# A PROJECT REPORT ON FLIGHT TICKET BOOKING SYSTEM

## ✈️ SKA Flight Ticket Booking System Documentation

### Overview 📝

The SKA Flight Ticket Booking System is a comprehensive, web-based airline management application developed using Flask (Python). It delivers a robust platform supporting airline administration, user bookings, and guest interactions. The system is modularized into three main segments for Admin, User, and Guest roles, ensuring scalability, maintainability, and clear separation of concerns.

### DECLARATION 📝

I hereby declare that the project report titled “SKA FLIGHT TICKET BOOKING SYSTEM” is original work done by Adepu Sai Kiran during the period June 2024 to August 2025 ✈️, to the best of my knowledge and belief.

### Abstract ✈️

The SKA Flight Ticket Booking System is a web-based application designed to manage flight bookings and passenger journey histories. This project demonstrates the core functionalities of an airline reservation system, aiming to simplify and streamline ticket booking processes for users. The system is developed using technologies such as Flask (Python), HTML, CSS, JavaScript, and MySQL for database management. 🖥️💾

It delivers efficient flight scheduling, fare management, ticketing, and user feedback functionalities. The system replaces traditional manual booking methods, saving time ⏳ and increasing operational efficiency 🚀. Proper requirements analysis and system design ensure the platform meets customer needs while maintaining compliance and scalability. 📈✅

# SKA FLIGHT BOOKING SYSTEM INDEX PAGE


1. Introduction
2. AIM of The Project
3. The Application Main Purpose
4. THE PROJECT ARCHITECTURE / PROJECT HIERACHY
5. The Index Page of SKA Flight Ticket Booking System
6. Application Modules
7. Flight Admin Management Module
8. Registered User Module
9. New User Registration Module
10. Guest User Module
11. SYSTEM REQUIREMENTS
12. SYSTEM DESIGN
13. SKA FLIGHT BOOKING SYSTEM ARCHITECTURE DIAGRAM
14. Flight Reservation Booking Terms
15. Flight Types / Terms
16. Flight Seats Layout System
17. Customer's Obligations
18. THE SKA FLIGHT TICKET BOOKING SYSTEM HIERARCHY / PROCESS
19. PROJECT SCHEDULING
20. Benefits of Online Flight Ticket Booking
21. SKA FLIGHT TICKET BOOKING SYSTEM APPLICATION DATABASE/TABLES
22. THE SKA FLIGHT BOOKING SYSTEM APPLICATION CALLS/FLOW
23. THE SKA FLIGHT BOOKING SYSTEM ENTIRE APPLICATION ENVIRONMENT
24. THE SKA FLIGHT BOOKING SYSTEM VERSIONS
25. Issues Encountered and Resolutions
26. Conclusion
27. Future Look of SKA Flight Ticket Booking System
28. Author's Note
29. Contact Information
30. Acknowledgements

# Introduction ✈️

The SKA Flight Ticket Booking System project is a model internet-based flight ticket reservation platform. This system enables customers/users to book flight tickets conveniently from their office or home using a PC or laptop. The system provides users the ability to create an account, search and book flights, and view reports of their entire travel history. Customers can log in to their accounts to manage bookings and access detailed flight information according to their preferences.

With internet booking, the traditional flight booking structure transforms into an online booking model, giving the concept of virtual booking a real shape. Today's booking is no longer limited to physical airline counters or offices. E-booking facilitates flight reservations by customers 24/7 globally. Anyone who registers as a user can become a member of the SKA Flight Booking System by providing personal details through a simple registration form.

To maintain customer trust and satisfaction, there is a vital need for seamless management of flight reservations that operates with ease and comfort. Efficient management positively impacts both customers and staff, empowering the management committee to make informed decisions for future enhancements. Managing flight bookings manually can be tedious and time-consuming, which makes this software essential for improving efficiency and speed in booking processes.

All transactions are securely conducted online, allowing for quick and hassle-free payment processing. This software aims to eliminate the drawbacks of manual booking systems by offering a fast, reliable, and user-friendly platform. 🌐 💼



## Synopsis 📋

The SKA Flight Ticket Booking System maintains a comprehensive daily record as a complete airline booking solution. It manages user account registrations, flight bookings, feedback submissions, mobile number updates, and password changes. This system is designed to streamline flight booking operations and enhance user experience with secure and efficient transaction handling. ✈️ 💻 💳

## AIM of The Project 🎯

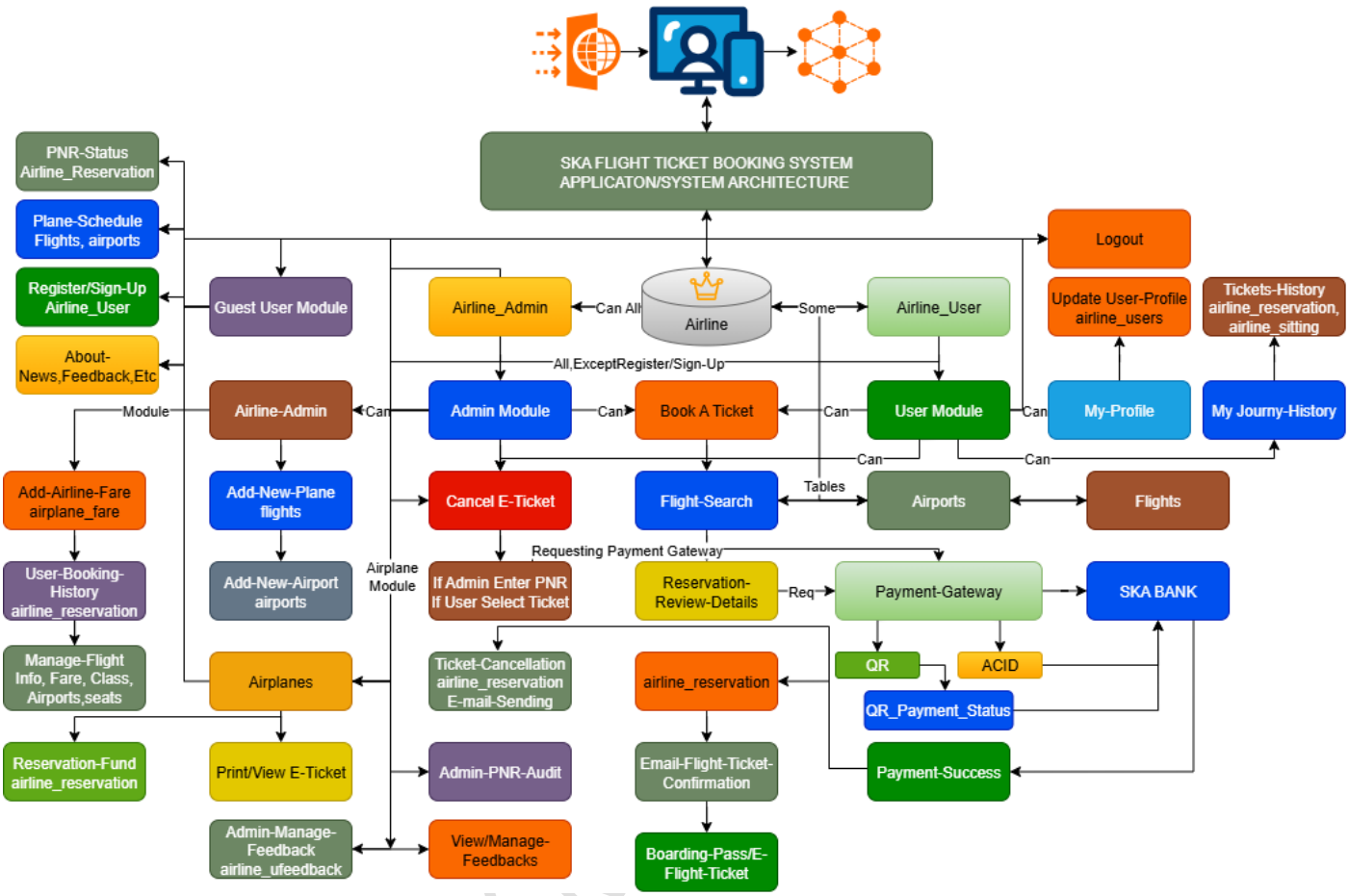
The main aim of designing and developing this Flight Ticket Booking System using Flask (Python) and MySQL is to provide secure and efficient online booking facilities to airline customers over the internet. The application runs on a web server and allows customers to log in through a secured webpage using their user ID, username, and password. Users can access all essential features such as booking flights, viewing journey history, and submitting feedback, ensuring a seamless travel booking experience. 🌐✈️🔐

## The Main Purpose 🛠️

Traditionally, maintaining user details and booking flights required users to visit airline counters physically, which could be inconvenient and time-consuming for both passengers and airline administrators. This project aims to automate the entire flight booking process online, providing real-life simulation of an airline reservation system and the activities performed by various roles.

The SKA Flight Booking System captures and manages booking activities, passenger details, payment processing, and administrative tasks, all through an intuitive internet-based platform. This automation enhances the efficiency and accuracy of flight reservation management while reducing manual workload and improving customer satisfaction. 🚀💼

# THE PROJECT ARCHITECTURE ✈ / PROJECT HIERACHY




**Main Modules:**

**Admin: Airlines← Admin-Dashboard←Admin.Routes(Python)←Admin.Routes(Html)**

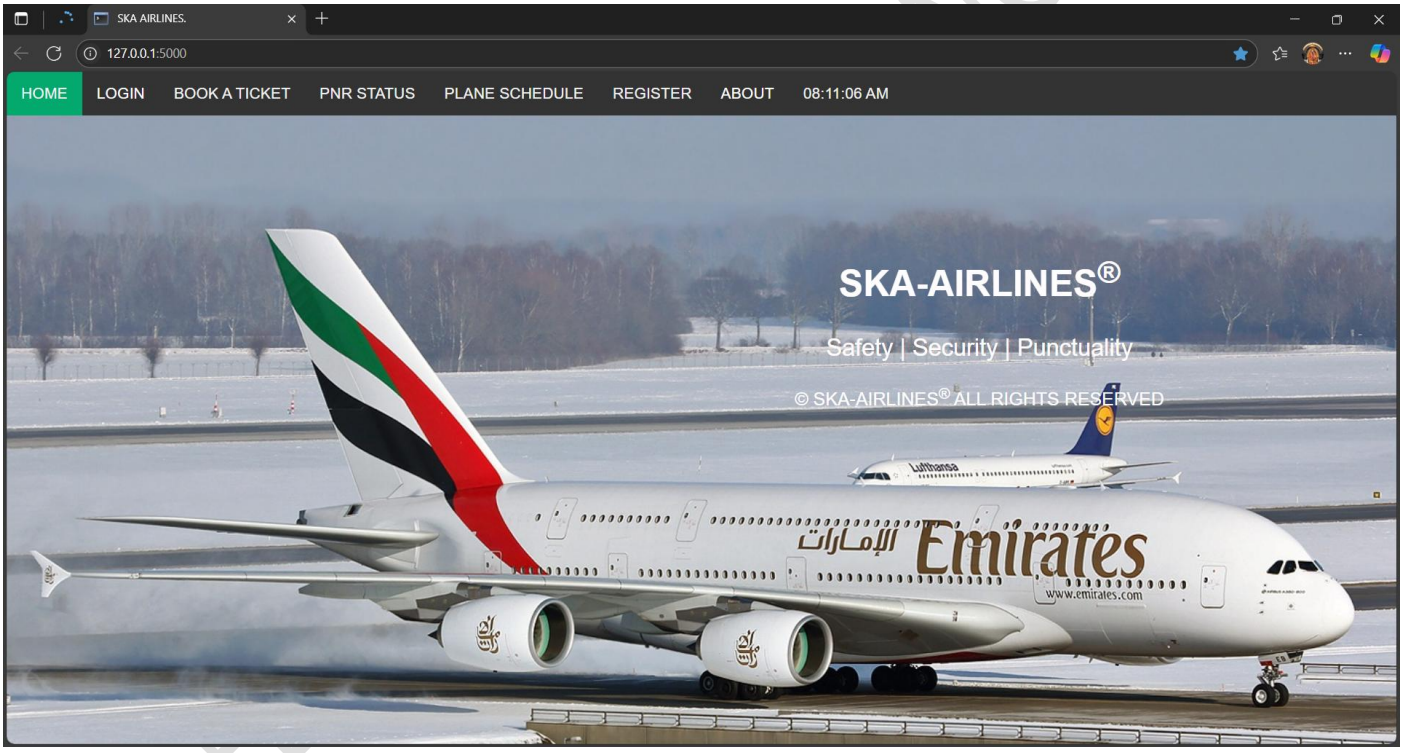
**User:Airlines←User-Dashboard←User.Routes(Python)←User.Routes(Html)**

**Guest User:Airlines←Guest-Dashboard←Guest.Routes(Py)←Guest.Routes(Html)**

# The Index Page of SKA Flight Ticket Booking System

The Index Page of the SKA Flight Ticket Booking System serves as the main entry point for users visiting the website. It provides easy navigation and access to key features such as user login, guest access, and flight search. The page is designed to be user-friendly, allowing registered users to quickly log in and manage bookings, while guests can explore available flights, schedules, and register for new accounts.

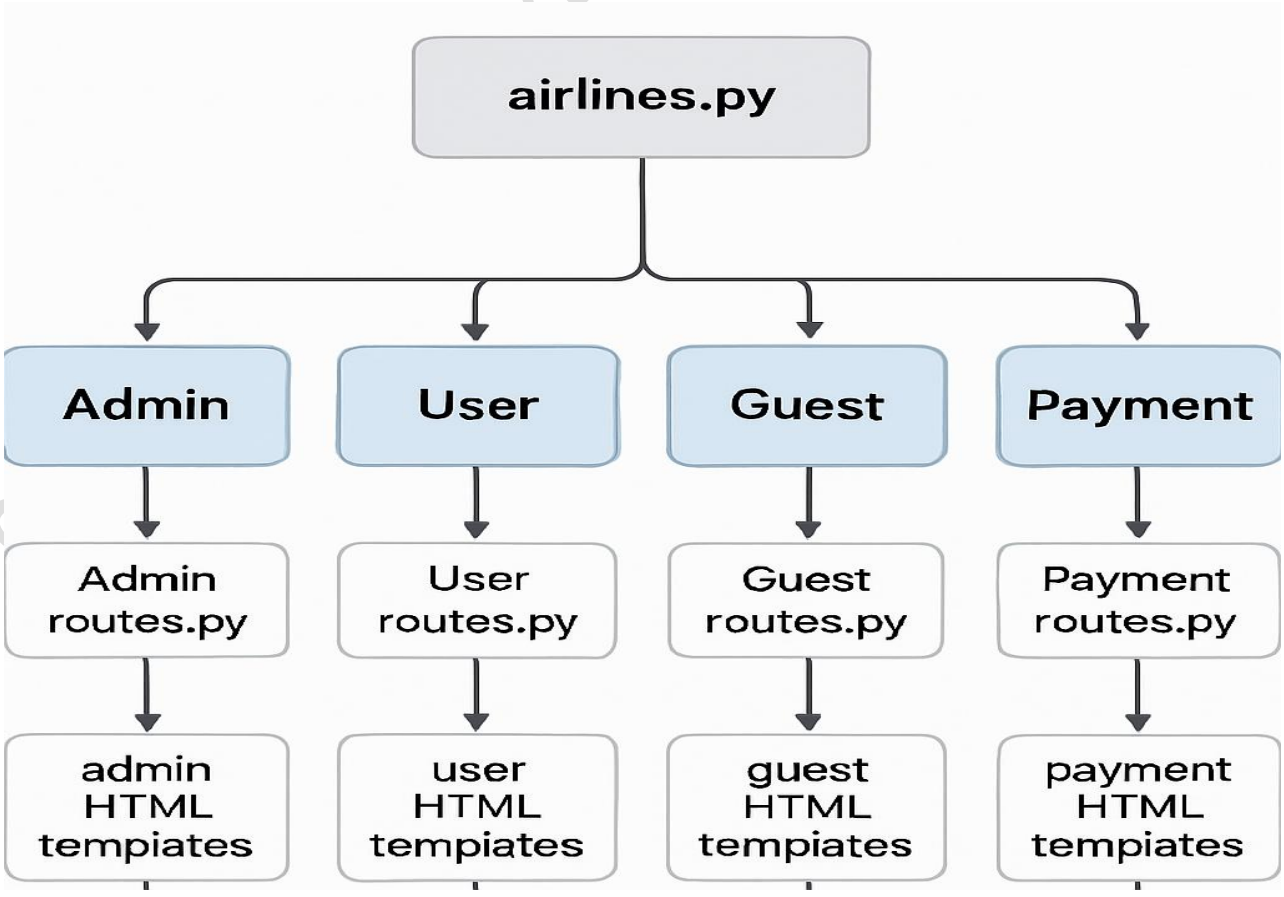
The index page also offers essential information and links to support pages like help, feedback, and contact details, creating a smooth and intuitive experience for all users. Its clean layout and responsive design ensure accessibility across various devices.



# Main Modules of SKA Flight Ticket Booking System 🚀

The system is primarily divided into four main modules, each serving different user roles and functionalities:

1. Flight Admin Module 🛬
   Responsible for managing all airline operations including flight scheduling, fare management, ticket administration, and handling feedback. Admins have secure access to control and monitor the entire system.
2. Registered User Module 🧳
   Allows registered customers to log in, book flights, view journey history, manage their profiles, make payments, and provide feedback.
3. New User Registration Module 📝
   Handles the onboarding of new users by enabling account creation with secure authentication, allowing effortless access to flight booking services.
4. Guest User Module 👥
   Caters to non-registered users by offering flight schedule browsing, PNR status checking, and access to general information without the need for logging in.



## What to Expect / Modules of SKA Flight Ticket Booking System 🚀

The system is designed with multiple modules, each packed with features tailored to different user roles. Below is a detailed overview of the key modules and functionalities you can expect in this platform:

### 1. Flight Admin Module 🛬

The Flight Admin is the primary user responsible for managing airline operations. Admins log in securely with their username and password, which are validated against the flight booking database. After successful authentication, the admin can perform a wide range of critical tasks to ensure the smooth operation of the airline's booking system.

**Main Features & Submodules for Flight Admin:**

Flight Management: Add and schedule new flights, assign aircraft to routes, and update flight details.

Airport Management: Add new airports and maintain existing airport information.

Fare Management: Define, update, and modify fare structures for various flights and routes.

Ticketing: Administer ticket issuance, cancellations, and downloads.

PNR Enquiries & E-ticket Management: Lookup passenger name records and issue/manage electronic tickets.

User Booking History: Access comprehensive user booking records for tracking and support.

Feedback Management: Review and respond to customer feedback efficiently.

Seat Availability Monitoring: Monitor and manage seat inventory in real-time.

Payment Gateway Oversight: Supervise payment processing, including ticket cancellations and refunds, Additional Controls: Manage administrative logins, notifications, and other system settings.

These tasks are organized into submodules powered by Flask Blueprints, such as:

new_flight_bp for flight scheduling, add_airport_bp for airport data

add_fare_bp and change_fare_bp for fare control, ticket_admin_bp for ticket management

user_booking_history_bp for booking tracking …and many more.

This modular design enhances flexibility, maintainability, and scalability of the admin functionalities. 🔧⚙️

**2. Registered User Module** 🧳

Users who register on the platform can log in to:

Search and book flights

Cancel bookings when necessary

Make payments via integrated gateways including QR code options

Update personal information such as mobile numbers and passwords

Submit feedback and view journey history

**3. New User Registration Module** 📝

New users can easily create accounts by filling out the registration form. Upon successful validation, they gain access to all user functionalities, enabling smooth transitions from guests to registered customers.

**4. Guest User Module** 👥

Non-registered users or guests can:

Browse flight schedules and availability

Check their PNR (Passenger Name Record) status

View airline news and events

Submit anonymous feedback

This comprehensive modular architecture brings together admins, users, and guests, offering an efficient, secure, and user-friendly flight booking ecosystem. ✈️🌐

# 1. Flight Admin Management Module 🛫✨

The Flight Admin Management module is the core of the SKA Flight Ticket Booking System's administration. It empowers administrators with full control over airline operations, ensuring smooth scheduling, fare management, and ticketing. Admins securely log in with their credentials maintained in the airline database. 🔐👨‍✈️

**Key Responsibilities and Functionalities** 🛠️

After successful login, the Flight Admin can perform the following crucial tasks, organized into several submodules:

1. Flight Operations Management ✈️

Add New Flights: Create and schedule flights, assigning airplanes to routes (new_flight_bp) 🆕🛫

Manage Flight Details: Update, modify, or cancel flights as needed 🔄

View Plane-Airport Mapping: Manage relationships between airplanes and airports (airplane_airports_bp) 🛩️📍

2. Airport Management 🏢

Add Airports: Register new airports into the system (add_airport_bp) ➕

Maintain Airport Details: Update existing airport information 🛠️

3. Fare Management 💰

Set New Fares: Define fares for flights and routes (add_fare_bp) 💵

Modify Existing Fares: Update or change fares based on requirements (change_fare_bp) ✍️

4. Ticket Administration 🎫

Ticket Issuance & Management: Handle booking confirmations, issuance, and ticket views (ticket_admin_bp) ✅

Ticket Downloads: Enable downloading of tickets (ticket_download_admin_bp) ⬇️

Ticket Cancellation: Manage cancellation requests and refunds (admin_can_ticket_bp, payment_gateway_tkt_can_admin_bp) ❌↩️

5. PNR Enquiries 🔍

Lookup and manage Passenger Name Records to assist passengers 🖨️

6. User Booking History 📚

Access detailed user booking records (user_booking_history_bp) 🗂️

7. Feedback Management 📨

Review and respond to customer feedback for service improvement (admin_view_feedback.py, admin_delete_feedback.py) 💬👍

8. Seat Availability Monitoring 💺

Monitor and manage seat inventory in real time (seat_avl_bp) 📊

9. Payment Gateway Oversight 💳

Oversee payment transactions including ticket purchases and cancellations (payment_gateway_tkt_can_admin_bp) 🔐💸

10. Additional Admin Controls ⚙️

Manage admin login/logout (admin_login_bp, admin_logout_bp) 🔑

Oversee airline funds and finances (airline_fund_bp) 💼

Configure system notifications and alerts 🔔

Route connection management (admin_routes.py, admin_bp) 🔄

Architecture & Implementation 🧩

These functionalities are implemented in a modular fashion using Flask Blueprints, encouraging maintainability and scalability. Some important files include:

new_flight.py, add_airport.py, add_fare.py, change_fare.py, ticket_admin.py, ticket_download_admin.py, user_booking_history.py, admin_can_ticket.py

payment_gateway_tkt_can_admin.py, airline_fund.py

…and more

**Summary** 🏁

The Flight Admin Management module offers a centralized, secure, and efficient system for airline administrators to handle backend operations, supporting seamless flight management and customer service excellence. ✈️🔧🛡️

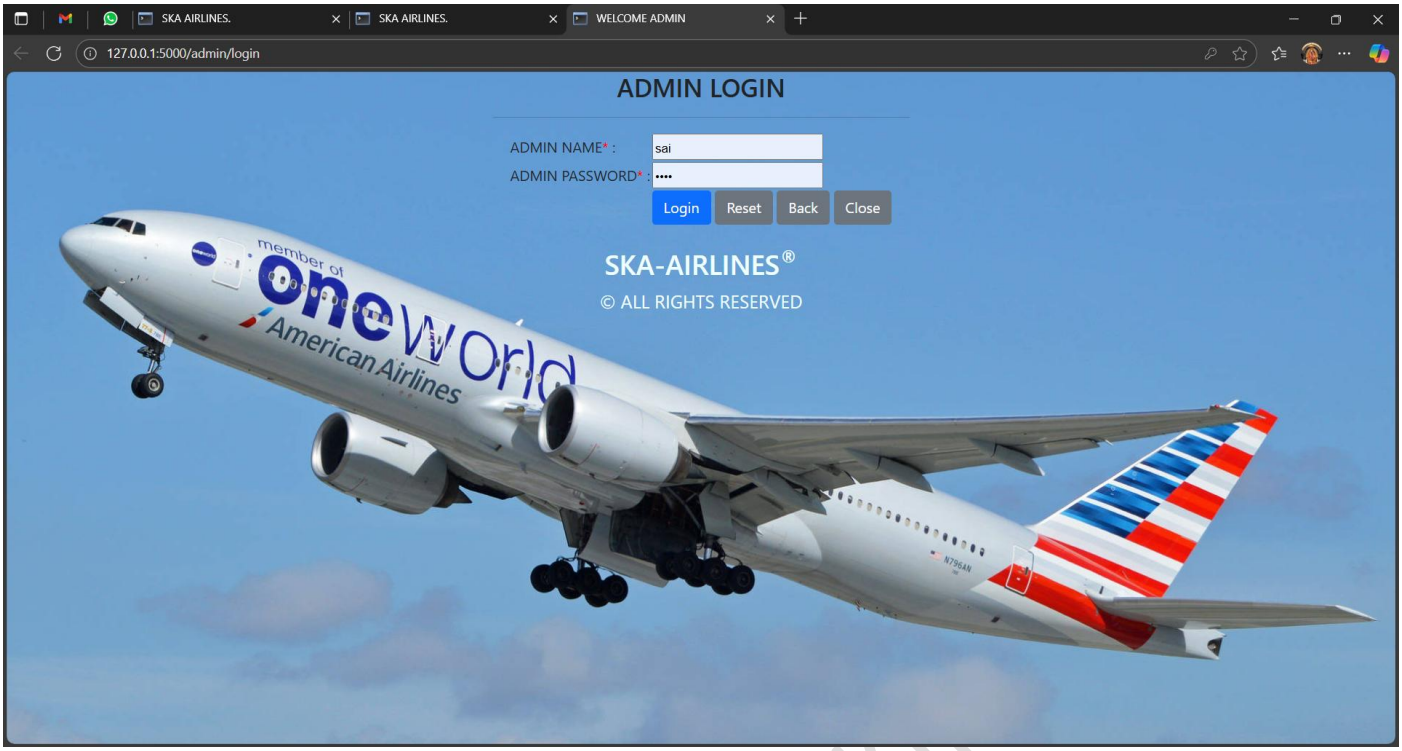
## Flight Admin Module

```
├── Admin Section
|   ├── admin_login_bp      <-- Login functions
|   ├── admin_logout_bp     <-- Logout logic
|   ├── admin_dashboard_bp  <-- HTML renderings (see below)
|   └── admin_bp            <-- Route connectors
|
|  admin_routes.py -----> Imports blueprints and submodules:
|    |
|     ├── new_flight_bp          --> new_flight.html
|     ├── add_airport_bp         --> add_airport.html
|     ├── add_fare_bp            --> add_fare.html
|     ├── change_fare_bp         --> change_fare.html
|     ├── ticket_admin_bp        --> ticket_admin.html, ticket_admin1.html, ...
|     ├── ticket_download_admin_bp  --> ticket_download_admin.html
|     ├── user_booking_history_bp   --> user_booking_history.html
|     ├── airplane_airports_bp      --> airline_airports_su.html
|     ├── admin_can_ticket_bp       --> admin_can_ticket.html, admin_can_ticket1.html, ...
|     ├── payment_gateway_tkt_can_admin_bp
|     ├── airline_fund_bp         --> airline_fund_su.html
|     └── (other submodules...)
```

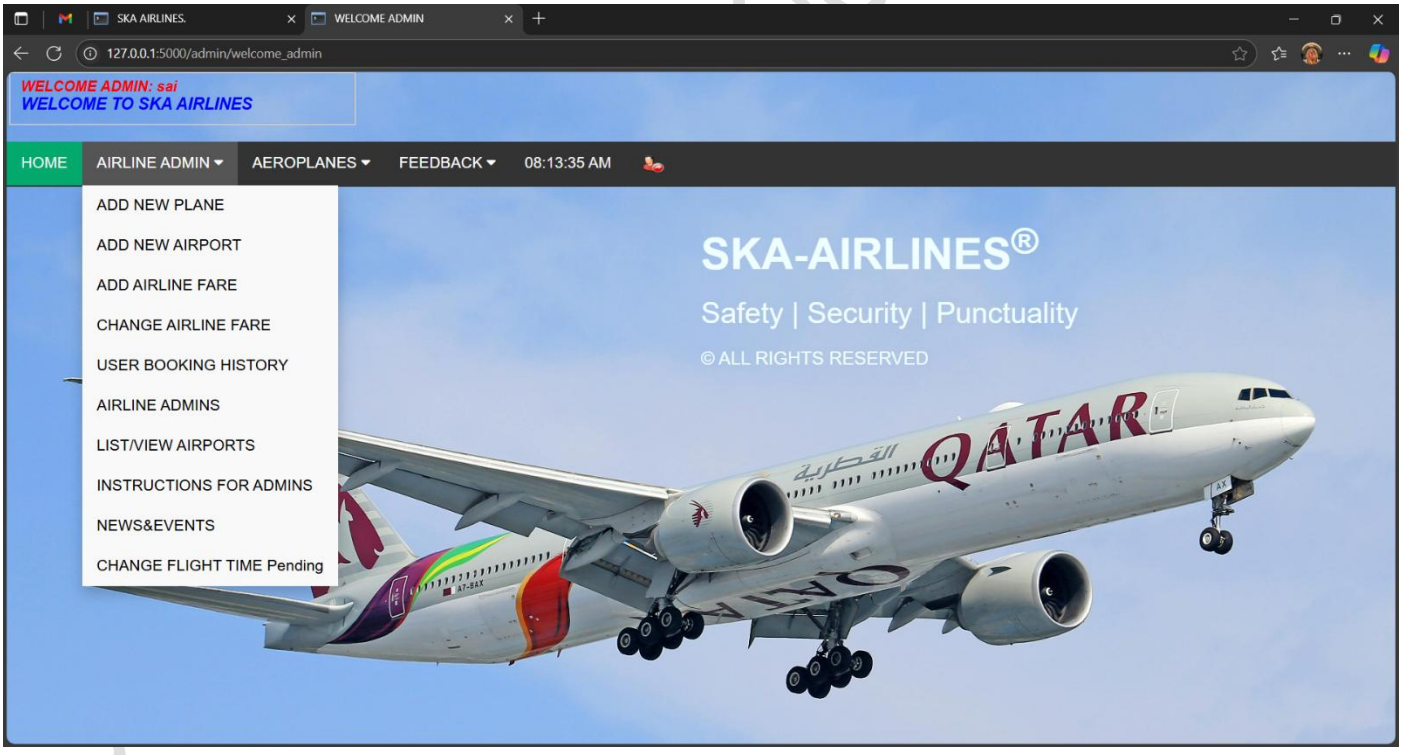


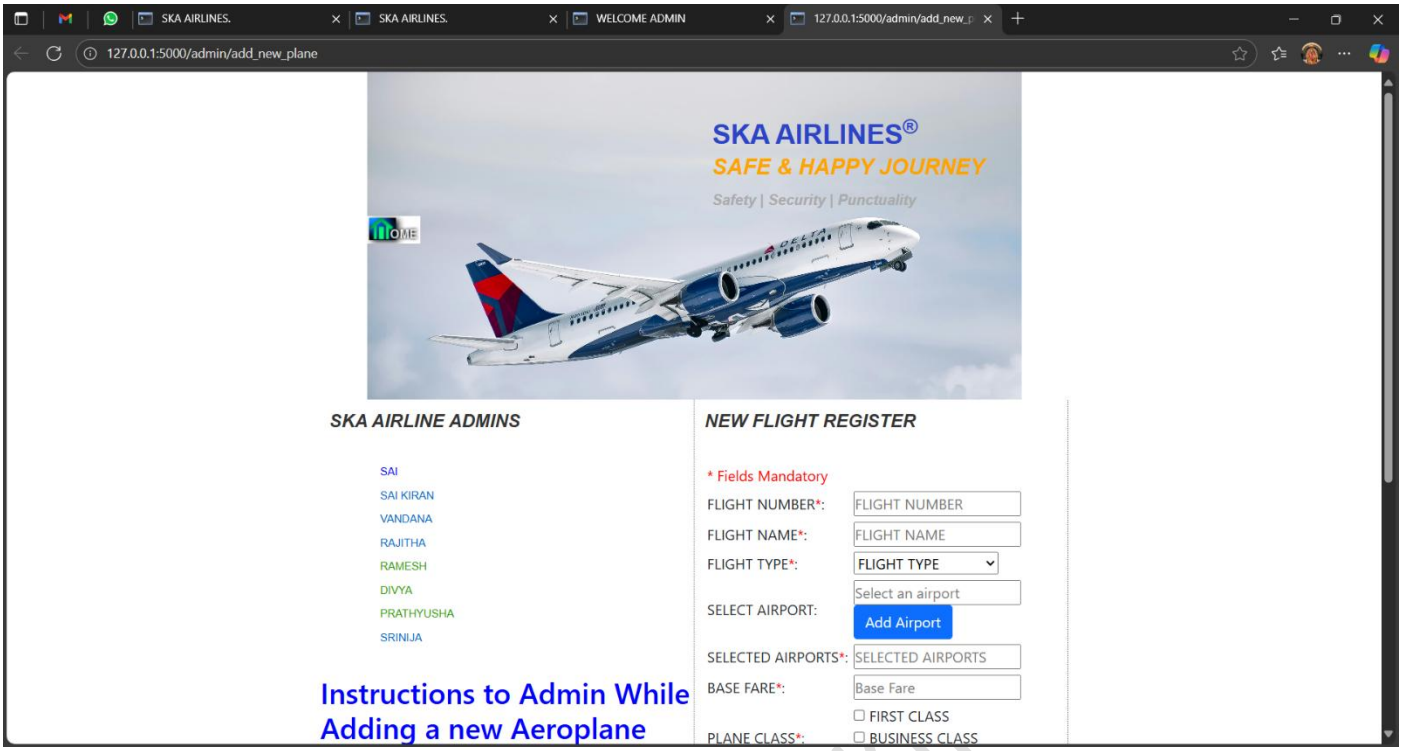
**Flight Admin Screenshots**

## ADMIN LOGIN PAGE

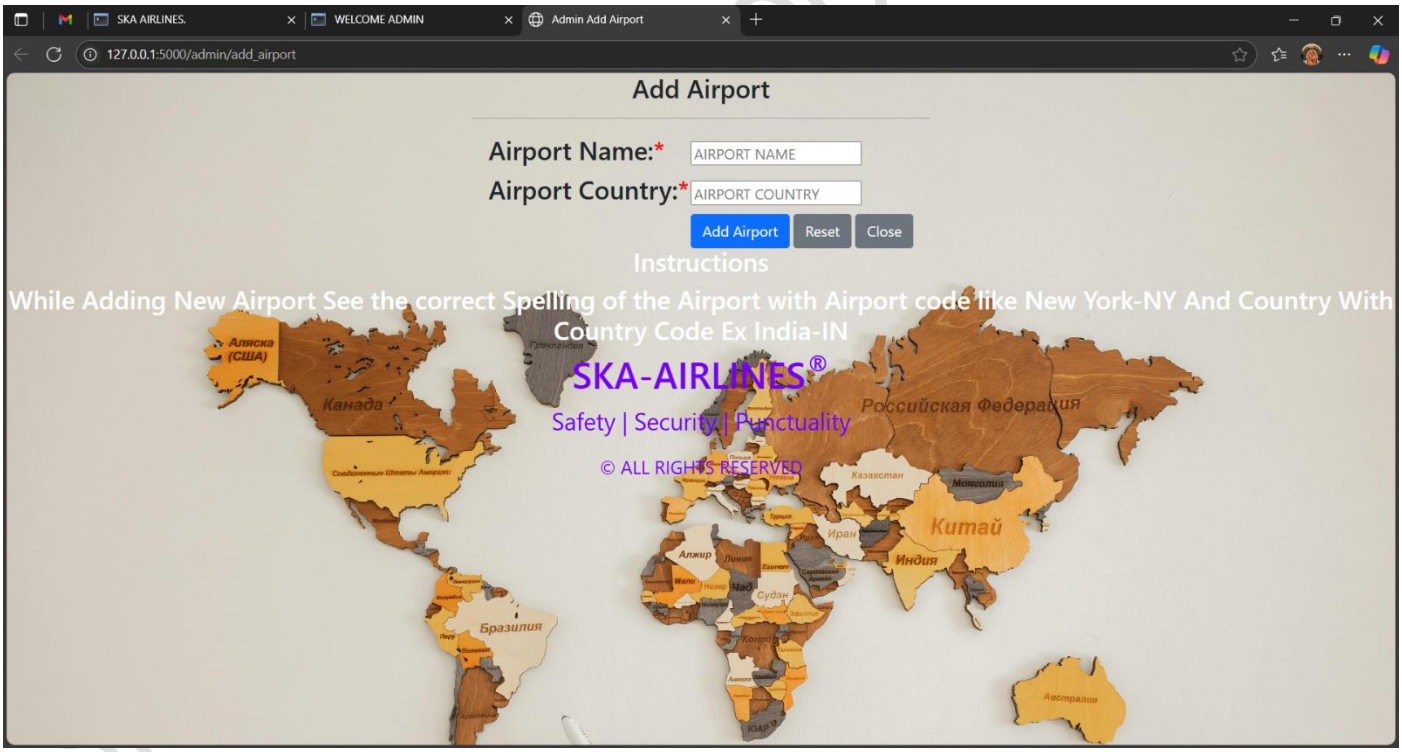


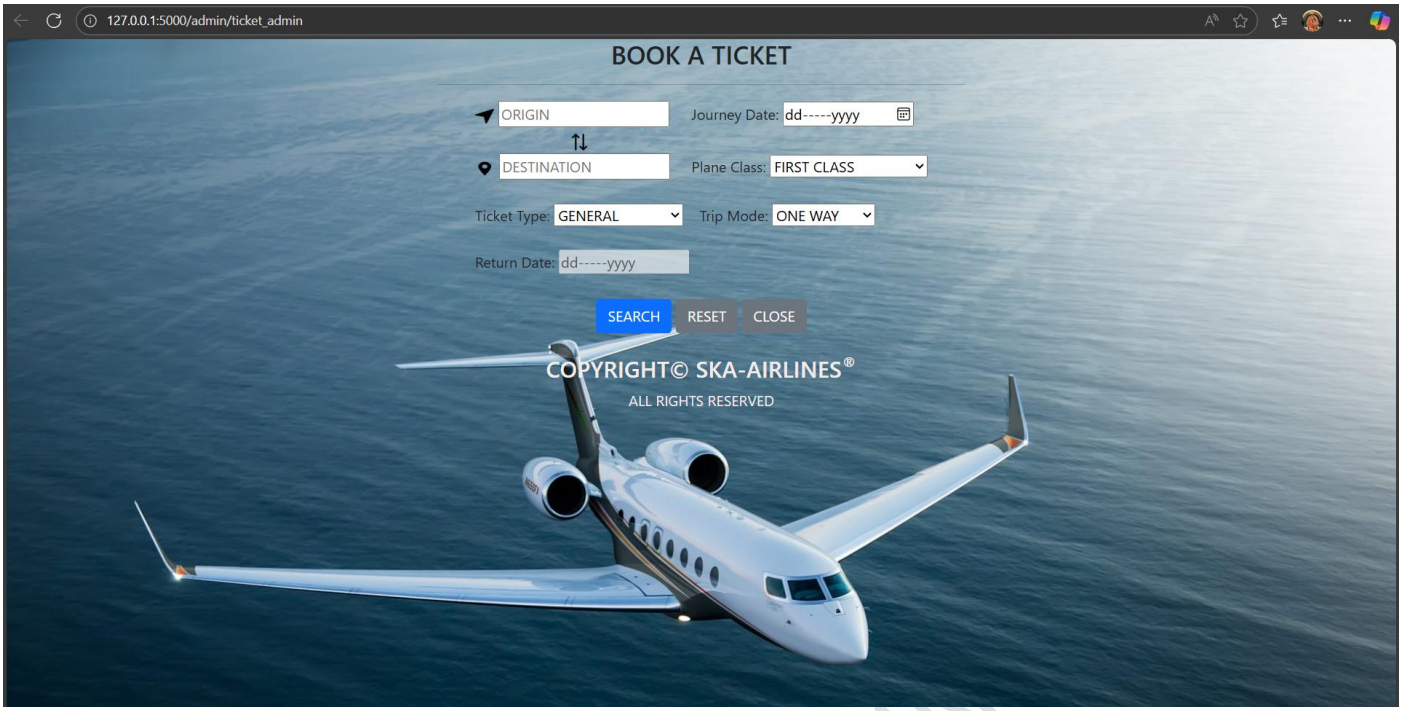

## INDEX PAGE OF FLIGHT ADMIN

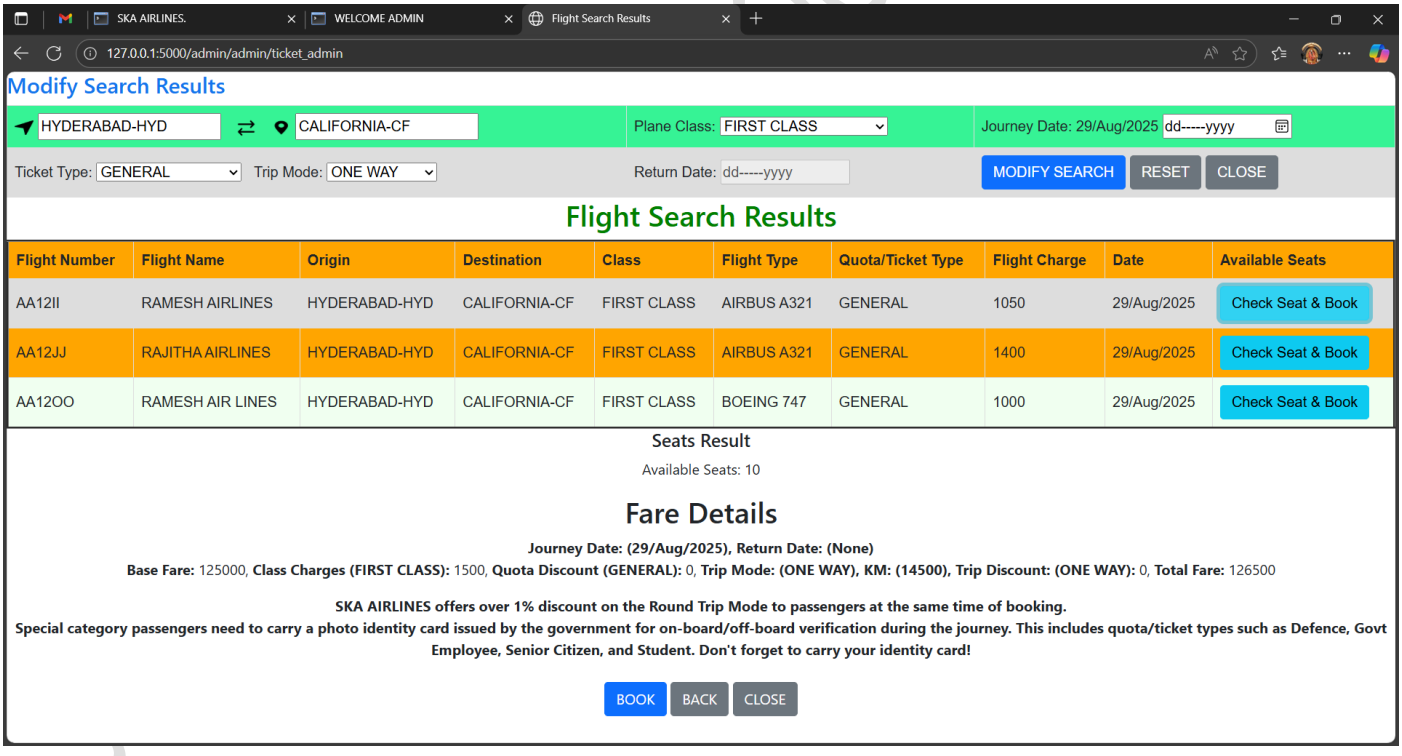


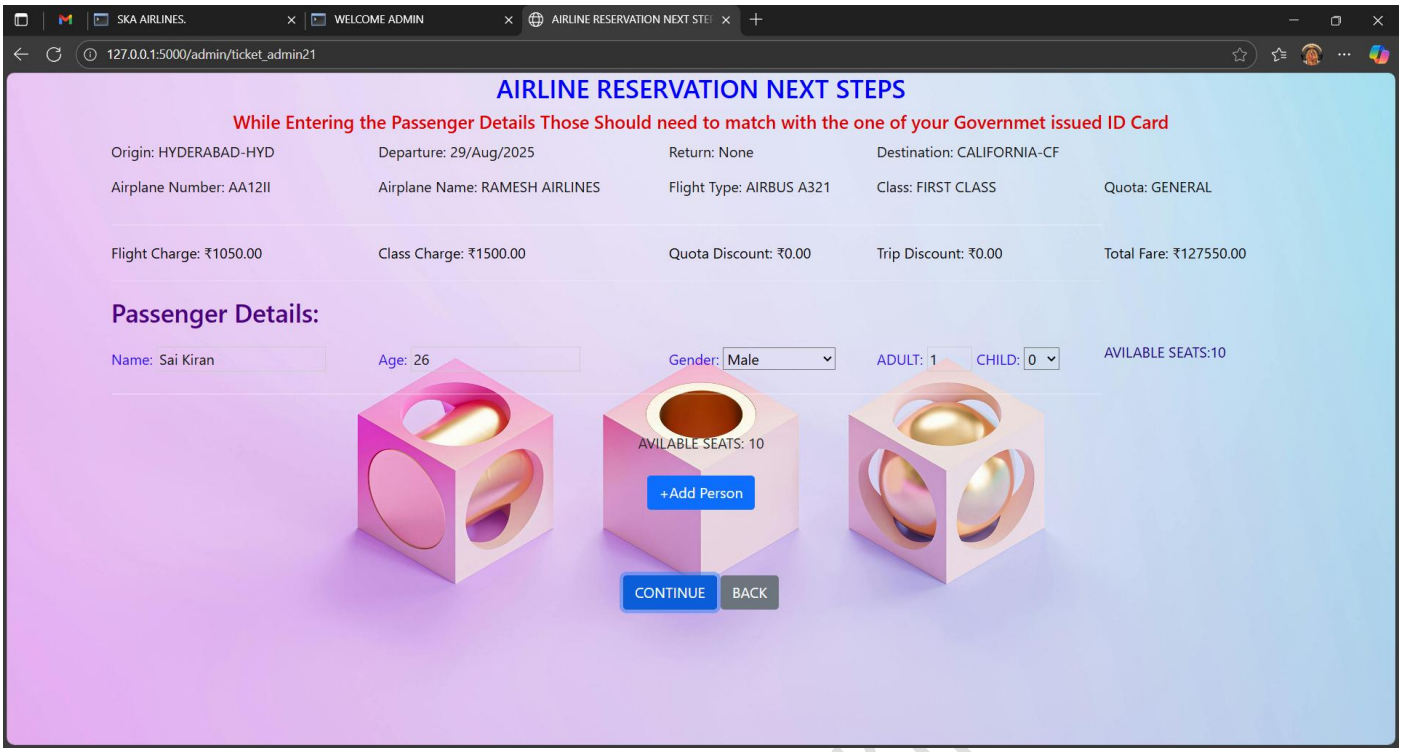


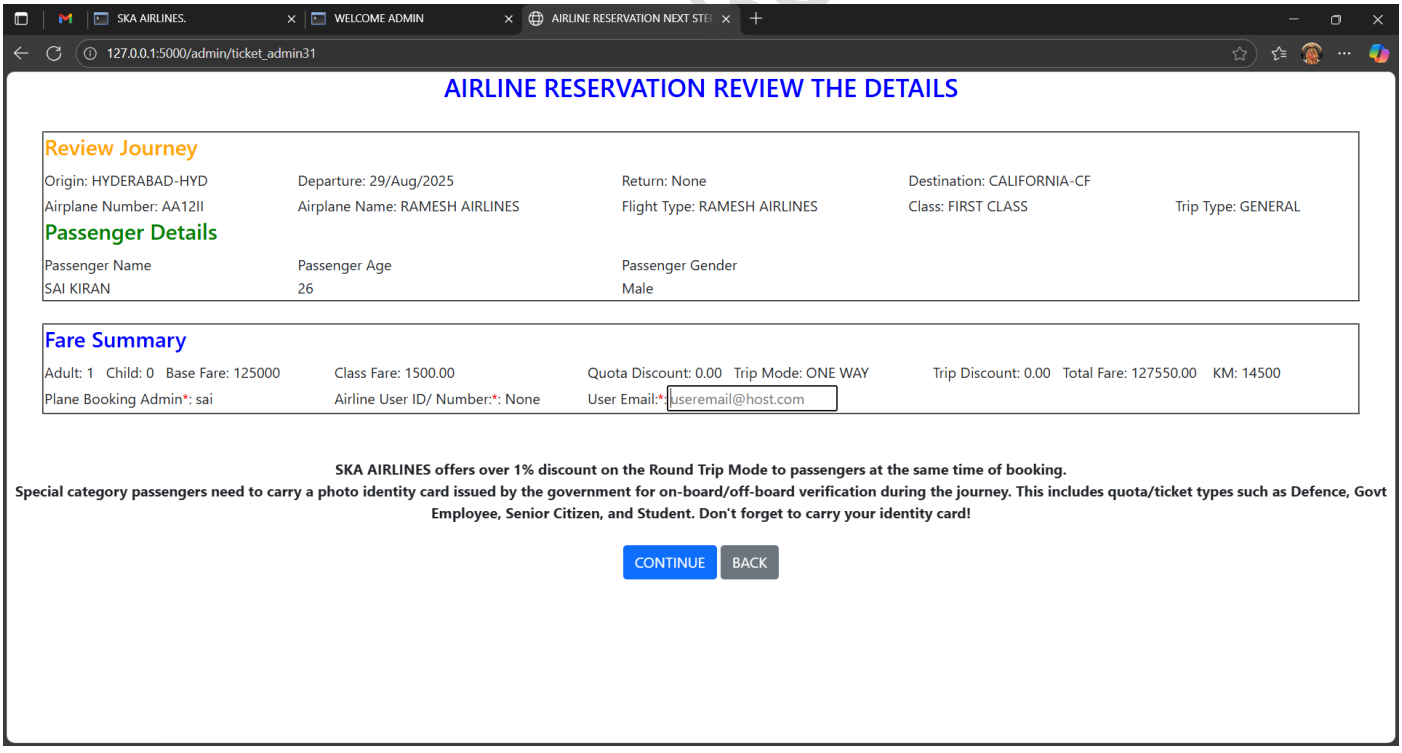

Instructions to Admin While Adding a new Aeroplane

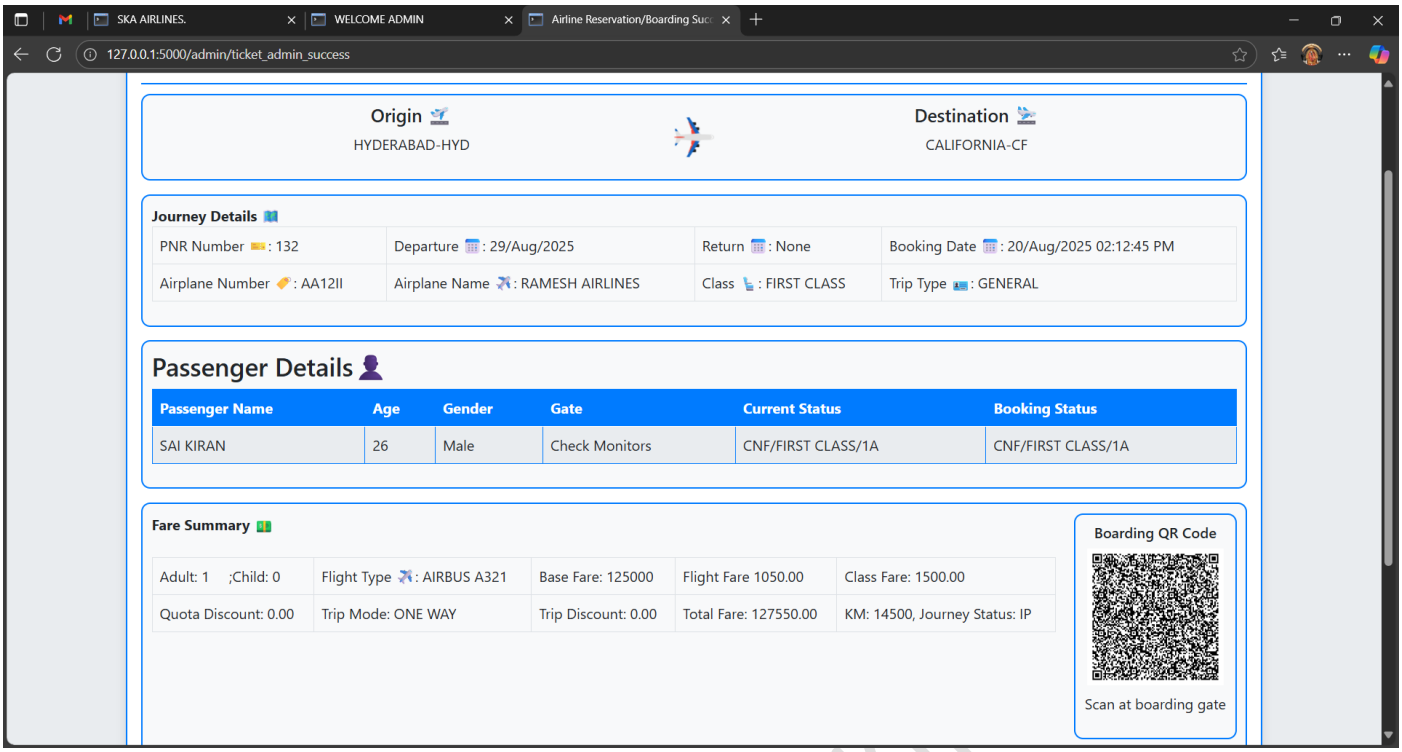


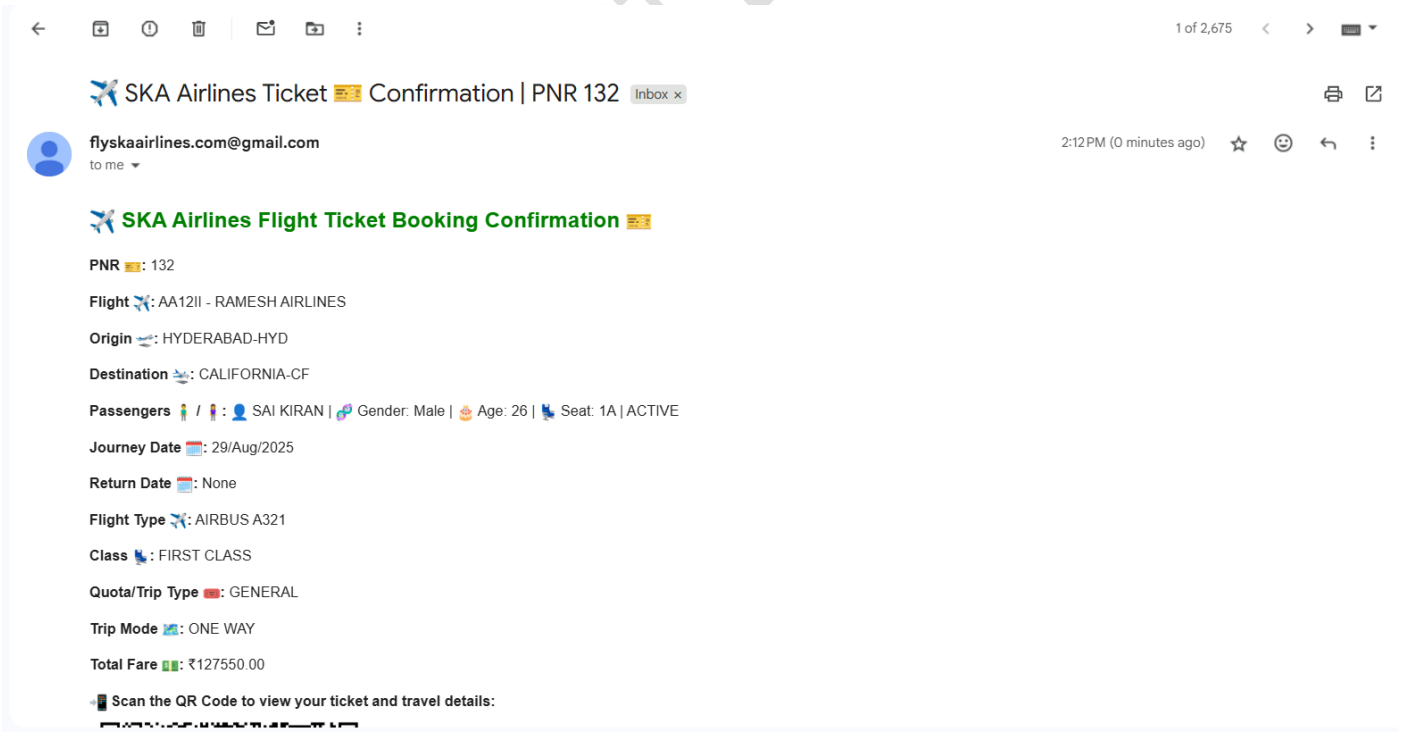

127.0.0.1:5000/admin/ticket_admin
BOOK A TICKET
ORIGIN
Journey Date: dd-----yyyy
DESTINATION
Plane Class: FIRST CLASS
Ticket Type: GENERAL
Trip Mode: ONE WAY
Return Date: dd-----yyyy
SEARCH
RESET
CLOSE
COPYRIGHT© SKA-AIRLINES®
ALL RIGHTS RESERVED


SKA AIRLINES.
WELCOME ADMIN
Flight Search Results
127.0.0.1:5000/admin/admin/ticket_admin
Modify Search Results
HYDERABAD-HYD
CALIFORNIA-CF
Plane Class: FIRST CLASS
Journey Date: 29/Aug/2025 dd-----yyyy
Ticket Type: GENERAL
Trip Mode: ONE WAY
Return Date: dd-----yyyy
MODIFY SEARCH
RESET
CLOSE
Flight Search Results
Flight Number | Flight Name | Origin | Destination | Class | Flight Type | Quota/Ticket Type | Flight Charge | Date | Available Seats
AA12II | RAMESH AIRLINES | HYDERABAD-HYD | CALIFORNIA-CF | FIRST CLASS | AIRBUS A321 | GENERAL | 1050 | 29/Aug/2025 | Check Seat & Book
AA12JJ | RAJITHA AIRLINES | HYDERABAD-HYD | CALIFORNIA-CF | FIRST CLASS | AIRBUS A321 | GENERAL | 1400 | 29/Aug/2025 | Check Seat & Book
AA12OO | RAMESH AIR LINES | HYDERABAD-HYD | CALIFORNIA-CF | FIRST CLASS | BOEING 747 | GENERAL | 1000 | 29/Aug/2025 | Check Seat & Book
Seats Result
Available Seats: 10
Fare Details
Journey Date: (29/Aug/2025), Return Date: (None)
Base Fare: 125000, Class Charges (FIRST CLASS): 1500, Quota Discount (GENERAL): 0, Trip Mode: (ONE WAY), KM: (14500), Trip Discount: (ONE WAY): 0, Total Fare: 126500
SKA AIRLINES offers over 1% discount on the Round Trip Mode to passengers at the same time of booking.
Special category passengers need to carry a photo identity card issued by the government for on-board/off-board verification during the journey. This includes quota/ticket types such as Defence, Govt Employee, Senior Citizen, and Student. Don't forget to carry your identity card!
BOOK
BACK
CLOSE


SKA AIRLINES.
WELCOME ADMIN
AIRLINE RESERVATION NEXT STEI
127.0.0.1:5000/admin/ticket_admin21
AIRLINE RESERVATION NEXT STEPS
While Entering the Passenger Details Those Should need to match with the one of your Governmet issued ID Card
Origin: HYDERABAD-HYD
Departure: 29/Aug/2025
Return: None
Destination: CALIFORNIA-CF
Airplane Number: AA12II
Airplane Name: RAMESH AIRLINES
Flight Type: AIRBUS A321
Class: FIRST CLASS
Quota: GENERAL
Flight Charge: ₹1050.00
Class Charge: ₹1500.00
Quota Discount: ₹0.00
Trip Discount: ₹0.00
Total Fare: ₹127550.00
Passenger Details:
Name: Sai Kiran
Age: 26
Gender: Male
ADULT: 1
CHILD: 0
AVILABLE SEATS:10
AVILABLE SEATS: 10
+Add Person
CONTINUE
BACK


SKA AIRLINES.
WELCOME ADMIN
AIRLINE RESERVATION NEXT STEI
127.0.0.1:5000/admin/ticket_admin31
AIRLINE RESERVATION REVIEW THE DETAILS
Review Journey
Origin: HYDERABAD-HYD
Departure: 29/Aug/2025
Return: None
Destination: CALIFORNIA-CF
Airplane Number: AA12II
Airplane Name: RAMESH AIRLINES
Flight Type: RAMESH AIRLINES
Class: FIRST CLASS
Trip Type: GENERAL
Passenger Details
Passenger Name
Passenger Age
Passenger Gender
SAI KIRAN
26
Male
Fare Summary
Adult: 1 Child: 0 Base Fare: 125000
Class Fare: 1500.00
Quota Discount: 0.00 Trip Mode: ONE WAY
Trip Discount: 0.00 Total Fare: 127550.00 KM: 14500
Plane Booking Admin*: sai
Airline User ID/ Number:*: None
User Email:*: useremail@host.com
SKA AIRLINES offers over 1% discount on the Round Trip Mode to passengers at the same time of booking.
Special category passengers need to carry a photo identity card issued by the government for on-board/off-board verification during the journey. This includes quota/ticket types such as Defence, Govt Employee, Senior Citizen, and Student. Don't forget to carry your identity card!
CONTINUE
BACK


SKA AIRLINES.
WELCOME ADMIN
Airline Reservation/Boarding Suc
127.0.0.1:5000/admin/ticket_admin_success
Origin
HYDERABAD-HYD
Destination
CALIFORNIA-CF
Journey Details
PNR Number : 132
Departure : 29/Aug/2025
Return : None
Booking Date : 20/Aug/2025 02:12:45 PM
Airplane Number : AA12II
Airplane Name : RAMESH AIRLINES
Class : FIRST CLASS
Trip Type : GENERAL
Passenger Details
Passenger Name
Age
Gender
Gate
Current Status
Booking Status
SAI KIRAN
26
Male
Check Monitors
CNF/FIRST CLASS/1A
CNF/FIRST CLASS/1A
Fare Summary
Adult: 1 ;Child: 0
Flight Type : AIRBUS A321
Base Fare: 125000
Flight Fare 1050.00
Class Fare: 1500.00
Quota Discount: 0.00
Trip Mode: ONE WAY
Trip Discount: 0.00
Total Fare: 127550.00
KM: 14500, Journey Status: IP
Boarding QR Code
Scan at boarding gate


1 of 2,675
SKA Airlines Ticket Confirmation | PNR 132
Inbox
flyskaairlines.com@gmail.com
to me
2:12 PM (0 minutes ago)
SKA Airlines Flight Ticket Booking Confirmation
PNR : 132
Flight : AA12II - RAMESH AIRLINES
Origin : HYDERABAD-HYD
Destination : CALIFORNIA-CF
Passengers : SAI KIRAN | Gender: Male | Age: 26 | Seat: 1A | ACTIVE
Journey Date : 29/Aug/2025
Return Date : None
Flight Type : AIRBUS A321
Class : FIRST CLASS
Quota/Trip Type : GENERAL
Trip Mode : ONE WAY
Total Fare : ₹127550.00
Scan the QR Code to view your ticket and travel details:

## 2. Registered User Module 🧳✨

The Registered User Module provides customers with a secure and interactive platform to manage their flight bookings, personal profiles, payments, and feedback efficiently. It ensures a seamless and personalized experience tailored to each user's needs. 🌐🛫

**Key Features & Functionalities** 🛠️

Secure Authentication 🔐

**User Registration & Login:** Secure sign-up and login with unique credentials (username, password).

Logout: Ensures safe termination of sessions for privacy.

**Flight Booking & Management** ✈️

Flight Search: Quick searches based on destination, dates, and preferences.

Ticket Booking: Easy booking with real-time seat availability and fare display.

**Multiple Payment Modes:**

Bank Account Payment: Users can pay through their bank account IDs (ACID).

QR Code Payment: Convenient QR-based payment integration for fast processing.

Ticket Cancellation: Ability to cancel bookings online subject to airline policies.

**Ticket Status & Notifications** 📧

After booking, ticket status and e-ticket details are sent to users via email, keeping them informed throughout the journey. 📨✈️

**Payment Processing** 💳

Integrated Payment Gateway: Supports secure transactions via bank accounts and QR code payments (payment_gateway_bp, qr_payment_bp).

Transaction Security: Ensures encrypted and validated payments protecting financial data.

**Profile & Account Management** 👤

Update Personal Info: Users can update mobile numbers (user_update_mbl_num_bp) and passwords (user_update_password_bp).

Journey History: Detailed access to past and upcoming bookings (user_journey_history_bp).

**User Feedback & Support** 📨

Users can provide valuable feedback to improve service quality (user_feedback_bp).

**Ticket Handling** 🎟️

E-ticket Access: Download, print, or view e-tickets at any time (ticket_user_bp).

Cancellation Requests: Manage cancellations and track status (user_can_ticket_bp).

and e-ticket details are sent to users via email, keeping them informed throughout the journey. 📨✈️

**User Experience & Interface** 🎨

The module features intuitive HTML templates for login, dashboards, booking, payment, feedback, and profile management pages. The design is fully responsive for desktop and mobile devices, ensuring smooth user experience everywhere. 📱💻

**Security & Data Integrity** 🔒

Strong user authentication, encrypted communication, and session management protect privacy and ensure integrity of user data.

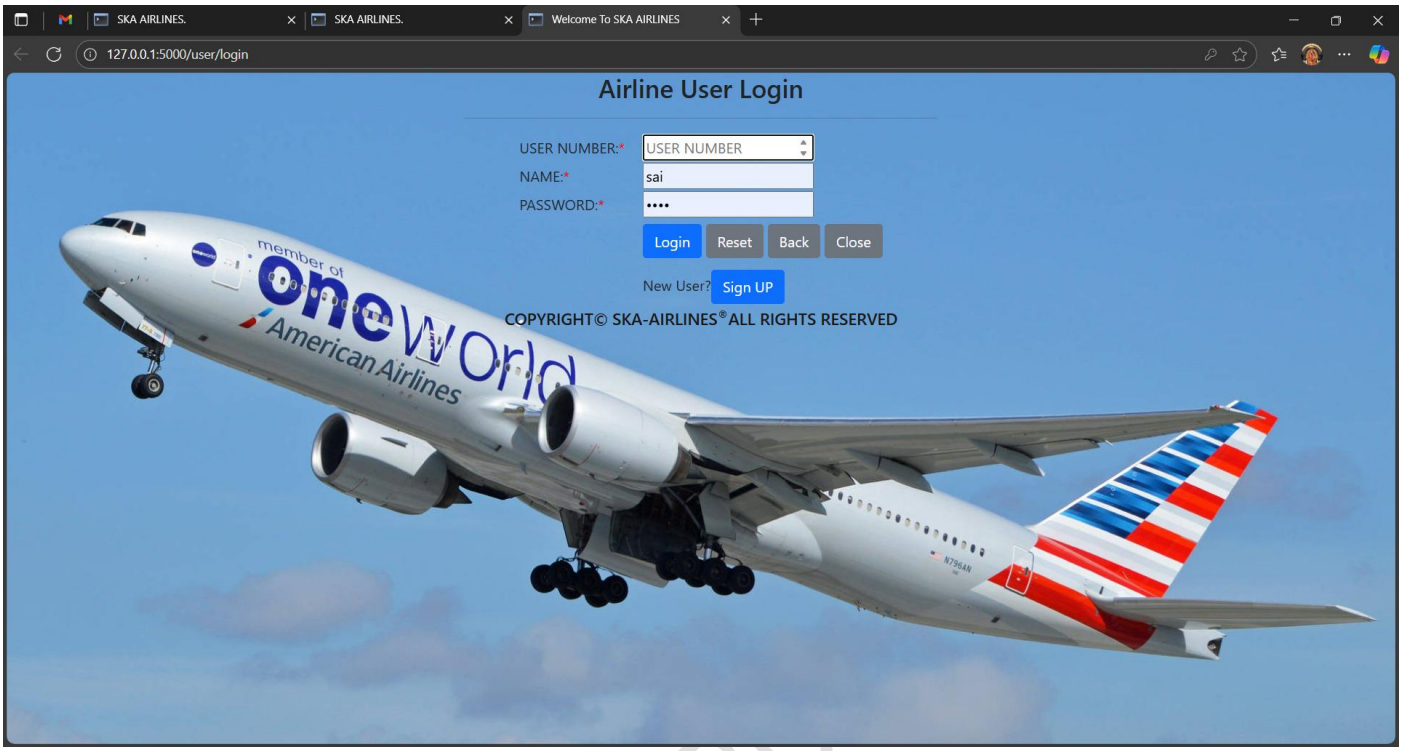
**Summary** 🌟

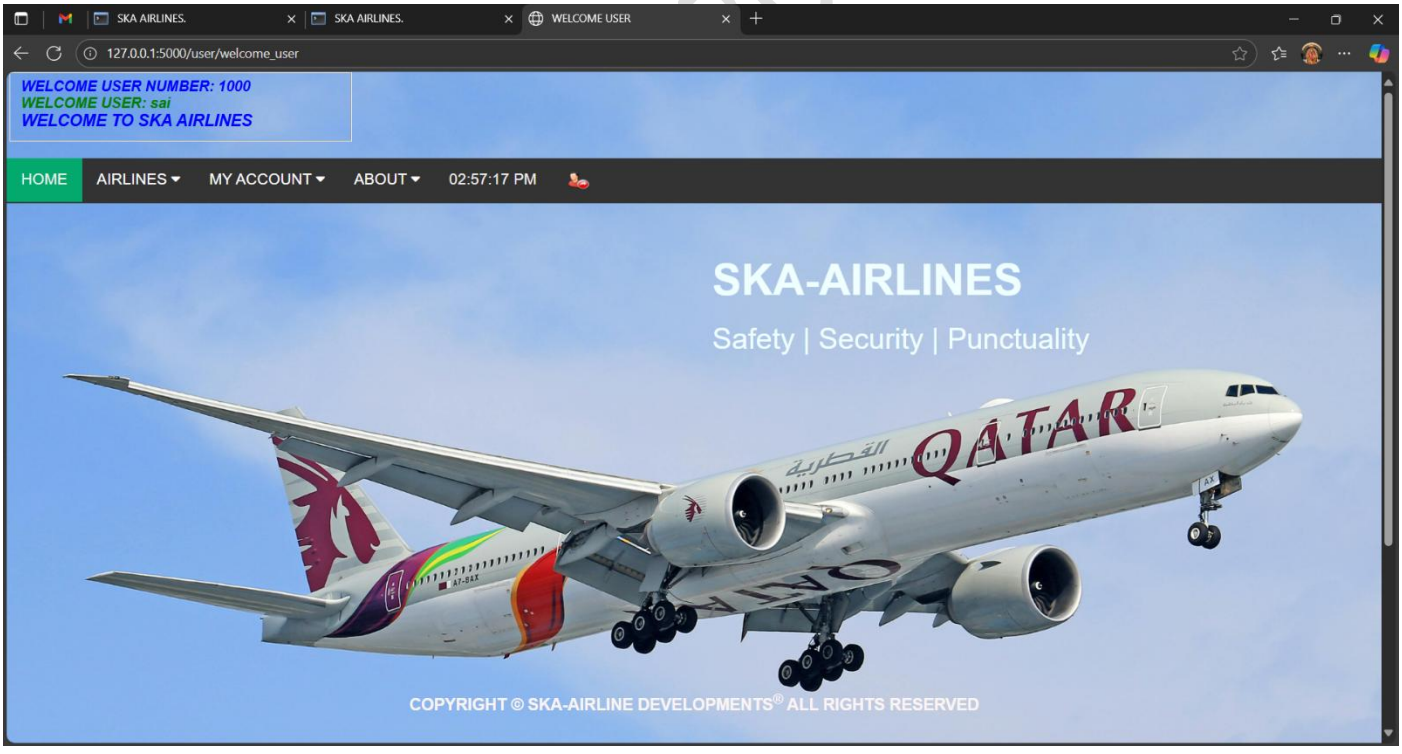
The Registered User Module offers a secure, flexible, and user-friendly environment with multiple payment options, ticket status email notifications, and comprehensive booking controls. This fosters confidence and satisfaction among air travellers. 🧳✈️💬

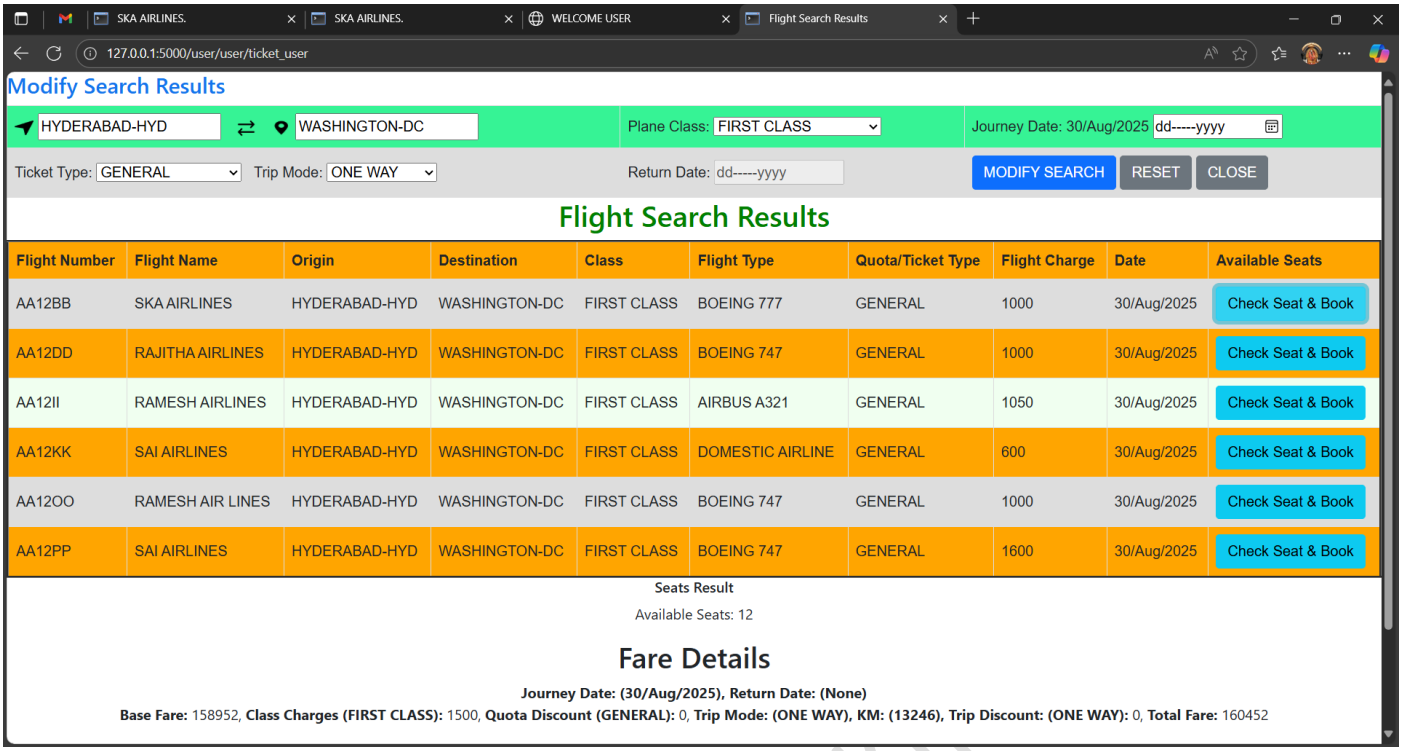
**Flight Registered User Module** ✈️

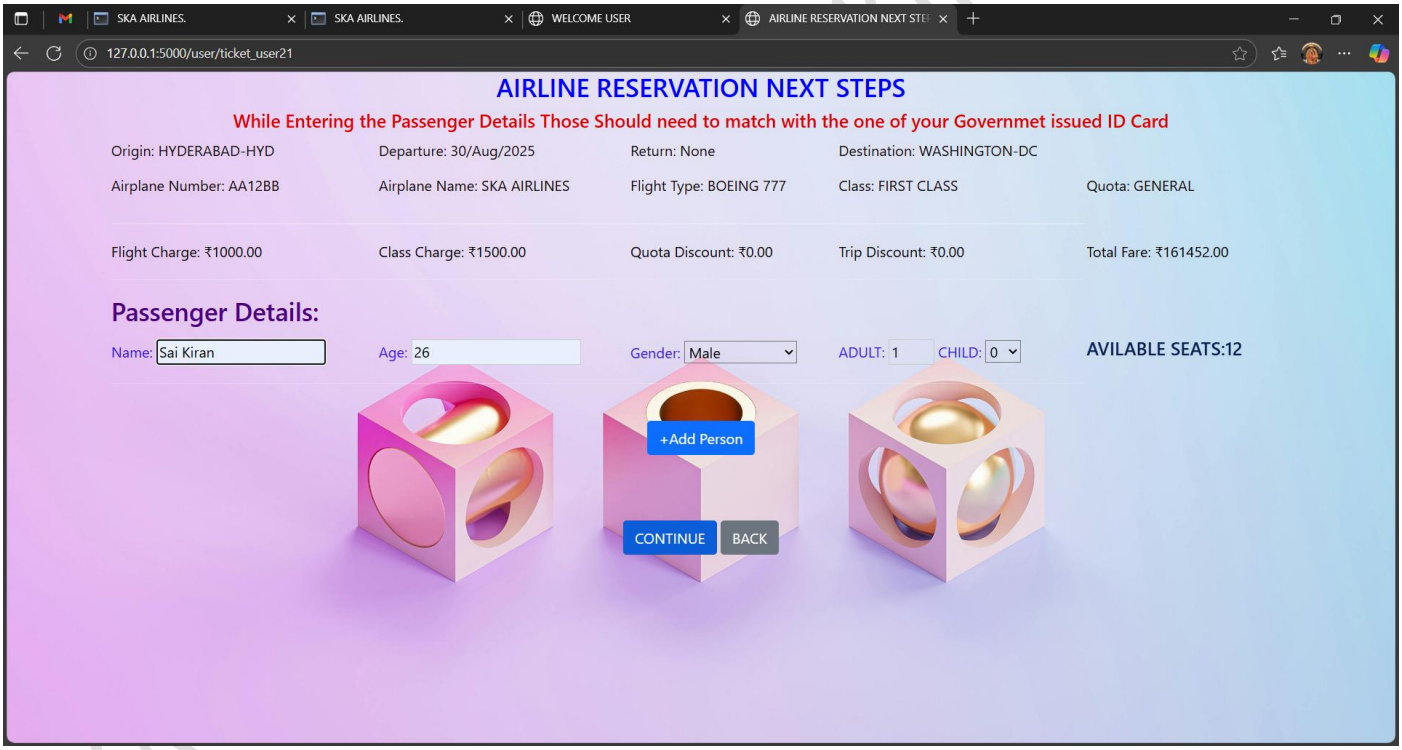
```
├── User Section
|   ├── user_login_bp
|   ├── user_logout_bp
|   ├── user_dashboard_bp
|   └── user_bp
|  user_routes.py -----> Handles:|
|     ├── ticket_user_bp          --> ticket_user.html, ticket_user1.html, ...
|     ├── payment_gateway_bp      --> payment_gateway_su.html
|     ├── ticket_userdb_bp
|     ├── user_can_ticket_bp      --> user_can_ticket.html, user_can_ticket1.html, ...
|     ├── user_journey_history_bp --> user_journey_history_su.html, ...
|     ├── user_feedback_bp        --> user_feedback.html, user_feedback_su.html
|     ├── user_update_mbl_num_bp  --> user_update_mbl_num.html, ...
|     ├── user_update_password_bp --> user_update_password.html, ...
|     └── qr_payment_bp           --> qr_payment_su.html
```

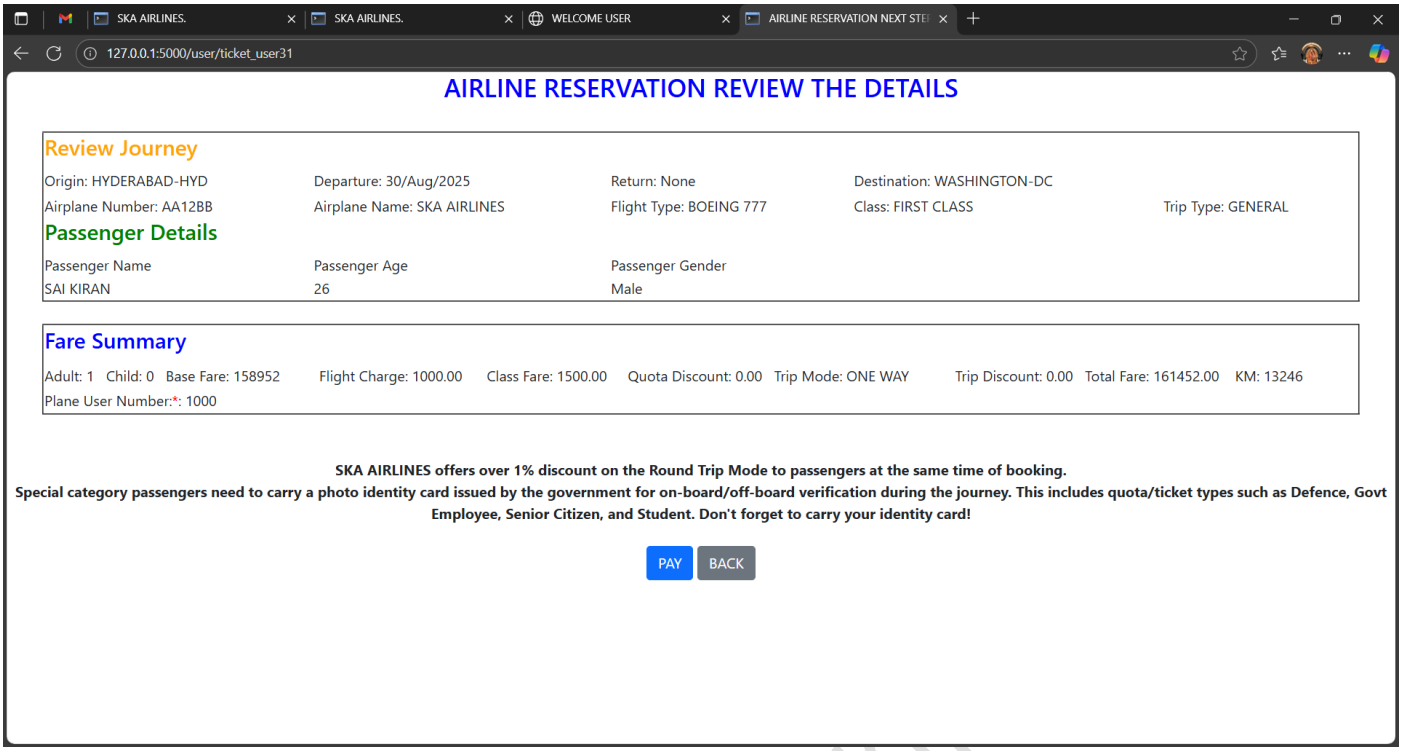
## Flight Registered User Module Screenshots

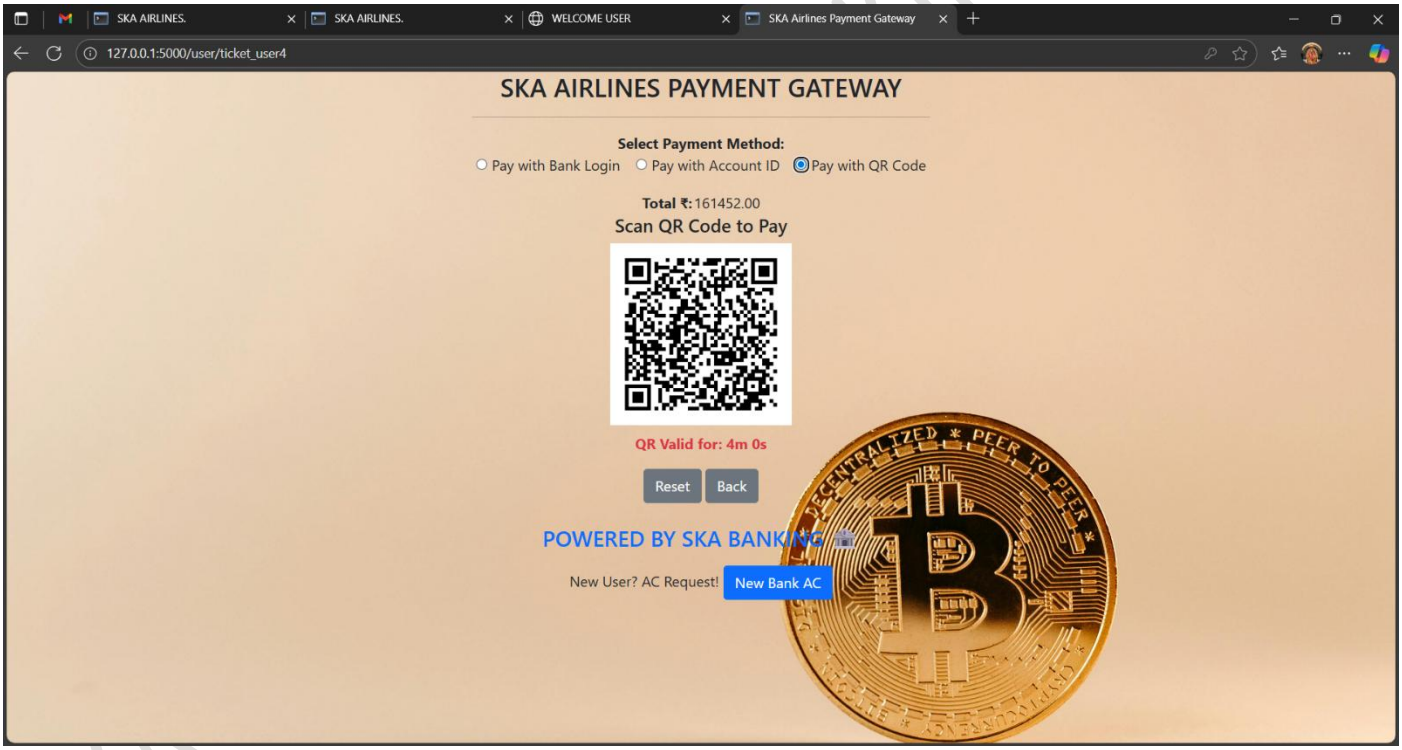


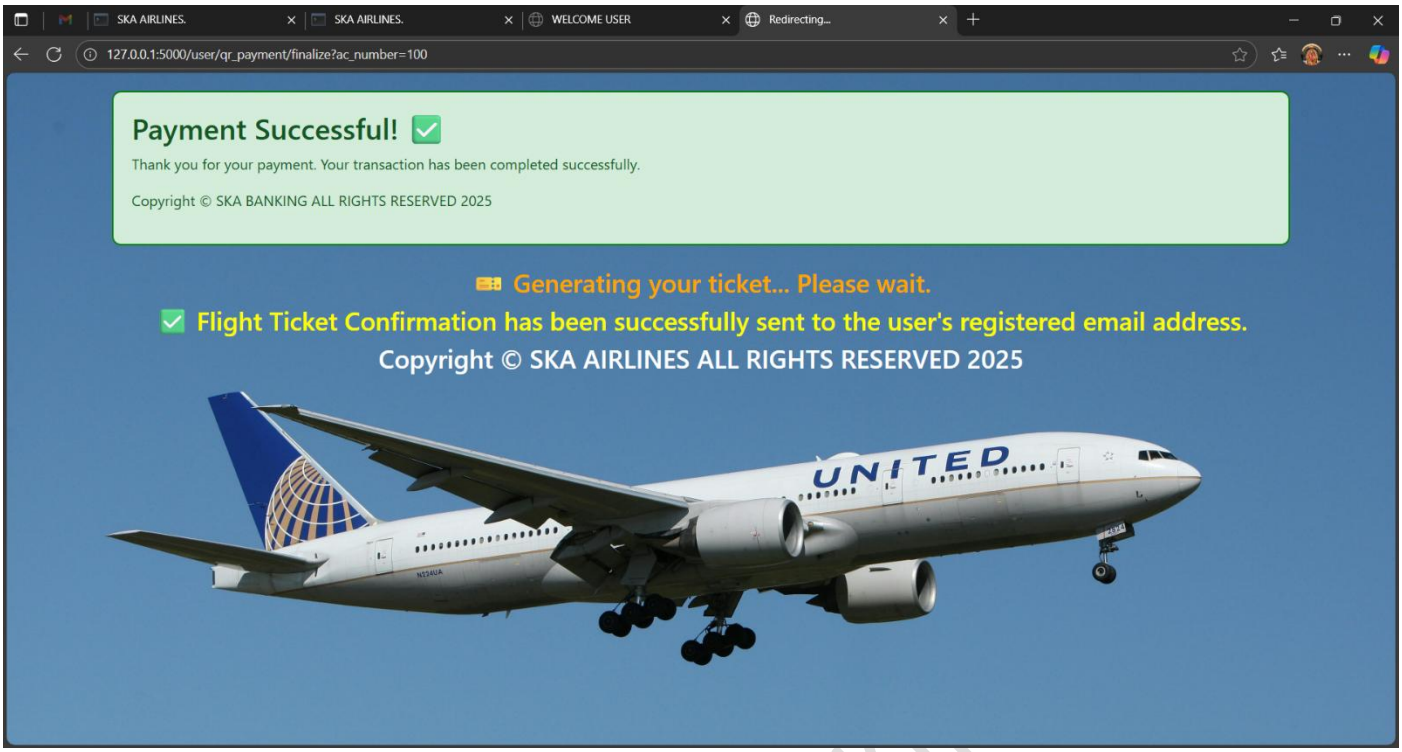



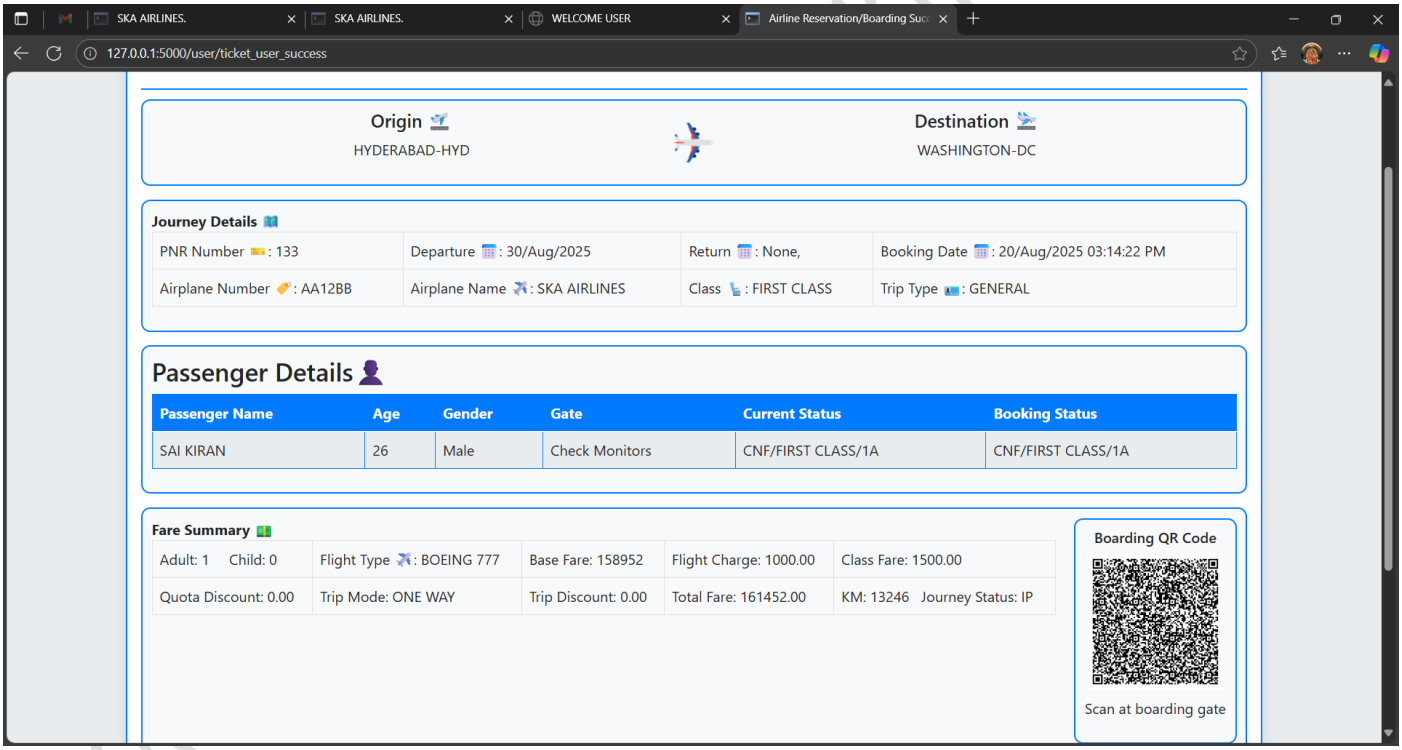

SKA AIRLINES.
SKA AIRLINES.
WELCOME USER
Flight Search Results
127.0.0.1:5000/user/user/ticket_user
Modify Search Results
HYDERABAD-HYD
WASHINGTON-DC
Plane Class: FIRST CLASS
Journey Date: 30/Aug/2025 dd-----yyyy
Ticket Type: GENERAL
Trip Mode: ONE WAY
Return Date: dd-----yyyy
MODIFY SEARCH
RESET
CLOSE
Flight Search Results
Flight Number | Flight Name | Origin | Destination | Class | Flight Type | Quota/Ticket Type | Flight Charge | Date | Available Seats
AA12BB | SKA AIRLINES | HYDERABAD-HYD | WASHINGTON-DC | FIRST CLASS | BOEING 777 | GENERAL | 1000 | 30/Aug/2025 | Check Seat & Book
AA12DD | RAJITHA AIRLINES | HYDERABAD-HYD | WASHINGTON-DC | FIRST CLASS | BOEING 747 | GENERAL | 1000 | 30/Aug/2025 | Check Seat & Book
AA12II | RAMESH AIRLINES | HYDERABAD-HYD | WASHINGTON-DC | FIRST CLASS | AIRBUS A321 | GENERAL | 1050 | 30/Aug/2025 | Check Seat & Book
AA12KK | SAI AIRLINES | HYDERABAD-HYD | WASHINGTON-DC | FIRST CLASS | DOMESTIC AIRLINE | GENERAL | 600 | 30/Aug/2025 | Check Seat & Book
AA12OO | RAMESH AIR LINES | HYDERABAD-HYD | WASHINGTON-DC | FIRST CLASS | BOEING 747 | GENERAL | 1000 | 30/Aug/2025 | Check Seat & Book
AA12PP | SAI AIRLINES | HYDERABAD-HYD | WASHINGTON-DC | FIRST CLASS | BOEING 747 | GENERAL | 1600 | 30/Aug/2025 | Check Seat & Book
Seats Result
Available Seats: 12
Fare Details
Journey Date: (30/Aug/2025), Return Date: (None)
Base Fare: 158952, Class Charges (FIRST CLASS): 1500, Quota Discount (GENERAL): 0, Trip Mode: (ONE WAY), KM: (13246), Trip Discount: (ONE WAY): 0, Total Fare: 160452

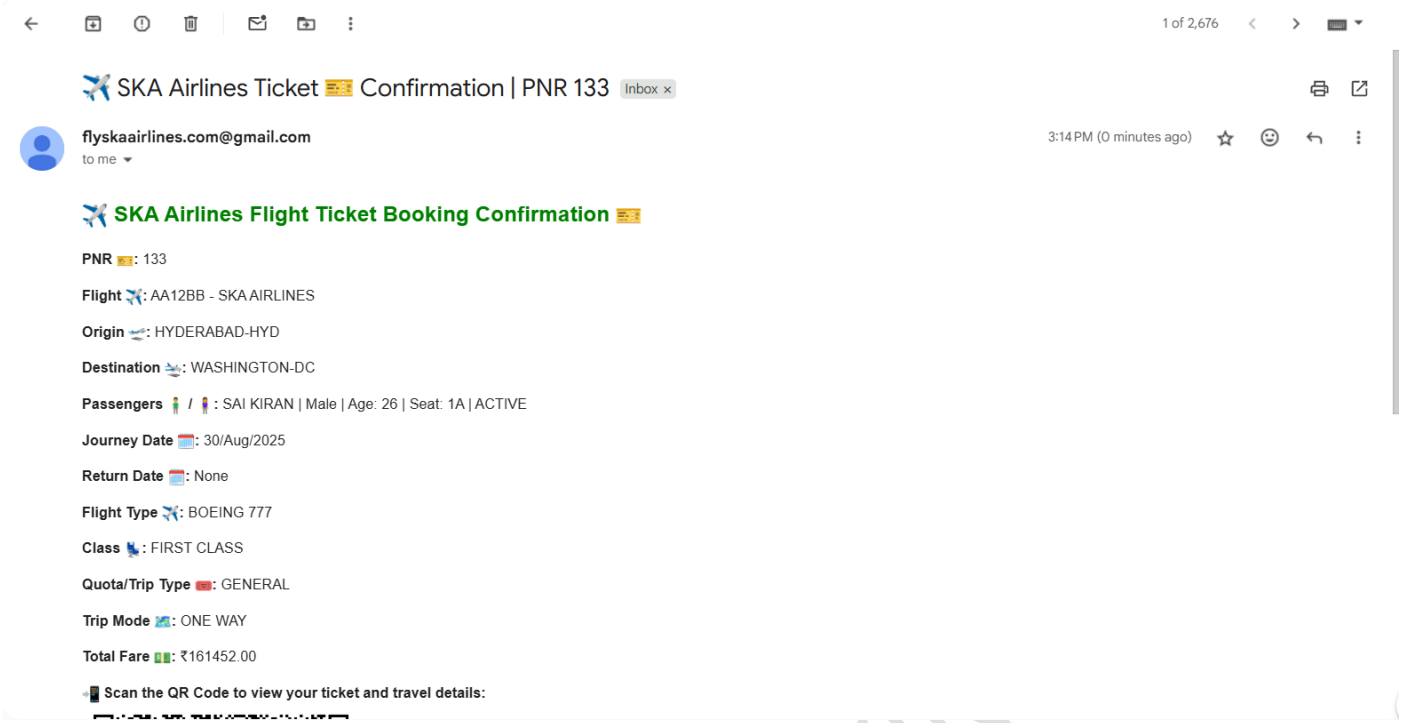

SKA AIRLINES.
SKA AIRLINES.
WELCOME USER
AIRLINE RESERVATION NEXT STEP
127.0.0.1:5000/user/ticket_user21
AIRLINE RESERVATION NEXT STEPS
While Entering the Passenger Details Those Should need to match with the one of your Governmet issued ID Card
Origin: HYDERABAD-HYD
Departure: 30/Aug/2025
Return: None
Destination: WASHINGTON-DC
Airplane Number: AA12BB
Airplane Name: SKA AIRLINES
Flight Type: BOEING 777
Class: FIRST CLASS
Quota: GENERAL
Flight Charge: ₹1000.00
Class Charge: ₹1500.00
Quota Discount: ₹0.00
Trip Discount: ₹0.00
Total Fare: ₹161452.00
Passenger Details:
Name: Sai Kiran
Age: 26
Gender: Male
ADULT: 1
CHILD: 0
AVILABLE SEATS:12
+Add Person
CONTINUE
BACK


SKA AIRLINES.
SKA AIRLINES.
WELCOME USER
AIRLINE RESERVATION NEXT STE
127.0.0.1:5000/user/ticket_user31
AIRLINE RESERVATION REVIEW THE DETAILS
Review Journey
Origin: HYDERABAD-HYD
Departure: 30/Aug/2025
Return: None
Destination: WASHINGTON-DC
Airplane Number: AA12BB
Airplane Name: SKA AIRLINES
Flight Type: BOEING 777
Class: FIRST CLASS
Trip Type: GENERAL
Passenger Details
Passenger Name
Passenger Age
Passenger Gender
SAI KIRAN
26
Male
Fare Summary
Adult: 1 Child: 0 Base Fare: 158952
Flight Charge: 1000.00
Class Fare: 1500.00
Quota Discount: 0.00
Trip Mode: ONE WAY
Trip Discount: 0.00
Total Fare: 161452.00
KM: 13246
Plane User Number:*: 1000
SKA AIRLINES offers over 1% discount on the Round Trip Mode to passengers at the same time of booking.
Special category passengers need to carry a photo identity card issued by the government for on-board/off-board verification during the journey. This includes quota/ticket types such as Defence, Govt Employee, Senior Citizen, and Student. Don't forget to carry your identity card!
PAY
BACK


SKA AIRLINES.
SKA AIRLINES.
WELCOME USER
SKA Airlines Payment Gateway
127.0.0.1:5000/user/ticket_user4
SKA AIRLINES PAYMENT GATEWAY
Select Payment Method:
Pay with Bank Login
Pay with Account ID
Pay with QR Code
Total ₹:161452.00
Scan QR Code to Pay
QR Valid for: 4m 0s
Reset
Back
POWERED BY SKA BANKING
New User? AC Request!
New Bank AC


SKA AIRLINES.
SKA AIRLINES.
WELCOME USER
Redirecting...
127.0.0.1:5000/user/qr_payment/finalize?ac_number=100
Payment Successful!
Thank you for your payment. Your transaction has been completed successfully.
Copyright © SKA BANKING ALL RIGHTS RESERVED 2025
Generating your ticket... Please wait.
Flight Ticket Confirmation has been successfully sent to the user's registered email address.
Copyright © SKA AIRLINES ALL RIGHTS RESERVED 2025
UNITED


SKA AIRLINES.
SKA AIRLINES.
WELCOME USER
Airline Reservation/Boarding Suc
127.0.0.1:5000/user/ticket_user_success
Origin
HYDERABAD-HYD
Destination
WASHINGTON-DC
Journey Details
PNR Number : 133
Departure : 30/Aug/2025
Return : None,
Booking Date : 20/Aug/2025 03:14:22 PM
Airplane Number : AA12BB
Airplane Name : SKA AIRLINES
Class : FIRST CLASS
Trip Type : GENERAL
Passenger Details
Passenger Name
Age
Gender
Gate
Current Status
Booking Status
SAI KIRAN
26
Male
Check Monitors
CNF/FIRST CLASS/1A
CNF/FIRST CLASS/1A
Fare Summary
Adult: 1 Child: 0
Flight Type : BOEING 777
Base Fare: 158952
Flight Charge: 1000.00
Class Fare: 1500.00
Quota Discount: 0.00
Trip Mode: ONE WAY
Trip Discount: 0.00
Total Fare: 161452.00
KM: 13246 Journey Status: IP
Boarding QR Code
Scan at boarding gate



✈️ SKA Airlines Ticket 🎫 Confirmation | PNR 133 Inbox ×

flyskaairlines.com@gmail.com
to me

3:14 PM (0 minutes ago)

## ✈️ SKA Airlines Flight Ticket Booking Confirmation 🎫

**PNR** 🎫: 133

**Flight** ✈️: AA12BB - SKA AIRLINES

**Origin** 🛫: HYDERABAD-HYD

**Destination** 🛬: WASHINGTON-DC

**Passengers** 🧍 / 🧍: SAI KIRAN | Male | Age: 26 | Seat: 1A | ACTIVE

**Journey Date** 🗓️: 30/Aug/2025

**Return Date** 🗓️: None

**Flight Type** ✈️: BOEING 777

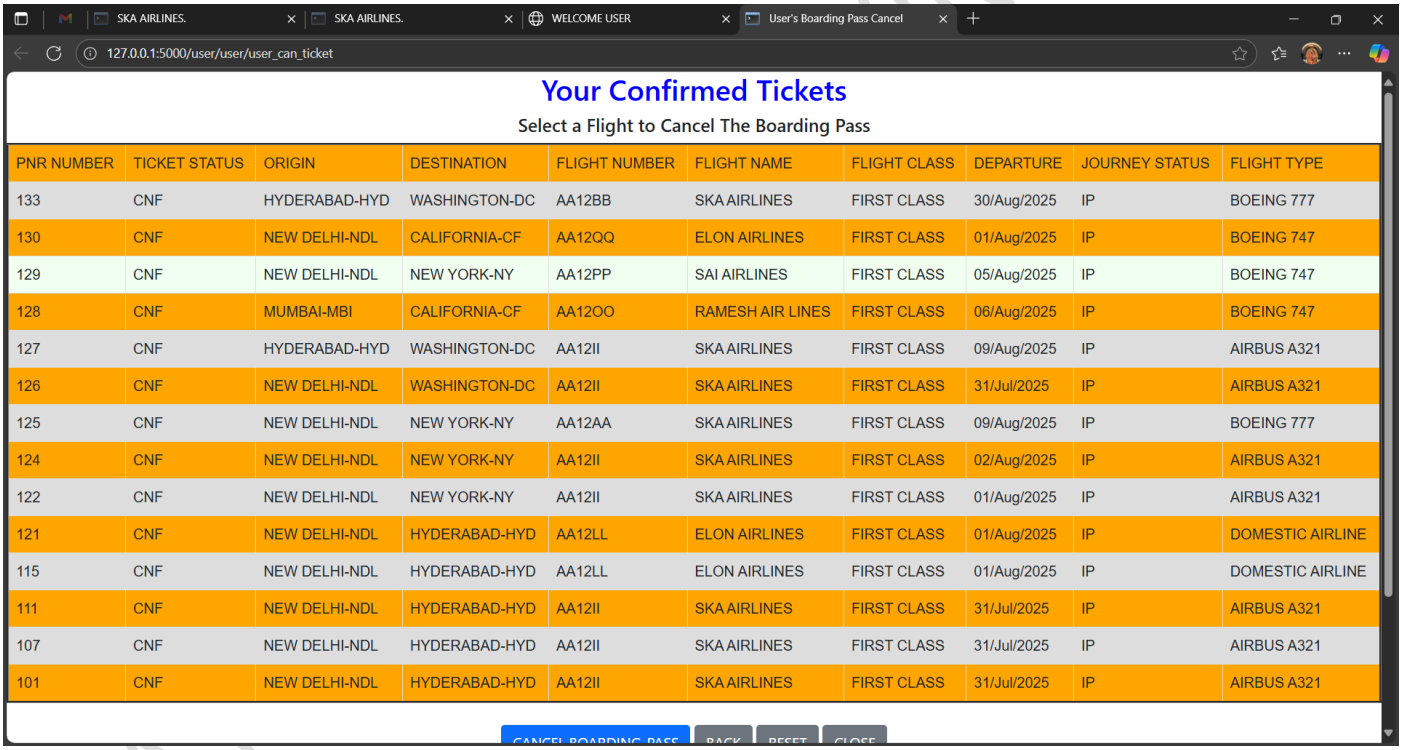
**Class** 💺: FIRST CLASS

**Quota/Trip Type** 🎟️: GENERAL

**Trip Mode** 🗺️: ONE WAY

**Total Fare** 💵: ₹161452.00

📲 **Scan the QR Code to view your ticket and travel details:**

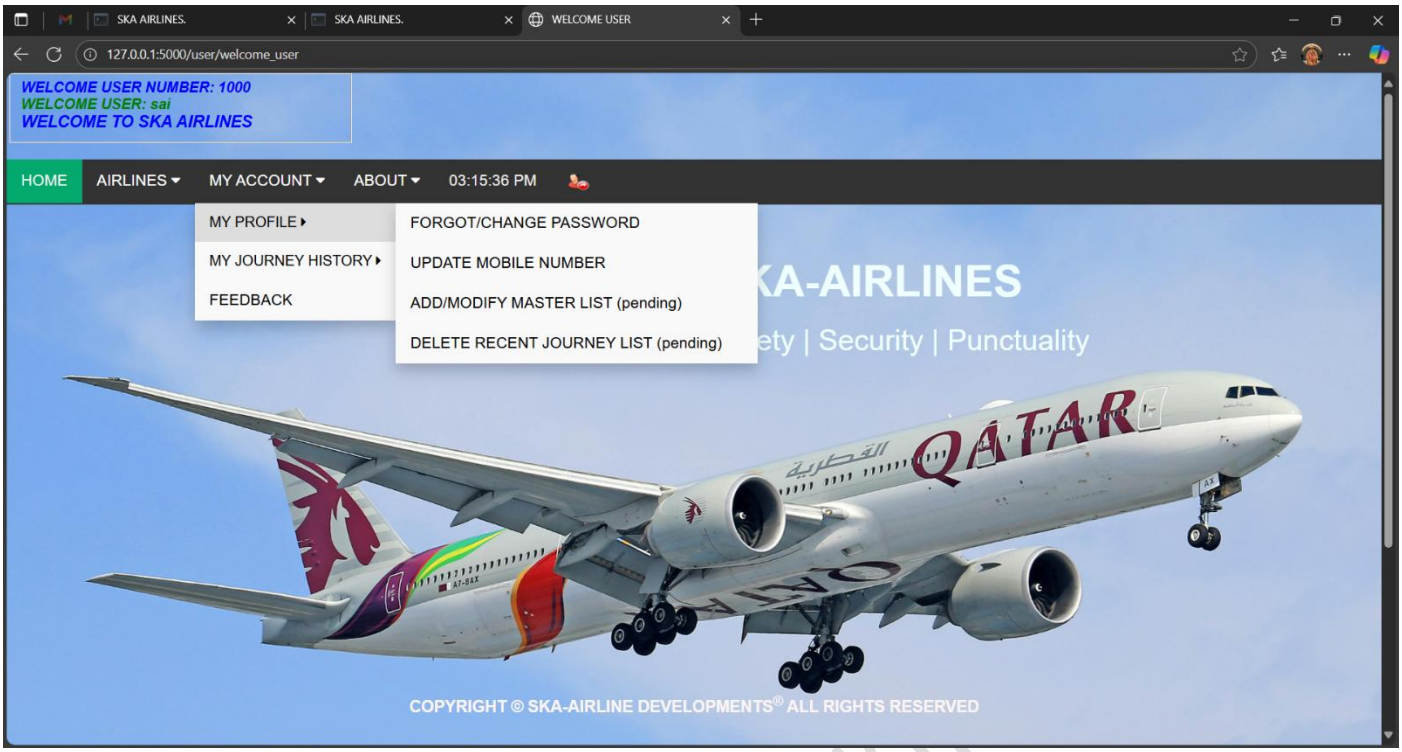


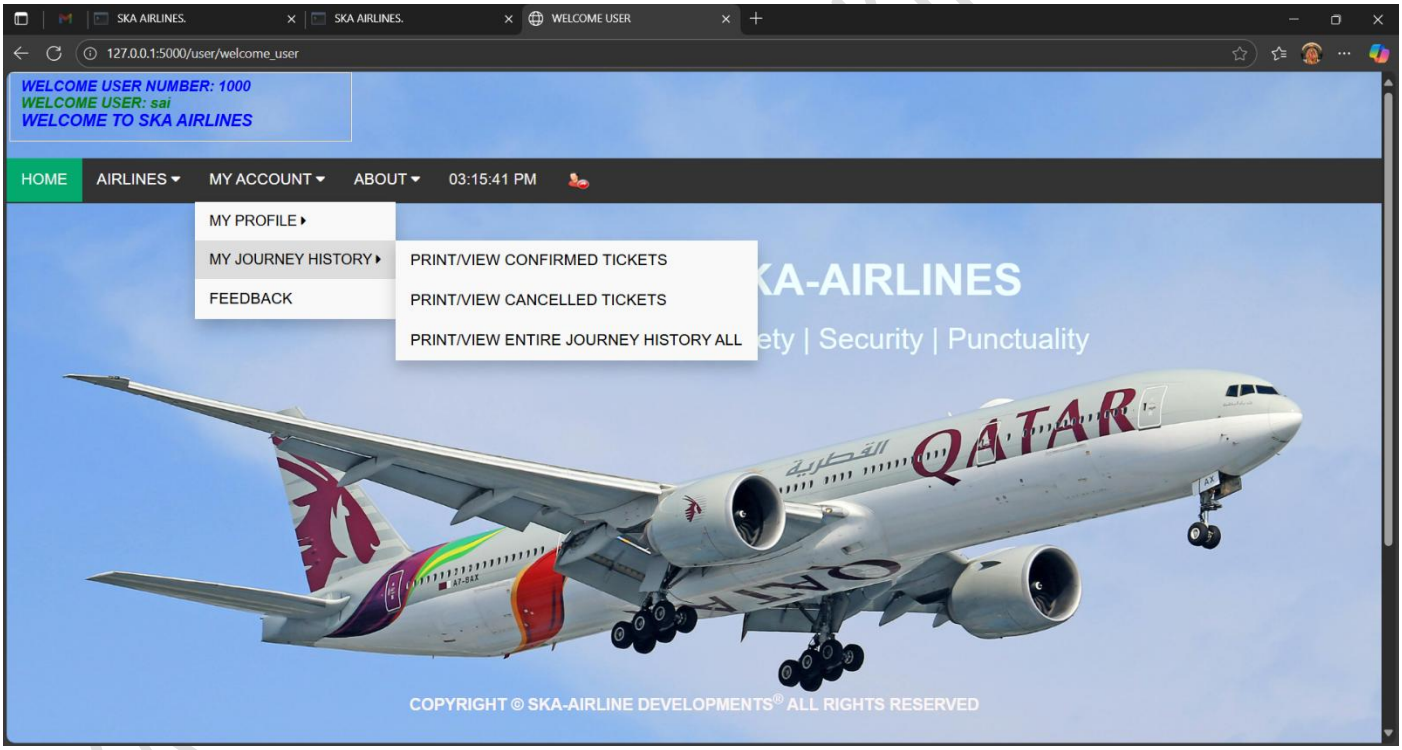

## Your Confirmed Tickets

Select a Flight to Cancel The Boarding Pass

| PNR NUMBER | TICKET STATUS | ORIGIN | DESTINATION | FLIGHT NUMBER | FLIGHT NAME | FLIGHT CLASS | DEPARTURE | JOURNEY STATUS | FLIGHT TYPE |
|---|---|---|---|---|---|---|---|---|---|
| 133 | CNF | HYDERABAD-HYD | WASHINGTON-DC | AA12BB | SKA AIRLINES | FIRST CLASS | 30/Aug/2025 | IP | BOEING 777 |
| 130 | CNF | NEW DELHI-NDL | CALIFORNIA-CF | AA12QQ | ELON AIRLINES | FIRST CLASS | 01/Aug/2025 | IP | BOEING 747 |
| 129 | CNF | NEW DELHI-NDL | NEW YORK-NY | AA12PP | SAI AIRLINES | FIRST CLASS | 05/Aug/2025 | IP | BOEING 747 |
| 128 | CNF | MUMBAI-MBI | CALIFORNIA-CF | AA12OO | RAMESH AIR LINES | FIRST CLASS | 06/Aug/2025 | IP | BOEING 747 |
| 127 | CNF | HYDERABAD-HYD | WASHINGTON-DC | AA12II | SKA AIRLINES | FIRST CLASS | 09/Aug/2025 | IP | AIRBUS A321 |
| 126 | CNF | NEW DELHI-NDL | WASHINGTON-DC | AA12II | SKA AIRLINES | FIRST CLASS | 31/Jul/2025 | IP | AIRBUS A321 |
| 125 | CNF | NEW DELHI-NDL | NEW YORK-NY | AA12AA | SKA AIRLINES | FIRST CLASS | 09/Aug/2025 | IP | BOEING 777 |
| 124 | CNF | NEW DELHI-NDL | NEW YORK-NY | AA12II | SKA AIRLINES | FIRST CLASS | 02/Aug/2025 | IP | AIRBUS A321 |
| 122 | CNF | NEW DELHI-NDL | NEW YORK-NY | AA12II | SKA AIRLINES | FIRST CLASS | 01/Aug/2025 | IP | AIRBUS A321 |
| 121 | CNF | NEW DELHI-NDL | HYDERABAD-HYD | AA12LL | ELON AIRLINES | FIRST CLASS | 01/Aug/2025 | IP | DOMESTIC AIRLINE |
| 115 | CNF | NEW DELHI-NDL | HYDERABAD-HYD | AA12LL | ELON AIRLINES | FIRST CLASS | 01/Aug/2025 | IP | DOMESTIC AIRLINE |
| 111 | CNF | NEW DELHI-NDL | HYDERABAD-HYD | AA12II | SKA AIRLINES | FIRST CLASS | 31/Jul/2025 | IP | AIRBUS A321 |
| 107 | CNF | NEW DELHI-NDL | HYDERABAD-HYD | AA12II | SKA AIRLINES | FIRST CLASS | 31/Jul/2025 | IP | AIRBUS A321 |
| 101 | CNF | NEW DELHI-NDL | HYDERABAD-HYD | AA12II | SKA AIRLINES | FIRST CLASS | 31/Jul/2025 | IP | AIRBUS A321 |

CANCEL BOARDING PASS BACK RESET CLOSE


SKA AIRLINES.
SKA AIRLINES.
WELCOME USER
127.0.0.1:5000/user/welcome_user
WELCOME USER NUMBER: 1000
WELCOME USER: sai
WELCOME TO SKA AIRLINES
HOME
AIRLINES
MY ACCOUNT
ABOUT
03:15:36 PM
MY PROFILE
MY JOURNEY HISTORY
FEEDBACK
FORGOT/CHANGE PASSWORD
UPDATE MOBILE NUMBER
ADD/MODIFY MASTER LIST (pending)
DELETE RECENT JOURNEY LIST (pending)
KA-AIRLINES
ety | Security | Punctuality
QATAR
COPYRIGHT © SKA-AIRLINE DEVELOPMENTS® ALL RIGHTS RESERVED


SKA AIRLINES.
SKA AIRLINES.
WELCOME USER
127.0.0.1:5000/user/welcome_user
WELCOME USER NUMBER: 1000
WELCOME USER: sai
WELCOME TO SKA AIRLINES
HOME
AIRLINES
MY ACCOUNT
ABOUT
03:15:41 PM
MY PROFILE
MY JOURNEY HISTORY
FEEDBACK
PRINT/VIEW CONFIRMED TICKETS
PRINT/VIEW CANCELLED TICKETS
PRINT/VIEW ENTIRE JOURNEY HISTORY ALL
KA-AIRLINES
ety | Security | Punctuality
QATAR
COPYRIGHT © SKA-AIRLINE DEVELOPMENTS® ALL RIGHTS RESERVED

## 3. New User Registration Module 📝✨

The New User Registration Module provides a simple, secure, and efficient way for new users to join the SKA Flight Ticket Booking System. This module is designed to onboard new customers smoothly, enabling them to become registered users and unlock all the benefits of the platform. 🌐🚀

**Key Features & Functionalities** 🛠️

User-Friendly Registration Form: New users fill out a comprehensive yet easy-to-use registration form to create their accounts. The form collects essential personal information necessary for account setup. 📝🖱️

Data Validation & Verification: The entered information undergoes thorough validation checks to ensure accuracy and prevent duplicate accounts. This includes email verification, mobile number checks, and secure password criteria enforcement. ✔️🔒

Secure Account Creation: Upon successful validation, accounts are securely created and users receive confirmation notifications, often via email, to activate their profiles. 📧✅

Access to Full User Functionalities: Registered users can immediately access all features such as flight booking, payment options, journey history, and profile management, making their transition from guests seamless and efficient. 🔓✈️

Privacy & Security: Personal data is handled with confidentiality, and secure protocols protect user information during registration.

**User Experience & Interface** 🎨

The registration page is designed with clean, intuitive layouts compatible with desktops and mobile devices alike, ensuring a smooth and hassle-free signup experience. Form validation provides instant feedback to users for a user-friendly onboarding process. 📱💻

**Summary** 🌟

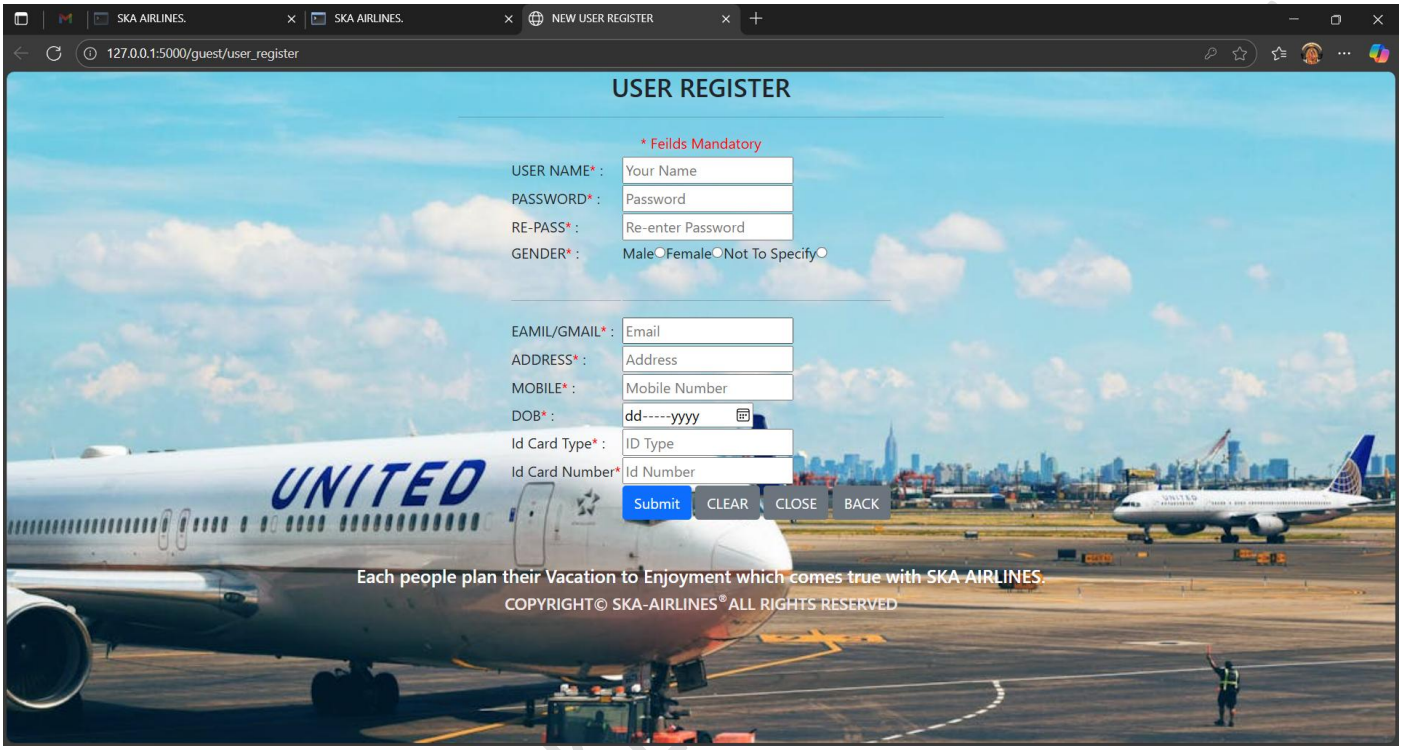
The New User Registration Module is a gateway to the SKA Flight Ticket Booking System, transforming visitors into valued registered customers through a secure, quick, and convenient signup process. This module lays the foundation for personalized and efficient flight booking experiences. 📝✈️✨

## Flight New-User/Register/Sign-Up Module

| guest_routes.py -----> Handles:

| ├— user_register_bp --> user_register.html, register_success.html

## Flight New-User/ Sign-Up Module Screenshots



# 4. Guest User Module 👥

The Guest User Module caters to users who have not registered or logged in but still want to interact with the SKA Flight Ticket Booking System. This module provides essential access and functionalities to non-registered visitors, ensuring they can explore the system and its services before deciding to sign up. 🌐 🚶 ♂️

**Key Features & Functionalities** 🛠️

View Flight Schedules & Availability: Guests can browse available flights, schedules, and routes without needing an account, helping them plan their travel efficiently. 🗓️ ✈️

PNR Status Checking: Visitors can check their Passenger Name Record (PNR) status to track bookings made previously without logging in, providing convenient access to important information. 🔍 🎫

User Registration Access: Guests are guided to create new accounts easily via clear prompts and links, facilitating smooth transition to registered user status. 📝 ➡️ 🧳

View News & Events: Access to the latest airline news, announcements, and special events keeps guests informed and engaged. 📰 📣

Submit Anonymous Feedback: Guests can provide feedback or report issues anonymously, helping improve the service without the need for registration. 💬 🔒

**User Experience & Interface** 🎨

The Guest Module features clean, navigable pages designed to encourage exploration and easy access to information. Responsive design ensures seamless browsing across devices, including desktops, tablets, and smartphones. 📱 💻

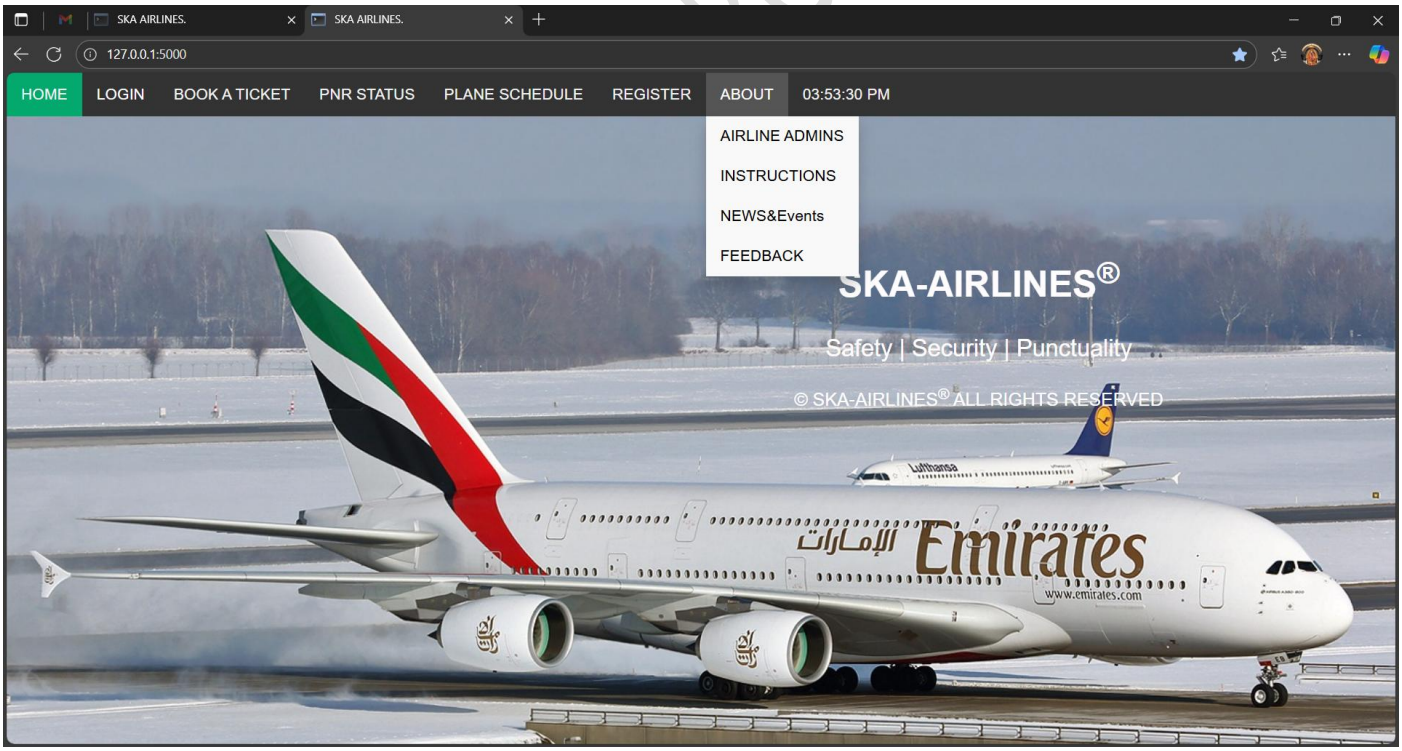
**Summary** 🌟

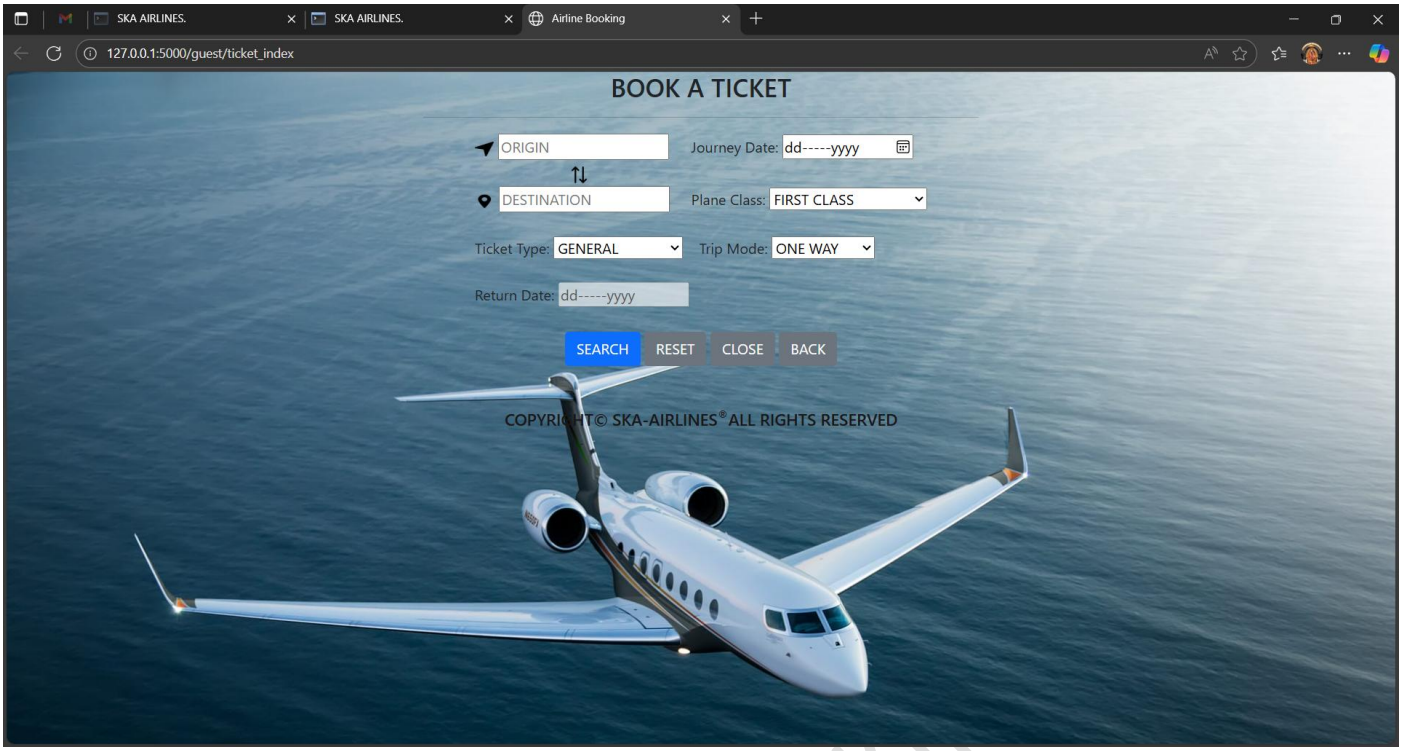
The Guest User Module offers a welcoming and informative gateway to the SKA Flight Ticket Booking System, allowing visitors to access essential travel information and interact anonymously before registering. This fosters engagement and builds trust with potential users. 👥 ✈️ ✨

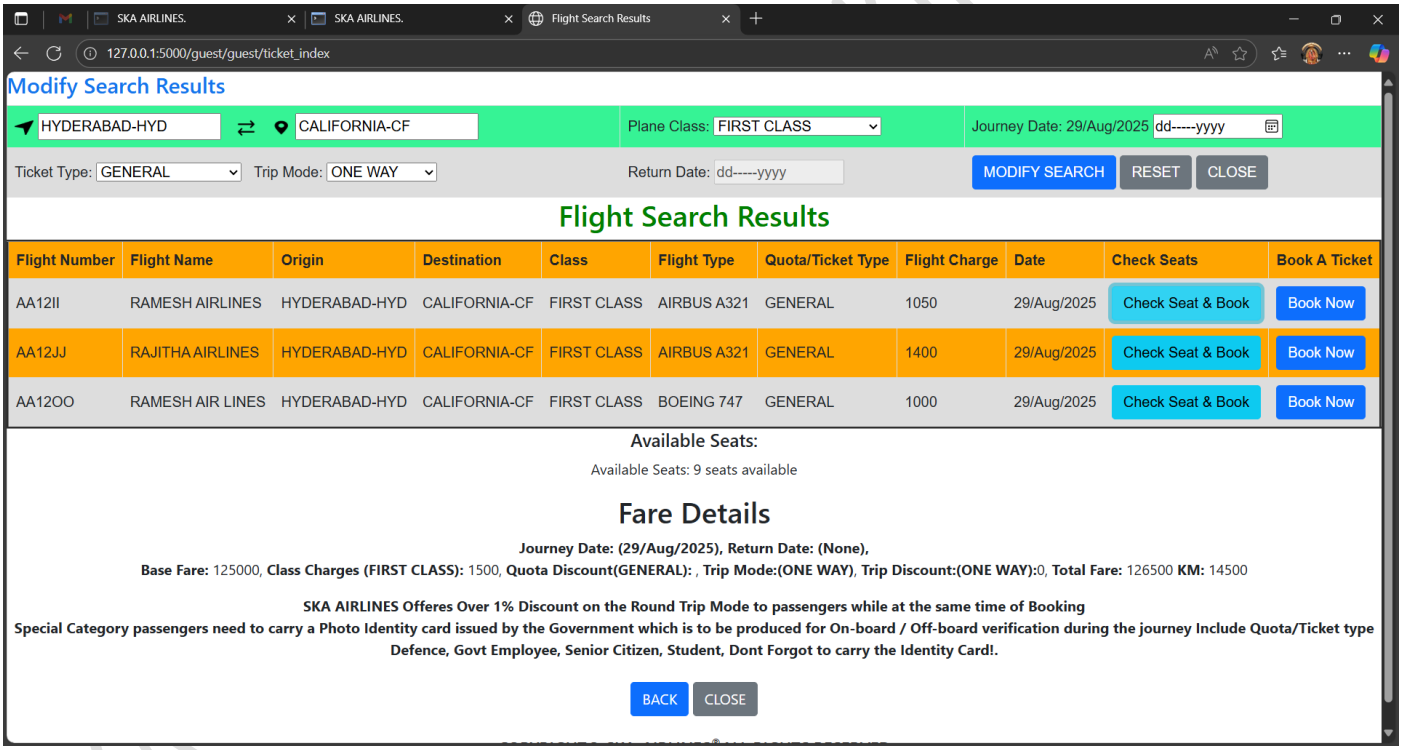
## Flight Guest-User Module ✈️

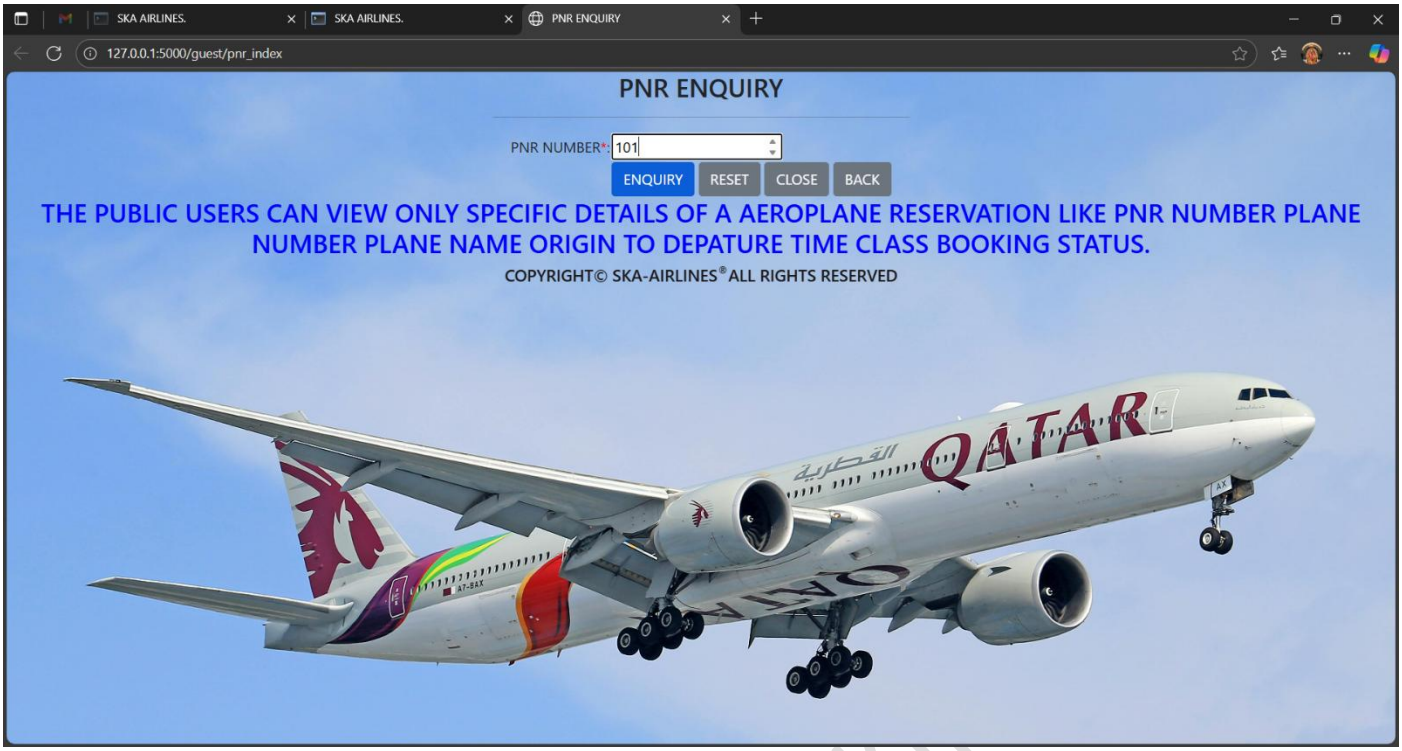
```
|
  ├── Guest Section
  |   ├── guest_dashboard_bp
  |   └── guest_bp
  |
  |  guest_routes.py -----> Handles:
  |   |
  |    ├── user_register_bp        --> user_register.html, register_success.html
  |    ├── feedback_index_bp       --> feedback_index.html, feedback_index_su.html
  |    ├── plane_schedule_bp       --> plane_schedule.html, plane_schedule_su.html
  |    ├── pnr_index_bp            --> pnr_index.html, pnr_index_su.html
  |    └── ticket_index_bp         --> ticket_index.html, ticket_index_su.html
  |
  |  guest_dashboard_bp: also maps routes directly to HTML
  |    ├── airline_admins.html
  |    ├── instructions.html
  |    └── newsevents_index.html
  |
  └── Other Modules
    ├── seat_avl_bp       --> seat_avl.html
    ├── payment_qr_bp
    └── (future modules...)
```

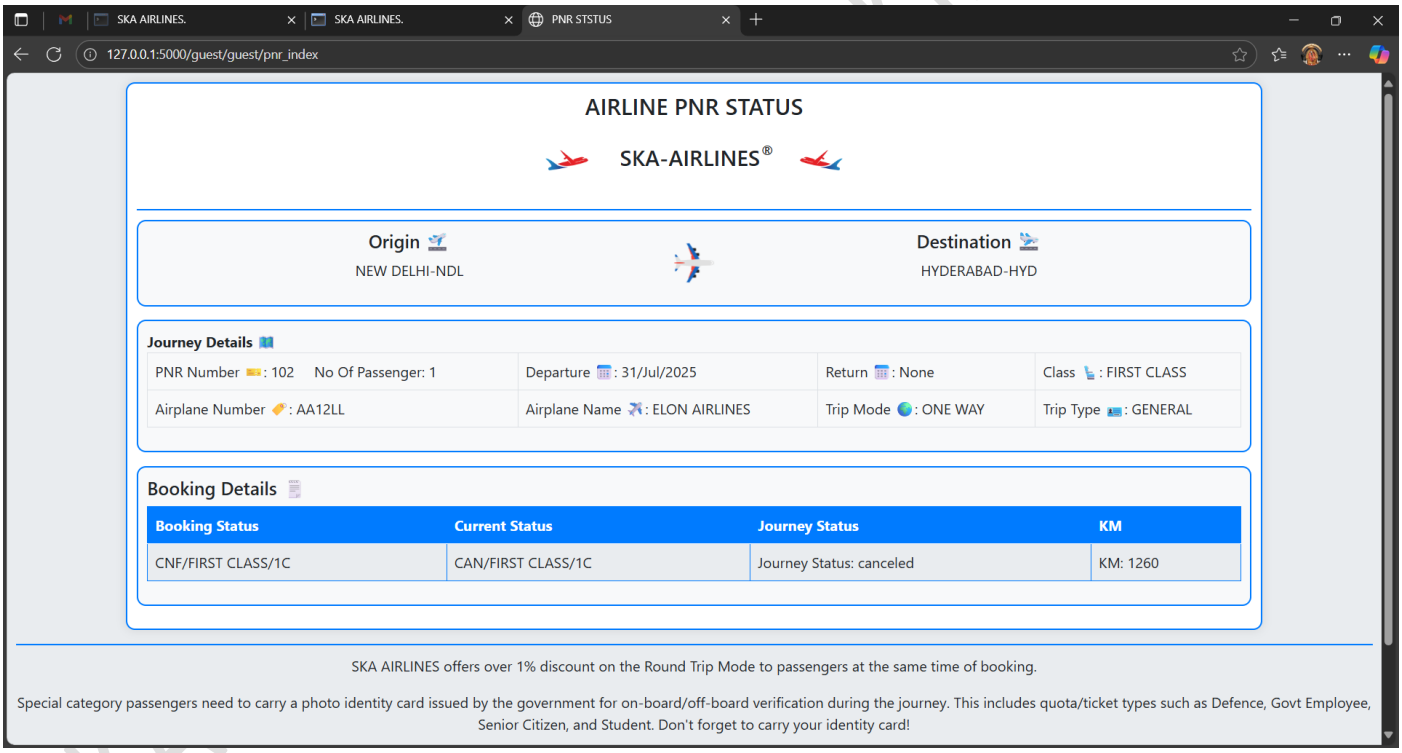
# Flight Guest-User Module Screenshots

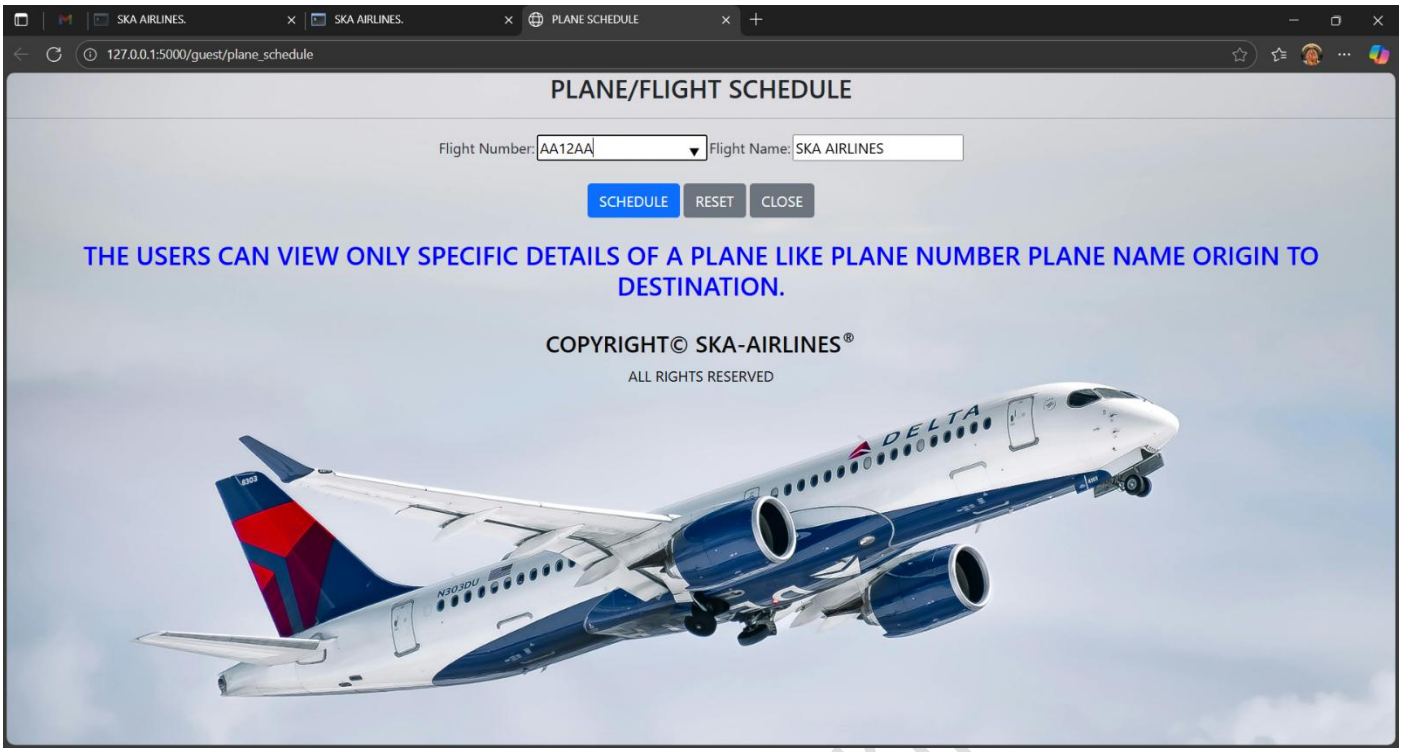



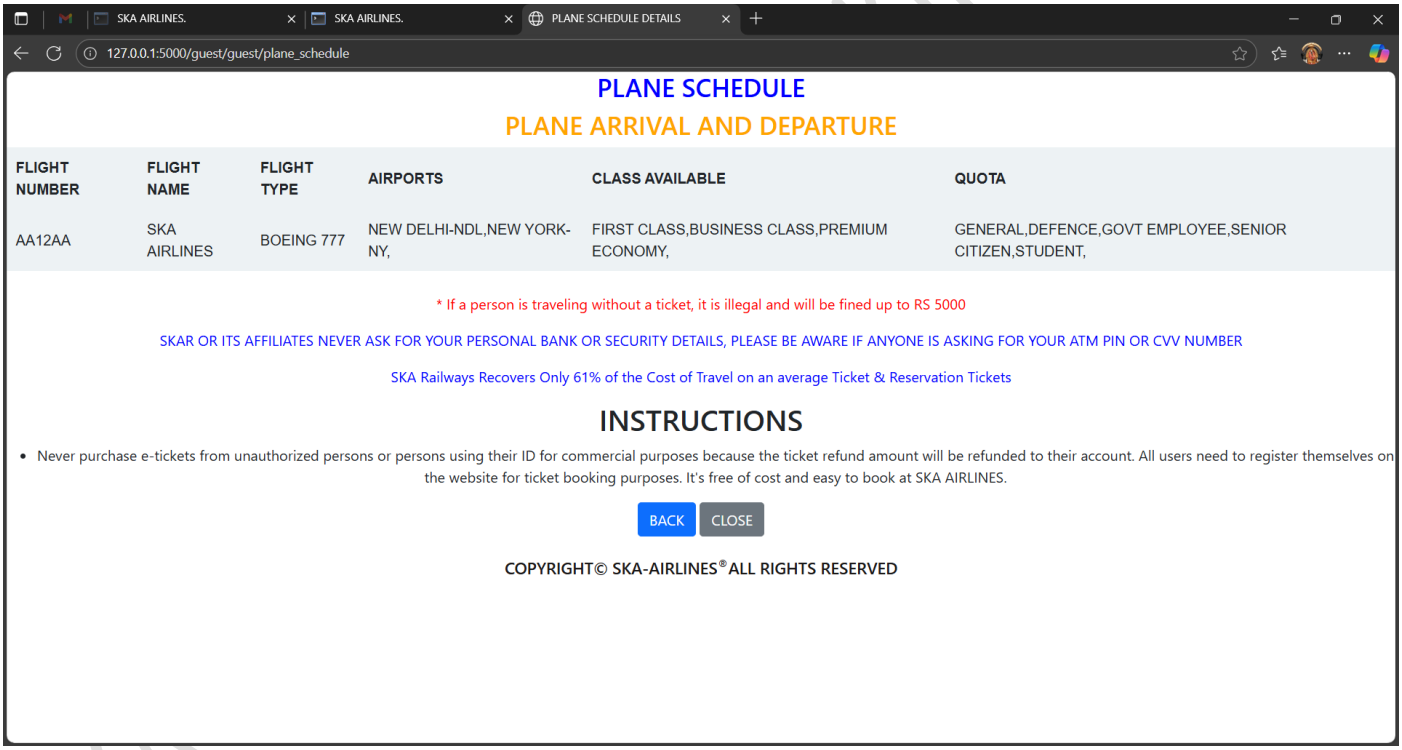

SKA AIRLINES.
SKA AIRLINES.
Airline Booking
127.0.0.1:5000/guest/ticket_index
BOOK A TICKET
ORIGIN
Journey Date: dd-----yyyy
DESTINATION
Plane Class: FIRST CLASS
Ticket Type: GENERAL
Trip Mode: ONE WAY
Return Date: dd-----yyyy
SEARCH
RESET
CLOSE
BACK
COPYRIGHT© SKA-AIRLINES® ALL RIGHTS RESERVED


SKA AIRLINES.
SKA AIRLINES.
Flight Search Results
127.0.0.1:5000/guest/guest/ticket_index
Modify Search Results
HYDERABAD-HYD
CALIFORNIA-CF
Plane Class: FIRST CLASS
Journey Date: 29/Aug/2025 dd-----yyyy
Ticket Type: GENERAL
Trip Mode: ONE WAY
Return Date: dd-----yyyy
MODIFY SEARCH
RESET
CLOSE
Flight Search Results
Flight Number | Flight Name | Origin | Destination | Class | Flight Type | Quota/Ticket Type | Flight Charge | Date | Check Seats | Book A Ticket
AA12II | RAMESH AIRLINES | HYDERABAD-HYD | CALIFORNIA-CF | FIRST CLASS | AIRBUS A321 | GENERAL | 1050 | 29/Aug/2025 | Check Seat & Book | Book Now
AA12JJ | RAJITHA AIRLINES | HYDERABAD-HYD | CALIFORNIA-CF | FIRST CLASS | AIRBUS A321 | GENERAL | 1400 | 29/Aug/2025 | Check Seat & Book | Book Now
AA12OO | RAMESH AIR LINES | HYDERABAD-HYD | CALIFORNIA-CF | FIRST CLASS | BOEING 747 | GENERAL | 1000 | 29/Aug/2025 | Check Seat & Book | Book Now
Available Seats:
Available Seats: 9 seats available
Fare Details
Journey Date: (29/Aug/2025), Return Date: (None),
Base Fare: 125000, Class Charges (FIRST CLASS): 1500, Quota Discount(GENERAL): , Trip Mode:(ONE WAY), Trip Discount:(ONE WAY):0, Total Fare: 126500 KM: 14500
SKA AIRLINES Offeres Over 1% Discount on the Round Trip Mode to passengers while at the same time of Booking
Special Category passengers need to carry a Photo Identity card issued by the Government which is to be produced for On-board / Off-board verification during the journey Include Quota/Ticket type Defence, Govt Employee, Senior Citizen, Student, Dont Forgot to carry the Identity Card!.
BACK
CLOSE


SKA AIRLINES.
SKA AIRLINES.
PNR ENQUIRY
127.0.0.1:5000/guest/pnr_index
PNR ENQUIRY
PNR NUMBER*: 101
ENQUIRY
RESET
CLOSE
BACK
THE PUBLIC USERS CAN VIEW ONLY SPECIFIC DETAILS OF A AEROPLANE RESERVATION LIKE PNR NUMBER PLANE NUMBER PLANE NAME ORIGIN TO DEPATURE TIME CLASS BOOKING STATUS.
COPYRIGHT© SKA-AIRLINES® ALL RIGHTS RESERVED
QATAR


SKA AIRLINES.
SKA AIRLINES.
PNR STSTUS
127.0.0.1:5000/guest/guest/pnr_index
AIRLINE PNR STATUS
SKA-AIRLINES®
Origin
NEW DELHI-NDL
Destination
HYDERABAD-HYD
Journey Details
PNR Number : 102 No Of Passenger: 1
Departure : 31/Jul/2025
Return : None
Class : FIRST CLASS
Airplane Number : AA12LL
Airplane Name : ELON AIRLINES
Trip Mode : ONE WAY
Trip Type : GENERAL
Booking Details
Booking Status
Current Status
Journey Status
KM
CNF/FIRST CLASS/1C
CAN/FIRST CLASS/1C
Journey Status: canceled
KM: 1260
SKA AIRLINES offers over 1% discount on the Round Trip Mode to passengers at the same time of booking.
Special category passengers need to carry a photo identity card issued by the government for on-board/off-board verification during the journey. This includes quota/ticket types such as Defence, Govt Employee, Senior Citizen, and Student. Don't forget to carry your identity card!


SKA AIRLINES.
SKA AIRLINES.
PLANE SCHEDULE
127.0.0.1:5000/guest/plane_schedule
PLANE/FLIGHT SCHEDULE
Flight Number: AA12AA
Flight Name: SKA AIRLINES
SCHEDULE
RESET
CLOSE
THE USERS CAN VIEW ONLY SPECIFIC DETAILS OF A PLANE LIKE PLANE NUMBER PLANE NAME ORIGIN TO DESTINATION.
COPYRIGHT© SKA-AIRLINES®
ALL RIGHTS RESERVED
DELTA


SKA AIRLINES.
SKA AIRLINES.
PLANE SCHEDULE DETAILS
127.0.0.1:5000/guest/guest/plane_schedule
PLANE SCHEDULE
PLANE ARRIVAL AND DEPARTURE
FLIGHT NUMBER
FLIGHT NAME
FLIGHT TYPE
AIRPORTS
CLASS AVAILABLE
QUOTA
AA12AA
SKA AIRLINES
BOEING 777
NEW DELHI-NDL,NEW YORK-NY,
FIRST CLASS,BUSINESS CLASS,PREMIUM ECONOMY,
GENERAL,DEFENCE,GOVT EMPLOYEE,SENIOR CITIZEN,STUDENT,
* If a person is traveling without a ticket, it is illegal and will be fined up to RS 5000
SKAR OR ITS AFFILIATES NEVER ASK FOR YOUR PERSONAL BANK OR SECURITY DETAILS, PLEASE BE AWARE IF ANYONE IS ASKING FOR YOUR ATM PIN OR CVV NUMBER
SKA Railways Recovers Only 61% of the Cost of Travel on an average Ticket & Reservation Tickets
INSTRUCTIONS
Never purchase e-tickets from unauthorized persons or persons using their ID for commercial purposes because the ticket refund amount will be refunded to their account. All users need to register themselves on the website for ticket booking purposes. It's free of cost and easy to book at SKA AIRLINES.
BACK
CLOSE
COPYRIGHT© SKA-AIRLINES® ALL RIGHTS RESERVED

# SYSTEM REQUIREMENTS ✈️💻

To operate efficiently, all software requires specific hardware components and software resources on the computer. These prerequisites are called system requirements and typically serve as guidelines rather than absolute rules. Most software defines two categories of system requirements: Minimum and Recommended.

With increasing demands for higher processing power and resources in newer software versions, system requirements tend to rise over time. Industry analysts suggest this trend drives more system upgrades than technological innovations themselves. Additionally, in a broader context, system requirements refer to the design specifications needed for a system or sub-system.

## SOFTWARE REQUIREMENTS 💻

Front End / Language: HTML5, CSS3, JavaScript

Backend Framework / Domain: Python (Flask)

Database: MySQL (or compatible SQL database)

Web Server: Gunicorn or Apache HTTP Server (for production), Flask built-in server for development

Server Logs: Standard logging formats (e.g., text, JSON)

Operating System: Windows 7, 10, 11, Linux, or macOS

## HARDWARE REQUIREMENTS 🛠️

Processor: Intel Core 2 Duo or later generations (i3, i5, i7, i9)

Main Memory (RAM): Minimum 4 GB

Hard Disk (Storage): Minimum 256 GB

Monitor: Colour Monitor (size as per user preference)

Mouse: Optical Mouse

## FUNCTIONAL REQUIREMENTS ⚙️

Functional requirements specify the actions the system must perform to fulfill the business needs of admins and users. These requirements are based on user needs and task analysis of the existing system.

Users must have a valid Admin/User ID and password to log in to the system. 🔒

If an admin or user enters an incorrect password three consecutive times, the server will mark the attempts as unauthorized and restrict access. 🚫

The system's behaviour is defined by the relationship between inputs (credentials, commands) and outputs (access granted/denied, system responses).

If an admin or user attempts to enter wrong credentials or access the system by directly copying the URL of an active session without proper login, the server will treat such attempts as unauthorized access. ❌🔒

## NON-FUNCTIONAL REQUIREMENTS 🔒⚡

Non-functional requirements define how the system performs rather than what functions it performs. For the SKA Flight Ticket Booking System, these requirements ensure reliability, security, and quality user experience. Key non-functional requirements include:

Security: Protect confidential data such as admin and user credentials, booking information, and payment details through encryption and secure access controls. 🔐

Performance: Ensure fast response times during booking, searches, and data retrieval to handle multiple concurrent users efficiently. ⚡

Safety: Maintain data integrity and prevent unauthorized access or data loss.

Software Quality Attributes: Maintain reliability, usability, scalability, maintainability, and availability to offer a seamless and robust service. 🌟

# SYSTEM DESIGN

The SKA Flight Ticket Booking System's design is structured into logical and physical phases to deliver a scalable, secure, and user-friendly online flight reservation service.

1) System Design for Registered Users

The user requests access by entering the URL of the SKA Flight Booking Server.

On the login page, the user enters their credentials (User ID and PIN). The system validates these credentials.

If valid, the user is granted access to the main menu, where options include View Journey History, Book a Flight, View Flight Schedule, Update Profile, and more.

For booking a flight, the user inputs Source (From), Destination (To), Travel Dates, and Class preferences, then submits the details

The system displays flight options; upon selecting a flight, the user confirms the booking by entering their secure PIN to finalize the transaction.

Upon logout, users are recommended to clear browser cache for security reasons.

## 2) System Design for Flight Admins

The admin requests the URL of the SKA Flight Booking Server.

Admins input their credentials (Admin ID and PIN) on the login page; the system checks their validity. , If valid, admins access the admin dashboard featuring options like Add New Flight, Add New Airport, Modify Fare, View PNR Status, Manage Reservations, View Flight Schedules, and Print/Download Tickets.

For example, when using the PNR enquiry menu, admins input a PNR number and submit it; the system retrieves and displays detailed booking information for confirmation.

3) Development Phases

The system design is developed through two major phases:

a) Logical Design 📋

Analyze current project requirements including data flows, file contents, volume, and frequencies, Define output specifications such as report formats and frequencies.

Specify input formats, content, and functions.

Establish edit, security, and control processes

Prepare and review an information flow walkthrough including inputs, outputs, controls, and implementation plans., Evaluate benefits, costs, timelines, and system constraints.

b) Physical Design 📐

Develop the actual working system based on the logical design.

Specify input/output devices and media formats.

Design and implement the database with backup and recovery protocols

Plan and design the physical information flow and system walk-through.

Schedule system implementation and prepare conversion and training plans.

Define test procedures and specify any additional required hardware or software.

Update project benefits, costs, milestone dates, and constraints accordingly.

This comprehensive design approach ensures the SKA Flight Ticket Booking System is secure, reliable, user-friendly, and scalable to meet modern airline reservation needs efficiently. ✈️🛠️🔒

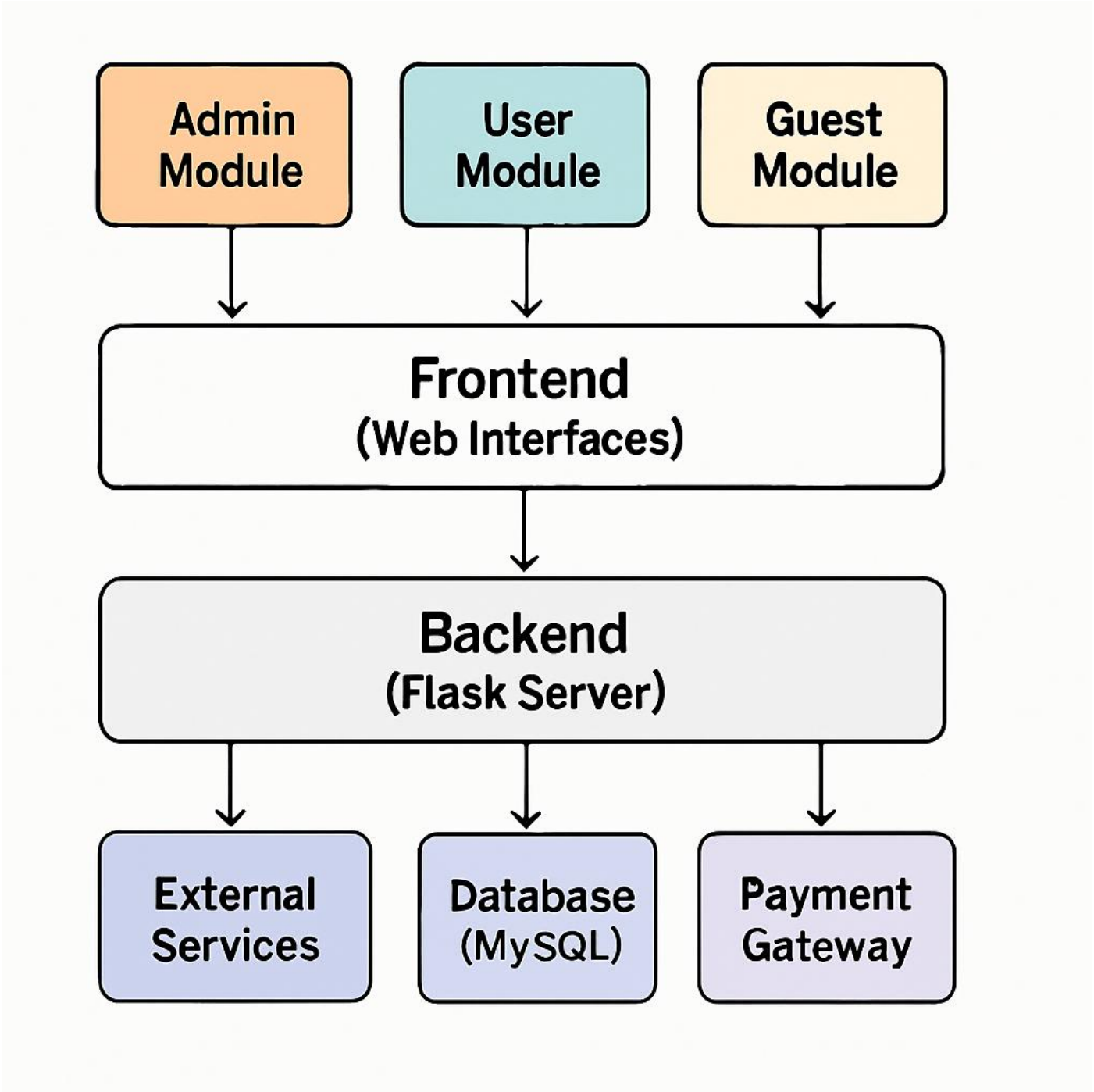
**Flowchart: Flight Admin System Flow**

1. Start
2. Admin requests URL of SKA Flight Booking System
3. Display Admin Login Page (Admin enters Admin ID and PIN)
4. Verify Admin ID and PIN
    - If Yes:
        - Show Admin Dashboard with options:
            - Add New Flight
            - Add New Airport
            - Modify Fare
            - View PNR Status
            - Manage Reservations
            - Print/Download Tickets
    - If No: Return to Login Page
5. If View PNR selected:
    - Input PNR Number
    - Submit request and display details
6. Logout
7. End

**Flowchart: Registered User System Flow**

1. Start
2. User requests the URL of SKA Flight Booking System
3. Display Login Page (User enters User ID and PIN)
4. Check if User ID and PIN are valid
    - If Yes:
        - Show Main Menu with options:
            - Book Flight
            - View Journey History
            - Update Profile
            - Other features
    - If No: Return to Login Page
5. If Book Flight selected:
    - Input Source, Destination, Travel Dates, and Class
    - Submit booking request
    - Confirm booking by entering secure PIN
6. Logout and clear browser cache for security
7. End

# SKA FLIGHT BOOKING SYSTEM ARCHITECTURE DIAGRAM



# Flight Reservation Booking Terms ✈️📋

- Reservation: The process of booking a seat on a specific flight for a particular date and route.
- PNR (Passenger Name Record): A unique identifier code generated for each booking that contains passenger details, itinerary, and ticket information. Used for tracking and managing bookings. 🔍
- E-ticket (Electronic Ticket): A digital version of a flight ticket issued electronically without a physical paper ticket. It contains all travel information and is proof of purchase. 🎟️
- Fare: The price paid by the passenger for a flight ticket, which can vary based on class, time, and other factors. 💰
- Booking Confirmation: The official acknowledgment by the airline that a reservation has been successfully made, often sent via email. ✅✉️
- Cancellation: The process by which a booked ticket is voided. Cancellation policies, fees, and refund eligibility depend on the fare rules. ❌↩️
- No-Show: A passenger who does not show up for the flight without prior cancellation or notification. This may lead to forfeiture of fare. 🚫
- Class of Service: The travel class selected by the passenger, e.g., Economy, Business, or First Class, affecting fare and amenities. 🛋️
- Connection / Layover: A stop or break between flights during a journey, where passengers may switch planes. ⏳✈️
- Seat Availability: The current number of free seats on a flight for booking, updated in real-time. 💺
- Payment Gateway: A service that processes online payments securely when booking tickets, supporting multiple payment modes like bank transfer or QR code. 💳🔒
- Booking Reference: Another term sometimes used interchangeably with PNR, referring to the unique code for a booking.

The Below table can be quick reference and user clarity.

| Term | Definition |
|---|---|
| Reservation | The process of booking a seat on a specific flight for a particular date and route. |
| PNR (Passenger Name Record) | A unique booking code containing passenger and itinerary details, used to track/manage bookings. 🔍 |
| E-ticket | An electronic version of a flight ticket; digital proof of purchase with travel details. 🎟️ |
| Fare | The price paid for a flight ticket, which varies by class, timing, and other factors. 💰 |
| Booking Confirmation | Official acknowledgment from the airline that the booking was successful, often via email. ✅✉️ |
| Cancellation | Voiding a booked ticket; refund policies depend on fare ❌ |
| No-Show | Passenger who misses the flight without cancellation, risking loss of fare. 🚫 |
| Class of Service | Travel class chosen by passenger (Economy, Business, First), affecting price and amenities. 🛋️ |
| Connection / Layover | A stop between flights allowing transfer to anotherflight. ⏳✈️ |
| Seat Availability | Number of free seats on a flight updated in real-time. 💺 |
| Payment Gateway | Secure online service processing payments (bank transfers, QR codes). 💳🔒 |
| Booking Reference | Another term for PNR; unique code used to identify a booking. |

# Flight Types / Terms

| Term | Description |
|---|---|
| Domestic Flight | Flights operating within the same country, connecting cities or regions domestically. |
| International Flight | Flights that cross international borders, connecting different countries. |
| Direct Flight | A flight with no change of plane but may include stopovers for refueling or passenger boarding. |
| Non-Stop Flight | A flight that goes from origin to destination without any stops in between. |
| Connecting Flight | A journey with one or more stopovers requiring passengers to change planes at an intermediate airport. |
| Charter Flight | A non-scheduled flight arranged for specific groups or purposes, not part of the regular airline schedule. |
| Red-Eye Flight | Overnight flights typically departing late at night and arriving early morning. |
| Code Share Flight | A flight operated by one airline but marketed by one or more other airlines under their own flight number. |
| Round-Trip Flight | Travel from the origin to a destination and back to the origin on a single booking. |
| One-Way Flight | Travel from the origin to the destination only, without a return ticket. |

| Term | Description |
|---|---|
| Standby Flight | A flight where passengers wait for available seats, usually after missing an earlier flight. |
| Seasonal Flight | Flights operated only during specific seasons or periods (e.g., holiday seasons or summer). |
| Code Share Flight | A collaborative flight sharing arrangement where multiple airlines sell seats on the same flight. |

This table can help users better understand the types of flights available and the terminology used in airline booking systems 📕📚.

**1. Emirates Airbus A380 (International Long-Haul Flight)** 🌍

- First Class: Private suites with doors for complete privacy, located at the front of the aircraft offering maximum comfort and luxury, Business Class: Spacious 2-2-2 seating with lie-flat seats, designed for long-haul travel comfort, Economy Class: High-density 3-4-3 layout maximizing capacity
- Flight Type: International, long-haul flights connecting major global hubs

**2. Boeing 777 (International and Some Domestic Flights)** 🌎🏙️

- Business Class: 2-3-2 seating arrangement, balancing space and capacity for medium to long-haul flights, Economy Class: Standard 3-3-3 layout with moderate legroom typical of many international and high-capacity domestic routes.
- Flight Type: Used commonly for international long-haul flights & busy domestic routes

**3. Regional Jet (Primarily Domestic or Short-Haul Flights)** 🏞️

- Economy Class: Compact 1-2 or 2-2 seating arrangements designed for short flights with quick turnarounds.
- Flight Type: Domestic or regional flights connecting smaller airports and shorter distances, focusing on efficient use of space and quick service.

# Flight Seats Layout System

Airlines use various seat layout configurations depending on aircraft type, flight duration, and service class. Understanding seat layouts helps users select preferred seats during booking.

**Common Seat Layout Types**

| Layout Type | Description | Typical Configuration |
|---|---|---|
| Single Aisle (Narrow-body) | Aircraft with one central aisle. Common in short-haul flights. | 3-3 (six seats per row, e.g., A-B-C aisle D-E-F) |
| Dual Aisle (Wide-body) | Aircraft with two aisles, used mainly for long-haul international flights. | 2-4-2, 3-3-3, or 3-4-3 seat arrangements |
| Regional Jet | Smaller jets with fewer seats, suitable for short regional routes. | Typically 1-2 or 2-2 seat configurations |

**Seat Classes & Layouts**

| Class | Description | Layout Characteristics |
|---|---|---|
| Economy Class | Standard seating, higher seat density, basic legroom and amenities. | Rows of seats in tight configuration, usually 17-18 inches wide seats. |
| Premium Economy | Enhanced economy seats with more legroom and amenities. | Wider seats, more legroom, sometimes recline feature. |
| Business Class | Spacious seating with enhanced amenities, priority boarding, and meals. | Lie-flat or angled seats, 1-2-1 or 2-2-2 layouts common. |
| First Class | Most luxurious seating, private suites, top-tier service. | Private suites or enclosed seats, spacious, can include beds. |

**Seat Selection Features**

- Window Seat: Located by the aircraft window, preferred for views and privacy.
- Aisle Seat: Located by the aisle, preferred for easier access.
- Exit Row: Extra legroom but may have restrictions on passengers (e.g., must be able to assist in emergencies).
- Bulkhead Seat: Located at cabin row dividing classes or sections, extra legroom but sometimes no under-seat storage.
- Seat Map Display: Interactive visual of seat availability shown during booking.
- Seat Availability Status: Indicated via colors (available, reserved, blocked).

**Example Seat Map Notation**

- A-F: Seat letters typically represent seats from left to right starting from the window.
- Row numbers: Numbered front to back of the cabin. E.g., row 12 A-F means seats in row 12 from window-left (A) to window-right (F).
-

Including detailed seat layout information enhances user experience by helping travelers choose preferred seats confidently during booking.

Let me know if you want me to create sample seat maps or diagrams!Here is an overview of common Seat Layout Systems used in airline reservation platforms, including their typical configurations and terminology, for inclusion in your SKA Flight Ticket Booking System documentation:

**Seat Layout Systems** 💺✈️

Airlines use various seat layout configurations depending on aircraft type, flight duration, and service class. Understanding seat layouts helps users select preferred seats during booking.

**Common Seat Layout Types**

| Layout Type | Description | Typical Configuration |
|---|---|---|
| Single Aisle (Narrow-body) | Aircraft with one central aisle. Common in short-haul flights. | 3-3 (six seats per row, e.g., A-B-C aisle D-E-F) |
| Dual Aisle (Wide-body) | Aircraft with two aisles, used mainly for long-haul international flights. | 2-4-2, 3-3-3, or 3-4-3 seat arrangements |
| Regional Jet | Smaller jets with fewer seats, suitable for short regional routes. | Typically 1-2 or 2-2 seat configurations |

---

**Seat Classes & Layouts**

| Class | Description | Layout Characteristics |
|---|---|---|
| Economy Class | Standard seating, higher seat density, basic legroom and amenities. | Rows of seats in tight configuration, usually 17-18 inches wide seats. |
| Premium Economy | Enhanced economy seats with more legroom and amenities. | Wider seats, more legroom, sometimes recline feature. |
| Business Class | Spacious seating with enhanced amenities, priority boarding, and meals. | Lie-flat or angled seats, 1-2-1 or 2-2-2 layouts common. |
| First Class | Most luxurious seating, private suites, top-tier service. | Private suites or enclosed seats, spacious, can include beds. |

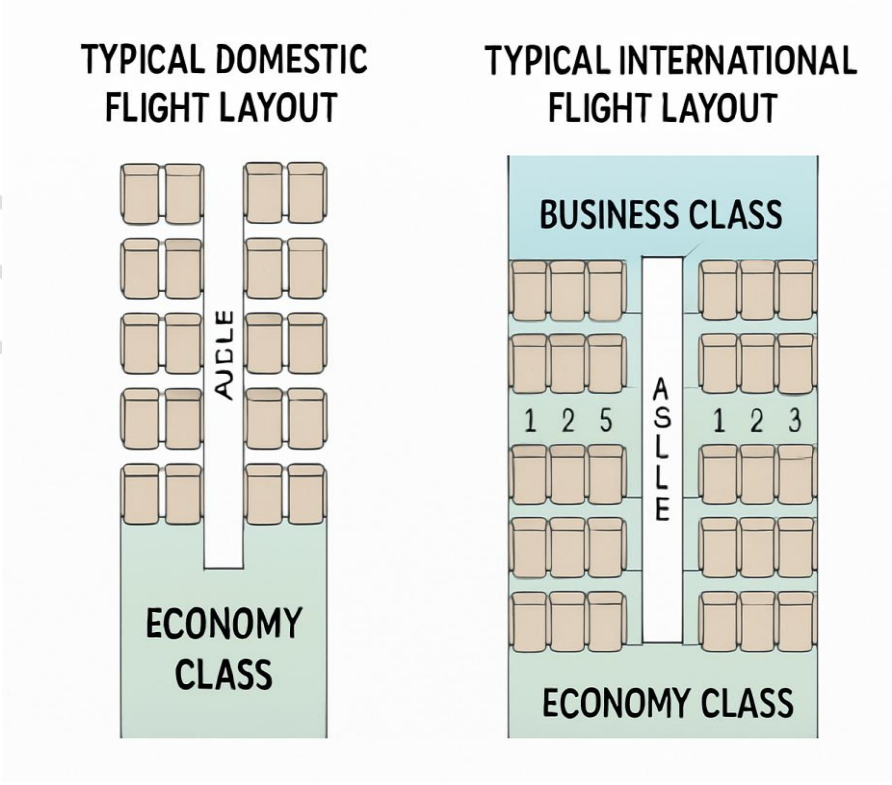
**Seat Selection Features**

- Window Seat: Located by the aircraft window, preferred for views and privacy.
- Aisle Seat: Located by the aisle, preferred for easier access.
- Exit Row: Extra legroom but may have restrictions on passengers (e.g., must be able to assist in emergencies).
- Bulkhead Seat: Located at cabin row dividing classes or sections, extra legroom but sometimes no under-seat storage.
- Seat Map Display: Interactive visual of seat availability shown during booking.
- Seat Availability Status: Indicated via colors (available, reserved, blocked).

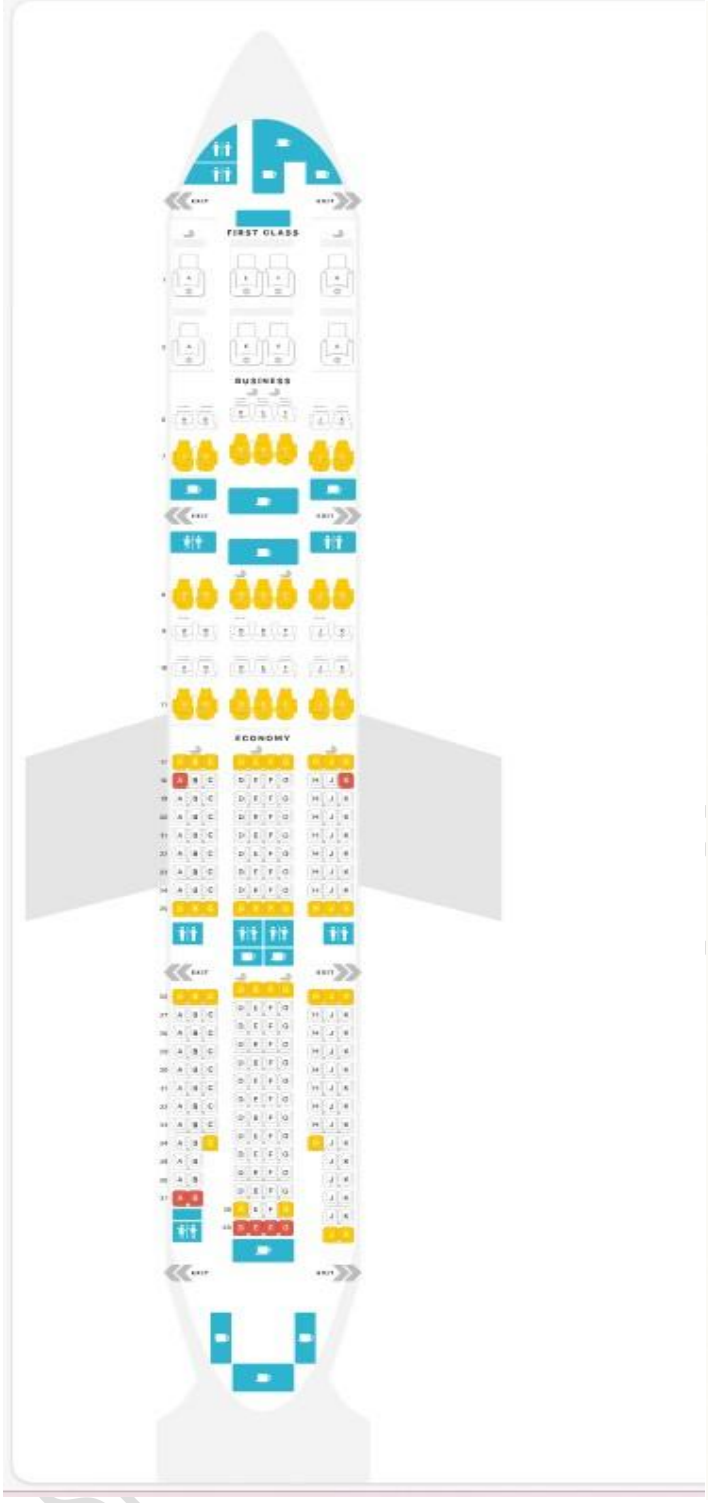
**Example 1 Seat Map Notation**

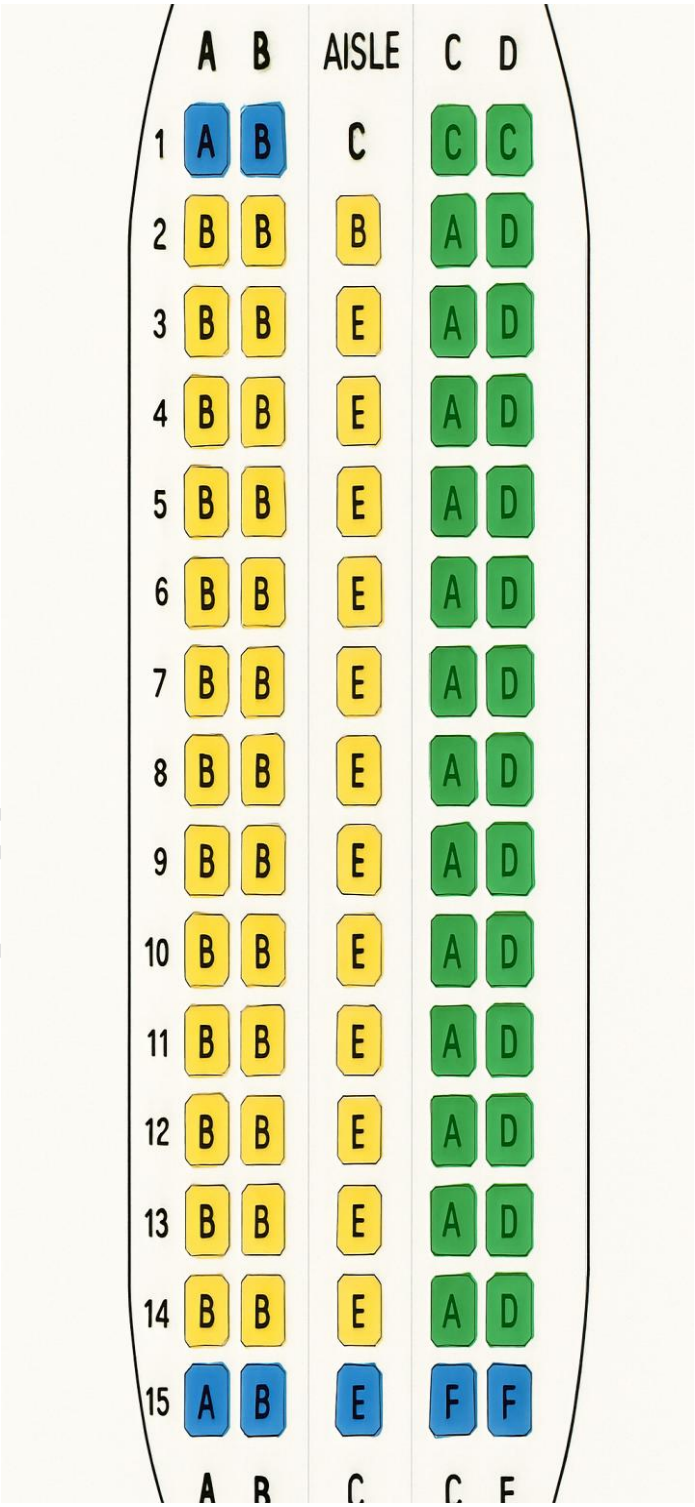
- A-F: Seat letters typically represent seats from left to right starting from the window.
- Row numbers: Numbered front to back of the cabin. E.g., row 12 A-F means seats in row 12 from window-left (A) to window-right (F).

Including detailed seat layout information enhances user experience by helping travelers choose preferred seats confidently during booking.



**Example 2 Seat Map Notation**



## Customer's Obligations 🧑‍💻🔐

1. The customer must keep their Username and Password confidential. The bank assumes that login with valid credentials represents a session initiated by the customer.
2. Transactions executed through a valid session are legally binding on the customer as originating from them. ✅
3. The customer must never attempt or allow others to access the SKA Flight Ticket Booking System through unlawful means. 🚫

- While registering on the website, users must note down their User Number, Username, and Password.
- Tickets booked under PHC (Physically Handicapped Card) or Ladies Quota require passengers to carry valid identity proof. 🆔
- When changing passwords, users must enter their old password, then the new password with confirmation. 🔄
- PNRs with full waiting list status are automatically dropped and refunded. 💸
- Never purchase e-tickets from unauthorized persons. ❌
- SKA or its affiliates will never ask for your personal bank or security details. 🛡️
- If a flight delay exceeds 4 hours, refunds are admissible. ⏳↩️
- In case of flight cancellation, the full refund is automatically credited to the user's account. ✈️💰

**Dos & Don'ts** ✅❌

1. Keep your User ID and password strictly confidential. Any loss due to negligence is the customer's responsibility; SKA Flight Booking is not liable. 🔐
2. Choose strong, unique passwords avoiding easily guessable personal information like name, birthdate, or phone number. Memorize passwords rather than writing them down. 🧠💡
3. Never leave your computer unattended during an active session, as others may access your account. 👀🚷

**Safe Online Flight Booking Tips** 🛡️💻

- Check that the URL in your browser begins with “https”, where “s” means secured. 🔒
- Look for the padlock icon in the browser’s address bar and verify the security certificate by clicking it. 🔐
- Do not enter sensitive information in any pop-up windows. 🚫
- A green address bar indicates a secured website. ✅

**Beware of Phishing Attacks** ⚠️📧

- Phishing is a fraudulent attempt via email, phone calls, or SMS to steal your confidential information. 🦹‍♂️💍
- SKA or its representatives never request personal information, passwords, or one-time SMS (OTP) passwords via email, SMS, or phone calls. 🚫📵
- Such communications are attempts to fraudulently withdraw money or access your account. Never respond! ❌
- Change your internet booking and banking passwords regularly. 🔄
- Always check your last login date and time on the secure post-login page. ⏰

**Important Reminder**

Scammers often send fake emails or texts pretending to be trusted companies like banks or utilities. They trick users into clicking malicious links or opening attachments to steal passwords or personal info.

Stay vigilant and protect your information! 🛡️🛑

## THE SKA FLIGHT TICKET BOOKING SYSTEM HIERARCHY / PROCESS

The SKA Flight Ticket Booking System is designed with a clear hierarchy of modules and a smooth process flow that ensures efficient flight reservation and management for users, admins, and guests.

**System Hierarchy Overview**

1. User Interface Layer (Frontend)
    - Web pages and dashboards accessed by
        - Registered Users
        - Flight Admins
        - Guest Users
2. Application Layer (Backend)
    - Flask framework handling business logic including:
        - User authentication and authorization
        - Flight search and booking
        - Payment processing (bank transfers, QR payments)
        - Ticket management and cancellations
        - Feedback collection
        - Admin operations (flight/airport/fare management, PNR enquiries)
3. Data Layer (Database)
    - MySQL database storing:
        - User profiles and credentials
        - Flight schedules and aircraft info
        - Booking and ticket details (PNRs, e-tickets)
        - Payments and refunds
        - Feedback and logs

4. External Services Integration 🌐
    - Payment gateways for secure transactions
    - Email services for ticket confirmations and notifications

**System Process Flow** 🔄

1. User / Guest Access
    - Request website URL and reach landing page.
    - Register (new user), log in (registered user/admin), or browse as guest.
2. Authentication & Authorization 🔐
    - User/Admin submits credentials.
    - System validates via backend and grants access.
3. Main Operations
    - Users search flights by criteria (source, destination, date).
    - Users select preferred flight and proceed to book.
    - Multiple payment options offered: bank accounts, QR code.
    - Tickets booked and e-tickets generated. Ticket status emailed.
4. Booking Management
    - Users can view journey history, update profile, cancel bookings.
    - Admins manage flights, fares, PNR enquiries, cancellations, refunds.
5. Security & Monitoring
    - Secure sessions maintained; improper access attempts logged.
    - Password and data confidentiality enforced strictly.

This hierarchy and process ensure a secure, efficient, and user-friendly flight booking experience for all stakeholders across the SKA Flight Ticket Booking System. ✈️ 🌓 ☑️

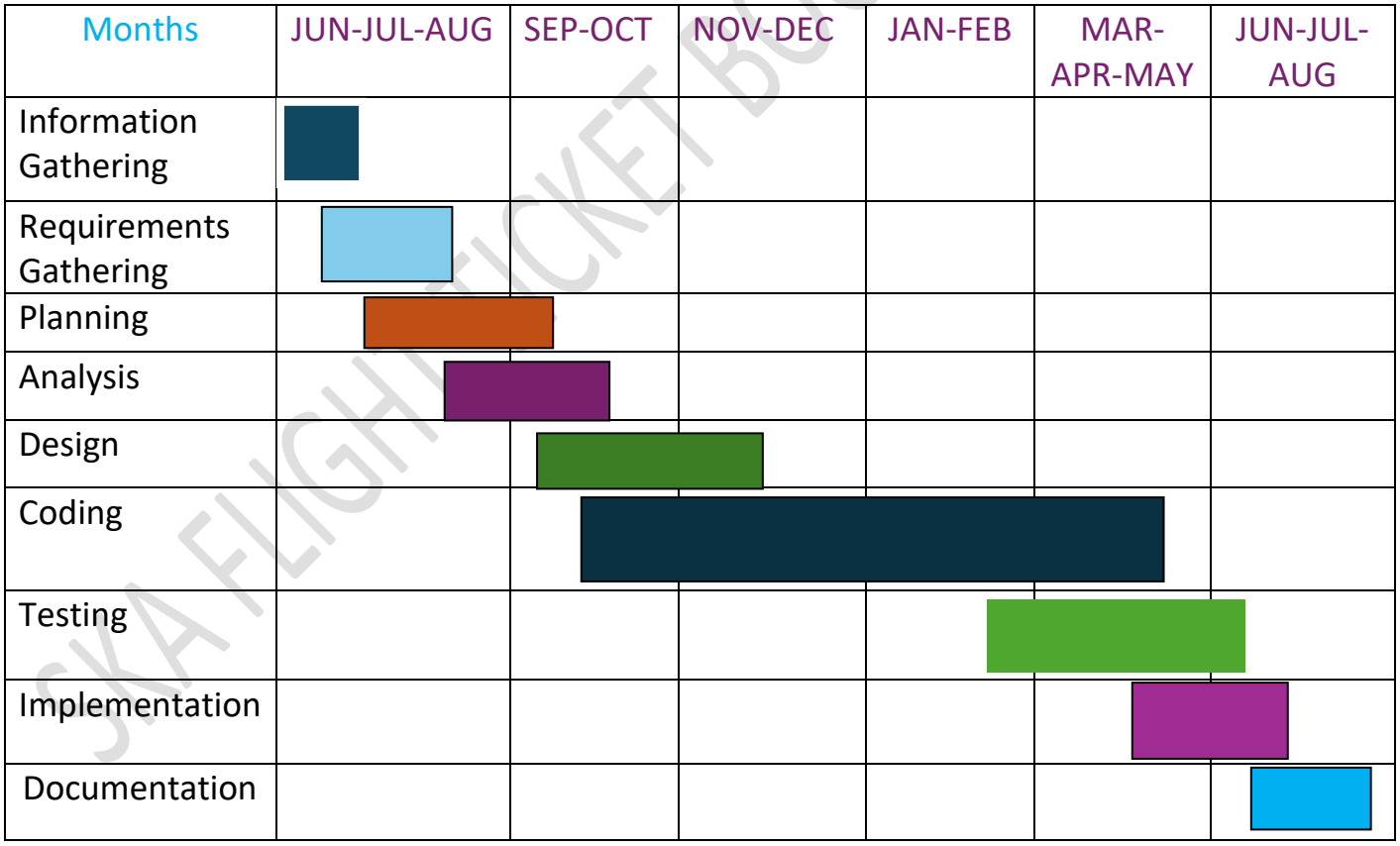
# PROJECT SCHEDULING

I estimated the time and resources needed for the SKA Flight Ticket Booking System project while planning the schedule. All project activities must be arranged in a coherent sequence, meaning tasks are organized logically and clearly for easy understanding and smooth execution.

Initial project estimates are often optimistic, assuming favorable conditions with no major threats or problems during development.

**Project Timeline**

- Duration: June 2024 to August 2025
- The project will be implemented in 2 major phases to ensure systematic development and deployment.



# Benefits of Online Flight Ticket Booking ✈️💻

Many of us lead busy lives. Some wake up before dawn, preparing ourselves and our families for the day ahead. We rush to work, get the kids to school, and at the end of the day, rush home only to prepare for the next day. After such a hectic schedule, the last thing anyone wants is to spend time waiting in long lines at the airport or ticket counters.

That's where Online Flight Ticket Booking with SKA comes in. Many benefits of booking your flight tickets online are obvious:

- You don't have to wait in line. ⏳🚫
- You don't need to plan your day around the airport or ticket office hours. 🕒
- You can check your flight information anytime, not just when you receive a statement. 📱

There are also some hidden benefits:

- If you are new to flight booking, using the SKA system helps you manage your account and travel plans while observing your spending patterns. 📊
- Online booking allows you to monitor your flights and schedule daily if desired. Keeping close tabs on your bookings means you're always aware of your travel status. 🗓️
- For experienced travelers, this system helps prevent surprises like running out of travel funds unexpectedly. 💸
- It's also useful to track any service charges, time savings, or fees you may have incurred, helping you manage your travel budget better. ⏰💰

Booking flights online with SKA is a smarter, faster, and more convenient way to plan your travels! ✈️✨

**More Benefits of Online Flight Ticket Booking with SKA ✈️🌐**

**Convenience Anytime, anywhere 🕒📱**

- Book your flights 24/7 from the comfort of your home, office, or on the go using any device—desktop, tablet, or smartphone.
- No more rushing to airport counters or dealing with long queues; your entire booking is just a few clicks away.

**Instant Access to Information** 🔎🛫

- View up-to-date flight schedules, delays, cancellations, and gate changes in real-time.
- Receive timely email and SMS alerts about your bookings and travel updates directly from SKA.

**Multiple Payment Options** 💳💸

- Pay securely using multiple methods including bank transfers, credit/debit cards, and QR code payments.
- Choose the payment method that best fits your convenience and comfort.

**Easy Modifications and Cancellations** 🔄❌

- Modify your bookings or cancel tickets anytime online without hassle.
- Get automatic notifications and refunds processed smoothly when applicable.

**Personalized Travel Experience** 👤🎯

- Save your preferences and travel history for faster bookings next time.
- Access special offers and loyalty rewards tailored just for you.

**Secure and Confidential** 🔒🛡️

- Protect your personal and payment information with SKA's advanced security features.
- SKA never asks for your full bank details outside secure payment gateways to keep your data safe.

**Environmentally Friendly Option** 🌱🌍

- Reduce paper waste by using e-tickets instead of printed tickets.
- Travel greener by embracing digital solutions powered by SKA.

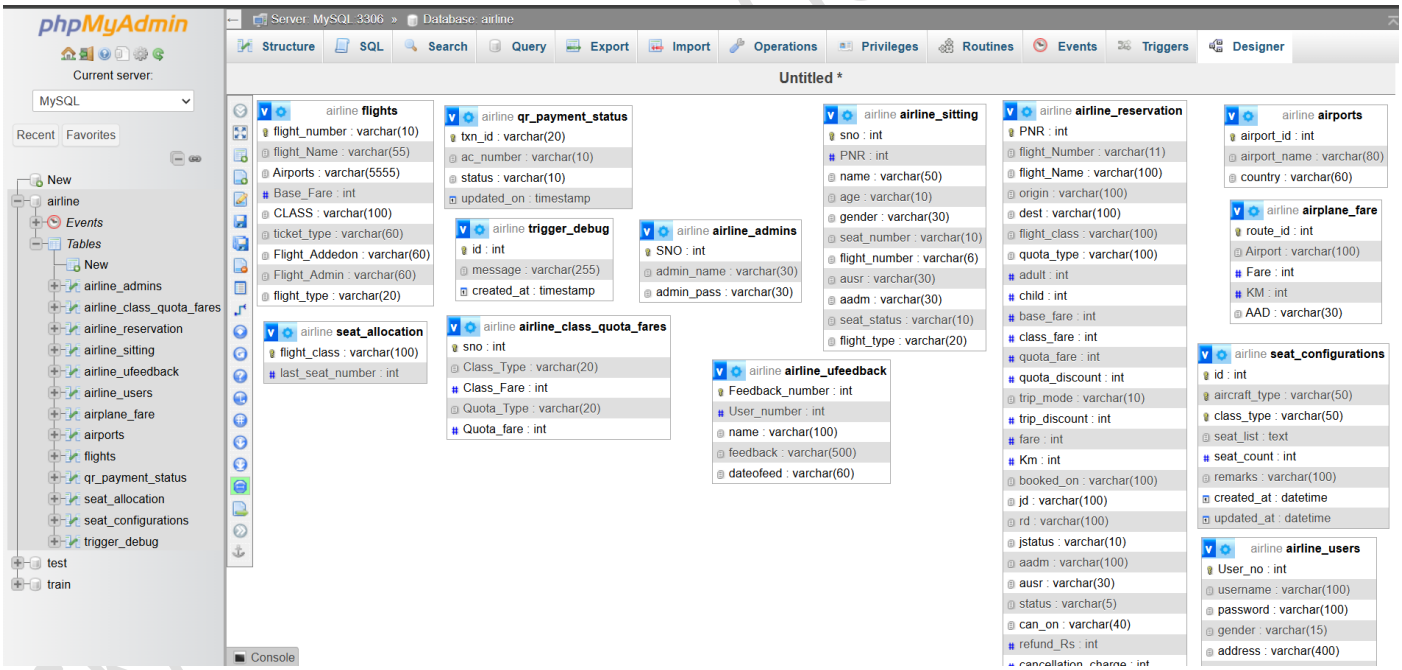
Booking online with SKA Flight Ticket Booking System is more than just purchasing a ticket—it's the start of a smooth, secure, and personalized travel journey.

# SKA FLIGHT TICKET BOOKING SYSTEM APPLICATION DATABASE/TABLES

The SKA Flight Booking System's database is the core component that powers the entire application by securely managing and organizing all essential data. It includes multiple interconnected tables designed to handle user profiles, admin credentials, flight schedules, seat layouts, ticket bookings, payments, and feedback.

Each table stores specific information—such as passenger details, flight types, fare structures, and transaction records—allowing for efficient querying and seamless data integrity across the platform. The relational design supports dynamic features like real-time seat availability, automated PNR generation, and flexible payment processing.

By structuring data meticulously, the database ensures fast access, reliable updates, and smooth synchronization between backend processes and frontend user interfaces, ultimately delivering a robust and responsive flight booking experience.



## THE SKA FLIGHT BOOKING SYSTEM APPLICATION CALLS/FLOW

airlines.py <-- Main entry, initializes and registers all blueprints

|

├── Admin Section

| ├── admin_login_bp <-- Login functions

| ├── admin_logout_bp <-- Logout logic

| ├── admin_dashboard_bp <-- HTML renderings (see below)

| └── admin_bp <-- Route connectors

| admin_routes.py -----> Imports blueprints and submodules:

| |

| ├── new_flight_bp --> new_flight.html

| ├── add_airport_bp --> add_airport.html

| ├── add_fare_bp --> add_fare.html

| ├── change_fare_bp --> change_fare.html

| ├── ticket_admin_bp --> ticket_admin.html, ticket_admin1.html, ...

| ├── ticket_download_admin_bp --> ticket_download_admin.html

| ├── user_booking_history_bp --> user_booking_history.html

| ├── airplane_airports_bp --> airline_airports_su.html

| ├── admin_can_ticket_bp --> admin_can_ticket.html, admin_can_ticket1.html, ...

| ├── payment_gateway_tkt_can_admin_bp

| ├── airline_fund_bp --> airline_fund_su.html

| └── (other submodules...)

├── User Section

| ├── user_login_bp

| ├── user_logout_bp

| ├── user_dashboard_bp

| └── user_bp

| user_routes.py -----> Handles:

```
|    ├── ticket_user_bp          --> ticket_user.html, ticket_user1.html, ...
|    ├── payment_gateway_bp      --> payment_gateway_su.html
|    ├── ticket_userdb_bp
|    ├── user_can_ticket_bp      --> user_can_ticket.html, user_can_ticket1.html, ...
|    ├── user_journey_history_bp --> user_journey_history_su.html, ...
|    ├── user_feedback_bp        --> user_feedback.html, user_feedback_su.html
|    ├── user_update_mbl_num_bp  --> user_update_mbl_num.html, ...
|    ├── user_update_password_bp --> user_update_password.html, ...
|    └── qr_payment_bp           --> qr_payment_su.html
├── Guest Section
|    ├── guest_dashboard_bp
|    └── guest_bp
|   guest_routes.py -----> Handles:
|    |
|    ├── user_register_bp        --> user_register.html, register_success.html
|    ├── feedback_index_bp       --> feedback_index.html, feedback_index_su.html
|    ├── plane_schedule_bp       --> plane_schedule.html, plane_schedule_su.html
|    ├── pnr_index_bp            --> pnr_index.html, pnr_index_su.html
|    └── ticket_index_bp         --> ticket_index.html, ticket_index_su.html
|   guest_dashboard_bp: also maps routes directly to HTML
|    ├── airline_admins.html
|    ├── instructions.html
|    └── newsevents_index.html
|
└── Other Modules
    ├── seat_avl_bp        --> seat_avl.html
    ├── payment_qr_bp
    └── (future modules...)
```

## THE SKA FLIGHT BOOKING SYSTEM ENTIRE APPLICATION ENVIRONMENT ✈

Airlines/

|

├── admin/

| ├── add_airport.py

| ├── add_fare.py

| ├── admin_can_ticket.py

| ├── admin_can_ticket1.py

| ├── admin_delete_feedback.py

| ├── admin_login.py

| ├── admin_routes.py

| ├── admin_view_feedback.py

| ├── airline_fund.py

| ├── airplane_airports.py

| ├── airports.py

| ├── change_fare.py

| ├── logout.py

| ├── new_flight.py

| ├── payment_gateway_tkt_can_admin.py

| ├── pnr_enq_admin.py

| ├── routes.py

| ├── seat_avl.py

| ├── ticket_admin.py

| ├── ticket_admindb.py

| ├── ticket_download_admin.py

| ├── usernumber.py

| ├── user_booking_history.py

| ├── utils.py

```
|   └── __init__.py
|
├── Extra Programs/
|   ├── Application_Issues_bugs
|   ├── brbr.txt
|   ├── check things in project.txt
|   ├── dateand time.txt
|   ├── detailed program.txt
|   ├── Email.txt
|   ├── from flask import Blueprint, render.txt
|   ├── good logic.txt
|   ├── mix.txt
|   ├── next steps.txt
|   └── Old trigger dynamic.txt
|
├── guest/
|   ├── feedback_index.py
|   ├── guest_routes.py
|   ├── plane_schedule.py
|   ├── pnr_index.py
|   ├── routes.py
|   ├── ticket_index.py
|   ├── user_register.py
|
├── guest/__pycache__/
|   ├── feedback_index.cpython-312.pyc
|   ├── guest_routes.cpython-312.pyc
|   ├── plane_schedule.cpython-312.pyc
```

```
|   ├── pnr_index.cpython-312.pyc
|   ├── routes.cpython-312.pyc
|   ├── routes_py.cpython-312.pyc
|   ├── seats_avl.cpython-312.pyc
|   ├── ticket_index.cpython-312.pyc
|   └── user_register.cpython-312.pyc
|
├── MYSQL/
|   ├── airline (30).sql
|   ├── airline 28.sql
|   ├── airline 33.sql
|   └── airline 34.sql
|
├── Paper Work/
|   ├── IMG_20240905_173915450.jpg
|   ├── IMG_20240905_173943677.jpg
|   ├── IMG_20240905_173954705.jpg
|   ├── IMG_20240905_174000415.jpg
|   ├── IMG_20240905_174005665.jpg
|   └── IMG_20240905_174011513.jpg
|
├── routes/
|   ├── db_config.py
|   ├── email_cnf_can.py
|   ├── oracle_db_config.py
|   ├── payment_gateway_Qr.py
|   ├── qr_code_generator.py
|
```

```
├── routes/__pycache__/
|   ├── db_config.cpython-312.pyc
|   ├── email_cnf.cpython-312.pyc
|   ├── email_cnf_can.cpython-312.pyc
|   ├── oracle_db_config.cpython-312.pyc
|   ├── payment_gateway_Qr.cpython-312.pyc
|   └── qr_code_generator.cpython-312.pyc
|
├── sample tickets/
|   ├── SKA AIRLINES Boarding Pass (11).pdf
|   ├── SKA-AIRLINES Boarding Pass (8).pdf
|   └── SKA-AIRLINES Boarding Pass (9).pdf
|
├── Screenshots/
|   ├── Screenshot 2025-07-30 080941.png
|   ├── Screenshot 2025-07-30 081116.png
|   ├── Screenshot 2025-07-30 081129.png
|   └── Screenshot 2025-07-30 081142.png
|
├── static/
|   ├── aeroplane9.jpg
|   ├── airport1.jpg
|   ├── airport2.jpg
|   ├── airport3.jpg
|   └── airport4.jpg
|
├── templates/
|   ├── adminlogin.html
```

```
|   ├── airline_admins.html
|   ├── base.html
|   ├── error.html
|   ├── index.html
|   ├── test.html
|   ├── ticket_index.html
|   ├── userlogin.html
|   ├── admin/
|   |   ├── add_airport.html
|   |   ├── add_fare.html
|   |   ├── admin_can_ticket.html
|   |   ├── admin_can_ticket1.html
|   |   ├── admin_can_ticket2.html
|   |   ├── admin_can_ticket_su.html
|   |   ├── admin_delete_feedback.html
|   |   ├── admin_delete_feedback_su.html
|   |   ├── admin_instructions.html
|   |   ├── admin_news_events.html
|   |   ├── admin_view_feedback_su.html
|   |   ├── airline_airports_su.html
|   |   ├── airline_fund_su.html
|   |   ├── airport_success.html
|   |   ├── change_fare.html
|   |   ├── change_fare_su.html
|   |   ├── fare_success.html
|   |   ├── new_flight.html
|   |   ├── pnr_enq_admin.html
|   |   ├── pnr_enq_admin_su.html
```

```
|   |   ├── seat_avl.html
|   |   ├── seat_avl_su.html
|   |   ├── success.html
|   |   ├── ticket_admin.html
|   |   ├── ticket_admin1.html
|   |   ├── ticket_admin2.html
|   |   ├── ticket_admin3.html
|   |   ├── ticket_admin_su.html
|   |   ├── ticket_download_admin.html
|   |   ├── ticket_download_admin_su.html
|   |   ├── user_booking_history.html
|   |   ├── user_booking_history_su.html
|   |   └── welcome_admin.html
|   ├── guest/
|   |   ├── feedback_index.html
|   |   ├── feedback_index_su.html
|   |   ├── instructions.html
|   |   ├── newsevents_index.html
|   |   ├── plane_schedule.html
|   |   ├── plane_schedule_su.html
|   |   ├── pnr_index.html
|   |   ├── pnr_index_su.html
|   |   ├── register_success.html
|   |   ├── ticket_index.html
|   |   ├── ticket_index_su.html
|   |   └── user_register.html
|   ├── partials/
|   |   └── seat_availability.html
```

```
|   ├── user/
|   |   ├── payment_gateway_su.html
|   |   ├── qr_payment_su.html
|   |   ├── ticket_user.html
|   |   ├── ticket_user1.html
|   |   ├── ticket_user2.html
|   |   ├── ticket_user3.html
|   |   ├── ticket_user4.html
|   |   ├── ticket_user_su.html
|   |   ├── user_can_ticket.html
|   |   ├── user_can_ticket1.html
|   |   ├── user_can_ticket2.html
|   |   ├── user_can_ticket_su.html
|   |   ├── user_feedback.html
|   |   ├── user_feedback_su.html
|   |   ├── user_journey_history1_su.html
|   |   ├── user_journey_history_can1_su.html
|   |   ├── user_journey_history_can_su.html
|   |   ├── user_journey_history_cnf1_su.html
|   |   ├── user_journey_history_cnf_su.html
|   |   ├── user_journey_history_su.html
|   |   ├── user_update_mbl_num.html
|   |   ├── user_update_mbl_num_su.html
|   |   ├── user_update_password.html
|   |   ├── user_update_password_su.html
|   |   └── welcome_user.html
|
 ├── user/
```

```
|   ├── logout.py
|   ├── payment_gateway.py
|   ├── payment_gateway_ticket_cancellation.py
|   ├── qr_payment.py
|   ├── routes.py
|   ├── ticket_user.py
|   ├── ticket_userdb.py
|   ├── user_can_ticket.py
|   ├── user_can_ticket1.py
|   ├── user_can_ticket2.py
|   ├── user_feedback.py
|   ├── user_journey_history.py
|   ├── user_journey_history1.py
|   ├── user_journey_history_can.py
|   ├── user_journey_history_can1.py
|   ├── user_journey_history_cnf.py
|   ├── user_journey_history_cnf1.py
|   ├── user_login.py
|   ├── user_routes.py
|   ├── user_update_mbl_num.py
|   ├── user_update_password.py
|   ├── utils.py
|   └── __init__.py
|
├── __pycache__/
|   ├── airline.cpython-312.pyc
|   ├── airports.cpython-312.pyc
|   ├── flight_name.cpython-312.pyc
```

```
|   ├── flight_number.cpython-312.pyc
|   ├── usernumber.cpython-312.pyc
|   └── user_numbers.cpython-312.pyc
|
├── airline.py
├── airports.py
├── Application_Issues_bugs/
├── filelist.txt
├── flight_number.py
├── models.py
├── Project Descriptions.txt
├── run.py
├── versions.txt
├── __init__.py
|
├── .idea/
|   ├── .gitignore
|   ├── .name
|   ├── Airline.iml
|   ├── misc.xml
|   ├── modules.xml
|   ├── workspace.xml
|   └── inspectionProfiles/
|       └── profiles_settings.xml
|
└── .vscode/
    └── launch.json
```

# THE SKA FLIGHT BOOKING SYSTEM VERSIONS

The SKA Flight Booking System has evolved through multiple versions, each improving features, performance, and user experience. This progression reflects ongoing enhancements in technology, security, and functionality to meet growing traveler and airline needs effectively.

THE SKA AIRLINES VERSIONS AND APPLICATION DEVELOPMENTS

SKA Airlines Application Changelog

The latest version is always the newest ; airline0 is old , and airline1 is newer than airline0 .

For reference, the last version has all the most recent updates .

airline3: Up to the index page, changes are applied there .

airline4: Up to the index page, with updates to ticket_index for guest users .

The file routes.py manages HTML routes , and common .py files are used throughout the website .

Airline v4

Data now retrieves from DB tables in the right places , such as routes_html.py inside the needed module, Progressed up to the airport_fare section .

Version 7

Added plane quota feature when adding a new plane; tracking capacity is much easier now!

Version 8

Implemented new user registration with error flashes when users try to register duplicates or refresh the page.

Checked pages for flash messages: Whenever you add things like airports or fares, you get an alert for duplicates, If the endpoint is user_register_success.html and the page is reloaded, you're redirected to the homepage with an error, preventing duplicate DB entries .

Version 9

All guest-related index functions completed, with tests , ticket index, and user feedback included .

If flights for a chosen origin-destination pair don't exist, there's an alert popup .

Fixed journey date logic : now the return date can't be before the journey date .

User-side reservation is paused , while admin-side implementation is underway .

Version 10 🔄

Shared data for HTML pages is now passed centrally through Python routes ⚙️, no more repetitive transfer.

Fares automatically update ⬆️, and all passenger names are shown in uppercase 🔠.

Version 11 📱

Admin-side booking features auto-assign seat numbers based on class selection 🚦 by Python code—all logic, zero SQL!

All tickets and tables now use a responsive design 📲.

Added user number at reservation time; registered users can re-download their ticket from the portal 🔃.

Version 12 🎯

Refined application responsiveness: focus is now on downloading tickets instead of printing 🖨️➡️💾.

UI and fare logic now support roundtrips: base fare is doubled (base_fare*2) ⏩, and dynamic updates are clearly visible!

Fixed issues with previous double-calculation logic; updates now show live 🔁.

Version 13 👴💼

Admin entering a username now reveals the user number instantly 👀.

Developed almost all admin features, including a robust PNR enquiry/audit system 🔍 for all bookings.

Admin and user booking histories are linked 📈; also added textbox for country code when adding new airports 🌏.

Version 14 🗓️

If the journey date is reached, journey status updates to “completed” automatically ✅.

When booking is "In Progress," the status is set to “IP” instantly ⚡.

Cancelled tickets are clearly flagged 🚩. Event handlers with SQL keep everything up to date.

Plane schedules are available to all users 👥, and entering a plane number instantly reveals the plane name 🛬.

Version 15 📲

Built PNR index module; user index page is now mobile-friendly 📱.

Plane schedule and PNR index pages are shared between main user and index pages 🔗.

When downloading tickets, each user only sees their own—privacy protected! 🔒

Version 16 🏦

Fully developed the Bank Gateway 💳 integrated with User Ticket Booking 🎫 with proper logic and data flow.

Implemented up to user ticket download 💾.

Version 17 ⚠️

Added caution alerts 🚨 for users and admin when entering the user name during booking.

Developed User Journey History tracking 📜: routes from index -> user_journey_history.py -> user -> user_journey_history_su.html -> user_journey_history1.py -> user_ticket_download_su.html

Used session management to ensure data is accessed only for the particular user 🔒.

Version 18 ❌🎟️

Developed fully user cancellable tickets with Bank Gateway integration 🔄🏦.

Reduced separate views for user to download CNF (confirmed) or CAN (canceled) tickets with all journey statuses.

Removed the old ticket download module 🔥 (no need to enter user PNR again).

Users can select a group of CNF, CAN, or both from the list for easy download 📥.

Version 19 🗣️📱

Developed User Feedback module 📝 and enhanced user mobile update 📲.

Version 20 🔑🔄

Developed User Password Update Module 🔐.

Improved some code usability and maintainability 🛠️.

Version 21 ⚠️📜

Developed Admin Cancellation Ticket Module 🛑 but pending payment mode handling.

Problem: Admin booked tickets without bank account details cause issues on cancellation (empty account info).

Planned: Added module to require account number at admin booking or cancellation time (mandatory if digital payments involved) 💳.

Version 22 💸

Completed Admin side cancellation with payment types: Cash and Digital Payment 💵📲.

Pending: Dynamic linking of account user number and name.

Version 23 📜💻

Developed full Admin cancel ticket module with automatic fetching of user account number and name in Python code 🤖.

Added JavaScript logic for both user and admin sides to display ticket details dynamically:

For CAN tickets ➡️ show cancellation fare info 🔴

For CNF tickets ➡️ show confirmed details only ✅

Reduced extra code for cancellation ticket handling 🎉.

Version 24 🗓️💻

Developed date handling completely in Python code (not HTML forms) 🗓️🖥️.

Reduced ticket download modules by unifying CNF and CAN ticket handling using JavaScript visibility toggles (hide empty cancellation details) 👻👁️.

Enhanced:

Admin side feedback control with deletion on selection 🗑️

Maintained accurate Airline Fund Details 💰 for SKA Airlines.

Improved code layout and structure for better maintenance 🖌️.

Version 25 🪑

Focused on seat availability based on user/admin seat selection 🎟️🛋️.

Created seats_avl.py in admin folder, shared with admin, user, and guest.

Used JSON + MySQL join queries for realistic seat booking and layout visualization 🎨.

Version 26 🔁

Added submodule to check/modify flight search details on the same page 👨💻👩💻 for admin and guest users.

Enables quick rechecking with different plane classes, dates, and fares without page reload 🕛.

Version 27 🔢

Developed seat number auto-allocation based on flight_number, flight_class, with seat resets upon reaching max seats 🔁 seats reset.

Version 28 🔎🪑

Improved seat allocation trigger considering flight_number, flight_class, and journey date (jd).

Example:

Flight: AA12AA, Class: First, JD: 15/Sep/2024 → Seat: 1A

Flight: AA12AA, Class: First, JD: 16/Sep/2024 → Seat: 1A

Flight: AA12AA, Class: First, JD: 15/Sep/2024 (next seat) → Seat: 1B

Allocation dynamically resets per flight, class, and journey date.

Attempted integration: linked SKA Airlines to SKA Bank at admin ticket booking time only.

User-side ticket booking and cancellation fully linked with bank gateway 🏦.

Admin side ticket cancellation done ✅, but booking-side bank account detail input is optional.

Fund increments to SKA Airlines bank account (account 106) handled if provided.

Version 29 💾🔒

Backed up and inserted all transactions into bank-side database with full control and auditing 🔐.

Issue noted: Cancelling single passenger tickets works, but multiple passenger reservation ticket cancellations fail ❗.

Version 30 🚀🔥 (Most Important Version!)

Solved all previous issues; implemented correct seat allocation trigger for cancelled seats → seats freed up as available based on flight number, journey date, and flight class 🪑✅.

Fixed issue where only single tickets could be cancelled, now multiple ticket cancellations work smoothly 🎉.

Bug fix: replaced cursor.fetchone() with cursor.fetchall() for multiple records handling 🦎🐞.

Added "Number of Passengers" display in public PNR status 👴🤝👴.

Fixed user journey history to accurately show confirmed, canceled, and all statuses 📜.

Removed problematic submodule showing user numbers on admin side during booking — avoiding counter ticket confusion 🚫.

Implemented counter ticket cancellation logic: tickets booked by admin must be cancelled only on admin side as counter tickets ⚠️.

Added "All rights reserved" notices on specified pages ©.

✅ PHASE 1: Previously Completed Modules (Theoretical Summary) 🚀✈️

1. Multi-Step Ticket Booking System 🛫🛬 (Admin & User Side)

Goal: Seamlessly streamline flight ticket booking across multiple pages for a smooth user experience.

ticket_admin1.html ✈️: Captures initial booking inputs like flight number, origin 🌆, destination 🌍, and dates 🗓️.

ticket_admin2.html 🎟️: Selects travel class (Economy, Business), quota type 🎫 (e.g., Senior, Student), trip type (One-way/Round-trip 🔁), and number of passengers 👴👧👴👴.

ticket_admin3.html 🕵️: Collects passenger details — name, age 🎂, gender 🚻, and more.

ticket_admin_final.html 📋: Displays a comprehensive booking summary for final confirmation ✔️.

Note: Sessions 🔁 seamlessly carry data across pages, ensuring no details are lost during the booking flow!

2. Fare Calculation System 💸🧮

Goal: Dynamically calculate ticket fares based on smart business logic.

Base Fare: Standard flight fare 🎫

Class Fare: Additional charge based on travel class 🧳 (e.g., Economy, Business).

Quota Discounts: Special discounts 🏷️ for categories like seniors 👴, students 🎓.

Trip Discounts: Extra savings for round trips 🔁.

Real-time fare updates powered by JavaScript ⚡ keep passengers informed as they add travelers or change selections — no page reloads needed!

3. Seat Allocation Logic 🪑✨

Goal: Automate seat assignment for every booking to maximize efficiency.

Seats stored in airline_sitting table 📊, e.g., First Class seats from 1A to 3D.

Seats are assigned sequentially 🎯, looping or managing overflow smoothly when full.

Candidate for implementation via SQL triggers 🗃️ or Python logic 🐍 — both approaches explored depending on scalability & flexibility needs.

4. Ticket Cancellation Logic 🔄❌ (User/Admin)

Goal: Enable smooth ticket cancellation, seat freeing, and refund processing.

Cancellation forms available for both admin and users 👴💼👩‍💼, Upon cancellation: Updates the reservation's status field 🚦 (e.g., canceled). Calculates and processes refund amount refund_Rs 💰. Frees or flags the seat in the airline_sitting table for reuse ♻️.

THE PHASE 2

✅ Phase 1 Recap — Locked & Completed

These are already implemented, tested, and finalized:

| ✔ Feature | Details |
|---|---|
| Multi-Step Ticket Booking | Admin + User Booking Flow |
| Dynamic Fare Calculation (JS) | Class, Quota, Trip Mode based |
| Round Trip Fare Logic | Fare doubling + Return Date check |
| Seat Allocation Trigger | CNF/Seat assignment via Trigger |
| Seat Reuse Logic | Based on Flight No + JD/RD |
| Ticket Cancellation + Refund | Refund logic based on status/date |
| PDF Ticket + Cancel PDF | Using reportlab or fpdf |
| Admin Dashboard (Text) | CRUD, Search, Manage |
| Passenger Manifest | Based on PNR & Flight filter |
| Payment Simulation (Manual) | Form input only |

---

🚀 PHASE 2 DEVELOPMENT PLAN

Unleashing dynamic, modern features for SKA Airlines! ✨

1. 📜 QR Code Integration in Ticket PDFs

Purpose: Fast airport check-in & digital ticket verification—COMPLETED ✅

pip install qrcode[pil]  # QR Code generation

Libraries Used: qrcode, base64, reportlab/fpdf

QR Code Data Example:

json

{

"PNR": "12345",

"Passenger": "John Doe",

"Flight No": "AI-202",

"Seat": "2A",

"Date": "2025-07-21"

}

Workflow:

Generate QR Code → Base64 encode → Embed in ticket PDF

Optionally preview QR in HTML before download 👀

2. 📧 Email Notification System

Purpose: Instantly update users via email on bookings and cancellations ✉️📩

When to Send:

On Booking ✅ (attach PDF ticket)

On Cancellation ❌ (with refund details or summary)

https://myaccount.google.com/apppasswords to Generate the app password to use in the email system

Libraries & Install Commands:

text

pip install Flask flask-mail reportlab

Email Package: Flask-Mail (SMTP-based)

Session Management:

text

pip install Flask-Session

python

from flask_session import Session

app.config['SESSION_TYPE'] = 'filesystem'  # Or use 'redis', 'sqlalchemy'

Session(app)

To Do:

Configure mail server and authentication 🔑

Trigger email on ticket booking/cancellation events 📬

3. 💳 QR Code Payment Gateway (Simulated)

Purpose: Simulate secure payments with QR before final booking 💰

Sample QR Payment Data:

json

{

  "Amount": 5600,

  "Reference": "PNR-12345",

  "Payee": "SKA Airlines",

  "Timestamp": "2025-07-08 11:23:12"

}

Steps:

Show payment QR on page

Simulate “Payment Done” → Confirm booking

4. 📊 Graphical Admin Dashboard

Purpose: Bring insights to life with eye-catching visuals!

Key Charts:

Bookings per Day 🗓️

Revenue per Flight/Class 💸

Cancellation vs Confirmed Trends 📊

Libraries:

Frontend: Chart.js, Plotly.js

Backend: Flask serves JSON endpoints

To Do:

Dashboard HTML

AJAX endpoints for live data

Note: Reservation funds visualization already implemented ✅

5. 🪄 Advanced Seat Allocation by Flight Type

Purpose: Automatically manage seat allocation by aircraft type 🚦

Features:

Flight types (Boeing, Airbus) + seat range config for FC, BC, PECO, ECO

Auto-assign seats according to chosen flight type ✈️

Implementation:

Add flight_type field in flights table

On selecting flight, auto-populate seat/class limits (JS or backend)

To Do:

Module: flight_type_seats.py

Embed logic in JS/frontend or backend

6. 🛠️ Common DB Configuration File

Purpose: Centralized & secure database connection 🔐

File: db_config.py

python

```
import mysql.connector

def get_connection():
    return mysql.connector.connect(
        host='localhost',
        user='root',
        password='tiger',
        database='airline'
    )
```

Usage:

python

```
from db_config import get_connection
```

Status: DONE ✅

7. 🔁 Dynamic Fare Calculation (From DB)

Purpose: No more hardcoded fares—fetch dynamically from DB 📂

Current: Static fare in HTML—OUTDATED 🚫

New: Calculate on backend from class, quota, base fare, origin, destination, distance

To Do:

Revise DB schema if needed

Implement logic to fetch accurate fare on every booking

Status: Successfully implemented! 🎉

8. 🧮 Admin Fare Update & Visualization

Purpose: Equip admin with powerful fare editing & analysis tools! 🛠️

Features:

Form-based UI: edit fares for class/km/origin-destination

Optional: auto-calculate fare increases by percentage

Action Items:

Fetch/update from fare_config table

Build responsive admin HTML page

9. 🌐 Dynamic IP Address / Domain Binding

Purpose: Make SKA Airlines accessible by domain name instead of raw IP 📲

Outcome: Users can reach the app at https://ska-airlines.com vs. http://xxx.xxx.xxx.xxx

Status: DONE ✅

10. ⏰ Enhanced Time Logic (Optional Future Step)

Purpose: Add robust Departure/Arrival Time management in booking system

What's Next?

Integrate time pickers in booking UI

Use backend logic to ensure valid, timezone-aware times

Installations Required

```
pip install qrcode[pil]        # QR Code generation
pip install Flask flask-mail     # Email sending
pip install Flask-Session        # Session management
pip install reportlab          # PDF generation
pip install mysql-connector-python
pip install mysql-connector-python  # MySQL database connector
```

🔥 VERSION 31

Implemented fully importable QR code generation on tickets 🎟️✨

Developed seamless ticket layouts integrating dynamic QR codes for fast scanning and verification.

Made the code modular and reusable wherever needed. 🧩✅

🚀 VERSION 32

Centralized Database Access Created 🗄️🔑

Developed a unified DB connection module for consistent and easy import across all Python programs.

Simplified database handling and improved maintainability. 💻🔧

💳 VERSION 33

Introduced Advanced Payment via QR Code on Payment Gateway ⚡🛡️

Integrated complex payment workflow involving three bank-side JSP programs:

uselqr.jsp, qr_codeaccept.jsp, authorize_payment.jsp

These communicate with our Flask backend via IP and port. 🌐🔌

Created new transaction table under airlines DB to capture txn_id and validate payment status dynamically. 💾✔️

Pending: Full ACID-compliant payment method processing under payment modules. ➡️SOON🛠️

🔁 VERSION 34

Dynamic Fare Calculation from DB Implemented 📊💰

Replaced static HTML fares with dynamic backend logic fetching from airline_class_quota_fares table.

Fare calculation now smartly considers:

Origin & Destination ✈️

Travel Class & Quota 🎫

Distance 🗺️

Flight base fare adjusted by flight priority for speciality flights 🚀

Timestamp auto-update added for auditing in airlines_users:

sql

ALTER TABLE airlines_users

MODIFY COLUMN DOR TIMESTAMP NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP;

📧 VERSION 35

User-Side Email System Launched for Ticket Booking & Cancellation 💌✅

Sent professional, grammatically polished emails for both confirmed (CNF) and cancelled (CAN) bookings.

Corrected ticket cancellation info wording on printed tickets for clarity and professionalism. 📝🎟️

✉️ VERSION 36

Enhanced Email Logic to Avoid Duplicate Sends on Refresh 🔁🚫

Prevented repeated emails on accidental browser refresh during booking or cancellation.

Fixed unknown error page on user ticket cancellation related to refunds.

Positioned email dispatch before ticket_su and ticket_can_su methods in ticket_userdb.py and user_can_ticket2.py.

Admin Side Improvements:

Added extra email field in admin booking for capturing user emails. 📩

Fetches & uses user email during admin cancellations for realistic communication. ✔️

Overall: Nearly all application issues resolved; future updates will focus on minor fixes and optimizations. 🛠️🎉

🛠️ VERSION 37

Fixed User Ticket Journey History Display on Cancellations 📜❌

Ensured cancelled tickets properly reflect in user journey history with clear status messages. ✅

Skipping detailed notes on Version 37 to proceed to next significant update.

🌟 VERSION 38

Highly Dynamic Seat Allocation Based on Flight Number, Class, Journey Date & Flight Type 🪑🔥

Introduced flight types (e.g., Boeing, Airbus) selectable by admin during flight creation, saved in flights table. 📋✈️

Advanced seat allocation logic extended to respect flight type-specific seat layouts.

Frontend restricts seat selection flow based on chosen flight type for accurate bookings. 🎯

Modified airline_sitting table: added flight_type column for seat trigger logic consistency. 🗃️

Print & Display:

Tickets now show Flight Type info along with flight charges for transparency. 🎫💡

Seat Availability Logic Improved:

seats_avl.py updated for robust seat availability checks — correctly handling switched origins and destinations. 🔁🛋️

Example:

Flight AA12AA, Class: First, Total Seats: 12

Origin: Hyderabad, Destination: New York, Date: 01/Aug/2025

Booking status now accurately reflects seat availability—even if origin/destination are swapped.

Technical Enhancements:

Refined MySQL query logic in seat availability module for precision and error-free operation. 🔍💾

Entire codebase polished and structured for maximum robustness and dynamic behavior. ✅🔄

This version represents a major milestone with zero errors, seamless integrations, and enhanced user/admin experience! 🚀👏

🆕 Version 39 & 40 Updates

🚫 Journey Date is Today—No Cancellation Allowed

Tickets with a journey date matching today's date will not be displayed for cancellation and cannot be cancelled.

📑 Note:

To view, print, or download your ticket, please use My Journey History where you'll find and print or view your confirmed tickets.

🔗 To View Your Confirmed Tickets:

🖨️/👀 View/Print Confirmed Tickets

💺 Dynamic Seat Availability

Seats are now dynamically checked and displayed in real-time from the database for accuracy during booking and management.

🛠️ Enhanced Admin Functionalities

Admins can now view and manage flight seats layouts dynamically based on flight type for future flights.

✈️ Admin access lets you log into the backend database to change settings like flight seats, airports, class fares, flight fares, and more—admin-only features for enhanced control!

🆕 Version 41 Update

⚠️ Critical Security Fix at Payment Gateway

Previously, the payment gateway QR code (UPI/Pay QR) was generated on the frontend (HTML/JS).

This allowed anyone to open Inspect Element → Edit QR value → Pay to their own account — breaking payment integrity. 🔓❌

🔒 Fix Implemented:

Now, sessions securely store important payment data like total fare, txn_id, transaction mode on the backend.

The QR code is generated server-side with these session values, so it cannot be tampered with via frontend inspection.

✅ Secure Payment Flow:

When the user reaches the payment page, backend saves total_fare, txn_id, and trm in session/DB.

Frontend requests /generate_payment_qr with no sensitive parameters.

Server builds and returns a secure QR as an image.

Even if a user edits HTML, the QR is always generated from backend session data, not from the frontend input.

⚡ Benefits:

Users cannot modify QR code via Inspect Element.

Payment integrity is fully protected.

Your payment status polling (check_payment_status) continues to work smoothly with secure txn_id.

Version 42

🚀 Version 42 Updates

🛠️ Admin Side Validation:

Implemented and fixed issues where, while adding a new flight, the system now alerts the admin to check and fill in all required flight information. ⚠️ ✈️

🗓️ Dynamic Year Display:

Added automatic current year display for All Rights Reserved © [Year] on specific key pages:

HTML Index

User Login

Admin Login

Admin Dashboard

User Dashboard

Register Module

📌 The year is dynamically generated by the system without any manual update needed.

✈️ Flight Types from Database:

Major enhancement — flight types like Boeing 777, 747, Airbus A320, Domestic, etc., are now retrieved dynamically from the flight_types database table instead of being hardcoded in HTML.

This improves the admin page experience when adding new flights and cleans up frontend code significantly.

Overall, this version is more enhanced and advanced! 🎉 ✨

---------- End of Changelog ----------

## 🚀 Feature Developments

👨‍💼💻 Developed a comprehensive admin module to display all operations, including charges and flight types (flight type display is optional).

🛠️📊 Flight types are now dynamically fetched from database tables during the admin's flight addition process, enabling seamless and flexible handling. (Implemented in Version 42)

🪑🗓️ Managed and configured flight layouts using data from the seats_avl.py module linked with the flight layout database table. (Implemented in Versions 39 & 40)

⚠️ Note: Admin seat updates are done directly via backend MySQL database, no dynamic UI yet.

⏰✈️ Implemented and maintained accurate flight timings for departure and arrival schedules.

💳🔐 (Optional) Integrated user-side payment gateway supporting payment_mode=ACID for robust, transactional payment processing.

# Issues Encountered and Resolutions ⚠☑

During the development of the SKA Flight Booking System, several challenges arose, including:

- Data synchronization complexities between flight schedules and real-time seat availability, which were resolved by implementing efficient database transaction controls and caching mechanisms. 🔄🗄
- Payment gateway integration hurdles, especially ensuring secure, ACID-compliant transactions, overcame by adopting robust API protocols and thorough testing. 💳🔒
- User authentication security risks, mitigated by strengthening password policies, session management, and encryption techniques. 🔐🛡
- Handling scalability for growing user base and flight data, addressed by modular system design and optimizing query performance. 📈⚙

These solutions helped enhance the system's reliability, security, and performance, laying a strong foundation for future upgrades and maintenance.

⚠ ISSUES & BUGS DURING SKA AIRLINES APPLICATION DEVELOPMENT 🛫🐞

Issue 1: Journey Date vs Return Date 📅➡BACK

Problem: Return date must be after the journey start date.

Status: Resolved ✅ with JavaScript validation ensuring the return date is always later than the journey date.

Issue 2: Date Format Consistency 📅

Problem: Date formats should appear as 12/Jul/2024 for consistency and clarity.

Status: Fixed ✅ by enforcing date input and display formatting via JavaScript.

Issue 3: Round Trip Discount 🎟🔄

Problem: When trip mode is Round Trip, a discount of 1% should apply on the total trip cost.

Status: Resolved ✅ using JavaScript logic to automatically calculate and apply the discount.

Issue 4: Automatic Airport Name Retrieval ✈️🌃

Problem: Airport names were hardcoded and not dynamically fetched.

Status: Solved ✅ by fetching airport data from the database using Python (airport.py globally and respective routes.py modules for guest/admin).

Issue 5: Realistic Ticket Layout & Seat Allocation 🎫🪑

Problem: Seat assignment during booking was not automatic or realistic.

Status: Fixed ✅ with backend Python code implementing automatic seat allocation at booking time.

Issue 6: Reservation Data Transfer via Sessions 🔄📄

Problem: Booking details were not preserved properly across multi-step forms.

Status: Resolved ✅ by managing data flow through Flask sessions, maintaining state across booking steps.

Issue 7: Dynamic Passenger Selection 👴👧👦👴➕

Problem: Passengers couldn't book tickets for multiple people simultaneously.

Status: Fixed ✅ by introducing dynamic JavaScript forms allowing multiple passenger entries in one booking.

Issue 8: Quota Reservation Discounts 🎖️💸

Problem: Defence students discount (Rs 1000) not applied correctly, especially for round trips.

Status: Resolved ✅ — Defence quota discount now applies Rs 1000 for one-way and Rs 2000 for round trips.

Issue 9: Automatic Fare & Class Updates 💵⚙️

Problem: Fare details (base fare, quota fare, class fare) did not auto-update when adding passengers.

Status: Fixed ✅ with JavaScript dynamically recalculating and updating all fare components instantly.

Issue 10: MySQL Trigger on airline_reservation Table 🔔🗃️

Problem: Needed DB-level triggers to synchronize seat assignments with reservations.

Status: Resolved ✅ through a MySQL trigger, updating the airline_sitting table and displaying all PNRs with passengers accordingly.

Issue 11: Single PNR for Multiple Passengers 👥🎫

Problem: Passengers booking together did not share the same PNR.

Status: Fixed ✅ — multiple passengers now share one PNR with distinct seats during a single booking.

Issue 12: Realistic Plane Seat Layouts 💺🪑

Problem: Seat allocation was not class-based or reflecting realistic flight seating.

Status: Resolved ✅ by Python code dynamically managing seat layout based on flight class selection.

Issue 13: Ticket Download Instead of Printing 💾📄

Problem: Printing tickets was inconvenient and non-digital.

Status: Solved ✅ via an HTML template allowing users to download tickets to their devices instead of printing.

Issue 14: Correct Round Trip Charge Calculation 💰🔁

Problem: Charges were not doubling for round trips; quota discounts also miscalculated.

Status: Fixed ✅ through JavaScript validation ensuring base fare doubles for round trip selections and discounts adjust accordingly.

Issue 15: Auto-fetch User Name Using User Number 🆔➡️👤

Problem: Admin had to manually look up user names after entering user numbers.

Status: Resolved ✅ — When admin enters user number, the username is automatically fetched from the database dynamically (airline_users.py with proper routes to HTML). Also applied to admin's user booking history views.

Issue 16: User Number vs Name in Admin Audit 👤🔢

Problem: Need dynamic display of user name when entering user number during admin audit or booking history review.

Status: Resolved ✅ — Username now dynamically displays in the text box in admin ticket booking and user booking history audit.

Issue 17: Mapping User Booking History with Admin PNR Enquiry 🎟️🔍

Problem: Need to reduce redundant code by mapping user booking history to admin PNR enquiry based on user number input.

Status: Done ✅ — Module developed to hold all user ticket bookings linked with PNR, accessible via admin enquiry for quick audit.

Issue 18: Auto-Clear Journey & Return Dates via SQL Trigger or Query 🗓️⚙️

Problem: Need to automatically clear journey and return dates after completion.

Status: Business logic changed — Instead of deleting dates, jstatus is updated to "completed" once journey and return dates pass. ✅

Issue 19: Auto-Use Bank Account Details on Ticket Cancellation 💳🔁

Problem: Avoid asking user for bank account details again upon cancellation; refund should go automatically to the account used at booking.

Status: Implemented ✅ — Added account number column in airline_reservation table and automated refund process to same account at cancellation.

Issue 20: Fare Auto-Population in Guest/Index Ticket Booking Page 💸✨

Problem: Fare was not auto-fetched with base fare in index ticket booking.

Status: Resolved ✅ — Fare now auto-populates on the ticket booking page for guests and users.

Issue 21: Combined Plane Schedule for Guest, User, and Admin 👥🛩️

Problem: Separate plane schedule modules caused code duplication.

Status: Fixed ✅ — Developed unified module combining guest, user, and admin plane schedules for efficient code maintenance.

Issue 22: Ticket Cancellation Module for Admin & User with Refund 💸❌

Problem: Need cancellation modules for both admin and user with automatic refund crediting.

Status: Implemented ✅ — Both user and admin ticket cancellation with refund processing in place.

Issue 23: Dynamic Flight Name Display on Plane Schedule by Flight Number 🛬📝

Problem: Flight name should dynamically display when flight number is entered in scheduling.

Status: Done ✅ — Flight names now fetch and display dynamically on input of flight number.

Issue 24: Form Close Button Should Not Trigger Empty Popups ❌✖️

Problem: Closing forms with buttons caused unwanted empty popup forms.

Status: Fixed ✅ — Close buttons use correct type="button" and proper form attributes to avoid empty popups, plus UI fixes.

Issue 25: Global PNR Status Common for Index & User Pages 🎫🌐

Problem: Need common PNR status view usable by guest, user, and admin for downloading tickets and viewing bookings.

Status: Developed ✅ — Shared module allows common access yet restricts downloads to respective users only.

Issue 26: Add “New User? Please Register” Link on User Login Page 🔑🔗

Problem: Login page lacked direct link to registration for new users.

Status: Fixed ✅ — Hyperlink/button added redirecting new users directly to registration page.

Issue 27: Capitalize Passenger Names Automatically ✍️🔠

Problem: Passenger names were entered inconsistently with regards to case.

Status: Resolved ✅ — Python code converting passenger names to uppercase before saving.

Issue 28: Passenger Details Alert in Booking 🚨🛂

Problem: Users/admin need to be cautioned to enter passenger details exactly as per government ID.

Status: Fixed ✅ — Alert message implemented on booking page. Plans for master list integration for added data validation.

Issue 29: Confirmation Popup on Ticket Cancellation ⚠️ ❓

Problem: Users should confirm cancellation intent via alert popup.

Status: Done ✅ — Confirmation alert implemented before processing cancellation.

Issue 30: Combined User Journey History & Ticket Download Module 🎟️ 📜

Problem: Required module to show user journey list with direct ticket download without entering PNR again.

Status: Developed ✅ — Users access complete journey with ticket download options seamlessly.

Issue 31: Separate Modules to Sort CNF, CAN & Complete Journey Tickets 📂 ✅ ❌

Problem: Need modular separation for Confirmed (CNF), Cancelled (CAN), and full ticket journeys.

Status: Completed ✅ — Separate modules created with clear logic for respective ticket statuses.

Issue 32: User Sessions to Store Number, Name, Feedback, Mobile & Password Update 🔐 🔄

Problem: Data such as user number and name should be stored in session for easier retrieval and to enhance code maintainability.

Status: Implemented ✅ — User sessions now hold essential data, accessed via Python backend to maintain high data integrity.

Issue 33: Admin Side Ticket Cancellation Requires Extra User Account Handling 👴 💼 ❌

Problem: Admin ticket cancellation needs to capture and validate user account number properly.

Status: Resolved ✅ — Admin side cancellation module enhanced with user account number retrieval and validation.

Issue 34: Auto-Generate Fare Based on Distance (KM) ⛽ 💸

Problem: Fare calculation should dynamically reflect distance between airports (e.g., Rs 12 per KM).

Status: Implemented ✅ — SQL trigger created for automatic fare generation based on KM distance between origin and destination.

Issue 35: Seat Allocation Trigger with Cancelled Seats Fix 🪑🔄

Problem: Trigger logic not correctly handling cancelled seats (seats not properly freed/reassigned).

Status: Fixed ✅ — Corrected trigger logic to handle seat availability accurately on cancellations.

Issue 36: Cancelling Confirmed Tickets Fails for Multiple Passengers (>2) ❌👥

Problem: Ticket cancellation fails when more than two passengers booked under same PNR.

Status: Resolved ✅ — Fixed by replacing fetchone() with fetchall() in seat handling logic.

Issue 37: Journey Status Should Be “In Progress” (IP) Until Ticket Cancelled 🔄👍

Problem: Journey status not reflecting “In Progress” until ticket cancelation.

Status: Implemented ✅ — Python code updated to set journey status to “IP” during booking lifecycle.

Issue 38: Auto-Fetch Bank Account Number for Refunds 💳⏬

Problem: Bank account used in booking should auto-fill in cancellation refund process.

Status: Fixed ✅ — Implemented code to retrieve and use user’s bank account during refund.

Issue 39: Match Transaction Scenarios With Banking Logic 🏦📊

Problem: DB transaction records didn’t correctly reflect debit/credit flow for bookings and cancellations.

Status: Fixed ✅ — Updated transaction entries so that booking debits user account & credits airline account, cancellation reverses credit/debit accordingly.

Issue 40: Admin-Booked Ticket Cancellation Logic 🎫👴💼❌

Problem: Tickets booked by admin must only be cancellable by admin (counter ticket cancellation), without involving user number fetching.

Note: Admin retains power to cancel user-booked tickets.

Status: Resolved ✅ — Removed user number requirement for admin-booked ticket cancellations to streamline process.

Issue 41: Professional “All Rights Reserved” Footer Addition ©®

Requirement: Add a consistent footer on every page displaying:

SKA AIRLINES ALL RIGHTS RESERVED © ®

Status: Implemented ✅ — Footer with copyright and registered mark added site-wide.

Issue 42: Improve Code Reusability ♻️💻

Goal: Create common reusable packages for HTML, including JS scripts and CSS, to avoid duplication and ease maintenance.

Status: Completed ✅ — Modular includes implemented for reusable frontend assets.

Issue 43: Seat Allocation Trigger Fix for Multiple Cancelled Seats 🪑🔄

Problem: Trigger logic failed to reassigned cancelled seats when more than one seat is available for the same flight/class/journey.

Example: For First Class with 12 seats, 2 cancelled seats were not reassigned correctly.

Status: Fixed ✅ — Trigger logic corrected to allocate cancelled seats properly and in order.

Issue 44: Phase 2 Development Module Summary 🚀✨

SKA Airlines is building dynamic & modern features:

🧾 QR Code on Ticket PDF: Fast airport check-in & digital verification (Generation and Base64 encoding done)

📧 Email System / WhatsApp Notifications: Booking and cancellation updates

💳 QR Code Payment Gateway (Simulated): Realistic payment simulation via QR (done)

📊 Graphical Admin Dashboard (Optional): Visual insights for bookings, revenue, cancellations

🪄 Advanced Seat Allocation by Flight Type: Flight types (Boeing, Airbus) with seat configs per class; auto seat range setup done

🛠️ Common DB Configuration File: Centralized DB connection (db_config.py)

🔁 Dynamic Fare Calculation from DB: Fare fetching and calculation done dynamically, avoiding static HTML fares

🧮 Admin Fare Update & Visualization Page: Admin can update fares by class/quota/km with optional fare increase calculations

🛜 Dynamic Domain/IP Configuration: Application accessible via domain instead of raw IP (done)

Issue 45: QR Code on Ticket Implementation 🎟️ ■

Requirement: Develop an importable Python module to generate QR codes containing ticket reservation details, embedded in ticket layout after booking.

Status: Completed ✅ — QR code generation module implemented and integrated in tickets successfully.

Issue 46: QR Code for User-Side Payment Gateway 💳 ■

Requirement:

Develop a dedicated QR code generator module that encodes transaction details like fare and bank URL.

When the user scans the QR, it opens a bank authentication window for login and payment authorization.

Upon successful payment, the bank backend notifies the ticketing system, triggering ticket generation and payment confirmation.

Maintain a transaction ID (txn_id) tracked in the qr_payment_status table.

Pass user's bank account ID (acid) into the airline_reservation table for automated refunds on cancellations.

Flow:

ticket_user4.html (QR module) → uselqr.jsp (bank) → accept_qr_payment.jsp → authorised_payment.jsp → qr_payment_gateway.py → qr_payment_su.html → ticket_userdb.py → ticket_user_su.html

Status: Debugging completed and issue resolved ✅.

Issue 47: Email-Based Ticket Confirmation & Cancellation Notifications 📧✈️

Requirement:

Implement an email system that automatically sends booking or cancellation confirmations to users with ticket details (flight number, passenger name, class, journey date, etc.).

Add an email_id column to the airline_users table to store user emails.

Create a dedicated email module linked to user-side booking and cancellation processes.

For admins, sending emails on ticket booking is optional but recommended.

Status: Implemented and fully operational ✅.

Issue 48: Centralized Database Handling & Security 🗄️🔐

Requirement:

Create centralized DB configuration for MySQL and Oracle to avoid exposing database credentials across multiple files/pages.

Facilitate easy maintenance and improved security.

Status: Successfully implemented ✅.

Issue 49: Including Flight-Specific Charges in Fare Calculations 💵✈️

Requirement:

Include individual airline flight charges dynamically during booking fare calculation.

Status: Completed and integrated successfully ✅.

Issue 50: Dynamic Fare Calculation from Database Based on Class & Quota 🧮🎟️

Requirement:

Fetch fares dynamically from DB tables storing class fares and quota discounts (e.g., Defence discount).

Allows admins to update fares in the DB without changing application code, enhancing business flexibility.

Example: First class fare = 1500 ₹.

Status: Fully implemented and tested ✅.

Issue 51: Reflecting Bank Account ID in QR Payment Gateway Flow 💳🔁

Requirement:

Add ac_number column in the qr_payment_status table to hold user bank account info.

After payment confirmation, insert the bank account ID into the corresponding record in the airline_reservation table for reconciliation and refund processing.

This solves cancellation/refund issues when payment mode is QR code.

Status: Logic corrected and issue resolved ✅.

Issue 52: Email Delivery Error on Ticket Cancellation Confirmation ✉️❌

Problem:

Cancellation confirmation emails were failing due to errors in passenger list formatting (specifically, incorrect access like p['name']).

The fix involved correctly fetching email and passenger details from the airline_users table for reliable email content.

Status: Bug fixed and emails are now delivered successfully ✅.

Issue 53: QR Code Missing in Ticket Cancellation Emails 🎟️✉️

Problem: When a user cancels a confirmed ticket, the email sent does not include the QR code—only textual information is delivered.

Root Cause: QR generation was implemented only for confirmation emails (email_cnf_can.py), not for cancellations.

Resolution: Redesigned email module to include QR codes for both ticket confirmations and cancellations.

Status: Done ✅ — QR code now generates correctly in cancellation emails.

Issue 54: Duplicate Emails Sent on Browser Refresh 🌐🔄📧

Problem: Refreshing the endpoint page causes the system to resend booking/cancellation emails again, causing duplicates.

Fix: Placed email-sending logic before success methods in backend Python code for user and admin booking/cancellation routes, preventing resends on page refresh.

Status: Resolved ✅.

Issue 55: Error Pages on Browser Refresh after Confirmation 🌐❌

Problem: Refreshing the confirmation pages after ticket generation causes error pages, particularly with cancellation endpoints for both user and admin sides.

Fix: Adjusted flow so that database confirmation data is properly passed to success pages, preventing errors on refresh. Status: Fixed ✅.

Issue 56: Add Email Column to airlines_reservation Table for Admin Use 📧🗄️

Requirement: Store users' email addresses in the reservations table to allow admins to send email notifications during ticket bookings.

If user email is null, no email sent (neutral handling).

Status: Implemented successfully ✅.

Issue 57: Fetch User Email ID during Admin Ticket Cancellations ✉️⚙️

Requirement: Retrieve stored email from airlines_reservation table to send cancellation notices when admin cancels tickets.

Status: Implemented and working ✅.

Issue 58: Insert User Email into Reservation Table during User Booking 📧🎟️

Requirement: Store email in email_user column during user-side bookings for use during admin-side cancellation notifications.

Benefits: Enables admins to notify users if admin cancels user-booked tickets, improving communication and trust.

Status: Completed ✅.

Issue 59: Dynamic Advanced Seat Allocation by Flight Details type 🪑✈️

Requirement: Implement dynamic seat allocation factoring in flight number, name, class, journey date, origin, destination, and flight type (e.g., Boeing seats allocation to Boeing flights).

Current Status:

Existing 3 scenarios had limitations; seats_avl.py restricts reservation seats at frontend.

Old trigger allocates seats based on flight type but needs enhancement to handle larger seat configurations (i.e., Boeing 747 FC with 14 seats, etc.).

Next Steps:

Upgrade trigger logic to handle larger seat layouts per aircraft type.

Add submodule in admin for selecting aircraft type while adding a flight, storing flight type in the flights table.

Store flight_type also in airlines_sitting table for accurate seat availability tracking.

Pass flight type info in tickets and emails (including QR codes) during reservation to ensure consistency.

Outcome:

A single, robust trigger plus frontend logic (seats_avl.py) enables seamless and accurate seat allocation while handling cancellations dynamically.

Status: Fully implemented and all related issues resolved ✅.

Issue 60: Display Flight Type & Flight Charge on Tickets ✈️💰

Requirement: Add Flight Type and Flight Charge details visibly on the ticket, QR code, and email confirmations.

Status: Fixed ✅ — Flight Type and Flight Charge now clearly displayed on tickets and included within QR codes and emails.

Issue 61: Increase Seat Allocation Trigger for Largest Flight Layouts 💺✈️

Requirement: Enhance the seat allocation database trigger to support the highest-capacity aircraft like Boeing 747, which have large seat layouts.

Status: Completed ✅ — The trigger was successfully scaled to handle the largest layouts seamlessly. Impressive performance ensured! 💪

Issue 62: Add Exact Seat Layouts in seats_avl.py for Frontend Validation 🪑⚙️

Requirement: Integrate precise seat layouts within the seats_avl.py module to validate seat availability and booking on the frontend, preventing overbooking or invalid selections.

Status: Resolved ✅ — The program now validates seat layout accurately during booking.

Issue 63: Add Flight Type Column in airlines_reservation Table ✈️📊

Requirement: Add a new column in the airlines_reservation DB table to store the flight type, enabling more accurate data retrieval especially during ticket cancellation workflows.

Status: Done ✅ — Flight Type column added and integrated successfully.

Issue 64: Correct Display of Seat Availability with Origin-Destination Consideration 💺🌐

Problem:

When origin and destination are swapped (e.g., Delhi ↔ New York on flight AA12AA), the seat availability was incorrectly shown as the same for both directions.

Requirement: Seat availability must reflect exact journey direction and other parameters like journey date, flight type, etc.

Fix: Updated seats_avl.py to consider origin and destination properly during seat availability checks, giving accurate display of available seats.

Status: Issue fixed ✅ — Seat availability now reflects exact journey data, ensuring correctness and user trust.

**Issue 65: Tickets Displaying Past Journey Dates** ✈️🗓️

**Requirement:**
Restrict displaying **past journey dates** in user ticket history using the query:

sql

**SELECT** * **FROM** airline_reservation

**WHERE** ausr = %s AND **status**='CNF' AND jstatus='IP'

AND STR_TO_DATE(jd, '%d/%b/%Y') >= CURDATE() //For the Todays date it will not allow to cancel the ticket

To print of view the ticket then user need to Go to the My Account/ My Journey History

**ORDER BY** PNR **DESC**, (uno,)

**Status:** Done ✅ — Successfully resolved the issue. Past journey tickets are no longer displayed.

**Issue 66: Dynamic Seat Allocation Fetching in seats_avl.py** 💺🔄

**Requirement:**
Fetch available seats dynamically from the database table aircraft_seats_layout.
Admins will be able to update seat details via a frontend HTML module under development.

**Status:** Done ✅ — Dynamic seat availability fetching implemented and working smoothly.

Issue 67: Display Flight Class and Quota Fares 💰✈️

Requirement:
Display fare details including flight class and quota charges by accessing backend database table with secure, read-only admin credentials. Database user creation example:

sql

CREATE USER 'airline_admin'@'localhost' IDENTIFIED BY 'airlineadmin';

GRANT SELECT ON airline.* TO 'airline_admin'@'localhost';

FLUSH PRIVILEGES;

For seat management by flight type, the following privileges are required:

sql

GRANT SELECT, INSERT, UPDATE, DELETE ON airline.my_table TO 'admin'@'localhost';

FLUSH PRIVILEGES;

(Note: DELETE permission is optional. If issues arise, airline admin should coordinate with the MySQL database developer/admin.)

**Status:** Ongoing — Database permissions configured to securely manage fare and seat data.

🐞 Issue 68: QR Code on Payment Gateway Editable via HTML Inspect 🪲

There was a major bug where the QR code on the payment gateway could be manipulated by editing the HTML elements through Inspect Element.

For example, the total fare displayed as Rs. 12,000/- 💰 could be changed to Rs. 10/- by tampering with the HTML, breaking payment security. 🔓❌

🔒 Fix Implemented: Secure Server-Side QR Code Generation Using Sessions

Instead of generating QR code values from editable frontend HTML/JS, all sensitive values come directly from backend sessions:

python

txn_id = session.get('txn_id')

total_fare = session.get('total_fare')

trm = session.get('trm', 'Db SKA-AIRLINES')

if not txn_id or not total_fare:

```
    abort(400, "Invalid session or missing transaction")
ip_address = get_local_ip()
# Secure QR generation with server-side values only
bank_url = f"http://{ip_address}:8181/bank/uselqr.jsp?amount={total_fare}&mode={trm}&txn_id={txn_id}"
img = qrcode.make(bank_url)
buffer = io.BytesIO()
img.save(buffer, format='PNG')
buffer.seek(0)
return send_file(buffer, mimetype='image/png')
```

Simplified flow in routes.py:

```
python
@user_dashboard_bp.route('/ticket_user4', methods=['GET', 'POST'])
@login_required
def ticket_user4():
    step4_data = session.get('step4_data', {})
    combined_data1 = {**step4_data}
    txn_id = str(uuid.uuid4())[:8]
    total_fare = combined_data1.get("total_fare1")  # safer than dot attr
    trm = "Db SKA-AIRLINES"

    # Store securely in session
    session['txn_id'] = txn_id
    session['total_fare'] = total_fare
    session['trm'] = trm

    return render_template('user/ticket_user4.html', txn_id=txn_id, combined_data1=combined_data1)
```

In payment_gateway_Qr.py, the QR is generated using session values only, ensuring security.

Backend trusts only session data, so end users cannot modify QR code contents via Inspect Element. 🛡️ ✅

⚠️ Summary of Critical Security Fix

Payment QR generation moved from insecure frontend (HTML/JS) to secure backend sessions.

Users cannot tamper with QR values by inspecting or editing HTML.

Payment integrity and transaction safety fully restored.

Payment status polling (check_payment_status) continues to work smoothly using secure txn_id. 🔄💳

🐞 Issue 69: Tickets Displaying Past Journey Dates ✈️🗓️

Requirement:

Restrict displaying past journey dates in user ticket history using the following query:

sql

SELECT * FROM airline_reservation

WHERE ausr = %s AND status='CNF' AND jstatus='IP'

AND STR_TO_DATE(jd, '%d/%b/%Y') > CURDATE()

ORDER BY PNR DESC, (uno,)

Status: Done ✅ — Successfully resolved the issue. Past journey tickets are no longer displayed in the user's ticket history.

Issue 70: Add Hyperlink for Ticket Cancellation Redirect 🔗🛬

When users want to cancel a ticket, display a message like:

To view, print, or download your ticket, please use My Journey History, where you can find, print, or view your confirmed tickets.

🔗 To View Your Confirmed Tickets:

🖨️ / 👀 [View/Print Confirmed Tickets]

Issue Resolved: ✅

A hyperlink was added to smoothly navigate users to their journey history for ticket printing or downloading.

🐞 Issue 71: Automatic Display of Current Year on Web Pages 🗓️✨

The year on the web pages should automatically update based on the system year, eliminating the need for manual edits annually.

This includes replacing hardcoded years like “All Rights Reserved 2025” with a dynamic year display.

✅ Implementation Details:

Created a separate JavaScript file with the code:

xml

<script>document.getElementById('year').textContent = new Date().getFullYear();</script>

Imported this file where needed across the project pages.

Applied only in specific key pages/modules:

HTML Index

User Login

Admin Login

Admin Dashboard

User Dashboard

Register Module

📌 Current Year is dynamically displayed without manual intervention.

🛠️ Impacted modules:

Index, admin login, user login, welcome admin, welcome user, guest ticket, user register, guest plane schedule, news and events, feedback.

⚠️ Excluded: No need to add this in ticket pages as copyright year is already declared.

🐞 Issue 72: Dynamic Retrieval of Flight Types from Database 🛬💻

Requirement:

Create a new database table named flight_types to store different types of flights dynamically. ✈️

This new table replaces the hardcoded flight types on HTML pages where admins add new flights.

The table should be editable, allowing admins to add new flight types as needed in the future. 🪴

✅ Status: Issue fixed — Flight types are now dynamically retrieved from the database table, ensuring flexibility and easier future updates.

🐞 Issue 73: Flight Class and Quota Validation While Adding New Flight ✈️✔️

Problem:

When the admin adds a new flight, if they miss selecting the flight class or quota, the system was accepting null values, causing data inconsistency. ❌

🔧 Fix Implemented:

Added JavaScript validation to check that the admin selects flight class and quota (along with other mandatory fields).

If any required field is missed, the admin is immediately notified to fill it properly before submitting. ⚠️🛑

✅ Status: Issue resolved — Admin cannot submit incomplete flight details without selecting essential flight class and quota.



## Final Note:

The entire app's latest stable and corrected version is Version 42 🚀 — incorporating all these fixes and improvements.

# Conclusion 🎯✈️

The SKA Flight Ticket Booking System represents a significant step forward in providing a modern, efficient, and secure online platform for flight reservations. It addresses the critical needs of today's travelers by offering a seamless booking experience that is accessible anytime and anywhere, reducing the hassles of traditional ticketing methods. 🛫🌐

With features like secure user authentication, flexible payment methods including QR code and bank account payments, and comprehensive admin controls for flight, fare, and ticket management, the system ensures both user convenience and operational effectiveness. 👤🔐💳 The integration of real-time seat availability and dynamic flight management ensures customers receive up-to-date information, empowering them to make informed travel decisions. 🪑📊

Moreover, the system design emphasizes strong security measures, data integrity, and user privacy, which are essential in maintaining trust and reliability in online transactions. The modular architecture, powered by Python Flask and MySQL, facilitates scalability and easy maintenance, making the system future-ready. 🛡️🧩

Looking ahead, planned feature enhancements such as dynamic flight type fetching, advanced seat layout configuration, and robust payment gateway integration with ACID compliance will elevate the platform's flexibility and transactional security. These advancements will not only improve administrative efficiency but also enhance the overall user experience, making flight booking more personalized and responsive to customer needs. 🚀💼

In conclusion, the SKA Flight Ticket Booking System is well-positioned to transform how airlines and passengers interact in the digital age. By blending technology, usability, and security, it offers a comprehensive solution that supports the evolving dynamics of air travel, ensuring smoother journeys and greater satisfaction for all stakeholders involved. 🌎✨

# Future Look of SKA Flight Ticket Booking System 🚀✨

## Feature Developments & Enhancements 👩‍💼💻

- Accurate Flight Timing Implementation:
  Ensure precise tracking and maintenance of flight departure and arrival times for all flights, enhancing the user booking experience and operational reliability. ⏰✈️
- Enhanced Payment Gateway Support (Optional):
  Integrate the user-side payment gateway with payment_mode=ACID support to provide strong transactional guarantees, making payment processes more robust and secure. This supports reliable online payments across multiple modes. 💳🔒

These future advancements will significantly improve the system's flexibility, usability, and security, keeping SKA Flight Ticket Booking System competitive and responsive to user needs.

## Author's Note ✍

This entire project, SKA Flight Ticket Booking System, has been solely designed, developed, and documented by Adepu Sai Kiran. All concepts, code, designs, and content reflect my personal effort and dedication to delivering a comprehensive and high-quality flight booking platform.

## Contact Information 📧

For any questions, feedback, or collaboration related to the SKA Flight Ticket Booking System, please feel free to reach out to me:

Full Stack Developer Name: Adepu Sai Kiran
GitHub: Github.com/SaiKiran-Adepu
Website: https://saikiran-adepu.github.io/SKA-Re/

I am happy to assist with any inquiries about the project or explore opportunities for further development and teamwork.

## Acknowledgements 🎉

**I sincerely thank my mentors, family members, friends, and Guruji for their constant support and encouragement throughout the development of this project.**

Thank you for reviewing the SKA Flight Ticket Booking System project documentation.



THE END