WIP-2

Eindopdracht

Leerlijn AI Developer (30 EC)

WORK IN PROGRESS!

Eindopdracht AI Developer

Integrale eindopdracht

De leerlijn AI-Developer bevat de curssussen AI Fundamentals, Technical Foundations, Integration & AI Tooling en Advanced AI Features & Production.

Om deze leerlijn af te ronden dien je de volgende leeruitkomsten aan te tonen:

1. **AI Fundamentals**
   De student zet generatieve AI doelgericht in bij het ontwerpen, uitwerken en automatiseren van technische en creatieve workflows en reflecteert kritisch op de betrouwbaarheid, ethiek en effectiviteit van AI-gebruik.
2. **Technical Foundations**
   De student bouwt schaalbare en professionele full-stack webapplicaties die modern getypeerd, goed gestileerd, gebruiksvriendelijk en veilig zijn.
3. **Integration & AI tooling**
   De student integreert AI-tools (zoals Cursor en Skills.sh) in het eigen ontwikkelproces, werkt doelgericht en iteratief aan functionerende prototypes en brengt technieken en workflows succesvol samen tot een werkend product.
4. **Advanced AI Features & Production**
   De student ontwerpt, realiseert en documenteert een professionele AI-first webapplicatie die complexe AI-features bevat (zoals agents, multi-step taken en contextmanagement), koppelt externe bronnen en laat zien het hele ontwikkelproces te kunnen overzien en uitdragen.

Algemene opdrachtbeschrijving

In de hedendaagse softwareontwikkeling worden AI-tools een essentieel onderdeel van het ontwikkelproces. Organisaties verwachten van developers dat zij niet alleen AI gebruiken als hulplijn, maar deze ook als volwaardig onderdeel in hun producten integreren. Met AI-Driven Development ben jij klaar voor dit nieuwe tijdperk. In dit afsluitende project breng je alles samen wat je geleerd hebt en bouw je zelfstandig een professionele, volwaardige AI-powered webapplicatie.

Als ontwikkelaar ga je laten zien dat jij moderne technieken en AI-integraties beheerst. Dit betekent: professioneel werken met AI-tools als Cursor en Skills.sh, databases ontwerpen en beheren met Supabase, AI-features bouwen met de Vercel AI SDK (agents met tools en externe data), alles gedocumenteerd én live in productie.

Voor deze eindopdracht bouw je een volledige webapplicatie met AI-integratie.

Om de leerlijn AI-developer met succes af te ronden is een voldoende nodig voor de integrale eindopdracht; met de deelopdrachten kunnen geen losse cursussen worden afgerond.

Op te leveren producten

- Broncode van een Next.js webapplicatie (gedeployed via Vercel)
- Een ingerichte Supabase backend
- Technisch verantwoordingsdocument
- Demo van de werkende applicatie (live of als videopresentatie)

Deelopdrachten

We gaan jouw vaardigheden als AI-ontwikkelaar toetsen door een full stack applicatie te bouwen waarin je externe data combineert met een intelligente AI-gedreven functionaliteit. Je ontwikkelt features waarbij AI bestaande data interpreteert of verrijkt en slaat deze informatie op in een zelf ontwikkelde cloud-backend met Supabase. Je past hierbij alle technieken toe die in de lessen zijn behandeld, waaronder Next.js met TypeScript, Supabase (database, authenticatie en RLS), Vercel AI SDK, deployment naar Vercel en maakt gebruik van ondersteunende tools zoals Cursor en Skills.

Voordat je begint met programmeren, werk je eerst jouw idee uit. Je formuleert duidelijke AI-instructies (bijvoorbeeld in de vorm van system prompts of agent-gedrag) en maakt onderbouwde keuzes die je gedurende het ontwikkelproces blijft documenteren. Uiteindelijk presenteer je jouw eindresultaat in een video. Al deze stappen zijn opgedeeld in deelopdrachten.

Voordat je kunt starten met de eerste deelopdracht lever je eerst jouw casus in ter goedkeuring bij de docent. Je mag ervoor kiezen om de wensen van de voorbeeldcasus te vertalen naar een idee voor jouw applicatie, of om een eigen casus te bedenken. In dat geval ga je eerst op zoek naar een passende API en bedenk je hoe jij deze data kunt gebruiken binnen jouw casus.

Je beschrijft in maximaal 250 woorden:

- Welk probleem jouw applicatie oplost;
- Wat jouw AI-interpretatie van de applicatie is (wat doet de AI concreet met de data?);
- Welke API je hiervoor gaat gebruiken (inclusief link naar de documentatie);
- Wat de 4 belangrijkste functionaliteiten van jouw applicatie zullen zijn. Hierin is één functionaliteit al vergeven aan registreren en inloggen en moet minimaal één functionaliteit een AI-integratie bevatten.

Na goedkeuring van de docent over jouw probleemstelling en de oplossing die jij daarvoor in gedachte hebt, kun je aan de slag met de eerste deelopdracht.

Voorbeeldcasus

VOORBEELDPROJECTEN:
Je mag zelf bepalen wat je bouwt, zolang je aan alle technische eisen voldoet. **Belangrijk**: je applicatie moet AI combineren met externe data via een gratis API. De AI moet de data **interpreteren en analyseren**, niet alleen doorsturen.

Hieronder drie voorbeelden ter inspiratie:

- **Cocktail Sommelier**
  Een AI-powered cocktail advisor die helpt bij het kiezen en maken van drinks.

- **Recipe remix**
  Een slimme recepten-assistent die helpt met koken op basis van wat je in huis hebt of wat je lekker vindt.
- **Quiz Master AI**
  Een adaptieve quiz app met een AI die moeilijkheid aanpast en uitlegt.

Beschikbare Gratis API's

API	Data	URL
TheCocktailDB	Cocktails & recepten	thecocktaildb.com/api.php
TheMealDB	Maaltijden & recepten	themealdb.com/api.php
Open Trivia DB	Quiz vragen	opentdb.com/api_config.php
REST Countries	Landen informatie	restcountries.com
Open Library	Boeken data	openlibrary.org/developers

Wil je liever een eigen casus bedenken? Je mag een eigen projectidee uitwerken met een andere gratis API, mits het voldoet aan alle technische eisen gesteld in Deelopdracht 1.

Deelopdracht 1. Broncode Next.js project

Een succesvolle AI-gedreven applicatie begint met een solide fundament. Daarom richt je eerst een gestructureerde projectmap in. Je zet een Next.js-project op met TypeScript en Tailwind CSS en configureert AI-tools zoals Cursor en Skills.sh. Daarnaast beheer je jouw project vanaf het begin met Git, zodat de ontwikkeling van je applicatie goed te volgen is.

De kern van deze opdracht is het ontwikkelen van een AI-gedreven webapplicatie waarin externe data op een intelligente manier wordt verwerkt en gepresenteerd aan de gebruiker. Allereerst wordt relevante data opgehaald uit een externe API. Op basis van de input van jouw gebruiker stuur je deze data naar een LLM via de Vercel AI SDK. Het model analyseert de beschikbare data, combineert deze met de intentie van de gebruiker en voert een gerichte taak uit, zoals het filteren, interpreteren of structureren van informatie. Het resultaat van deze verwerking wordt vervolgens teruggestuurd naar de applicatie en op een duidelijke manier gepresenteerd in de UI. De verreikte data sla je – waar nodig – op in je eigen database.

Je bouwt hiermee feitelijk een eigen AI-functionaliteit bovenop bestaande data. Dit betekent dat je niet alleen verantwoordelijk bent voor het ophalen van data, maar ook voor hoe deze data wordt geïnterpreteerd, welke instructies je aan het model geeft en hoe het eindresultaat tot stand komt. De logica achter deze flow werk je uit in code en in de prompts die je schrijft. Tijdens het ontwikkelproces documenteer je deze keuzes, prompts en instructies zorgvuldig. Deze documentatie vormt een belangrijk onderdeel van je Verantwoordingsdocument (deelopdracht 3).

Jouw applicatie heeft minimaal de volgende eigenschappen:

- De applicatie is een fullstack webapplicatie die je hebt gebouwd met behulp van Cursor. De frontend bouw je in Next.js met TypeScript en Tailwind CSS en de backend in Supabase.
- Er wordt gebruikgemaakt van externe data die je verreik met AI om functionaliteiten te realiseren. Deze data moet worden opgehaald door middel van netwerk requests naar een API en resultaten worden opgeslagen in jouw eigen database.
- De applicatie bevat minimaal 4 kernfunctionaliteiten, waaronder registreren en inloggen, minimaal één AI-gedreven functionaliteit (agent-based flow) en twee aanvullende functionaliteiten gebaseerd op jouw casus. Bijvoorbeeld:
    - Kernfunctionaliteit 1: Registreren en inloggen
    - Kernfunctionaliteit 2: Bibliotheekboeken kunnen doorzoeken en filteren met AI
    - Kernfunctionaliteit 3: Bibliotheekboeken kunnen beheren (bestaande boeken aanpassen, nieuwe boeken toevoegen, genres koppelen)
    - Kernfunctionaliteit 4: Uitleningen kunnen aanmaken en beheren
- Je maakt gebruik van minimaal 2 Vercel skills (Open Agent Skills Ecosystem via skills.sh);
- Er zijn minimaal 3 tools gedefinieerd binnen de agent. Deze voert zelfstandig meerdere stappen uit met maxSteps
- Het project/repository is opgebouwd volgens de structuur zoals behandeld in de leerlijn (zie hoofdstuk Structuur)
- Het project is gedeployed naar een productie-omgeving via Vercel;
- Je maakt gebruik van Git om jouw projectvoortgang te beheren en zet het project direct op GitHub. De eerste commit bevat jouw initiele projectopzet en configuratie. Je zorgt voor kleine beschrijvende commits, maakt pull requests voor iedere nieuwe feature en mergt regelmatig naar de main branch.

Op te leveren:

- De projectmap met daarin de broncode van de Next.js applicatie, inclusief README.md met installatie-instructies;
- De link naar de Github repository en live productie-omgeving in het Verantwoordingsdocument*;*

Deelopdracht 2. Supabase cloud-backend met authenticatie

Nadat je de basis van je applicatie hebt opgezet, richt je je op de backend. In deze fase staat het modelleren van je database centraal. Je ontwerpt een logisch gestructureerd datamodel waarin entiteiten en hun onderlinge relaties duidelijk zijn vastgelegd. Hierbij houd je rekening met uitbreidbaarheid, dataconsistentie en performance.

Je implementeert dit datamodel in Supabase door tabellen aan te maken, relaties te definiëren en constraints toe te passen. Supabase fungeert hierbij als jouw cloud-backend, waarin databasefunctionaliteit, API-toegang en authenticatie samenkomen.

Vervolgens koppel je jouw Next.js-applicatie aan deze backend. Je zorgt ervoor dat data vanuit de frontend op een gestructureerde manier wordt opgehaald, aangepast en opgeslagen. Hiervoor maak je gebruik van React Query (TanStack Query). Met behulp van Supabase Auth zorg je ervoor dat gebruikers zich kunnen registreren en inloggen. Iedere gebruiker krijgt een eigen identiteit binnen het systeem. Om ervoor te zorgen dat gebruikers alleen toegang hebben tot hun eigen gegevens, configureer je Row Level Security (RLS). Je definieert policies die bepalen welke data een gebruiker mag lezen, toevoegen, aanpassen of verwijderen. Dit betekent concreet dat je in je database een koppeling maakt tussen data en de ingelogde gebruiker (bijvoorbeeld via een `user_id`), en dat je policies opstelt die deze relatie afdwingen.

De technische keuzes die je maakt, zoals de structuur van je database, de manier waarop je relaties legt en hoe je RLS toepast, licht je toe uit in je Verantwoordingsdocument (deelopdracht 3).

Jouw Supabase backend heeft minimaal de volgende eigenschappen:

- Bevat een database schema met minimaal 4 tabellen;
- De tabellen zijn logisch opgebouwd en bevatten passende relaties (foreign keys) en contrains (bijv. primary keys, not null, unique);
- Voor alle tabellen zijn werkende CRUD-operaties geïmplementeerd met React Query;
- Authenticatie is geïmplementeerd met Supabase Auth (gebruikers kunnen zich registreren en inloggen en sessies worden correct afgehandeld in de applicatie
- Row Level Security (RLS) is ingeschakeld op relevante tabellen en je hebt policies gedefinieerd die ervoor zorgen dat gebruikers alleen hun eigen data kunnen benaderen en aanpassen

Op te leveren:

- Persoonlijke base URL van jouw Supabase REST API in README.md;
- Schemarepresentatie van de database (via de Schema Visualizer) in het Verantwoordingsdocument.

Deelopdracht 3. Verantwoordingsdocument

Tijdens het ontwikkelen van jouw applicatie schrijf je ook een technisch verantwoordingsdocument. In deze leerlijn werk je intensief met AI-tools zoals Cursor en Claude. Dit maakt het mogelijk om snel complexe functionaliteit te bouwen, maar vraagt ook een kritische en bewuste houding. In dit document laat je zien dat je begrijpt wat je hebt gebouwd, hoe jouw applicatie technisch werkt en welke keuzes je hebt gemaakt tijdens het ontwikkelproces. Het is daarom slim om tijdens het ontwikkelen al te beginnen met documenteren.

Je gebruikt concrete voorbeelden uit je eigen project, zoals code snippets, prompts en configuraties om je keuzes en leermomenten te onderbouwen. Zo beschrijf je hoe je data verwerkt met jouw AI-agent en hoe deze de data interpreteert in combinatie met gebruikersinput. Je beschrijft hoe je jouw database schema hebt opgebouwd en hoe je met Row Level Security (RLS) hebt gezorgd voor veilige, user-specifieke data. Je laat zien hoe je prompts hebt ingezet om functionaliteit te bouwen, hoe je iteratief hebt gewerkt aan het verbeteren van de agent-instructies en waar je tegenaan bent gelopen. Ook onderbouw je waarom bepaalde aanpakken wel of niet werkten en wat je daaruit meeneemt in toekomstige projecten.

Jouw verantwoordingsdocument geeft minimaal antwoord op de volgende vragen:

- **Technische Implementatie AI-functionaliteit**
  Beantwoord onderstaande vragen door ze tekstueel toe te lichten én te onderbouwen met *concrete* codevoorbeelden.
    1. Hoe haalt jouw applicatie data op uit de externe API?
    2. Waar en hoe wordt deze data verwerkt voordat deze naar het model gaat?
    3. Hoe ziet jouw input naar het model eruit? *(prompt + data)*
    4. Hoe wordt de output van het model verwerkt in jouw applicatie?
    5. Hoe heb je de stappen binnen jouw AI-agent met maxSteps ingericht? (bijv. met maxSteps)?
    6. Welke tools en skills heb je geïntegreerd binnen de agent en waarom?
    7. Hoe heb je foutafhandeling en fallback-mechanismen ingericht binnen de AI-flow?
    8. Welke ontwerpkeuzes heb je gemaakt om de autonomie, betrouwbaarheid en voorspelbaarheid van de AI-functionaliteit te waarborgen?
- **Database ontwerp en AI-gerealteerde dataverwerking**
    9. Laat visueel zien hoe jouw Supabase database eruit ziet met behulp van een duidelijk leesbaar screenshot van het ERD
    10. Licht jouw database-schema toe aan de hand van het ERD en beschrijf hoe de relaties tussen tabellen geïmplementeerd zijn. Denk aan primaire sleutels, foreign keys en kardinaliteiten.
    11. Hoe heb je Row Level Security (RLS) ingericht?
    12. Welke policies heb je geschreven en wat doen deze precies? Benoem *concrete* voorbeelden.
    13. Welke rol speelt jouw database in de AI-functionaliteit van jouw applicatie? Beschrijf hoe data uit jouw database wordt gebruikt als input voor het model en hoe modeloutput wordt opgeslagen.
    14. Welke risico's of beperkingen zie je in de combinatie van AI en jouw datamodel? Denk aan inconsistente data, foutieve AI-output die wordt opgeslagen, afhankelijkheid van datakwaliteit, etc.
    15. Hoe zou je jouw databaseontwerp aanpassen als je AI-functionaliteit betrouwbaarder of schaalbaarder moet worden?

- **AI-Assisted ontwikkelproces**
    16. Wat waren je initiële instructies aan Cursor bij de start van het project? Hoe heb je deze instructies gedurende het project aangepast of verbeterd? Welke concrete invloed had dit op je output?
    17. Welke gebruikte `.cursorrules` werkten goed en waarom?
    18. Hoe heb je gecontroleerd dat de gegenereerde code daadwerkelijk correct was en werkte
    19. In hoeverre heb je het gevoel dat je controle had over de gegenereerde code en het ontwikkelproces?
    20. In hoeverre heb je gegenereerde code in jouw project nog zelf moeten aanpassen of moeten herschrijven?
    21. Noem minimaal 3 concrete beperkingen of onzekerheden die je hebt ervaren in de output van het AI-model, en leg per geval uit wat de oorzaak was en wat het effect was op jouw applicatie.
    22. Welke maatregelen heb je genomen om de betrouwbaarheid van AI-output in jouw applicatie te waarborgen? Denk aan validatie, filtering, fallback, UI-feedback, etc.
    23. Als je een nieuwe student zou helpen met het gebruik van een tool als Cursor in softwareontwikkeling, welke drie belangrijkste inzichten of richtlijnen zou je diegene dan meegeven?
- **Prompt design**

  Beantwoord onderstaande vragen door ze tekstueel toe te lichten én te onderbouwen met concrete prompts. Deze vragen hebben betrekking op prompts die onderdeel zijn van jouw applicatie (zoals system prompts en input voor het model) en niet op prompts die je gebruikt hebt in Cursor tijdens het ontwikkelen.

    24. Welke belangrijke prompts heb je gebruikt om jouw kern-features te bouwen?
    25. Hoe zag een typische iteratie eruit bij jou? *(prompt → output → aanpassing → nieuwe prompt?)*
    26. Geef drie voorbeelden van prompts die heel goed werkten. Hoe komt dat volgens jou?
    27. Geef drie voorbeelden van prompts die minder goed werkten. Wat was hier de oorzaak van? En hoe heb je deze verbeterd?
    28. Welke onderdelen van jouw prompts hebben de grootste invloed gehad op het gedrag van het model, en waarom? Denk aan structuur, rolomschrijving, *constraints*, voorbeelden, etc.
    29. In hoeverre is de kwaliteit van de output afhankelijk van jouw prompt en inputdata?

Houdt bij het schrijven van dit document de structuur en ontwerpen aan zoals beschreven in het hoofdstuk Structuur. Blijf weg van algemeenheden en benoem zo concreet mogelijke voorbeelden uit jouw project of ervaring, wanneer je vragen beantwoord of onderbouwd. Je voegt een overzicht van *alle* gebruikte prompts en jouw uitgewerkte agent-instructie toe in de bijlage.

Op te leveren:

- Technisch verantwoordingsdocument (*.pdf*)

Deelopdracht 4. (Video)presentatie

In deze deelopdracht presenteer je jouw applicatie en toon je aan dat alle onderdelen daadwerkelijk werken zoals bedoeld. Dit kan in de vorm van een live presentatie of een vooraf opgenomen video. De nadruk ligt op het aantoonbaar demonstreren van functionaliteit. Je laat zien dat jouw applicatie technisch correct werkt, inclusief database, AI-functionaliteit en beveiliging. Let op: ondanks dat AI een grote rol speelt in deze leerlijn, ben je ten alle tijde zelf verantwoordelijk voor de kwaliteit van de opgeleverde code. Je moet in staat zijn om goed uit te leggen hoe jouw code werkt en is opgebouwd.

De presentatie duurt circa **8–12 minuten** en volgt onderstaande opbouw.

1. Start je project lokaal op en toon de globale structuur van jouw project (kort, max 30 – 60 seconden).

2. Laat de koppeling met de Supabase database zien.
3. Laat je Supabase-project zien in het dashboard. Toon minimaal:
    a. De aanwezige tabellen
    b. Relaties tussen tabellen
    c. Dat Row Level Security (RLS) is ingeschakeld
    d. Zoom kort in op één tabel met een `user_id` (of vergelijkbare koppeling) en geef één voorbeeld van een policy.
    e. Toon aan dat gebruikers alleen toegang hebben tot hun eigen data. Dus als gebruiker A een record aan zou maken, wat zorgt er dan voor dat gebruiker B die record niet kan zien of niet kan aanpassen of verwijderen?
4. Toon de applicatie op de productie URL.
5. Demonstreer alle dat alle vier kernfunctionaliteiten van jouw applicatie werken. Toon:
    a. Registreren en inloggen;
    b. De AI-functionaliteit (hierbij maak je goed zichtbaar wat de gebruiker invoert, hoe de applicatie reageert en wat het resultaat is);
    c. Twee overige functionaliteiten uit jouw casus

Beperk je hierin niet alleen tot de happy-flow: laat ook zien wat er gebeurt als een gebruiker verkeerde keuzes maakt of foutieve data ingeeft.

Op te leveren:

- (Video)presentatie van maximaal 12 minuten (live of *.mp4* of *.mov*)

Quickscan

Hieronder vind je een aantal randvoorwaarden waaraan de eindopdracht moet voldoen. De beoordelaar kan op basis van deze eisen de eindopdracht teruggeven en zal deze niet verder nakijken tot aan de eisen is voldaan.

Gebruik de quickscan om te zien of je voldoet aan de inlevereisen.

Algemene eisen

- ☐ Naam van de student, inleverdatum en leerlijntitel zijn opgenomen op de titelpagina van het verantwoordingsdocument
- ☐ Het technisch verantwoordingsdocument is digitaal ingeleverd als PDF
- ☐ De GitHub repository URL is bijgevoegd in het verantwoordingsdocument
- ☐ De Vercel productie URL is bijgevoegd in het verantwoordingsdocument
- ☐ Alle deelopdrachten zijn uitgewerkt en de gevraagde deelproducten zijn aanwezig
- ☐ Het verantwoordingsdocument is goed leesbaar zonder storende grammatica- en spellingsfouten
- ☐ De omvang van het verantwoordingsdocument beslaat 2500-3000 woorden (exclusief inhoudsopgave en bijlagen)
- ☐ De presentatie duurt maximaal 12 minuten
- ☐ Pagina's in het verantwoordingsdocument zijn genummerd

Inhoudelijke eisen

- ☐ Het project is geüpload naar een GitHub repository: deze repository staat op *public*.
- ☐ Skills.sh is geïnstalleerd met minimaal 2 actieve skills (waaronder vercel-react-best-practices)
- ☐ README.md bevat heldere installatie-instructies die ervoor zorgen dat een collega developer het project zelfstandig kan draaien
- ☐ De applicatie is geprogrammeerd Next.js 14 (App Router) met TypeScript en Tailwind CSS
- ☐ De applicatie is gekoppeld aan een Supabase database met minimaal 4 tabellen
- ☐ Het database schema bevat relaties en authenticatie
- ☐ CRUD-operaties zijn geïmplementeerd met React Query
- ☐ De Vercel AI SDK is geïmplementeerd met werkende AI Agent
- ☐ Minimaal 1 externe API is geïntegreerd via Tool Calling
- ☐ Minimaal 3 tools zijn gedefinieerd met Zod parameters
- ☐ Agent heeft maxSteps van minimaal 3
- ☐ AI interpreteert/combineert data (niet alleen doorsturen)
- ☐ Streaming responses zijn geïmplementeerd
- ☐ De applicatie is gedeployed op Vercel en bevat vier werkende functionaliteiten

AI gebruik

- ☐ Het gebruik van AI is toegestaan en verwacht bij deze eindopdracht
- ☐ Alle AI-prompts en uitgewerkte agent-instructie zijn toegevoegd als bijlage in het verantwoordingsdocument
- ☐ Architectuurbeslissingen gemaakt met AI-hulp zijn gedocumenteerd

Structuur

Technisch verantwoordingsdocument

Voor het technisch verantwoordingsdocument wordt het aanbevolen de volgende structuur te volgen:

```
project-naam/
├── .cursor/
│   └── rules/
│       ├── general.mdc          # Algemene project regels
│       ├── components.mdc       # Component conventies (optioneel)
│       └── api.mdc              # API/Supabase regels (optioneel)
├── .skills/                     # Skills.sh configuratie
├── docs/
│   ├── PROJECT-BRIEF.md         # Project beschrijving
│   ├── PROMPT-LOG.md            # Gedocumenteerde prompts
│   └── AI-DECISIONS.md          # Architectuur beslissingen
├── src/
│   ├── app/                     # Next.js App Router
│   │   ├── api/
│   │   │   └── agent/
│   │   │       └── route.ts     # AI Agent API route (met tools)
│   │   ├── (auth)/              # Auth route group
│   │   │   ├── login/
│   │   │   └── register/
│   │   ├── layout.tsx
│   │   └── page.tsx
│   ├── components/
│   │   ├── ui/                  # Basis UI components
│   │   ├── layout/              # Layout components
│   │   └── features/            # Feature-specifieke components
│   ├── hooks/
│   │   ├── useAuth.ts
│   │   └── [custom hooks]
│   ├── lib/
│   │   ├── supabase.ts          # Supabase client
│   │   ├── utils.ts
│   │   └── queryClient.ts       # React Query client
│   └── types/
│       ├── database.ts          # Supabase types
│       └── index.ts
├── .env.example                 # Template voor env vars
├── .env.local                   # Lokale env vars (NIET committen)
├── .gitignore
├── next.config.js
├── package.json
├── README.md
├── tailwind.config.ts
└── tsconfig.json
```

Titelblad

Vermeld op het titelblad in elk geval:

- Naam
- Inleverdatum
- Titel leerlijn en eventuele ondertitel
- GIthub URL
- Vercel URL

Samenvatting

Een korte samenvatting (max 150 woorden) van wat je hebt gebouwd.

Inhoudsopgave

Voeg een overzichtelijk inhoudsopgave toe met correcte paginanummering. Vermeld hier ook de eventuele bijlagen.

1.Introductie

Beschrijf het probleem dat je oplost, wie jouw doelgroep is en wat de belangrijkste features van jouw applicatie zijn. *Tip:* voeg 1 tot 3 screenshots van de UI toe ter illustratie.

2. Technische Implementatie AI dataflow

Beantwoord de vragen uit de deelopdracht door ze tekstueel toe te lichten én te onderbouwen met concrete codevoorbeelden.

3. Database ontwerp

Voeg een screenshot van het ERD van de database toe en geef antwoord op de vragen uit de deelopdracht.

4. AI-Assisted Development Proces

Beantwoord de vragen over jouw ontwikkelproces met AI en de prompts die je hebt toegepast. Doe dit door de antwoorden tekstueel toe te lichten én te onderbouwen met concrete prompts.

5. Conclusie

Beschrijf wat je hebt geleerd over het werken met AI. Welke aspecten van AI vind jij een toevoeging voor je workflow en over welke aspecten ben je nog kritisch?

Bijlagen

- Alle gebruikte prompts binnen de applicatie, zoals system prompts en input voor het model;
- De volledige uiteindelijke agent-instructie zoals geïmplementeerd in de applicatie;
- De gebruikte Cursor instructies tijdens het ontwikkelproces (zoals Cursor-prompts en/of `.cursorrules`)

Repository structuur

Je project-repository moet de volgende structuur hebben:

Beoordelingscriteria

De eindopdracht wordt beoordeeld op basis van de volgende beoordelingscriteria. Per criterium kent de beoordelaar een aantal punten toe.

Deelopdrachten/ producten		***Weging***
1. Broncode Next.js project		**25 %**
Criterium 1.1	De student gestructureerd het Next.js-project op correcte wijze met TypeScript en Tailwind CSS zodat het correct functioneert in zowel de development- als productieomgeving (VerceL) zonder kritieke runtime errors.	5%
Criterium 1.2	De student haalt externe data correct op via netwerkrequests naar een API en interpreteert en verrijkt de data via een LLM in combinatie met gebruikersinput op passende wijze.	5%
Criterium 1.3	De student implementeert een agent-based flow met minimaal 3 tools, minimaal 2 geïntegreerde skills en een werkende meerstaps-aanpak (maxSteps), waarbij de agent zelfstandig meerdere stappen uitvoert om tot een betekenisvol resultaat te komen.	5%
Criterium 1.4	De student schrijft met behulp van AI schone, gestructureerde en onderhoudbare TypeScript- en Next.js-code, waarbij componenten, server- en clientlogica, utilities en databewerkingen logisch zijn opgezet en clean code-principes zichtbaar worden toegepast.	5%
Criterium 1.5	De student beheert de code met correct van Git, maakt hierbij gebruik van minimaal 20 kleine commits, met compacte zinvolle commits messages en maakt minimaal 5 pull requests die naar de main branch gemerged zijn.	5%
2. Database Design & Supabase		**15%**
Criterium 2.1	De student ontwerpt en implementeert een logisch databaseschema in Supabase met minimaal 4 tabellen, passende relaties en correcte constraints zoals primary keys, foreign keys, not null en unique.	5%
Criterium 2.2	De student implementeert werkende CRUD-operaties voor de relevante entiteiten en maakt daarbij op correcte wijze gebruik van React Query voor databeheer, caching en synchronisatie met de UI.	5%

Criterium 2.3	De student implementeert authenticatie met Supabase Auth op correcte wijze, past Row Level Security correct toe op relevante tabellen en definieert werkende policies waarmee gebruikers uitsluitend hun eigen data kunnen lezen, toevoegen, aanpassen en verwijderen.	5%
3. Verantwoording		**45%**
Criterium 3.1	De student beschrijft en onderbouwt op inhoudelijk correcte, gestructureerde en diepgaande wijze de technische implementatie van de AI-flow binnen de applicatie en onderbouwt keuzes met concrete voorbeelden.	10%
Criterium 3.2	De student onderbouwt het databaseontwerp en de beveiliging op inzichtelijke wijze aan de hand van een duidelijke en correcte schemarepresentatie (ERD) en reflecteert op de risico's en beperkingen van de combinatie van AI en data opslag.	10%
Criterium 3.3	De student ontwerpt het gedrag van de agent op zorgvuldige wijze (inclusief gebruik van maxSteps, system prompts en foutafhandeling) en toont aan dat er bewuste keuzes zijn gemaakt om autonomie, betrouwbaarheid en voorspelbaarheid van de AI-functionaliteit te waarborgen.	10%
Criterium 3.4	De student analyseert en onderbouwt op kritische en iteratieve wijze het gebruik van prompts binnen het ontwikkelproces en toont inzicht in de invloed van promptstructuur, context en specificiteit op modelgedrag.	10%
Criterium 3.5	De student reflecteert op kritische en zelfbewuste wijze op het gebruik van AI-tools tijdens het ontwikkelproces en formuleert concrete leerpunten en verbeterstrategieën voor toekomstig gebruik van AI in softwareontwikkeling om overmatig vertrouwen in gegenereerde code te beperken.	10%
5. Presentatie		**10%**
Criterium 5.1	De student implementeert vier belangrijke kernfunctionaliteiten op correcte wijze, waaronder het authenticatieproces van registreren en inloggen, minimaal één complete AI-use case en twee aanvullende, zelfbedachte en volledig uitgewerkte use cases passend bij de gekozen casus.	10%
Totaal		100%