Train Tracker Backend – Phase 1 to Phase 3 API by API Hinglish Explanation

Introduction – Ye document kis liye hai?

Ye document specially isliye banaya gaya hai taaki har API ko ye samjha ja sake ki:
• Ye API kyun bani?
• Iska kaam kya hai?
• Ye backend ke flow me kahan fit hoti hai?

Is document ko padhne ke baad tum Swagger dekho to turant samajh aa jana chahiye ki kaunsi API kya role play kar rahi hai.

PHASE 1 – Core Backend APIs (Foundation)

GET /stations

Ye API railway stations ka master data deti hai.

Is API ka kaam:
• App ko station list dena (source/destination selection)
• Dropdowns, search, filters ke liye base data

Backend me kya hota hai:
• Station table se data fetch hota hai
• Sirf read operation, koi logic nahi

Ye API simple hai kyunki ye foundation layer ka hissa hai.

GET /trains

Ye API trains ka master data return karti hai.

Iska role:
• App ko batana kaun-kaun si trains system me exist karti hain
• Train number ke base par aage ke APIs ka entry point

Ye API 'reference data' provide karti hai.

GET /trains/{train_no}/route

Ye API kisi ek train ka complete route batati hai.

Is API ka flow:
• Train number input me aata hai
• TrainRoute table se stations ka ordered list milta hai

Real-life mapping:
• 'Is train kaunsa-kaunsa station cross karegi?'

Ye API Phase 1 ka sabse important endpoint hai.

PHASE 2 – Authentication & User APIs

POST /auth/register

Ye API naya user create karti hai.

Flow:
• Email + password aata hai
• Password hash hota hai
• User DB me save hota hai

Security reason:
• Plain password kabhi store nahi hota

POST /auth/login

Ye API user ko authenticate karti hai.

Flow:
• Email/password verify hota hai
• Access token + refresh token milta hai

Is API ka role:
• Session start karna
• User ko system me entry dena

POST /auth/refresh

Ye API bina dobara login ke naya access token deti hai.

Flow:
• Refresh token validate hota hai
• Naya access token issue hota hai

Ye API long-lived sessions ke liye hoti hai.

GET /auth/me

Ye API current logged-in user ka data return karti hai.

Use case:
• Profile page
• Token valid hai ya nahi check karna

POST /me/favorites/{train_no}

Ye API user ke favorite trains me ek train add karti hai.

Flow:
• Token se user identify hota hai
• Train ID map hoti hai
• Favorite table me entry hoti hai

Is API ke bina personalization possible nahi.

GET /me/favorites

Ye API user ke saare favorite trains return karti hai.

Real-life use:
• Dashboard pe tracked trains dikhana

GET /me/notifications

Ye API user ki notifications return karti hai.

Iska role:
• System ne user ke liye kya updates generate kiye
• Delay alerts, status changes

PHASE 3 – Real-Time & Automation APIs

WS /ws/train/{train_no}

Ye WebSocket endpoint real-time train status push karta hai.

Difference from HTTP:
• Client request nahi karta
• Server khud data bhejta hai

Real-life example:
• Live train tracking
• Chat applications

Background Scheduler (No direct API)

Scheduler ek API nahi hai, balki background process hai.

Iska kaam:
• Train status check karna
• Redis se last state compare karna
• Change aane par notification generate karna

Redis Cache (Internal component)

Redis direct API nahi expose karta.

Iska role:
• DB hit kam karna
• Last known train status store karna
• Scheduler ko efficient banana

Final Samajh – System ka flow

User login karta hai → token milta hai → user favorite train add karta hai.
Scheduler background me train check karta hai.
Agar delay change hota hai → Redis update hota hai → DB me notification aati hai.
WebSocket user ko real-time update push karta hai.

Yahi poora backend system hai.