EasySale – Stripe Connect (OAuth) Setup Guide

Developer configuration guide for the Full build (multi■tenant / white■label).

Version: 1.1 **Date:** 2026-01-30

This guide assumes EasySale already includes the full Stripe Connect OAuth flow in the product (**Connect** → **OAuth redirect** → **callback** → **status/summary**), but the developer still must configure their own Stripe platform credentials and URLs. No credentials are hardcoded, and nothing is stored in the frontend.

What you get when finished

- Tenants can click **Connect Stripe** and authorize their own Stripe account (OAuth).
- EasySale stores the tenant's connected account ID and tokens (encrypted) and shows **real summary data** (business name, country, currency).
- You can create Stripe Checkout Sessions (Phase 2) once webhooks + checkout endpoints are enabled.

What you must set up

- A Stripe **platform** account for EasySale (your account).
- Stripe Connect enabled + OAuth settings (client ID and redirect URI).
- Webhook endpoint + webhook signing secret (needed for Phase 2).
- Environment variables on the backend (never commit secrets).

1) One■time Stripe platform setup (you / developer)

These steps are done **once** for your EasySale deployment. After this, each tenant can onboard their own Stripe via OAuth.

A. Create / configure your Stripe account

- Create a Stripe account for EasySale (the platform).
- Complete the Stripe onboarding questions (business info).
- Switch to **Test mode** while wiring the integration; you can repeat the same steps for live later.

B. Enable Stripe Connect

- In Stripe Dashboard, enable **Connect** (this turns your Stripe account into a platform).
- Pick the Connect model that fits: **Standard** (simplest) is usually enough for white■label onboarding. If you need deeper control later, you can evolve to Express/Custom.

C. Configure OAuth redirect URL

- Set the OAuth redirect/callback URL to your backend endpoint:
- **https://YOUR_DOMAIN/api/integrations/stripe/callback**
- For local dev, use a tunnel URL (ngrok/cloudflared) instead of localhost. Keep prod and dev redirect URIs separate.

D. Create (or locate) your Connect client ID

- In the Connect settings, locate your **Client ID** (looks like **ca_…**).
- You will set this as STRIPE_CLIENT_ID on the backend.

E. Create webhook endpoint (Phase 2)

If you are doing Phase 2 (Checkout Sessions) you must add a webhook endpoint. If you are only doing Phase 1 (connect/status/summary), you can skip this for now.

- Create webhook endpoint URL: **https://YOUR_DOMAIN/api/payments/webhooks/stripe**
- Select events: **checkout.session.completed**, **checkout.session.expired**, (optional) **payment_intent.*** if you expand later.
- Copy the webhook signing secret **whsec_…** and set STRIPE_WEBHOOK_SECRET.

Required environment variables

Set these on the backend runtime environment (Docker env, systemd, .env for dev).

Env var	Purpose	Where you get it
STRIPE_CLIENT_ID	Stripe Connect OAuth client ID (platform)	Stripe Dashboard → Connect settings (Client ID, ca_…)

STRIPE_SECRET_KEY	Platform secret key (used for Stripe API + token exchange)	Stripe Dashboard → Developers → API keys (sk_…)
STRIPE_REDIRECT_URI	Your backend callback URL	Your app URL, e.g. https://YOUR_DOMAIN/api/integrations/stripe/callback
STRIPE_WEBHOOK_SECRET	Webhook signature secret (Phase 2)	Stripe Dashboard → Developers → Webhooks (whsec_…)

Notes

- **Never** store secret keys in the frontend, git, or logs. Mask secrets in UI and logs.
- Use separate keys for Test and Live. Your app should surface which mode is active.
- If your backend rejects localhost redirect URIs in production mode, that is expected and desired.

2) Tenant onboarding flow (what your users do)

This is the end■user experience inside EasySale once you've configured the platform.

- Tenant opens **Settings** → **Integrations** → **Stripe**.
- Clicks **Connect Stripe**.
- Stripe OAuth opens; tenant signs in (or creates a Stripe account) and approves access.
- User returns to EasySale; Stripe card shows **Connected** and displays summary (business name, country, currency, masked acct id).
- Tenant can click **Test** to verify access.
- Tenant can click **Disconnect** to revoke / remove the mapping.

Verification checklist (dev)

- OAuth callback stores connected account ID per tenant (acct_…).
- Status endpoint returns connected/disconnected without leaking tokens.
- Summary endpoint returns real Stripe account fields (business name, country, currency).
- Logs endpoint records connect/test/disconnect events (no secrets).

Troubleshooting

- **Invalid redirect_uri**: confirm STRIPE_REDIRECT_URI exactly matches the Stripe Connect setting.
- **State invalid**: ensure oauth_states are stored per tenant and expire; don't reuse state values.
- **403 / permissions**: confirm Connect is enabled and the tenant authorized successfully.
- **Webhook signature failed** (Phase 2): confirm STRIPE_WEBHOOK_SECRET matches the endpoint's signing secret.

If something fails, open the Integration Logs drawer and confirm the backend wrote an error event with a useful message.