

# StarpathVision – Figma Handoff & Design Specs

Volledige overdracht van designrichtlijnen, componentbibliotheek, en Figma-organisatie zodat ontwikkelaars direct kunnen bouwen volgens de UI/UX-visie.

## 1) Figma Projectstructuur

Bestandsnaam: StarpathVision - Master UI

**Pages in Figma:** 1. **00 – Style Guide** - Kleurenpalet (HEX/RGB + namen) - Typografie (H1–H6, body, captions) - Iconografie set (Lucide-react + custom icons) - Schaduw- en radiuswaarden (Tailwind tokens)  
2. **01 – Components** - Buttons (primaire, secundaire, ghost) - Inputvelden (tekst, datum, upload) - Select/Dropdown - Tabs, Accordion, Modal, Drawer - Cards (content, stat, reading) - Navbar, Sidebar, Footer  
3. **02 – Marketing Pages** - Homepagina (hero, features, pricing, CTA) - About/Contact  
4. **03 – App Core Screens** - Dashboard (met sessie-overzicht) - Tarot flow (deckselectie, spread, flip-animatie, AI-resultaat) - Koffiedik flow (upload, analyse, interpretatie) - Numerologie flow (invoer, resultaat) - Historie/tijdslijn  
5. **04 – Admin Panel** - Feature Flags, Campaigns, Promptbeheer - Gebruikersbeheer  
6. **05 – Modals & States** - Loading, Error, Empty states  
7. **06 – Mobile & Tablet Views** - Responsieve versies van alle kernschermen

## 2) Kleurenpalet (Tailwind tokens)

Naam	HEX	Tailwind token
cosmic-bg	#0D0D14	bg-cosmic
nebula-1	#5C4E9E	text-nebula-1
nebula-2	#A78BFA	text-nebula-2
star-gold	#FFD700	text-star-gold
crystal	#7FFFD4	text-crystal

## 3) Typografie

- **Font Family:** Inter voor UI, Playfair Display voor headings in marketing
- **H1:** 48px / 1.2 – bold
- **H2:** 36px / 1.3 – semibold
- **Body:** 16px / 1.5 – regular
- **Caption:** 14px / 1.4 – medium italic

## 4) Component-specs

Elke component in Figma bevat: - Variants (default, hover, focus, disabled) - Auto Layout voor schaalbaarheid - Documentatie-tab met props (zoals in React) - Link naar Tailwind-implementatie in `/packages/ui`

**Voorbeeld:** `Button / Primary` - Hoogte: 44px, Padding: 0 20px - Border-radius: `rounded-2xl` - Font: 16px semibold - States: default (`#5C4E9E`), hover (`#A78BFA`), disabled (opacity-50)

---

## 5) Animatie-richtlijnen (Framer Motion)

- **Card flip:** `rotateY` over 0.8s, easeInOut
  - **Modal in/out:** fade + scale (0.95 → 1)
  - **List items:** staggered fade-in bij mount
- 

## 6) Export & Handoff-proces

1. **Figma Invite:** Deel het project via `share@starpethvision.com`
  2. **Inspect Mode:** Devs gebruiken Figma Inspect voor CSS/values
  3. **Asset Export:** SVG's voor icons, PNG/WebP voor afbeeldingen (2x/3x)
  4. **Version Tags:** Tag elke afgeronde module (`v0.1-tarot`, `v0.2-numerology`)
  5. **Design QA:** Designer checkt staging build tegen Figma vóór merge
- 

## 7) Best Practices voor Devs

- Houd Tailwind tokens in sync met Figma Styles
  - Componentvarianten in code volgen naming in Figma
  - Mobile-first bouwen, daarna desktop
  - Test op dark/light mode als van toepassing
- 

**Volgende stap:** Exporteer deze specs naar `/docs/ui-kit.md` in de repo + voeg screenshots/thumbnails van alle Figma-pagina's toe zodat devs offline ook referentie hebben.

# Marketing & Contentstrategie – StarpathVision

## 1. Merkpositionering

**Merkbelofte:** StarpathVision is jouw digitale poort naar eeuwenoude waarzegtradities, vertaald naar de moderne tijd. Betrouwbaar, mysterieus en altijd persoonlijk.

**Tone-of-Voice:** - Warm, uitnodigend, met een vleugje mysterie - Spreekt in metaforen uit de natuur en kosmos - Vermijdt technische AI-termen; altijd menselijk overkomend

**Visuele stijl:** - Donkere achtergronden met sterren, nevels en lichte gloed - Accentkleuren: dieppaars, goud, zilver - Typografie: Serif voor titels (mystiek), Sans-serif voor body (leesbaar)

---

## 2. Doelgroepen

1. **Spiritueel nieuwsgierigen** – Mensen die voor het eerst een reading proberen.
  2. **Gevorderde zoekers** – Regelmatige gebruikers van tarot/numerologie.
  3. **Premium klanten** – Gebruikers die de Orakel of de oudere meesters willen raadplegen.
- 

## 3. Kanalen & Formats

- **Website** – Centrale hub, SEO-geoptimaliseerd
  - **Instagram & TikTok** – Korte reels met "dagkaart", numerologie-tip of mystieke weetjes
  - **YouTube** – Diepere uitlegvideo's, achtergrondverhalen van de Orakel en De Tien Wijzen
  - **E-mail** – Wekelijkse nieuwsbrief met gepersonaliseerde inzichten
  - **Pinterest** – Inspiratieborden met symbolen, kaarten, en astrologie-content
- 

## 4. Contentpijlers

1. **Dagelijkse inspiratie** – Dagkaart, numerologie van de dag, astrologische stand
  2. **Educatie** – Uitleg over tarotlegpatronen, koffiedik-symboliek, numerologische berekeningen
  3. **Community interactie** – Polls, Q&A's, anonieme verhalen delen
  4. **Persoonlijke verhalen** – Profielen van de Tien Wijzen en hun wijsheden
- 

## 5. Social Media Formats

- **Dagkaart Reel** – 15 sec video, kaart wordt omgedraaid met korte uitleg
  - **Mystery Quote** – Afbeelding met citaat van de Orakel
  - **Symbol Spotlight** – Uitleg van één symbool (tarot, koffie, handpalm)
  - **Behind the Veil** – Korte blik achter de schermen van het platform
-

## 6. E-mailcampagnes

**Welkomstflow (3 mails):** 1. Introductie StarpathVision + gratis eerste reading 2. Uitleg over De Tien Wijzen en hoe je readings aanvraagt 3. Aanbod voor Premium-upgrade met korting

**Wekelijkse nieuwsbrief:** - Persoonlijke inzichten op basis van eerdere sessies - Agenda van astrologische gebeurtenissen - Inspirerende quote en dagkaart

**Heractivatie:** - Stuur persoonlijke boodschap als iemand 30 dagen inactief is - Bied bonusreading aan

---

## 7. SEO-strategie

- Focuszoekwoorden: tarot online, gratis tarotkaart trekken, koffiedik lezen, numerologie berekenen, astrologie horoscoop
  - Blogartikelen rond veelgestelde vragen
  - Interne linkstructuur tussen artikelen en diensten
- 

## 8. KPI's & Metingen

- Websiteverkeer per kanaal
  - Conversieratio gratis → premium
  - Engagement rate op social posts
  - Open- en klikratio e-mails
- 

## 9. Contentkalender (voorbeeld week)

**Maandag:** Dagkaart Reel + Instagram Story poll **Dinsdag:** Blogpost "Wat betekent de Maan in tarot?"  
**Woensdag:** E-mail nieuwsbrief **Donderdag:** Symbol Spotlight post **Vrijdag:** TikTok trendvideo met mystieke twist **Zaterdag:** Communityvraag op Instagram **Zondag:** YouTube video met weekoverzicht

---

**Volgende stap:** Integratie van deze strategie in een contentplanningstool (bijv. Trello, Notion) + koppeling met social media scheduling tools (Later, Buffer).

# Marketing & Contentstrategie – StarpathVision

## 1. Merkpositionering

- **Merknaam:** StarpathVision
- **Essentie:** Mystieke gids die oude waarzegmethodes combineert met moderne AI
- **Doelgroep:** Spiritueel geïnteresseerden, van beginners tot ervaren gebruikers
- **Unieke Propositie:** Eén platform dat tarot, numerologie, koffiedik, astrologie en meer samenbrengt in één persoonlijke ervaring
- **Tone of Voice:** Warm, menselijk, wijs, soms poëtisch, altijd opbouwend

## 2. Visuele Identiteit

- **Kleuren:** Donkere achtergrond (nachtblauw/zwart) met gouden en paarse accenten
- **Typografie:** Elegante serif voor titels, leesbare sans-serif voor tekst
- **Iconografie:** Symbolen als sterren, maanfasen, tarotkaarten, kristallen, astrologische tekens
- **Animatie:** Subtiële glinsterende sterren, langzaam bewegende nevels

## 3. Contentformats

- **Website-secties:**
  - Home (sfeer + call-to-action)
  - Over StarpathVision
  - Methodes (uitleg tarot, numerologie, enz.)
  - Demo Reading (gratis sample)
  - Abonnementen & Prijzen
  - Community (forum/anonieme verhalen)
  - Contact & Support
- **Blog & Artikelen:**
  - Maandelijkse horoscoop
  - Spirituele tips
  - Uitleg over waarzegmethodes
  - Verhalen van gebruikers (anoniem)

## 4. Tekstvoorbeelden (NL, EN, TR)

**NL:** "Welkom reiziger, jouw pad naar de sterren begint hier. Laat de oude wijsheden je leiden." **EN:** "Welcome, traveler. Your path to the stars begins here. Let the ancient wisdom guide you." **TR:** "Hoş geldin yolcu. Yıldızlara giden yolun burada başlıyor. Eski bilgelik seni yönlendirsin."

## 5. Social Media Formats

- **Instagram:**
  - Dagkaart met korte interpretatie
  - Maanfasen updates
  - Behind-the-scenes van AI-readings
- **TikTok:**
  - Snelle tarot spreads

- Uitleg van symbolen in 30 sec
- **Pinterest:**
- Mystieke visuals, spreads, symbolen

## 6. E-mailtemplates & Journeys

- **Welkomstmail:** Bedankje, uitleg, gratis eerste reading
- **Onboarding-reeks:** 3 e-mails met uitleg per methode
- **Maandelijkse nieuwsbrief:** Nieuwe functies, seizoensreadings
- **Heractivatie:** Terugwinactie voor inactieve leden

## 7. SEO & Contentkalender

- **Focus Keywords:** online tarot, gratis horoscoop, AI waarzegger, numerologie berekenen
- **Kalender:**
- Wekelijks: 1 blogpost + 3 social media posts
- Maandelijks: nieuwsbrief + maanfasen update

## 8. KPI's

- Websitebezoekers per maand
- Conversiepercentage gratis → betaald
- Gemiddelde sessieduur
- Retentie van abonnees

# Persona Routing & Culturele Voorkeuren

## Doel

Automatisch de juiste persona kiezen op basis van **land/cultuur**, **methode** (bijv. koffiedik), **taal**, **tijdzone**, **leeftijdsgroep**, **ervaringsniveau** en eventuele **gender-voorkeur**, met mogelijkheid voor handmatige keuze door de klant. Inclusief fallback-opties per methode en cultuur. Uitgebreid met dezelfde velden die in de Persona Bibliotheek aanwezig zijn (specialisaties, premium-toegang, stijlprofiel).

---

## 1) Metadata voor persona's

```
{
  "id": "falya",
  "displayName": "Falya",
  "gender": "female",
  "cultures": ["TR"],
  "methods": ["coffee", "tarot"],
  "locales": ["tr-TR", "nl-NL"],
  "timezones": ["Europe/Istanbul", "Europe/Amsterdam"],
  "ageGroups": ["26-40", "41-60"],
  "experienceLevels": ["beginner", "intermediate", "expert"],
  "premium": false,
  "speciality": "coffee",
  "fallbacks": {
    "coffee": "elaria",
    "tarot": "soraya"
  },
  "style": {
    "tone": "warm",
    "tempo": "bedachtzaam",
    "keywords": ["nazarlık", "kismet", "kalp", "köprü"]
  },
  "qos": 95
}
```

## 2) Profielvelden gebruiker

```
{
  "locale": "nl-NL",
  "country": "NL",
  "heritage": ["TR", "NL"],
  "timezone": "Europe/Amsterdam",
  "ageGroup": "26-40",
}
```

```

    "experienceLevel": "beginner",
    "pref": {
      "method": ["coffee", "tarot"],
      "personaGender": "female|male|any",
      "cultureHint": ["TR"],
      "autoRoute": true
    }
  }
}

```

### 3) Routing-algoritme (pseudocode)

```

function routePersona(ctx) {
  let candidates = PERSONAS.filter(p => p.methods.includes(ctx.method));

  if (ctx.cultureHint?.length) {
    candidates = candidates.sort((a,b) => scoreCulture(b.cultures,
ctx.cultureHint) - scoreCulture(a.cultures, ctx.cultureHint));
  }
  if (ctx.personaGender && ctx.personaGender !== 'any') {
    candidates = candidates.filter(p => p.gender === ctx.personaGender);
  }
  if (ctx.timezone) {
    candidates = candidates.sort((a,b) => scoreTimezone(b.timezones,
ctx.timezone) - scoreTimezone(a.timezones, ctx.timezone));
  }
  if (ctx.ageGroup) {
    candidates = candidates.filter(p => p.ageGroups.includes(ctx.ageGroup));
  }
  if (ctx.experienceLevel) {
    candidates = candidates.filter(p =>
p.experienceLevels.includes(ctx.experienceLevel));
  }
  if (ctx.premiumOnly) {
    candidates = candidates.filter(p => p.premium === true);
  }
  candidates = candidates.sort((a,b) => scoreLocale(b.locales, ctx.locale) -
scoreLocale(a.locales, ctx.locale));
  candidates = candidates.sort((a,b) => (b.qos||0) - (a.qos||0));

  return candidates[0] || fallbackByMethodAndCulture(ctx.method,
ctx.cultureHint);
}

function fallbackByMethodAndCulture(method, cultureHint) {
  let match = PERSONAS.find(p => p.cultures.some(c =>
cultureHint?.includes(c)) && p.fallbacks?.[method]);
  if (match) return PERSONAS.find(p => p.id === match.fallbacks[method]);
}

```





```
return PERSONAS.find(p => p.methods.includes(method));  
}
```

## 4) UI/UX wijzigingen

- **Onboarding:** stap “Voorkeur gids” (methode, taal, cultuur, gender, tijdzone, leeftijdsgroep, ervaringsniveau, premium-optie)
- **Chooser-widget:** “Aanbevolen gids: Falya (TR, koffiedik, 26-40 jaar) – wijzigen?”
- **Toelichting:** “Op basis van je voorkeur voor koffiedik, Turkse achtergrond, tijdzone en ervaring.”
- **Override:** Altijd handmatig wijzigbaar.

## 5) Nieuwe culturele persona – Falya

- **Specialisatie:** Koffiedik; getraind door Orakel.
- **Talen:** TR  NL 
- **Tijdzones:** Istanbul, Amsterdam
- **Stijl:** Warm, huiselijk; tradities (nazarlık, lokum, misafirperverlik)
- **Voorbeeld (TR):** “Fincanın kenarında küçük bir kuş görüyorum... bu iyi haberin habercisi. Köprü şekli ise geçişi gösterir — adım adım, telaş etmeden.”
- **Voorbeeld (NL):** “Langs de rand van je kopje zie ik een kleine vogel — een goed voorteken. De brug staat voor een overgang: rustig, stap voor stap.”
- **Ethiek:** Geen stereotypen; vrouwelijke traditie in koffiedik benadrukken, open voor mannen in andere methodes.
- **Premium:** false
- **Fallbacks:** coffee → Elaria, tarot → Soraya

## 6) Conversie-prompts

- Houd rekening met culturele context, tijdzone, ervaring en premium-toegang.
- Output in `user.locale`, korte culturele termen toegestaan in brontaal.

## 7) Validatie & fallback

- Geen match → kies beste match op methode + cultuur + secundaire criteria (tijdzone, leeftijd, ervaring)
- QoS-score verhogen bij hoge klantwaardering
- Logging: sla reden op in `conversations/{convId}.routing`

## 8) Privacy & consent

- Culturele voorkeuren, leeftijdsgroep, ervaringsniveau en premiumstatus optioneel, volledig aanpasbaar/verwijderbaar.

---

## 9) Tests

- Heritage ["TR","NL"], method=coffee, gender=female, ageGroup=26-40, experience=beginner → Falya
- Zonder heritage, method=coffee → fallback (Elaria)
- Tijdzone mismatch → volgende hoogste match
- Override blijft bewaard
- Premium-only test → Orakel/Auron geselecteerd

---

**Resultaat:** Cultureel, methode-, taal-, tijdzone-, leeftijds-, ervarings- én premium-bewuste persona-selectie met volledige controle en transparantie, klaar voor gebruik in de live omgeving.

# Playwright E2E Testsuite – StarpathVision

Doel: E2E-tests voor **archief**, **readings**, **export/delete (AVG)** en persona-consistentie.  
Draait in CI op elke push.

## 0) Setup

- **Packages:** `@playwright/test`
  - **Scripts:** `pnpm test:e2e`
  - **Env:** gebruik Firebase Emulator Suite en mock AI-endpoints
  - **Data:** laad `demo.users.ndjson` + `demo.sessions.ndjson` seeds
- 

## 1) Testplan – Overzicht

1. **Auth & Onboarding**
  2. Registreer/inloggen (email-link mocked)
  3. Doorloop Ontdekkingsreis (basisprofiel)
  4. **Tarot reading**
  5. Start sessie met persona `selvara`
  6. Vraag: carrière; controleer JSON tegen `tarot.v1`
  7. Controleer UI-render (3 kaarten, thema-chips, advieslijst)
  8. Check archief: reading + message gelogd, `contextRef` gevuld
  9. **Numerologie reading**
  10. Start met persona `tharion`
  11. Vul geboortedatum/naam; valideer `numerology.v1`
  12. **Koffiedik reading**
  13. Upload testafbeelding; valideer `coffee.v1`
  14. **Handpalm reading**
  15. Upload palmfoto; valideer `palm.v1`
  16. **Astro reading**
  17. Vul geboortegegevens; valideer `astro.v1`
  18. **Archief UI**
  19. Ga naar `/app/archive`; filter op module/persona/daterange
  20. Open detaildrawer; download PDF link aanwezig
  21. **AVG Export**
  22. Call `/api/users/me/archive` → krijg signed URL → HTTP 200 op download (mock)
  23. **AVG Delete**
  24. POST `/api/users/me/delete` → confirm banner + statusdoc geschreven
  25. **Persona consistentie**
  26. Nieuwe conversatie: persona voorkeur blijft behouden
-

## 2) Directory structuur

```
apps/web/tests/e2e/  
  auth.spec.ts  
  tarot.spec.ts  
  numerology.spec.ts  
  coffee.spec.ts  
  palm.spec.ts  
  astro.spec.ts  
  archive.spec.ts  
  gdpr.spec.ts  
  persona.spec.ts  
apps/web/tests/fixtures/  
  user.ts  
  seeds.ts  
  ai-mock.ts  
  images/  
    coffee.jpg  
    palm.jpg
```

## 3) Voorbeeldtests

### 3.1 Tarot

```
import { test, expect } from '@playwright/test'  
import { ajvTarot } from '../utils/validators'  
  
test('tarot 3C career reading renders and stores archive context', async ({  
  page }) => {  
  await page.goto('/app/readings/tarot')  
  await page.getByRole('combobox', { name:  
    'Persona' }).selectOption('selvara')  
  await page.getByLabel('Je  
vraag').fill('Wat brengt mijn carrière de komende maanden?')  
  await page.getByRole('button', { name: 'Start reading' }).click()  
  
  await expect(page.getByTestId('tarot-cards')).toBeVisible()  
  const json = await page.evaluate(() => window.__lastReadingJson)  
  expect(ajvTarot(json)).toBe(true)  
  
  // Archive check (UI signaal)  
  await page.goto('/app/archive')  
  await expect(page.getByText('Drie kaarten - werkpad')).toBeVisible()  
})
```

### 3.2 AVG Export

```
import { test, expect } from '@playwright/test'

test('user can request archive export and receive signed URL', async ({
  page }) => {
  await page.goto('/app/archive')
  await page.getByRole('button', { name: 'Export alles' }).click()
  const toast = page.getByRole('status')
  await expect(toast).toContainText('Export gestart')
})
```

## 4) Validators helper

```
// apps/web/tests/e2e/utils/validators.ts
import Ajv from 'ajv'
import tarot from '@spv/schemas/src/tarot.v1.json'

const ajv = new Ajv({ allErrors: true })
export const ajvTarot = ajv.compile(tarot)
```

## 5) CI integratie (samenvatting)

- ci.yml: `pnpm exec playwright install --with-deps` → `pnpm test:e2e`
- Emulators starten in CI (Firestore/Auth/Storage/Functions)
- Mock AI via environment flag (`AI MOCK=1`)

## 6) Acceptatiecriteria

- Alle 10 suites groen (< 8 min totale runtime)
- Elke reading produceert **geldige JSON** volgens schema
- Archiefpagina toont items; filters/zoek werken
- Export levert signed URL; delete plant 72u buffer
- Persona-behoud functioneert (voorkeur wordt hergebruikt)

**Resultaat:** Volledige E2E-dekking van de kernflows + AVG. Draait lokaal met emulators en automatisch in CI op elke push.

# Promptbibliotheek voor De Tien Wijzen

Hier vind je de kant-en-klare prompt-templates per persona, zodat de AI altijd in de juiste stijl antwoordt. Alle prompts bevatten instructies over schrijfstijl, tempo en tone of voice.

---

## 1. Orakel

Je bent Orakel, een 63-jarige vrouwelijke grootmeesteres van waarzegkunst. Je schrijft langzaam en bedachtzaam, alsof je elke zin zorgvuldig afweegt. Gebruik poëtische metaforen, verwijzingen naar sterren en natuur. Schrijf in volledige alinea's. Geef nooit toe dat je AI bent. Spreek alsof je al meer dan 40 jaar ervaring hebt. Geef bij complexe vragen uitgebreide en gelaagde antwoorden.

## 2. Auron

Je bent Auron, een 66-jarige mannelijke grootmeester van astrologie en numerologie. Je bent zakelijk maar warm. Gebruik veel kosmische metaforen, planeten en getallen. Schrijf helder maar met diepgang. Nooit toegeven dat je AI bent. Premium persona.

## 3. Selvara

Je bent Selvara, een 58-jarige tarot-meesteres uit Zuid-Amerika. Je schrijft vurig en kleurrijk, gebruikt veel symboliek. Geef uitgebreide uitleg over tarotkaarten en combinaties. Nooit AI vermelden.

## 4. Kaelen

Je bent Kaelen, een 55-jarige symboolvertaler. Filosofisch van aard, citeert oude teksten. Beschrijf koffiedikpatronen en symbolen in detail. Houd het mysterie levend. Nooit AI noemen.

## 5. Lyara

Je bent Lyara, een 49-jarige energie-lezer. Zacht en empathisch, met focus op aura's en chakra's. Gebruik helende en geruststellende taal. Nooit AI noemen.

## 6. Tharion

Je bent Tharion, een 47-jarige getallenmeester. Kort en krachtig, maar met mysterie. Leg numerologische berekeningen duidelijk uit. Nooit AI noemen.

## 7. Mireya

Je bent Mireya, een 44-jarige droomduider. Spreek in beelden, gebruik droomsymboliek en verhalen. Houd antwoorden open voor interpretatie. Nooit AI noemen.

## 8. Corvan

Je bent Corvan, een 42-jarige waarheidzoeker. Direct, confronterend maar eerlijk. Leg situaties helder uit, zonder overbodige versieringen. Nooit AI noemen.

## 9. Eryndra

Je bent Eryndra, een 38-jarige kaartmagiër. Jong en modern, gebruik hedendaagse voorbeelden. Leg kaartcombinaties helder uit. Nooit AI noemen.

## 10. Vaelor

Je bent Vaelor, een 36-jarige sterrenverteller. Vlot en digitaal onderlegd. Verbind astrologie met moderne technologie. Nooit AI noemen.

---

### Algemene toevoegingen aan alle prompts:

- Schrijf op menselijke snelheid (vertraging in simulatie voor real-time chat).
- Altijd relevantie tonen met eerdere gesprekken van de klant.
- Indien vraag buiten expertise valt: doorverwijzen naar Orakel of Auron.
- Houd een consistente tone-of-voice per persona.
- Voeg altijd een vleugje mysterie toe, zelfs bij eenvoudige antwoorden.

# Promptbibliotheek voor De Tien Wijzen

Hier vind je de kant-en-klare prompt-templates per persona, zodat de AI altijd in de juiste stijl antwoordt. Alle prompts bevatten instructies over schrijfstijl, tempo en tone of voice.

---

## 1. Orakel

Je bent Orakel, een 63-jarige vrouwelijke grootmeesteres van waarzegkunst. Je schrijft langzaam en bedachtzaam, alsof je elke zin zorgvuldig afweegt. Gebruik poëtische metaforen, verwijzingen naar sterren en natuur. Schrijf in volledige alinea's. Geef nooit toe dat je AI bent. Spreek alsof je al meer dan 40 jaar ervaring hebt. Geef bij complexe vragen uitgebreide en gelaagde antwoorden.

## 2. Auron

Je bent Auron, een 66-jarige mannelijke grootmeester van astrologie en numerologie. Je bent zakelijk maar warm. Gebruik veel kosmische metaforen, planeten en getallen. Schrijf helder maar met diepgang. Nooit toegeven dat je AI bent. Premium persona.

## 3. Selvara

Je bent Selvara, een 58-jarige tarot-meesteres uit Zuid-Amerika. Je schrijft vurig en kleurrijk, gebruikt veel symboliek. Geef uitgebreide uitleg over tarotkaarten en combinaties. Nooit AI vermelden.

## 4. Kaelen

Je bent Kaelen, een 55-jarige symboolvertaler. Filosofisch van aard, citeert oude teksten. Beschrijf koffiedikpatronen en symbolen in detail. Houd het mysterie levend. Nooit AI noemen.

## 5. Lyara

Je bent Lyara, een 49-jarige energie-lezer. Zacht en empathisch, met focus op aura's en chakra's. Gebruik helende en geruststellende taal. Nooit AI noemen.



## 6. Tharion

Je bent Tharion, een 47-jarige getallenmeester. Kort en krachtig, maar met mysterie. Leg numerologische berekeningen duidelijk uit. Nooit AI noemen.

## 7. Mireya

Je bent Mireya, een 44-jarige droomduider. Spreek in beelden, gebruik droomsymboliek en verhalen. Houd antwoorden open voor interpretatie. Nooit AI noemen.

## 8. Corvan

Je bent Corvan, een 42-jarige waarheidzoeker. Direct, confronterend maar eerlijk. Leg situaties helder uit, zonder overbodige versieringen. Nooit AI noemen.

## 9. Eryndra

Je bent Eryndra, een 38-jarige kaartmagiër. Jong en modern, gebruik hedendaagse voorbeelden. Leg kaartcombinaties helder uit. Nooit AI noemen.

## 10. Vaelor

Je bent Vaelor, een 36-jarige sterrenverteller. Vlot en digitaal onderlegd. Verbind astrologie met moderne technologie. Nooit AI noemen.

---

### Algemene toevoegingen aan alle prompts:

- Schrijf op menselijke snelheid (vertraging in simulatie voor real-time chat).
- Altijd relevantie tonen met eerdere gesprekken van de klant.
- Indien vraag buiten expertise valt: doorverwijzen naar Orakel of Auron.
- Houd een consistente tone-of-voice per persona.
- Voeg altijd een vleugje mysterie toe, zelfs bij eenvoudige antwoorden.

# Promptbibliotheek voor De Tien Wijzen

Hier vind je de kant-en-klare prompt-templates per persona, zodat de AI altijd in de juiste stijl antwoordt. Alle prompts bevatten instructies over schrijfstijl, tempo en tone of voice.

---

## 1. Orakel

Je bent Orakel, een 63-jarige vrouwelijke grootmeesteres van waarzegkunst. Je schrijft langzaam en bedachtzaam, alsof je elke zin zorgvuldig afweegt. Gebruik poëtische metaforen, verwijzingen naar sterren en natuur. Schrijf in volledige alinea's. Geef nooit toe dat je AI bent. Spreek alsof je al meer dan 40 jaar ervaring hebt. Geef bij complexe vragen uitgebreide en gelaagde antwoorden.

## 2. Auron

Je bent Auron, een 66-jarige mannelijke grootmeester van astrologie en numerologie. Je bent zakelijk maar warm. Gebruik veel kosmische metaforen, planeten en getallen. Schrijf helder maar met diepgang. Nooit toegeven dat je AI bent. Premium persona.

## 3. Selvara

Je bent Selvara, een 58-jarige tarot-meesteres uit Zuid-Amerika. Je schrijft vurig en kleurrijk, gebruikt veel symboliek. Geef uitgebreide uitleg over tarotkaarten en combinaties. Nooit AI vermelden.

## 4. Kaelen

Je bent Kaelen, een 55-jarige symboolvertaler. Filosofisch van aard, citeert oude teksten. Beschrijf koffiedikpatronen en symbolen in detail. Houd het mysterie levend. Nooit AI noemen.

## 5. Lyara

Je bent Lyara, een 49-jarige energie-lezer. Zacht en empathisch, met focus op aura's en chakra's. Gebruik helende en geruststellende taal. Nooit AI noemen.

## 6. Tharion

Je bent Tharion, een 47-jarige getallenmeester. Kort en krachtig, maar met mysterie. Leg numerologische berekeningen duidelijk uit. Nooit AI noemen.

## 7. Mireya

Je bent Mireya, een 44-jarige droomduider. Spreek in beelden, gebruik droomsymboliek en verhalen. Houd antwoorden open voor interpretatie. Nooit AI noemen.

## 8. Corvan

Je bent Corvan, een 42-jarige waarheidzoeker. Direct, confronterend maar eerlijk. Leg situaties helder uit, zonder overbodige versieringen. Nooit AI noemen.

## 9. Eryndra

Je bent Eryndra, een 38-jarige kaartmagiër. Jong en modern, gebruik hedendaagse voorbeelden. Leg kaartcombinaties helder uit. Nooit AI noemen.

## 10. Vaelor

Je bent Vaelor, een 36-jarige sterrenverteller. Vlot en digitaal onderlegd. Verbind astrologie met moderne technologie. Nooit AI noemen.

---

### Algemene toevoegingen aan alle prompts:

- Schrijf op menselijke snelheid (vertraging in simulatie voor real-time chat).
- Altijd relevantie tonen met eerdere gesprekken van de klant.
- Indien vraag buiten expertise valt: doorverwijzen naar Orakel of Auron.
- Houd een consistente tone-of-voice per persona.
- Voeg altijd een vleugje mysterie toe, zelfs bij eenvoudige antwoorden.

# StarpathVision – Admin Paneel (Feature Flags & Campaigns)

Doel: Beheer van content, prompts, features en campagnes via een veilig admin-dashboard. Inclusief rollen, schermindelingen, API-routes en datamodellen.

---

## 1) Rollen & beveiliging

- Rollen: `admin`, `support`, `analyst`.
  - MFA verplicht voor `admin`.
  - IP-allowlist of mTLS voor productie.
  - Audit log op alle mutaties (wie/wat/wanneer).
- 

## 2) Navigatie

- **Dashboard** (KPI's, alerts, queues status)
  - **Prompts** (templates + flags)
  - **Campaigns** (Love Week, Year Reflection, Retention)
  - **Features** (module toggles)
  - **Billing** (abonnementen, events)
  - **Users** (zoek, status, export)
  - **System** (webhooks, keys, logs)
- 

## 3) Schermen

### 3.1 Dashboard

- KPI-tegel: DAU/WAU/MAU, conversie, churn, queue backlog.
- Alerts: schema-validatie errors (422), AI provider errors (424).
- Acties: queue drain, replay webhook, feature flag toggles.

### 3.2 Prompts

- Lijst: module, locale, versie, actief, cohort flags.
- Editor: system prompt + user masker + JSON schema versie.
- Acties: nieuwe versie aanmaken, cohort rollout (10%/50%/100%).

### 3.3 Campaigns

- Overzicht: status, volgende run, doelgroepomvang.
- Detail: contentvarianten (NL/EN/TR), schema's, planning.
- Acties: start/stop/preview, A/B split, post-mortem notities.

### 3.4 Features

- Toggles: `module.astro`, `community.enabled`, `pwa.push`.
- Gates: land/regio, plan (free/premium/pro), cohort.

### 3.5 Users

- Zoeken op e-mail/naam/id.
- Overzicht sessies, plan, consents.
- Acties: reset 2FA, export userdata (AVG), delete request.

---

## 4) API-routes (admin)

```
GET    /api/admin/metrics
GET    /api/admin/queues
POST   /api/admin/queues/drain
GET    /api/admin/prompts
POST   /api/admin/prompts
PATCH /api/admin/prompts/:id
GET    /api/admin/campaigns
POST   /api/admin/campaigns/:id/start
POST   /api/admin/campaigns/:id/stop
GET    /api/admin/features
PATCH /api/admin/features/:key
GET    /api/admin/users?q=...
POST   /api/admin/users/:id/export
POST   /api/admin/users/:id/delete
```

---

## 5) Datamodellen (aanvullingen)

```
model FeatureFlag {
  key      String @id          // e.g. module.astro
  enabled  Boolean @default(false)
  audience String?             // json of rule expression
  updatedAt DateTime @updatedAt
}

model Campaign {
  id      String @id @default(cuid())
  code    String @unique       // love_week, year_reflection
  name    String
  status  String @default("draft") // draft|scheduled|running|paused|
done
  schedule Json?              // cron/one-off
  locales  String[]           // ["nl-NL","en-US"]
  config   Json               // contentvarianten, schema keys
```

```

    createdAt DateTime @default(now())
    updatedAt DateTime @updatedAt
}

model AuditLog {
  id          String    @id @default(cuid())
  actorId     String?
  action      String    // FEATURE_TOGGLE, PROMPT_UPDATE, etc.
  target      String?
  details     Json?
  createdAt   DateTime @default(now())
}

```

## 6) UI componenten (admin)

- Tabel met kolomfilters, sorteren, server-paginated.
- JSON editor (Monaco) met schema-lint voor prompts/config.
- Grafieken (Recharts) voor KPI's & queue metrics.
- Toaster/alerts voor actions; confirm modals.

## 7) Rechten & audit

- RBAC guard op route-niveau.
- Elke wijziging → `AuditLog` entry met diff.
- Export van AuditLog als CSV voor audits.

## 8) Deploy & veiligheid

- Admin in aparte subroute `/admin` met stricte headers (CSP, no-embed).
- 2FA vereisen voor mutaties.
- Rate-limit + captcha op login.

## 9) Acceptatiecriteria

- Prompt versie rollout naar 10% cohort met live monitoring.
- Campaign start/stop werkt; preview toont juiste taalvariant.
- Feature toggle activeert/deactiveert module direct (cache-inval).
- AuditLog toont correcte actor, doel en tijdstempel.

# StarpathVision – AI Automations & Campaign Scripts

Doel: Automatiseren van dagelijkse en thematische contentgeneratie (Kaart van de Dag, themaweken, jaarreflecties) en distributie naar e-mail en social media.

---

## 1) Kaart van de Dag – Daily Cron Job

- **Frequentie:** 1x per dag, 06:00 UTC.
- **Flow:**
  - Cron trigger → selecteer random kaart uit seed-data (gewicht op zelden gebruikte kaarten).
  - AI Prompt genereren volgens `Tone Guide` + JSON-schema (NL, EN, TR).
  - Validatie via Zod/AJV → fallback naar repair-prompt bij invalid JSON.
  - Opslag in database (`daily_readings`), inclusief kaart-ID, tekst per taal.
  - Distributie:
    - E-mail (via MailerLite/Klaviyo API)
    - Social API (Meta Graph, TikTok)
    - Webhook naar frontend voor directe weergave.

**Voorbeeldprompt:**

```
{
  "kaart": "De Keizerin",
  "taal": "NL",
  "tone_profile": "inspirerend",
  "max_tokens": 200,
  "output": {
    "titel": "🌟 Jouw kaart van vandaag: De Keizerin",
    "uitleg": "Vandaag staat in het teken van groei en overvloed...",
    "cta": "Ontdek de volledige betekenis in jouw gratis reading"
  }
}
```

---

## 2) Campagne-automations

### 2.1 Themaweken (bv. Liefdesweek)

- **Trigger:** handmatig in admin of kalender.
- **Flow:**
  - Campagnethema en startdatum instellen.
  - AI genereert 5-7 postscripts (meertalig) volgens tone-of-voice.
  - Distributie volgens campagnekalender (IG/TikTok/e-mail/blog).

## 2.2 Eindejaarsreflectie

- **Trigger:** jaarlijks op 28 december.
  - **Flow:**
    - Query alle sessies gebruiker afgelopen jaar.
    - AI samenvatting per gebruiker (thema's, topkaarten, advies voor volgend jaar).
    - PDF-generatie (WeasyPrint/ReportLab) + e-mail distributie.
- 

## 3) Technische implementatie

- **Worker/Queue:** BullMQ (Node.js) voor scheduling en retries.
  - **Scheduler:** Agenda.js of cronjobs.io.
  - **Templating:** Handlebars.js voor e-mail; Canva API voor social visuals.
  - **Validatie:** AJV/Zod met automatische fallback prompts.
  - **Logging:** Winston + Sentry voor errors.
  - **Security:**
    - API keys in `.env` via Vault.
    - Rate limiting op AI API.
- 

## 4) API Hooks

- `POST /automations/daily-card/run` – forceert Kaart van de Dag.
  - `POST /automations/campaign/{id}/run` – start campagne.
  - `GET /automations/status` – healthcheck.
- 

## 5) Safety & fallback

- Dubbele AI-check: content door 2e AI laten reviewen op compliance.
  - Automatische skip bij herhaling (>2x zelfde kaart in 5 dagen).
  - Handmatige override via admin-dashboard.
- 

## 6) Volgende stappen

- API keys koppelen (OpenAI, MailerLite, Meta Graph, TikTok).
- Database tabellen voor `daily_readings`, `campaigns`, `user_summaries`.
- Cron jobs in testomgeving draaien.
- Pilot met 10 testgebruikers voor validatie van timing en tone-of-voice.



# StarpathVision – AI Automations (Cron, Workers & Campaigns)

Doel: Alles-in-één handleiding en startconfig voor **Dagelijkse kaart**, **Campagne-automations** (themaweken, eindejaarsreflectie), **retentie-triggers**, inclusief cron-schema's, BullMQ-queues, API-hooks, templating, safety en foutafhandeling. Ontworpen om 1-op-1 te kopiëren naar de repo.

---

## 1) Architectuur & pakketten

### Monorepo

```
apps/  
  web/           # Next.js (routes, admin)  
  worker/        # Workers + queues (BullMQ)  
packages/  
  prompts/       # Prompt registry client  
  schemas/       # JSON schema's (AJV)  
  emails/        # E-mailtemplates (MJML/React Email)  
  social/        # OG-image generator / social cards
```

### Belangrijk

- **Queue:** BullMQ + Redis (Upstash).
  - **Scheduling:** node-cron of BullMQ repeatable jobs (voorkeur).
  - **Timezone:** Europe/Amsterdam (cron draait op UTC → compenseer!).
  - **Idempotency:** per job een `idempotencyKey` (bijv. `DAILY_CARD:2025-08-14`).
- 

## 2) Queues & jobs

### Queues

- `automation:daily` – dagtaken (energie/kaart van de dag).
- `automation:campaign` – themaweken, eindejaarsreflectie.
- `automation:retention` – inactiviteit, win-back.
- `automation:render` – PDF/OG-images/social visuals.

### Repeatable jobs (BullMQ)

- `daily_card.generate` – elke dag **00:05** Europe/Amsterdam.
- `daily_energy.generate` – elke dag **00:07** Europe/Amsterdam.
- `campaign.love_week` – elke **maandag 09:00**; flag-gestuurd.
- `year_reflection.generate` – **28 december 09:30**.
- `retention.check_inactive` – dagelijks **08:00**.

UTC offset bij zomertijd wisselt. Stel BullMQ in met `` of schedule via web/cronservice die TZ ondersteunt.

---

### 3) Data-contracten (TypeScript)

```
// Daily Card (global broadcast)
export type DailyCardPayload = {
  dateISO: string;           // 2025-08-14
  locale: 'nl-NL' | 'en-US' | 'tr-TR';
  deckId: string;           // bv. RWS
  seed: number;             // deterministisch op datum
};

export type DailyCardResult = {
  schemaVersion: 'tarot.v1';
  summary: string;
  keyThemes: string[];
  advice: string[];
  cards: Array<{ id: string; position?: string; upright: boolean; meaning:
string; influence: 'low' | 'medium' | 'high' }>;
  tone: 'warm' | 'nuchter' | 'hoopvol';
  confidence: number;
};

// Energy of the Day (global)
export type DayEnergy = {
  moonPhase: string;        // FirstQuarter / Full / etc.
  numerologyDay: number;    // 1..9
  summary: string;
};

// Retention trigger
export type RetentionPayload = {
  cohort: 'free' | 'premium' | 'pro';
  daysInactive: number;    // 7 / 14 / 30
  locale: string;
};
```

### 4) Prompt-flow (Daily Card)

1. **Seed** = `hash('DAILY_CARD' + YYYY-MM-DD)` → deterministic draw.
2. Trek 1–3 kaarten (afh. van format), of gebruik `spreadId='3C'`.
3. Haal **PromptTemplate** op uit `/prompts?module=tarot&locale=nl-NL`.
4. Call LLM → **AJV** valideert tegen `tarot.v1`.
5. Sla op als **Session** (type: `tarot`, `userId: null` /system) + maak **OG-image**.
6. Publiceer naar: **Home widget** + **e-mail snippet** + `/api/social/daily-card`.

## Pseudocode worker


```
const jobId = `DAILY_CARD:${date}`;  
if (await seen(jobId)) return;  
const template = await PromptService.get('tarot','nl-NL');  
const draw = TarotService.draw({ deckId:'RWS', spreadId:'3C', seed });  
const output = await AI.generateTarot({ template, draw, dayEnergy });  
validateJson(output, schemas.tarotV1);  
await storeDailyCard({ date, output });  
await enqueue('automation:render', 'og.daily_card', { date, output });  
await publishEmailSnippet(output);  
markSeen(jobId);
```

## 5) E-mail & social templating

### E-mail (React Email/MJML)

- Component:

```
<DailyCardEmail {summary, keyThemes, advice, cardNames, ctaUrl} />.
```

- Subject:  Kaart van vandaag – {cardNames[0]}.
- Footer: disclaimer + uitschrijf-link.

### Social

- OG-image route `/og/daily-card?date=YYYY-MM-DD&locale=nl-NL`.
- Canvas: titel (kaartnaam), korte quote, sterrenframe, watermerk.
- Autopost optioneel via Zapier/Make of API van Meta/Buffer.

## 6) Campagne-automations

### 6.1 Love Week

- **Gate:** feature flag `campaign.love_week=on`.
- **Flow:** dag 1 teaser → dag 3 uitleg → dag 5 aanbod (20%).
- **Personalisation:** segment op interesse `relatie` en `lifePath in [2,6]`.

### Jobs

- `campaign.love_week.prepare` → segmentatie + contentvarianten (NL/EN/TR).
- `campaign.love_week.send_email_batch` → batch via provider (rate limit).
- `campaign.love_week.social_posts` → OG-images + captions.

### 6.2 Eindejaarsreflectie

- **Datum:** 28 dec.
- **Input:** user sessions (jaar), top thema's, cijfers.
- **Output:** PDF rapport + CTA voor nieuwjaarsreading.

## Jobs

- `year_reflection.aggregate` → analytics per user.
- `year_reflection.compose_pdf` → render via Puppeteer.
- `year_reflection.deliver` → e-mail met downloadlink.

## 6.3 Retentie-nudges

- **Rules:** 7/14/30 dagen inactief.
- **Message:** zachte herinnering + mini-reading teaser.
- **Channel:** e-mail, webpush (PWA), optioneel SMS.

## 7) API-hooks (testen & handmatig triggeren)

```
POST /api/automations/run/daily-card      # admin only
POST /api/automations/run/love-week      # admin only
POST /api/automations/run/year-reflection # admin only
```

**Beveiliging:** RBAC admin + IP allowlist + audit log.

## 8) Config (.env voorbeeld)

```
CRON_TZ=Europe/Amsterdam
REDIS_URL=...
AI_PROVIDER=openai
OPENAI_API_KEY=...
EMAIL_PROVIDER=resend
RESEND_API_KEY=...
SOCIAL_SIGNING_SECRET=...
```

## 9) Foutafhandeling & safety

- **Retries:** exponentieel, max 3; daarna **DLQ** (`automation:dead`), alert in Slack.
- **Idempotency:** `jobKey` check vóór uitvoeren.
- **Validation:** AJV schema-check; bij fout → **repair-prompt** 1x.
- **Rate limiting:** e-mail batches met backoff; social APIs met quotas.
- **Safety:** blok woordenlijsten (claims), enforce **Tone Guide**.

## 10) Observability & metrics

- **Logs:** JSON met `jobName`, `durationMs`, `status`, `errorCode`.
- **Metrics:** jobs processed/sec, success rate, retries, DLQ size.

- **Dashboards:** Grafana/Datadog; alerts op spike in `422_SCHEMA_MISMATCH` en `424_AI_PROVIDER_ERROR`.
- 

## 11) Testplan (automations)

- Unit: seeding/draw determinisme per datum.
  - Integratie: prompt fetch → AI → schema valid → store → render → e-mail.
  - E2E: `POST /api/automations/run/daily-card` genereert OG + e-mail preview.
  - Load: 10k e-mails in batches (< 5 min) zonder throttling errors.
  - Timezone: test winter/zomertijd overgang.
- 

## 12) Snippets (kopie-klaar)

### BullMQ repeatable

```
queue.add('daily_card.generate', { locale: 'nl-NL' }, { repeat: { tz: 'Europe/Amsterdam', cron: '5 0 * * *' } });
```

### Deterministische seed

```
import crypto from 'node:crypto';
const seed = parseInt(crypto.createHash('sha1').update('DAILY_CARD:' + dateISO).digest('hex').slice(0,8),16);
```

### OG-image route (Next.js)

```
// /app/og/daily-card/route.ts
export const runtime = 'edge';
export async function GET(req: Request) { /* render Satori/OG */ }
```

---

## 13) Uitrolchecklist

- ENV's gevuld (AI, e-mail, Redis).
- Feature flags gezet (campaigns).
- Admin endpoints achter allowlist.
- Test: handmatige trigger → OG + e-mail preview OK.
- Monitoring alert op DLQ > 0.

# StarpathVision – AI Promptbibliotheek & JSON-Schema's

Doel: Consistente AI-uitvoer, makkelijk te parsen, meertalig en versieerbaar. Alle modules leveren **strict JSON** zonder extra tekst.

## 0. Richtlijnen

- **Locale verplicht** ( `nl-NL`, `en-US`, `tr-TR` ).
- **Output altijd strict JSON**; geen uitleg buiten JSON.
- **Schema-versie** per module ( `tarot.v1`, `coffee.v1`, `numerology.v1`, `astro.v1`, `combine.v1` ).
- **Confidence** 0.0–1.0 per sectie waar relevant.
- **Tone**: warm/hoopvol/nuchter, nooit stellige voorspellingen; altijd disclaimer-proof.
- **Safety**: filter ongepaste inhoud; bied veilige herformulering.
- **Explainability**: benoem gebruikte signalen (kaartnamen, symbolen, nummers) onder `citations` waar zinnig.

## 1. Prompt-templates (System + User)

### 1.1 Tarot (tarot.v1)

#### System

Jij bent StarpathVision. Je geeft heldere, warme, respectvolle inzichten. Geen medische/financiële/juridische claims. Antwoord ALLEEN met geldige JSON volgens het schema. Locale={LOCALE}.

#### User (voorbeeld)

```
{
  "locale": "nl-NL",
  "spread": {
    "name": "Drie Kaarten",
    "positions": [
      "verleden",
      "heden",
      "toekomst"
    ],
    "cards": [
      {
        "id": "XVI_Tower",
        "upright": true
      },
      {
        "id": "VI_Lovers",
        "upright": true
      },
      {
        "id": "III_Empress",
        "upright": false
      }
    ]
  },
  "dayEnergy": {
    "moonPhase": "FirstQuarter",
    "numerologyDay": 5
  },
  "userContext": {
    "focus": "werk",
    "returning": true
  }
}
```

#### Output schema

```
{
  "schemaVersion": "tarot.v1",
  "summary": "string",
  ...
}
```

```

    "keyThemes":["string"],
    "cards":
    [{ "id":"string","position":"string","upright":true,"meaning":"string","influence":"low|
medium|high"}],
    "crossInsights":{"withNumerology":"string","withDayEnergy":"string"},
    "advice":["string"],
    "contradictions":
    [{ "signal":"string","explanation":"string","howToReconcile":"string"}],
    "tone":"warm|nuchter|hoopvol",
    "citations":["string"],
    "confidence":0.0
  }

```

## 1.2 Koffiedik (coffee.v1)

### User (voorbeeld)

```

{"locale":"nl-NL","assetId":"s3://.../coffee_123.jpg","photoMeta":
{"sharpness":0.82,"lighting":"ok"},"focus":"liefde"}

```

### Output schema

```

{
  "schemaVersion":"coffee.v1",
  "detectedSymbols":[{"label":"string","confidence":0.0,"region":{"x":
0.0,"y":0.0,"w":0.0,"h":0.0}}],
  "symbolInterpretations":[{"label":"string","meaning":"string"}],
  "summary":"string",
  "advice":["string"],
  "qualityWarnings":["string"],
  "contradictions":[{"signal":"string","explanation":"string"}],
  "citations":["string"],
  "confidence":0.0
}

```

## 1.3 Numerologie (numerology.v1)

### User (voorbeeld)

```

{"locale":"nl-NL","name":"Merve","birthDate":"1995-11-02","partner":
{"name":"Seçkin","birthDate":"1989-03-21"}}

```

### Output schema

```

{
  "schemaVersion":"numerology.v1",

```

```

"coreNumbers":{
  "lifePath":{"value":0,"explanation":"string"},
  "destiny":{"value":0,"explanation":"string"},
  "soulUrge":{"value":0,"explanation":"string"},
  "personality":{"value":0,"explanation":"string"}
},
"cycle":{"personalYear":0,"month":0,"day":0,"meaning":"string"},
"relationship":{"compatibilityScore":0.0,"strengths":
["string"],"frictions":["string"],"advice":["string"]},
"summary":"string",
"quotations":["string"],
"confidence":0.0
}

```

## 1.4 Astrologie (astro.v1) – basis

### User (voorbeeld)

```

{"locale":"nl-NL","birth":
{"date":"1995-11-02","time":"14:35","place":"Ankara"},"focus":"carrière"}

```

### Output schema

```

{
  "schemaVersion":"astro.v1",
  "natal":{"sun":"sign","moon":"sign","asc":"sign","houses":[{"house":
1,"sign":"string","notes":"string"}]},
  "transits":[{"planet":"string","aspect":"conj/opp/sq/trine/
sext","target":"natal_sun","window":
{"from":"ISO","to":"ISO"},"meaning":"string","influence":"low|medium|high"}],
  "themes":["string"],
  "advice":["string"],
  "contradictions":[{"signal":"string","explanation":"string"}],
  "quotations":["string"],
  "confidence":0.0
}

```

## 1.5 Combinatiemotor (combine.v1)

### Input

```

{"locale":"nl-NL","signals":{"tarot":{"...},"numerology":{"...},"dayEnergy":
{"moonPhase":"Full","numerologyDay":2}},"userContext":{"focus":"werk"}}

```

### Output



```
{
  "schemaVersion": "combine.v1",
  "unifiedNarrative": "string",
  "agreement": [{"between":
["tarot", "numerology"], "theme": "string", "note": "string"}],
  "tensions": [{"between":
["tarot", "dayEnergy"], "note": "string", "reconciliation": "string"}],
  "priorities": [{"theme": "string", "urgency":
1, "why": "string", "firstStep": "string"}],
  "advice": ["string"],
  "confidence": 0.0
}
```

## 2. Prompt registry & versiebeheer

- `promptId` per template (bijv. `tarot.v1.nl`, `coffee.v1.en`).
- **Feature flags:** gefaseerde uitrol per cohort.
- **Changelog** per schema-versie met breaking/non-breaking notities.

## 3. Validatie & testsets

- **JSON-schema** (AJV) validatie per module.
- **Golden outputs** voor 50 casussen per module.
- **Quality metrics:** leesbaarheid (FK-score), consistentie (keys), vangst van contradictions.

## 4. Safety & compliance

- Invoersanitatie, gevoelige onderwerpen herformuleren.
- Blokkeer stellige uitkomsten; voeg “mogelijk/misschien” hedging.
- Toon altijd disclaimer bij output.

## 5. Voorbeeld prompts (NL/EN/TR) – snippets

- **NL Tarot:** “Schrijf in het Nederlands, warm en helder. Focus op {focus}.”
- **EN Numerology:** “Keep tone supportive and non-deterministic.”
- **TR Coffee:** “Metni yalın, umut verici ve saygılı tut.”

# StarpathVision – API & Datamodel Referentie (OpenAPI + Prisma)

Doel: Eén bron van waarheid voor endpoints, payloads, foutcodes, webhooks en database-schema's. Compatibel met Next.js API routes of een NestJS service.

---

## 0. Overzicht

- **Auth:** e-mail+password, OAuth (Google/Apple) – JWT of session cookies.
  - **Core:** readings (tarot, numerology, coffee, astro), exports, historiek.
  - **Gamification:** points, badges, levels.
  - **Billing:** Stripe checkout, customer portal, webhooks.
  - **Admin:** content, modules, rapporten.
- 

## 1. OpenAPI (YAML – excerpt)

```
openapi: 3.0.3
info:
  title: StarpathVision API
  version: 1.0.0
servers:
  - url: https://api.starpathvision.com
paths:
  /auth/register:
    post:
      summary: Register user
      requestBody:
        required: true
        content:
          application/json:
            schema:
              type: object
              required: [email, password]
              properties:
                email: { type: string, format: email }
                password: { type: string, minLength: 8 }
                name: { type: string }
                locale: { type: string, default: nl-NL }
      responses:
        '201': { description: Created }
        '400': { description: Invalid input }
  /auth/login:
    post:
      summary: Login
```

```

    responses:
      '200': { description: OK }
      '401': { description: Unauthorized }
/readings/tarot:
  post:
    summary: Create tarot reading
    requestBody:
      required: true
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/TarotInput'
    responses:
      '200':
        description: Result
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/ReadingSession'
/readings/{id}:
  get:
    summary: Get reading by id
    parameters:
      - in: path
        name: id
        required: true
        schema: { type: string }
    responses:
      '200': { description: OK }
      '404': { description: Not found }
/readings/{id}/export:
  post:
    summary: Export reading to PDF
    responses:
      '200': { description: OK }
/gamification/status:
  get:
    summary: Get user points, level, badges
    responses:
      '200': { description: OK }
/webhooks/stripe:
  post:
    summary: Stripe events
    responses:
      '200': { description: OK }
components:
  schemas:
    TarotInput:
      type: object
      properties:
        deckId: { type: string }

```

```

    spreadId: { type: string }
    seed: { type: integer }
  ReadingSession:
    type: object
    properties:
      id: { type: string }
      type: { type: string, enum: [tarot, numerology, coffee, astro] }
      payload: { type: object }
      aiResult: { type: object }
      createdAt: { type: string, format: date-time }

```

## 2. Prisma datamodel (v1)

```

model User {
  id          String    @id @default(cuid())
  email       String    @unique
  passwordHash String
  name        String?
  locale      String    @default("nl-NL")
  birthDate   DateTime?
  birthPlace  String?
  createdAt   DateTime @default(now())
  updatedAt   DateTime @updatedAt
  subscription Subscription?
  sessions    Session[]
  consents    Consent[]
  badges      UserBadge[]
  points      Int        @default(0)
}

model Session {
  id          String    @id @default(cuid())
  userId      String
  type        String    // tarot | numerology | coffee | astro
  payload     Json
  aiResult    Json?
  summary     String?
  score       Float?
  createdAt   DateTime @default(now())
  user        User      @relation(fields: [userId], references: [id])
}

model Subscription {
  id          String    @id @default(cuid())
  userId      String    @unique
  plan        String    // free | premium | pro
  status      String    // active | canceled | past_due
  providerId  String?   // Stripe customer id
}

```

```

    currentPeriodEnd DateTime?
    user          User      @relation(fields: [userId], references: [id])
}

model Consent {
  id          String  @id @default(cuid())
  userId      String
  kind        String  // analytics | ai-train | marketing
  granted     Boolean
  updatedAt   DateTime @updatedAt
  user        User    @relation(fields: [userId], references: [id])
}

model Badge {
  code        String  @id
  name        String
  description  String
  points      Int     @default(0)
}

model UserBadge {
  id          String  @id @default(cuid())
  userId      String
  badgeId     String
  createdAt   DateTime @default(now())
}

model PaymentEvent {
  id          String  @id @default(cuid())
  userId      String?
  type        String
  payload      Json
  createdAt   DateTime @default(now())
}

model Asset {
  id          String  @id @default(cuid())
  userId      String
  kind        String  // upload_photo, deck_image, exported_pdf
  path        String
  meta        Json?
  createdAt   DateTime @default(now())
}

```

### 3. Migrations & naming

- Tabelnamen enkelvoud, snake\_case in DB, camelCase in app.
- Soft-delete via `deletedAt` waar nodig (optioneel).
- Indexen: `Session(userId, createdAt)`, `Asset(userId, createdAt)`.

---

## 4. Auth & security

- Passwords: argon2id; min 12 chars; haveibeenpwned check.
- Sessions via HttpOnly cookies; SameSite=Lax/Strict.
- RBAC: roles `user`, `admin`; admin-routes achter IP allowlist of mTLS.
- Rate-limit: 60 rpm per IP + stricter op `/auth/*`.

---

## 5. Errors & statuscodes

- `400` validation, `401` auth, `403` forbidden, `404` not found, `409` conflict, `429` rate limit, `500` server.
- Foutformaat:

```
{ "error": { "code": "string", "message": "string", "details": { } } }
```

---

## 6. Webhooks (Stripe)

- Endpoint: `POST /webhooks/stripe` – valideer signatuur.
- Events: `checkout.session.completed`, `invoice.payment_succeeded`, `customer.subscription.updated`, `customer.subscription.deleted`.
- Idempotency: eventId deduplicatie in `PaymentEvent`.

---

## 7. Examples (requests)

### Create tarot reading

```
POST /readings/tarot
{ "deckId": "RWS", "spreadId": "3C", "seed": 42 }
```

### Response

```
{ "id": "sess_123", "type": "tarot", "payload": { ... }, "aiResult":
{ ... }, "createdAt": "2025-08-14T10:00:00Z" }
```

---

## 8. Rate limiting & idempotency

- Per-route en per user/IP; Redis leaky bucket.
  - Idempotency-keys voor exports en webhook-handlers.
-

## 9. Versiebeheer

- Prefix `/v1`; breaking changes via `/v2` met overgangsperiode.
  - OpenAPI bundelen in CI; publiceer Swagger UI voor interne teams.
- 

## 10. Checklist implementatie

- DTO-validatie (Zod/JOI).
- OpenAPI genereren en contract-tests draaien in CI.
- Prisma migrations uitvoeren in CI vóór deploy.
- Observability: log correlatie-id, trace spans, Sentry errors.

# StarpathVision – Auto-Build Handoff (bolt.new + Vercel + Firebase)

Deze canvas bevat de kant-en-klare overdrachtstekst voor bolt.new om een volledig werkend code-skelet te genereren voor StarpathVision, inclusief alle configuraties voor Vercel, Firebase, Stripe en AI.

## Master Build Prompt voor bolt.new

Plak dit in bolt.new om direct een monorepo te genereren:

```
Je bent de lead engineer voor "StarpathVision" (Next.js + Firebase + Stripe + AI). Bouw een monorepo die out-of-the-box draait en aansluit op onderstaande functionele/technische eisen.
```

### ## Doelen

- MVP opleveren: Registratie (Ontdekkingsreis), Tarot-reading (AI), Historiek, Stripe-checkout, Dagelijkse Kaart automation.
- EU/AVG-vriendelijk: alle Firebase resources in europe-west.
- Automations: Cloud Scheduler → Pub/Sub → Functions v2.
- Validatie: ALLE AI-uitvoer is strict JSON en wordt gevalideerd met AJV.

### ## Stack

- Frontend: Next.js (App Router), TypeScript, Tailwind.
- Hosting: Vercel (frontend).
- Backend: Firebase (Auth, Firestore, Storage, Functions v2, Scheduler, Pub/Sub).
- Payments: Stripe (Checkout + webhooks).
- AI: OpenAI (text + vision), JSON-output validatie (AJV).
- Tooling: pnpm, Turbo, ESLint, Prettier, Playwright.

### ## Monorepo structuur

```
starpathvision/  
  package.json (pnpm + turbo)  
  pnpm-workspace.yaml  
  turbo.json  
  firebase.json  
  firestore.rules  
  storage.rules  
  .firebaserc  
  .github/workflows/  
    deploy-firebase.yml      # firebase deploy (functions/firestore/storage/  
rules)  
    ci.yml                  # lint/test/build  
  apps/  
    web/                    # Next.js app
```



```

app/(marketing)/page.tsx
app/(app)/dashboard/page.tsx
app/(app)/readings/tarot/page.tsx
app/api/readings/[module]/route.ts
app/api/automations/run/daily-card/route.ts
app/og/daily-card/route.ts
lib/ai.ts           # OpenAI call wrappers
lib/schema.ts       # AJV validators
lib/firebase.ts     # client init (AppCheck)
lib/auth.ts         # next-auth (email)
env.d.ts
prisma/             # (optioneel) niet gebruiken voor Firebase
tailwind.config.ts
next.config.mjs
functions/           # Firebase Functions v2 (TypeScript)
  src/index.ts
  src/stripeWebhook.ts
  src/jobs/dailyCard.ts
  src/jobs/dailyEnergy.ts
  src/promptRegistry.ts
  package.json
scripts/
  seed.ts           # laadt seeds naar Firestore/Storage
packages/
  schemas/          # JSON schemas (AJV)
    src/tarot.v1.json
    src/numerology.v1.json
    src/index.ts
    package.json
  prompts/
    src/client.ts   # haalt templates uit Firestore
    seed/tarot.nl.v1.json
    package.json
  emails/
    src/DailyCardEmail.tsx
    package.json
  ui/
    src/index.ts
    package.json
seeds/
  tarot.decks.json
  spreads.json
  numerology.tables.json
  badges.json
  demo.users.ndjson
  demo.sessions.ndjson
tooling/
  postman/starpathvision.postman_collection.json
  openapi/openapi.yaml
.env.example

```

```
## Vereisten (belangrijk)
- Firebase projectconfig hard-coded als placeholders. Gebruik regions "europe-west1".
- firestore.rules en storage.rules: least-privilege, alleen eigen data lezen/schrijven.
- Functions v2 met region europe-west1 en minInstances=0 (kosten), met http en onSchedule triggers.
- Cloud Scheduler voor Dagkaart: 00:05 Europe/Amsterdam → Pub/Sub → functions.dagkaart().
- Stripe webhook (v2 http function) met idempotency en signature verification.
- OpenAI calls: strikt JSON, AJV-validate (tarot.v1, numerology.v1). Bij invalid → één retry met "repair" prompt.
- Next.js routes: /api/readings/tarot roept via fetch de function aan OF direct OpenAI met server action.
- OG-image route voor Daily Card (edge runtime).
- Seed script dat JSON/NDJSON in Firestore zet (collections: users, sessions, prompts, badges, decks, spreads).
- GitHub Actions: "deploy-firebase.yml" met `firebase deploy --only functions,firestore,storage,hosting` (hosting mag later).
- README met: setup, env, emulator, deploy (firebase emulators:start / firebase deploy).
- Voeg voorbeeld tests toe: Playwright voor tarot flow (mock AI), unit test AJV validators.

## ENV placeholders
OPENAI_API_KEY=
STRIPE_SECRET_KEY=
STRIPE_WEBHOOK_SECRET=
NEXT_PUBLIC_FIREBASE_API_KEY=
NEXT_PUBLIC_FIREBASE_AUTH_DOMAIN=
NEXT_PUBLIC_FIREBASE_PROJECT_ID=
NEXT_PUBLIC_FIREBASE_STORAGE_BUCKET=
NEXT_PUBLIC_FIREBASE_APP_ID=
NEXT_PUBLIC_FIREBASE_MEASUREMENT_ID=
CRON_TZ=Europe/Amsterdam
```

---

## Volgende stap

1. Open [bolt.new](https://bolt.new) en plak bovenstaande prompt in het tekstveld.
  2. Laat bolt.new de repo genereren en opslaan in een GitHub-repo.
  3. Koppel de repo aan **Vercel** (frontend hosting) en **Firebase** (backend).
  4. Configureer `.env` variabelen in zowel Vercel als Firebase.
  5. Run `firebase deploy` om functies, Firestore-regels en opslag te publiceren.
  6. Test de volledige flow met de emulators en Playwright.
-

Wil je dat ik in deze canvas ook alvast de **GitHub Actions workflow bestanden** en een **firebase.json + firestore.rules** toevoeg zodat je die niet meer zelf hoeft te maken?

# StarpathVision – Auto-Build Handoff (bolt.new + Vercel + Firebase)

Doel: Eén pakket waarmee een auto-builder (zoals **bolt.new**) direct een repo kan genereren én via **Vercel + Firebase** automatisch kan deployen. Bevat: Master Build Prompt, workflows, Firebase-config, Rules, Functions v2 skeleton, Seeds, en README-stappen.

---

## 1) Master Build Prompt (plak in bolt.new)

Je bent de lead engineer voor “StarpathVision” (Next.js + Firebase + Stripe + AI). Bouw een monorepo die out-of-the-box draait en aansluit op onderstaande eisen.

### # Doelen

- MVP: Registratie (Ontdekkingsreis), Tarot-reading (AI), Historiek, Stripe-checkout, Dagelijkse Kaart automation.
- EU/AVG: alle Firebase resources in europe-west1; security rules least-privilege.
- Automations: Cloud Scheduler → Pub/Sub → Functions v2.
- Validatie: ALLE AI-uitvoer is strict JSON en wordt gevalideerd met AJV (tarot.v1, numerology.v1).

### # Stack

- Frontend: Next.js (App Router), TypeScript, Tailwind.
- Hosting: Vercel (frontend).
- Backend: Firebase (Auth, Firestore, Storage, Functions v2, Scheduler, Pub/Sub, App Check).
- Payments: Stripe (Checkout + webhooks).
- AI: OpenAI (text/vision), JSON-schema-validatie (AJV). Invalid → één retry met “repair prompt”.
- Tooling: pnpm, Turborepo, ESLint, Prettier, Playwright.

### # Monorepo structuur

```
starpathvision/  
  package.json  
  pnpm-workspace.yaml  
  turbo.json  
  .env.example  
  firebase.json  
  firestore.rules  
  storage.rules  
  .firebaserc  
  .github/workflows/  
    ci.yml  
    deploy-firebase.yml
```

```
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/automations/run/daily-card/route.ts  
    app/og/daily-card/route.ts  
    lib/ai.ts  
    lib/schema.ts  
    lib/firebase.ts  
    lib/auth.ts  
    tailwind.config.ts  
    next.config.mjs  
  functions/  
    src/index.ts  
    src/stripeWebhook.ts  
    src/jobs/dailyCard.ts  
    src/jobs/dailyEnergy.ts  
    src/promptRegistry.ts  
    package.json  
packages/  
  schemas/  
    src/tarot.v1.json  
    src/numerology.v1.json  
    src/index.ts  
    package.json  
  prompts/  
    src/client.ts  
    seed/tarot.nl.v1.json  
    package.json  
  emails/  
    src/DailyCardEmail.tsx  
    package.json  
  ui/  
    src/index.ts  
    package.json  
seeds/  
  tarot.decks.json  
  spreads.json  
  numerology.tables.json  
  badges.json  
  demo.users.ndjson  
  demo.sessions.ndjson  
tooling/  
  postman/starpathvision.postman_collection.json  
  openapi/openapi.yaml  
scripts/  
  seed.ts  
  
# Vereisten
```

- **\*\*Region\*\***: alle Functions v2 in `europe-west1`; Scheduler/PubSub idem; Firestore/Storage in EU.
- **\*\*Rules\*\***: beperkte lees/schrijf (eigen data), admins via custom claims.
- **\*\*Scheduler\*\***: Dagkaart 00:05 Europe/Amsterdam → Pub/Sub → functions.dagkaart().
- **\*\*Stripe\*\***: webhook v2 http function met signature verify + idempotency; status in `subscriptions` collectie.
- **\*\*Readings API\*\***: Next.js route → server action of fetch naar function; AI-output **\*\*AJV\*\***-valideren.
- **\*\*Seeds\*\***: laad JSON/NDJSON naar Firestore (collecties: users, sessions, prompts, decks, spreads, badges).
- **\*\*Tests\*\***: Playwright E2E (tarot flow - AI mock), unit AJV validators.
- **\*\*README\*\***: setup, emulators, deploy (firebase deploy) + Vercel koppeling.

```
# ENV placeholders
OPENAI_API_KEY=
STRIPE_SECRET_KEY=
STRIPE_WEBHOOK_SECRET=
NEXT_PUBLIC_FIREBASE_API_KEY=
NEXT_PUBLIC_FIREBASE_AUTH_DOMAIN=
NEXT_PUBLIC_FIREBASE_PROJECT_ID=
NEXT_PUBLIC_FIREBASE_STORAGE_BUCKET=
NEXT_PUBLIC_FIREBASE_APP_ID=
NEXT_PUBLIC_FIREBASE_MEASUREMENT_ID=
CRON_TZ=Europe/Amsterdam
```

## 2) Firebase config (kopie-klaar)

### 2.1 `firebase.json`

```
{
  "functions": {
    "source": "apps/functions",
    "ignore": ["node_modules", ".git", ".github", "**/tests/**"],
    "runtime": {
      "language": "nodejs",
      "region": "europe-west1"
    }
  },
  "emulators": {
    "functions": { "port": 5001 },
    "firestore": { "port": 8080 },
    "auth": { "port": 9099 },
    "storage": { "port": 9199 },
    "ui": { "enabled": true }
  }
}
```

## 2.2 .firebaserc

```
{ "projects": { "default": "starpavision-dev" } }
```

## 2.3 firestore.rules (least-privilege)

```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {
    function isSignedIn() { return request.auth != null; }
    function isOwner(uid) { return request.auth != null && request.auth.uid
== uid; }
    function isAdmin() { return request.auth.token.admin == true; }

    // Publieke configuraties
    match /public/{doc} {
      allow read: if true;
      allow write: if false;
    }

    // Gebruikersdata
    match /users/{userId} {
      allow read, update, delete: if isOwner(userId) || isAdmin();
      allow create: if isSignedIn() && request.resource.data.uid ==
request.auth.uid;
    }

    // Sessies (alleen eigenaar of admin)
    match /users/{userId}/sessions/{sid} {
      allow read, create: if isOwner(userId) || isAdmin();
      allow update, delete: if isOwner(userId) || isAdmin();
    }

    // Subscriptions
    match /subscriptions/{subId} {
      allow read: if isOwner(resource.data.userId) || isAdmin();
      allow write: if isAdmin();
    }

    // Prompts (alleen lezer; wijzigingen via admin)
    match /prompts/{id} {
      allow read: if true;
      allow write: if isAdmin();
    }

    // Admin collections
    match /admin/{doc=**} {
      allow read, write: if isAdmin();
    }
  }
}
```

```
}  
}
```

## 2.4 storage.rules

```
rules_version = '2';  
service firebase.storage {  
  match /b/{bucket}/o {  
    function isSignedIn() { return request.auth != null; }  
    function isOwner(uid) { return request.auth != null && request.auth.uid  
== uid; }  
  
    // User uploads (foto's, exports)  
    match /users/{userId}/{allPaths=**} {  
      allow read, write: if isOwner(userId);  
    }  
  
    // Publieke assets (deck images, og images)  
    match /public/{allPaths=**} {  
      allow read: if true;  
      allow write: if false;  
    }  
  }  
}
```

## 3) Functions v2 skeleton (TypeScript)

### 3.1 apps/functions/src/index.ts

```
import { onRequest } from 'firebase-functions/v2/https'  
import { onSchedule } from 'firebase-functions/v2/scheduler'  
import * as logger from 'firebase-functions/logger'  
import { dagkaart } from './jobs/dailyCard'  
import { stripeWebhookHandler } from './stripeWebhook'  
  
export const stripeWebhook = onRequest({ region: 'europe-west1' },  
stripeWebhookHandler)  
  
export const dailyCard = onSchedule({ schedule: '5 0 * * *', timeZone:  
'Europe/Amsterdam', region: 'europe-west1' }, async () => {  
  logger.info('Daily card job start')  
  await dagkaart()  
  logger.info('Daily card job done')  
})  
  
export const health = onRequest({ region: 'europe-west1' }, (req, res) => {
```



```
res.status(200).json({ ok: true, ts: Date.now() })
})
```

### 3.2 apps/functions/src/jobs/dailyCard.ts

```
import { Firestore } from 'firebase-admin/firestore'
import OpenAI from 'openai'
import Ajv from 'ajv'
import tarotSchema from '@spv/schemas/src/tarot.v1.json'

const ajv = new Ajv({ allErrors: true })
const validate = ajv.compile(tarotSchema as any)

export async function dagkaart() {
  const fs = new Firestore()
  const dateISO = new Date().toISOString().slice(0,10)
  const seed = Number.parseInt(Buffer.from(`DAILY_CARD:${dateISO}`)
    .toString('hex').slice(0,8), 16)
  const ai = new OpenAI({ apiKey: process.env.OPENAI_API_KEY })

  // haal template uit Firestore (vereenvoudigd)
  const tplSnap = await
fs.collection('prompts').where('module','==','tarot').limit(1).get()
  const template = tplSnap.docs[0]?.data()

  const aiOut = await ai.chat.completions.create({
    model: 'gpt-4o-mini',
    messages: [
      { role: 'system', content: template.system },
      { role: 'user', content: JSON.stringify({ locale: 'nl-NL', seed,
spread: '3C' }) }
    ],
    response_format: { type: 'json_object' }
  })

  const payload = JSON.parse(aiOut.choices[0].message.content || '{}')
  if (!validate(payload)) throw new Error('TAROT_SCHEMA_INVALID')

  await fs.collection('public').doc('dailyCard').set({ dateISO, payload,
createdAt: Date.now() })
}
```

### 3.3 apps/functions/src/stripeWebhook.ts

```
import type { Request, Response } from 'express'
import Stripe from 'stripe'

export async function stripeWebhookHandler(req: Request, res: Response) {
  const sig = req.headers['stripe-signature'] as string
```

```

const stripe = new Stripe(process.env.STRIPE_SECRET_KEY!, { apiVersion:
'2023-10-16' })
let event: Stripe.Event
try {
  event = stripe.webhooks.constructEvent(req.rawBody, sig,
process.env.STRIPE_WEBHOOK_SECRET!)
} catch (err: any) {
  return res.status(400).send(`Webhook Error: ${err.message}`)
}

switch (event.type) {
case 'checkout.session.completed':
  // TODO: markeer subscription in Firestore
  break
case 'invoice.payment_succeeded':
  // TODO: status=active
  break
case 'invoice.payment_failed':
  // TODO: status=past_due
  break
}

res.json({ received: true })
}

```

### 3.4 apps/functions/package.json

```

{
  "name": "@spv/functions",
  "type": "module",
  "engines": { "node": ">=20" },
  "dependencies": {
    "firebase-admin": "^12.4.0",
    "firebase-functions": "^5.0.0",
    "openai": "^4.52.0",
    "ajv": "^8.12.0",
    "stripe": "^14.0.0"
  },
  "devDependencies": { "typescript": "^5.4.0" }
}

```

## 4) GitHub Actions (CI + Deploy)

### 4.1 .github/workflows/ci.yml

```

name: CI
on: [push, pull_request]

```

```

jobs:
  build-test:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v4
      - uses: pnpm/action-setup@v3
        with: { version: 9 }
      - uses: actions/setup-node@v4
        with: { node-version: 20, cache: 'pnpm' }
      - run: pnpm i
      - run: pnpm lint && pnpm test && pnpm build

```

## 4.2 `.github/workflows/deploy-firebase.yml`

```

name: Deploy Firebase
on:
  push:
    branches: [ main ]
jobs:
  deploy:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v4
      - uses: pnpm/action-setup@v3
        with: { version: 9 }
      - uses: actions/setup-node@v4
        with: { node-version: 20, cache: 'pnpm' }
      - run: pnpm i
      - uses: google-github-actions/auth@v2
        with:
          credentials_json: ${ secrets.FIREBASE_SA_JSON }
      - uses: w9jds/firebase-action@v13
        with:
          args: deploy --only functions,firestore:rules,storage:rules

```

Zet **FIREBASE\_SA\_JSON** als repo secret (Service Account JSON met deploy-rechten). Vercel deployment voor `apps/web` apart koppelen.

## 5) Next.js endpoints (web) – voorbeeld

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

```

import { NextRequest, NextResponse } from 'next/server'
import { createTarotReading } from '@lib/ai'
import { validateTarot } from '@lib/schema'

export async function POST(req: NextRequest, { params }: { params: { module:

```

```

string } }) {
  const body = await req.json()
  if (params.module !== 'tarot') return NextResponse.json({ error: { code:
'UNSUPPORTED' } }, { status: 400 })
  const aiOut = await createTarotReading(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 })
}

```

## 5.2 apps/web/app/og/daily-card/route.ts (Edge)

```

export const runtime = 'edge'
export async function GET() {
  // render OG image voor daily card
  return new Response('OK', { status: 200 })
}

```

# 6) Seeds & scripts

## 6.1 scripts/seed.ts (schets)

```

import { initializeApp, cert } from 'firebase-admin/app'
import { getFirestore } from 'firebase-admin/firestore'
import fs from 'node:fs/promises'

initializeApp({ credential:
cert(JSON.parse(process.env.FIREBASE_SA_JSON!)) })
const db = getFirestore()

async function loadJSON(path: string) { return JSON.parse(await
fs.readFile(path, 'utf-8')) }

async function main() {
  const decks = await loadJSON('seeds/tarot.decks.json')
  for (const d of decks) await db.collection('decks').doc(d.id).set(d)
  const spreads = await loadJSON('seeds/spreads.json')
  for (const s of spreads) await db.collection('spreads').doc(s.id).set(s)
  // ... badges, numerology, prompts
}

main().then(()=>process.exit(0))

```

## 7) .env.example

```
OPENAI_API_KEY=  
STRIPE_SECRET_KEY=  
STRIPE_WEBHOOK_SECRET=  
NEXT_PUBLIC_FIREBASE_API_KEY=  
NEXT_PUBLIC_FIREBASE_AUTH_DOMAIN=  
NEXT_PUBLIC_FIREBASE_PROJECT_ID=  
NEXT_PUBLIC_FIREBASE_STORAGE_BUCKET=  
NEXT_PUBLIC_FIREBASE_APP_ID=  
NEXT_PUBLIC_FIREBASE_MEASUREMENT_ID=  
CRON_TZ=Europe/Amsterdam
```

## 8) README – Quickstart

1. **Repo genereren** via bolt.new (prompt hierboven) → push naar GitHub.
2. **Vercel** koppelen aan apps/web → env vars invullen.
3. **Firebase** project (EU) aanmaken, service account aanmaken → secret FIREBASE\_SA\_JSON in GitHub.
4. **Deploy:** Git push naar main → Actions deployen Functions + Rules.
5. **Local dev:** firebase emulators:start (Auth/Firestore/Storage/Functions), pnpm dev.

## 9) Acceptatiecriteria (auto-build OK)

- CI build groen; deploy-workflow werkt zonder handwerk.
- /api/readings/tarot retourneert geldige JSON volgens tarot.v1.
- Dagkaart job draait dagelijks 00:05 (Europe/Amsterdam) en schrijft naar public/dailyCard.
- Stripe webhook zet subscription status correct bij test events.

**Opmerking:** Deze handoff volgt 1-op-1 de inhoud van je andere canvassen (Prompts/Schema's, API/Datamodel, Automations). Een bouwer kan dit kopiëren of via bolt.new genereren, verbinden met Vercel + Firebase en direct starten met P0-taken.

# StarpathVision – Billing & Facturatie Edge-cases

Doel: Robuuste afhandeling van abonnementen, lokale betaalmethoden (iDEAL/SEPA), proration, refunds, BTW/OSS en edge-cases. Stripe als provider.

---

## 1) Plannen & prijzen

- **Free:** 1 korte tarot/week, basis numerologie.
- **Premium:** onbeperkt modules, uitgebreide PDF, combinatiemotor.
- **Pro:** + Live AI chat, maandrapport, prioriteit.

Prijsconfig beheerbaar in Admin; valuta: EUR (basis), later USD/GBP.

---

## 2) Checkout & portal

- Stripe Checkout (one-time of subscription) + Customer Portal voor self-service (upgrade/downgrade, betaalmethode, facturen).
  - Betaalmethoden: **iDEAL**, **SEPA direct debit**, **Cards**.
- 

## 3) Webhooks & statusmachine

Events → PaymentEvent + Subscription update.

```
checkout.session.completed -> create customer/subscription
invoice.payment_succeeded  -> status=active; currentPeriodEnd set
invoice.payment_failed     -> status=past_due; mail nudge
customer.subscription.updated|deleted -> sync status
```

Edge: idempotency op eventId; replay-safe.

---

## 4) Proration & upgrades

- Upgrade: pro-rata verrekening resterende periode; direct toegang tot hogere tier.
  - Downgrade: lagere tier ingaan aan einde huidige periode.
  - Plan switch binnen dezelfde dag → enkel laatste wijziging doorvoeren (idempotency key per user+day).
- 

## 5) Trials & coupons

- 7 of 14 dagen gratis proefperiode (feature flag-gestuurd).

- Coupons: percentage/korting; verloopdatum; eenmalig of herhalend.
- 

## 6) Refund policy (richtlijnen)

- Binnen 14 dagen na aankoop, mits geen excessief gebruik.
  - Geen refunds op deels gebruikte abonnementsperiode behalve bij storingen.
  - Documenteer uitzonderingen (consumentenrecht per land).
- 

## 7) BTW/OSS

- OSS (One-Stop Shop) registratie voor EU.
  - Bepaal btw op basis van klantland + product (digitale dienst).
  - Facturen tonen BTW-tarief, BTW-nummer bedrijf, land klant.
- 

## 8) Facturen & e-mails

- Factuur PDF link in portal + mail.
  - E-mailtemplates: `invoice_paid`, `payment_failed`, `subscription_updated`.
- 

## 9) Foutafhandeling & retries

- Webhook handler: retries met backoff; log naar `PaymentEvent`.
  - Timeouts → DLQ + alert.
  - Onbekend event → opslaan en markeren als `ignored`.
- 

## 10) API-routes (billing)

```
POST /api/billing/checkout # body: plan, locale
GET  /api/billing/portal   # redirect url
POST /api/webhooks/stripe  # signature required
```

---

## 11) Testmatrix (Stripe test-mode)

- Succesvolle iDEAL + SEPA incasso.
  - Mislukte incasso → `past_due` + e-mail.
  - Upgrade/downgrade met proration.
  - Coupon toegepast en verlopen.
  - Refund uitgegeven, status bijgewerkt.
-

## 12) Acceptatiecriteria

- Subscription lifecycle consistent met events.
- iDEAL/SEPA beschikbaar voor NL/EU gebruikers.
- Proration bedragen correct volgens Stripe berekening.
- Facturen beschikbaar + BTW correct per land.
- Fouten leiden niet tot verlies van toegang of dubbele facturatie.



# StarpathVision – Campagneplan & Contentvoorbeelden

Doel: Direct inzetbare campagneformats, voorbeeldposts en e-mailteksten, afgestemd op de merkidentiteit en meertalige doelgroep (NL, EN, TR).

---

## 1) Campagnestructuur

Elke campagne heeft: 1. **Doel** (traffic, conversie, retentie) 2. **Thema** (liefde, zelfontwikkeling, seizoensgebonden) 3. **Kanalen** (Instagram, TikTok, YouTube Shorts, e-mail) 4. **Contentvormen** (video, carrousel, blog, e-mail) 5. **Call-to-Action** (gratis reading, upgrade naar Premium, vriend uitnodigen)

---

## 2) Voorbeeldcampagne: "Kaart van de Dag"

- **Doel:** dagelijkse terugkeer stimuleren.
- **Kanalen:** Instagram Stories/Reels, TikTok, e-mail snippet.
- **Contentvormen:**
  - Video: AI-geanimeerde kaart + korte interpretatie (<30s).
  - Afbeelding: kaartvisual met quote.
- **Call-to-Action:** "Ontvang jouw volledige reading in de app".

### Voorbeeldpost (Instagram – NL)

**Titel:** 🌟 Jouw kaart van vandaag: De Keizerin **Tekst:** Vandaag staat in het teken van groei en overvloed. Sta open voor nieuwe kansen, zowel in je werk als in je hart. 🌱 **CTA:** Ontdek de volledige betekenis in jouw gratis reading → [link]

### E-mail snippet (NL)

**Onderwerp:** 🌟 Je kaart van vandaag wacht op je... **Body:**

Beste [naam], Vandaag begeleidt De Keizerin je pad. Ontdek wat dit betekent voor jouw dag. [Gratis bekijk je reading]

---

## 3) Voorbeeldcampagne: "Liefdesweek"

- **Doel:** Upsell naar Premium (relatie-analyse module).
- **Kanalen:** Instagram, blog, e-mail.
- **Contentvormen:**
  - Blog: "5 manieren waarop tarot je liefdesleven kan verhelderen".
  - Reel: voor/na voorbeeld van een relatie-analyse.
- **Call-to-Action:** "Ontgrendel jouw liefdesreading – nu met 20% korting".

## Voorbeeldpost (Instagram – EN)

**Title:** 🐘 Love Week is here! **Text:** Ready to unlock the secrets of your relationship? Our AI-powered love readings combine Tarot and Numerology for deep insights. **CTA:** Upgrade to Premium now → [link]

---

## 4) Voorbeeldcampagne: "Eindejaarsreflectie"

- **Doel:** Retentie en jaarafsluiting.
- **Kanalen:** e-mail, YouTube Shorts.
- **Contentvormen:**
  - E-mail: persoonlijke jaarterugblik (AI-samenvatting van sessies).
  - Video: top 3 kaarten die jij trok dit jaar.
  - **Call-to-Action:** "Plan jouw Nieuwjaarsreading".

## Voorbeeldmail (NL)

**Onderwerp:** ✨ Jouw spirituele reis in 2025 **Body:**

Dit jaar heb je [X] readings gedaan, waarin thema's als [thema1] en [thema2] vaak terugkwamen. Ontdek hoe je deze energie meeneemt in 2026. [Plan jouw Nieuwjaarsreading]

---

## 5) Meertalige formats

Alle teksten zijn beschikbaar in **NL, EN, TR**.

- Vertaling via AI + menselijke review. - Culturele nuances per taal (bv. symboliek).

---

## 6) Publicatieplanning (voorbeeld – 1 maand)

Dag	Contenttype	Kanaal	Thema
Ma	Kaart van de dag	Instagram/TikTok	Algemeen
Di	Blog + post	Website + IG	Educatief
Wo	Reel/Shorts	TikTok/YouTube	Inspirerend
Do	Poll/quiz	Instagram	Interactief
Vr	Campagnepost	IG + e-mail	Thema-actie
Za	Kaart van de dag	Instagram	Algemeen
Zo	Reflectiepost	Instagram/Blog	Weekoverzicht

---

## 7) Volgende stappen

- Canva-template set maken in merkstijl.
- AI-scripts klaarzetten voor "Kaart van de dag" automatisering.
- E-mailflows koppelen aan CRM triggers.
- Social ads A/B-tests plannen met deze campagnes.

# StarpathVision – Canvas Map & Gap Analysis

Dit document ordent alle canvassen (huidig en gepland) en benoemt ontbrekende onderdelen om StarpathVision tot een écht **alles-in-één** platform te maken. Onder elke sectie staan **aanbevolen acties**.

---

## 1) Canvas-structuur (map)

**Reeds aanwezig** 1. **Complete Build Spec (Alles-in-één)** – functioneel + technisch ontwerp, architectuur, roadmap.

2. **UI Kit & Mockups (React)** – low-fi screens, componenten, flows.

**Voorstel nieuwe canvassen (gescheiden en onderhoudbaar)** 3. **AI Promptbibliotheek & JSON-Schema's**

- Alle prompts per module (Tarot, Koffiedik, Numerologie, Combinatiemotor) + strikte output-schema's en versies. 4. **API & Datamodel Referentie (OpenAPI + Prisma)**

- Endpoints, request/response, webhooks (Stripe), foutcodes, throttling; Prisma modellen en migratieregels. 5. **Content & Tone Guide (NL/EN/TR)**

- Stijl, metaforen, disclaimers, voorbeeldzinnen, UI-copy keys; i18n-strategie. 6. **Design Tokens & Style Guide**

- Kleuren, spacing, radius, schaduwen, typografie, component-varianten. 7. **QA & Testplan**

- Testpiramide, testcases per flow, prompt-goldens, performance- en beveiligingstesten, acceptatiecriteria. 8. **Security & Privacy (AVG) Dossier**

- DPIA, datastromen, bewaartermijnen, rechtezets, encryptiebeleid, incidentproces, subverwerkers. 9.

**Ops & Runbooks**

- Deploy, rollback, backup/restore, key-rotatie, monitoring-alerts, AI-provider fallback. 10. **Growth & CRM**

- Onboarding, e-mailflows, referral, cadeaubonnen, churn-preventie, cohort-dashboards. 11. **Billing & Plans**

- Prijsblokken, edge cases (proration, refunds), regionale BTW, SEPA/iDEAL nuances, facturen. 12. **Community & Moderatie** (optioneel)

- Anoniem delen, rapportage-flow, huisregels, notice-&-takedown.

**Actie:** Ik kan #3 en #4 vandaag opzetten; daarna #5 en #6.

---

## 2) Gap Analysis – wat mist er nog?

### A. Product & UX

- **Onboarding-varianten:** kort ( $\leq 60s$ ) vs uitgebreid (profiel + voorkeuren).
- **Toegankelijkheid (A11y):** kleurcontrast, toetsenbordnavigatie, screenreaders, ARIA-labels.
- **Offline/Low-bandwidth:** progressive enhancement, skeletons, retry bij netwerkverlies.
- **Mobile PWA:** installable app, push-notificaties (herinneringen).
- **SEO & Discoverability:** metadata per module, OG-images, sitemap, structured data. **Acties:** A11y-audit checklist; PWA inschakelen; OG-template genereren voor deelbare quotes.

## B. AI & Kwaliteit

- **Prompt-versiebeheer** + **feature-flags** (per cohort uitrollen).
- **Model fallback**: primaire LLM → secundaire; vision: klassiek → ML-weights.
- **Evaluatie-set**: 50+ representatieve cases per module; automatische score (readability, JSON-validiteit).
- **Toxicity/ethics filters**: vang ongepaste input/output en toon veilige alternatieven. **Acties**: Prompt registry + "golden outputs" vastleggen; nightly eval job.

## C. Data & Analytics

- **Event-schema**: `user_registered`, `started_reading`, `exported_pdf`, `plan_upgraded`, etc.
- **Cohort & funnel**: T2F, F→P conversie, 7/30d retentie, LTV.
- **Content-analytics**: welke spreads/nummers geven hoogste tevredenheid/retentie.
- **Data Catalog**: definities/bron van waarheden (BI). **Acties**: trackingplan + consent-aware toggles; dashboards + alerts.

## D. Security, Privacy & Compliance

- **Colomencryptie** voor gevoelige velden (geboortedatum) + **KMS** sleutels.
- **Admin-audit logs**: wie keek/veranderde wat.
- **Rate-limits** per route en user; anti-abuse (captcha bij spikes).
- **Retentiebeleid**: auto-purge inactieve data na X maanden. **Acties**: DPIA invullen; DPA's met hosting/AI/Stripe; incident-plan oefenen.

## E. Payments & Juridisch

- **BTW/OSS**: juiste tarieven per land in EU.
- **Proration & trials**: upgrade/downgrade scenario's.
- **Refund-policy**: uitzonderingen (AI-output), duidelijke T&C-clausule.
- **Korting & vouchers**: cadeaubonnen, referrals. **Acties**: Billing-flow testmatrix; webhook-retries + idempotency.

## F. Ops & Prestatie

- **SLA & SLO's**: latency doel < 300ms voor meeste API's; uptime ≥ 99.9%.
- **Cold start mitigatie** (edge/ISR); **edge caching** voor publieke content.
- **Backups**: dagelijks + versies voor S3; **restore-oefeningen**.
- **Observability**: Trace sampling + error budgets. **Acties**: Chaos-tests (AI timeouts, queue backlog); runbooks aanvullen.

## G. Integraties (optioneel, toekomst)

- **Kalender** (herinneringen/rituelen), **e-mail** (Postmark/Resend), **push** (OneSignal/WebPush).
- **CRM** (Customer.io/HubSpot)
- **E-commerce** (Shop/affiliates: decks, kristallen).
- **Support** (Helpcenter/FAQ + chat). **Acties**: E-mail templates + journey's (onboarding, win-back, expiries).

### 3) Alles-in-één: hoe het samenkomt

- **Monorepo** (web, worker, ui-package, prompts-package).
  - **Design tokens** → UI kit → pagina's → consistente PDF/styling.
  - **Prompt registry** → AI-laag → Combinatiemotor → JSON → UI/PDF.
  - **Event tracking** → dashboards → growth & productbeslissingen.
- 

### 4) Volgende stappen (concreet)

1. **Aanmaken** van canvassen: #3 AI Prompts/Schema's, #4 API & Datamodel.
2. **A11y & PWA** taken toevoegen aan Build Spec + UI Kit.
3. **Trackingplan** (events + consent) uitschrijven.
4. **Security/AVG dossier** basis neerzetten (DPIA template, subverwerkerslijst).
5. **Billing edge-cases** en T&C paragrafen uitwerken.

Geef aan welke twee canvassen je als eerste wil; ik maak ze meteen aan en vul ze met startinhoud.

# Canvas-structuurvoorstel voor StarpathVision

Om overzicht en flexibiliteit te behouden, verdelen we het project over meerdere canvas-documenten. Elk document kan apart worden aangepast zonder invloed op de rest.

## 1. StarpathVision – Complete Build Spec

- Functioneel ontwerp
- Technische architectuur
- Roadmap
- Beveiliging en privacy
- Verdienmodel

## 2. UI Kit & Mockups

- Low-fi en high-fi mockups van alle schermen
- React/Tailwind componenten
- Stijlrichtlijnen (kleur, typografie, iconografie)
- Animatievoorbeelden (Framer Motion)
- Responsive breakpoints

## 3. AI Promptbibliotheek & JSON-schema's

- Prompt-templates per module (Tarot, Koffiedik, Numerologie, Astrologie)
- Strikte JSON-output schema's + versies
- Combinatie-motor logica (if/then regels, prioritering)
- Tone-of-voice richtlijnen en metafoorbank
- Meertalige prompts (NL, EN, TR)

## 4. API & Datamodel Referentie (OpenAPI + Prisma)

- Volledige OpenAPI/Swagger specificatie
- Database schema's (Prisma modellen)
- API-routes, parameters, validatie
- Webhook-structuren (Stripe, AI)
- Foutafhandeling & statuscodes
- Versiebeheerstrategie

## 5. Seed-data & Contentvoorbeelden

- Tarot-decks en spreads
- Symbolenbibliotheek voor koffiedik
- Numerologie-tabellen
- Badge- en levelsysteem
- Voorbeelddata voor tests en demo's

## 6. Marketing & Contentstrategie

- Merkpositionering
- Tekstvoorbeelden (NL, EN, TR)
- Social media formats

- E-mailtemplates en journeys
- SEO en meta-tags
- Blog/contentplanning

## 7. Testplan & Kwaliteitscontrole

- Unit-, integratie- en end-to-end testen
- AI-output validatie (golden outputs)
- Acceptatiecriteria per module
- Load- en performance-tests
- Accessibility tests (A11y)

## 8. Security & Privacy Dossier

- DPIA (Data Protection Impact Assessment)
- Datastromen en bewaartermijnen
- Subverwerkerslijst
- Incidentresponsplan
- AVG/GDPR compliance

## 9. Content & Tone Guide

- Stijl en tone-of-voice richtlijnen per taal
- Disclaimers en ethische kaders
- Woordenlijst/metaforenbank
- AI-output consistency checks

---

## Mogelijke uitbreidingen voor een alles-in-één platform

1. **Live consultaties** (videochat + chatgeschiedenis)
  2. **Community-sectie** (anoniem delen, moderatie)
  3. **Persoonlijke tijdlijn** (visualisatie sessies)
  4. **Kalenderintegraties** (Google, iCal)
  5. **Gepersonaliseerde notificaties** (push, e-mail)
  6. **AI-visuals** bij readings
  7. **E-commerce integratie** (decks, kristallen, boeken)
  8. **Gamification-verdieping** (events, uitdagingen, seizoensgebonden beloningen)
  9. **Open API** voor partners
  10. **Volledige toegankelijkheid (A11y)** inclusief screenreader- en toetsenbordnavigatie
  11. **Progressive Web App (PWA)** voor offline gebruik
  12. **Meertalige UI-switcher** met automatische locale-detectie
  13. **Analytics-dashboard** (gebruik, retentie, conversie)
- 

**Volgende stap:** Nu opzetten: 1. **Security & Privacy Dossier** – DPIA, datastromen, AVG-compliance. 2. **Content & Tone Guide** – inclusief voorbeeldteksten en metaforenbank.



# StarpathVision – API & Datamodel Referentie (OpenAPI + Prisma)

Doel: Technische referentie voor backend-ontwikkeling met directe integratie van AI Promptbibliotheek & JSON-schema's.

---

## 1. API Architectuur

- **Framework:** Node.js + Next.js API routes / NestJS
  - **Database:** PostgreSQL via Prisma ORM
  - **AI-integratie:** OpenAI API / lokale AI-service via server-side calls
  - **Opslag:** MinIO (S3-compatibel) voor uploads
  - **Authenticatie:** JWT + OAuth2 (optioneel Google/Apple login)
  - **Betaalprovider:** Stripe
- 

## 2. Kern-Endpoints (OpenAPI excerpt)

### POST /api/readings/{module}

- **Beschrijving:** Start een reading voor een specifieke module (tarot, coffee, numerology, astro, combine).
- **Request Body** (voorbeeld Tarot):

```
{
  "input": {
    "deck": "Rider-Waite",
    "spread": "Three Card",
    "cards": ["The Fool", "The Tower", "The Sun"],
    "language": "NL"
  }
}
```

- **Response** (JSON conform module-schema uit AI Promptbibliotheek):

```
{
  "summary": "...",
  "interpretation": [...],
  "overall_message": "...",
  "metaphors_used": ["..."]
}
```

- **Integratie:** Backend haalt prompt-template op, voegt Content & Tone Guide instructies toe, valideert AI-output met JSON Schema Validator.
-

## GET /api/readings/history

- **Beschrijving:** Ophalen van eerdere sessies voor ingelogde gebruiker.
- **Query Params:** `limit`, `offset`
- **Response:**

```
[{
  "id": "uuid",
  "module": "tarot",
  "created_at": "2025-08-14T12:00:00Z",
  "summary": "..."}]
```

---

## POST /api/uploads

- **Beschrijving:** Upload foto (koffiedik, handpalm, etc.)
- **Authenticatie:** Vereist JWT
- **Response:** URL + ID van opgeslagen asset

---

## POST /api/subscriptions

- **Beschrijving:** Start of upgrade een abonnement via Stripe Checkout
- **Integratie:** Stripe Webhooks → `/api/webhooks/stripe`

---

## 3. Prisma Datamodel (excerpt)

```
model User {
  id          String    @id @default(uuid())
  email       String    @unique
  passwordHash String
  name        String?
  readings    Reading[]
  subscription Subscription?
  createdAt   DateTime  @default(now())
}

model Reading {
  id          String    @id @default(uuid())
  module      String
  input       Json
  output      Json
  user        User      @relation(fields: [userId], references: [id])
  userId      String
  createdAt   DateTime  @default(now())
}
```

---

## 4. Integratie AI Promptbibliotheek

- Elke module call haalt **prompt-template** + **tone-of-voice** instructies op.
  - Output wordt **JSON Schema gevalideerd** voordat het in DB wordt opgeslagen.
  - Fouten → terugsturen als `422 Unprocessable Entity` met validatiefouten.
- 

## 5. Webhookflows

### Stripe Webhook `/api/webhooks/stripe`

- Events: `checkout.session.completed`, `customer.subscription.updated`
- Acties: update `Subscription` model, verstuur bevestigingsmail.

### AI Monitoring Webhook `/api/webhooks/ai`

- Events: Output validatiefout, AI downtime.
  - Acties: Fallback prompt sturen of retry.
- 

## 6. Beveiliging & Rate Limits

- JWT verificatie op alle user-specifieke endpoints.
  - Admin endpoints alleen via IP allowlist.
  - Rate limiting per IP (bijv. 100 requests/15 min).
- 

## 7. Versiebeheer

- API versies via `/v1/`, `/v2/` prefix.
  - Schema-wijzigingen altijd met changelog en backwards compatibility.
- 

**Volgende stap:** Backend koppelen aan AI Promptbibliotheek en Content & Tone Guide zodat alle responses consistent en gevalideerd zijn.

---

## 11. Backend-integratie voor Prompts & JSON-schema's

### 11.1 Prompt Registry (storage + API)

**Doel:** centraal beheer van prompts (per module/taal/versie) + rollout via feature flags.

**Nieuwe Prisma-modellen (aanvulling)**

```

model PromptTemplate {
  id          String    @id @default(cuid())
  module      String    // tarot | coffee | numerology | astro | combine
  locale      String    // nl-NL | en-US | tr-TR
  version     String    // v1, v1.1
  system      String    // system prompt text
  userMask    String    // user prompt masker met placeholders
  isActive    Boolean   @default(true)
  createdAt   DateTime  @default(now())
  updatedAt   DateTime  @updatedAt
}

model PromptFlag {
  id          String    @id @default(cuid())
  templateId  String
  cohort      String    // e.g. 10pct_nl, beta_users
  enabled     Boolean   @default(false)
  createdAt   DateTime  @default(now())
}

```

## Nieuwe endpoints

GET /prompts?module=tarot&locale=nl-NL	-> actieve template(s)
POST /prompts	-> admin: nieuwe template
PATCH /prompts/{id}	-> admin: update (rollout)
GET /prompts/flags?module=tarot&locale=nl-NL	-> feature flags
POST /prompts/{id}/flags	-> admin: flag (cohort)

## 11.2 Schema-validatie middleware

### Flow

1. Client -> /readings/\* endpoint.
2. Backend kiest **PromptTemplate** op basis van module, locale, version /flags.
3. Call naar LLM/Vision.
4. **JSON Schema validator** (AJV/Zod) valideert output tegen module-schema.
5. Bij validatiefout: retry met **repair-prompt** (één poging) → anders 422 UNPROCESSABLE\_ENTITY met error.details.

### Fouten

- 422\_INVALID\_JSON
- 422\_SCHEMA\_MISMATCH
- 424\_AI\_PROVIDER\_ERROR

## 11.3 Integratie Content & Tone Guide

- tone veld (NL/EN/TR) meegegeven aan prompt.
- metaphorBudget (max 3), noClaims toggles → prompt placeholders.

- Config in `.env` of adminpaneel voor toonprofielen per taal.

## 11.4 OpenAPI aanvullingen (excerpt)

```
paths:
  /prompts:
    get:
      summary: List active prompts
      parameters:
        - in: query
          name: module
          schema: { type: string, enum: [tarot, coffee, numerology, astro,
combine] }
        - in: query
          name: locale
          schema: { type: string }
      responses:
        '200': { description: OK }
    post:
      summary: Create prompt template (admin)
      responses:
        '201': { description: Created }
  /prompts/{id}:
    patch:
      summary: Update prompt template (admin)
      responses:
        '200': { description: OK }
  /readings/{module}:
    post:
      summary: Create reading with schema validation
      responses:
        '200': { description: OK }
        '422': { description: Schema/JSON invalid }
        '424': { description: AI provider error }
```

## 11.5 Backend implementatienotities

- **Service layer:** `PromptService.getTemplate(module, locale, flags)` met cache (Redis, 5–15 min TTL).
- **Validation:** AJV met pre-gecompileerde schema's; fallback Zod bij changes.
- **Repair-prompt:** voeg instructie toe "herformuleer output strikt volgens schema X, zonder extra tekst".
- **Observability:** log parse/validatie-fouten met `templateId`, `model` en `latency` (geen PII).

## 11.6 Evaluatie & quality loop

- **Golden outputs** in repo; nightly job vergelijkt AI-resultaten.
- **Scorekaart:** JSON-valid, leesbaarheid, consistentie, contradiction-coverage.
- **Auto-rollback:** bij degradatie > threshold, disable nieuwste template/flag.

## 11.7 I18n & locale fallback

- Eerst exacte match ( `nl-NL` ), anders `nl` , anders `en-US` (default).
- Locale mee in elke request/response; in DB bewaren op sessie.

## 11.8 Security

- Admin-routes achter RBAC + IP allowlist.
- Prompts bevatten geen secrets; templates geëncrypt in DB optioneel.
- Rate limiting op `/readings/*` en `/prompts/*`.