StarpathVision – Integraties Uitbreiding

Doel

Platform uitbreiden met extra functionaliteiten via externe integraties.

---

1. Kalenderkoppeling

- **Gebruik:** Herinneringen voor rituelen, readings.
- **Integraties:** Google Calendar, Outlook, iCal.
- **Techniek:** OAuth 2.0 + Calendar API events pushen.

---

2. CRM-koppeling

- **Gebruik:** Gebruikerssegmentatie, e-mailcampagnes.
- **Integraties:** Customer.io, HubSpot.
- **Techniek:** Webhooks + REST API sync.

---

3. E-commerce

- **Gebruik:** Verkoop fysieke producten (tarotdecks, boeken).
- **Integratie:** Shopify of WooCommerce embed.

---

4. Helpcenter & Chat

- **Gebruik:** Zelfservice + snelle vragen.
- **Integratie:** Intercom of Zendesk.
- **Functie:** AI-FAQ + live chat (optioneel).

---

5. Meetpunten (Analytics)

- Event participation rate (kalender).
- CRM campagne open/click rates.
- E-commerce conversie.

Volgende stap: Test & QA uitbreidingen

StarpathVision – Marketing & Contentstrategie

Doel: Merkbekendheid opbouwen, gebruikers aantrekken en behouden via consistente en inspirerende content, meertalige kanalen en datagedreven optimalisatie.

---

1) Merkpositionering

• **Identiteit**: Mystiek, toegankelijk, betrouwbaar.
• **Tone-of-Voice**: Empathisch, inspirerend, helder. Nooit belerend of angstaanjagend.
• **USP's**:
• Alles-in-één platform voor AI-gebaseerde readings.
• Kruisbestuiving van methodes voor diepere inzichten.
• Privacy & AVG-conform.

---

2) Doelgroepen & Persona's

• **Curieuze ontdekkers**: nieuw in spiritualiteit.
• **Ervaren gebruikers**: gewend aan tarot/numerologie.
• **Lifestyle-influencers**: delen inzichten online.
• **Internationale markt**: NL, EN, TR.

---

3) Kanalen & tactieken

3.1 Website & SEO

• Blogsectie: wekelijkse artikelen ("Betekenis van Tarotkaart X", "Numerologie in het dagelijks leven").
• Landingspagina's per module (tarot, koffiedik, numerologie) met zoekwoorden.
• Technische SEO: schema.org markup (FAQ, Article, Product).
• Snelheid & mobile-first optimalisatie (Core Web Vitals).

3.2 E-mailmarketing

• **Flows**:
• Welkomstserie (3 mails) → onboarding tips, eerste gratis reading.
• Retentie: herinnering aan nieuwe sessies, verjaardagsreading.
• Upsell: Premium/Pro voordelen na X gratis sessies.
• **Templates**: responsive, meertalig, AI-gegenereerde gepersonaliseerde intro.

3.3 Social media

• **Kanalen**: Instagram, TikTok, YouTube Shorts, Pinterest.
• **Contentvormen**:
• Kaart-van-de-dag video's.
• Behind-the-scenes AI reading demo's.

- Uitlegseries ("Hoe werkt koffiedik lezen?").
- User generated content (met toestemming).
- **Frequentie**: min. 3 posts/week per kanaal.

3.4 Partnerships & PR

- Samenwerkingen met spirituele coaches.
- Podcast/interview-series.
- Gastartikelen op lifestyleblogs.

3.5 Paid ads

- Testen met Meta Ads en Google Ads op kernzoekwoorden.
- Retargeting via Meta Pixel.

---

4) Contentstrategie

- **Kernformats**:
- Educatief (uitleg, tips)
- Inspirerend (quotes, verhalen)
- Interactief (polls, quizzen)
- **Meertaligheid**: NL, EN, TR; AI-pretranslated + menselijke review.
- **Tone-guide integratie**: AI-output volgt merk-TOV.
- **Visuele stijl**: consistente kleuren, lettertypes, kosmische elementen.

---

5) Growth & CRM

- **Lead magnets**: gratis PDF reading bij inschrijving.
- **Referral programma**: punten/extra readings voor uitnodigen vrienden.
- **Segmentatie**: activiteit, interesse (werk, liefde, algemeen).
- **Churn preventie**: triggers bij dalende activiteit → gepersonaliseerde incentives.

---

6) KPI’s & Metingen

- Website: organisch verkeer, conversieratio naar registratie.
- E-mail: open rate, CTR, upgrade naar betaald.
- Social: bereik, engagement rate, shares.
- Paid ads: CPA, ROAS.
- Retentie: % actieve gebruikers na 30/90 dagen.

Tools: Google Analytics 4, Hotjar, Meta/Google Ads Manager, MailerLite/Klaviyo.

---

7) Campagnekalender (voorbeeld)

- **Q1**: Lancering (gratis reading campagne).
- **Q2**: Thema "Liefde & relaties" (Valentijn, lente).

- **Q3**: Thema "Zelfontwikkeling" (back to school).
- **Q4**: Thema "Reflectie & doelen" (eindejaarsreadings).

---

8) Volgende stappen

- Keyword research per module.
- E-mailtemplate set ontwerpen in lijn met UI kit.
- Social contentplanning opzetten (3 maanden vooruit).
- Testbudget instellen voor eerste paid ads (A/B varianten).

StarpathVision – Master To-Do (Alles-in-één Checklist)

> Eén overzicht met **alle** resterende taken om StarpathVision van nu → live → schaalbaar te brengen. Geordend per domein, met prioriteit (P0/P1/P2), afhankelijkheden en acceptatiecriteria.

Legenda: **P0** = must-have vóór launch · **P1** = kort na launch · **P2** = nice-to-have / latere fase.

---

1) Product & UX

- [ ] **P0** MVP-flows finaliseren (Home → Ontdekkingsreis → Eerste reading → PDF)
- Acceptatie: binnen 2 min tot eerste PDF; mobile ok; A11y labels aanwezig.
- [ ] **P0** Dashboard widgets (Energie van de dag, Snel starten, Historiek)
- Acceptatie: 3 laatste sessies zichtbaar; "Start" knoppen werken.
- [ ] **P0** Tarot flow (3-kaarten) met flip, AI-tekst streaming, export
- Acceptatie: AI stream < 3s, PDF met watermerk, deelbare visual.
- [ ] **P0** Numerologie basis (Life Path, Destiny, Soul Urge)
- Acceptatie: deterministisch resultaat voor testcases.
- [ ] **P1** Koffiedik v1 (upload + symbolen + confidence + tips)
- Acceptatie: bij lage kwaliteit fototips i.p.v. fout.
- [ ] **P1** Historiek & detailview per sessie (delen/opslaan)
- [ ] **P1** Pricing/Paywall schermen (Free/Premium/Pro)
- [ ] **P1** Gamification basis (punten, badges, levels + beloningen)
- [ ] **P2** Astrologie v1 (geboortehoroscoop + transits light)
- [ ] **P2** Community (anoniem delen + moderatie)

---

2) Frontend (Next.js + UI Kit)

- [ ] **P0** UI Kit integreren in `packages/ui` + Storybook stories
- [ ] **P0** App Router routes (marketing/app) + i18n (NL/EN, TR later)
- [ ] **P0** State & data fetching (React Query of server actions) afronden
- [ ] **P0** OG-image routes (daily card, shareable quote)
- [ ] **P1** PWA (manifest, service worker, install banner, webpush opt-in)
- [ ] **P1** A11y-audit (axe) fixes (contrast, focus states, aria-labels)
- [ ] **P1** Skeleton loaders & error boundaries
- [ ] **P2** Dark/Light toggle + theming tokens

---

3) Backend & API

- [ ] **P0** Prisma schema migreren + seed scripts draaien
- [ ] **P0** `/api/readings/{module}` endpoints met JSON-schema validatie (AJV)
- [ ] **P0** Prompt Registry endpoints (`/prompts`, flags) + caching (Redis)

- [ ] **P0** PDF export service (React-PDF of Puppeteer)
- [ ] **P0** Auth.js (email+password), e-mail verificatie, forgot password
- [ ] **P1** File uploads → S3/R2 (presigned) + virus scan + EXIF strip
- [ ] **P1** Admin API (metrics, prompts, campaigns, features, users)
- [ ] **P1** Rate limiting & request logging
- [ ] **P2** OpenAPI public spec + Swagger UI voor partners

---

4) AI-laag (Prompts, Models, Validatie)

- [ ] **P0** Prompt templates activeren (Tarot/Numerologie) + Tone Guide injectie
- [ ] **P0** JSON-schema validators (tarot.v1, numerology.v1) pre-compile
- [ ] **P0** Repair-prompt fallback bij schemafout (1 retry)
- [ ] **P1** Koffiedik vision v1 (klassiek + labels; later ML-weights)
- [ ] **P1** Combinatie-motor v1 (tarot + numerology + dayEnergy)
- [ ] **P1** Golden outputs (≥50 cases/module) + nightly eval job
- [ ] **P2** Model fallback (primair/secundair) + telemetrie per template

---

5) Automations & Workers

- [ ] **P0** Daily Card job (BullMQ) + OG-image + e-mail snippet
- [ ] **P0** Retentie checker (7/14/30 dagen) + nudge mail
- [ ] **P1** Love Week campaign jobs (prepare/send/social)
- [ ] **P1** Year Reflection aggregation + PDF + mailing
- [ ] **P1** Admin test endpoints voor handmatige trigger
- [ ] **P2** Social autopost integraties (Buffer/Meta API)

---

6) Billing & Facturatie

- [ ] **P0** Checkout (Stripe) Free→Premium/Pro + iDEAL/SEPA/Cards
- [ ] **P0** Webhooks (signature validate) + Subscription statusmachine
- [ ] **P1** Customer Portal integratie (upgrade/downgrade/coupons)
- [ ] **P1** Proration & trials + e-mailnotificaties (paid/failed)
- [ ] **P1** BTW/OSS configuratie + factuur PDF velden
- [ ] **P2** Refund flow in admin + beleidsteksten in T&C

---

7) Security, Privacy & Compliance (AVG)

- [ ] **P0** DPIA basis invullen en reviewen
- [ ] **P0** RBAC (user/admin/support) + MFA voor admin
- [ ] **P0** Encryptie at-rest (DB/S3) + TLS en security headers (CSP/HSTS)
- [ ] **P0** Consent management (analytics/ai-train/marketing)
- [ ] **P1** Data export/delete self-service (AVG-rechten)
- [ ] **P1** Back-ups + restore oefening (tabletop)
- [ ] **P1** Pen-test/DAST scan + fixes
- [ ] **P2** Kolom-encryptie (geboortedatum) + KMS

8) Observability & Ops

- [ ] **P0** Sentry (errors) + structured JSON logging
- [ ] **P0** Health checks + uptime monitor (Checkly)
- [ ] **P0** Tracing (OpenTelemetry) voor kritieke flows
- [ ] **P1** Dashboards: API latency, queue backlog, schema-fouten
- [ ] **P1** Alerts: DLQ>0, webhook failures, AI provider errors
- [ ] **P2** Runbooks (rollback, cache flush, queue drain, provider switch)

---

9) Testen & Kwaliteit

- [ ] **P0** Unit tests (parsers, validators, services) ≥ 80% coverage
- [ ] **P0** Integratietests (readings API ↔ Prisma ↔ validators)
- [ ] **P0** E2E (Playwright): kernflows + visuele regressie per route
- [ ] **P1** Performance tests (k6) + CWV (Lighthouse) budgets
- [ ] **P1** Accessibility tests (axe) en fixes
- [ ] **P2** Chaos tests (AI timeouts, webhook retries, Redis uitval)

---

10) Content & Marketing

- [ ] **P0** Content & Tone Guide finaliseren + koppelen aan prompts
- [ ] **P0** Website SEO-basics (titles, meta, sitemap, schema.org)
- [ ] **P1** Blogplanning (12 artikelen) + social kalender (1 maand)
- [ ] **P1** E-mailjourneys (welkom, retentie, upsell) live
- [ ] **P1** Campagne: Kaart van de Dag automatiseren
- [ ] **P2** Partnerships/PR outreach lijst en pitches

---

11) Legal & Policies

- [ ] **P0** Terms of Service + Privacy Policy + Cookies
- [ ] **P0** DPA’s met subverwerkers (hosting, AI, e-mail, betalingen)
- [ ] **P1** Cookie consent (CMP) met granular toggles
- [ ] **P2** Notice & Takedown + Community Guidelines (bij community)

---

12) Launchvoorbereiding

- [ ] **P0** Staging smoke tests + handmatige QA checklist
- [ ] **P0** Feature flag set voor MVP (astrologie/community uit)
- [ ] **P0** Incident-kanalen klaar (Slack/Email/On-call)
- [ ] **P1** Beta cohort uitnodigen (50–100 users)
- [ ] **P1** Feedbackformulier + NPS in app

---

13) Post-launch

- [ ] **P1** Retrospective + bug-burn-down week 1
- [ ] **P1** A/B test pricing/CTA varianten
- [ ] **P1** Roadmap v1.1: Koffiedik v1, combinatie-motor, PWA
- [ ] **P2** Roadmap v1.2: Astrologie v1, community, live AI chat

---

14) Dependencies & volgorde (samengevat)

1) **P0-blokken**: Auth, Readings (Tarot+Numerologie), PDF, Billing + Webhooks, Observability, DPIA, SEO-basics.
2) **P1**: Koffiedik v1, Combinatie-motor, Admin, Retentie & Campaign jobs, PWA & A11y, e-mailjourneys.
3) **P2**: Astrologie, Community, Social autopost, uitgebreide theming.

---

15) Checklists per omgeving

- **Dev**: docker compose up; seeds ok; E2E smoke groen.
- **Staging**: Stripe test-keys; OG-routes; daily jobs in dry-run.
- **Prod**: real keys; webhooks verified; DLQ=0; alerts actief.

Laat weten als je prioriteiten of eigenaars (RACI) per taak wilt; dan vul ik de tabel aan met **Owner**, **Reviewer** en **Dependency links** naar de juiste canvassen.

StarpathVision – Overdrachtsdocument (voor Bouwer)

1. Index van Canvassen

1. **Complete Build Spec** – Functioneel ontwerp, technische architectuur, roadmap.
2. **UI Kit & Mockups** – Componenten, stijlrichtlijnen, animaties.
3. **AI Promptbibliotheek & JSON-schema's** – Templates en strikte outputformaten.
4. **API & Datamodel Referentie** – OpenAPI, Prisma modellen, endpoints.
5. **Seed-data & Contentvoorbeelden** – Tarotdecks, symbolen, numerologie tabellen.
6. **Marketing & Contentstrategie** – Merkpositionering, social media formats.
7. **Testplan & Kwaliteitscontrole** – Testscenario's, AI-output validatie.
8. **High-Fidelity Prototypes (Figma)** – Pixelperfect designs.
9. **Operations & Deployment Handleiding** – Hosting, CI/CD, security.
10. **Contentbibliotheek Orakel & Wijzen** – Profielen, tone-of-voice.
11. **Data-archiefmodule Specificatie** – Klantgeschiedenis en AI-herinneringen.
12. **Integratiehandleiding Externe API's** – Koppelingen met vision en payment.

---

2. Technische Quickstart

- **Frontend:** Next.js + TailwindCSS + Framer Motion.
- **Backend:** Node.js + Express + Prisma.
- **Database:** PostgreSQL.
- **Auth:** Firebase Authentication (email + social login).
- **Storage:** Firebase Storage + S3 compatibel.
- **Payments:** Stripe.
- **AI:** OpenAI GPT-4 + Vision API.
- **Hosting:** Vercel (frontend) + Railway/Render (backend) of eigen VPS.

---

3. Buildvolgorde (Sprints)

1. Core backend + auth + database schema.
2. UI implementatie + mockdata.
3. AI prompt integratie per module.
4. Payment & abonnementslogica.
5. Testen, bugfixes, en performance optimalisatie.
6. Marketing en livegang.

---

4. Acceptatiecriteria & Deliverables

- Alle modules functioneel.
- AI-outputs voldoen aan tone-of-voice en JSON-format.
- 100% responsive design.
- AVG/GDPR compliant.

---

5. Mappenstructuur voor ZIP-overdracht

```
StarpathVision/
  01_BuildSpec/
  02_UI_Kit/
  03_AI_Prompts_JSON/
  04_API_Datamodel/
  05_Seed_Data/
  06_Marketing/
  07_Testplan/
  08_Figma_Designs/
  09_Deployment_Manual/
  10_Orakel_Profiles/
  11_Data_Archive_Module/
  12_API_Integrations/
  OVERDRACHTSDOCUMENT.pdf
```

---

6. Rollen & Communicatie

- **Product Owner:** Jij.
- **Lead Developer:** Hoofdverantwoordelijk voor tech.
- **Designer:** UI/UX + Figma.
- **QA Tester:** Test alle functies.

---

7. Juridisch & Security

- NDA ondertekenen voor overdracht.
- Versleuteling voor gevoelige data.
- Duidelijke privacy policy.

Opmerking: Dit document samen met de mappenstructuur vormt een compleet overdrachts- en bouwpakket dat een ontwikkelaar direct kan gebruiken om StarpathVision te realiseren.

StarpathVision – Overdrachtsmap & Voorbeeldrepo

> Alles wat een bouwer nodig heeft om direct te starten: mappenstructuur, kernbestanden, voorbeeldcode, seeds, API-spec, en sprintbord (CSV) om te importeren in Trello/Jira.

---

0) 00_README.md (kopie-klaar)

```
# StarpathVision – Overdrachtsmap

## Wat is dit?
Een klik-en-start voorbeeldrepo + alle specificaties om MVP (P0) te leveren.

## Snelstart
1. Vereisten: Node 20, pnpm 9, Docker, Git
2. `pnpm i`
3. `docker compose up -d`
4. `.env.example` kopiëren naar `.env` en invullen
5. `pnpm --filter apps/web prisma migrate dev`
6. `pnpm seed`
7. `pnpm dev` (start web + worker)

## Documentatie
- Build Spec, API/Prisma, Prompts/Schema's, Security/AVG, Testplan,
Marketing: zie `/docs` (uit canvassen geëxporteerd)
```

---

1) Mappenstructuur (zip-klaar)

```
starpathvision/
  README.md
  pnpm-workspace.yaml
  package.json
  turbo.json
  docker-compose.yml
  .env.example
  docs/                         # geëxporteerde canvassen (PDF/MD)
  apps/
    web/
      app/(marketing)/page.tsx
      app/(app)/dashboard/page.tsx
      app/(app)/readings/tarot/page.tsx
      app/api/readings/[module]/route.ts
      app/api/prompts/route.ts
```

```
      app/api/automations/run/daily-card/route.ts
      app/og/daily-card/route.ts
      lib/ai.ts
      lib/schema.ts
      lib/db.ts
      lib/auth.ts
      prisma/schema.prisma
      next.config.ts
      tailwind.config.ts
      postcss.config.js
    worker/
      src/queues/automation.ts
      src/jobs/dailyCard.ts
      src/index.ts
  packages/
    ui/
      src/index.ts
      package.json
    prompts/
      src/client.ts
      seed/tarot.nl.v1.json
      package.json
    schemas/
      src/tarot.v1.json
      src/numerology.v1.json
      src/index.ts
      package.json
    emails/
      src/DailyCardEmail.tsx
      package.json
    config/
      tailwind-preset.ts
      tsconfig.base.json
      eslint.config.js
  tooling/
    postman/starpathvision.postman_collection.json
    openapi/openapi.yaml
  seeds/
    tarot.decks.json
    spreads.json
    numerology.tables.json
    badges.json
    demo.users.ndjson
    demo.sessions.ndjson
```

---

2) Kernbestanden (inhoud)

2.1 `pnpm-workspace.yaml`

```
packages:
  - "apps/*"
  - "packages/*"
  - "tooling/*"
```

2.2 root `package.json`

```
{
  "name": "starpathvision",
  "private": true,
  "packageManager": "pnpm@9",
  "scripts": {
    "dev": "turbo run dev --parallel",
    "build": "turbo run build",
    "lint": "turbo run lint",
    "test": "turbo run test",
    "seed": "node scripts/seed.js"
  },
  "devDependencies": { "turbo": "^2.0.0" }
}
```

2.3 `docker-compose.yml`

```
version: "3.9"
services:
  db:
    image: postgres:16
    environment:
      POSTGRES_USER: spv
      POSTGRES_PASSWORD: spv
      POSTGRES_DB: spv
    ports: ["5432:5432"]
  redis:
    image: redis:7
    ports: ["6379:6379"]
  minio:
    image: minio/minio
    command: server /data --console-address ":9001"
    environment:
      MINIO_ROOT_USER: minio
      MINIO_ROOT_PASSWORD: miniosecret
    volumes:
      - minio:/data
    ports: ["9000:9000", "9001:9001"]
```

```
volumes:
  minio: {}
```

2.4 `.env.example`

```
DATABASE_URL=postgres://spv:spv@localhost:5432/spv
REDIS_URL=redis://localhost:6379
S3_ENDPOINT=http://localhost:9000
S3_BUCKET=spv
S3_ACCESS_KEY=minio
S3_SECRET_KEY=miniosecret
AI_PROVIDER=openai
OPENAI_API_KEY=
STRIPE_KEY=
PROMPTS_ENDPOINT=http://localhost:3000/api/prompts
CRON_TZ=Europe/Amsterdam
JWT_SECRET=change-me
```

2.5 `apps/web/app/api/readings/[module]/route.ts`

```
import { NextRequest, NextResponse } from 'next/server'
import { getTemplate } from '@/lib/prompts'
import { aiTarot } from '@/lib/ai'
import { validateTarot } from '@/lib/schema'

export async function POST(req: NextRequest, { params }: { params: { module:
string } }) {
  const body = await req.json()
  const locale = body?.locale ?? 'nl-NL'
  if (params.module !== 'tarot') return NextResponse.json({ error: { code:
'UNSUPPORTED' } }, { status: 400 })
  const tmpl = await getTemplate('tarot', locale)
  const aiOut = await aiTarot({ template: tmpl, input: body })
  const valid = validateTarot(aiOut)
  if (!valid.ok) return NextResponse.json({ error: { code:
'422_SCHEMA_MISMATCH', details: valid.errors } }, { status: 422 })
  return NextResponse.json({ result: aiOut }, { status: 200 })
}
```

2.6 `apps/worker/src/index.ts`

```
import { Queue, Worker, QueueScheduler } from 'bullmq'
import { dailyCardJob } from './jobs/dailyCard'
const connection = { connection: { url: process.env.REDIS_URL! } }
export const automationQ = new Queue('automation:daily', connection)
new QueueScheduler('automation:daily', connection)
automationQ.add('daily_card.generate', {}, { repeat: { tz: 'Europe/
Amsterdam', cron: '5 0 * * *' } })
```

```
new Worker('automation:daily', async (job) => {
  if (job.name === 'daily_card.generate') return dailyCardJob(job)
}, connection)
```

2.7 `tooling/openapi/openapi.yaml` (excerpt)

```
openapi: 3.0.3
info: { title: StarpathVision API, version: 1.0.0 }
paths:
  /readings/tarot:
    post:
      summary: Create tarot reading
      responses: { '200': { description: OK }, '422': { description: Schema
invalid } }
```

2.8 `tooling/postman/starpathvision.postman_collection.json` (excerpt)

```
{
  "info": { "name": "StarpathVision", "schema": "https://
schema.getpostman.com/json/collection/v2.1.0/collection.json" },
  "item": [
    { "name": "Create Tarot Reading", "request": { "method": "POST", "url":
"http://localhost:3000/api/readings/tarot", "body": { "mode": "raw", "raw":
"{\n  \"deckId\": \"RWS\",\n  \"spreadId\": \"3C\",\n  \"locale\": \"nl-
NL\"\n}" } } }
  ]
}
```

---

3) Seeds (samenvatting)

- `seeds/tarot.decks.json`, `seeds/spreads.json`, `seeds/numerology.tables.json`, `seeds/badges.json`
- `seeds/demo.users.ndjson`, `seeds/demo.sessions.ndjson`

Seed script: `scripts/seed.js` (laadt JSON/NDJSON in Postgres/S3; idempotent)

---

4) Sprintbord – import (CSV/Jira/Trello)

Bestand: `/tooling/boards/sprint1_p0.csv`

```
Title,Description,Priority,Labels
Repo opzetten,"Monorepo + Docker + .env.example",P0,setup
DB + Prisma migrate,"Init schema + seeds draaien",P0,backend
Auth flow,"Register/Login/Reset + e-mail verify",P0,auth
```

```
Tarot API endpoint,"/api/readings/tarot + AJV schema",P0,ai,api
Tarot UI flow,"Deck/spread/flip + stream + PDF",P0,frontend
Stripe checkout,"Free→Premium/Pro + webhooks",P0,billing
Daily Card automation,"BullMQ job + OG + e-mail snippet",P0,automations
Observability basics,"Sentry + health checks + logs",P0,ops
Security baseline,"MFA admin, headers, rate limiting",P0,security
SEO basics,"Sitemap, titles/descriptions, OG",P0,seo
```

---

5) Export docs (uit canvassen)

Plaats alle huidige canvassen als **PDF + MD** in `/docs/` met dezelfde titels:

- Build Spec, API & Datamodel, Promptbibliotheek, UI Kit, Security & Privacy, Testplan, Marketing, Campagneplan, Automations, Master To-Do, Repo Skeleton, Admin Paneel, Billing.

---

6) Overdracht-checklist

-

StarpathVision – Repo Integratie & Code Skeleton

> Doel: Kopie-klaar skelet voor monorepo + installatie-instructies + voorbeeldcode om alle bestaande canvassen (prompts, schema's, automations, UI kit) te integreren.

---

1) Monorepo structuur (pnpm workspaces)

```
starpathvision/
  package.json            # pnpm workspaces + scripts
  pnpm-workspace.yaml
  turbo.json              # (optioneel) Turborepo pipelines
  .github/workflows/
  apps/
    web/                  # Next.js (App Router)
    worker/               # BullMQ workers + cron
  packages/
    ui/                   # UI kit (React + Tailwind + shadcn)
    prompts/              # Prompt registry client + types
    schemas/              # JSON schema's (AJV) + Zod types
    emails/               # React Email/MJML templates
    config/               # ESLint, TSconfig, Tailwind preset
```

1.1 `pnpm-workspace.yaml`

```
packages:
  - "apps/*"
  - "packages/*"
```

1.2 Root `package.json` (scripts)

```
{
  "private": true,
  "packageManager": "pnpm@9",
  "scripts": {
    "dev": "turbo run dev",
    "build": "turbo run build",
    "lint": "turbo run lint",
    "test": "turbo run test",
    "seed": "pnpm --filter @spv/schemas run seed && pnpm --filter @spv/
prompts run seed"
  },
  "devDependencies": {
    "turbo": "^2.0.0"
```

```
  }
}
```

---

2) Apps/web (Next.js)

```
apps/web/
  app/
    (marketing)/
      page.tsx           # Home
      pricing/page.tsx
    (app)/
      dashboard/page.tsx
      readings/
        tarot/page.tsx
        coffee/page.tsx
      api/
        readings/
          [module]/route.ts   # POST create reading
        prompts/route.ts      # GET prompt templates
        automations/run/
          daily-card/route.ts # admin test trigger
    og/
      daily-card/route.ts     # OG-image
  lib/
    ai.ts                # provider, call wrappers
    schema.ts            # AJV validators
    db.ts                # Prisma client
    auth.ts              # Auth.js
  prisma/
    schema.prisma
  tailwind.config.ts
  next.config.ts
```

2.1 Voorbeeld API route – schema-gevalideerde reading

```
// apps/web/app/api/readings/[module]/route.ts
import { NextRequest, NextResponse } from 'next/server'
import { getTemplate } from '@/lib/prompts'
import { aiTarot } from '@/lib/ai'
import { validateTarot } from '@/lib/schema'

export async function POST(req: NextRequest, { params }: { params: { module:
string } }) {
  const body = await req.json()
  const locale = body?.locale ?? 'nl-NL'
  if (params.module !== 'tarot') return NextResponse.json({ error: { code:
'UNSUPPORTED' } }, { status: 400 })
```

```
  const tmpl = await getTemplate('tarot', locale)
  const aiOut = await aiTarot({ template: tmpl, input: body })
  const valid = validateTarot(aiOut)
  if (!valid.ok) return NextResponse.json({ error: { code:
'422_SCHEMA_MISMATCH', details: valid.errors } }, { status: 422 })
  return NextResponse.json({ result: aiOut }, { status: 200 })
}
```

---

3) Apps/worker (BullMQ + cron)

```
apps/worker/
  src/
    queues/
      automation.ts      # define queues
    jobs/
      dailyCard.ts       # generate daily card
      dailyEnergy.ts
      loveWeek.ts
      yearReflection.ts
    index.ts             # boot worker + repeatable jobs
  env.d.ts
```

3.1 Worker bootstrap

```
// apps/worker/src/index.ts
import { Queue, Worker, QueueScheduler } from 'bullmq'
import { dailyCardJob } from './jobs/dailyCard'

const connection = { connection: { url: process.env.REDIS_URL! } }
export const automationQ = new Queue('automation:daily', connection)
new QueueScheduler('automation:daily', connection)

// Repeatable (Europe/Amsterdam)
automationQ.add('daily_card.generate', {}, { repeat: { tz: 'Europe/
Amsterdam', cron: '5 0 * * *' } })

new Worker('automation:daily', async (job) => {
  if (job.name === 'daily_card.generate') return dailyCardJob(job)
}, connection)
```

---

4) Packages/schemas (AJV + Zod)

```
packages/schemas/
  src/
```

```
    tarot.v1.json
    coffee.v1.json
    numerology.v1.json
    index.ts            # export validators
  package.json         # name: @spv/schemas
```

4.1 Validator voorbeeld

```
// packages/schemas/src/index.ts
import Ajv from 'ajv'
import tarot from './tarot.v1.json'
const ajv = new Ajv({ allErrors: true, removeAdditional: 'all' })
const tarotValidate = ajv.compile(tarot)
export function validateTarot(data: unknown) {
  const ok = tarotValidate(data)
  return ok ? { ok: true as const } : { ok: false as const, errors:
tarotValidate.errors }
}
```

---

5) Packages/prompts (registry client)

```
packages/prompts/
  src/
    client.ts           # fetch active templates
    types.ts
  seed/
    tarot.nl.v1.json
  package.json         # name: @spv/prompts
```

5.1 Client

```
export async function getTemplate(module: string, locale: string) {
  const url = process.env.PROMPTS_ENDPOINT ?? 'http://localhost:3000/api/
prompts'
  const res = await fetch(`${url}?module=${module}&locale=${locale}`, {
cache: 'no-store' })
  if (!res.ok) throw new Error('PROMPTS_FETCH_FAILED')
  const data = await res.json()
  return data.templates?.[0]
}
```

---

6) Packages/ui (UI kit)

- Exporteer alle componenten uit `index.ts`.
- Storybook integratie in `packages/ui/.storybook`.
- Tailwind preset in `packages/config/tailwind-preset.ts` en consumeren in apps.

---

7) Packages/emails (React Email)

- `DailyCardEmail.tsx`, `RetentionNudge.tsx`, `Receipt.tsx`.
- Render in worker of web API.

---

8) Installatie & run

```
# 1) Vereisten
asdf direnv docker

# 2) Repo clonen
git clone git@github.com:you/starpathvision.git && cd starpathvision

# 3) Dependencys
pnpm i

# 4) Dev infra
docker compose up -d  # postgres, redis, minio

# 5) Env
cp .env.example .env  # vul AI/Stripe/Redis/S3

# 6) Prisma
pnpm --filter @spv/schemas build
pnpm --filter apps/web prisma migrate dev

# 7) Seed
pnpm seed

# 8) Start
pnpm dev
```

---

9) Env voorbeeld (root `.env`)

```
DATABASE_URL=postgres://user:pass@localhost:5432/spv
REDIS_URL=redis://localhost:6379
S3_ENDPOINT=http://localhost:9000
S3_BUCKET=spv
```

```
S3_ACCESS_KEY=minio
S3_SECRET_KEY=miniosecret
AI_PROVIDER=openai
OPENAI_API_KEY=sk-...
STRIPE_KEY=sk_test_...
PROMPTS_ENDPOINT=http://localhost:3000/api/prompts
CRON_TZ=Europe/Amsterdam
```

---

10) Kwaliteitschecks

- ESLint + Prettier configs delen vanuit `packages/config`.
- CI pipeline: lint → typecheck → unit → integratie → e2e → build.
- Playwright traces uploaden als artifact.

---

11) Uitrol (staging/prod)

- Vercel voor `apps/web`, Fly.io/Render voor `apps/worker`.
- Upstash Redis, Neon/Render Postgres, Cloudflare R2/S3.
- Secrets via Doppler/Vercel env vars.

Checklist: health checks groen, webhooks (Stripe) dry-run OK, DLQ leeg, OG routes bereikbaar.

StarpathVision – Repo Integratie & Structuur

> Doel: Complete integratiehandleiding voor de AI Automations (Dagelijkse kaart, campagnes, retentie) in de bestaande repo. Inclusief mappenstructuur, installaties, voorbeeldcode en koppeling met API's.

---

1) Bestandsstructuur (Monorepo)

```
apps/
  web/           # Next.js frontend/admin
  worker/        # Node workers (BullMQ)
  api/           # API-handlers (Next.js API routes / tRPC)
packages/
  prompts/       # Prompt-templates
  schemas/       # JSON-schema's (AJV)
  emails/        # React Email/MJML templates
  social/        # OG-image generator

.env             # root env vars
package.json     # werkspaces config
```

Tip: Gebruik **PNPM Workspaces** of **Yarn Workspaces** voor dependency management.

---

2) Installatie & dependencies

```
# Core
pnpm add next react react-dom

# Workers/queues
pnpm add bullmq ioredis node-cron

# AI & validatie
pnpm add openai ajv ajv-formats

# E-mail & visuals
pnpm add @react-email/components mjml satori @vercel/og

# Beveiliging
pnpm add zod bcrypt jsonwebtoken

# Dev tools
devDependencies: typescript ts-node nodemon eslint prettier
```

---

3) Integratie Workers

apps/worker/index.ts

```
import { Queue, Worker } from 'bullmq';
import { redis } from './redis';

const dailyQueue = new Queue('automation:daily', { connection: redis });

new Worker('automation:daily', async (job) => {
  if (job.name === 'daily_card.generate') {
    await runDailyCard(job.data);
  }
});
```

apps/worker/redis.ts

```
import { Redis } from 'ioredis';
export const redis = new Redis(process.env.REDIS_URL!);
```

---

4) API Routes

apps/api/automations.ts

```
import { NextApiRequest, NextApiResponse } from 'next';
import { dailyQueue } from '../worker/queues';

export default async function handler(req: NextApiRequest, res:
NextApiResponse) {
  if (req.method === 'POST') {
    await dailyQueue.add('daily_card.generate', { locale: 'nl-NL' });
    return res.status(200).json({ ok: true });
  }
  res.status(405).end();
}
```

---

5) Prompt & Schema import

- **Prompts**: in `packages/prompts/{module}/{locale}.md`
- **Schemas**: in `packages/schemas/{module}.schema.json`

Gebruik

```
import tarotPrompt from 'prompts/tarot/nl-NL.md';
import tarotSchema from 'schemas/tarot.schema.json';
```

---

6) OG-image generator

packages/social/daily-card.tsx

```
import { ImageResponse } from '@vercel/og';

export const runtime = 'edge';
export async function GET(req: Request) {
  const { searchParams } = new URL(req.url);
  const title = searchParams.get('title');
  return new ImageResponse(
    <div style={{ fontSize: 48, background: 'black', color: 'white' }}
>{title}</div>,
    { width: 1200, height: 630 }
  );
}
```

---

7) E-mail templates

packages/emails/DailyCardEmail.tsx

```
import { Html } from '@react-email/components';

export const DailyCardEmail = ({ summary, cardNames, ctaUrl }) => (
  <Html>
    <h1>🔮 Kaart van vandaag</h1>
    <p>{summary}</p>
    <p>{cardNames.join(', ')}</p>
    <a href={ctaUrl}>Bekijk meer</a>
  </Html>
);
```

---

8) ENV-variabelen

```
REDIS_URL=...
OPENAI_API_KEY=...
```

```
EMAIL_PROVIDER=resend
RESEND_API_KEY=...
```

---

9) Testen

```
# Worker starten
pnpm --filter worker dev

# API test
curl -X POST http://localhost:3000/api/automations
```

---

10) Uitrol

- **Vercel**: `apps/web` & `apps/api` builden.
- **Railway/Fly.io**: `apps/worker` deployen.
- **Upstash Redis** koppelen.
- ENV's in alle omgevingen zetten.

StarpathVision – Security & Privacy Dossier

Doel: Waarborgen van gegevensbescherming, naleving van AVG/GDPR en het opstellen van een incidentresponsplan.

---

1. Juridisch kader

• **AVG/GDPR**: Toepasbaar op alle gebruikers in de EU en vergelijkbare wetgeving wereldwijd.
• **Verwerkersovereenkomsten**: Met hostingprovider, AI-diensten, betaalproviders.
• **Subverwerkerslijst**: Opgemaakt en publiek beschikbaar op de website.

---

2. Data Protection Impact Assessment (DPIA)

• **Doel**: Identificeren van privacyrisico's en mitigerende maatregelen.
• **Scope**: Alle persoonsgegevens (naam, e-mail, geboortedata, uploads).
• **Risicoanalyse**:
• Onbevoegde toegang → versleutelde opslag (AES-256), HTTPS/TLS 1.3.
• AI-output kan gevoelige interpretaties bevatten → bewaartermijnen beperken.
• **Mitigatie**:
• Role-based access control.
• Logging en monitoring op toegang.
• Geautomatiseerde data-verwijdering.

---

3. Datastromen & bewaartermijnen

Gegevenstype	Opslaglocatie	Versleuteling	Bewaartermijn	Verwijderproces
Accountgegevens	Postgres	AES-256-at-rest	2 jaar na inactiviteit	Soft-delete + permanente verwijdering na 30 dagen
Geboortedata	Postgres	AES-256	2 jaar na inactiviteit	idem
Foto's/uploads	S3/MinIO	Server-side encryptie	90 dagen tenzij klant bewaart	Automatisch verwijderen
Betaalgegevens	Stripe	n.v.t.	conform Stripe	via Stripe

---

4. Beveiligingsmaatregelen

• **Authenticatie**: Argon2id hashing, 2FA optioneel.
• **Transport**: Alleen HTTPS, HSTS.

- **Netwerk**: WAF + rate limiting.
- **Monitoring**: Sentry + audit logs.
- **Back-ups**: Versleuteld, dagelijks, bewaard 30 dagen.

---

5. Toegangsbeheer (RBAC)

- Rollen: `user`, `admin`.
- Beperking van toegang tot productiedata voor ontwikkelaars.
- IP-allowlist voor adminpaneel.

---

6. Incidentresponsplan

1. Detectie via monitoring/alerts.
2. Classificatie (laag/midden/hoog risico).
3. Escalatie naar security officer.
4. Melding aan betrokkenen binnen 72 uur (AVG-plicht).
5. Forensisch onderzoek.
6. Verbetermaatregelen documenteren.

---

7. Gebruikersrechten (AVG)

- Recht op inzage, rectificatie, verwijdering.
- Recht op dataportabiliteit.
- Recht op beperking van verwerking.
- Zelfservice in accountinstellingen + supportkanaal.

---

8. Compliance en audits

- Jaarlijkse interne privacy-audit.
- Penetratietests door derde partijen.
- Bijwerken DPIA bij nieuwe modules.

---

9. Documentatie & training

- Interne security-handboeken.
- Jaarlijkse training voor alle teamleden.
- Awareness-campagnes rond phishing en social engineering.

Volgende stap: Integratie met **Content & Tone Guide** voor consistente privacy- en veiligheidscommunicatie naar gebruikers.

StarpathVision – Seed-data & Contentvoorbeelden

Doel: Standaard datasets en voorbeeldinhoud om direct te gebruiken bij ontwikkeling, testen en AI-trainingsvalidatie.

---

1. Tarot-decks & spreads

1.1 Tarotdecks

- **Rider-Waite-Smith**
- 78 kaarten (22 Grote Arcana, 56 Kleine Arcana)
- Afbeeldingen in hoge resolutie, rechtenvrij of gelicenseerd
- **Thoth Deck**
- Alternatieve symboliek, minder tekstlabels
- **Starpath Custom Deck** (optioneel)
- Uniek design afgestemd op merkidentiteit

1.2 Spreads (leggingen)

- **Three Card Spread**: verleden, heden, toekomst
- **Celtic Cross**: 10 kaarten, uitgebreide context
- **One Card Pull**: snelle dagelijkse energie
- **Relationship Spread**: jij, de ander, dynamiek
- **Year Ahead Spread**: 12 kaarten (maand per maand)

---

2. Koffiedik-symbolenbibliotheek

2.1 Structuur

```
{
  "symbol": "Vlinder",
  "meaning": {
    "NL": "Verandering, nieuwe fase",
    "EN": "Transformation, new phase",
    "TR": "Değişim, yeni dönem"
  },
  "tags": ["transformatie","licht"]
}
```

2.2 Voorbeeldsymbolen

- Vlinder
- Sleutel

- Hart
- Berg
- Vis
- Zon
- Maan

---

3. Numerologie-tabellen

3.1 Levenspadnummers (Life Path)

Nummer	NL-betekenis	EN-meaning	TR-anlam
1	Leiderschap, initiatief	Leadership, initiative	Liderlik, girişimcilik
2	Harmonie, samenwerking	Harmony, cooperation	Uyum, işbirliği
3	Creativiteit, expressie	Creativity, expression	Yaratıcılık, ifade
…			

3.2 Overige nummers

- Bestemmingsnummer
- Persoonlijkheidsnummer
- Zielsverlangen

---

4. Badge- & levelsysteem

4.1 Badges

- **Nieuweling**: eerste reading voltooid
- **Ontdekker**: 5 verschillende spreads geprobeerd
- **Symbolenmeester**: 20 koffiediksymbolen herkend
- **Numerologie-expert**: alle getallen 1–9 uitgelegd gekregen
- **Sterrenreiziger**: 50 sessies voltooid

4.2 Levels

- Level 1–5: op basis van XP-punten (bijv. 10 XP per sessie)
- Bonus XP voor uploads, feedback geven, en het delen (anoniem) van resultaten in de community

---

5. Dataformaat & opslag

- Opslag in **JSON** voor flexibiliteit
- Meertalige velden (`NL`, `EN`, `TR`)
- ID-veld voor koppeling in database (Prisma model: `TarotCard`, `CoffeeSymbol`, `NumerologyNumber`, `Badge`)

---

6. Gebruik in AI Promptbibliotheek

- AI kan automatisch de juiste betekenis ophalen en verwerken
- Consistentie van interpretaties in alle modules
- Versiebeheer van seed-data (v1.0, v1.1…)

Volgende stap: Integratie met **Testplan & Kwaliteitscontrole** zodat alle seed-data automatisch wordt gevalideerd en AI-outputs hiermee worden vergeleken.

StarpathVision – Seed-data & Contentvoorbeelden

> Doel: Direct inzetbare startdata voor dev, test, demo en AI-fine-tuning. Formaten: JSON/NDJSON. **Let op**: alle teksten generiek/veilig, zonder persoonsdata.

---

1) Tarot

1.1 Decks


```
[
  { "id": "RWS", "name": "Rider–Waite–Smith", "lang": ["nl","en","tr"], "copyright": "public-domain", "cards": 78 },
  { "id": "THOTH", "name": "Thoth", "lang": ["en"], "copyright": "restricted", "cards": 78 },
  { "id": "MARSEILLE", "name": "Tarot de Marseille", "lang": ["nl","en","fr"], "copyright": "public-domain", "cards": 78 }
]
```


1.2 Spreads


```
[
  { "id": "3C", "name": "Drie Kaarten", "positions": ["verleden","heden","toekomst"], "min": 3, "max": 3 },
  { "id": "CC", "name": "Keltisch Kruis", "positions": ["huidige situatie","uitdaging","bewust","onbewust","verleden","nabije toekomst","zelf","omgeving","hoop/vrees","uitkomst"], "min": 10, "max": 10 },
  { "id": "QA", "name": "Vraag & Antwoord", "positions": ["vraag","inzicht","actie"], "min": 3, "max": 3 }
]
```


1.3 Kaart-metadata (excerpt RWS)


```
[
  { "id": "XVI_Tower", "name": {"nl":"De Toren","en":"The Tower","tr":"Kule"}, "arcana":"major", "number":16, "keywords": {"nl": ["doorbraak","plotselinge verandering"], "en":["upheaval","revelation"]} },
  { "id": "VI_Lovers", "name": {"nl":"De Geliefden","en":"The Lovers","tr":"Aşıklar"}, "arcana":"major", "number":6, "keywords": {"nl": ["keuze","verbinding"], "en":["choice","union"]} },
  { "id": "III_Empress", "name": {"nl":"De Keizerin","en":"The Empress","tr":"İmparatoriçe"}, "arcana":"major", "number":3, "keywords":
```

```
{"nl":["groei","zorg"], "en":["nurture","abundance"]} }
]
```

1.4 Statische betekenissen (compacte seed)

```
{
  "XVI_Tower": {"upright":
{"nl":"Schok die bevrijdt; waarheden komen aan het licht.", "en":"Shock that
liberates; truths revealed."}, "reversed": {"nl":"Verzet tegen verandering;
vasthouden aan oude structuren.",
"en":"Resistance to change; clinging to old structures."}},
  "VI_Lovers": {"upright": {"nl":"Keuze in verbinding; hart en verstand
verenigen.", "en":"Choice in union; align heart and mind."}},
  "III_Empress": {"upright": {"nl":"Creativiteit en verzorging.",
"en":"Creativity and care."}, "reversed": {"nl":"Overzorg of leegte; tijd
voor zelfvoeding.", "en":"Overgiving or emptiness; nourish yourself."}}
}
```

---

2) Koffiedik – Symbolenbibliotheek

```
[
  { "label":"bird", "names": {"nl":"vogel","en":"bird","tr":"kuş"},
"meanings": {"nl":["bericht","nieuws","beweging"], "en":
["message","news","movement"]} },
  { "label":"path", "names": {"nl":"pad","en":"path","tr":"yol"},
"meanings": {"nl":["keuze","richting"], "en":["choice","direction"]} },
  { "label":"ring", "names": {"nl":"ring","en":"ring","tr":"yüzük"},
"meanings": {"nl":["verbintenis","afspraak"], "en":
["commitment","agreement"]} }
]
```

---

3) Numerologie – Tabellen (basis)

```
{
  "lifePath": {
    "1": {"nl":"Initiatief en leiderschap; begin iets nieuws.",
"en":"Initiative and leadership; start something new."},
    "5": {"nl":"Verandering, vrijheid en leren door ervaring.",
"en":"Change, freedom and learning by experience."},
    "7": {"nl":"Reflectie, onderzoek en innerlijke wijsheid.",
"en":"Reflection, research, inner wisdom."}
  },
  "destiny": { "3": {"nl":"Expressie en creativiteit.", "en":"Expression and
```

```
creativity."} }
}
```

---

4) Gamification – Badges & Levels

4.1 Badges

```
[
  {"code":"STARS_DISCOVERER","name":"Sterrenontdekker","description":"Eerste
reading","points":10},
  {"code":"PATH_SEEKER","name":"Padzoeker","description":"Vijf
readings","points":20},
  {"code":"SKY_TRAVELER","name":"Hemelreiziger","description":"Eerste
combinatie-reading","points":30}
]
```

4.2 Levels

```
[
  {"level":1,"title":"Eerste stap in de sterren","minPoints":0,"benefits":
["extra tips"]},
  {"level":2,"title":"Onder de maan","minPoints":100,"benefits":["1 extra
gratis reading/maand"]},
  {"level":3,"title":"Reis door de sterren","minPoints":250,"benefits":
["speciale combinatie-analyse"]}
]
```

---

5) Demo-gebruikers & Sessies (synthetisch)

5.1 Users (NDJSON)

```
{"email":"demo+nl@starpathvision.com","name":"Sterren Reiziger","locale":"nl-
NL"}
{"email":"demo+en@starpathvision.com","name":"Star Traveler","locale":"en-
US"}
{"email":"demo+tr@starpathvision.com","name":"Yolcu","locale":"tr-TR"}
```

5.2 Sessions (NDJSON)

```
{"type":"tarot","user":"demo+nl@starpathvision.com","payload":
{"deckId":"RWS","spreadId":"3C"},"aiResult":
{"schemaVersion":"tarot.v1","summary":"Korte verandering
vooruit.","keyThemes":["transitie"],"advice":["Zet 1 stap vandaag"]}}
```

{"type":"coffee","user":"demo+en@starpathvision.com","payload":{"assetId":"s3://seed/coffee_demo.jpg"},"aiResult":{"schemaVersion":"coffee.v1","detectedSymbols":[{"label":"bird","confidence":0.82}]}}

---

6) PDFs & Social-cards – sjablonen

```
{
  "pdf": { "headerLogo": "s3://assets/logo.pdf", "watermark": "StarpathVision", "footer": "Inzichten & inspiratie; geen medisch/juridisch/financieel advies." },
  "social": { "quoteFrame": "s3://assets/quote_frame.png", "size": "1080x1080", "font": "Inter" }
}
```

---

7) Import & tooling

- **Seed scripts**: `pnpm seed` → laad JSON/NDJSON in Postgres/S3.
- **Validatie**: Zod/AJV schemas voor deck/kaart/spread/symbool.
- **Licenties**: check `copyright` veld; geen restricted assets in productie zonder toestemming.

---

8) Uitbreiding (placeholders)

- Astrologie: tekens/huizen/aspecten basis-tabellen.
- Handlijnkunde: lijnlabels + beschrijvingen.
- Meertalige uitbreidingen voor TR (vollediger glossarium).

StarpathVision – Testplan & Kwaliteitscontrole

Doel: Waarborgen van stabiliteit, correctheid en beleving door systematisch testen in alle fasen.

---

1) Testdoelen

- **Functioneel**: Alle modules (Tarot, Koffiedik, Numerologie, Combinatie) werken volgens specificatie.
- **AI-outputvalidatie**: Resultaten voldoen aan JSON-schema's en tone-of-voice.
- **Performance**: Snelle laadtijden (<2s voor belangrijkste flows).
- **Veiligheid**: AVG-conform, geen datalekken.
- **UX-consistentie**: UI volgt designrichtlijnen en is toegankelijk (A11y).

---

2) Testsoorten

2.1 Unit tests

- **Scope**: Pure functies, helpers, services.
- **Tools**: Vitest/Jest.
- **Coverage target**: ≥85% lines/branches.

2.2 Integratietests

- **Scope**: Componenten + API samen.
- **Tools**: Playwright/TestCafe.
- **Voorbeelden**: TarotFlow (selectie + AI-call + resultaat), Koffiedik-upload.

2.3 End-to-end (E2E) tests

- **Scope**: Volledige user journeys.
- **Tools**: Playwright + mock AI-API.
- **Scenario's**:
- Nieuwe gebruiker → registratie → eerste reading.
- Premium upgrade → onbeperkt readings.
- Fout in AI-output → retry + foutmelding.

2.4 AI-outputvalidatie

- **Golden outputs**: Voor elk schema, per taal, 5 referentievoorbeelden.
- **Checks**:
- JSON valid
- Schema valid
- Tone-of-voice consistent
- Geen verboden claims (medisch/juridisch/financieel)

2.5 Performance & load tests

- **Tooling**: k6 / Artillery.
- **Doelen**:
- 500 gelijktijdige gebruikers zonder >5% foutpercentage.
- Gemiddelde API-responstijd <500ms (excl. AI-latency).

2.6 Security tests

- **Scope**: OWASP Top 10.
- **Tooling**: OWASP ZAP, npm audit, Snyk.
- **Extra**: Pen-test voor productie.

---

3) Acceptatiecriteria per module

3.1 Tarot

- Deck/spread laden <1s.
- Correcte kaarten & posities.
- AI-output bevat min. 2 sleutelwoorden uit kaartmetadata.

3.2 Koffiedik

- Upload accepteert alleen afbeeldingen ≤5MB.
- AI herkent ≥1 symbool bij seed-image.

3.3 Numerologie

- Berekeningen matchen referentie-tabellen.
- Output bevat levenspad + advies.

3.4 Combinatie-motor

- Gebruikt inputs uit ≥2 methodes.
- Geen tegenstrijdigheden zonder uitleg.

---

4) Testdata & omgevingen

- **Dev**: lokale DB, mock AI.
- **Staging**: productiedata gesimuleerd, AI-sandbox.
- **Prod**: real AI, monitoring actief.
- **Seed-data**: zie *Seed-data & Contentvoorbeelden* document.

---

5) Kwaliteitscontroleproces

1. **Code review**: Min. 1 reviewer; check op security, performance, UX.
2. **CI-pipeline**:

3. Lint & typecheck
4. Unit & integratie-tests
5. Build & E2E-tests (staging)
6. **Release check**:
7. Handmatige QA op staging
8. Toegankelijkheidsaudit
9. **Post-release**:
10. Monitoring errors (Sentry)
11. AI-output sampling

---

6) Load- & performance-testplan (voorbeeld)

```
k6 run load_test.js # 1-500 users in 10m ramp-up
```

- Rapporteer P95 latency, foutpercentage, throughput.

---

7) Rapportage & metrics

- **Dagelijks**: build status, testresultaten.
- **Wekelijks**: bug count, AI-outputscorekaart.
- **Per release**: changelog + QA-rapport.

---

8) Volgende stappen

- Golden outputs definitief vastleggen.
- AI-repair-prompt testen op seed-data.
- Performance baseline vastleggen op staging.

StarpathVision – Testplan & Kwaliteitscontrole

Doel: Hoge productkwaliteit borgen met een pragmatisch, geautomatiseerd en traceerbaar testproces. Omvat unit, integratie, E2E, AI-quality, security, performance en acceptatie.

---

1) Testpiramide & tooling

- **Unit** (snelle feedback): Vitest/Jest + Testing Library (React).
- **Integratie** (API/DB): Supertest + Prisma test-DB.
- **End-to-End**: Playwright (multi-browser, mobile viewports).
- **Contract**: OpenAPI schema checks (Prism, Zod).
- **AI Quality**: JSON-schema validatie (AJV) + golden outputs.
- **Security**: OWASP ZAP/Dastardly, dependency scan (npm audit/Snyk).
- **Performance**: k6/Artillery (API), Lighthouse (web).
- **Accessibility**: axe-core + Playwright-axe, kleurcontrast checks.
- **Visual Regression**: Playwright screenshots per route.

---

2) Omgevingen

- **Local**: docker-compose (postgres, redis, minio), seed-script.
- **CI**: parallel test jobs (unit/integratie/E2E).
- **Staging**: feature flags, test-keys Stripe/AI.
- **Prod**: canary releases, observability alerts.

---

3) Testdata & seeding

- Gebruik **Seed-data & Contentvoorbeelden** (tarot, numerologie, symbolen).
- Synthetische demo-gebruikers en sessies (NDJSON).
- Idempotente seeds per test run.

---

4) Testscenario’s per flow

4.1 Auth & onboarding

- Registratie → login → wachtwoord reset.
- Consent toggles (analytics/ai-train/marketing).
- Misbruik: brute force → rate limit, captcha fallback.

4.2 Tarot

- Deck/spread selecteren → kaarten trekken → AI JSON valide → samenvatting toont binnen 3s (stream start).

- PDF export (bestand aanwezig, watermerk, datum/tijd).
- Opslaan in historiek → detailpagina correct.
- Fout: AI JSON invalid → repair-prompt/422 melding met nette UI.

4.3 Koffiedik

- Upload validatie (type/size), EXIF strip.
- Vision output met minimaal 1 symbool + confidence.
- Lage kwaliteit foto → UI toont tips, geen crash.
- PDF export werkt.

4.4 Numerologie

- Levenspadberekening deterministisch voor testcases.
- Relatieanalyse toont score + advies.
- I18n: NL/EN/TR renders correct.

4.5 Abonnementen & betalingen

- Free → Premium/Pro checkout (Stripe test-mode).
- Webhooks: `checkout.session.completed` → `Subscription.status=active`.
- Annuleren, proration, mislukt incasso → correcte status.
- Factuur PDF link beschikbaar in account.

4.6 Gamification

- Punten toekenning per actie; level-up event en badge toekenning.
- Fraudepreventie (dubbele claims) via idempotency.

---

5) AI-kwaliteit & evaluatie

- **JSON-validatie**: 100% geldig tegen schema's (fail build bij < 99%).
- **Golden outputs**: 50 casussen per module/talen; Levenscycli vastleggen in repo.
- **Leesbaarheid**: FK score-range; toon positief en empathisch.
- **Contradiction coverage**: indien tarot+numerologie aanwezig → min. 1 `agreement` of `tension`.
- **Nightly eval**: run tegen actuele modellen; bij degradatie auto-rollback van prompt-template/flag.

---

6) Performance & SLO's

- **T2F (first reading)**: < 2 min.
- **API latency p95**: < 300 ms (excl. AI-calls).
- **AI stream start**: < 3 s p95.
- **Uptime**: ≥ 99.9%.
- **Front-end Core Web Vitals**: LCP < 2.5 s, CLS < 0.1, INP < 200 ms.

Loadtests (k6):
- 100 rps 5 min steady → error rate < 0.5%, p95 < 500 ms.
- Spike 5x → geen crash; queue drain < 2 min.

---

7) Security & privacy tests

- Auth bypass, CSRF/XSS/SSRF/SQLi probes.
- RBAC checks (admin routes afgeschermd).
- Upload antivirus (ClamAV) simulatie met EICAR.
- Data export/delete rechten (AVG) werkend.
- Back-ups restore oefening (tabletop) elk kwartaal.

---

8) Acceptatiecriteria (MVP)

- Binnenkomst → Ontdekkingsreis → Gratis tarot PDF in < 2 min.
- Checkout Premium werkt met iDEAL testflow; subscription actief.
- Historiek toont 2+ sessies; share-quote werkt.
- JSON-schema's voor alle modules groen in CI.
- A11y: toetsenbordnav en labels op kritieke schermen.

---

9) CI/CD integratie

- Github Actions: lint/typecheck → unit → integratie → E2E → build.
- Playwright artifacts (traces, video, screenshots) uploaden.
- OpenAPI lint + drift check.
- Coverage drempels: 80% unit, 70% integratie.
- Canary deploy + synthetic monitoring (Checkly) gate voor prod.

---

10) Rapportage & regressie

- Testrapport in PR (summary badges).
- Flaky tests detectie en quarantaine.
- Release notes koppelen aan passed test suites.
- Post-incident review met concrete preventies.

---

11) Volgende stappen

- Axes voor A11y uitbreiden (screenreader scripts).
- Extra golden cases voor TR.
- Chaos scenario's (AI time-outs, webhook retries mislukken).
- Visual regression baseline per release tag.

StarpathVision – UI Kit & Componenten

> Doel: Ontwikkelaars en ontwerpers voorzien van consistente bouwstenen (React + Tailwind + Framer Motion) met documentatie, props en voorbeeldgebruik.

---

1) Designprincipes

- **Mystiek & modern**: Donkere basis (bg-gray-900) met kosmische accenten (#a78bfa, #fbbf24).
- **Rustige animaties**: Gebruik Framer Motion voor subtiele fades/slides.
- **Responsief**: Mobile-first, grid-based layout, min. 320px breed.
- **Toegankelijkheid (A11y)**: ARIA-labels, toetsenbordnavigatie, kleurcontrast ≥ 4.5:1.

---

2) Component-map

```
src/components/
  atoms/
    Button.tsx
    Icon.tsx
    Card.tsx
    Input.tsx
    Avatar.tsx
  molecules/
    TarotCard.tsx
    CoffeeSymbol.tsx
    NumerologyChart.tsx
    Stepper.tsx
    Modal.tsx
  organisms/
    ReadingFlow.tsx
    DeckSelector.tsx
    SpreadBoard.tsx
    PhotoUploader.tsx
    ResultPanel.tsx
    CombinationView.tsx
  layout/
    Header.tsx
    Footer.tsx
    Sidebar.tsx
    ResponsiveGrid.tsx
```

---

3) Componentdocumentatie (voorbeeld)

3.1 Atom – Button

```
<Button
  variant="primary" // primary | secondary | ghost
  size="lg"         // sm | md | lg
  onClick={() => doSomething()}
  disabled={false}
>
  Begin jouw reis
</Button>
```

- **Styling**: Tailwind classes + variant map.
- **Animatie**: hover\:scale-105, tap\:scale-95.

3.2 Molecule – TarotCard

```
<TarotCard
  cardId="XVI_Tower"
  faceUp={true}
  onFlip={(id) => console.log('flipped', id)}
/>
```

- **Props**: `cardId` (string), `faceUp` (bool), `onFlip` (callback).
- **Data**: haalt kaartmetadata uit context/API.
- **Animatie**: 3D flip via Framer Motion.

---

4) Stijlrichtlijnen

- **Kleuren**:

```
const colors = {
  primary: '#a78bfa',
  secondary: '#fbbf24',
  bg: '#0f172a',
  accent: '#38bdf8'
}
```

- **Typografie**: Inter (UI), Playfair Display (titels).
- **Iconen**: Lucide-react, custom kosmos-iconen in `assets/icons/`.

---

5) Animatievoorbeelden

5.1 Fade-in

```
<motion.div initial={{opacity:0}} animate={{opacity:1}}
transition={{duration:0.5}}>
  <Content />
</motion.div>
```

5.2 Card Flip

```
<motion.div animate={{rotateY: faceUp ? 0 : 180}} transition={{duration:0.8}}
>
  <CardFront />
  <CardBack />
</motion.div>
```

---

6) UI-states & varianten

- **Loading**: skeletons in plaats van spinners waar mogelijk.
- **Error**: vriendelijke foutmeldingen met retry-knop.
- **Empty**: zachte call-to-action (“Nog geen readings – begin vandaag”).
- **Success**: bevestigingsscherm met confetti-animatie.

---

7) Integratie

- Alle componenten exporteren vanuit `index.ts` in hun map.
- Storybook gebruiken voor visuele test & documentatie.
- Props en events documenteren in JSDoc + Storybook.

---

8) Volgende stappen

- Mockups koppelen aan componentdocs.
- Accessibility-audit uitvoeren in Storybook.
- Dark/light-mode variant toevoegen (toggle in settings).

```
'use client'

import React, { useMemo, useRef, useState } from 'react'
import { motion } from 'framer-motion'
import { Star, Moon, Sparkles, Upload, FileDown, Wand2, History, Coffee, Layers, BadgeCheck, Coins, Share2 } from 'lucide-react'
import { LineChart, Line, CartesianGrid, XAxis, YAxis, Tooltip, ResponsiveContainer } from 'recharts'

// shadcn/ui (basic components)
import { Button } from '@/components/ui/button'
import { Card, CardContent, CardFooter, CardHeader, CardTitle } from '@/components/ui/card'
import { Tabs, TabsContent, TabsList, TabsTrigger } from '@/components/ui/tabs'
import { Input } from '@/components/ui/input'
import { Label } from '@/components/ui/label'
import { Badge } from '@/components/ui/badge'
import { Progress } from '@/components/ui/progress'

// ------------------------------------------------------------
// StarpathVision – UI Kit & Low-Fi Mockups (single-file preview)
// ------------------------------------------------------------
// Richtlijnen:
// - Mobile-first, donkere kosmische stijl
// - Geen echte backend; dit is een visueel klikbaar prototype
// - Componenten en layout zijn afgestemd op het Build Spec-document

const activityData = [
  { name: 'M', value: 2 },
```

```
  { name: 'D', value: 3 },
  { name: 'W', value: 1 },
  { name: 'D', value: 4 },
  { name: 'V', value: 2 },
  { name: 'Z', value: 5 },
  { name: 'Z', value: 3 },
]

const badges = [
  { code: 'STARS_DISCOVERER', name: 'Sterrenontdekker' },
  { code: 'PATH_SEEKER', name: 'Padzoeker' },
  { code: 'SKY_TRAVELER', name: 'Hemelreiziger' },
]

function StarryBackground() {
  return (
    <div className="pointer-events-none fixed inset-0 -z-10 bg-gradient-to-b from-[#070814] via-[#0a0f2a] to-black">
      {/* zachte nevel */}
      <motion.div
        initial={{ opacity: 0 }}
        animate={{ opacity: 1 }}
        transition={{ duration: 1.2 }}
        className="absolute inset-0"
        style={{
          background: 'radial-gradient(1200px 600px at 70% 20%, rgba(95,88,196,0.16), transparent 60%), radial-gradient(900px 500px at 20% 80%, rgba(124,58,237,0.12), transparent 60%)',
        }}
      />
```

```
      {/* sterren (lichtjes) */}
      <div className="absolute inset-0 opacity-40 [background-image:radial-gradient(1px_1px_at_10px_10px,rgba(255,255,255,0.5),transparent_0),radial-gradient(1px_1px_at_50px_80px,rgba(255,255,255,0.35),transparent_0),radial-gradient(1px_1px_at_200px_120px,rgba(255,255,255,0.3),transparent_0)] [background-size:200px_200px]" />
    </div>
  )
}

function SectionHeader({ title, subtitle, icon: Icon }) {
  return (
    <div className="flex items-center gap-3">
      {Icon ? <Icon className="h-5 w-5 text-indigo-300" /> : null}
      <div>
        <h2 className="text-lg md:text-xl font-semibold text-indigo-100">{title}</h2>
        {subtitle ? <p className="text-xs md:text-sm text-indigo-300/80">{subtitle}</p> : null}
      </div>
    </div>
  )
}

function EnergyWidget() {
  return (
    <Card className="bg-white/5 border-white/10">
      <CardHeader>
```

```
        <CardTitle className="text-indigo-100 flex items-center gap-
2"><Moon className="h-4 w-4" /> Energie van de dag</CardTitle>
      </CardHeader>
      <CardContent className="text-indigo-200 text-sm">
        <div className="flex items-center justify-between">
          <div>
            <p>Maanfase: Eerste Kwartier</p>
            <p>Numerologie-dag: 5 — focus op verandering en
beweging.</p>
          </div>
          <Sparkles className="h-6 w-6 text-indigo-300" />
        </div>
      </CardContent>
    </Card>
  )
}

function ActivityChart() {
  return (
    <Card className="bg-white/5 border-white/10">
      <CardHeader>
        <CardTitle className="text-indigo-100 flex items-center gap-
2"><History className="h-4 w-4" /> Activiteit (7 dagen)</CardTitle>
      </CardHeader>
      <CardContent>
        <div className="h-40">
          <ResponsiveContainer width="100%" height="100%">
            <LineChart data={activityData} margin={{ top: 5, left: -20, right:
10, bottom: 0 }}>
              <CartesianGrid strokeDasharray="3 3"
stroke="rgba(255,255,255,0.1)" />
```

```
            <XAxis dataKey="name" stroke="rgba(255,255,255,0.6)" tick={{
fill: 'rgba(255,255,255,0.6)', fontSize: 12 }} />
            <YAxis stroke="rgba(255,255,255,0.6)" tick={{ fill:
'rgba(255,255,255,0.6)', fontSize: 12 }} allowDecimals={false} />
            <Tooltip contentStyle={{ backgroundColor: 'rgba(12,15,34,0.9)',
border: '1px solid rgba(255,255,255,0.1)', color: 'white' }} />
            <Line type="monotone" dataKey="value" stroke="#8b5cf6"
strokeWidth={2} dot={false} />
          </LineChart>
        </ResponsiveContainer>
      </div>
    </CardContent>
  </Card>
 )
}

function GamificationWidget() {
 return (
  <Card className="bg-white/5 border-white/10">
    <CardHeader>
      <CardTitle className="text-indigo-100 flex items-center gap-
2"><BadgeCheck className="h-4 w-4" /> Jouw voortgang</CardTitle>
    </CardHeader>
    <CardContent>
      <div className="flex items-center justify-between text-indigo-200">
        <div>
          <div className="text-sm">Level 2 — Onder de maan</div>
          <div className="text-xs opacity-75">Punten: 120 / 150</div>
        </div>
        <Coins className="h-6 w-6" />
      </div>
```

```
        <div className="mt-3">
          <Progress value={80} className="h-2" />
        </div>
        <div className="mt-3 flex flex-wrap gap-2">
          {badges.map(b => (
            <Badge key={b.code} variant="secondary" className="bg-indigo-600/30 text-indigo-100 border-indigo-500/40">{b.name}</Badge>
          ))}
        </div>
      </CardContent>
    </Card>
  )
}

function QuickStart({ goTarot, goNumerology, goCoffee }) {
  return (
    <Card className="bg-white/5 border-white/10">
      <CardHeader>
        <CardTitle className="text-indigo-100 flex items-center gap-2"><Wand2 className="h-4 w-4" /> Snel starten</CardTitle>
      </CardHeader>
      <CardContent className="grid grid-cols-1 md:grid-cols-3 gap-3">
        <Button onClick={goTarot} className="bg-indigo-600 hover:bg-indigo-500">Tarot reading</Button>
        <Button onClick={goNumerology} className="bg-indigo-600 hover:bg-indigo-500">Numerologie</Button>
        <Button onClick={goCoffee} className="bg-indigo-600 hover:bg-indigo-500 flex items-center gap-2"><Coffee className="h-4 w-4" /> Koffiedik upload</Button>
      </CardContent>
    </Card>
```

```
  )
}


function Dashboard({ setTab }) {
  return (
    <div className="grid gap-4 md:grid-cols-2">
      <EnergyWidget />
      <QuickStart goTarot={() => setTab('tarot')} goNumerology={() => setTab('journey')} goCoffee={() => setTab('coffee')} />
      <ActivityChart />
      <GamificationWidget />
      <Card className="md:col-span-2 bg-white/5 border-white/10">
        <CardHeader>
          <CardTitle className="text-indigo-100 flex items-center gap-2"><History className="h-4 w-4" /> Laatste sessies</CardTitle>
        </CardHeader>
        <CardContent className="text-indigo-200 text-sm grid gap-2">
          <div className="flex items-center justify-between border border-white/10 rounded-lg p-3">
            <span>Tarot — Drie kaarten — 12 aug 2025</span><Button variant="secondary" size="sm" className="bg-white/10">Bekijken</Button>
          </div>
          <div className="flex items-center justify-between border border-white/10 rounded-lg p-3">
            <span>Numerologie — Levenspad — 09 aug 2025</span><Button variant="secondary" size="sm" className="bg-white/10">Bekijken</Button>
          </div>
        </CardContent>
      </Card>
    </div>
```

```
  )
}


function Home({ onStart }) {
  return (
    <div className="flex flex-col items-center text-center gap-6">
      <motion.div initial={{ opacity: 0, y: 10 }} animate={{ opacity: 1, y: 0 }} transition={{ duration: 0.6 }} className="space-y-3">
        <h1 className="text-3xl md:text-5xl font-semibold text-white tracking-tight">
          StarpathVision: <span className="text-indigo-300">Jouw Gids naar de Sterren</span>
        </h1>
        <p className="text-indigo-200/90 max-w-2xl mx-auto">
          Eén plek voor Tarot, Numerologie en Koffiedik — met AI-uitleg en samengestelde inzichten. Rustig, helder, respectvol.
        </p>
      </motion.div>
      <Button onClick={onStart} size="lg" className="bg-indigo-600 hover:bg-indigo-500 h-12 px-8">Begin jouw Reis</Button>
      <div className="grid grid-cols-1 md:grid-cols-3 gap-4 w-full max-w-5xl">
        <Card className="bg-white/5 border-white/10"><CardHeader><CardTitle className="text-indigo-100">Tarot</CardTitle></CardHeader><CardContent className="text-indigo-200 text-sm">Kies een deck en legging, draai kaarten om en ontvang heldere AI-uitleg.</CardContent></Card>
        <Card className="bg-white/5 border-white/10"><CardHeader><CardTitle className="text-indigo-100">Numerologie</CardTitle></CardHeader><CardContent className="text-indigo-200 text-sm">Automatische berekeningen op basis van naam en geboortedatum.</CardContent></Card>
        <Card className="bg-white/5 border-white/10"><CardHeader><CardTitle className="text-indigo-
```

```
100">Koffiedik</CardTitle></CardHeader><CardContent
className="text-indigo-200 text-sm">Upload een foto, AI herkent
patronen en symbolen, met confidence.</CardContent></Card>

      </div>

    </div>

  )

}


function Journey({ onComplete }) {

  const [date, setDate] = useState('')

  const [place, setPlace] = useState('')

  const [partnerName, setPartnerName] = useState('')

  const [partnerDate, setPartnerDate] = useState('')


  return (

    <div className="grid gap-4 max-w-2xl mx-auto">

      <SectionHeader title="Ontdekkingsreis" subtitle="In plaats van een
saai formulier — een korte start met 3 vragen" icon={Star} />

      <Card className="bg-white/5 border-white/10">

        <CardContent className="grid gap-4 pt-6">

          <div className="grid gap-2">

            <Label className="text-indigo-200">Wanneer ben jij geboren
onder de sterren?</Label>

            <Input type="date" value={date} onChange={e =>
setDate(e.target.value)} className="bg-white/10 border-white/20 text-
indigo-100 placeholder:text-indigo-300/50" placeholder="Geboortedatum"
/>

          </div>

          <div className="grid gap-2">

            <Label className="text-indigo-200">Waar op aarde is jouw reis
begonnen?</Label>
```

```
          <Input value={place} onChange={e => setPlace(e.target.value)} className="bg-white/10 border-white/20 text-indigo-100" placeholder="Geboorteplaats" />
        </div>
        <div className="grid gap-2">
          <Label className="text-indigo-200">Wie zou je graag beter willen begrijpen? (optioneel)</Label>
          <div className="grid md:grid-cols-2 gap-3">
            <Input value={partnerName} onChange={e => setPartnerName(e.target.value)} className="bg-white/10 border-white/20 text-indigo-100" placeholder="Naam" />
            <Input type="date" value={partnerDate} onChange={e => setPartnerDate(e.target.value)} className="bg-white/10 border-white/20 text-indigo-100" />
          </div>
        </div>
      </CardContent>
      <CardFooter className="flex justify-end">
        <Button onClick={onComplete} className="bg-indigo-600 hover:bg-indigo-500">Start Gratis Reading</Button>
      </CardFooter>
    </Card>
  </div>
 )
}


function TarotCard({ title, flipped, onFlip }) {
 return (
  <div className="[perspective:1000px] h-40 w-28 mx-auto" onClick={onFlip}>
    <div className={`relative h-full w-full transition-transform duration-500 [transform-style:preserve-3d] ${flipped ? '[transform:rotateY(180deg)]' : ''}`}>
```

```
        <div className="absolute inset-0 bg-indigo-900/60 border border-indigo-500/40 rounded-xl grid place-items-center text-indigo-200 [backface-visibility:hidden]">
          ★
        </div>
        <div className="absolute inset-0 bg-white/95 text-indigo-900 rounded-xl border border-indigo-800/40 grid place-items-center font-medium [transform:rotateY(180deg)] [backface-visibility:hidden] p-2 text-center">
          {title}
        </div>
      </div>
    </div>
  )
}

function TarotFlow() {
  const [deck, setDeck] = useState('Rider-Waite-Smith')
  const [spread, setSpread] = useState('Drie Kaarten')
  const [flips, setFlips] = useState([false, false, false])

  const cardTitles = ['Toren (XVI)', 'Geliefden (VI)', 'Keizerin (III, omgekeerd)']

  const flip = (i) => setFlips(prev => prev.map((f, idx) => (idx === i ? !f : f)))

  return (
    <div className="grid gap-4">
      <SectionHeader title="Tarot reading" subtitle="Kies deck & legging, draai kaarten om en zie AI-uitleg" icon={Layers} />
```

```
      <Card className="bg-white/5 border-white/10">
        <CardContent className="grid md:grid-cols-3 gap-4 pt-6">
          <div className="grid gap-2">
            <Label className="text-indigo-200">Deck</Label>
            <select value={deck} onChange={(e) =>
setDeck(e.target.value)} className="bg-white/10 border border-white/20
rounded-md p-2 text-indigo-100">
              <option>Rider-Waite-Smith</option>
              <option>Thoth</option>
              <option>Marseille</option>
            </select>
          </div>
          <div className="grid gap-2">
            <Label className="text-indigo-200">Legging</Label>
            <select value={spread} onChange={(e) =>
setSpread(e.target.value)} className="bg-white/10 border
border-white/20 rounded-md p-2 text-indigo-100">
              <option>Drie Kaarten</option>
              <option>Keltisch Kruis</option>
              <option>Vraag & Antwoord</option>
            </select>
          </div>
          <div className="grid gap-2">
            <Label className="text-indigo-200">Acties</Label>
            <div className="flex gap-2">
              <Button className="bg-indigo-600 hover:bg-indigo-500">Trek
kaarten</Button>
              <Button variant="secondary"
className="bg-white/10">Opnieuw</Button>
            </div>
          </div>
```

```
      </CardContent>
    </Card>

    <div className="grid grid-cols-3 gap-4">
      {cardTitles.map((t, i) => (
        <TarotCard key={t} title={t} flipped={flips[i]} onFlip={() => flip(i)} />
      ))}
    </div>

    <Card className="bg-white/5 border-white/10">
      <CardHeader>
        <CardTitle className="text-indigo-100">AI-uitleg (voorbeeld)</CardTitle>
      </CardHeader>
      <CardContent className="text-indigo-100/90 text-sm space-y-3">
        <p><strong>Samenvatting:</strong> Verandering vraagt om moed, verbinding en zorg voor jezelf. Wat loslaat, maakt ruimte voor iets voedends.</p>
        <p><strong>Thema’s:</strong> transitie, relatie-keuzes, creativiteit.</p>
        <ul className="list-disc pl-5">
          <li><em>Kruisinzicht met dagenergie:</em> dag 5 ondersteunt experimenteren, kleine stappen en flexibiliteit.</li>
          <li><em>Advies:</em> 1) maak 1 concrete keuze deze week, 2) plan 2 momenten voor zelfzorg, 3) vraag steun aan een vertrouwd persoon.</li>
        </ul>
      </CardContent>
      <CardFooter className="flex gap-2">
```

```
        <Button className="bg-indigo-600 hover:bg-indigo-500 flex items-center gap-2"><FileDown className="h-4 w-4" /> Exporteer PDF</Button>
        <Button variant="secondary" className="bg-white/10 flex items-center gap-2"><Share2 className="h-4 w-4" /> Deel quote</Button>
      </CardFooter>
    </Card>
  </div>
 )
}

function CoffeeFlow() {
 const [preview, setPreview] = useState(null)
 const [analysis, setAnalysis] = useState(null)
 const boxes = useMemo(() => (
  [
   { x: '35%', y: '30%', w: '15%', h: '12%' },
   { x: '60%', y: '45%', w: '18%', h: '14%' },
  ]
 ), [])

 const onUpload = (e) => {
  const file = e.target.files?.[0]
  if (!file) return
  const url = URL.createObjectURL(file)
  setPreview(url)
  setAnalysis(null)
 }

 const runAnalysis = () => {
```

```
    setAnalysis({
      detectedSymbols: [
        { label: 'vogel', confidence: 0.78 },
        { label: 'pad', confidence: 0.64 },
      ],
      summary: 'Bericht in aantocht; beweging en keuze. Neem een lichte eerste stap.',
      advice: ['Bel 1 persoon vandaag', 'Plan 30 minuten reflectie', 'Laat 1 ding los dat je niet dient'],
    })
  }


  return (
    <div className="grid gap-4">
      <SectionHeader title="Koffiedik lezen" subtitle="Upload een foto; AI herkent patronen en symbolen" icon={Coffee} />
      <Card className="bg-white/5 border-white/10">
        <CardContent className="pt-6 grid gap-3">
          <Label className="text-indigo-200">Foto van het kopje</Label>
          <Input type="file" accept="image/*" onChange={onUpload} className="bg-white/10 border-white/20 text-indigo-100" />
          <div className="grid md:grid-cols-2 gap-4">
            <div className="aspect-square rounded-xl border border-white/10 bg-black/30 relative overflow-hidden grid place-items-center">
              {preview ? (
                <div className="relative w-full h-full">
                  <img src={preview} alt="preview" className="object-cover w-full h-full opacity-80" />
                  {boxes.map((b, i) => (
                    <div key={i} className="absolute border-2 border-indigo-400/80 rounded" style={{ left: b.x, top: b.y, width: b.w, height: b.h }} />
```

```
            ))}
          </div>
        ) : (
          <div className="text-indigo-300 flex flex-col items-center">
            <Upload className="h-8 w-8 mb-2" />
            <span className="text-sm">Nog geen afbeelding geüpload</span>
          </div>
        )}
      </div>
      <div className="text-indigo-200 text-sm">
        <p className="mb-2">Fototips: goed licht, scherp beeld, bovenaanzicht.</p>
        <Button onClick={runAnalysis} className="bg-indigo-600 hover:bg-indigo-500">Analyseer</Button>
        {analysis && (
          <div className="mt-4 space-y-2">
            <div className="text-indigo-100 font-medium">Gevonden symbolen</div>
            <ul className="list-disc pl-5">
              {analysis.detectedSymbols.map((s, i) => (
                <li key={i}>{s.label} — confidence {(s.confidence * 100).toFixed(0)}%</li>
              ))}
            </ul>
            <div className="text-indigo-100 font-medium">Samenvatting</div>
            <p>{analysis.summary}</p>
            <div className="text-indigo-100 font-medium">Advies</div>
            <ul className="list-disc pl-5">
              {analysis.advice.map((a, i) => (<li key={i}>{a}</li>))}
```

```
              </ul>
            </div>
          )}
        </div>
      </div>
    </CardContent>
  </Card>
 </div>
 )
}


function Pricing() {
 return (
  <div className="grid gap-4 md:grid-cols-3">
   {[
    { name: 'Free', price: '€0', features: ['1 korte tarot p/w', 'Basis numerologie', 'Beperkte historiek'] },
    { name: 'Premium', price: '€12/m', features: ['Onbeperkt modules', 'Uitgebreide PDF\'s', 'Combinatie-inzichten', 'Volledige historiek'] },
    { name: 'Pro', price: '€24/m', features: ['Alles uit Premium', 'Live AI chat', 'Maandrapport'] },
   ].map((plan) => (
    <Card key={plan.name} className="bg-white/5 border-white/10 flex flex-col">
     <CardHeader>
      <CardTitle
className="text-indigo-100">{plan.name}</CardTitle>
     </CardHeader>
     <CardContent className="text-indigo-200 text-sm space-y-2 flex-
1">
```

```
            <div className="text-3xl font-semibold text-white">{plan.price}</div>
            <ul className="list-disc pl-5">
              {plan.features.map((f) => (<li key={f}>{f}</li>))}
            </ul>
          </CardContent>
          <CardFooter>
            <Button className="bg-indigo-600 hover:bg-indigo-500 w-full">Kies {plan.name}</Button>
          </CardFooter>
        </Card>
      ))}
    </div>
  )
}

export default function StarpathVisionUIKit() {
  const [tab, setTab] = useState('home')

  return (
    <div className="min-h-screen text-indigo-100">
      <StarryBackground />

      <div className="max-w-6xl mx-auto px-4 py-6 md:py-10">
        {/* Header */}
        <div className="flex items-center justify-between mb-6">
          <div className="flex items-center gap-3">
            <motion.div initial={{ scale: 0.9, opacity: 0 }} animate={{ scale: 1, opacity: 1 }} className="grid place-items-center h-10 w-10 rounded-2xl bg-indigo-600/30 border border-indigo-400/40">
```

```
            <Star className="h-5 w-5 text-indigo-200" />
          </motion.div>
          <div>
            <div className="text-white font-semibold tracking-tight">StarpathVision</div>
            <div className="text-xs text-indigo-300">Jouw Gids naar de Sterren</div>
          </div>
        </div>
        <div className="hidden md:flex items-center gap-2">
          <Badge variant="secondary" className="bg-white/10 text-indigo-100 border-white/20">MVP Mockup</Badge>
        </div>
      </div>

      {/* Tabs */}
      <Tabs value={tab} onValueChange={setTab} className="space-y-6">
        <TabsList className="bg-white/10 border border-white/10">
          <TabsTrigger value="home">Home</TabsTrigger>
          <TabsTrigger value="journey">Ontdekkingsreis</TabsTrigger>
          <TabsTrigger value="dashboard">Dashboard</TabsTrigger>
          <TabsTrigger value="tarot">Tarot</TabsTrigger>
          <TabsTrigger value="coffee">Koffiedik</TabsTrigger>
          <TabsTrigger value="pricing">Pricing</TabsTrigger>
        </TabsList>

        <TabsContent value="home">
          <Home onStart={() => setTab('journey')} />
        </TabsContent>
```

```
          <TabsContent value="journey">
            <Journey onComplete={() => setTab('tarot')} />
          </TabsContent>
          <TabsContent value="dashboard">
            <Dashboard setTab={setTab} />
          </TabsContent>
          <TabsContent value="tarot">
            <TarotFlow />
          </TabsContent>
          <TabsContent value="coffee">
            <CoffeeFlow />
          </TabsContent>
          <TabsContent value="pricing">
            <Pricing />
          </TabsContent>
        </Tabs>

        {/* Footer */}
        <div className="mt-10 text-center text-xs text-indigo-300/80">
          © {new Date().getFullYear()} StarpathVision — Inzichten & inspiratie; geen medisch/financieel/juridisch advies.
        </div>
      </div>
    </div>
  )
}
```

StarpathVision – Analytics & Event Tracking

Doel

Inzicht krijgen in gebruikersgedrag, conversies en contentprestaties door middel van gestructureerde event tracking.

---

1. Trackingplan

Core events: - `user_registered` - `onboarding_completed` - `reading_started` (methode, persona) - `reading_completed` (methode, persona) - `plan_upgraded` / `plan_downgraded` - `session_shared` (kanaal) - `push_opt_in` - `push_sent` - `push_clicked`

---

2. Parameters

- **User:** id, taal, plan-type.
- **Reading:** methode, persona, focus.
- **Device:** type, OS, browser.

---

3. Dashboards

- **Cohort analyse:** retentie per onboarding-type.
- **Funnel analyse:** conversie free → premium.
- **Content analytics:** best presterende spreads/persona's.

---

4. Tools

- **Collectie:** Google Analytics 4 + server-side tagging.
- **Visualisatie:** Google Data Studio / Looker.
- **Event forwarding:** Segment/Amplitude (optioneel).

---

5. Privacy

- Anonimiseer IP's.
- Geen tracking zonder consent.

Volgende stap: Security & Privacy – Uitbreiding

StarpathVision – Canvas Checklist & Status

✅Voltooid

1. **Complete Build Spec**
   – Functioneel ontwerp, technische architectuur, roadmap, beveiliging & privacy-overzicht, verdienmodel.

2. **AI Promptbibliotheek & JSON-schema's**
   – Prompt-templates per module, strikte JSON-outputdefinities, combinatie-motor logica, tone-of-voice richtlijnen.

3. **API & Datamodel Referentie (OpenAPI + Prisma)**
   – OpenAPI-spec, Prisma-modellen, API-routes, validatie, webhooks, foutafhandeling.

4. **Seed-data & Contentvoorbeelden**
   – Tarotdecks, spreads, symbolenbibliotheek, numerologie-tabellen, gamification-systeem.

5. **UI Kit & Mockups**
   – Low-fi mockups, React/Tailwind componenten, stijlrichtlijnen, animatievoorbeelden.

6. **Testplan & Kwaliteitscontrole**
   – Unit-, integratie-, end-to-end tests, AI-output validatie, acceptatiecriteria, performance-tests.

7. **Marketing & Contentstrategie**
   – Merkpositionering, contentpijlers, social media formats, e-mailtemplates en journeys.

8. **Beveiliging & Privacy**
   – AVG/GDPR deep-dive, encryptie, dataverwijderingsflows, logging, audit, AI-identiteitsbescherming.

9. **Uitbreidingsmodules**
   – Live consultaties, community, tijdlijn, kalenderintegraties, notificaties, AI-visuals, e-commerce, gamification-verdieping, open API, toegankelijkheid.

🔆Nog te maken / Optioneel

1. **Visuele High-Fidelity Prototypes** *(optioneel)*
   – Pixel-perfect UI in Figma met alle states en interacties.

2. **Operations & Deployment Handleiding** *(optioneel)*
   – CI/CD pipeline, serverconfiguraties, hostingkeuze, monitoring & incident response.

3. **Contentbibliotheek Orakel & Wijzen** *(optioneel)*
   – Uitgebreide voorbeeld-antwoorden, spreekstijlgids per persona, achtergrondverhalen.

4. **Data-archiefmodule Specificatie** *(optioneel)*
   – Detailbeschrijving van hoe sessies, chatlogs en AI-output gearchiveerd en opgehaald worden.

5. **Integratiehandleiding Externe API's** *(optioneel)*
   – Stapsgewijze instructies voor integratie met AI-modellen, beeldherkenning, betalingen, e-commerce, notificaties.

Volgende stap: Beslissen welke optionele canvassen nog worden uitgewerkt voor overdracht naar de bouwer.

StarpathVision – Contentbibliotheek Orakel & Wijzen

Doel

Een centrale bibliotheek met persoonlijkheden, verhalen, spreekwijzen en voorbeeldantwoorden voor De Orakel en de tien Wijzen.

---

1. Hoofdpersonages

1.1 Orakel (vrouw, 63 jaar)

- **Stijl:** Rustig, wijs, spreekt in metaforen.
- **Achtergrond:** Opgegroeid in een klein kustdorp, decennia ervaring in Tarot, koffiedik lezen, astrologie.
- **Unieke kenmerken:** Typt langzaam, gebruikt pauzes (`...`) om na te denken.
- **Specialiteit:** Moeilijke vragen, diepe spirituele inzichten.
- **Voorbeeldantwoord:** "Zoals de maan haar schijnsel spreidt over de stille zee, zo onthult jouw pad zich langzaam, stap voor stap..."

1.2 Meester Asterion (man, 65 jaar)

- **Stijl:** Formeel, analytisch.
- **Achtergrond:** Reiziger, bezocht oude tempels en sterrenwachten.
- **Specialiteit:** Numerologie, astrologische berekeningen.
- **Voorbeeldantwoord:** "Volgens de stand van Mars en jouw geboortecijfer, zie ik een periode van herstructurering in je leven..."

1.3 Selvara (vrouw, 54 jaar)

- **Stijl:** Warm, moederlijk.
- **Specialiteit:** Relatielezingen, liefdescompatibiliteit.

1.4 Calvantis (man, 52 jaar)

- **Stijl:** Kort en to the point.
- **Specialiteit:** Zakelijke beslissingen, loopbaanadviezen.

1.5 Lumeria (vrouw, 49 jaar)

- **Stijl:** Mysterieus, poëtisch.
- **Specialiteit:** Dromenanalyse.

1.6 Draegon (man, 47 jaar)

- **Stijl:** Vurig, gebruikt krachtige beeldspraak.
- **Specialiteit:** Levensveranderingen, doorbraken.

1.7 Cyrelle (vrouw, 44 jaar)

- **Stijl:** Praktisch, no-nonsense.
- **Specialiteit:** Dag-tot-dag beslissingen.

1.8 Tharion (man, 42 jaar)

- **Stijl:** Filosofisch.
- **Specialiteit:** Spirituele groei.

1.9 Elaris (vrouw, 38 jaar)

- **Stijl:** Creatief, luchtig.
- **Specialiteit:** Inspiratie en motivatie.

1.10 Voran (man, 36 jaar)

- **Stijl:** Jong en modern.
- **Specialiteit:** Jonger publiek, technologie en spiritualiteit.

---

2. Stijlrichtlijnen

- Nooit aangeven dat zij AI zijn.
- Elke wijze gebruikt eigen woordenschat en zinsbouw.
- Geen moderne tech-termen bij oudere personages, wel bij jongere indien relevant.
- Historische voorbeelden toegestaan.

---

3. Voorbeeldvragen voor testen

1. Wat brengt de liefde voor mij dit jaar?
2. Hoe kan ik mijn carrière verbeteren?
3. Welke gevaren moet ik vermijden?
4. Wat zegt mijn droom over water?
5. Hoe ziet mijn financiële toekomst eruit?
6. Welke verborgen talenten heb ik?
7. Wat betekent deze dag voor mij?
8. Hoe kan ik omgaan met een moeilijke keuze?
9. Wat zegt het lot over mijn relatie?
10. Waar moet ik op letten deze maand?

---

4. Archiefkoppeling

- Elke sessie wordt opgeslagen in het klantarchief.
- Personages kunnen terugverwijzen naar eerdere gesprekken.

Volgende stap: Data-archiefmodule specificatie.

StarpathVision – Data-archiefmodule Specificatie

Doel

Een robuuste module die automatisch alle interacties, sessies en media van klanten archiveert, zodat de AI (Orakel en Wijzen) kan teruggrijpen op eerdere gesprekken.

---

1. Functies

- **Automatisch loggen:** Elke chatregel, vraag, antwoord, en AI-output wordt automatisch opgeslagen.
- **Media-opslag:** Geüploade foto's (tarotkaarten, koffiekopjes, handpalmen) worden gekoppeld aan de bijbehorende sessie.
- **Versiebeheer:** Wijzigingen in interpretaties of correcties door de AI worden als nieuwe versies opgeslagen.
- **Doorzoekbaar archief:** AI kan relevante passages uit eerdere sessies ophalen.
- **Klanttoegang:** Gebruiker kan eigen sessie-archief bekijken/downloaden (AVG/GDPR-compliant).

---

2. Technische Architectuur

- **Database:**
- **Tabel: Users** → klantgegevens.
- **Tabel: Sessions** → metadata per sessie (datum, type reading, gebruikte wijze).
- **Tabel: Messages** → losse berichten met tijdstempel, afzender (klant, AI), gekoppeld aan sessie.
- **Tabel: Media** → bestands-URL, type (foto/video), gekoppeld aan sessie.
- **Tabel: AIReferences** → links naar relevante vorige sessies voor context.
- **Opslag:**
- AWS S3 / Firebase Storage voor media.
- Geëncrypteerde opslag voor gevoelige data.
- **Indexering:**
- Full-text search index (bijv. Elasticsearch of PostgreSQL FTS) voor snelle zoekopdrachten.

---

3. JSON Schema – Voorbeeld Sessieobject

```
{
  "sessionId": "sess_20250814_001",
  "userId": "user_12345",
  "date": "2025-08-14T12:34:56Z",
  "type": "tarot",
  "wiseEntity": "Orakel",
  "messages": [
    {
      "timestamp": "2025-08-14T12:35:10Z",
```

```
      "sender": "user",
      "content": "Wat brengt de liefde voor mij dit jaar?"
    },
    {
      "timestamp": "2025-08-14T12:35:45Z",
      "sender": "Orakel",
      "content": "Zoals de maan haar schijnsel spreidt..."
    }
  ],
  "media": [
    {
      "url": "https://cdn.starpathvision.com/uploads/sess_20250814_001/
tarotspread.jpg",
      "type": "image/jpeg",
      "description": "Foto van de getrokken kaarten"
    }
  ],
  "aiReferences": [
    {
      "sessionId": "sess_20240612_007",
      "reason": "Vorige liefdeslezing"
    }
  ]
}
```

---

4. Privacy & AVG

- Gegevens alleen inzien door klant en systeem.
- Gegevens verwijderbaar op verzoek.
- Downloadoptie in JSON of PDF.

Volgende stap: Integratiehandleiding Externe API's.

StarpathVision – High-Fidelity Prototypes (Figma)

Doel

De pixel-perfect visuele uitwerking van de belangrijkste schermen, zodat ontwikkelaars en designers exact weten hoe het platform eruit moet zien en aan moet voelen.

Richtlijnen

• **Tooling:** Ontwerpen in Figma, desktop + mobiel varianten.
• **Resoluties:**
• Desktop: 1440px breed
• Tablet: 1024px breed
• Mobiel: 390px breed (iPhone 14 Pro referentie)
• **Kleurthema:**
• Donkere achtergrond (#0B0C10) met subtiele sterrenhemel overlay
• Accenten in goud (#D4AF37) en kosmisch paars (#6A0DAD)
• **Typografie:**
• Titel: Cinzel Decorative (Google Fonts)
• Tekst: Lato / Open Sans
• **Iconografie:**
• Mystieke symbolen (maanfasen, sterren, taroticonen)
• SVG's voor schaalbaarheid

Schermen

1. Homepage

• Hero sectie met grote titel: *"StarpathVision – Jouw Gids naar de Sterren"*
• Call-to-action: "Begin jouw reis" knop
• Mystieke illustratie (tarotkaarten + sterrenbol)
• Navigatie: Home | Ontdekkingsreis | Methodes | Over Ons | Inloggen

2. Ontdekkingsreis / Registratie

• Speelse vragenstroom (geboortedatum, geboorteplaats, focusvraag)
• Animaties van vallende sterren bij elke vraag
• Progress bar bovenin

3. Dashboard

• Persoonlijk welkom: "Welkom terug, [naam]"
• Overzicht van eerdere sessies (met mini-thumbnail per methode)
• Snelle start tegels: Tarot | Numerologie | Koffiedik | Astrologie | Palmistry
• Dagelijkse boodschap-widget

4. Tarot Reading

- Virtuele kaarttafel met drag & drop kaarten
- Animatie bij omdraaien (flip)
- Uitleg verschijnt in zijpaneel met scroll

5. Live Chat met Orakel

- Chatvenster met langzaam typende AI (3 sec per zin)
- Profiel van gekozen Wijze (naam + korte beschrijving)
- Historie zichtbaar in zijpaneel

6. Profiel & Archief

- Persoonlijke gegevens bewerken
- Alle sessies chronologisch
- Zoek- en filterfunctie (per methode / datum)

Leverables

- 1 Figma-bestand met alle schermen in desktop, tablet en mobiel formaat
- Componentbibliotheek voor hergebruik (knoppen, velden, iconen)
- Export van SVG/icon pack
- Stijlgids in PDF

StarpathVision – Integratiehandleiding Externe API’s

Doel

Deze handleiding beschrijft hoe externe API’s moeten worden geïntegreerd in het StarpathVision-platform om functionaliteit uit te breiden, zoals betalingen, AI, mediaopslag en notificaties.

---

1. Kern-API’s

1.1 AI API (OpenAI / GPT-4 + Vision)

- **Gebruik:** Voor interpretaties, combinatiemotor, natuurlijke taal output.
- **Endpoints:**
- `/v1/chat/completions` → interpretatie, menselijke stijl
- `/v1/images/edits` → tarot- en koffiedik-beeldanalyse
- **Authenticatie:** API Key in server-side omgeving.
- **Beveiliging:** Nooit API-key clientside laden.

1.2 Beeldherkenning (Vision API)

- **Gebruik:** Analyseren van geüploade tarotkaarten, koffiedik, handlijnen.
- **Opties:** OpenAI Vision, Google Vision AI, of eigen TensorFlow-model.
- **Output:** JSON met herkende objecten, posities en labels.

1.3 Betaalverwerking (Stripe)

- **Gebruik:** Abonnementen, losse consulten.
- **Endpoints:**
- `/checkout/sessions` → nieuwe betaling starten
- `/webhook` → statusupdates betalingen
- **Beveiliging:** Webhook secret valideren.

1.4 Media-opslag (AWS S3 of Firebase Storage)

- **Gebruik:** Opslag van foto’s/video’s uit sessies.
- **Integratie:**
- Presigned URLs voor directe upload.
- CDN voor snelle levering.

1.5 Notificaties (OneSignal / Firebase Cloud Messaging)

- **Gebruik:** Pushmeldingen en e-mail alerts.
- **Opties:**
- OneSignal API voor multi-platform push.
- SendGrid / Postmark voor e-mail.

---

2. Authenticatie & Autorisatie

- **Protocol:** OAuth 2.0 / JWT.
- **Toegangsniveaus:**
- `user` → toegang tot eigen data.
- `admin` → beheer van platform en content.
- **Keyrotatie:** API-sleutels periodiek vernieuwen.

---

3. Webhooks

- **Stripe:** Betalingsstatussen (paid, failed, refunded).
- **AI Pipeline:** Status van beeldanalyse en interpretatie.
- **Notificaties:** Bevestiging van verzonden meldingen.

---

4. Errorhandling

- **Retry-mechanisme:** Max. 3 pogingen bij API-fout.
- **Logging:** Elke API-call loggen met request/response.
- **Foutcodes:**
- 400 → Ongeldige input
- 401 → Niet geautoriseerd
- 500 → Externe serverfout

---

5. Testen

- Sandbox-omgevingen gebruiken waar mogelijk (Stripe testmode, AI staging).
- Mock API-responses voor E2E-tests.

Opmerking: Alle API-integraties moeten voldoen aan AVG en minimale datavergaring toepassen.