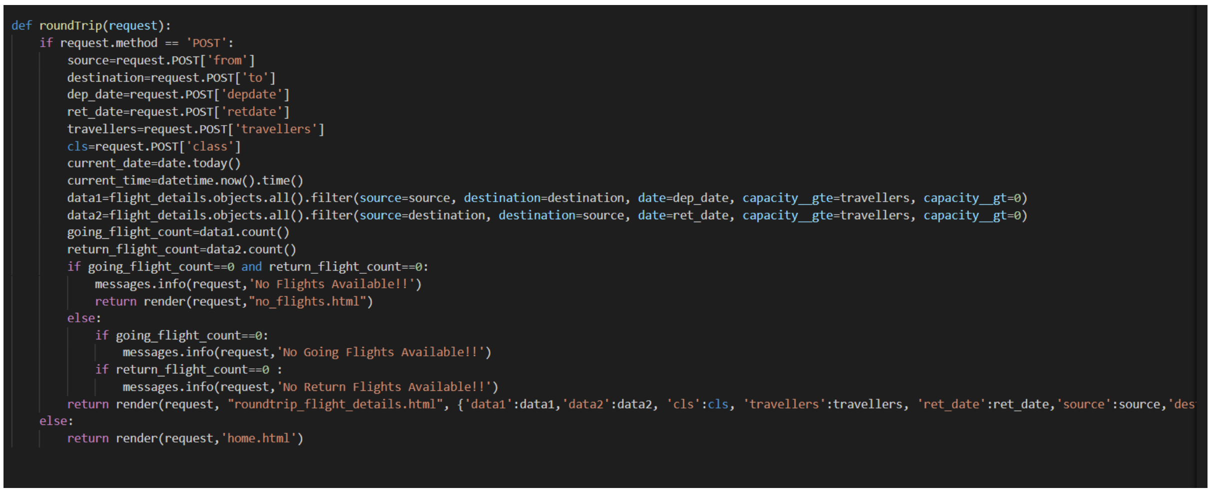
2.Major Functions Prototypes:

I) Search Flights:

User can search the flight details for either one-way trip or round trip by entering flight details.



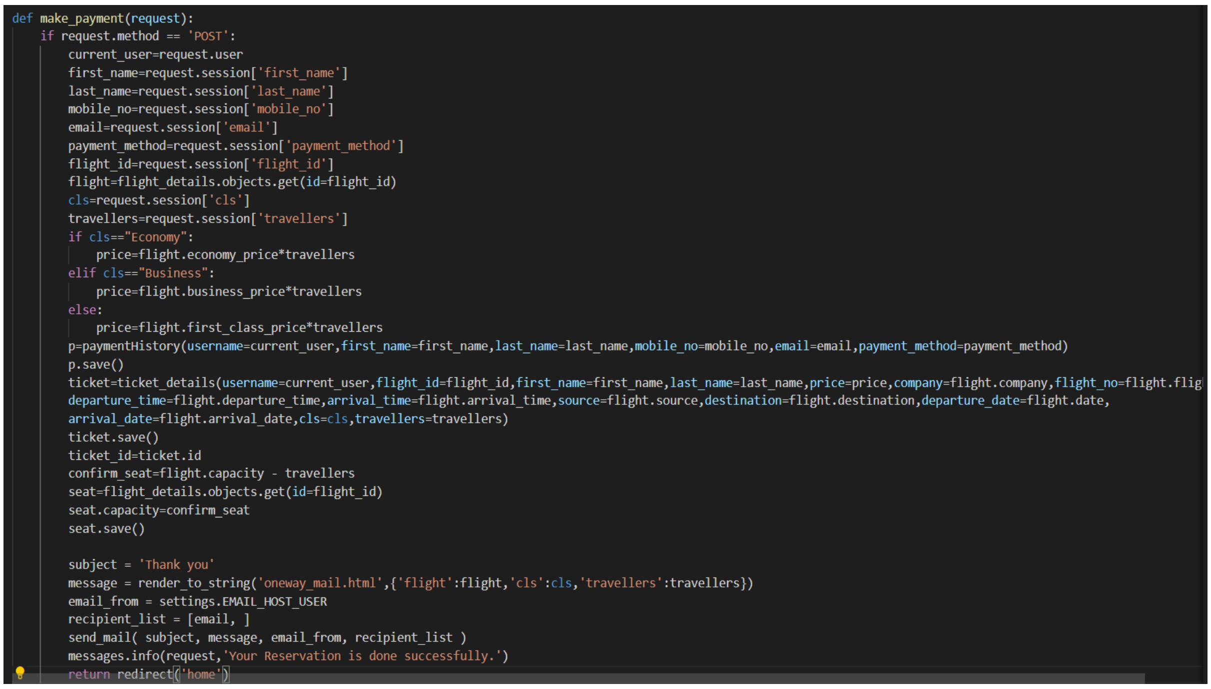
```python
def onewayTrip(request):
    if request.method == 'POST':
        source=request.POST['from']
        destination=request.POST['to']
        dep_date=request.POST['depdate']
        travellers=request.POST['travellers']
        cls=request.POST['class']
        current_date=date.today()
        current_time=datetime.now().time()
        data=flight_details.objects.all().filter(source=source, destination=destination, date=dep_date, capacity__gte=travellers, capacity__gt=0)
        if data.count()==0:
            messages.info(request,'No Flights Available!!')
            return render(request,'oneway_flight_details.html')
        return render(request, "oneway_flight_details.html", {'data':data, 'cls':cls, 'travellers':travellers,'source':source,'destination':destination,'current_date':cur
    else:
        return render(request,'home.html')
```



```python
def roundTrip(request):
    if request.method == 'POST':
        source=request.POST['from']
        destination=request.POST['to']
        dep_date=request.POST['depdate']
        ret_date=request.POST['retdate']
        travellers=request.POST['travellers']
        cls=request.POST['class']
        current_date=date.today()
        current_time=datetime.now().time()
        data1=flight_details.objects.all().filter(source=source, destination=destination, date=dep_date, capacity__gte=travellers, capacity__gt=0)
        data2=flight_details.objects.all().filter(source=destination, destination=source, date=ret_date, capacity__gte=travellers, capacity__gt=0)
        going_flight_count=data1.count()
        return_flight_count=data2.count()
        if going_flight_count==0 and return_flight_count==0:
            messages.info(request,'No Flights Available!!')
            return render(request,"no_flights.html")
        else:
            if going_flight_count==0:
                messages.info(request,'No Going Flights Available!!')
            if return_flight_count==0 :
                messages.info(request,'No Return Flights Available!!')
        return render(request, "roundtrip_flight_details.html", {'data1':data1,'data2':data2, 'cls':cls, 'travellers':travellers, 'ret_date':ret_date,'source':source,'des
    else:
        return render(request,'home.html')
```

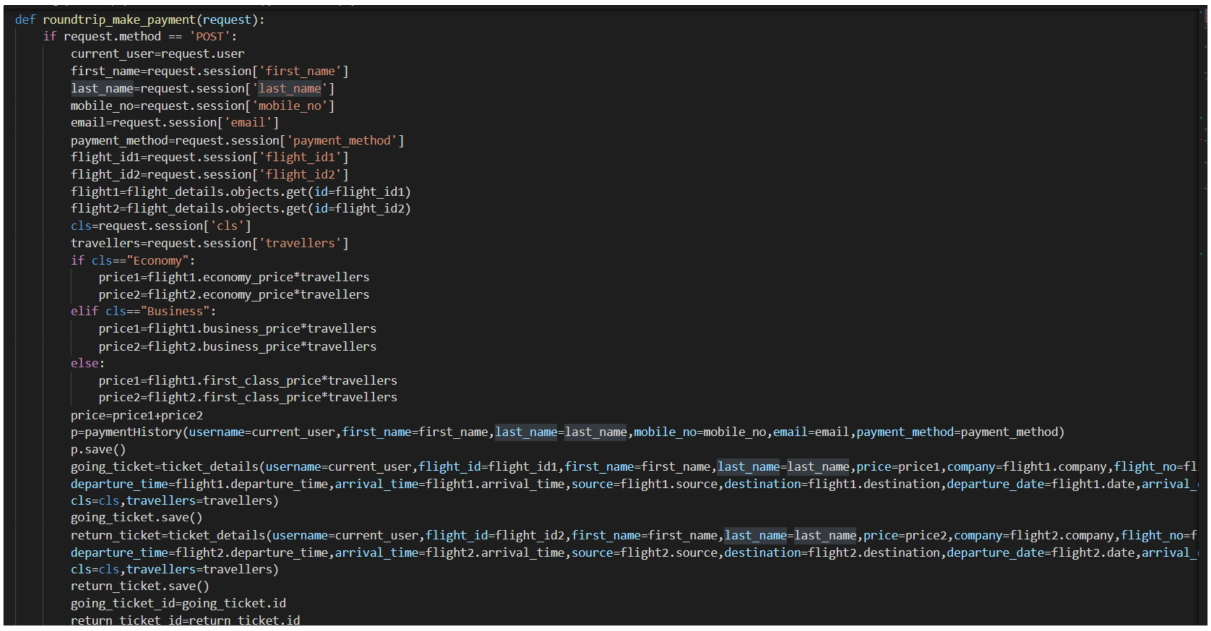
II) Make Payment:

User can book ticket by making payment for either one-way trip or round trip and will get the email for the confirmation of ticket.

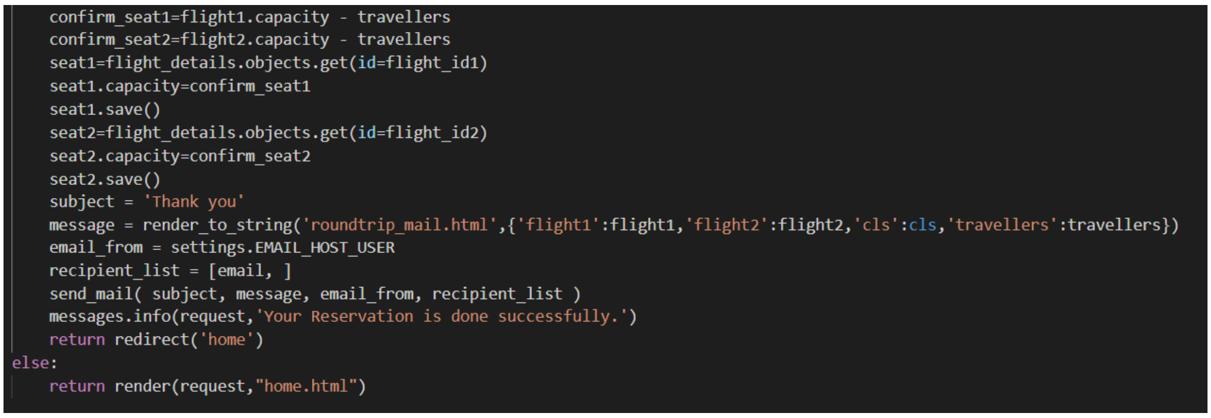


```python
def make_payment(request):
    if request.method == 'POST':
        current_user=request.user
        first_name=request.session['first_name']
        last_name=request.session['last_name']
        mobile_no=request.session['mobile_no']
        email=request.session['email']
        payment_method=request.session['payment_method']
        flight_id=request.session['flight_id']
        flight=flight_details.objects.get(id=flight_id)
        cls=request.session['cls']
        travellers=request.session['travellers']
        if cls=="Economy":
            price=flight.economy_price*travellers
        elif cls=="Business":
            price=flight.business_price*travellers
        else:
            price=flight.first_class_price*travellers
        p=paymentHistory(username=current_user,first_name=first_name,last_name=last_name,mobile_no=mobile_no,email=email,payment_method=payment_method)
        p.save()
        ticket=ticket_details(username=current_user,flight_id=flight_id,first_name=first_name,last_name=last_name,price=price,company=flight.company,flight_no=flight.flig
        departure_time=flight.departure_time,arrival_time=flight.arrival_time,source=flight.source,destination=flight.destination,departure_date=flight.date,
        arrival_date=flight.arrival_date,cls=cls,travellers=travellers)
        ticket.save()
        ticket_id=ticket.id
        confirm_seat=flight.capacity - travellers
        seat=flight_details.objects.get(id=flight_id)
        seat.capacity=confirm_seat
        seat.save()

        subject = 'Thank you'
        message = render_to_string('oneway_mail.html',{'flight':flight,'cls':cls,'travellers':travellers})
        email_from = settings.EMAIL_HOST_USER
        recipient_list = [email, ]
        send_mail( subject, message, email_from, recipient_list )
        messages.info(request,'Your Reservation is done successfully.')
        return redirect('home')
```



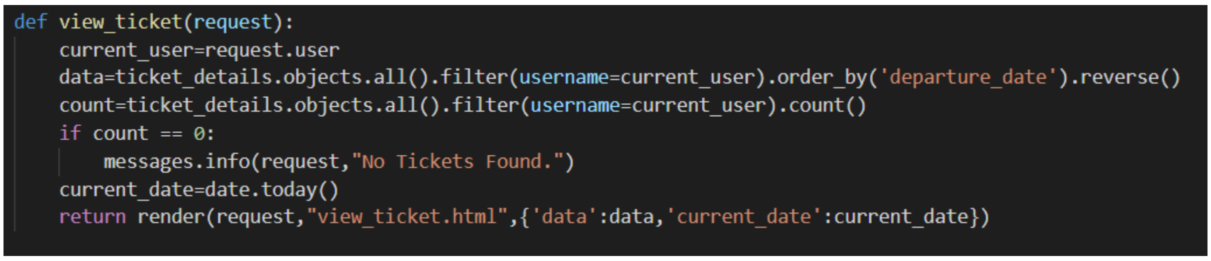
```python
def roundtrip_make_payment(request):
    if request.method == 'POST':
        current_user=request.user
        first_name=request.session['first_name']
        last_name=request.session['last_name']
        mobile_no=request.session['mobile_no']
        email=request.session['email']
        payment_method=request.session['payment_method']
        flight_id1=request.session['flight_id1']
        flight_id2=request.session['flight_id2']
        flight1=flight_details.objects.get(id=flight_id1)
        flight2=flight_details.objects.get(id=flight_id2)
        cls=request.session['cls']
        travellers=request.session['travellers']
        if cls=="Economy":
            price1=flight1.economy_price*travellers
            price2=flight2.economy_price*travellers
        elif cls=="Business":
            price1=flight1.business_price*travellers
            price2=flight2.business_price*travellers
        else:
            price1=flight1.first_class_price*travellers
            price2=flight2.first_class_price*travellers
        price=price1+price2
        p=paymentHistory(username=current_user,first_name=first_name,last_name=last_name,mobile_no=mobile_no,email=email,payment_method=payment_method)
        p.save()
        going_ticket=ticket_details(username=current_user,flight_id=flight_id1,first_name=first_name,last_name=last_name,price=price1,company=flight1.company,flight_no=fl
        departure_time=flight1.departure_time,arrival_time=flight1.arrival_time,source=flight1.source,destination=flight1.destination,departure_date=flight1.date,arrival_
        cls=cls,travellers=travellers)
        going_ticket.save()
        return_ticket=ticket_details(username=current_user,flight_id=flight_id2,first_name=first_name,last_name=last_name,price=price2,company=flight2.company,flight_no=f
        departure_time=flight2.departure_time,arrival_time=flight2.arrival_time,source=flight2.source,destination=flight2.destination,departure_date=flight2.date,arrival_
        cls=cls,travellers=travellers)
        return_ticket.save()
        going_ticket_id=going_ticket.id
        return_ticket_id=return_ticket.id
```



```python
    confirm_seat1=flight1.capacity - travellers
    confirm_seat2=flight2.capacity - travellers
    seat1=flight_details.objects.get(id=flight_id1)
    seat1.capacity=confirm_seat1
    seat1.save()
    seat2=flight_details.objects.get(id=flight_id2)
    seat2.capacity=confirm_seat2
    seat2.save()
    subject = 'Thank you'
    message = render_to_string('roundtrip_mail.html',{'flight1':flight1,'flight2':flight2,'cls':cls,'travellers':travellers})
    email_from = settings.EMAIL_HOST_USER
    recipient_list = [email, ]
    send_mail( subject, message, email_from, recipient_list )
    messages.info(request,'Your Reservation is done successfully.')
    return redirect('home')
else:
    return render(request,"home.html")
```

III) View Ticket:

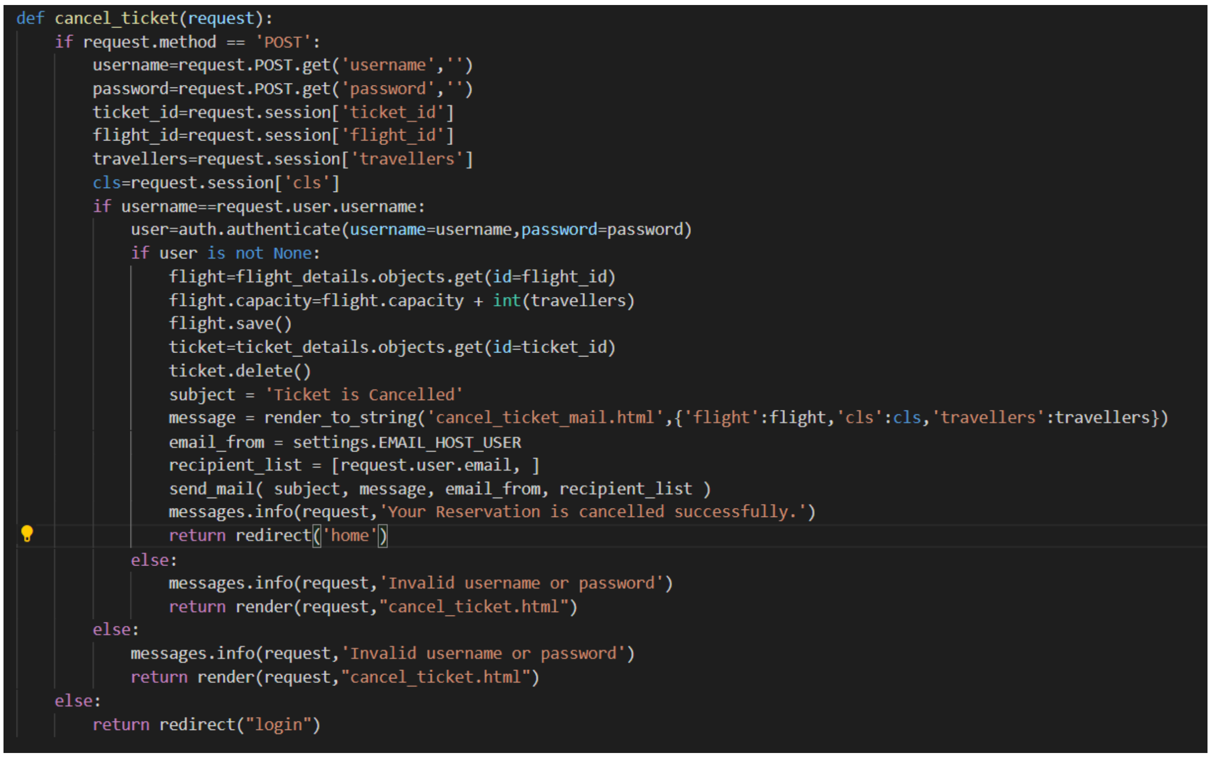
User can see all booked tickets.



```python
def view_ticket(request):
    current_user=request.user
    data=ticket_details.objects.all().filter(username=current_user).order_by('departure_date').reverse()
    count=ticket_details.objects.all().filter(username=current_user).count()
    if count == 0:
        messages.info(request,"No Tickets Found.")
    current_date=date.today()
    return render(request,"view_ticket.html",{'data':data,'current_date':current_date})
```

IV) Cancel Ticket:

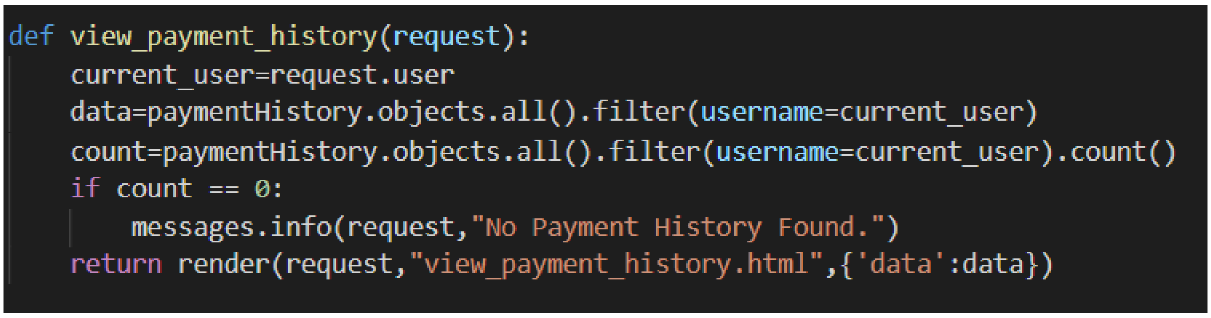
User can cancel the ticket by providing authentication details.



```
def cancel_ticket(request):
    if request.method == 'POST':
        username=request.POST.get('username','')
        password=request.POST.get('password','')
        ticket_id=request.session['ticket_id']
        flight_id=request.session['flight_id']
        travellers=request.session['travellers']
        cls=request.session['cls']
        if username==request.user.username:
            user=auth.authenticate(username=username,password=password)
            if user is not None:
                flight=flight_details.objects.get(id=flight_id)
                flight.capacity=flight.capacity + int(travellers)
                flight.save()
                ticket=ticket_details.objects.get(id=ticket_id)
                ticket.delete()
                subject = 'Ticket is Cancelled'
                message = render_to_string('cancel_ticket_mail.html',{'flight':flight,'cls':cls,'travellers':travellers})
                email_from = settings.EMAIL_HOST_USER
                recipient_list = [request.user.email, ]
                send_mail( subject, message, email_from, recipient_list )
                messages.info(request,'Your Reservation is cancelled successfully.')
                return redirect('home')
            else:
                messages.info(request,'Invalid username or password')
                return render(request,"cancel_ticket.html")
        else:
            messages.info(request,'Invalid username or password')
            return render(request,"cancel_ticket.html")
    else:
        return redirect("login")
```

V) View Payment History:

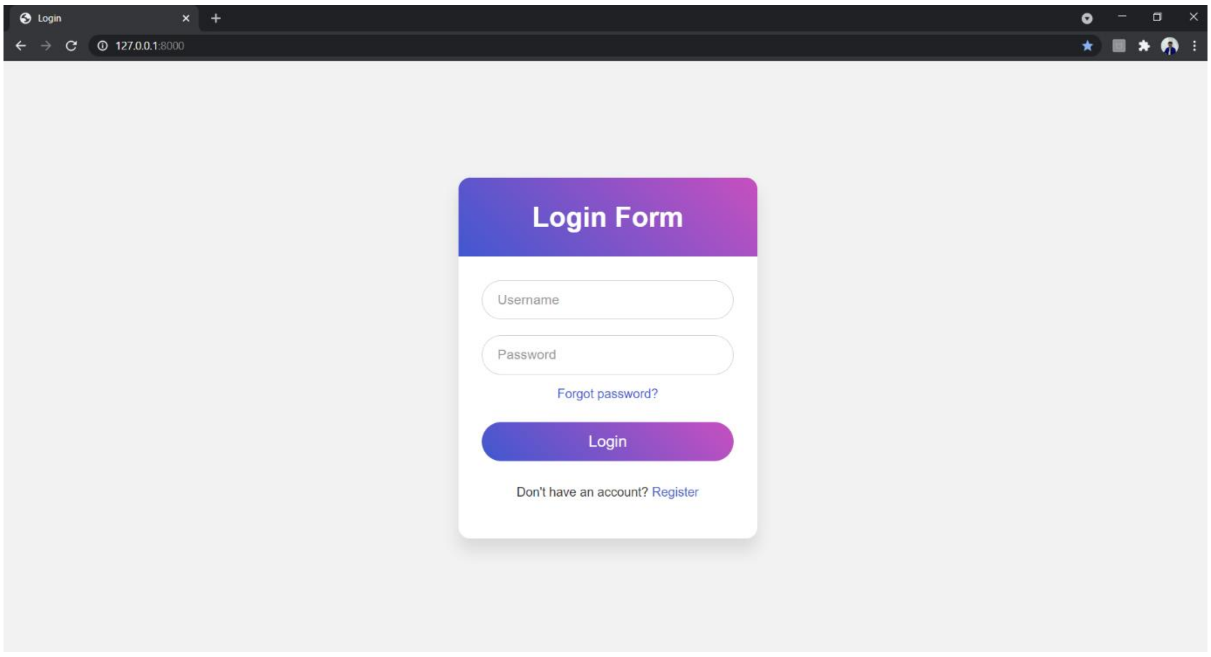
User can see payment history.

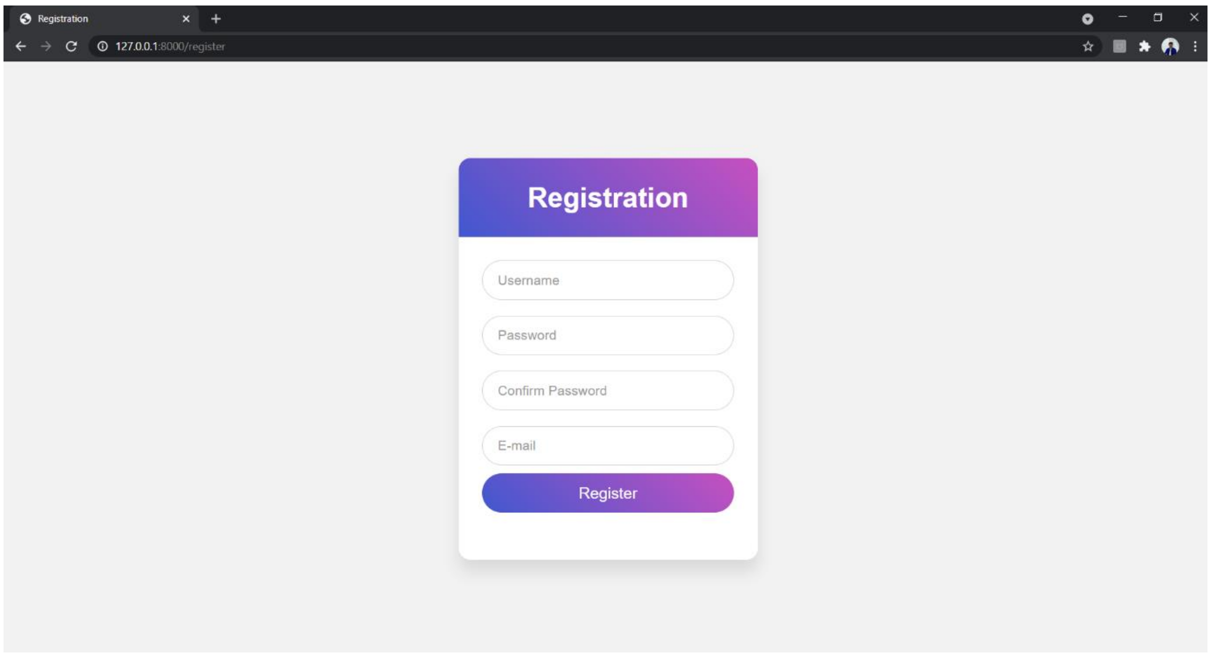


```
def view_payment_history(request):
    current_user=request.user
    data=paymentHistory.objects.all().filter(username=current_user)
    count=paymentHistory.objects.all().filter(username=current_user).count()
    if count == 0:
        messages.info(request,"No Payment History Found.")
    return render(request,"view_payment_history.html",{'data':data})
```

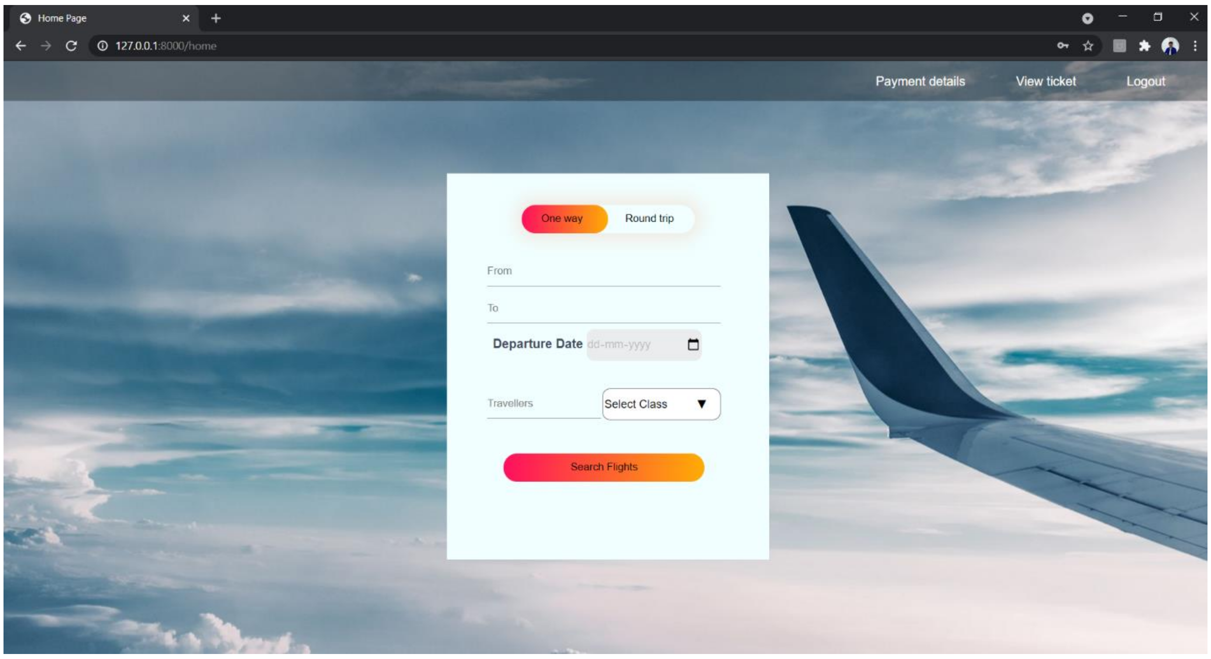
Work Flow/Layouts:

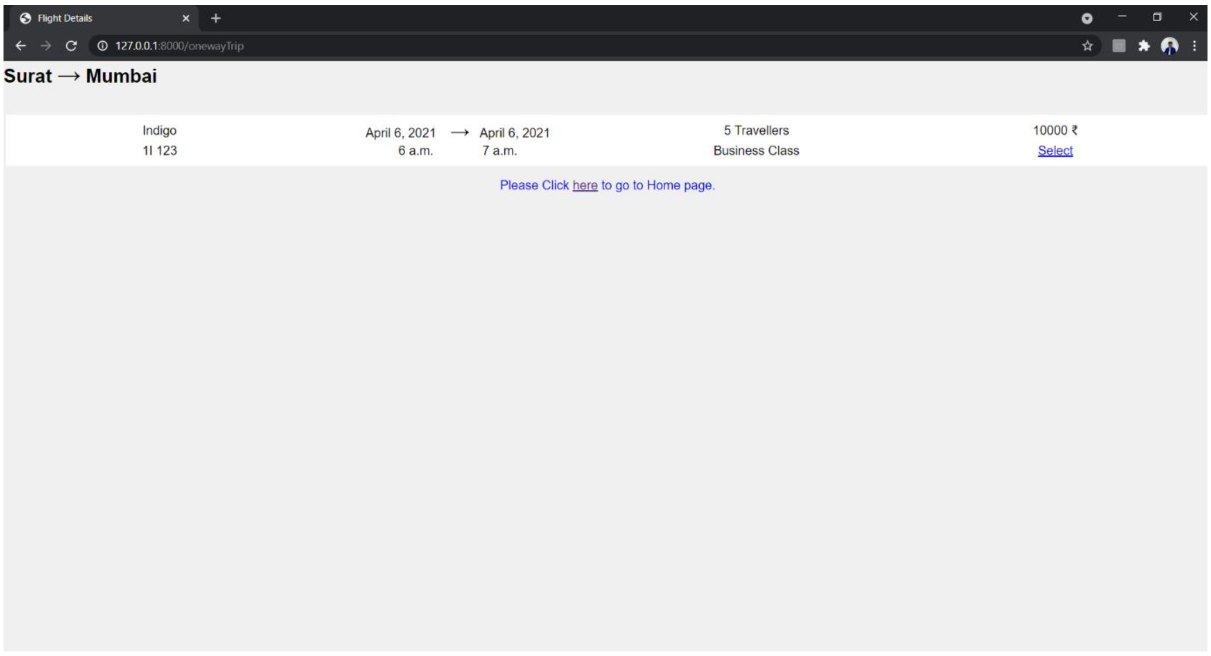
Manage Accounts:





Search Flight:





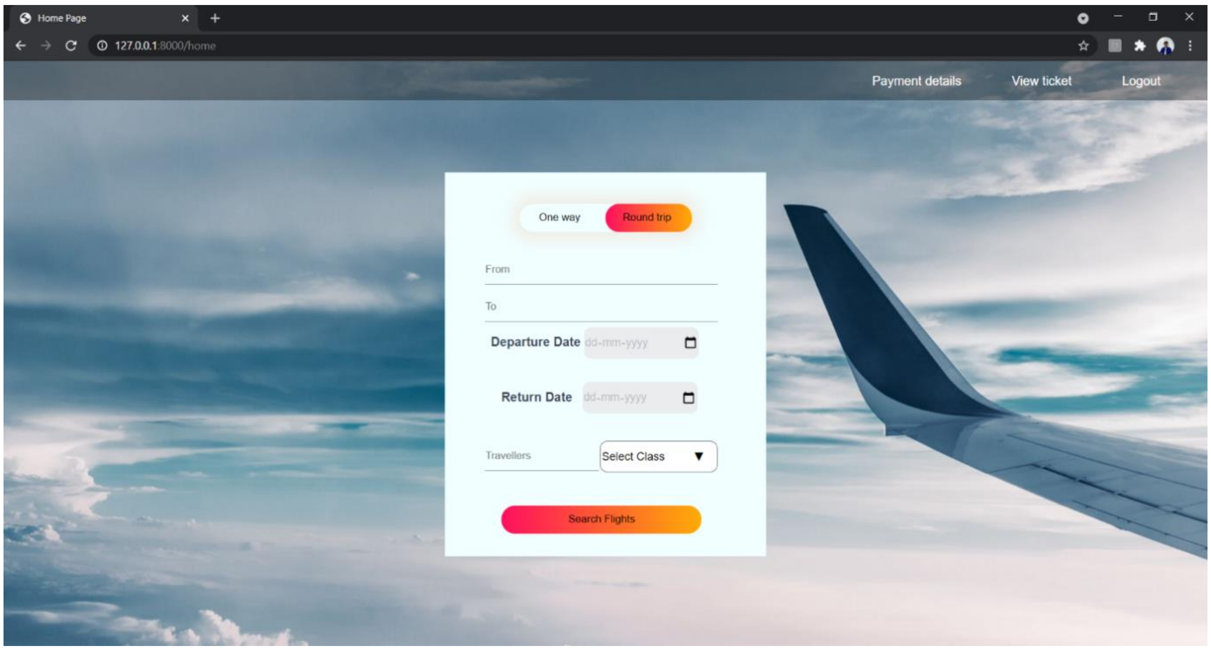
Flight Details

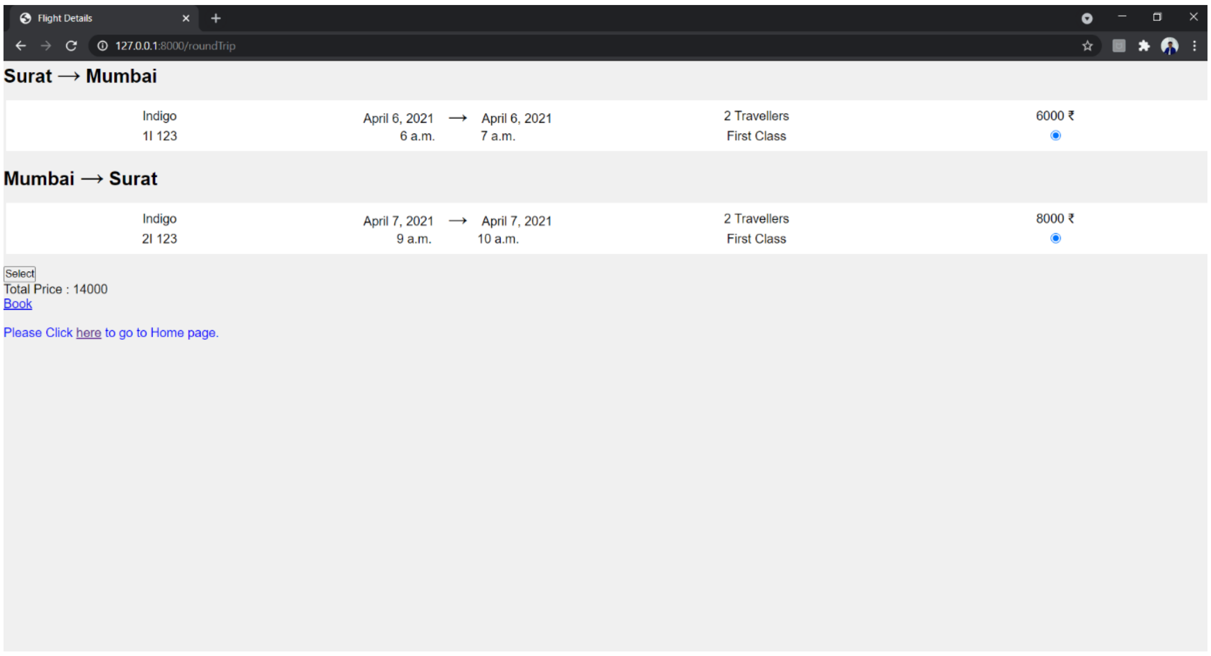
127.0.0.1:8000/onewayTrip

Surat ⟶ Mumbai

Indigo 1I 123	April 6, 2021 ⟶ April 6, 2021 6 a.m. 7 a.m.	5 Travellers Business Class	10000 ₹ Select

Please Click here to go to Home page.





Surat ⟶ Mumbai

Indigo 1I 123	April 6, 2021 6 a.m. ⟶ April 6, 2021 7 a.m.	2 Travellers First Class	6000 ₹ ●

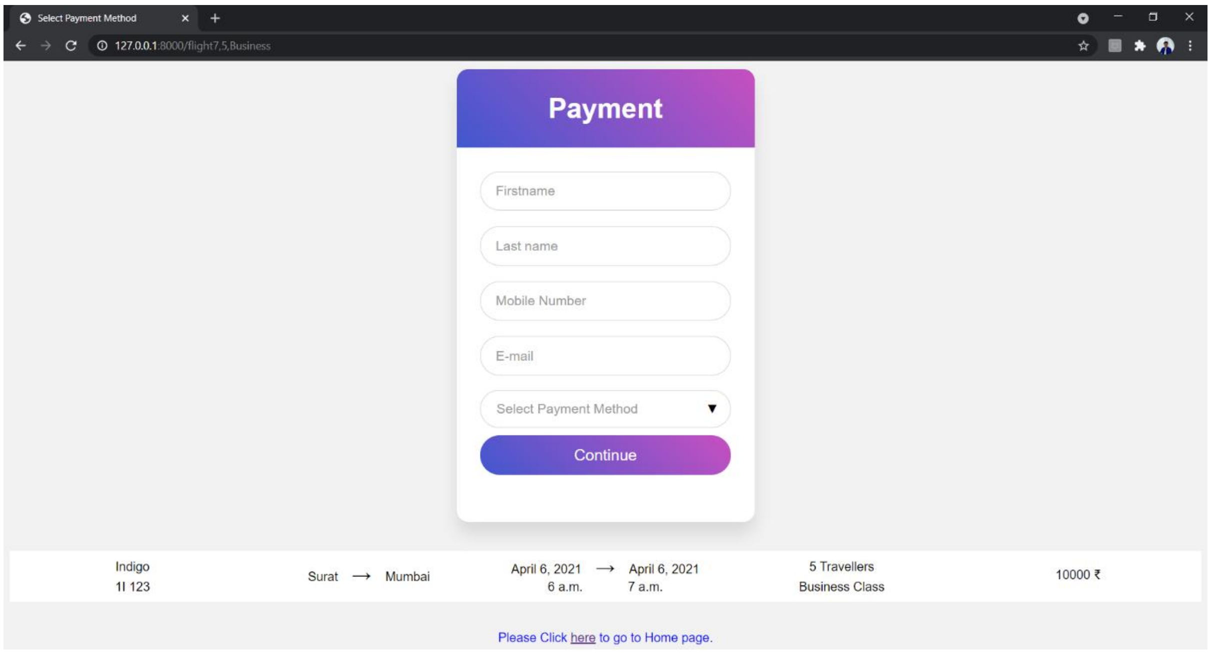
Mumbai ⟶ Surat

Indigo 2I 123	April 7, 2021 9 a.m. ⟶ April 7, 2021 10 a.m.	2 Travellers First Class	8000 ₹ ●

Select
Total Price : 14000
Book

Please Click here to go to Home page.

Manage Payment:


Select Payment Method

127.0.0.1:8000/flight7,5,Business

Payment

Firstname

Last name

Mobile Number

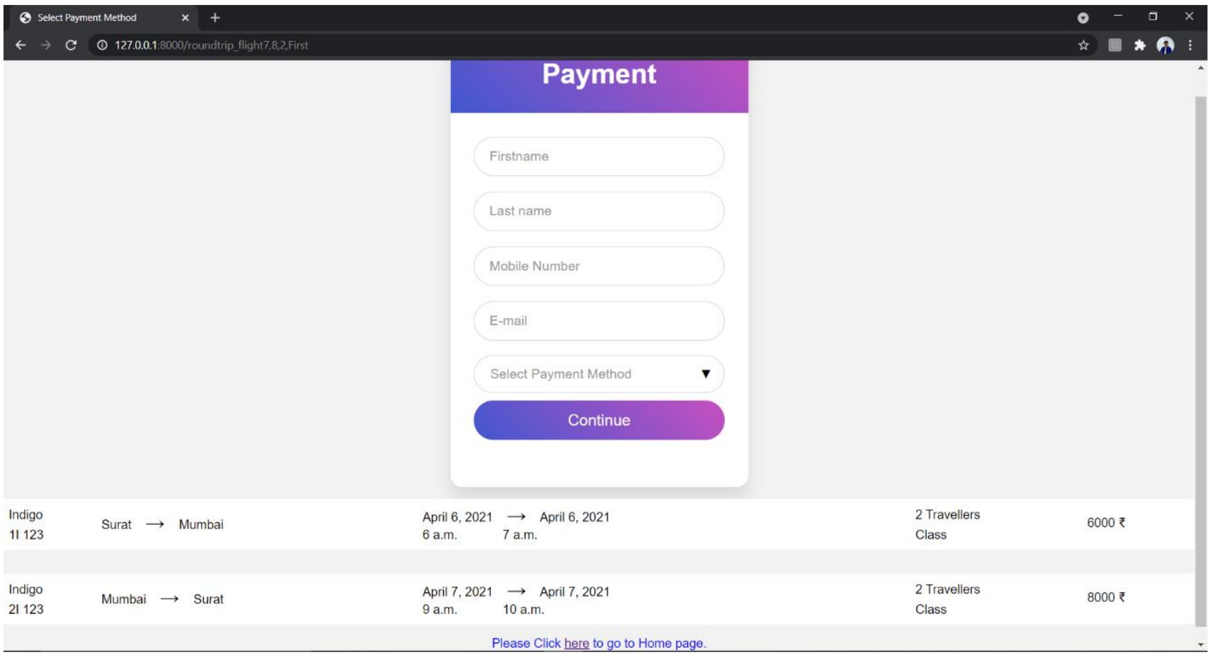
E-mail

Select Payment Method

Continue

Indigo 1I 123	Surat ⟶ Mumbai	April 6, 2021 ⟶ April 6, 2021 6 a.m. 7 a.m.	5 Travellers Business Class	10000 ₹

Please Click here to go to Home page.


Select Payment Method

127.0.0.1:8000/roundtrip_flight7,8,2,First

Payment

Firstname

Last name

Mobile Number

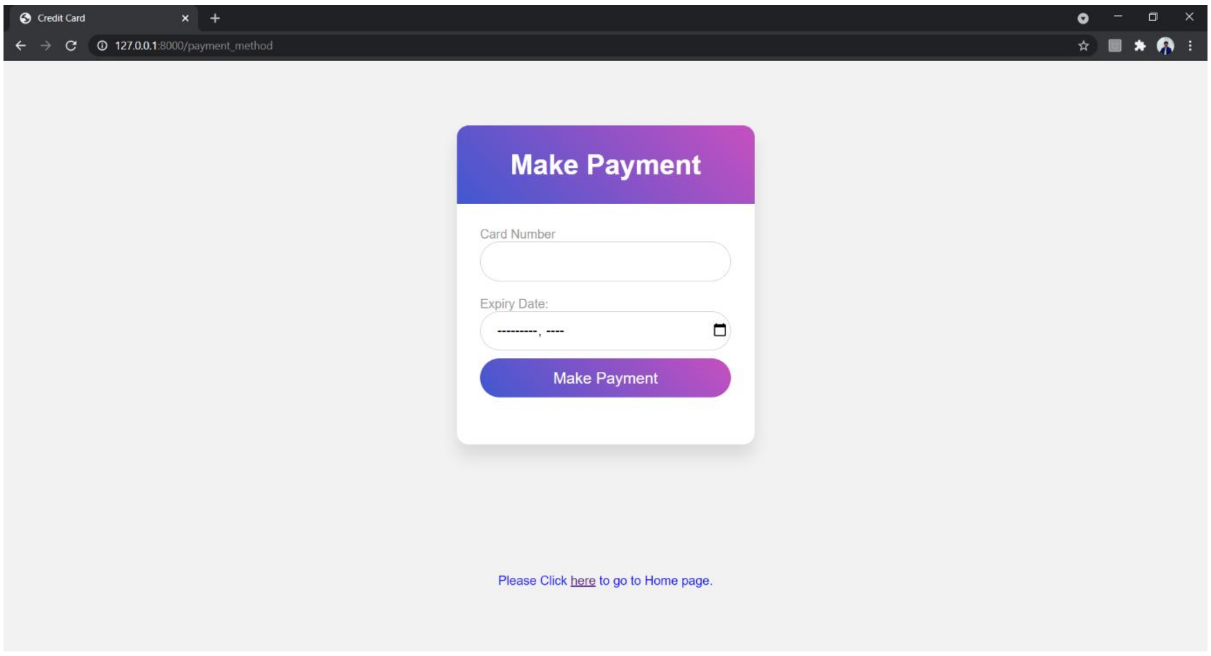
E-mail

Select Payment Method

Continue

Indigo 1I 123	Surat ⟶ Mumbai	April 6, 2021 ⟶ April 6, 2021 6 a.m. 7 a.m.	2 Travellers Class	6000 ₹
Indigo 2I 123	Mumbai ⟶ Surat	April 7, 2021 ⟶ April 7, 2021 9 a.m. 10 a.m.	2 Travellers Class	8000 ₹

Please Click here to go to Home page.



Credit Card

127.0.0.1:8000/payment_method

Make Payment

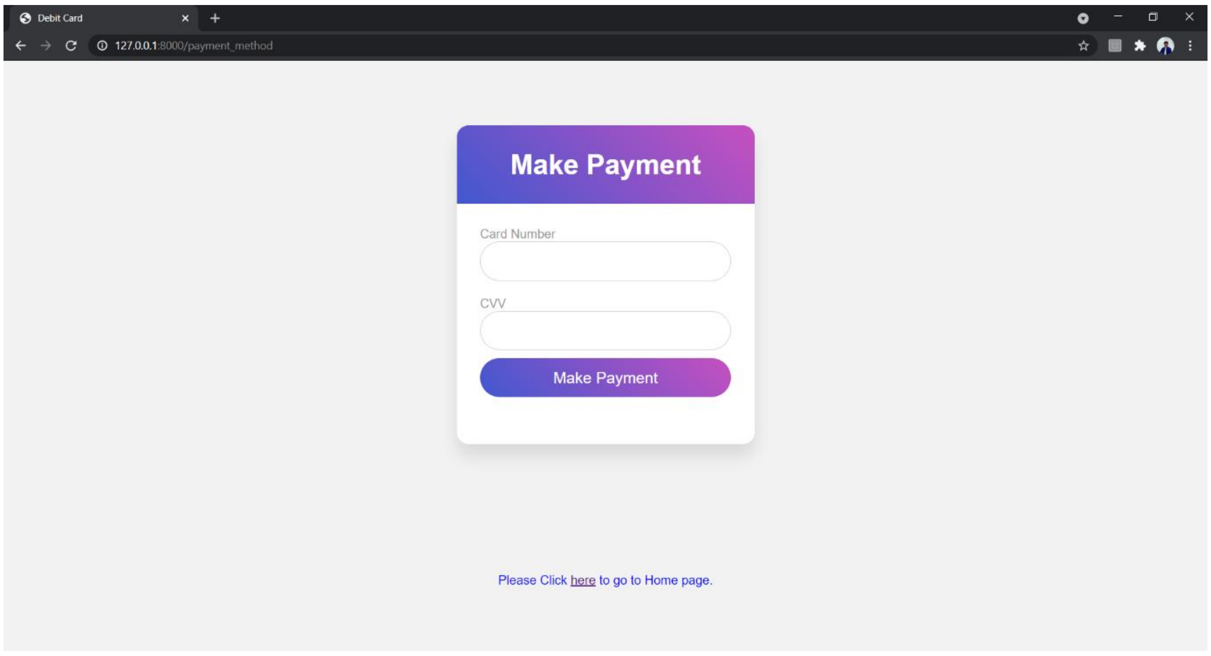
Card Number

Expiry Date:

---------, ----

Make Payment

Please Click here to go to Home page.



Debit Card

127.0.0.1:8000/payment_method

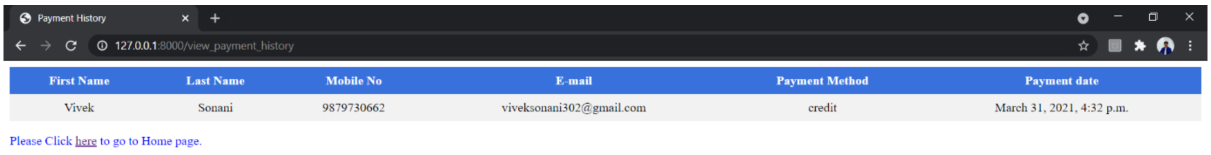
Make Payment

Card Number

CVV

Make Payment

Please Click here to go to Home page.

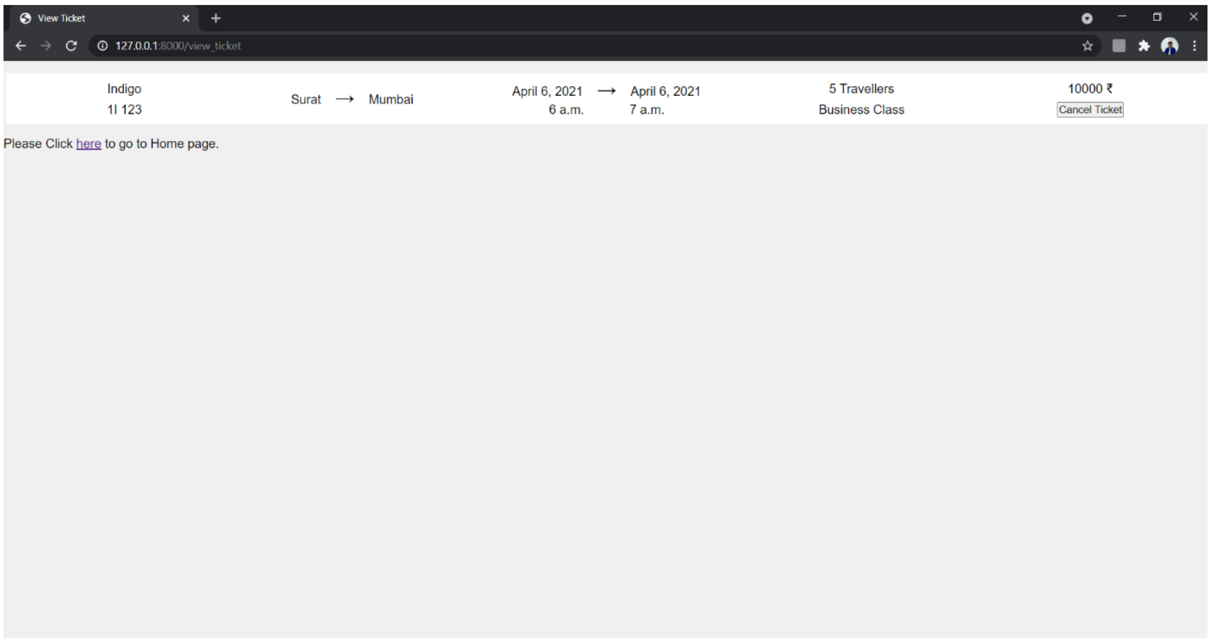

Payment History

127.0.0.1:8000/view_payment_history

First Name	Last Name	Mobile No	E-mail	Payment Method	Payment date
Vivek	Sonani	9879730662	viveksonani302@gmail.com	credit	March 31, 2021, 4:32 p.m.

Please Click here to go to Home page.

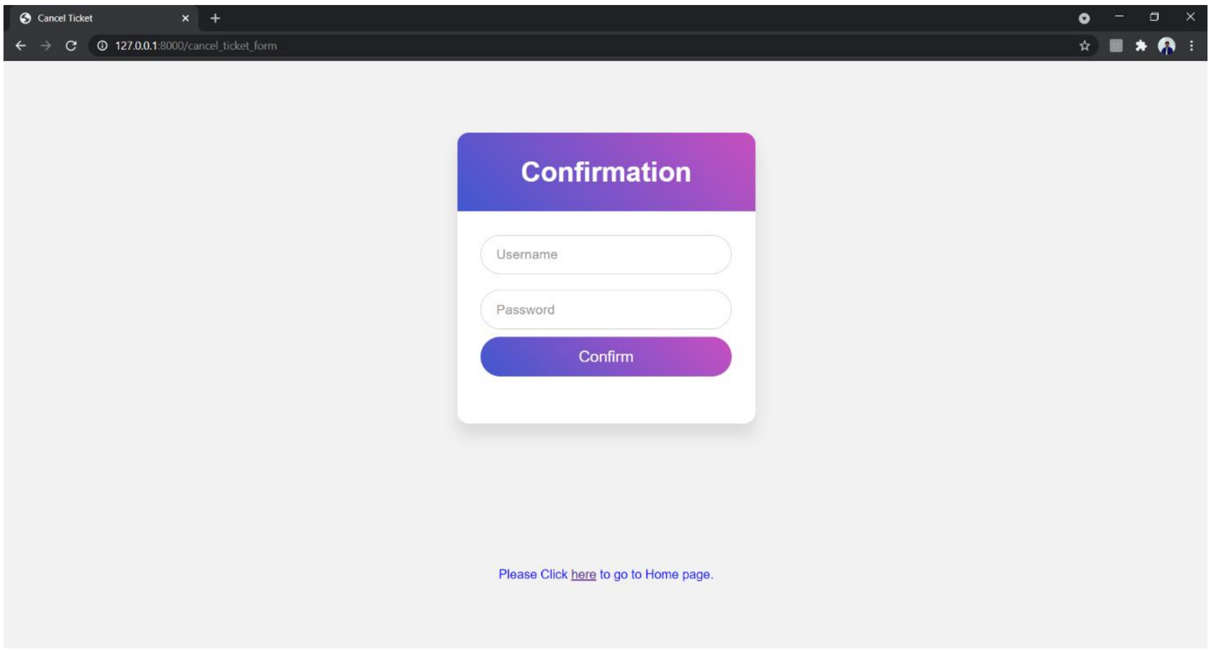
Manage Ticket:


View Ticket

127.0.0.1:8000/view_ticket

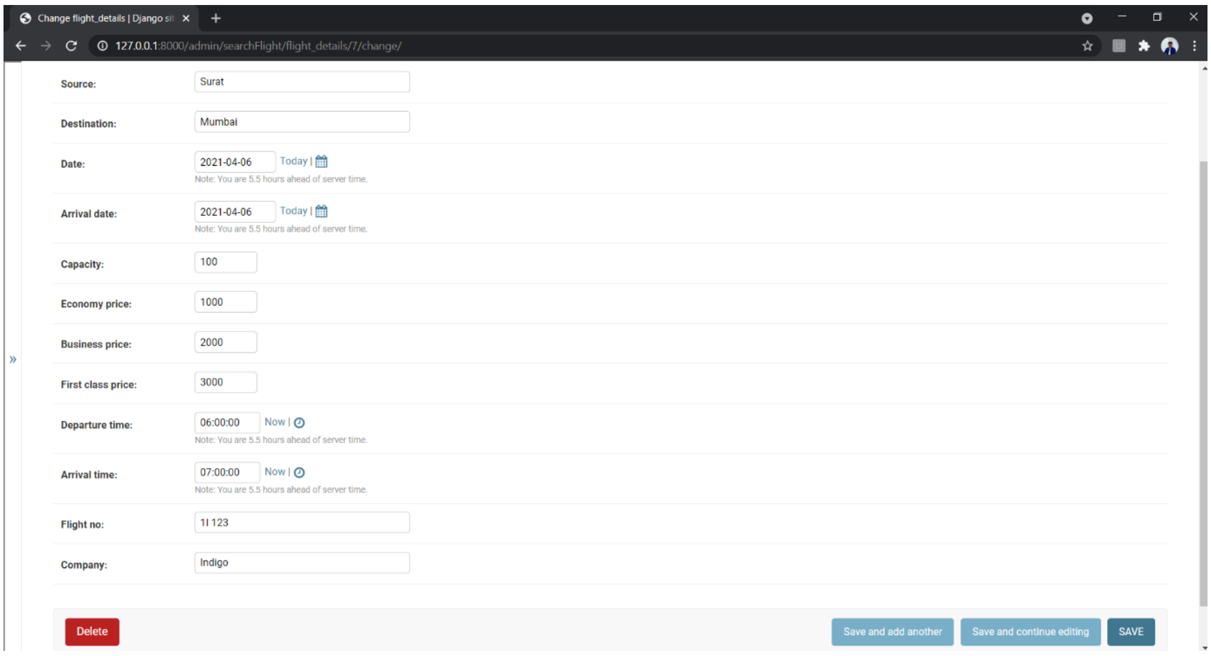
Indigo 1I 123	Surat ⟶ Mumbai	April 6, 2021 6 a.m. ⟶ April 6, 2021 7 a.m.	5 Travellers Business Class	10000 ₹ Cancel Ticket

Please Click here to go to Home page.



Please Click here to go to Home page.

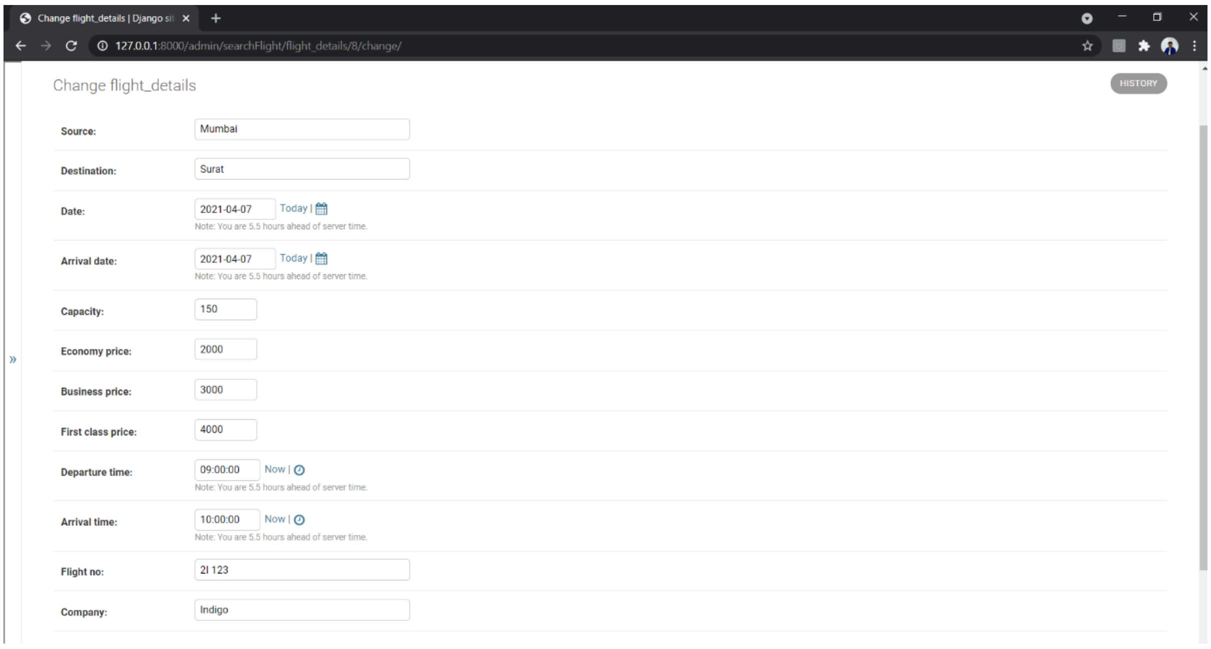
Manage Flight Details:



Change flight_details | Django sit
127.0.0.1:8000/admin/searchFlight/flight_details/7/change/

Source:	Surat
Destination:	Mumbai
Date:	2021-04-06 Today | Note: You are 5.5 hours ahead of server time.
Arrival date:	2021-04-06 Today | Note: You are 5.5 hours ahead of server time.
Capacity:	100
Economy price:	1000
Business price:	2000
First class price:	3000
Departure time:	06:00:00 Now | Note: You are 5.5 hours ahead of server time.
Arrival time:	07:00:00 Now | Note: You are 5.5 hours ahead of server time.
Flight no:	1I 123
Company:	Indigo

Delete
Save and add another
Save and continue editing
SAVE



Change flight_details | Django sit
127.0.0.1:8000/admin/searchFlight/flight_details/8/change/
Change flight_details
HISTORY

Source:	Mumbai
Destination:	Surat
Date:	2021-04-07 Today | Note: You are 5.5 hours ahead of server time.
Arrival date:	2021-04-07 Today | Note: You are 5.5 hours ahead of server time.
Capacity:	150
Economy price:	2000
Business price:	3000
First class price:	4000
Departure time:	09:00:00 Now | Note: You are 5.5 hours ahead of server time.
Arrival time:	10:00:00 Now | Note: You are 5.5 hours ahead of server time.
Flight no:	2I 123
Company:	Indigo

Conclusion:

Admin can enter the all-flight details and user will be able to see the flight details after login to the system. User can book the ticket by making payment and also cancel the booked ticket. Reserved tickets and Transaction history can be seen from the system.

Limitations and Future Extension:

This system does not ask to enter the details of all commuters but the details of only one commuter can be entered. This system does not allow to download and print the ticket.

This system can be extended to ask user to enter the details of all commuters. We can extend the system to allow user to download and print the ticket.

Bibliography:

References used for developing project:

- https://youtu.be/OTmQOjsl0eg
- https://stackoverflow.com/
- https://docs.djangoproject.com/en/3.1/
- https://www.w3schools.com/