

N O	REQUEST TYPE	API ENDPOINT	REQUEST HEADERS	REQUEST BODY	RESPONSE CODE	RESPONSE BODY	Comments
1	POST	/api/v1.0/flight/airline/inventory/add	Content-Type: application/json	<pre>json { "airlineName": "Indigo", "flightNumber": "AE101", "fromPlace": "DELHI", "toPlace": "MUMBAI", "departureTime": "2025-11-25T09:00:00", "arrivalTime": "2025-11-25T11:00:00", "totalSeats": 180, "price": 4500, "specialFare": 3500, "fareCategory": "STUDENT" }</pre>	201	<pre>json { "id": 1, "airlineName": "Indigo", "flightNumber": "AE101", "fromPlace": "DELHI", "toPlace": "MUMBAI", "departureTime": "2025-11-25T09:00:00", "arrivalTime": "2025-11-25T11:00:00", "totalSeats": 180, "availableSeats": 180, "price": 4500, "specialFare": 3500, "fareCategory": "STUDENT" }</pre>	<p>Adds a new flight schedule (inventory) into the system with airline, flight number, route, timings, seats and fares.</p> <p>Generates a database id automatically using @GeneratedValue when inventory is saved.</p> <p>Initializes availableSeats to totalSeats when a new inventory record is created.</p> <p>Persists full flight metadata: airline name, flight number, fromPlace, toPlace, departureTime, arrivalTime.</p> <p>Stores pricing info: price (base fare) and specialFare (category-specific reduced fare).</p> <p>Stores fareCategory on inventory (STUDENT/SENIOR/REGULAR/ARMY/CORPORATE) so booking logic can apply discounts.</p> <p>Admin-only endpoint (POST /api/v1.0/flight/airline/inventory/add) accepts DTO and returns the saved entity.</p> <p>Uses DTO → Entity mapping in service layer to prevent exposing JPA entities directly to the API.</p> <p>Validates input and rejects invalid or malicious inventory data before persisting.</p>
					400	{ "airlineName": "Airline name is required" }	
					500	{ "message": "Something went wrong" }	

					201	1) IF return flight is not there	Supports searching only the onward journey based on fromPlace, toPlace, and travelDate.
				<pre>1) json [{ "fromPlace": "DELHI", "toPlace": "MUMBAI", "travelDate": "2025-11-25", "travelTime": "", "tripType": "ROUND_TRIP", "returnDate": "2025-11-28" }]</pre>		<pre>json [{ "airlineName": "Indigo", "flightNumber": "AE101", "departureTime": "2025-11-25T09:00:00", "arrivalTime": "2025-11-25T11:00:00", "oneWayPrice": 4500, "roundTripPrice": 9000, "returnFlightNumber": null, "returnDepartureTime": null, "returnArrivalTime": null, "message": "Only onward flight available. Return flight not found." }]</pre>	<p>Performs separate searches for onward and return flights and combines them in a single response.</p> <p>If no return flight is found, onward flight is still returned with a message indicating return unavailability.</p> <p>If travelTime is empty, system performs a full-day search (00:00–23:59).</p> <p>If travelTime is provided, a ±30-minute window is used to find nearby flights.</p> <p>fromPlace and toPlace are matched in lowercase to avoid case sensitivity issues.</p> <p>Leading or trailing spaces in city names are trimmed</p>

2 POST	/api/v1.0/flight/search	<pre>2) json{ "fromPlace": "DELHI", "toPlace": "MUMBAI", "travelDate": "2025-11-25", "travelTime": "", "tripType": "ROUND_TRIP", "returnDate": "2025-11-28" }</pre> <pre>3) json{ "fromPlace": "DELHI", "toPlace": "MUMBAI", "travelDate": "2025-11-25", "travelTime": "", "tripType": "ONE WAY" }</pre>	<p>2) IF return flight is there</p> <pre>[{ "airlineName": "Indigo", "flightNumber": "AE101", "departureTime": "2025-11-25T09:00:00", "arrivalTime": "2025-11-25T11:00:00", "oneWayPrice": 4500.0, "roundTripPrice": 9000.0, "message": null, "returnDepartureTime": "2025-11-28T09:00:00", "returnArrivalTime": "2025-11-28T11:00:00", "returnFlightNumber": "AE102" }]</pre> <p>3) ONE Way flight</p> <pre>[{ "airlineName": "Indigo", "flightNumber": "AE101", "departureTime": "2025-11-25T09:00:00", "arrivalTime": "2025-11-25T11:00:00", "oneWayPrice": 4500.0, "roundTripPrice": 9000.0, "message": "One-way trip available", "returnDepartureTime": null, "returnArrivalTime": null, "returnFlightNumber": null }]</pre>	<p>before searching</p> <p>Converts travelDate into a time range to fetch only flights on the specified day.</p> <p>Returns roundTripPrice as oneWayPrice × 2 (placeholder for future pricing logic).</p> <p>Adds messages like "One-way trip available" or "Only onward flight available; return flight not found".</p>

400	Validation failed
404	No flights found
500	Internal Server Error

3	POST	/api/v1.0/flight/booking/{flightId}	content-type : application/json	<pre>{ "email": "amankumar@gmail", "numberOfSeats": 2, "passengers": [{ "name": "aman", "gender": "MALE", "age": 31, "meal": "VEG", "fareCategory": "STUDENT", "seatNumber": 14 }, { "name": "amen", "gender": "MALE", "age": 31, "meal": "VEG", "fareCategory": "ARMY", "seatNumber": 15 }] }</pre>	<p>201</p> <pre>{ "pnr": "5RWB4Z", "email": "amankumar@gmail", "numberOfSeats": 2, "totalPrice": 8000.0, "bookingTime": "2025-11-17T11:52:31.010319200Z", "message": "Booking successful", "passengers": [{ "name": "aman", "seatNumber": 14, "gender": "MALE", "age": 31, "meal": "VEG", "fareCategory": "STUDENT", "fareApplied": 3500.0, "fareMessage": "Special fare applied for category: STUDENT" }, { "name": "amen", "seatNumber": 15, "gender": "MALE", "age": 31, "meal": "VEG", "fareCategory": "ARMY", "fareApplied": 4500.0, "fareMessage": "No special fare available. Extra benefits applied." }] }</pre> <p>Allows a logged-in user to book seats on a selected flight using its flightId.</p> <p>Generates a unique 6-character alphanumeric PNR for every successful booking.</p> <p>Automatically links each passenger to the booking and the flight.</p> <p>Deducts seat count from flight inventory based on number of seats booked.</p> <p>Stores booking confirmation details including total price, passenger details, and booking timestamp.</p> <p>Ensures atomic booking using @Transactional to guarantee safe seat updates.</p> <p>Applies special fare automatically when passenger's fare category matches the flight's configured special category.</p> <p>Falls back to standard base fare when special fare is not applicable.</p> <p>Generates personalized messages such as "Special fare applied" or "No special fare available, extra benefits awarded."</p> <p>Calculates per-passenger fare individually and stores the applied fare in the database.</p> <p>Computes final total price using the actual per-passenger applied fares (special + normal mix).</p> <p>Validates that number of seats requested matches the number of passengers provided.</p> <p>Ensures at least 1 seat and at most available seats are requested.</p> <p>Checks that no seat number is duplicated in the request (each passenger must be unique).</p> <p>Prevents booking of already-taken seats using a database query for booked seats.</p> <p>Ensures the flight exists for the provided flightId, otherwise throws "Flight not found."</p> <p>Validates all passenger details (name, age, gender, meal preference, seat number).</p> <p>Requires each passenger to provide a valid fare category (STUDENT/SENIOR/ARMY/etc.).</p>	
					<p>500</p> <pre>{ "status": 500, "error": "Internal Server Error", "message": "Something went wrong while processing booking", "path": "/api/v1.0/flight/booking/1" }</pre>	
					<p>400</p> <pre>{ "status": 400, "error": "BAD_REQUEST", "message": "Duplicate seat numbers in request", "path": "/api/v1.0/flight/booking/1" }</pre>	

				200	<pre>{ "pnr": "5RWB4Z", "email": "amankumar@gmail.com", "numberOfSeats": 2, "totalPrice": 8000.0, "bookingTime": "2025-11-17T11:52:31.010319Z", "message": "Booking retrieved", "passengers": [{ "name": "aman", "seatNumber": 14, "gender": "MALE", "age": 31, "meal": "VEG", "fareCategory": "STUDENT", "fareApplied": 3500.0, "fareMessage": "Special fare applied for category: STUDENT" }, { "name": "amen", "seatNumber": 15, "gender": "MALE", "age": 31, "meal": "VEG", "fareCategory": "ARMY", "fareApplied": 4500.0, "fareMessage": "No special fare available. Extra benefits applied" }] }</pre> <p>Fetches complete ticket details using a unique PNR number.</p> <p>Returns booking information including flight details, passenger list, and booking metadata.</p> <p>Provides passenger-wise fare details (special or regular).</p> <p>Displays current booking status (BOOKED / CANCELLED).</p> <p>Validates that PNR is a non-empty 6-character alphanumeric string.</p> <p>Ensures booking exists for the provided PNR; otherwise returns 404 "Ticket not found".</p> <p>Ensures cancelled tickets still return history but clearly show "status": "CANCELLED".</p> <p>Blocks invalid PNR formats (symbols, incorrect length, blank values).</p> <p>Returns 400 for invalid/malformed PNR.</p> <p>Returns 404 if booking does not exist.</p> <p>Returns structured JSON error response through GlobalErrorHandler</p>
4	GET	/api/v1.0/flight/ticket/{pnr}	content-type : application/json	404	<pre>{ "status": 404, "error": "NOT_FOUND", "message": "Ticket not found for PNR: XYZ123", "path": "/api/v1.0/flight/ticket/XYZ123" }</pre>

				200	<pre>[{ "pn": "5RWB4Z", "email": "amankumar@gmail", "numberOfSeats": 2, "totalPrice": 8000.0, "bookingTime": "2025-11-17T11:52:31.010319Z", "message": "Booking history item", "passengers": [{ "name": "aman", "seatNumber": 14, "gender": "MALE", "age": 31, "meal": "VEG", "fareCategory": "STUDENT", "fareApplied": 3500.0, "fareMessage": "Special fare applied for category: STUDENT" }, { "name": "amen", "seatNumber": 15, "gender": "MALE", "age": 31, "meal": "VEG", "fareCategory": "ARMY", "fareApplied": 4500.0, "fareMessage": "No special fare available. Extra benefits applied." }] }]</pre>	<p>Fetches all bookings ever made using a specific email ID.</p> <p>Returns past, current, and cancelled bookings for complete user history.</p> <p>Shows booking date/time, PNR, total price, and status for each booking.</p> <p>Supports multiple bookings on different flights and different dates.</p> <p>Ensures email format is valid according to RFC-compliant regex.</p> <p>Rejects blank or malformed email values.</p> <p>Returns an empty list if user has no bookings instead of throwing an error.</p> <p>Prevents exposing invalid/static resources by using proper path variable escaping.</p> <p>400 returned for invalid email format.</p> <p>404 only when email is syntactically correct but not found (optional behavior).</p> <p>Clean JSON error via GlobalErrorHandler for validation issues.</p>
5	GET	/api/v1.0/flight/booking/history/{emailId}		400	<pre>{ "status": 400, "error": "BAD_REQUEST", "message": "Invalid email format", "path": "/api/v1.0/flight/booking/history/kunal@gmail" }</pre>	
				404	<pre>{ "status": 404, "error": "NOT_FOUND", "message": "No bookings found for email: user@example.com", "path": "/api/v1.0/flight/booking/history/user@example.com" }</pre>	

				200	<p>Ticket with PNR K3Q22C has been cancelled successfully.</p> <p>Updates booking status to CANCELLED without deleting booking history.</p> <p>Restores previously booked seats back into the flight's available seat count.</p> <p>Ensures passengers remain stored as historical records.</p> <p>Validates PNR format (6 uppercase letters/numbers).</p> <p>Ensures booking exists and is currently in "BOOKED" state (cannot cancel twice).</p> <p>Restricts cancellation if journey date is within 24 hours of departure.</p> <p>Prevents cancellation of already cancelled or expired bookings.</p> <p>400 returned for invalid PNR format.</p> <p>404 returned when booking does not exist.</p> <p>409 Conflict returned when trying to cancel an already cancelled ticket.</p> <p>400 returned if cancellation attempted within 24 hours of flight departure.</p> <p>Clean JSON error response via GlobalErrorHandler.</p>
6	DELETE	/api/v1.0/flight/booking/cancel/{pnr}	content-type : application/json Authorization : Bearer eyJhbGciOiJIUzI	400	{ "status": 400, "error": "BAD_REQUEST", "message": "Ticket cannot be cancelled within 24 hours of flight departure.", "path": "/api/v1.0/flight/booking/cancel/A34KDX" }
				404	{ "timestamp": "2025-11-17T14:28:11.869283500Z", "status": 404, "error": "Not Found", "message": "PNR 'K3Q22C' not found", "path": "/api/v1.0/flight/booking/cancel/K3Q22C" }
				409	{ "status": 409, "error": "CONFLICT", "message": "Ticket with PNR A34KDX is already cancelled", "path": "/api/v1.0/flight/booking/cancel/A34KDX" }

