Full Stack Development with MERN

1. Introduction

- **Project Title**: Flight Booking APP
- **Team Members**: Chandra Kanth. Jinka
  A. Surya Teja
  S. Koushik
  A. Anurag Reddy

2. Project Overview

- **Purpose**

The VIT Flights booking system is designed to streamline and enhance the process of booking flights for users. By incorporating distinct interfaces for admins, flight operators, and users, the system aims to provide a seamless experience for all stakeholders involved. The primary goals are to simplify flight management, improve booking efficiency, and ensure a user-friendly experience for customers.

Goals

- ➤ **Efficiency:** Streamline the flight booking process to reduce the time and effort required for both customers and flight operators.
- ➤ **User Experience:** Provide an intuitive and easy-to-navigate interface for users to book flights.
- ➤ **Management:** Enable admins and flight operators to manage flights, schedules, and bookings effectively.
- ➤ **Reliability:** Ensure that the system is robust and reliable, minimizing downtime and errors.

- **Features**

◊ **Admin Page**

- ➤ **Dashboard:** Overview of system performance, user statistics, and recent activity.
- ➤ **User Management:** Add, edit, or remove users (customers and flight operators) from the system.

➤ **Flight Management:** Create, update, or delete flights, including details such as destination, departure time, and pricing.

◊ **Flight Operator Page**

➤ **Flight Schedule Management:** View and manage flight schedules, including changes and cancellations.
➤ **Booking Management:** Oversee and manage user bookings, including seat assignments and special requests.

◊ **User Page**

➤ **Flight Search:** Search for available flights based on destination, date, and other criteria.
➤ **Booking:** Book flights, select seats, and make payments securely.
➤ **Booking History:** View past bookings and print tickets or receipts.
➤ **Notifications:** Receive updates on flight status, special offers, and other relevant information.
➤ **Profile Management:** Manage personal information, payment methods, and preferences.

3. Architecture

Frontend:

- Framework: React.js
- Structure: Component-based for reusability and maintainability.
- State Management: React Context API or Redux.
- Routing: React Router for SPA experience.
- UI Libraries: Material-UI or Bootstrap.
- Form Handling: Formik and Yup.
- API Integration: Axios or Fetch API.

Backend:

- Framework: Node.js and Express.js
- Structure: Modular for specific functionalities (user management, bookings).
- API Endpoints: RESTful APIs.
- Middleware: Authentication (JWT), logging, error handling.

Database:

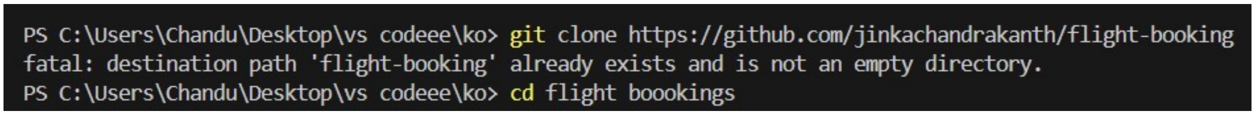
- Database: MongoDB with Mongoose
- Schema Design:
    - Users: Personal details, credentials, settings.
    - Flights: flight_id , flight_number , airline , origin , destination , departure_time
    - Bookings: booking_id , user_id , flight_id , booking_date , total_price
- Interactions: CRUD operations, indexing for performance, references for data consistency, ACID transactions.

4. Setup Instructions Prerequisites:

- Node.js (version 14.x or later)
- MongoDB (version 4.x or later)
- Git
- npm or yarn (package manager)

Installation:

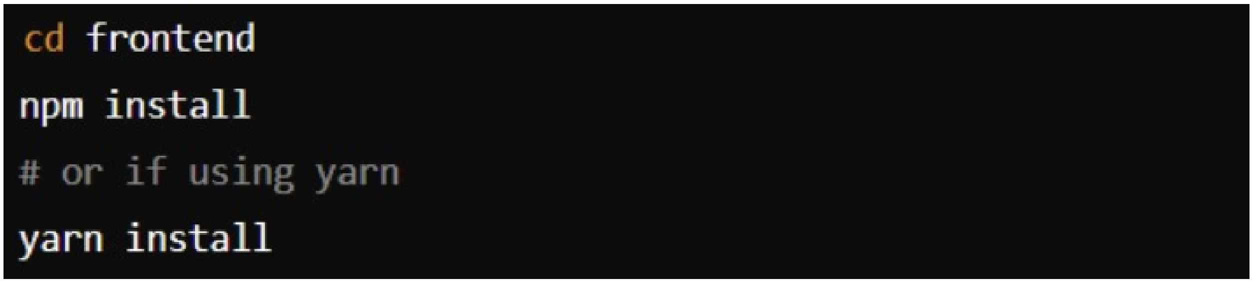
1. Clone the Repository:



```
PS C:\Users\Chandu\Desktop\vs codeee\ko> git clone https://github.com/jinkachandrakanth/flight-booking
fatal: destination path 'flight-booking' already exists and is not an empty directory.
PS C:\Users\Chandu\Desktop\vs codeee\ko> cd flight boookings
```

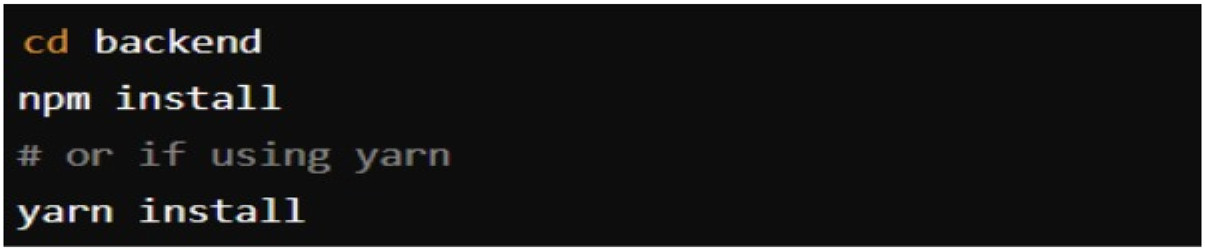
2. Install Dependencies:

For the frontend:



```
cd frontend
npm install
# or if using yarn
yarn install
```

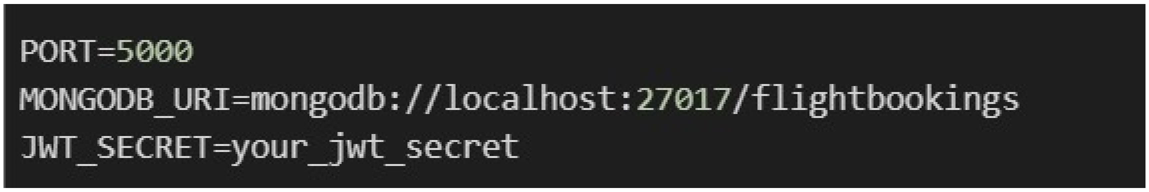
For the backend:



```
cd backend
npm install
# or if using yarn
yarn install
```

3. Set Up Environment Variables:

Create a .env file in the backend directory with the following contents:



```
PORT=5000
MONGODB_URI=mongodb://localhost:27017/flightbookings
JWT_SECRET=your_jwt_secret
```

4. Run MongoDB: Make sure MongoDB is running on your system. You can start it with:

5. Start the Backend Server:

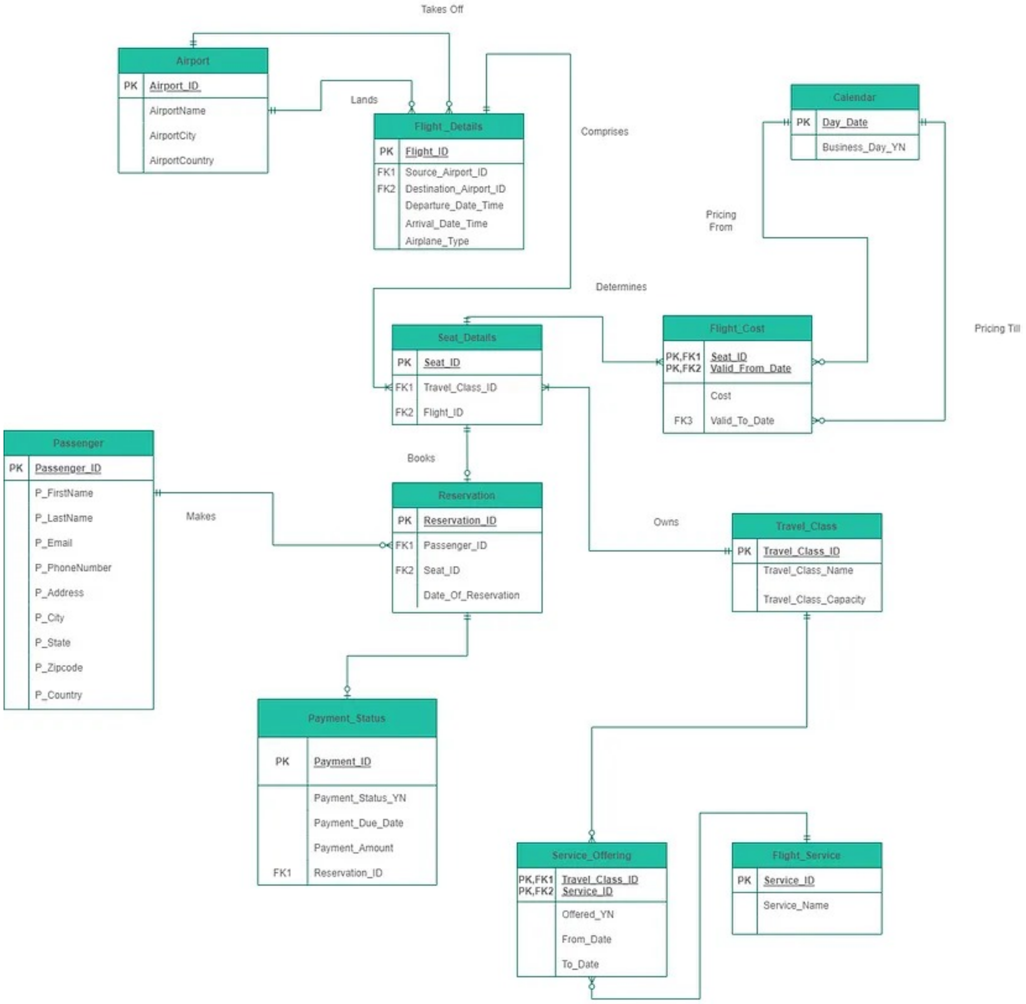
6. Start the Frontend Development Server:

7. Access the Application:

Open your browser and navigate to http://localhost:3000 to access the frontend. The backend API will be running on http://localhost:5000.

5. Folder Structure

● Clientside and Serverside:



6. Running the Application:

commands to start the frontend and backend servers locally.

- Frontend: npm start in the client directory.
  1. Navigate to the frontend directory.
     Command: cd client
     npm start.
- Backend: npm start in the server directory.
  2. Navigate to the Backend directory.
     Command: cd server
     npm start.

7. API Documentation

1. User Registration and Authentication

- Endpoint:/api/auth/register

- Method:POST

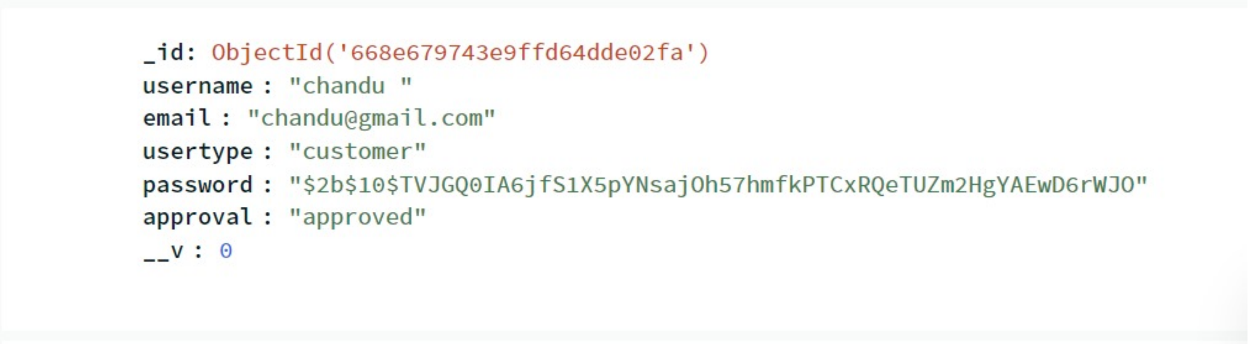
- Parameters:

  name (string) - Required

  email (string) - Required

  password (string) - Required

Example Request:



```
_id: ObjectId('668e679743e9ffd64dde02fa')
username : "chandu "
email : "chandu@gmail.com"
usertype : "customer"
password : "$2b$10$TVJGQ0IA6jfS1X5pYNsajOh57hmfkPTCxRQeTUZm2HgYAEwD6rWJO"
approval : "approved"
__v : 0
```

2. Flight Details

- Method:POST
- Parameters:

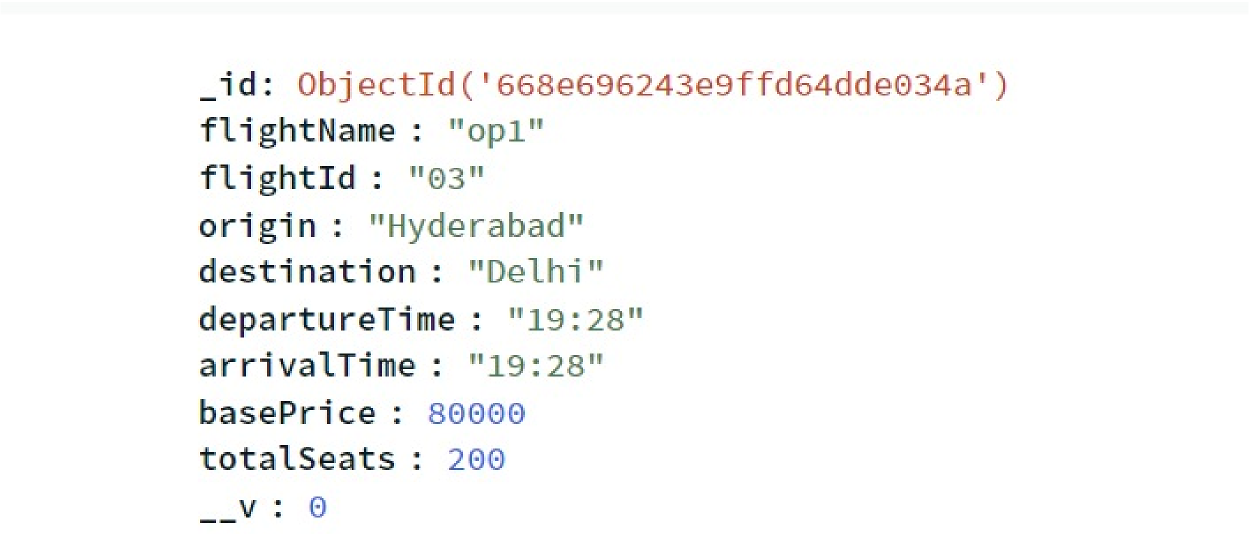
    email (string) - Required

    password (string) - Required

    origin(string) - Required

    Destination(string) - required

Example Request:



```
_id: ObjectId('668e696243e9ffd64dde034a')
flightName : "op1"
flightId : "03"
origin : "Hyderabad"
destination : "Delhi"
departureTime : "19:28"
arrivalTime : "19:28"
basePrice : 80000
totalSeats : 200
__v : 0
```

2. Booking Details

- Method:POST
- Parameters:

    email (string) - Required

    password (string) - Required

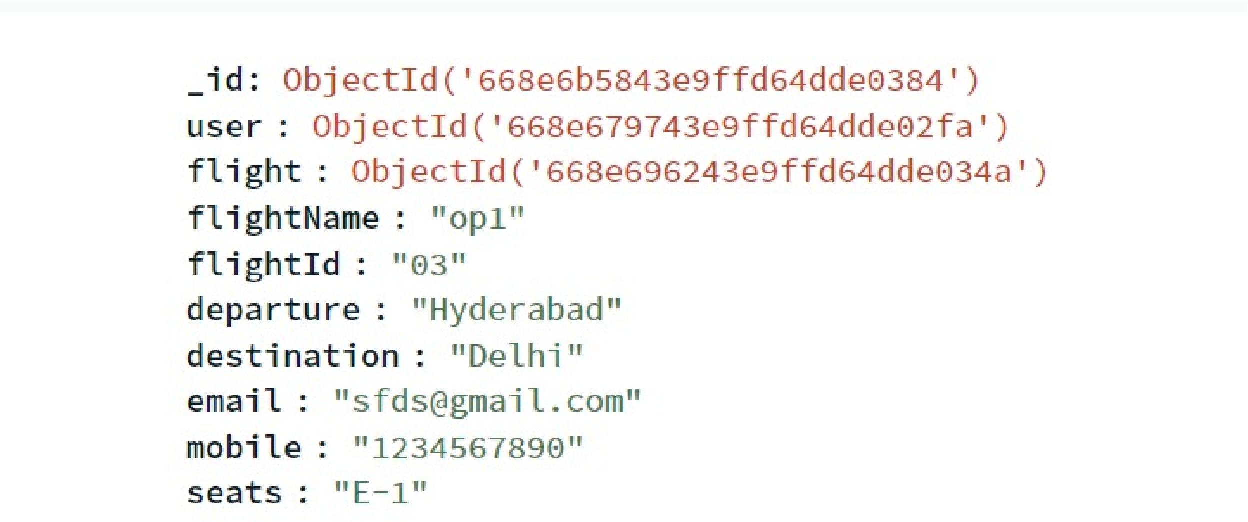
Flight ID: Required

Flight Name: Required

Departure: Required

Destination: Required

Example Request:



```
_id: ObjectId('668e6b5843e9ffd64dde0384')
user : ObjectId('668e679743e9ffd64dde02fa')
flight : ObjectId('668e696243e9ffd64dde034a')
flightName : "op1"
flightId : "03"
departure : "Hyderabad"
destination : "Delhi"
email : "sfds@gmail.com"
mobile : "1234567890"
seats : "E-1"
```

8. Authentication :

In the flight booking app, authentication and authorization are critical components for ensuring secure and efficient access to user-specific features and data.

Authentication

1. User Registration:

- Users can register through a form, Gmail, or LinkedIn. During registration, user details are captured and stored securely in the database.
- Passwords are hashed using a secure hashing algorithm (e.g., bcrypt) before being stored.

2. Login Process:

- Users log in using their email/username and password, or via third-party authentication (Gmail, LinkedIn).

- For form-based login, the provided credentials are verified against the stored hashed password.
- For third-party login, OAuth 2.0 protocol is used to authenticate users via Gmail or LinkedIn.

3. Token-Based Authentication:

- Upon successful login, a JSON Web Token (JWT) is generated and issued to the user.
- The JWT contains encoded information about the user and has a limited validity period.
- The JWT is signed using a secret key known only to the server to ensure its integrity.

4. Token Storage:

- The JWT is stored on the client side, typically in local storage or a secure HTTP-only cookie.

5. Subsequent Requests:

- For subsequent requests, the client includes the JWT in the Authorization header (e.g., `Authorization: Bearer <token>`).
- The server validates the token, ensuring it is not expired and has not been tampered with.
- If the token is valid, the server processes the request and provides access to the requested resource.

Authorization

1. Role-Based Access Control (RBAC):

- Different user roles (e.g., admin, user, guest) are defined with specific permissions.
- User roles are assigned during registration or by an admin.

2. Access Control:

- Each API endpoint is protected by middleware that checks the user's role and permissions.
- The middleware extracts the JWT from the request header, validates it, and decodes the user information.
- Based on the user’s role and permissions, the middleware grants or denies access to the endpoint.

3. Sensitive Operations:

- Operations such as modifying bookings, canceling flights, or accessing payment

information require higher levels of authorization.
- Additional checks are performed to ensure that the user has the necessary permissions.

Sessions

1. Session Management:

- While JWTs are used for stateless authentication, session management can be implemented for specific use cases (e.g., tracking user activities, storing temporary data).
- Sessions are stored on the server side, typically in a database or in-memory store like Redis.

2. Session Expiry:

- Sessions have a timeout period after which they expire and the user must re-authenticate.
- The session timeout can be refreshed on user activity.

Security Considerations

1. Secure Communication:

- All data transmission between the client and server is encrypted using HTTPS.

2. Token Expiry and Refresh:

- JWTs have an expiration time to limit the risk of token misuse.
- Refresh tokens can be issued to allow users to obtain a new JWT without re-authenticating.

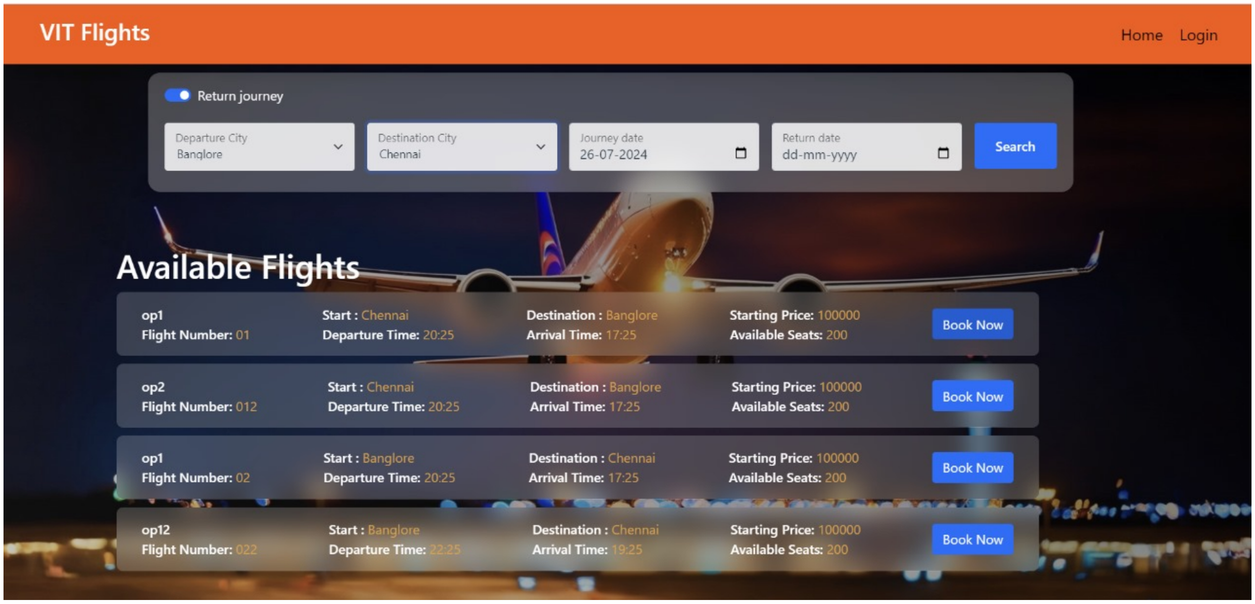
3. Protection Against CSRF:

- CSRF tokens are used to protect against cross-site request forgery attacks, especially if cookies are used for storing JWTs.

4. Rate Limiting and Throttling:

- Rate limiting is applied to authentication endpoints to prevent brute force attacks.

9. User Interface



VIT Flights

Home Login

Return journey

Departure City
Banglore

Destination City
Chennai

Journey date
26-07-2024

Return date
dd-mm-yyyy

Search

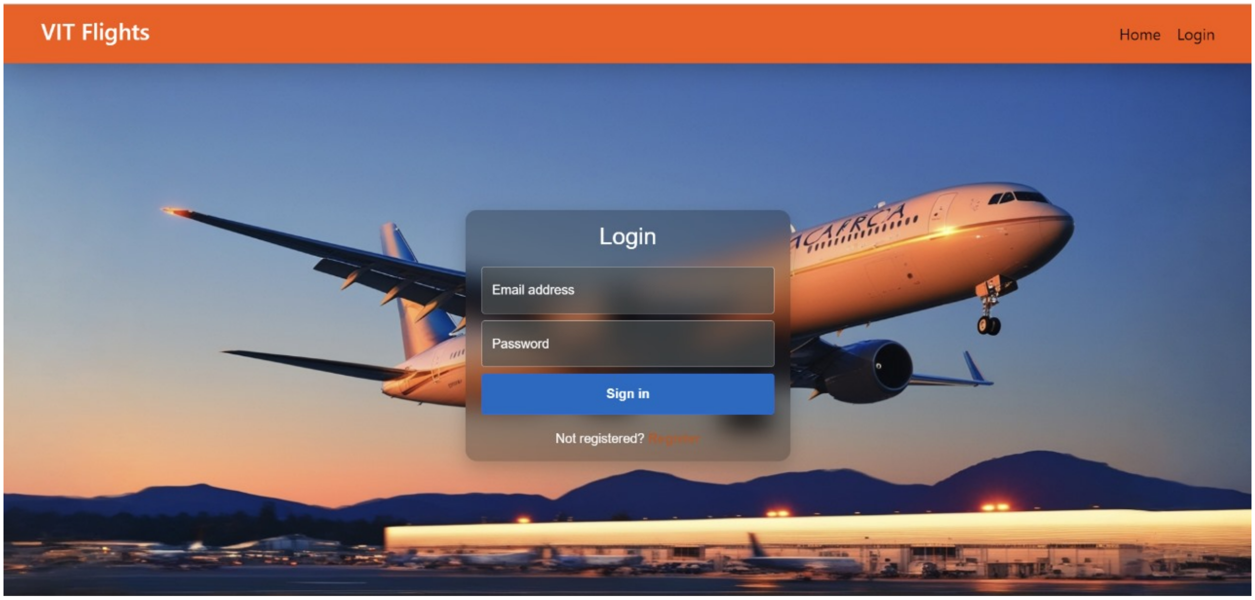
Available Flights

op1
Flight Number: 01
Start : Chennai
Departure Time: 20:25
Destination : Banglore
Arrival Time: 17:25
Starting Price: 100000
Available Seats: 200
Book Now

op2
Flight Number: 012
Start : Chennai
Departure Time: 20:25
Destination : Banglore
Arrival Time: 17:25
Starting Price: 100000
Available Seats: 200
Book Now

op1
Flight Number: 02
Start : Banglore
Departure Time: 20:25
Destination : Chennai
Arrival Time: 17:25
Starting Price: 100000
Available Seats: 200
Book Now

op12
Flight Number: 022
Start : Banglore
Departure Time: 22:25
Destination : Chennai
Arrival Time: 19:25
Starting Price: 100000
Available Seats: 200
Book Now



VIT Flights

Home Login

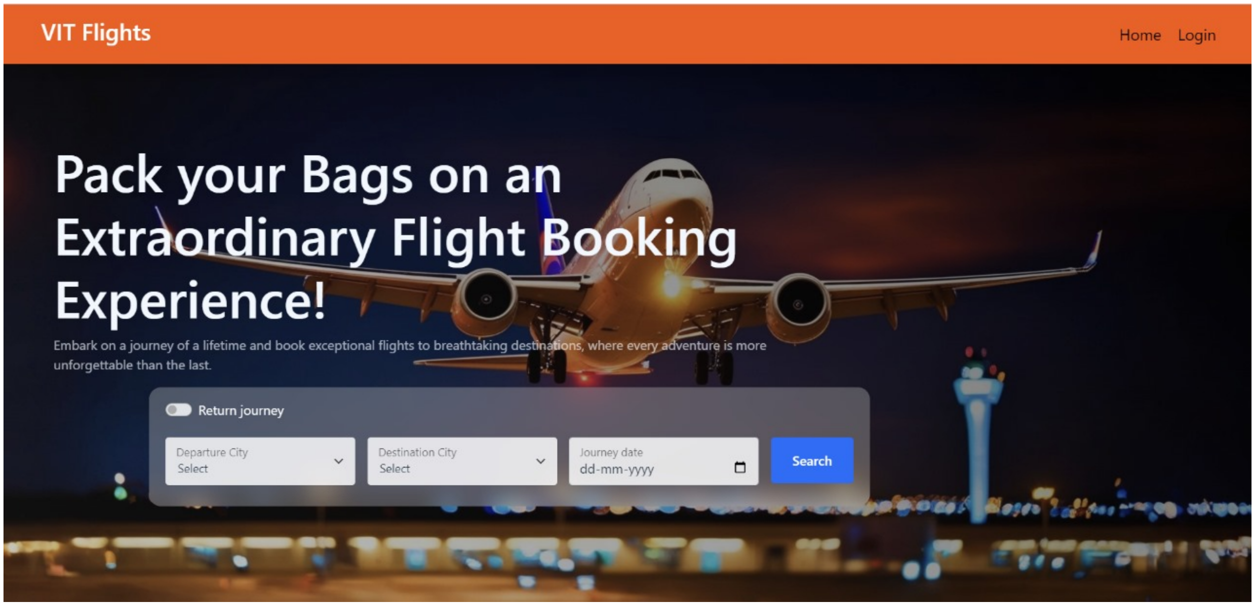
Login

Email address

Password

Sign in

Not registered? Register



Pack your Bags on an Extraordinary Flight Booking Experience!

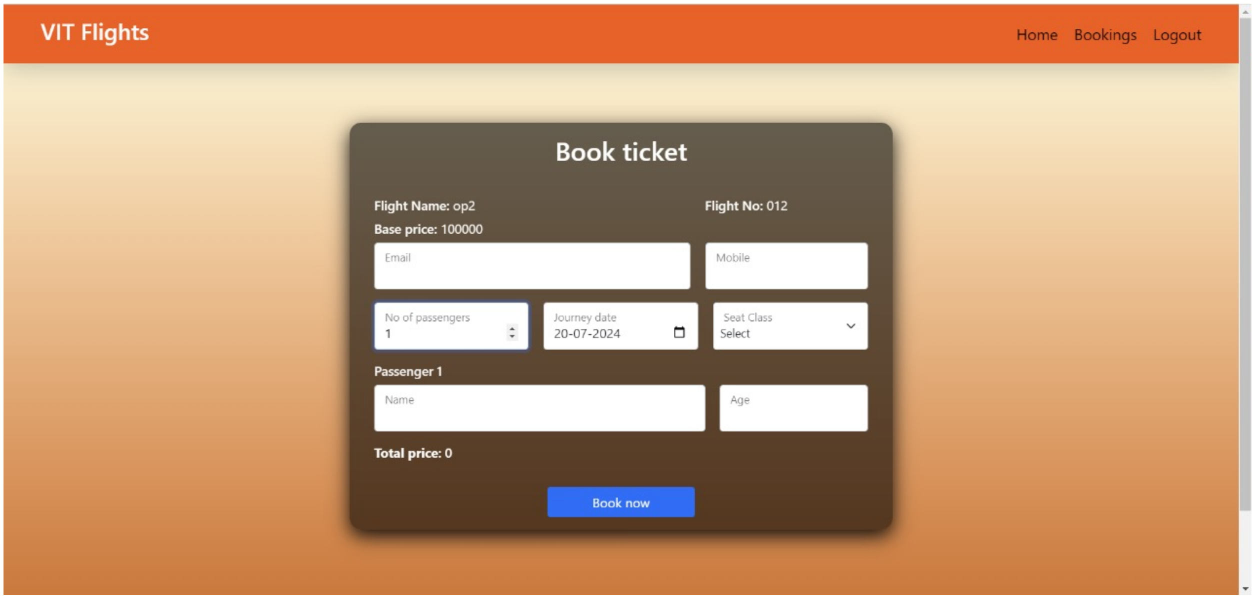
Embark on a journey of a lifetime and book exceptional flights to breathtaking destinations, where every adventure is more unforgettable than the last.

Return journey

Departure City: Select

Destination City: Select

Journey date: dd-mm-yyyy



Book ticket

Flight Name: op2

Flight No: 012

Base price: 100000

Email

Mobile

No of passengers: 1

Journey date: 20-07-2024

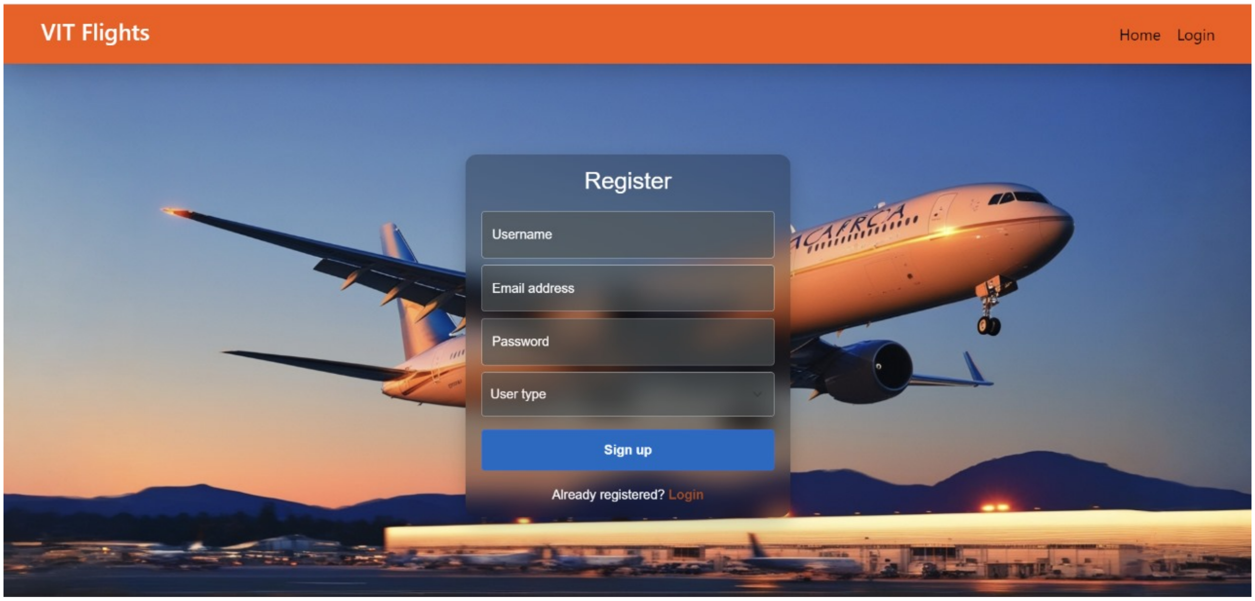
Seat Class: Select

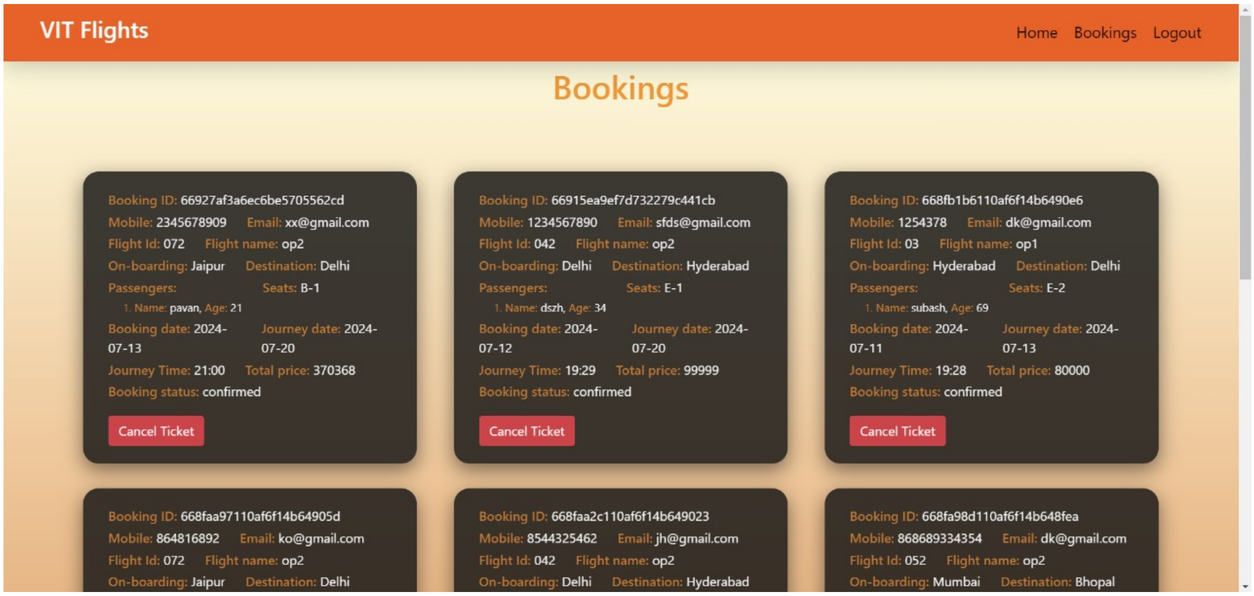
Passenger 1

Name

Age

Total price: 0





Bookings

Booking ID: 66927af3a6ec6be5705562cd
Mobile: 2345678909 Email: xx@gmail.com
Flight Id: 072 Flight name: op2
On-boarding: Jaipur Destination: Delhi
Passengers: Seats: B-1
1. Name: pavan, Age: 21
Booking date: 2024-07-13 Journey date: 2024-07-20
Journey Time: 21:00 Total price: 370368
Booking status: confirmed

Cancel Ticket

Booking ID: 66915ea9ef7d732279c441cb
Mobile: 1234567890 Email: sfds@gmail.com
Flight Id: 042 Flight name: op2
On-boarding: Delhi Destination: Hyderabad
Passengers: Seats: E-1
1. Name: dszh, Age: 34
Booking date: 2024-07-12 Journey date: 2024-07-20
Journey Time: 19:29 Total price: 99999
Booking status: confirmed

Cancel Ticket

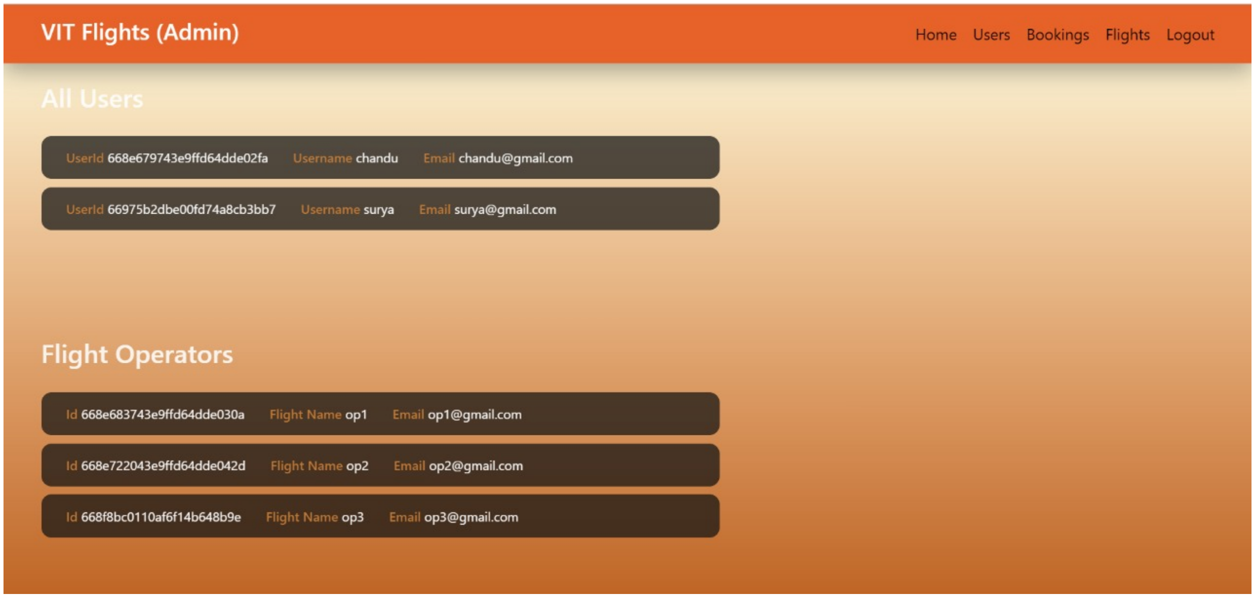
Booking ID: 668fb1b6110af6f14b6490e6
Mobile: 1254378 Email: dk@gmail.com
Flight Id: 03 Flight name: op1
On-boarding: Hyderabad Destination: Delhi
Passengers: Seats: E-2
1. Name: subash, Age: 69
Booking date: 2024-07-11 Journey date: 2024-07-13
Journey Time: 19:28 Total price: 80000
Booking status: confirmed

Cancel Ticket

Booking ID: 668faa97110af6f14b64905d
Mobile: 864816892 Email: ko@gmail.com
Flight Id: 072 Flight name: op2
On-boarding: Jaipur Destination: Delhi

Booking ID: 668faa2c110af6f14b649023
Mobile: 8544325462 Email: jh@gmail.com
Flight Id: 042 Flight name: op2
On-boarding: Delhi Destination: Hyderabad

Booking ID: 668fa98d110af6f14b648fea
Mobile: 868689334354 Email: dk@gmail.com
Flight Id: 052 Flight name: op2
On-boarding: Mumbai Destination: Bhopal



All Users

UserId 668e679743e9ffd64dde02fa Username chandu Email chandu@gmail.com

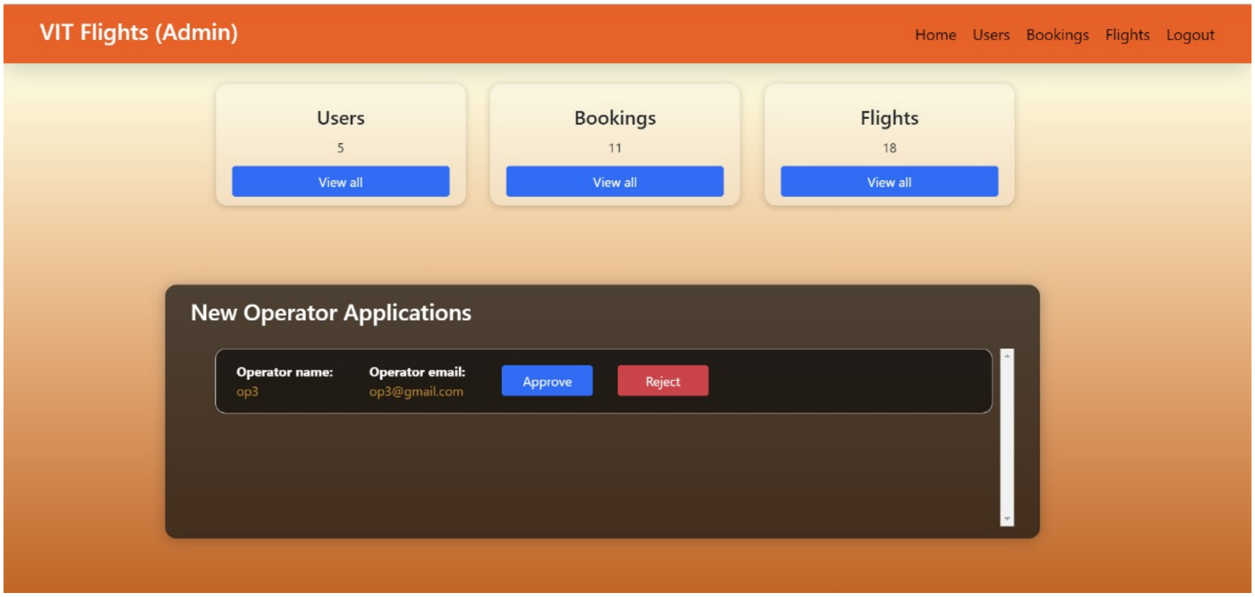
UserId 66975b2dbe00fd74a8cb3bb7 Username surya Email surya@gmail.com

Flight Operators

Id 668e683743e9ffd64dde030a Flight Name op1 Email op1@gmail.com

Id 668e722043e9ffd64dde042d Flight Name op2 Email op2@gmail.com

Id 668f8bc0110af6f14b648b9e Flight Name op3 Email op3@gmail.com



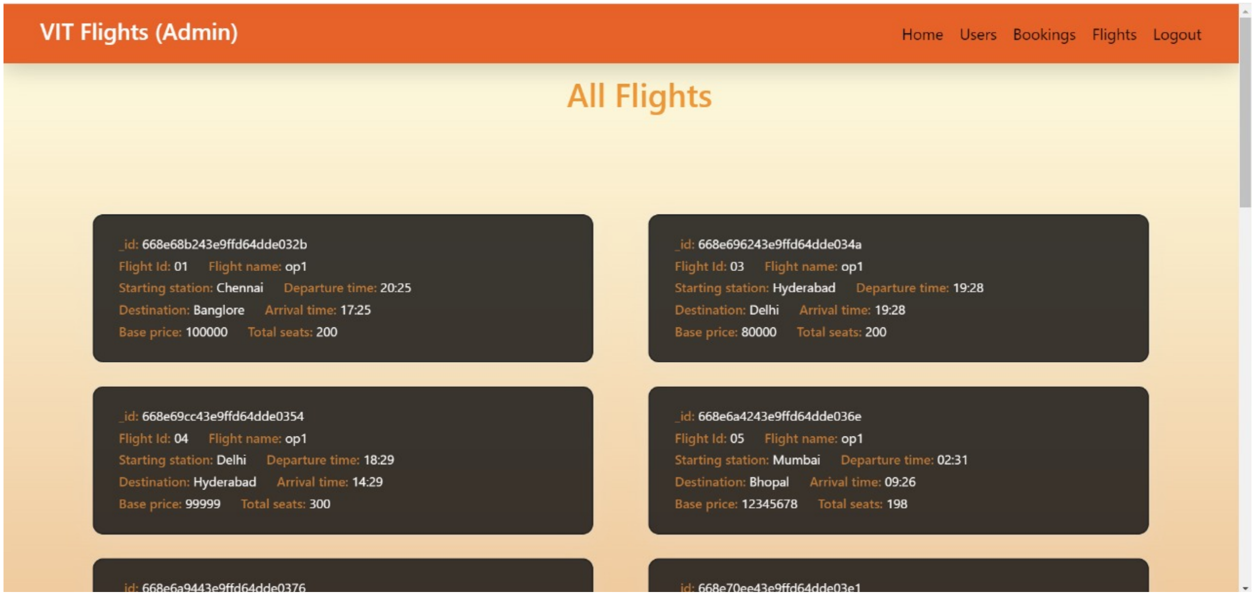
Users
5

Bookings
11

Flights
18

New Operator Applications

Operator name: op3 **Operator email:** op3@gmail.com



All Flights

_id: 668e68b243e9ffd64dde032b
Flight Id: 01 Flight name: op1
Starting station: Chennai Departure time: 20:25
Destination: Banglore Arrival time: 17:25
Base price: 100000 Total seats: 200

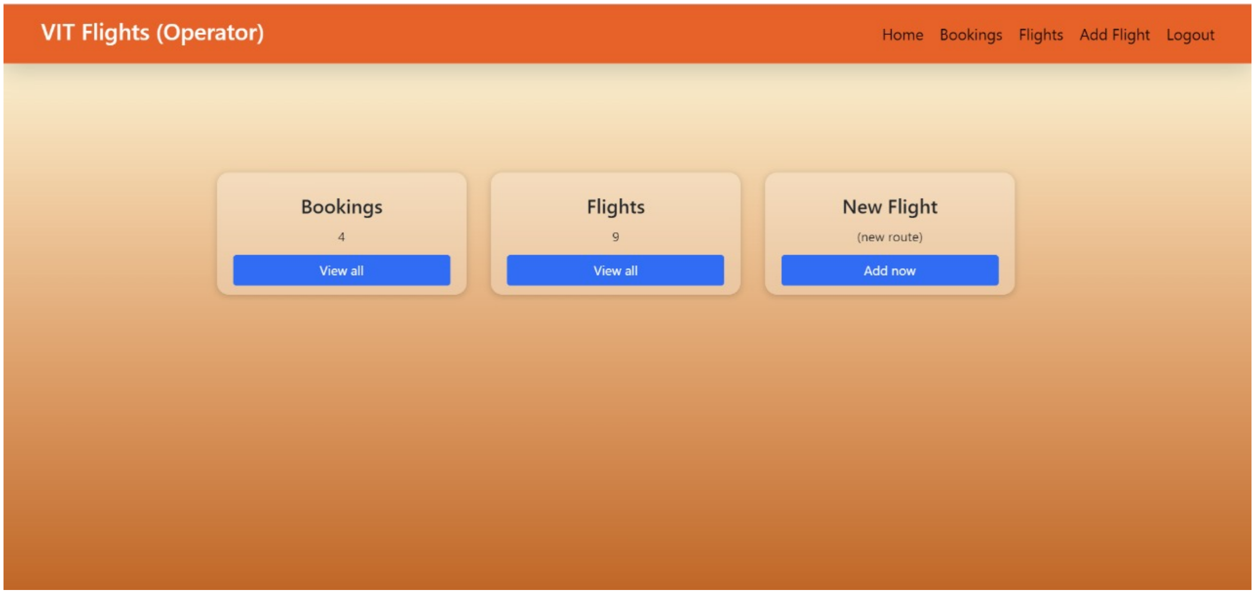
_id: 668e696243e9ffd64dde034a
Flight Id: 03 Flight name: op1
Starting station: Hyderabad Departure time: 19:28
Destination: Delhi Arrival time: 19:28
Base price: 80000 Total seats: 200

_id: 668e69cc43e9ffd64dde0354
Flight Id: 04 Flight name: op1
Starting station: Delhi Departure time: 18:29
Destination: Hyderabad Arrival time: 14:29
Base price: 99999 Total seats: 300

_id: 668e6a4243e9ffd64dde036e
Flight Id: 05 Flight name: op1
Starting station: Mumbai Departure time: 02:31
Destination: Bhopal Arrival time: 09:26
Base price: 12345678 Total seats: 198

_id: 668e6a9443e9ffd64dde0376

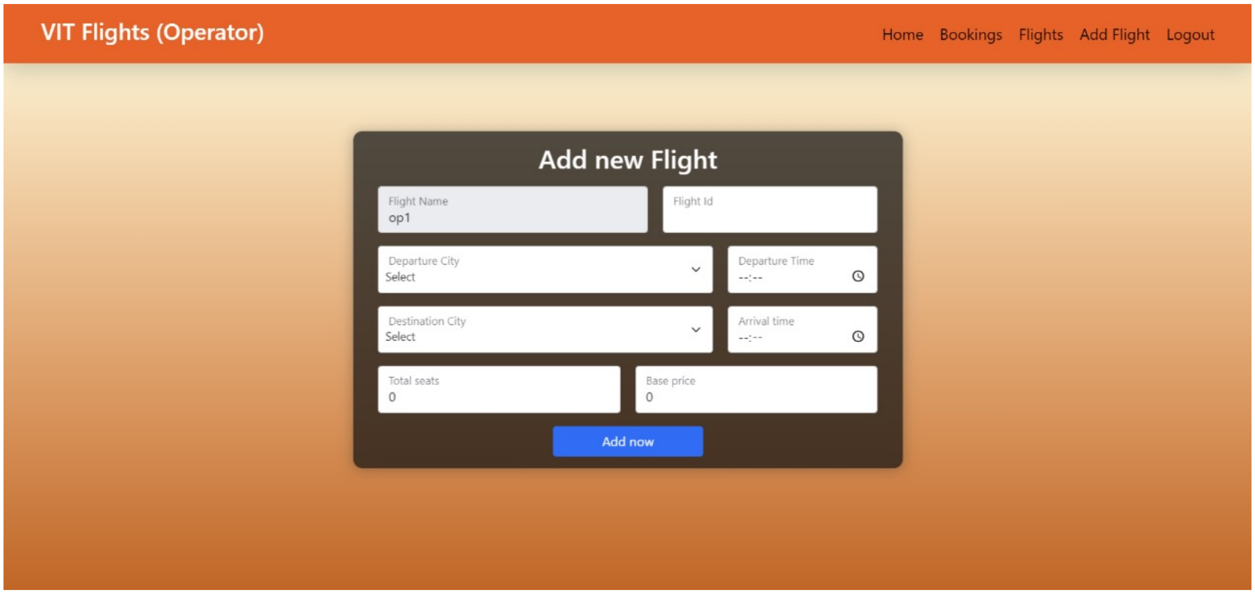
_id: 668e70ee43e9ffd64dde03e1



Bookings
4

Flights
9

New Flight
(new route)



VIT Flights (Operator)

Home Bookings Flights Add Flight Logout

Add new Flight

Flight Name: op1

Flight Id:

Departure City: Select

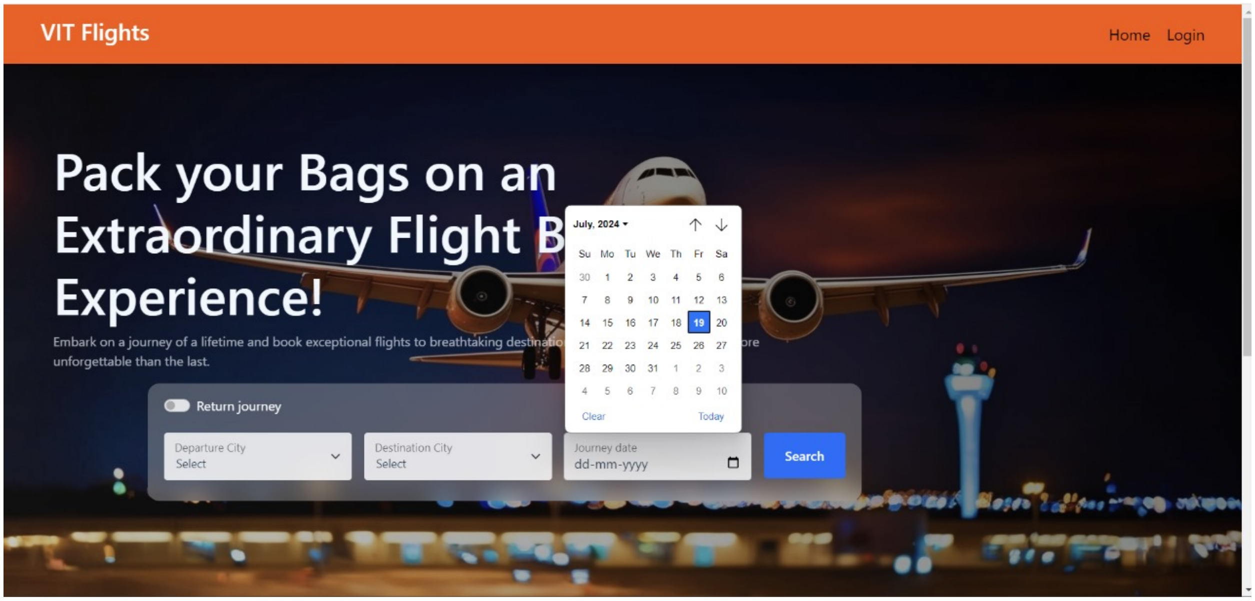
Departure Time: --:--

Destination City: Select

Arrival time: --:--

Total seats: 0

Base price: 0



VIT Flights

Home Login

Pack your Bags on an Extraordinary Flight B Experience!

Embark on a journey of a lifetime and book exceptional flights to breathtaking destinatio ore unforgettable than the last.

July, 2024

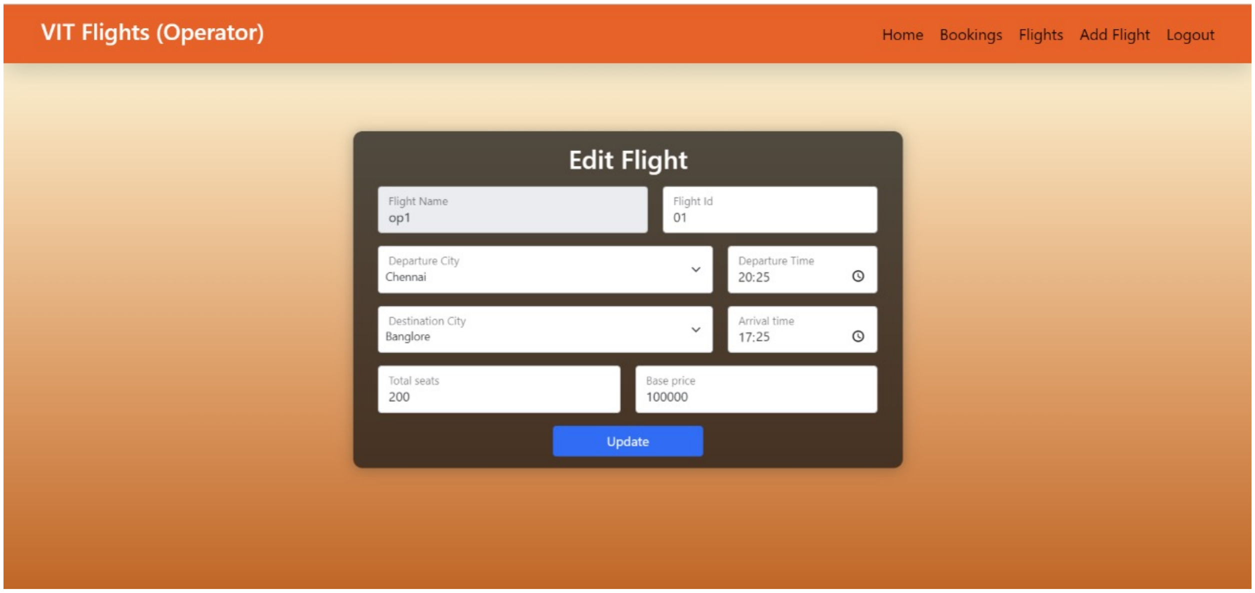
Su	Mo	Tu	We	Th	Fr	Sa
30	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31	1	2	3
4	5	6	7	8	9	10

Return journey

Departure City: Select

Destination City: Select

Journey date: dd-mm-yyyy



Edit Flight

Flight Name: op1
Flight Id: 01
Departure City: Chennai
Departure Time: 20:25
Destination City: Banglore
Arrival time: 17:25
Total seats: 200
Base price: 100000

10. Testing

Functional Testing

1. **Unit Testing**
   - **Definition**: Tests individual components or functions of the flight booking system.
   - **JIRA Management**: Track unit test coverage and results as part of development tasks. Developers can log issues for failed tests and link them to specific code commits.
2. **Integration Testing**
   - **Definition**: Tests the interaction between different modules or systems.
   - **JIRA Management**: Create test cases for integration points (e.g., booking service interacting with payment gateway) and link them to the respective user stories or tasks.
3. **System Testing**
   - **Definition**: Tests the entire system as a whole to ensure it meets requirements.
   - **JIRA Management**: Organize system test cases into a comprehensive test plan, executed before release. Track execution results and log defects.
4. **User Acceptance Testing (UAT)**
   - **Definition**: Ensures the system meets business requirements and is usable by end-users.

- **JIRA Management**: Create UAT test cases based on user stories and business requirements. Involve stakeholders in executing these tests and gather feedback.

Non-Functional Testing

1. **Performance Testing**
   - **Definition**: Assesses system performance under load, stress, and scalability conditions.
   - **JIRA Management**: Log performance test scenarios, results, and any identified bottlenecks. Link these to performance improvement tasks.
2. **Security Testing**
   - **Definition**: Identifies vulnerabilities and ensures the system is secure.
   - **JIRA Management**: Document security test cases (e.g., SQL injection, XSS). Log any security issues found and track their resolution.
3. **Usability Testing**
   - **Definition**: Evaluates how user-friendly and intuitive the system is.
   - **JIRA Management**: Create tasks for usability test sessions. Document findings and recommendations for UI/UX improvements.
4. **Compatibility Testing**
   - **Definition**: Ensures the system works across different devices, browsers, and operating systems.
   - **JIRA Management**: Create test cases for different combinations of devices and browsers. Log compatibility issues and track their resolution.

API Testing

- **Definition**: Tests the interactions between your system and external APIs (e.g., for flight data, payment processing).
- **JIRA Management**: Document API test cases and results. Log issues related to API integrations and track their resolution.

End-to-End Testing

- **Definition**: Simulates a full user journey from searching for flights to booking and receiving confirmation.
- **JIRA Management**: Create end-to-end test scenarios. Ensure all steps of the journey are tested, and log any issues found during the process.

Managing Tests in JIRA

1. **Using a Test Management Tool**

    - Integrate JIRA with a test management tool like Zephyr, Xray, or TestRail.
    - Create and organize test cases within the tool, link them to JIRA issues, and track execution results.
2. **Creating and Tracking Issues**
    - Log individual test cases as sub-tasks of user stories or tasks.
    - Track test execution results and link defects to the relevant test cases and user stories.
3. **Reporting**
    - Use JIRA dashboards and reports to track test progress, execution results, and defect status.
    - Generate detailed reports for different testing phases and share them with stakeholders.

11. Screenshots or Demo

https://drive.google.com/file/d/19xVM5pkZ-qp3vH6eHdJ4vwVBtlqeGk2e/view?usp=sharing

12. Known Issues

Session Timeouts: Users may experience unexpected session timeouts during booking process.

Payment Gateway Errors: Occasional errors when processing payments through third-party gateways.

Search Delays: Long search times during peak hours due to high server load.

Display Issues: Inconsistent display of flight details on certain mobile browsers.

Email Notifications: Delayed or missing email confirmations for bookings.

Seat Selection: Limited options for seat selection on some partner airlines.

Currency Conversion: Inaccurate currency conversion rates displayed during booking.

13. Future Enhancements

Mobile App Development: Create a dedicated mobile app for seamless booking and management.

Enhanced Search Algorithms: Implement advanced search algorithms to improve speed and accuracy.

User Profiles: Allow users to create profiles for faster bookings and personalized offers.

Integration with Loyalty Programs: Partner with airlines to integrate loyalty programs for frequent flyers.

Real-time Updates: Provide real-time updates on flight status and delays.

Multi-language Support: Expand language support to cater to international users.

Improved Accessibility: Ensure the platform is accessible to users with disabilities.

Virtual Reality (VR) Seat Selection: Implement VR technology for interactive seat selection experiences.