Train Tracker Backend – Phase 1 Documentation

1. Project Overview

This project is a backend system inspired by the 'Where Is My Train' concept. Phase 1 focuses on building a stable, scalable, and production-style backend that exposes public APIs for train tracking using FastAPI and PostgreSQL.

2. Technology Stack

- Python 3.12
- FastAPI (API framework)
- PostgreSQL (Database)
- SQLAlchemy ORM
- JWT Authentication
- Passlib + bcrypt (Password security)

3. Architecture Overview

The system follows a clean layered architecture:
Router Layer → Service Layer → Model Layer → Database.
Each layer has a single responsibility, making the system maintainable and scalable.

4. Core Models

Station:
- id, code, name

Train:
- id, train_no, name

TrainRoute:
- id, train_id, station_id, stop_order

User:
- id, email, hashed_password

5. Database Relationships

- A Train has multiple Stations through TrainRoute
- A Station can belong to multiple Trains
- A User is independently managed for authentication

6. Implemented APIs

Public APIs:
- GET /api/stations
- GET /api/trains
- GET /api/trains/{train_no}/route
- GET /api/trains/{train_no}/status

Admin API:
- POST /admin/seed-all

Auth APIs:
- POST /auth/register
- POST /auth/login
- GET /auth/me (Protected)

7. Data Seeding Strategy

Master data (stations, trains, routes) is loaded using JSON-based seed services. Seeding is idempotent and can be triggered via an admin endpoint.

8. Authentication Flow

Users register with email and password. Passwords are securely hashed. On login, a JWT access token is generated. Protected endpoints require the token via Authorization: Bearer header.

9. Live Status Engine (Mock)

A simulated real-time engine provides current train status by selecting a station from the train route and generating a delay dynamically.

10. Phase 1 Summary

Phase 1 delivers a fully functional backend MVP with clean architecture, secure authentication, real-world API design, and extensibility for future phases.