Quick Start — your storefront in about 2 hours

This guide is for someone who just downloaded the steep kit and wants to get a working storefront online today. You don't need to be a senior developer — if you can copy and paste, follow numbered steps, and click buttons in a website, you can finish this.

If a word looks like jargon, it probably is. The first time we use it, we'll explain it. There's also a longer plain-English glossary on the live storefront at `/glossary` once you're up.

What you'll have when you're done

- A live online store at your own domain (or a `*.vercel.app` URL if you don't have one)
- The ability to log into an admin panel and create products, run sales, refund orders
- Real credit-card payments through **Stripe** (the company that handles cards for most modern stores)
- Automated email confirmations, GitHub-invite delivery, and order tracking

Time estimate: 30-45 minutes to running locally, another 30-60 minutes to live on Vercel. Coffee break in the middle is encouraged.

What you need before you start (10 minutes — gather accounts)

All of these are free to start. Sign up for whichever you don't already have. Don't worry about which "plan" — the free tier of each is plenty for testing.

Service	What it is	Sign up
GitHub	Where your code lives. You already have access since you're reading this from a GitHub repo invite.	already done
Supabase	The database — stores your products, orders, customers.	https://supabase.com
Stripe	Processes credit-card payments.	https://stripe.com
Resend	Sends transactional email (order confirmations, etc.).	https://resend.com
Vercel	Hosts the live site.	https://vercel.com

You also need on your computer:

- **Node.js 22 or newer** — `node -v` in your terminal should print `v22.x` or higher. If not, install from https://nodejs.org

- **A code editor** — VS Code is free and works great: https://code.visualstudio.com

---

Step 1 — Get the code on your computer (3 min)

1. Open a terminal (on Mac: `Cmd + Space` → type "Terminal" → enter; on Windows: Start → "Terminal")
2. Type:

   ```
   git clone https://github.com/<YOUR-USERNAME>/<YOUR-REPO>.git my-store
   cd my-store
   ```

   Replace `<YOUR-USERNAME>/<YOUR-REPO>` with whatever the GitHub invite email gave you. After this you'll have a folder called `my-store` with all the files inside.
3. Install the dependencies (this is what fetches all the libraries the kit needs):

   ```
   npm install
   ```

   This takes 1-2 minutes. You'll see a wall of text — that's normal.

You'll know it worked when: the terminal prompt comes back to its normal state with no red `ERROR` lines. Some yellow warnings are fine.

Step 2 — Set up the database (Supabase) (10 min)

The database is where your products, customers, orders, and everything else live.

1. Open https://supabase.com/dashboard and sign in
2. Click the green **"New project"** button
3. Fill in:
   - **Name:** anything (e.g. "my-store")
   - **Database Password:** click **Generate a password** and save it somewhere safe (you almost never need it but losing it is annoying)
   - **Region:** pick the one closest to you
   - **Plan:** Free
4. Click **Create new project**. Wait ~90 seconds while Supabase provisions the database.

5. While that's spinning up, open the file `supabase/schema/current.sql` from your code folder in your editor. Select all (`Cmd+A` / `Ctrl+A`), copy (`Cmd+C` / `Ctrl+C`).
6. Back in Supabase: left sidebar → **SQL Editor** (icon looks like `>_`). Click **+ New query**. Paste the entire file. Click the green **Run** button (or `Cmd+Enter`).
7. You'll see "Success. No rows returned." That's good. The entire database structure is now in place.

Now grab your three Supabase keys. Sidebar → **Project Settings** (gear icon at the bottom) → **API**. You'll see three values you need (we'll paste them into a config file later):

- **Project URL** (something like `https://xyzabc123.supabase.co`)
- **anon / public** key (a long string starting with `eyJ...`)
- **service_role** key (also starts with `eyJ...`) — **NEVER share this one publicly**, it's the master key for your database

Copy these three to a temporary text file. We'll use them in Step 6.

Enable the Custom Access Token Hook (3 minutes)

This is a Supabase feature the kit needs to recognize who's an admin. One-time setup.

1. Sidebar → **Authentication** → **Auth Hooks** (it's at the bottom of the auth submenu, may have a "BETA" tag)
2. Find **Custom Access Token** → click the toggle to enable
3. **Schema:** `public` . **Function:** `add_admin_claim` (it should appear in the dropdown — that function was created when you pasted `current.sql`)
4. **Save**

Enable TOTP for two-factor authentication (1 minute)

Same Authentication menu → **Multi-Factor**. Toggle **TOTP** on. Save. (You'll enroll yourself later — this just unlocks the feature.)

Step 3 — Set up payments (Stripe) (10 min)

You'll start in **test mode** — Stripe has a separate sandbox so you can test the whole flow without using real money.

1. Open https://dashboard.stripe.com → make sure the **"Test mode"** toggle in the top-right is ON (it should be by default for new accounts)
2. Sidebar → **Developers** → **API keys**. You'll see two keys you need:
   - **Publishable key** (starts with `pk_test_...`)
   - **Secret key** — click "Reveal test key" (starts with `sk_test_...`)

3. Copy both. Save them with your Supabase keys for now.

You'll set up the webhook (the thing that tells your store when a payment succeeds) once your store is live on Vercel — Stripe needs to know your real URL first. We'll come back to that in Step 9.

Step 4 — Set up email sending (Resend) (5 min)

Order confirmation emails go out via Resend.

1. Open https://resend.com → sign up
2. Once in, sidebar → **API Keys** → **Create API Key**. Name it "steep dev". Copy the key (starts with `re_`). Save it.

That's it for now. You can configure your own sending domain later (the kit ships with a sensible default).

Step 5 — Set up GitHub fulfillment (5 min)

When buyers purchase a code/template product from your store, they get an automated invite to your private GitHub repo. To make that work, your store needs a GitHub access token.

1. Open https://github.com/settings/personal-access-tokens
2. Click **Generate new token** → **Fine-grained token**
3. Fill in:
   - **Token name:** "steep store"
   - **Expiration:** 1 year (you can rotate later)
   - **Resource owner:** your GitHub account
   - **Repository access:** "Only select repositories" → pick the repo(s) buyers will be invited to
   - **Repository permissions:**
     - **Administration:** Read and write (this is what lets the kit add collaborators)
     - **Metadata:** Read-only (gets added automatically)
4. **Generate token** → copy it (starts with `github_pat_...`)

> ***Read-only invites*** *— buyers receive collaborator access at the lowest permission level GitHub offers (`pull`). They can clone and read your code, but they cannot push, modify, delete branches, or change anything in your repo. Your code stays under your control.*

Step 6 — Wire it all together (10 min)

Now you'll create a config file with all the keys you've collected.

1. In your code folder, copy the example config to a real one:

```
cp .env.example .env.local
```

2. Open `.env.local` in your editor. You'll see a long file with empty `=` values.

3. Paste in the values you collected:

```
# From Supabase Step 2
NEXT_PUBLIC_SUPABASE_URL=https://xyzabc123.supabase.co
NEXT_PUBLIC_SUPABASE_ANON_KEY=eyJhbG...
SUPABASE_SERVICE_ROLE_KEY=eyJhbG...

# From Stripe Step 3
NEXT_PUBLIC_STRIPE_PUBLISHABLE_KEY=pk_test_...
STRIPE_SECRET_KEY=sk_test_...
STRIPE_WEBHOOK_SECRET=whsec_...   # we'll get this in Step 9

# From Resend Step 4
RESEND_API_KEY=re_...
ADMIN_ALERT_EMAIL=you@yourdomain.com

# From GitHub Step 5
GITHUB_TOKEN=github_pat_...

# Your store's URL — for now, use the local dev server
NEXT_PUBLIC_APP_URL=http://localhost:3001
NEXT_PUBLIC_APP_NAME=My Store
```

4. Save the file.

Don't worry about the optional sections (Sentry, Crisp, Turnstile, etc.) for now. You can turn those on later from `OPTIONAL_FEATURES.md`.

Step 7 — Boot it up (3 min)

```
npm run dev
```

After ~30 seconds you should see:

```
▲ Next.js 16.x.x (Turbo)
- Local:        http://localhost:3001
✓ Ready in 1.2s
```

Open http://localhost:3001 in your browser. You should see the storefront homepage.

If you see an error like "Environment configuration is incomplete": the message tells you which env var is missing. Fix it in `.env.local` and restart with `Ctrl+C` then `npm run dev` again.

Step 8 — Make yourself the admin (3 min)

Right now no one is an admin. You need to:

1. Sign up for a normal user account on your local store: open http://localhost:3001/login → click "Sign up" → use your real email → confirm via the email link Supabase sends.
2. Once signed up, go back to your Supabase dashboard → **SQL Editor** → **+ New query**. Paste this, replacing the email with the one you just used:

   ```
   UPDATE profiles
   SET role = 'admin'
   WHERE id = (SELECT id FROM auth.users WHERE email =
   'YOUR_EMAIL_HERE');
   ```

   Click **Run**. You'll see "Success. 1 row updated."
3. **Sign out and sign back in** on your store (the role change takes effect on the next login).
4. Click your profile icon → you should see an "Admin" link. Click it. You're in.

You're now in the admin panel where you can create products, manage orders, run promotions, etc. There's an onboarding checklist on the dashboard pointing out what's still left to set up.

Step 9 — Deploy to Vercel (15-30 min)

Time to put your store on the actual internet.

1. Open https://vercel.com → sign in (you can use your GitHub account — easiest)
2. Click **Add New** → **Project**
3. **Import Git Repository** — find your repo. Click **Import**.
4. **Configure Project** screen:
   - **Framework Preset:** Next.js (auto-detected)
   - Click **Environment Variables**. Now you need to paste in **every variable from your `.env.local` file**, one at a time. Vercel UI lets you paste a key=value pair quickly. Make sure to add them all.
   - **Important:** change `NEXT_PUBLIC_APP_URL` to the URL Vercel will give you (something like `https://my-store.vercel.app` — Vercel shows the URL above the deploy button) OR your custom domain if you've added one
5. Click **Deploy**. Watch the logs. Takes ~90 seconds.
6. When it's done, you'll get a public URL. Open it. Your store is live.

Set up the Stripe webhook now (5 min)

Stripe needs to call your live store when payments succeed. We couldn't do this in Step 3 because there was no live URL yet.

1. Open https://dashboard.stripe.com → **Developers** → **Webhooks** → **Add endpoint** (purple button top-right)
2. **Endpoint URL:** `https://YOUR-STORE-URL/api/webhooks/stripe` (use whatever Vercel gave you OR your custom domain)
3. **Listen to events** — click **Select events** → tick these:
   - `checkout.session.completed`
   - `charge.refunded`
   - `charge.dispute.created`
   - `payment_intent.payment_failed`
   - `checkout.session.expired`
4. **Add endpoint** → on the next page, click **Reveal** under "Signing secret". Copy the value (starts with `whsec_`).
5. Back in Vercel → your project → **Settings** → **Environment Variables**. Find `STRIPE_WEBHOOK_SECRET` → edit → paste the value. **Save** → **Redeploy** (Vercel asks; click yes).

Make yourself admin on the live site (2 min)

Same as Step 8 but on the live site:

1. Sign up at `https://YOUR-STORE-URL/login`

2. In Supabase SQL Editor, run the same `UPDATE profiles SET role = 'admin'` query but with the live email
3. Sign out and back in. You're admin on production.

What to do next

You have a live store. Here's what most sellers do in their first week:

1. **Create your first product** — admin → Products → New. Add name, price, description, upload an image. Set the fulfillment method (most likely "GitHub repo invite" with the repo set to whatever you want buyers to receive).
2. **Test a real purchase** — use a Stripe test card (`4242 4242 4242 4242` , any future expiry, any 3-digit CVC). Verify the order confirmation email arrives, the GitHub invite goes out, and the order shows up in `/admin/orders` .
3. **Set up a custom domain** — Vercel → your project → Settings → Domains → add your domain. Cloudflare or your registrar will need DNS pointing at Vercel.
4. **Switch Stripe to live mode** — when you're ready to take real money, follow the "Activate your account" flow in Stripe (business details, bank account). Then in Stripe dashboard toggle to "live mode", grab the live API keys (start with `pk_live_` / `sk_live_`), update Vercel env vars, redeploy.
5. **Turn on optional features** — read `OPTIONAL_FEATURES.md` for anti-bot CAPTCHA, BNPL badges, live chat, analytics, error monitoring.

When something doesn't work

Symptom	Most likely cause	Fix
`npm install` errors out	Wrong Node version	Run `node -v` . If not v22+, install from nodejs.org
Build fails on Vercel with "Environment configuration is incomplete"	Missing env var on Vercel	Vercel → Settings → Environment Variables → add the missing one. Redeploy with cache UNCHECKED.
`/admin` says 403 Forbidden	You forgot to run the SQL admin promotion	Step 8.2 — `UPDATE profiles SET role = 'admin'`

Stripe webhook fails repeatedly	Wrong webhook secret OR URL doesn't end in `/api/webhooks/stripe`	Re-check both in Stripe Dashboard + Vercel env
GitHub invites don't send after purchase	GitHub token doesn't have Administration: Write permission	Regenerate the token with the right permissions
TOTP enroll button does nothing	Forgot to enable TOTP in Supabase	Step 2 → Authentication → Multi-Factor → toggle TOTP
Buyer can't see the product	Product `status` is "draft"	Edit product → set status to "active"

For deeper troubleshooting, see `TROUBLESHOOTING.md`.

For everything-explained reference (auth, RLS, storage, cron jobs, etc.), see `DEEP_DIVE.md`.

For optional feature toggles (BNPL, live chat, analytics, etc.), see `OPTIONAL_FEATURES.md`.

Stuck? Email the support address on your invoice. We aim to respond within one business day.