Payment Gateway Communication Overview

1. Overview
- Front end never handles raw card data.
- All payment initiation requests go through the backend (`/api/payments/...`) with JWT in `Authorizatio
- The backend orchestrates Paystack, Flutterwave, Stripe, and bank transfer flows via dedicated servic

2. Property Purchase (Escrow) Flow
1. User clicks “Proceed to Purchase (Escrow)” on the property detail page.
2. `EscrowPaymentFlow` loads the property, calculates the 0.5% escrow fee, and presents a multi-ste
3. The frontend posts to `POST /api/payments/initialize` with:
   - `amount`: property price
   - `paymentMethod`: `flutterwave | paystack | stripe | bank_transfer`
   - `paymentType`: `property_purchase`
   - `relatedEntity`: { type: "property", id: propertyId }
   - `description`, `currency`, other metadata
4. Backend validates payload, creates a `Payment` record, and routes it to the appropriate provider se
5. Provider service sends the request to Paystack/Flutterwave/Stripe and returns authorization data.
6. Frontend redirects the user to the provider’s checkout (or uses Stripe client).
7. Webhooks (`/api/payments/webhook/:provider`) confirm success and update payment status.
8. Escrow transaction is created on backend (`/api/escrow`), fees/timeline are recorded, and the prope

3. Vendor Listing Fee (Registration)
- Admin sets `vendorListingFee` via `/api/admin/settings`.
- During vendor registration, we render `VendorRegistrationPayment`, which calls `/api/payments/initia
- After payment success, we call `registerAsVendor` to assign the vendor role and save the payment r

4. Backend Payment Orchestration
- `backend/routes/payments.js` enforces authentication (JWT) and validation (`express-validator`).
- Provider metadata includes amount, fees, customer details, and `relatedEntity`.
- Provider services compute platform (2.5%) + processing (1.5%) fees before invoking Paystack/Flutte
- Webhooks verify signatures and change payment statuses to `completed`, `failed`, etc.

5. Escrow Transaction Creation
- `EscrowPaymentFlow` uses `createEscrowTransaction` (Escrow context).
- `POST /api/escrow` checks property availability, prevents duplicate escrows, calculates fees, and cre
- Escrow records expose timeline entries and can be retrieved via `/api/escrow/:id`.

6. Security & Compliance Notes
- JWT required on all backend payment routes (`Authorization: Bearer <token>`).
- Validation middleware catches missing/invalid data before provider calls.

- Webhooks verify provider signatures.
- Sensitive keys stored in `.env` (Paystack, Flutterwave, Stripe).
- Escrow/payment data kept in MongoDB; no provider secrets reach the browser.

7. Bank Integration Checklist
- Provide test credentials for Paystack/Flutterwave (or bank API).
- Configure webhook endpoint(s) at `/api/payments/webhook/:provider`.
- Validate flows using existing Cypress E2E coverage or mock data.
- Monitor transactions through admin dashboards and backend logs.
- Escrow flow demonstrates multi-step proof for compliance (review → hold funds → release).