DAYWISE TASK AND CHALLENGES

Day 1 – Backend Setup & Authentication

Tasks Completed

- **Project Initialization**:
    - Created Spring Boot backend project with dependencies (Spring Web, Spring Security, JPA, MySQL, JWT).
    - Set up database schema with entities: User, Role.
- **Authentication Module**:
    - Implemented **user registration** (/api/v1/auth/register) and **login** (/api/v1/auth/login).
    - Integrated **JWT Authentication** for secure login.
    - Configured **BCrypt password encryption**.
- **Role-based Access Control**:
    - Defined roles: ADMIN and CUSTOMER.
    - Configured Spring Security to allow different endpoints per role.

Challenges Faced

1. **JWT Token Handling**:
    - Initially faced issues with token expiration & parsing.
    - Solved by adding JwtFilter and testing with Swagger headers.
2. **Role Mapping in SQL**:
    - Had to ensure roles (ADMIN, CUSTOMER) were inserted in DB before registration.
    - Challenge: default role assignment for new users.

Day 2 – Bus, Route & Trip Management

Tasks Completed

- **Bus & Route Management (Admin)**:
    - Endpoints created:
        - /api/v1/buses → Add new bus.
        - /api/v1/routes → Add new route.
- **Trip Scheduling & Seat Inventory**:

    - /api/v1/trips → Create trip with busId, routeId, departureTime, arrivalTime, and fare.
    - /api/v1/trips/search → Search available trips.
    - /api/v1/trips/{id}/seats → Retrieve seat layout & availability.
- **Database Relationships Implemented**:
    - Bus → Trip (1:M).
    - Route → Trip (1:M).
    - Trip → Seat (1:M).

Challenges Faced

1. **Seat Inventory Generation**:
    - Automatically mapping seat layouts from Bus entity to each Trip.
    - Solution: Implemented logic to generate seats dynamically per trip.
2. **Date-based Trip Search**:
    - Handling search by origin, destination, and date.
    - Solved by writing JPA query with multiple parameters.
3. **Swagger Testing with JWT**:
    - Admin APIs required JWT token; had to test role-based restrictions.

Day 3 – Booking, Payments & Ticketing

Tasks Completed

- **Booking Module (Customer)**:
    - /api/v1/bookings/hold → Hold selected seats temporarily.
    - Prevent double booking using seat-lock mechanism.
- **Payment Processing**:
    - /api/v1/payments/checkout → Mock payment gateway integration.
    - Status tracking: PENDING → SUCCESS/FAILED.
- **Ticketing & Cancellations**:
    - /api/v1/tickets/{id} → Fetch ticket with QR code + PDF download link.
    - /api/v1/bookings/{id}/cancel → Cancel booking & process refund (policy-based).
- **Reports (Admin)**:
    - /api/v1/reports/sales → Generate sales summary (total revenue, top routes).

Challenges Faced

1. **Concurrency in Seat Booking**:

    - Faced race condition when multiple customers booked the same seat.
    - Solved by **synchronizing booking transaction** + adding seat status checks.
2. **Refund & Cancellation Policy**:
    - Needed business rules (full/partial refund depending on cancellation time).
    - Implemented configurable refund percentages.
3. **PDF Ticket Generation with QR Code**:
    - Challenge: Integrating ticket details into PDF format.
    - Solved using libraries like **iTextPDF / JasperReports**.

Final Outcomes

- ✅ Secure authentication & role-based access.
- ✅ Real-time trip scheduling and seat inventory.
- ✅ End-to-end booking flow with payments, ticketing, and cancellation.
- ✅ Admin dashboards for reports & revenue tracking.