na correctie chatgpt en claude als plan maker:

Here is Claude's plan:

Restaurant Auto-Import Pipeline — Startplan

Context

Recepten.pb.nl heeft nu 69 handmatig toegevoegde restaurants. Om op te schalen willen we een geautomatiseerde pipeline bouwen die:
- Restaurants ophaalt van de Michelin Guide NL als seed-list
- Feitelijke data verzamelt van restaurant-websites
- Screenshot-gebaseerde hero images maakt
- Eigen redactionele teksten genereert met OpenAI (GPT-4o)
- Alles als geldige JSON in de database zet met auto-translate NL→EN

De pilot: 10 restaurants, volledig automatisch, publiceerbaar zonder schaamte.

Architectuur — 3 Lagen

```
┌─────────────────────────────────────────────┐
│  LAAG 1: Brondata verzamelen                │
│                                             │
│  Michelin Guide NL  ──→  seed-list (10 rest.)  │
│       +                                     │
│  Restaurant website ──→  telefoon, uren, adres  │
│       +                                     │
│  Screenshot service ──→  hero image            │
└─────────────────────────────────────────────┘
                    ▼
┌─────────────────────────────────────────────┐
│  LAAG 2: Feitelijke validatie               │
│                                             │
│  Normaliseer velden naar ons DB-schema      │
│  Verwijder niet-verifieerbare data          │
│  Deduplicatie check (naam + website domain) │
└─────────────────────────────────────────────┘
                    ▼
┌─────────────────────────────────────────────┐
│  LAAG 3: Redactionele generatie (OpenAI)    │
│                                             │
│  Input: gevalideerde brondata               │
│  Output: intro, seoText, tags, cuisine      │
│  → Alleen schrijven op basis van echte signalen │
│  → Nooit feiten verzinnen                   │
└─────────────────────────────────────────────┘
                    ▼
┌─────────────────────────────────────────────┐
│  OUTPUT                                     │
│                                             │
│  Upload image → Cloudflare R2                │
│  Insert restaurant → Neon PostgreSQL         │
│  Auto-translate NL→EN (bestaande pipeline)   │
│  Revalidate pagina's                        │
└─────────────────────────────────────────────┘
```

Bestaande code die we hergebruiken

Component	Bestand	Wat het doet
OpenAI client	app/lib/translate.ts	Singleton GPT-4o client, temp 0.3, JSON mode
Restaurant schema	shared/schema.ts:207-251	27 velden incl. phone, openingHours, address
Import route	app/api/restaurants/import/route.ts	Flexibele JSON→DB mapping met auto-slug
Auto-translate	app/api/restaurants/route.ts POST	NL→EN vertaling bij insert
R2 upload	app/api/objects/upload/route.ts	Sharp JPEG compressie → Cloudflare R2

Revalidatie	app/lib/revalidate.ts	ISR cache invalidatie per restaurant
Auth	app/lib/auth.ts	HMAC-SHA256 cookie auth
DB	lib/db.ts	Neon PostgreSQL + Drizzle ORM, max 5 pool

Stap 1 — Michelin Seed-List

Probleem

Michelin Guide blokkeert bots agressief. Scrapen is fragiel.

Oplossing

Voor de pilot: handmatige seed-list. We kiezen 10 restaurants van guide.michelin.com/nl/nl/restaurants die nog NIET in onze database staan. Per restaurant noteren we:

```
{
  "name": "Restaurant X",
  "city": "Amsterdam",
  "michelin_cuisine": "Modern French",
  "michelin_price": "€€€€",
  "michelin_distinction": "1 ster",
  "michelin_summary": "Korte Michelin beschrijving...",
  "website": "https://www.restaurantx.nl"
}
```

Later (post-pilot)

Michelin Guide scraper bouwen met rate-limiting, of Google Places API als alternatieve bron.

Stap 2 — Website Scraping

Per restaurant scrapen we de website voor feitelijke data:

Veld	Bron	Methode
phone	Website, Google	WebFetch + regex
openingHours	Website, Google	WebFetch + parsing
reserveUrl	Website	Link detection (couverts, thefork, formitable, etc.)
addressStreet	Website	Structured data / schema.org
addressZipcode	Website	Structured data
addressCity	Website	Structured data
addressMaps	Gegenereerd	google.com/maps/search/?api=1&query=naam+stad
website_signals	Website	Meta description, headings, body keywords

Techniek

- WebFetch voor HTML ophalen → cache als raw_html_{slug}.html (voor later hergebruik)
- Parse schema.org/JSON-LD structured data (veel restaurants hebben dit)
- Phone parsing prioriteit: 1) schema.org, 2) <a href="tel:..."> links, 3) regex
- openingHours: ALLEEN uit schema.org. Als die ontbreken → null. Niet proberen vrije tekst te parsen.
- Opslaan als raw_sources.json per restaurant (voor audit trail)

Stap 3 — Screenshot Hero Image

Aanpak

Screenshot van de restaurant homepage als hero image.

Service

Optie A (pilot): thum.io — gratis, geen API key nodig
https://image.thum.io/get/width/1200/https://restaurantwebsite.nl
Geen crop — veel sites hebben video headers, cookie banners, of hero onder de fold.
Full-page screenshot op 1200px breed, later handmatig bijsnijden als nodig.

Optie B (schaalbaar): microlink.io — meer controle, betaald bij volume

Flow

1. Screenshot ophalen (timeout: 10s, retry: 2x)
2. Comprimeren met Sharp (JPEG Q80) — bestaande logica in upload route
3. Upload naar R2: restaurants/{slug}/hero.jpg
4. URL opslaan als imageUrl

Fallback keten

screenshot (thum.io, 2 retries)
    ↓ mislukt
OG image van website (og:image meta tag)
    ↓ mislukt
geen image (null) — handmatig later toevoegen

Juridisch

Screenshots van publiek toegankelijke websites voor editorial/directory gebruik is juridisch veilig (vergelijkbaar met Google zoekresultaten, Yelp, TripAdvisor).

Stap 4 — OpenAI Redactionele Generatie

Model & configuratie

- Model: gpt-4o (consistent met bestaande translate code)
- Temperature: 0.5 (iets creatiever dan vertaling, maar nog consistent)
- Response format: JSON Schema (structured output)
- Max tokens: 1024

System Prompt

Je bent de redacteur van Recepten.pb.nl, een culinaire site van Peter Blaas.
Je schrijft korte, warme restaurantbeschrijvingen in het Nederlands.

Regels:
- Schrijf in een persoonlijke, beschrijvende stijl — alsof je er net hebt gegeten.
- Verzin NOOIT feiten. Baseer alles op de aangeleverde brondata.
- Als informatie ontbreekt, laat het veld weg. Gebruik geen "onbekend" of placeholders.

Velden die je schrijft:
- intro: 1-2 zinnen, pakkende opening die de essentie vangt. Max 200 tekens.
- seoText: 3-5 zinnen, uitgebreidere beschrijving van keuken, sfeer en ervaring.
  Gebaseerd op cuisine, Michelin-context, en website-signalen.
- tags: max 6 korte tags die het restaurant typeren (bijv. "wijnkaart", "seizoensmenu",
  "intiem", "terras"). Alleen op basis van echte signalen.
- cuisine: exact één waarde uit de toegestane lijst.

Schrijf GEEN:
- Marketingtaal of superlatieven ("het beste", "uniek", "niet te missen")
- Letterlijke Michelin-tekst
- Informatie die niet in de brondata staat

Input payload (user message)


```
{
  "seed": {
    "name": "Restaurant X",
    "city": "Amsterdam",
    "michelin_cuisine": "Modern French",
    "michelin_price": "€€€€",
    "michelin_distinction": "1 ster",
    "michelin_summary": "..."
  },
  "website_signals": {
    "meta_description": "...",
    "headings": ["..."],
```

```
    "body_keywords": ["seasonal", "wine pairing", "tasting menu"]
  }
}
```

Output schema

```
{
  "intro": "string",
  "seoText": "string",
  "cuisine": "string (uit toegestane lijst)",
  "tags": ["string", "max 6"]
}
```

Toegestane cuisine waarden

Italiaans, Frans, Japans, Chinees, Thais, Indiaas, Mexicaans, Grieks, Spaans, Amerikaans, Koreaans, Vietnamees, Indonesisch, Turks, Marokkaans, Mediterraans, Midden-Oosters, Fusion, Modern, Klassiek, Steakhouse, Visrestaurant, Sushi, Pizzeria, Bistro, Brasserie, Tapas, Grill, BBQ, Vegetarisch, Veganistisch, Biologisch, Farm-to-table, Overig

---

Stap 5 — Assemblage & Database Insert

Veld-mapping naar ons schema

Ons schema-veld	Bron
name	Michelin seed
slug	Auto-gegenereerd (slugify naam)
city	Michelin seed
intro	OpenAI output
rating	Default 40 (alle Michelin restaurants zijn goed). Niet afleiden van sterren.
price	Michelin price hint (€€, €€€, €€€€)
cuisine	OpenAI output
tags	OpenAI output
imageUrl	R2 screenshot URL
website	Michelin seed / website scrape
reserveUrl	Website scrape
phone	Website scrape
openingHours	Website scrape (JSON string)
addressStreet	Website scrape
addressZipcode	Website scrape
addressCity	Website scrape
addressMaps	Gegenereerd of gescraped
province	Handmatig in seed-list, of null. Niet automatisch afleiden.
seoText	OpenAI output
isOffline	true (draft, handmatige review voor publicatie)
language	"nl"
autoSyncEnglish	true

Insert methode

Gebruik de bestaande POST /api/restaurants route (die ook auto-translate triggert).

Of direct DB insert via een script, met handmatige trigger van translate.

JSON schema validatie

Voordat een restaurant in de DB gaat, valideren we het complete JSON object
tegen het DB-schema. Ontbrekende verplichte velden → fout, niet invullen.

Audit trail

Per restaurant bewaren we drie bestanden:
- raw_html_{slug}.html — gecachte website HTML (voor re-parsing zonder re-crawl)
- raw_sources_{slug}.json — alle geëxtraheerde brondata
- final_{slug}.json — de gegenereerde output na OpenAI + validatie

---
Stap 6 — Review & Publicatie

Na de automatische pipeline:

1. Handmatige review van alle 10 restaurants op:
  - Feitelijke juistheid (adres, telefoon, uren)
  - Schrijfstijl (klinkt het als PB?)
  - Uniekheid (geen gekopieerde Michelin-tekst)
  - Image kwaliteit (screenshot bruikbaar?)
2. Goedkeuring per restaurant:
  - isOffline: false zetten via CMS
  - Eventueel tekst bijschaven
3. Evaluatie:
  - Hoeveel van de 10 zijn direct publiceerbaar?
  - Welke stappen hadden de meeste fouten?
  - Is de pipeline klaar om op te schalen naar 50+?

Nieuw te bouwen bestanden

Bestand	Doel
scripts/restaurant-pipeline.ts	Hoofdscript: orchestreert de hele pipeline
scripts/lib/scrape-website.ts	Website scraper: telefoon, uren, adres, signals
scripts/lib/screenshot.ts	Screenshot ophalen + R2 upload
scripts/lib/generate-copy.ts	OpenAI aanroep voor intro/seoText/tags/cuisine
scripts/lib/validate.ts	Normalisatie + validatie naar DB-schema
scripts/data/seed-restaurants.json	10 Michelin restaurants als input
scripts/data/output/	Map voor raw_html + raw_sources + final JSON per restaurant

Bestaande bestanden die we aanpassen

Geen. De pipeline draait als standalone script en gebruikt bestaande API routes
of directe DB operaties. Geen wijzigingen aan de productie-app nodig.

Technische details

Dependencies (al aanwezig)

- openai v6.7.0 — GPT-4o client
- sharp — image compressie
- @neondatabase/serverless — DB
- drizzle-orm — ORM
- @aws-sdk/client-s3 — R2 upload

Geen nieuwe dependencies nodig

- Screenshot via HTTP fetch (thum.io URL)
- Website scraping via bestaande fetch

Environment variables (al geconfigureerd)
- OPENAI_API_KEY — GPT-4o

- DATABASE_URL — Neon PostgreSQL
- R2_ACCOUNT_ID, R2_ACCESS_KEY_ID, R2_SECRET_ACCESS_KEY — Cloudflare R2
- R2_PUBLIC_URL — CDN URL voor images

Pilot uitvoering

Stap-voor-stap

```
# 1. Vul seed-restaurants.json met 10 restaurants van Michelin
#    (handmatig, jij kiest ze)

# 2. Draai de pipeline
npx tsx scripts/restaurant-pipeline.ts

# 3. Review output in scripts/data/output/
#    Per restaurant: raw_sources + final JSON

# 4. Bij goedkeuring: insert in DB
npx tsx scripts/restaurant-pipeline.ts --insert

# 5. Review in CMS, zet online
```

Geschatte kosten per restaurant

- OpenAI: ~$0.01 (gpt-4o, ~500 tokens in + 300 out)
- Screenshot: gratis (thum.io) of ~$0.01 (microlink)
- Totaal pilot (10 restaurants): < $0.20

Geschatte tijd

- Pipeline bouwen: 1 sessie
- Seed-list vullen: 15 min (jij)
- Pipeline draaien: ~2 min
- Review: 15 min (jij)

Na de pilot — Opschalen

Als de pilot slaagt (≥8 van 10 direct publiceerbaar):

1. Michelin scraper bouwen voor automatische seed-list (of Google Places API)
2. Batch mode — 50 restaurants per run
3. CMS integratie — "Auto-import" knop in het CMS
4. Scheduling — wekelijkse check op nieuwe Michelin-vermeldingen
5. Image verbetering — Puppeteer voor betere screenshots, of AI-generated images als fallback

---
Samenvatting

Onderdeel	Aanpak	Risico
Seed-list	Handmatig (pilot) → Michelin scraper (later)	Laag
Website scraping	WebFetch + schema.org parsing	Medium (JS-heavy sites)
Screenshots	thum.io (gratis, simpel)	Laag
AI teksten	GPT-4o, temp 0.5, JSON schema	Laag (review stap)
Images opslag	Bestaande R2 pipeline	Laag
DB insert	Bestaande schema + auto-translate	Laag
Juridisch (images)	Screenshots = editorial fair use	Laag
Juridisch (teksten)	Eigen gegenereerd, niet gekopieerd	Laag

Kernprincipe: Feiten en copy strikt gescheiden. De scraper levert feiten, OpenAI schrijft alleen tekst op basis van die feiten. Niet-verifieerbare velden worden weggelaten, niet gevuld.

```
---
Verbeterde pipeline flow (na ChatGPT review)

seed-list (10 restaurants)
    ↓
fetch website HTML
    ↓
save raw_html_{slug}.html          ← cache voor re-parsing
    ↓
extract facts
  ■■ schema.org / JSON-LD          ← primaire bron
  ■■ tel: links                    ← phone fallback
  ■■ meta tags                     ← signals voor AI
  ■■ openingHours ALLEEN schema.org
    ↓
dedup check (naam + website domain)
    ↓
screenshot (thum.io, retry 2×, timeout 10s)
  ■■ success → upload R2
  ■■ fail → try OG image
  ■■ fail → null
    ↓
OpenAI generatie (intro, seoText, tags, cuisine)
    ↓
schema validatie (verplichte velden check)
    ↓
save raw_sources + final JSON      ← audit trail
    ↓
[--insert flag] → DB insert + auto-translate EN
```