Backend Development Report – Week 2

Project: Household Services Booking Platform

Developer: Ahmad Raza **Role:** Backend Developer

Internship: Xynerotech Solutions

Week: 2 (Monday to Friday)

Q Project Objective

To build the backend system for a service booking platform where users can:

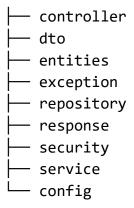
- Register and log in securely
- View available household services (e.g., AC repair, electrician)
- (Upcoming) Book services (planned for Week 3)

✓ Tasks Completed This Week

1. Backend Setup

- Spring Boot project initialized using Java 17
- Configured folder structure:

Folder Structure



Health check endpoint added:

GET /api/health → "Server is running"

2. Database Integration

- MySQL database connected successfully.
- Created the following tables/entities:
 - User (name, email, password, role)
 - Service (name, description, price)
 - Booking (userId, serviceId, date, status structure only for now)

3. n Authentication APIs

- User Registration
 - Endpoint: POST /api/register
 - o Features:
 - Stores hashed password
 - Validates email & password
 - Returns JWT token and user details
- Login API
 - Endpoint: POST /api/login
 - o Returns token with role-based access (USER/ADMIN)

4. Role-Based Service APIs

- Public Endpoints
 - GET /api/service/get List all services
 - GET /api/service/get/{id} View service by ID
- Admin-Only Endpoints
 - o POST /api/services Add new service
 - PUT /api/services/{id} Update service
 - DELETE /api/services/{id} Delete service
- ✓ Dummy services like "AC Repair", "Water Purifier Repair", etc. added to DB.

5. User Management (Admin Only)

- POST /api/user/add Add new user
- GET /api/user/get List all users
- GET /api/user/get/{id} Get user by ID
- PUT /api/user/update/{id} Update user
- DELETE /api/user/delete/{id} Delete user

6. API Testing & Documentation

- Tested all APIs via Postman
- JWT-based authentication tested for protected routes
- Request & Response formats included in README.md
- Example:

```
// Login Response
{
   "message": "Login successful",
   "data": {
      "token": "JWT_TOKEN",
      "email": "ahmad123@gmail.com",
      "role": "USER"
},
```

```
"success": true
```

}

 Detailed API docs cover all endpoints, access roles, validation rules, and error responses.

W Validation & Error Handling

- Registration validation:
 - o Email format
 - o Password length (min. 6 characters)
- Service validation:
 - o Name: 3-50 chars
 - o Description: 10–255 chars
 - o Price: Must be positive
- Global exception handling implemented

Tools Used

- Backend: Java 17, Spring Boot 3, Spring Security, JWT
- Database: MySQLTesting: Postman
- Version Control: GitHub
- Lombok used for boilerplate code reduction

Deliverables Completed

- Spring Boot backend project set up
- ✓ User & Service models implemented
- Auth & Service APIs working

- Dummy service data added
- API documentation written
- Code pushed to GitHub
- ✓ Postman-tested endpoints shared

№ Next Steps (Week 3 Preview)

- Implement Booking APIs (create, list, cancel)
- Add Booking history for users
- Secure booking endpoints with JWT
- Integrate with frontend UI pages

Developer Info

Ahmad Raza

Backend Intern @ Xynerotech

mustafaraza03898@gmail.com

GitHub Profile