Train Tracker Backend – Phase 2 Documentation

1. Phase 2 Overview

Phase 2 focuses on converting the backend from a railway-centric system to a user-centric system. The objective is to introduce personalization, secure user-specific data handling, and event-driven notifications.

2. Key Features Implemented

- User Favorites (Add / List / Remove)
- Secure user-scoped APIs
- Notification system (mock engine)
- Notification read/unread management
- JWT-based access control integration

3. Architecture Extension

Phase 2 builds on Phase 1 architecture by adding:
User → Favorites → Background Engine → Notifications.
This mirrors real-world event-driven backend systems.

4. Database Models (Phase 2)

Favorite Model:
- user_id, train_id (Unique per user & train)

Notification Model:
- user_id, title, message, is_read, created_at

5. Database Relationships

- Users ↔ Trains: Many-to-Many via Favorites
- Users → Notifications: One-to-Many
- Notifications are immutable events, marked read/unread by users

6. Favorites APIs

POST /me/favorites/{train_no}
- Adds a train to the user's favorites

GET /me/favorites

- Lists all favorite trains of the user

DELETE /me/favorites/{train_no}
- Removes a train from the user's favorites

7. Notifications APIs

POST /me/notifications/mock
- Triggers mock background notification generation

GET /me/notifications
- Fetches all notifications for the logged-in user

POST /me/notifications/{id}/read
- Marks a notification as read

GET /me/notifications/unread-count
- Returns count of unread notifications

8. Notification Engine Design

A mock background engine simulates real-time monitoring of favorite trains. In production, this engine would be replaced by scheduled jobs, message queues, or real railway data streams.

9. Security & Access Control

All Phase 2 APIs are protected using JWT-based authentication. Each request is scoped to the authenticated user, ensuring strict data isolation.

10. Phase 2 Summary

Phase 2 delivers a fully personalized backend experience with favorites, notifications, and secure user-specific workflows. This phase establishes the foundation for real-time features and scalability.

Phase 2 – High Level Flow (Textual Diagram)

User Login → JWT Token Issued
User Adds Favorite Train
Background Engine Monitors Favorites
Notification Generated
User Fetches Notifications
User Marks Notification as Read