Real-Time Location Tracker for Multivendor Delivery Platform-Assignment

Objective: - Implement a **real-time location tracking system** like **Rapido or Dunzo** for a **multivendor marketplace**. Vendors assign delivery partners to orders, and users can track the rider's live location.

Time Limit: **48 hours**

Requirements

Frontend (Next.js + TypeScript)

- A **vendor dashboard** to:
  - See list of their orders
  - Assign a **delivery partner** to an order
- A **delivery partner dashboard**:
  - Simulate live location updates (use geolocation API or simulate coordinates with a timer)
  - "Start Delivery" button begins tracking the location in real time
- A **customer tracking page**:
  - Shows a **map (Leaflet.js or Google Maps)** with the **real-time location of the delivery partner**
  - Auto-updates every 2–3 seconds

Backend (Node.js + TypeScript)

- APIs for:
  - Vendor login/signup
  - Delivery partner login/signup
  - Assigning delivery partner to an order
  - Updating and retrieving current location
  - Order tracking API for customers
- Use **WebSockets or Server-Sent Events** (Socket.IO preferred) for pushing real-time updates
- **Authentication**: JWT or session-based login for all user types (vendor, delivery, customer)
- **Multitenancy logic**: Each vendor only sees their orders

Tech Stack

- Frontend: Next.js, TypeScript
- Backend: Node.js, Express, TypeScript
- Database: MongoDB or PostgreSQL

- WebSocket: Socket.IO or alternative
- Map: Google Maps API or Leaflet.js (OpenStreetMap)
- Auth: JWT or cookie-based
- Deployment not mandatory, but if done, use Render, Vercel, or Railway

Expected Features

Role	Feature
Vendor	View/assign orders, see delivery status
Delivery	See assigned order, start tracking, send location
Customer	Track assigned delivery partner in real-time on map

Submission Guidelines

- GitHub repo with clear README:
  - Architecture
  - Setup instructions
  - Features
- Bonus points for:
  - Deployment link
  - Clean TypeScript typing
  - Git commits showing development process

Evaluation Criteria

Criteria	Weight
Code quality (TS usage)	25%
Functional features	25%
Real-time implementation	20%
Project structure	15%
UI/UX	10%
README/docs	5%