

Transport-You API

Backend Development Status Report

A breakdown of what has been developed and what remains, for the multi-tenant bus booking & scheduling platform.

Prepared: 26 October 2026

Stack: Laravel (PHP) • Passport (OAuth2) • PostgreSQL • REST API v1

Status: Core booking & scheduling complete • 78 automated tests passing

1. Executive Summary

Transport-You is a multi-tenant API that lets transport operators publish trips and customers search and book seats. The backend is built in Laravel with a clean controller → service → model layering, OAuth2 authentication (Laravel Passport), and PostgreSQL.

The **core journey is complete end-to-end**: operators define vehicle types and seat arrangements, publish one-off or recurring schedules, and customers search available trips and book specific seats. Recurring schedules are stored as a single pattern and expand into bookable trips on demand; booking a specific day materialises a concrete “scheduled trip” that operators manage through a dedicated module.

16	12	23	78
API controllers	Modules live	Data models	Tests passing

2. Architecture & Foundations

Area	Implementation
Framework	Laravel REST API, versioned under /api/v1
Auth	Laravel Passport (OAuth2) with email + OTP verification
Tenancy	Multi-tenant: super-admin manages all; tenant users scoped to their own data
Structure	Thin controllers, business logic in Services, Eloquent models
Database	PostgreSQL, 52 migrations
Responses	Standard success/error envelope; form-request validation

3. Completed — What Has Been Developed

The following modules are implemented, wired to routes, and covered (directly or indirectly) by the passing test suite.

Module	What it does	Status
Authentication	Register, login, logout, current-user; email OTP verify & resend.	DONE
Tenant onboarding	Self-service operator onboarding endpoint.	DONE
Tenant profile	View & update the authenticated tenant (logo, details).	DONE
Locations	CRUD for terminals/locations with state + city.	DONE
States	Reference CRUD for states.	DONE
Routes	CRUD + smart route resolver (by id, location ids, or city names).	DONE
Vehicle categories	Global reference CRUD (super-admin).	DONE
Vehicle classes	Reference CRUD.	DONE
Seat arrangement types	Global catalogue: name, number_of_seats, image, description; filter by vehicle type; seat-layout diagram images generated.	DONE
Vehicle types	CRUD; many-to-many link to possible seat arrangements with one default.	DONE
Vehicles	CRUD for physical vehicles (registration, type, status).	DONE
Schedules – one-off	Create a concrete dated trip (outbound + optional round trip).	DONE
Schedules – recurring	Create a single recurring pattern (interval + weekdays); no per-day rows are generated.	DONE
Derived capacity	Capacity & seat map come from the selected seat arrangement, not stored.	DONE
Seat selection toggle	Per-schedule flag: seats are pickable only when enabled.	DONE
Trip search (public)	Search merges concrete trips and virtual occurrences expanded from recurring patterns by date-in-interval; trip detail endpoint.	DONE
Seat status endpoint	Available/unavailable seat map for a concrete schedule or a recurring occurrence (group + travel_date).	DONE
Booking module	Per-passenger details (name, phone, email, seat); seat selection enforced only when enabled; capacity & clash validation; round trips; booking status & cancel.	DONE
Recurring materialisation	Booking a recurring day find-or-creates the concrete scheduled trip.	DONE
Scheduled trips module	Manage concrete trips: list/filter, show with passenger manifest, update status, cancel (cascades to bookings).	DONE
Image upload	Upload endpoint for vehicle / arrangement images.	DONE
Automated tests	Feature test suite across auth, routes, schedules, trips, vehicles, bookings.	DONE

4. API Endpoint Inventory

Group	Endpoints
Auth / account	register, login, logout, verify-otp, resend-otp, user, tenant-onboarding
Tenant	GET /tenant, PATCH /tenant
Reference	locations, states, vehicle-categories, vehicle-classes (full CRUD)
Fleet	vehicle-types (CRUD + arrangements), vehicles (CRUD), seat-arrangement-types (CRUD + ?vehicle_type_id)
Schedules	schedules (CRUD; store handles one-off & recurring pattern)
Scheduled trips	GET scheduled-trips, GET {id}, PATCH {id}/status, PATCH {id}/cancel
Trips (public)	GET trips, GET trips/{id}, GET trips/{id}/seats, GET trips/recurring/{group}/seats
Bookings	GET bookings, POST bookings, GET {id}, PATCH {id}/status, PATCH {id}/cancel
Media	POST upload/images

All endpoints are under **/api/v1**. Write operations on reference/fleet catalogues are super-admin only; tenant data is scoped per operator; trip search is public.

5. Pending — What Is Left To Be Developed

Several data models exist as placeholders (Payment, Ticket, Manifest, Promotion, LoyaltyTransaction, Role/Permission) but have no endpoints or business logic yet. The items below are grouped by priority.

Item	Description	Priority
Payments	Payment gateway integration; pay-for-booking flow; Payment model is unused.	HIGH
Booking confirmation	Auto-confirm a booking once payment succeeds; expiry of unpaid holds.	HIGH
Seat hold / locking	Short-lived seat reservation + DB locking to prevent double-booking under load.	HIGH
Tickets	Issue/generate tickets (QR/reference) per passenger after payment.	HIGH
Notifications	Email/SMS for OTP delivery, booking confirmation, trip reminders, cancellations.	HIGH
Driver & vehicle assignment	Assign a physical vehicle + driver to a scheduled trip.	MED
Driver management	Driver records, availability, and assignment module.	MED
Manifest module	Standalone manifest generation/export (partial view exists on scheduled-trip show).	MED
Recurring pattern management	List/edit/delete recurring patterns (schedule_groups) endpoints.	MED
Recurring round-trip pairing	Pair outbound + return recurring occurrences in search results.	MED
RBAC permissions	Granular roles/permissions enforcement (Role/Permission models exist).	MED
Promotions / discounts	Promo codes, fare discounts (Promotion model is unused).	LOW
Loyalty program	Points accrual/redemption (LoyaltyTransaction model is unused).	LOW
Refunds & policies	Refund handling and cancellation/refund policy engine.	LOW
Reporting & analytics	Operator dashboards: sales, occupancy, revenue.	LOW
API documentation	Published OpenAPI/Swagger or Postman collection.	LOW
Fix pre-existing tests	Auth OTP response shape & Route state_id not-null (3 failing tests, unrelated to new work).	MED

6. Testing & Known Notes

- **78 automated feature tests pass.** 3 remaining failures are pre-existing and unrelated to the schedule/booking work: the auth OTP registration response shape, and two route tests hitting a `locations.state_id` not-null constraint from separate in-progress work.
- **Recurring model:** a recurring schedule is one pattern record. Concrete trips exist only for one-off schedules and for recurring days that have been booked (materialised on demand).
- **Capacity** is always derived from the trip's seat arrangement; neither vehicles nor vehicle types store a capacity number.
- **Seat selection** is optional per schedule; when off, bookings consume seats by headcount only.
- **Concurrency:** seat double-booking is currently guarded by validation inside a DB transaction but not by row locking — see “Seat hold / locking” in pending work.

End of report • Transport-You API • generated for internal planning.