# Tembea Africa

## How the System Works

A tour-first marketplace for multi-day African travel — travelers discover curated, guide-led departures, pay by card or African mobile money (with deposits & instalments), and operators are paid via an escrow-style hold. This document explains the architecture, data model, and every core journey end to end.

Technical & product overview · Laravel API + Nuxt frontend
Generated for the Tembea team

# 1. What Tembea is

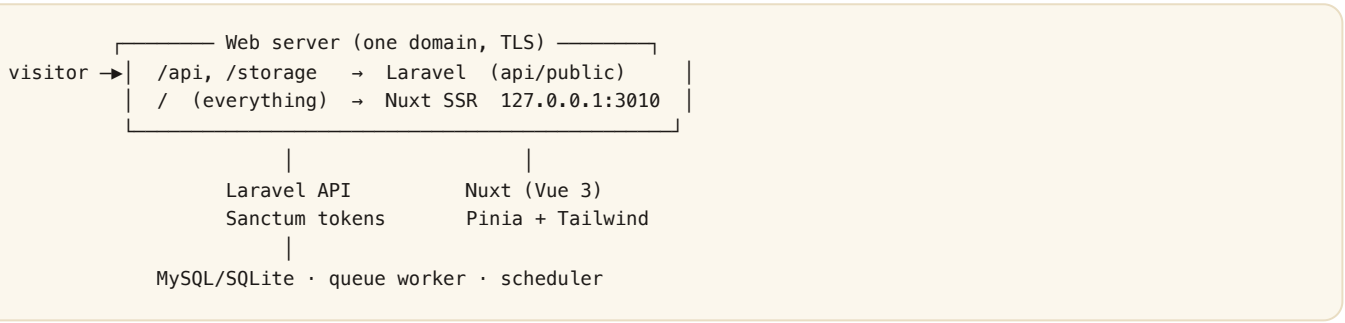
Tembea Africa is an online **marketplace for multi-day African tours**. Operators publish tours with dated **departures**; travelers book seats, pay online, and Tembea takes a commission while holding the operator's payout in escrow until shortly before travel. Beyond the table-stakes booking flow, Tembea adds three differentiators aimed at the African travel market:

- **Africa-native payments** — card *and* mobile money (M-Pesa via Flutterwave), plus operator payouts across African corridors.
- **An AI concierge** — a multi-turn chat planner that only recommends real catalogue tours.
- **Guide-as-creator** — travelers follow individual tour leaders and get notified of their new trips.

The word **tembea** is Swahili for *"to walk / wander / travel."*

## 2. Architecture

Two processes behind **one web-server vhost** (single origin, so no CORS): Laravel serves `/api` and `/storage`; the Nuxt SSR Node service serves everything else.




### Backend

- Laravel 12, PHP 8.3
- Sanctum bearer-token auth
- Eloquent + service classes
- Queue (database) for emails
- Scheduler for recurring jobs
- Laravel Pint (style, CI-enforced)

### Frontend

- Nuxt 3/4 + Vue 3 + TypeScript
- Pinia (auth store), composables
- Tailwind v4 (brand theme)
- SSR via `useFetch`; authed calls via `useApi()`
- Leaflet + OSM for itinerary maps
- i18n: English, Deutsch, Français, Kiswahili

80 API routes · 30 Eloquent models · 17 services · 7 transactional emails.

## 3. Domain model

The core entities and how they relate:

| Entity | Purpose & key relationships |
|---|---|
| **User** | Account (traveler / operator_admin / admin). Has bookings, wallet, loyalty account, follows leaders, referral code. |
| **Operator** | A tour company. Owns tours; has a commission rate and a payout account; receives payouts. |
| **Tour** | A product (title, itinerary, base price, currency, deposit %, cancellation policy). Belongs to an operator; has departures, itinerary days, activities, reviews. |
| **Departure** | A dated instance of a tour with capacity & seats remaining, price override, private flag. |
| **Booking** | A traveler's reserved seats on a departure. Holds money breakdown, currency, FX snapshot, amount paid / balance due, cancellation snapshot, addresses. Bound by `reference`. |
| **Passenger** | Each traveler on a booking (name, email, DOB, gender, nationality, passport, dietary/medical, emergency contact). Identity fields encrypted at rest. |
| **Payment / Payout / Refund** | Money movement: inbound charges, outbound operator payouts (escrow-held), and refunds on cancellation. |

| | |
|---|---|
| **TourLeader** | A named guide travelers can follow; linked to tours/departures. |
| **Review** | Verified post-trip rating + text, tied to a completed booking. |
| **Wallet / LoyaltyAccount** | Travel-credit ledger and points balance/tier; append-only transactions. |
| **Currency / Country** | Reference data: FX rates (live-refreshed) and the ISO country list (dial codes + flags). |
| **MessageThread / Message** | In-app traveler ↔ operator messaging. |
| **Activity** | Optional paid add-ons (heli, scuba, wine) attached to a booking. |
| **AiConversation / AiMessage** | Multi-turn AI concierge history. |
| **Post** | Blog content for discovery/SEO. |

## 4. Users & auth

- **Roles:** `traveler` `operator_admin` `admin` — gate access to the operator console and admin tools.
- **Auth:** Sanctum **bearer tokens** (not cookies). The token is stored client-side (Pinia auth store, persisted) and sent as `Authorization: Bearer …` by `useApi()`. A 401 auto-logs-out.
- **Profile:** travelers manage name, phone (with dial code), country, home currency, residential address, avatar (uploaded & resized), and password — all from an account area with a sidebar.

## 5. Discovery & pricing

### Discovery

Public, SSR-rendered: home/explore grid, `/top-tours`, tour detail (gallery + a real Leaflet map plotting every itinerary stop), guides, blog. Prices display in the viewer's currency (resolved from `?currency` → cookie → geo header → Accept-Language). Wishlisted tours show a filled heart.

### Quote & the money split

`PricingService::quote` computes, for a departure + guest count:

| Field | How it's derived |
|---|---|
| Subtotal | price per guest × guests (+ selected activity add-ons) |
| Commission | operator's rate (default 15%) × subtotal — Tembea's take |
| Operator payable | subtotal − commission |
| Total | subtotal + fees (fees absorbed by platform in v1) |
| FX snapshot | settlement currency, display currency, and the rate — **frozen at purchase** |
| Payment policy | deposit eligibility, minimum deposit, balance due date (see §7) |

## 6. Booking creation

`BookingService::create` runs in a DB transaction:

```
1. Lock the departure row (FOR UPDATE) → prevents overselling seats
2. Validate status = scheduled & enough seats remain
3. Quote the price; add any activity add-ons
4. Resolve the first payment from the chosen plan (full/deposit/custom)
5. Create the Booking (pending) + its Passengers + activity pivots
6. Decrement seats_remaining; mark departure FULL if it hits 0
7. Return the booking → controller initiates the first payment
```

> **Validation highlights:** the lead traveler is the booking holder and must be **18+**; every passenger needs name, **email**, gender, nationality, and passport details. Logged-in travelers get the lead auto-filled from their profile and most recent trip.

## 7. Flexible payments (deposits, instalments, custom amounts)

Operators can accept less than 100% upfront. The platform layers rules on top:

| Rule | Behaviour |
|---|---|
| Minimum deposit | The first payment must be ≥ **20%** of the total. |
| Eligibility | Deposits/instalments are only offered when the departure is **> 30 days** away; inside 30 days, full payment is required. |
| Balance deadline | The balance must be cleared at least **30 days before departure** (or earlier if the operator sets a longer lead). |
| Custom amount | Traveler can pay any amount ≥ the 20% minimum; the rest becomes the balance. |
| Instalments | Partial top-ups allowed up to the balance-due date; after that, the full balance is required. |

On the trips page travelers see the outstanding balance and can pay it in full or as a custom instalment. Each payment updates `amount_paid` and recomputes `balance_due`.

> **Currency lock:** the customer-facing currency and FX rate are captured at purchase, so a booking is shown — and its balance paid — in one consistent currency from deposit to final payment, regardless of the site's currency switcher. Settlement (the actual charge) is always the tour's currency.

## 8. Payment gateways & routing

A single `PaymentGateway` contract with a `GatewayManager` that routes per booking:

| Driver | Used for |
|---|---|
| **Stripe** | Default pay-in for cards worldwide; payouts for ZA/US/EU/UK/AU/CA operators. |
| **Flutterwave** | Forced for **M-Pesa** (KES) pay-in; operator payouts across African corridors (KE, NG, TZ, RW, BW, EG, MA…). |
| **Paystack** | Alternative African gateway (Nigeria-centric). |
| **Mock** | Automatic fallback until real keys are set — keeps the dev loop working. |

```
Card:    create charge → hosted checkout → redirect /book/return → webhook confirms
M-Pesa:  STK push to phone → status pending → signed webhook settles the payment
```

Webhooks (`POST /api/webhooks/{gateway}`) are **signature-verified**. In production a client cannot self-confirm a live charge — confirmation only comes from the signed webhook. A payment row is locked on confirm so a webhook + client-confirm can't double-credit.

## 9. Escrow-style payout

When a booking is **paid in full**, Tembea schedules the operator payout for **~2 days before departure** (configurable) — funds are held until then to protect the traveler. Deposit-only bookings confirm the seat but **defer the payout** until the balance is paid. A scheduled job disburses due payouts via the routed gateway; the operator gets a `PayoutPaid` email.

## 10. Cancellations & refunds

- Each booking snapshots the tour's **cancellation policy** at purchase (full-refund window, partial %, no-refund cutoff).
- On cancel, `CancellationPolicy::refundFractionForDaysOut` computes the refundable fraction based on days to departure; the refund is calculated on **amount actually paid** (not the full total), and seats are released.
- `tembea:process-refunds` (hourly) auto-processes pending refunds via the gateway; the traveler gets a `RefundProcessed` email. Refund status shows on the trips page.

## 11. Currency: live FX + per-booking lock

- **Live rates:** `tembea:refresh-rates` (hourly) pulls USD-based rates from a configurable provider (default `open.er-api.com`, keyless) into the `currencies` table; a missing rate keeps the last-known value.
- **Display:** browsing prices convert to the viewer's currency via the USD base.
- **Lock:** once booked, the display currency + rate are frozen (§7), so an existing booking never drifts and can't be paid in a different currency than the deposit.

## 12. The other journeys

### Reviews

Only travelers with a completed booking can review (verified). Ratings roll up onto the tour.

### Messaging

In-app threads between traveler and operator, with unread badges across the header, sidebar, and messages page.

### Loyalty & wallet

Points earned per USD of confirmed booking (normalised across currencies), redeemable into **wallet credit**; referrals credit both sides. The wallet is an append-only ledger; applying credit converts across currencies correctly.

### Wishlist

Shared state; hearts reflect saved tours everywhere, syncing on login/logout.

### Follow-your-guide

Travelers follow leaders and see a feed of their upcoming departures; followers are emailed (`FollowedLeaderNewTrip`) when a new trip is published.

### AI concierge

A multi-turn chat (`/ai/chat`) backed by the Anthropic API with a `search_tours` tool, so it only ever recommends **real** catalogue tours. Degrades to a deterministic planner with no API key.

### Affiliates

Post-booking "complete your trip" attachments (flights, insurance, transfers…) earning revenue share.

## 13. Scheduled jobs & emails

| Command (schedule) | What it does |
|---|---|
| `tembea:refresh-rates` (hourly) | Refresh live FX rates. |
| `tembea:process-refunds` (hourly) | Auto-process pending refunds via the gateway. |
| `tembea:collect-balances` (daily) | Email balance reminders — **before** the due date (14/7/1 days) and an **overdue** notice after (1/7 days), deduped per day. |
| `tembea:purge-sensitive` (daily) | Null out passport/medical/accessibility data 90 days after a trip ends (data minimisation). |

**Transactional emails** (queued): booking confirmation, operator new-booking, payout paid, refund processed, balance due / overdue, followed-leader new trip.

## 14. Security & data protection

- **Encryption at rest** for identity data (passport number, medical & accessibility notes).
- **Data minimisation** — sensitive passenger data auto-purged after travel.
- **Signed webhooks**; no client self-confirm of live charges; row-locking against double-payment.
- **Escrow hold** protects travelers until near departure.
- Bearer-token auth; role-based access to operator/admin surfaces.

## 15. Deployment, testing & ops

- **Single domain** via Apache (or nginx): `/api` + `/storage` → Laravel, everything else proxied to the Nuxt SSR node (port 3010); systemd units for web + queue worker.
- **CI** (GitHub Actions): PHP 8.3 + Pint + backend tests; Node build + vitest; a deploy job that ships the build and reloads services.
- **Tests:** 64 backend feature/unit tests (money splits, deposits/instalments, gateway routing & webhook signatures, refunds, rewards, currency, reminders).
- **Go-live runbook:** `docs/OPS-CHECKLIST.md` (verify env, seed reference data, queue worker, mail transport, live gateway keys + webhooks, scheduler).

> **Required reference data** on deploy: seed currencies *and* countries — the currency switcher and country/phone dropdowns are empty without them.

---

Tembea Africa — system overview. The code is production-shaped; most remaining work to go live is configuration (payment keys, mail transport, deploy) rather than features.