perplexity

Opzetten van je Tech Stack in Replit

Om je stock2coat project met de voorgestelde tech stack succesvol op te zetten in Replit, volg je een gestructureerde aanpak die rekening houdt met de monorepo-architectuur en de verschillende componenten van je stack.

Initiële Setup

Stap 1: Replit Project Aanmaken

Begin met het aanmaken van een nieuwe Repl in Replit[1]. Je hebt twee opties:

- **Optie A**: Start met een Node.js template en bouw van daaruit verder
- **Optie B**: Start met een Turborepo template als deze beschikbaar is

Aanbevolen aanpak: Start met de Node.js template en configureer handmatig je monorepo structuur[2].

Stap 2: Turborepo Monorepo Setup

Configureer je monorepo structuur:

```
# In de Replit shell
npx create-turbo@latest stock2coat
cd stock2coat
```

Dit creëert automatisch de basis monorepo structuur met packages en apps directories[2] [3].

Project Structuur Implementeren

Frontend Setup (Next.js 14)

1. **Next.js App Configureren**[1]:

```
# In apps/web directory
npx create-next-app@14 . --typescript --tailwind --app
```

2. **shadcn/ui Installeren**[4] [5]:

```
npx shadcn@latest init
```

Configureer shadcn/ui met de volgende settings:

- TypeScript: Yes
- Style: Default of New York
- Base color: Zinc
- CSS variables: Yes

3. **Dependencies Installeren**:

```
npm install zustand @tanstack/react-query react-hook-form zod @tanstack/react-table
```

Backend Setup (Express.js)

1. **Express App Initialiseren**[6]:

```
# In apps/api directory
npm init -y
npm install express drizzle-orm @supabase/supabase-js zod cors
npm install -D typescript @types/node @types/express nodemon
```

2. **Drizzle ORM Configureren**[7] [8]:

```
npm install drizzle-orm drizzle-kit
```

Maak een `drizzle.config.ts` bestand:

```
import { defineConfig } from "drizzle-kit"

export default defineConfig({
  schema: "./src/db/schema.ts",
  out: "./drizzle",
  dialect: "postgresql",
  dbCredentials: {
    url: process.env.DATABASE_URL!,
  },
})
```

Database & External Services Setup

Supabase Database

1. **Supabase Project Aanmaken** en configureren[9] [10]
2. **Environment Variables Instellen** in Replit Secrets[11] [12]:

In de Replit Secrets tool voeg je toe:

- `DATABASE_URL`: Supabase connection string
- `SUPABASE_URL`: Project URL
- `SUPABASE_ANON_KEY`: Public anon key

Authentication met Clerk

Hoewel Clerk in je stack staat, is **Replit Auth** een eenvoudigere optie voor Replit-projecten[13] [14]:

```
# Voor Replit Auth (aanbevolen in Replit)
# Vraag aan Replit Agent: "Add Replit Auth to my app"
```

Alternatief voor Clerk:

```
npm install @clerk/nextjs
```

Ports en Development Setup

Multi-Port Configuratie

Replit ondersteunt meerdere ports voor monorepo development[15] [16]. Configureer je `.replit` bestand:

```
[[ports]]
localPort = 3000
externalPort = 80

[[ports]]
localPort = 8000
externalPort = 8000
```

Development Scripts

In je root `package.json`:

```
{
  "scripts": {
    "dev": "turbo dev",
    "build": "turbo build",
    "start": "turbo start"
  }
}
```

Secrets en Environment Management

Environment Variables Beheren

Gebruik Replit's Secrets feature[12] [17] voor gevoelige informatie:

1. Open de Secrets tool in de workspace
2. Voeg toe:
   - `DATABASE_URL`
   - `SUPABASE_URL`
   - `SUPABASE_ANON_KEY`
   - `CLERK_SECRET_KEY` (indien gebruikt)

Toegang tot Secrets in Code

```
// Node.js
const databaseUrl = process.env.DATABASE_URL

// Next.js
const supabaseUrl = process.env.NEXT_PUBLIC_SUPABASE_URL
```

Deployment Overwegingen

Replit vs External Deployment

Voor Development: Gebruik Replit's built-in hosting[18] [19]
Voor Production: Overweeg externe platforms zoals Railway of Vercel[20] [21]

Railway Deployment Setup

Als je kiest voor Railway deployment:

1. Connect je GitHub repository aan Railway
2. Configureer environment variables
3. Setup build commands voor je monorepo

CI/CD Pipeline

Voor automated deployment[22] [23]:

```
# .github/workflows/deploy.yml
name: Deploy
on:
  push:
    branches: [main]
jobs:
  deploy:
    runs-on: ubuntu-latest
    steps:
```

```
      - uses: actions/checkout@v3
      - name: Setup Node.js
        uses: actions/setup-node@v3
        with:
          node-version: '20'
      - name: Install dependencies
        run: npm install
      - name: Build
        run: npm run build
```

Praktische Tips

Replit-Specifieke Aandachtspunten

1. **Package Management**: Replit installeert automatisch packages wanneer je ze importeert[24]
2. **Hot Reloading**: Configureer nodemon voor backend development[25]
3. **File Structure**: Houd je aan de voorgestelde monorepo structuur voor optimale Turborepo performance

Development Workflow

1. Start beide services tegelijk:

```
# Terminal 1 - Frontend
cd apps/web && npm run dev

# Terminal 2 - Backend
cd apps/api && npm run dev
```

2. **Database Schema Changes**:

```
# Genereer migraties
drizzle-kit generate

# Apply migraties
drizzle-kit migrate
```

Troubleshooting Common Issues

- **Port Conflicts**: Gebruik Replit's Networking tool om ports te beheren[16]
- **Environment Variables**: Controleer of Secrets correct zijn ingesteld[12]
- **Package Installations**: Herstart de Repl als packages niet correct laden

Door deze systematische aanpak te volgen, kun je je complexe full-stack monorepo succesvol opzetten in Replit, waarbij je profiteert van zowel de ontwikkelingsgemak van Replit als de kracht van moderne development tools zoals Turborepo, Next.js 14, en Drizzle ORM.

⁂

Aanpak voor Stock2coat Saaas in 2025: Praktische, actiegerichte aanbevelingen

Hoofdadvies (inverted pyramid):
Voor een solo-developer in 2025 is **Next.js 14** de optimale keuze voor Stock2coat: het combineert razendsnelle development, ingebouwde SaaS-features en een lage leercurve, ideaal voor inventory-management apps waarvoor je snel een MVP wilt uitrollen.

1. Tech Stack Selectie

1.1 Full-stack framework vergelijking

Framework	Dev-snelheid	SaaS-features	Learning curve	Community
Next.js 14	Zeer hoog	API-routes, ISR, Edge Functions ingebouwd	Laag tot gemiddeld	Groot, Vercel-ecosysteem [26]
Remix	Hoog	Forms + loaders, data mutations	Gemiddeld	Groeiend [27]
SvelteKit	Hoog	Uitgebreide PWA-support, server-first	Laag	Stijgend [28]
Nuxt (Vue)	Hoog	Similar SSG, SSR	Gemiddeld	Groot
Laravel (PHP)	Gemiddeld	Batteries-included (queuing, billing)	Laag	Groot
Django (Python)	Gemiddeld	Admin UI, ORM, auth ingebouwd	Laag	Groot

Aanbeveling: kies Next.js 14 voor razendsnelle ontwikkeling met scherpe integratie naar Vercel (hosting, Analytics, Edge) en uitgebreide community en plugins voor forms, auth en realtime [26] [27].

2. SaaS Essentials Architectuur

2.1 Multi-tenant database

- **Shared schema (tenant_id)**: eenvoudig, goedkoop, één database, let op ‘noisy neighbor’ en correcte query-filters [29].
- **Separate schemas**: betere isolatie, nog betaalbaar tot ~100 tenants.
- **Database-per-tenant**: maximale isolatie, complex migraties, pas toe bij >500 klanten [29] [30].

Aanbeveling: start met *shared schema* (tenant_id) en monitor performance; stap eventueel over op separate schemas rond 200+ klanten.

2.2 Auth & Authorization

- Gebruik **Clerk** of **Auth0** voor volledige turnkey SSO, MFA, org-management.
- **Row-level permissions** implementeer via query-filters in ORM (Drizzle): `.where({tenant_id: ctx.tenantId})`.

2.3 Subscription & Payments

Provider	Voordelen	Nadelen
Stripe	API-centric, lage fees, wereldwijd	Zelf compliance regelen
Paddle	MoR: VAT + chargebacks inbegrepen	Duurder, minder API-flexibel

Aanbeveling: start met Stripe Billing (metered, usage-based) voor volledige controle[31]; implementeer via `@stripe/stripe-js`, server-side met `stripe.subscriptions.create()` en webhooks voor dunning.

3. MVP Features Prioritering

Feature	Prioriteit	Technologie/Implementatie
CRUD voorraad	Must	Next.js API routes + Drizzle ORM
Real-time updates	High	WebSockets via Pusher of Supabase Realtime
Gebruikersbeheer	Must	Clerk + RBAC via middleware
Dashboard & rapportages	High	React Chart.js, TanStack Query
Mobile-first design	Must	Tailwind CSS, shadcn/ui
Offline capability	Medium	PWA met service worker (Next.js PWA plugin)
Barcode scanning	Medium	Barcode Detection API in PWA of native via Capacitor[32]

Aanbeveling: implementeer eerst CRUD, auth en dashboard; voeg real-time en offline toe in v2; barcode in v3.

4. Development Workflow

- **AI-assisted coding**: gebruik Cursor Pro voor code completion en refactoring; Replit voor snelle prototypes.
- **Monorepo**: Turborepo v9 met apps/web (Next.js) en apps/api (Express + Drizzle).
- **Database**: PostgreSQL (Supabase) voor RLS en realtime features; lokaal SQLite voor dev.
- **CI/CD**: GitHub Actions → build/test → Vercel (frontend) & Railway (backend).

5. Kostenschatting & Hosting

Service	0–100 klanten/mo	Schaalbaarheid	Opmerkingen
Vercel Pro	€20–50	Auto scale	Frontend, Edge functions
Railway Hobby	Gratis–€20	Kleine units	Backend, Postgres add-on
Supabase Scale	Gratis–€25	RLS, Realtime	Database + storage
Redis (Upstash)	Gratis–€5	On demand	Caching, pub/sub

Maandkosten: ~€70–100 voor volledige stack, opschalen lineair met klant- en data-groei.

6. Tijdlijn & Milestones (solo, lancering 1 sept 2025)

Periode	Milestone
Jul–Aug ’25	Architectuur, Dev-omgeving, CI/CD, basis CRUD
Sep–Oct ’25	Auth, multi-tenant, database design
Nov–Dec ’25	MVP frontend, CRUD voorraad, gebruikers auth
Jan–Feb ’26	Dashboard, rapportages, real-time features
Mrt–Apr ’26	Offline PWA, barcode scanning integratie
Mei–Jun ’26	Subscription billing, onboarding flow
Jul–Aug ’26	Beta-tests, performance tuning, buffer

Buffer: 4 weken extra rond onverwachte issues.

7. Specifiek voor poedercoating industrie

- **Batch tracking & lot-traceability**: sla coating batches op per lot nr.
- **Compliance**: RAL-kleuren, recycling data, veiligheidsbladen.
- **Integraties**: ERP-connectors (SAP), weegschalen API’s.

9. Pricing & Billing Implementatie

- **Tiers**: Free (1 org), Starter (tot 5 MAU), Pro (tot 50 MAU), Enterprise (meer)
- **Trial**: 14-dagen, auto-converteer na CC validatie
- **Pricing model**: seat-based + usage (API calls)
- Gebruik Stripe Billing met `usage_record` en webhooks voor dunning en failed payments handling[31].

10. Security & Compliance

- **GDPR**: Data-lokalisatie EU, Data Processing Addendum
- **Backup**: dagelijkse dumps + PITR
- **Security**: OWASP Top10, CSP, helmet.js
- **RBAC**: rol-based middleware in Express; tenant filter in Next.js middleware

Met deze actiegerichte roadmap, tech-stack en concrete codevoorbeelden (Drizzle tenant filter, Stripe integratie, React barcode scanner) ben je klaar om van je ML Coating webapp een schaalbaar Stock2coat-SaaS te maken.

⁂

Overzicht van ingebouwde SaaS-features voor Stock2coat

Hoofdadvies: implementeer een robuuste basis set SaaS-essentials (multitenancy, auth, provisioning, billing, monitoring, audit) én pak poederlak-specifieke workflows mee (batch-tracking, receptbeheer, jobbeheer) voor een schaalbare, onderhoudsvriendelijke web-app.

1. Must-have SaaS-features (volgens r/SaaS)

Feature	Omschrijving	Citaat
User & account management	Authenticatie (SSO, social login), registratie, verificatie, wachtwoordherstel	[33]
Email flows	Automatische e-mails voor onboarding, wachtwoordreset, facturen, notificaties	[33]
Billing & subscriptions	Recurring billing, seat-based en usage-based modellen, facturatie, webhooks	[33] [34]
Multi-tenancy	Shared schema met `tenant_id` of per-schema isolate, automatische provisioning	[34]
Automated provisioning	On-demand tenant- en user provisioning/de-provisioning via API	[34]
High availability & scaling	Elastic infrastructure, autoscaling, API-monitoring	[34] [35]
Security & compliance	Data-encryptie, RLS, OWASP-protectie, GDPR, audit logs	[34] [33]
Logging & monitoring	Error-tracking, health checks, metrics, alerting	[33]
Rate limiting/QoS	API-throttling, tiers, QoS-configuratie	[34]
Audit & analytics	Gebruikers- en transactie-logs, BI-dashboards for usage en performance	[34]

2. Poederlak-specifieke workflows en features

Analyse van bestaande poederlak-software (Steelhead, Bluestreak, Colour Werx[Best Powder Coating Software]):

Feature	Toelichting
Batch- & lot tracking	Lot-nummers, productie-batch, traceerbaarheid
Receptbeheer	Beheer van poederrecepten (kleur, samenstelling, additieven)
Work-order & job management	Quotes, opdrachten, job status, werkplanning, toewijzing aan productie-lijnen
Container & rack management	Track & trace voor rekken, karren, dozen
Barcode & scan integratie	Barcode-/QR-scanner-integratie voor werkvloer, PWA of native
Digital approvals	Klanten- en QC-handtekeningen, automatische certificaten
Rapportages & analytics	KPI's: doorlooptijd, foutmeldingen, verbruik, rendement
Mobiele UI / offline	PWA met caching, offline voorraadbeheer
Integraties ERP/ERP-connectors	SAP, Odoo, weegschalen, MES, weeg-en weegdata

3. Reddit-inspiratie en community-insights

- **Reddit r/microsaas**: B2B-IMS trekt features als facturatie, containerbeheer, werkorders, stock transfers, gebruikers-rechten, kleine productieondersteuning[36].
- **Marketing op Reddit**: authentieke AMA-posts in r/SaaS en r/Powdercoating, target subreddits, zorg voor waarde (niet pure ads)[37].

4. Concrete aanbevelingen

1. **Multitenant architectuur**: shared database met `tenant_id`, Zod/Drizzle filters op alle queries.
2. **Auth & provisioning**: integratie met Clerk/Auth0 voor SSO/MFA; auto-provision via webhook/Azure-AD-SCIM.
3. **Billing & metering**: Stripe Billing met usage_records voor poederverbruik (kg), seat-based licenties, webhooks voor dunning.
4. **Jobflow module**:
   - CRUD voor job orders + batchnummers
   - Recept-entiteit met kleur/secundaire additieven
   - Barcode scanning component (zowel PWA ServiceWorker als native Capacitor)
5. **Reële-time updates**: Supabase Realtime of Pusher voor live job-status en voorraadwijzigingen.
6. **Mobile-first PWA**: Next.js PWA plugin, cache voorraaddata, background sync voor offline.

7. **Compliance & audit**: Drizzle ORM hooks logging, geïntegreerde audit-logs (ELK/Sentry), dagelijkse DB backups + PITR.
8. **Integraties**: ERP (SAP/Odoo connector), weegschalen API, MES-link via webhooks.
9. **UI/UX**: shadcn/ui + Tailwind; dashboards met Chart.js voor rendements- en verbruiksinzichten.
10. **Community engagement**: lanceer AMA op r/SaaS en r/Powdercoating vóór lancering; deel sneak peeks, verzamel feedback.

Met deze featureset en werkwijze bouw je een toekomstbestendige Stock2coat-SaaS, specifiek toegespitst op de poederlakindustrie en gebaseerd op best practices uit de SaaS-community [33] [34] [36].

⁂

Overzicht van ingebouwde SaaS-features voor Stock2coat

Hoofdadvies:
Kies voor **Next.js 14 + Supabase** en implementeer vanaf dag 1 een robuuste set SaaS-essentials—multitenancy, auth, provisioning, billing, monitoring, audit—én verrijk met poederlak-specifieke workflows (batch-tracking, receptbeheer, jobbeheer) voor een schaalbare, onderhoudsvriendelijke web-app.

1. Onmisbare SaaS-features

Feature	Omschrijving
Multi-tenancy	Shared DB + RLS per `tenant_id` in Supabase; subdomeinen (acme.stock2coat.com) voor premium UX
Authenticatie & autorisatie	Clerk voor SSO, MFA, organisaties; RBAC met roles (Owner, Manager, Operator)
Provisioning	Automatisch tenant- en user-provision via Next.js middleware & webhooks
Billing & subscriptions	Stripe Billing: seat-based + usage (API-calls, kg coating); webhooks voor dunning
Onboarding & activation	Conversational UI met @chatscope/chat-ui-kit-react; focus: time-to-first-value (< 5 min installatie)
Monitoring & logging	Sentry (Free t/m 50 K events; Team $26 p/m [38]); ELK of Vercel Analytics voor performance & uptime alerts
Security & compliance	TypeScript + Zod-validators, parameterized queries, CSP, helmet.js; GDPR: data-export & -deletion API's
Audit trails & backups	DB-level logging via Drizzle hooks; dagelijkse dumps + PITR in Supabase Pro

2. Poederlak-specifieke workflows

Feature	Toelichting
Batch- & lot-tracking	Lot-nr, batch, traceerbaarheid; RLS-beleid per tenant
Receptbeheer	Beheer poederrecepten (kleur, additieven, mengverhoudingen)
Job management	CRUD job-orders, status-updates, werkplanning, toewijzing aan lijnen
Container & rack management	Track & trace rekken, karren; barcodescanner in PWA (Barcode Detection API)
Compliance alerts	Automatische waarschuwingen voor SDS-vervaldatum, afval-meldingen volgens RCRA/EPA
QC-documentatie	Voor/na foto's (Cloudinary), dikte-logs, temperatuur-schema's
ERP-/MES-integraties	Webhook-connectors voor SAP/Odoo, weegschalen API's

3. MVP-featuresprioritering

Feature	Prioriteit	Implementatie
CRUD voorraad	Must	Next.js API-routes + Drizzle ORM + Supabase RLS
Auth & gebruikersbeheer	Must	Clerk + middleware-checks `.where({tenant_id})`
Dashboard & rapporten	High	TanStack Query + Chart.js
Real-time updates	Medium	Supabase Realtime of Pusher
Mobile-first & PWA	Must	Tailwind + Next.js PWA plugin + service worker
Offline capability	Low	Background sync + IndexedDB
Barcode scanning	Low	PWA Barcode Detection API

4. Development workflow

1. **Monorepo** via Turborepo: `apps/web` (Next.js) + `apps/api` (Express + Drizzle).
2. **AI-assisted**: rapid prototyping in Replit Core, production uitbouw met Cursor Pro & Claude.
3. **DB**: Supabase (PostgreSQL) lokaal SQLite dev[39]; RLS per tenant; backups in Pro (€25 p/m)[40].
4. **CI/CD**: GitHub Actions → Vercel (frontend, Pro @ €20/user p/m[41] [42]) + Railway Hobby ($5 p/m + $5 usage credit[43]).
5. **Secrets**: beheer in Replit Secrets; gebruik `process.env`.

5. Kosten en hosting

Component	0–10 klanten	10–30 klanten	30–100 klanten
Vercel	Hobby (100 GB BW)	Pro €20 p/seat[41]	Pro + overage
Supabase	Free (500 MB, 50 K MAU)[39]	Pro €25 p/project[44]	Scale (€…)

Component	0–10 klanten	10–30 klanten	30–100 klanten
Railway	Hobby €5 p/m + credits [43]	Hobby + usage	Pro €20 p/m + usage
Neon	Free (0.5 GB, 190 CU-h) [45]	Launch €19	Scale €69
Sentry	Free	Team €26 [46]	Business €80 [46]

> **Totaal initieel** ≈ €70–100/m [39] [43] [44] [41] [46].

6. Week-by-week roadmap (8 wkn)

Wkn 1–2	Foundation: Next.js + Supabase multi-tenancy & Clerk auth + RLS
Wkn 3–4	Core ML Coating → CRUD voorraad + batch- & receptbeheer
Wkn 5	Stripe Billing basics (customers, subscriptions, webhooks) [47]
Wkn 6	Dashboard + rapportages + real-time flows
Wkn 7	PWA offline & barcode scanning integratie
Wkn 8	Onboarding flow (conversational UI) + compliance alerts
Buffer (wk 9)	Bugfixes, performance tuning, beta-tests

7. Conclusie

Met **Next.js 14 + Supabase**, AI-tools & een scherp gefaseerde roadmap kun je als solo developer Stock2coat binnen 8 weken tot MVP brengen. Bouw eerst de kern (multitenancy, auth, CRUD), dan differentiators (batch-tracking, compliance) en kies de juiste hosting & tooling (Vercel, Railway, Supabase Pro). Hiermee positioneer je Stock2coat als premium inventory-SaaS voor de poederlakindustrie.

⁂

Hostingmix uitgelegd met een metafoor

Hostingmix in fase 1 = een kant-en-klaar tosti-ijzer

- Je koopt een eenvoudig tosti-ijzer (Vercel + Supabase)
- Het toestel doet precies wat je nodig hebt: brood roosteren en kaas smelten
- Geen gedoe met losse pannen, speciale hitte-instellingen of extra apparaten

Zodra je tosti's specialer worden (bijvoorbeeld een croque monsieur met béchamelsaus, ham én een laagje truffelmayo), merk je dat het simpele tosti-ijzer tekortschiet. Dan schakel je over naar:

- Een multifunctionele keukenmachine (Railway, Fly.io, Hetzner)
- Die biedt extra standen (background jobs), uitneembare delen (custom networking) en meer vermogen voor complexe recepten

Kort samengevat

- **Fase 1 (MVP)**: Vercel + Supabase = je compacte tosti-ijzer voor een snel, betrouwbaar en onderhoudsarm 'tosti-MVP'.

- **Fase 2 (opschalen)**: overstappen wanneer je recepten complexer worden en je extra kracht, fijninstellingen of speciale tools nodig hebt.

Wanneer overstappen van Vercel + Supabase Hobby/Free?

Kernboodschap (inverted pyramid):
Met één Stock2coat-project op Vercel Hobby en Supabase Free kun je makkelijk de eerste ~20 klanten bedienen. Zodra je structureel meer dan ~100 GB bandbreedte per maand, >500 MB database of >5 GB egress nodig hebt, of intensieve realtime- en achtergrondtaken draait, schakel je best naar betaalde plannen.

1. Vercel Hobby ("gratis") limieten [48] [49] [50] [51]

Resource	Hobby (gratis)	Periode
Function invocations	1 000 000	per maand
Fast data transfer	100 GB	per maand
Build tijd	100 uur	per maand
Concurrency (CPU-uren)	4 CPU-uur	per maand
Bandbreedte CDN	100 GB	per maand
Project-count	200 projecten maximaal	–
Project-pauze bij overschrijding	Ja (30 dagen reset)	30 dagen na overschrijding

- **Praktisch:** met 20 klanten die elk 5 GB / maand downloaden (bvb. rapport-PDF's, foto's) ben je aan 100 GB bandbreedte. Bij zwaarder gebruik pauzeert Vercel je site en moet je upgraden [50].
- **Fauteuil-metafoor:** Vercel Hobby is je "zitbank" – comfortabel voor 1–2 personen, maar vol bij een klein feestje (>5 mensen).

2. Supabase Free Tier limieten [52]

Resource	Free Tier
Postgres database	500 MB
File storage	1 GB
Bandwidth (egress+ingress)	5 GB
MAU (auth)	50 000
Edge Function calls	500 000 calls
Realtime connections	200 concurrent
Project-pauze bij inactiviteit	Ja (na 1 week)

- **Data-gebruik:** als elke klant 25 MB database en 50 MB uploads gebruikt, kun je ~10 klanten vlot bedienden. 20 klanten brengt je tegen 1 GB opslag-limit.
- **Applicatie-voorbeeld:** 20 klanten × 100 acties/maand × realtime-updates = 20 000 realtime-events, prima binnen limiet. Maar >200 gelijktijdige websockets overschrijdt Free Tier[53].

3. Praktische grens & overstapmoment

1. **1–10 klanten:** puur Vercel Hobby + Supabase Free; prototype-fase.
2. **10–20 klanten:** houd gebruiksstatistieken scherp in de gaten:
   - Bandbreedte Vercel ≈ 100 GB/maand
   - Supabase opslag ≈ 1 GB & egress 5 GB
3. **20–50 klanten:**
   - Upgraden Vercel → Pro (€20 per seat) voor 1 TB bandbreedte, 16 CPU-uur[48].
   - Upgraden Supabase → Pro ($25 / project) voor 8 GB DB, 100 GB opslag, 250 GB bandbreedte en hogere realtime-quota[52].
4. **>50 klanten of intensieve achtergrondtaken (batch-jobs, heavy analytics):**
   - Overweeg dedicated compute (Railway, Fly.io) en Neon/Postgres-branches.

4. Samengevatte analogie

- **Fase 1 (0–10 klanten):** Vercel Hobby + Supabase Free = je **eenvoudig stadsfietsje**: goedkoop, wendbaar, perfect voor korte ritten.
- **Fase 2 (10–20 klanten):** fiets kinderkarretje → **goede elektrische fiets** (Vercel Pro + Supabase Pro): meer power, grotere accu.
- **Fase 3 (>20 klanten of zware taken):** neem je **bestelwagen** (Railway/Hetzner + Neon): hogere capaciteit, eigen motor, geen limieten meer.

Zo heb je een heldere richtlijn wanneer je met je groeiende klantenbestand van poederlakbedrijven de overstap maakt van gratis tiers naar betaalde, schaalbare infrastructuur.

⁂

Conclusie

Stock2coat's hostingstrategie groeit mee met je klantenbestand: begin eenvoudig en goedkoop met Vercel Hobby + Supabase Free, en schakel pas over zodra gebruikslimieten – bandbreedte, opslag, realtime-quota of achtergrondjobs – consequent overschreden worden.

- **0–10 klanten**: Vercel Hobby + Supabase Free = stadsfietsje
- **10–20 klanten**: Vercel Pro (€20/seat) + Supabase Pro (€25/project) = e-bike
- **>20 klanten of zware taken**: gespecialiseerde infrastructuur (Railway/Fly.io/Hetzner + Neon/Postgres) = bestelwagen

Zo houd je de infrastructuurkosten laag tijdens MVP en kleine groei, en heb je een helder schaalpad om soepel op te schalen zodra je meer power, opslag en netwerk-functies nodig hebt.

Samenvatting voor jou als solo Developer

Belangrijkste uitgangspunten:

- **Tech Stack:** Next.js 14 (App Router) + Supabase (PostgreSQL met RLS)
- **Monorepo-structuur:** Turborepo met `apps/web` (Next.js) en `apps/api` (Express + Drizzle ORM)
- **AI-tools:** Replit Core voor prototyping, Cursor Pro & Claude voor productie-code en architectuur

1. Architectuur & SaaS Essentials

- **Multi-tenant:** gedeeld schema met `tenant_id` + RLS; subdomeinen voor elke klant
- **Authenticatie:** Clerk voor SSO, MFA, org-management; RBAC (Owner/Manager/Operator)
- **Provisioning:** Next.js middleware + webhooks voor automatische tenant- en user-setup
- **Billing:** Stripe Billing met seat-based en usage-based componenten; webhooks voor dunning
- **Onboarding:** Conversational UI (@chatscope/chat-ui-kit-react) → time-to-first-value < 5 min
- **Logging & monitoring:** Sentry (error-tracking), Vercel Analytics (performance), ELK voor audit logs
- **Security & compliance:** CSP, helmet.js, TypeScript+Zod, parameterized queries, GDPR (export/deletion API)
- **Backups & audit:** Drizzle hooks voor changelogs, dagelijkse dumps + PITR in Supabase Pro

2. MVP-featureprioriteit

Prioriteit	Feature	Technologie
Must	CRUD voorraad	Next.js API-routes + Drizzle + RLS
Must	Auth & gebruikersbeheer	Clerk + tenant-filters
High	Dashboard & rapportages	TanStack Query + Chart.js
Medium	Real-time updates	Supabase Realtime of Pusher
Must	Mobile-first & PWA	Tailwind + Next.js PWA plugin
Low	Offline capability	Service worker + IndexedDB
Low	Barcode scanning	PWA Barcode Detection API

3. Hosting & Kosten (stapsgewijs)

1. **0–10 klanten:** Vercel Hobby + Supabase Free (~€0)
2. **10–20 klanten:** Vercel Pro (€20/seat) + Supabase Pro (€25/project)
3. **>20 klanten of zware achtergrondtaken:** Railway/Fly.io/Hetzner + Neon/Postgres (vanaf ~€60–100/m)

4. Ontwikkel-workflow

1. **Setup:** `npx create-turbo` → monorepo
2. **Prototype:** Replit Core
3. **Productie:** Cursor Pro & Claude – AI-assisted coding
4. **CI/CD:** GitHub Actions → Vercel (frontend) & Railway (backend)
5. **Secrets:** beheer in Replit Secrets / Vercel Environments

5. Tijdlijn (8 weken + buffer)

- **Wk 1–2:** Multi-tenant + auth + RLS
- **Wk 3–4:** CRUD voorraad + batch- & receptbeheer
- **Wk 5:** Stripe Billing basics + webhooks
- **Wk 6:** Dashboard + real-time flows
- **Wk 7:** PWA offline + barcode scanning
- **Wk 8:** Conversational onboarding + compliance alerts
- **Buffer:** wk 9 voor bugfixes & beta-tests

6. Poederlak-specificaties

- **Batch-tracking & lot-traceability**
- **Receptbeheer** (kleur, additieven)
- **Job management & werkorders**
- **Compliance-alerts** (SDS-expiratie, EPA waste manifests)
- **QC-documentatie** (foto's, dikte-logs)
- **ERP/MES-integraties** via webhooks

Kortom: implementeer eerst de kern (multitenancy, auth, CRUD), bouw snel de MVP-features, houd infrastructuur eenvoudig met Vercel + Supabase, en schakel gefaseerd op naarmate je groeit. Hiermee lanceer je Stock2coat binnen je gestelde tijd en budget als een schaalbare, industrie-specifieke SaaS.

Analyse van het `stock2coat-vanilla.html` voorraadoverzicht

Onderstaande tabelstructuur komt uit het aangeleverde HTML-bestand:

Kolom	Beschrijving
RAL Code	Unieke kleurcode volgens RAL-systeem
Merk/Product	Naam van leverancier en product (poedermerk en type)
Status	Indicator: “LAAG”, “GEM” (gemiddeld), “OK”
Voorraad	Hoeveelheid beschikbaar
Locatie	Fysieke opslagplaats (magazijnsectie, reknummer, etc.)
Laatst Bijgewerkt	Datum/tijd van laatste voorraadaanpassing
Acties	Knoppen of links voor bewerkingen (bijv. “bewerken”, “bestellen”)

Bij het inladen bleek de voorbeeldtabel geen itemsrecords te bevatten (lege `<tbody>`).

Verbeterpunten en aanbevelingen

1. **Data-weergave**
   - Toon direct een voorbeeldrij of “Geen resultaten”-melding in lege staat.
   - Voeg kolom-icons voor sortering (‘↑↓’) en filter-knoppen per kolom toe.
2. **UX-verrijking**
   - **Voorvertoning kleurvlakje** naast de RAL Code in plaats van alleen tekst.
   - **Inline badges** voor `Status` (rood/amber/groen) om leesbaarheid te verhogen.
3. **Functionaliteit**
   - **Zoeken en filteren**: vrije tekst + dropdown-filters per kolom (status, locatie).
   - **Bulk-acties**: selectievakjes per rij met bewerk- en bestelopties voor meerdere items tegelijk.
   - **Paginering**: configureerbare rijen-per-pagina (10/25/50/100), plus “ga naar pagina”-navigatie.
4. **Performance**
   - **Server-side paging & filtering** met TanStack Query voor grote datasets.
   - **Virtualisatie** (react-virtualized) bij >1.000 rijen om rendervertraging te voorkomen.
5. **Responsiviteit**
   - Maak mobiele weergave compact: verberg kolommen “Locatie” en “Laatst Bijgewerkt” achter een uitklap-icoon.
   - Voeg horizontaal scrollen of kolom-picker toe op smalle schermen.
6. **Toegankelijkheid (a11y)**
   - Voorzie `aria-sort` en `aria-label` op sortable headers.

- Zorg dat `Acties`-knoppen voldoende contrast en `aria-describedby` hebben.

Met deze aanpassingen wordt het voorraaddashboard zowel gebruiksvriendelijker, performant als onderhoudbaar voor de poedercoating-workflow.

⁂

Eerste prompt voor Replit

```
Projectnaam: stock2coat

Beschrijving:
Ik bouw een multi-tenant SaaS-inventorysysteem voor poedercoatbedrijven met Next.js 14 (A
Doel: een werkende monorepo-MVP opzetten met frontend, backend en CI/CD.

Taken:
1. Initialiseer een Turborepo-monorepo:
   • apps/web → Next.js 14 (TypeScript, App Router, Tailwind CSS)
   • apps/api → Express.js + Drizzle ORM
2. Configureer Supabase:
   • Datamodel voor tenanten met `tenant_id` & RLS-policies
   • Auth-integratie via Clerk (SSO + MFA)
3. Zet dev-omgeving op:
   • .replit-config voor frontend (poort 3000) en backend (poort 8000)
   • Replit Secrets: DATABASE_URL, SUPABASE_KEY, CLERK_KEY
4. Scripts:
   • `npm run dev:web` → Next.js frontend
   • `npm run dev:api` → Express backend
5. CI/CD:
   • GitHub Actions → Vercel (frontend) & Railway (backend)

Belangrijkste doel deze sessie: een werkend skeleton-project met:
- Next.js-pagina’s en API-routes
- Drizzle-schema + migratie
- Supabase-auth met RLS
- Run-commando’s in Replit

Voeg direct een voorbeeldcomponent toe: dashboardpagina met “Hello, Tenant!”.
```

⁂

1. https://docs.replit.com/getting-started/quickstarts/next-js-app
2. https://gist.github.com/cedrickchee/dfdb66c457c7b9e1682feedcc4fd6302
3. https://vercel.com/new/templates/next.js/monorepo-turborepo
4. https://dev.to/sudhanshudevelopers/how-to-use-shadcn-ui-in-your-react-vite-project-5g0h
5. https://www.youtube.com/watch?v=1SWpboGyRII
6. https://dev.to/sandrarodgers/basic-express-server-using-replitcom-2ba9
7. https://orm.drizzle.team/docs/get-started
8. https://dev.to/devgauravjatt/zero-config-db-setup-with-drizzle-orm-using-drizzle-setup-cli-4905

9. https://supabase.com/blog/using-supabase-replit
10. https://blog.replit.com/making-real-time-chat-app-with-supabase-on-replit
11. https://www.reddit.com/r/replit/comments/1jbwfe9/programmatically_updating_secrets/
12. https://docs.replit.com/replit-workspace/workspace-features/secrets
13. https://blog.replit.com/auth
14. https://docs.replit.com/replit-workspace/replit-auth
15. https://docs.replit.com/replit-workspace/ports
16. https://blog.replit.com/ports
17. https://blog.replit.com/secrets
18. https://blog.replit.com/deployments-updates
19. https://www.youtube.com/watch?v=PTie3KQhos8
20. https://www.reddit.com/r/replit/comments/1l2rh50/free_beginner_tips_for_using_replit_from_someone/
21. https://www.reddit.com/r/replit/comments/1l6heu1/struggling_with_deploying_on_railway/
22. https://www.rapidevelopers.com/replit-tutorial/how-to-integrate-a-ci-cd-pipeline-with-replit-for-automated-deployments
23. https://www.codementor.io/@manishshrivastava/ci-cd-via-github-actions-just-2-min-copy-paste-done-12edqvoi63
24. https://dev.to/logrocket/using-replit-with-nodejs-to-build-and-deploy-apps-54gi
25. https://docs.replit.com/getting-started/quickstarts/from-scratch
26. https://javascript.plainenglish.io/next-js-remix-or-sveltekit-the-ultimate-guide-top-10-differences-03953ca20af5
27. https://merge.rocks/blog/remix-vs-nextjs-2025-comparison
28. https://prismic.io/blog/sveltekit-vs-nextjs
29. https://www.bytebase.com/blog/multi-tenant-database-architecture-patterns-explained/
30. https://dev.to/tak089/multi-tenant-architecture-a-complete-guide-basic-to-advanced-119o
31. https://ecommerce-platforms.com/articles/paddle-vs-stripe
32. https://progressier.com/pwa-capabilities/qr-code-and-barcode-reader
33. https://www.reddit.com/r/SaaS/comments/vixqke/what_are_some_standard_features_most_saas_apps/
34. https://www.jamcracker.com/blogs/features-characteristics-software-as-a-service-saas
35. https://www.netguru.com/blog/saas-features
36. https://www.reddit.com/r/microsaas/comments/1koqbnz/built_a_comprehensive_inventory_management_system/
37. https://www.factors.ai/guides/saas-reddit-ads/why-saas-companies-should-choose-reddit
38. https://stripe.com/docs/implementation-guides/billing
39. https://www.linkedin.com/pulse/free-tier-web-hosting-status-report-2025-opportunity-hack-ik7fc
40. https://codehooks.io/docs/alternatives/supabase-pricing-comparison
41. https://www.withorb.com/blog/vercel-pricing
42. https://vercel.com/docs/plans/pro
43. https://docs.railway.com/reference/pricing/plans

44. https://www.supadex.app/blog/supabase-pricing-what-you-really-need-to-know
45. https://neon.com/pricing
46. https://www.g2.com/products/sentry/pricing
47. https://www.reddit.com/r/SaaS/comments/16vntj7/how_long_does_it_take_to_you_integrate_a_payment/
48. https://vercel.com/docs/limits
49. https://vercel.com/docs/plans/hobby
50. https://stackoverflow.com/questions/78257939/vercel-usage-limits-when-does-it-reset-or-will-it-reset
51. https://www.reddit.com/r/nextjs/comments/1cfxuz1/what_happens_when_you_outspend_limits_on_hobby/
52. https://www.supadex.app/blog/supabase-pricing-what-you-really-need-to-know
53. https://supabase.com/docs/guides/realtime/quotas