# Prompt Engineering 3

## *1, Introductie*

We hebben in prompt engineering 1 en 2 geleerd waaraan een goede prompt moest voldoen.

Dit zijn de basis kenmerken van een goede prompt. De eerste drie kenmerken moet je prompt **altijd** aan voldoen!

1. **Context** - een goede prompt heeft voldoende contexrt.
2. **Details/Specifiek** - een goed prompt heeft voldoende details en is zo specifiek mogelijk.
3. **Duidelijkheid** - een goede prompt is concreet en bevat geen vage termen zoals (mooi, groot, .....), zei hier.
4. **Doelgericht** - een goede prompt is doelgericht.
5. **Vorm** - in een goede prompt kan je de output in een bepalade vorm vragen.
6. **Toon** -door in de prompt de toon op te nemen, bepaal je de vorm van het antwoord.

In deze module gaan we nog meer **advanched prompt-technieken** leren. Deze technieken heb je niet altijd nodig maar het is handig om deze technieken te kennen.

Bovendien zijn de meeste techniekken ook toepasbaar in als '**problem solving**' technieken.

1. **Gebruik een rol (role prompting)**
   Laat de AI een specifieke rol of expertise aannemen (bijv. leraar, programmeur, leerling).
2. **Vraag om alternatieven of variaties**
   Stimuleer de AI om meerdere ideeën of invalshoeken te geven.
3. **Iteratief verbeteren**
   Vraag de AI om haar eigen antwoord te verbeteren of herschrijven.

4. **Gebruik prompt templates**
   Werk met vaste structuren zoals: context, doel, outputvorm, toon.
5. **Vraag om zelfvragen (self-questioning)**
   Laat de AI zelf bedenken welke vragen nodig zijn om het probleem goed op te lossen.
6. **Gebruik kritiek als leermoment (prompt debugging)**
   Laat de AI uitleggen waarom een prompt niet goed werkte en hoe je die kunt verbeteren.

# *1. Role Prompting*

## Uitleg

Je kunt de AI vragen om te reageren **alsof ze een bepaalde rol heeft**. Bijvoorbeeld:

- een ervaren programmeur
- een beginnende student
- een HTML-docent
- een code-reviewer

Door een rol te geven, pas je automatisch de **toon, moeilijkheidsgraad en stijl** van het antwoord aan. Dit helpt je om antwoorden te krijgen die beter passen bij jouw doelgroep of bij jouw situatie.

## Voorbeeld

### Prompt

> *Leg uit wat een `<form>`-element is in HTML, alsof je een docent bent die het uitlegt aan een groep mbo-studenten zonder programmeerervaring.*

### Verwachte AI-reactie

> Een `<form>` is een stuk HTML-code waarmee gebruikers informatie kunnen invullen, zoals hun naam of e-mailadres. Denk aan een online formulier dat je invult om iets te bestellen. De informatie die je invult wordt verstuurd naar de website zodat er iets mee gedaan kan worden.

# Laat de AI vragen stellen als een docent

Je kunt ook aan de AI vragen om **deelvragen** te stellen, zoals een goede docent dat zou doen.

## Prompt

> *Speel de rol van een HTML-docent. Stel mij eerst drie vragen waarmee je kan inschatten hoeveel ik al weet over formulieren in HTML. Geef daarna pas een uitleg op mijn niveau.*

## Wat dit doet

- De AI stelt vragen zoals: “Wat denk je dat een `<form>` doet?”, “Heb je eerder inputvelden gebruikt?”
- De AI past daarna de uitleg aan op basis van jouw antwoorden (of een inschatting daarvan)

# Opdracht

Je wilt dat de AI uitleg geeft over een klein stukje code, maar dan **in een specifieke rol**. Bijvoorbeeld: als docent, als programmeermaatje, of als code-reviewer.

1. Je bent een student die uitleg wil over hoe een formulier werkt in HTML
   Je wilt dat de AI zich gedraagt als een docent die eerst **deelvragen** stelt voordat hij iets uitlegt
2. **Schrijf een prompt** waarin je duidelijk de rol beschrijft die de AI moet aannemen.

# Inleveren

1. Je volledige prompt
2. Het AI-antwoord

# *2. Vraag om alternatieven of variaties*

# Uitleg

De AI geeft standaard vaak **slechts één oplossing**. Maar in de praktijk zijn er meestal meerdere manieren om iets te doen — zeker in programmeertalen als HTML, CSS, PHP of JavaScript.

Je kunt dus expliciet vragen om **meerdere oplossingen** of **variaties**, zoals:

- “Geef 3 manieren om een knop te centreren met CSS.”
- “Laat 2 varianten zien van hetzelfde formulier: één minimalistisch en één met veel opmaak.”
- “Bedenk 3 manieren om input te valideren in JavaScript.”

## Voordelen

- Je leert alternatieve technieken kennen
- Je kunt zelf kiezen welke stijl of methode bij jouw project past
- Je voorkomt dat je AI-antwoorden blind overneemt zonder kritisch te zijn

# Voorbeeld

## Prompt

*Geef 3 manieren om een knop in het midden van het scherm te tonen met behulp van CSS.*

## Verwachte AI-reactie (beknopt)

1. `text-align: center` op de container + `display: inline-block` op de knop
2. `margin: 0 auto` bij een block-level knop met vaste breedte
3. Flexbox op de container met `justify-content: center` en `align-items: center`

# Opdracht

Je wil verschillende manieren zien om een veelvoorkomende webtaak uit te voeren.

1. Kies één van deze taken:
    - Validatie toevoegen aan een inputveld, of

- Een PHP-script schrijven dat data toont

2. **Schrijf een prompt** waarin je de AI vraagt om **minstens 3 alternatieven of varianten** van die oplossing.
3. Gebruik in je prompt woorden als:
   - “Geef meerdere mogelijkheden…”
   - “Laat minstens 3 manieren zien…”
   - “Toon varianten van deze code…”
   - "Beschrijf van elke variant de voor- en nadelen...."
4. Zorg voor een goede opbouw in je prompt met:
   - **Context:** wat ben je aan het doen en welke technieken gebruik je?
   - **Details:** wat zijn de randvoorwaarden?
   - **Duidelijkheid:** hoeveel oplossingen wil je?

# Inleveren

1. Je volledige prompt
2. De gegenereerde alternatieven van de AI
3. Een korte reflectie (reflectie.txt): welke oplossing vind jij het beste, en waarom?

# *3. Iteratief verbeteren*

# Uitleg

De AI geeft vaak een eerste versie van een antwoord, maar dat hoeft niet perfect te zijn. Je kunt de AI daarom vragen om **het eigen antwoord te beoordelen en te verbeteren**.

Dit heet *iteratief verbeteren*.

## Bijvoorbeeld

- “Herschrijf dit zodat het eenvoudiger is voor een beginner.”
- “Geef dezelfde code, maar nu met uitleg erbij.”
- “Kun je deze oplossing optimaliseren voor leesbaarheid?”

Je leert zo hoe je feedback kunt geven aan de AI, en je oefent tegelijkertijd **zelfkritisch denken over codekwaliteit**.

# Voorbeeld

## Prompt

> *Geef me een formulier in HTML waarmee iemand zich kan inschrijven voor een nieuwsbrief. Voeg daarna uitleg toe over hoe elk onderdeel werkt.*

## Vervolgprompt

> *Herschrijf je antwoord zodat het ook begrijpelijk is voor iemand die nog nooit HTML heeft gezien.*

# Opdracht

Je wilt dat de AI **het eigen werk verbetert** — of dat jij feedback geeft, en dat de AI dat verwerkt.

1. Vraag de AI eerst om een korte opdracht uit te voeren. Bijvoorbeeld:
   - Een stukje JavaScript schrijven dat controleert of een naam de juiste hoofdletters en punten heeft. (Dus goede namen zijn: M. Bisschop en J. K. van der Velden, onjuiste namen zijn M Bisschop (punt mist) en k. van der Velden (k is gene hoofdletter).
2. Lees het antwoord goed door. Stel daarna een vervolgprompt waarin je de AI vraagt om:
   - het duidelijker te maken,
   - het aan te passen voor beginners,
   - het te herschrijven met uitleg/commentaar,
   - of het korter, leesbaarder of efficiënter te maken.
3. **Schrijf beide prompts uit:**
   - de eerste vraag, gebruik hierbij de few-shot prompt techniek (voorbeelden).
   - de verbeteropdracht

4. Zorg dat je feedback duidelijk is: **wat wil je precies anders of beter?**
5. **Zorg ervoor dat je de code begrijpt.**

# Inleveren

1. Je eerste prompt
2. Het eerste AI-antwoord
3. Je tweede prompt (de verbeteropdracht)
4. Het verbeterde AI-antwoord

Deze opdracht laat je zien aan een docent. Je laat zien wat je eerste- en twede eprompt is en je kunt zelf uitleggen hoe de code werkt.

---- vanaf hier nog corrigeren/aanvullen ----

# *4 Gebruik prompt templates*

## Uitleg

Als je regelmatig AI gebruikt voor soortgelijke taken, is het handig om te werken met een **vaste structuur**: een *prompt template*.

Een template helpt je om:

- niets te vergeten in je vraag;
- sneller en consistenter te werken;
- betere output te krijgen van de AI.

Een goede prompt bestaat vaak uit onderdelen zoals:

- **Context** – wat is de situatie?
- **Specifiek** - wat wil precies je bereiken?
- **Duidelijk -**zorg dat je prompt geen 'vage' aanwijzingen heeft
- **Outputvorm** – in welke vorm wil je het antwoord?
- **Toon** – formeel, informeel, eenvoudig, technisch?

## Voorbeeld van een prompt template

> Context: Ik werk aan een HTML-formulier
>
> Doel: Ik wil dat de AI me helpt met het maken van een formulier voor e-mailinvoer
>
> Outputvorm: Alleen HTML-code, zonder uitleg
>
> Toon: Neutraal en duidelijk

**Voorbeeldprompt:**
*Gebruik onderstaande context en geef alleen de code:*

*Ik wil een formulier voor e-mailadres, zonder uitleg.*
*Context: HTML-formulier*
*Output: HTML-code*

# Opdracht

Je gaat zelf een prompt schrijven met behulp van een prompt-template.

1. Gebruik het volgende sjabloon en vul de velden in:

   ```
   Context:
   Doel:
   Outputvorm:
   Toon:
   ```

2. Schrijf daarna een **volledige prompt** op basis van jouw ingevulde template.
3. Stuur je prompt naar de AI en bekijk het resultaat:
   - Krijg je precies wat je bedoelde?
   - Is de toon en vorm correct?
   - Zou je iets willen toevoegen aan je template?
4. (Optioneel) Pas je template aan en probeer het nog eens.

# Inleveren

1. Je ingevulde template
2. De volledige prompt
3. De output van de AI
4. Een korte reflectie: werkte het goed? Wat zou je verbeteren aan het sjabloon?

# *5. Vraag om zelfvragen (self-questioning)*

# Uitleg

Soms weet je nog niet goed **wat je precies moet vragen**, of wil je dat de AI eerst **nadenkt over het probleem** vóórdat het met een oplossing komt.

Dan kun je de AI vragen om eerst **zelf vragen te stellen** over de situatie of opdracht.

Deze techniek helpt om:

- een probleem beter te begrijpen
- complexe opdrachten op te breken
- geen belangrijke dingen over het hoofd te zien

Deze techniek lijkt op *chain-of-thought prompting*, maar het verschil is:

□□ **De AI stelt eerst vragen aan zichzelf of aan jou** om het probleem helder te krijgen.

# Voorbeeld

**Prompt:**
*Ik wil een inschrijfformulier bouwen voor mijn website.*
*Stel jezelf eerst 3 vragen zodat je weet wat je precies moet bouwen.*
*Beantwoord die vragen.*
*Bouw daarna pas het formulier in HTML.*

**Verwachte AI-antwoord:**

1. Welke gegevens moet de gebruiker invullen?
2. Moet de invoer gevalideerd worden?
3. Wat moet er gebeuren na het verzenden?

Daarna volgt uitleg én pas daarna de code.

# Opdracht

Je wilt dat de AI een simpele programmeeropdracht uitvoert, maar eerst **zelf nadenkt over wat er nodig is**, door zichzelf (of jou) vragen te stellen.

1. Kies een opdracht, bijvoorbeeld:

- Maak een contactformulier
- Valideer een inputveld met JavaScript
- Toon resultaten in PHP

2. **Schrijf een prompt** waarin je de AI vraagt om:
   - eerst 3 vragen te stellen over de opdracht,
   - die vragen zelf te beantwoorden,
   - daarna pas te starten met uitleg en/of code
3. Je kunt de prompt afsluiten met: *“Geef de code pas nadat je de vragen hebt beantwoord.”*

## Inleveren

1. Je volledige prompt
2. De 3 vragen + antwoorden van de AI
3. De gegenereerde uitleg en code
4. Korte toelichting: hielp dit om het probleem beter te begrijpen?

# *6. Gebruik kritiek als leermoment (prompt debugging)*

## Uitleg

Niet elke prompt die je aan de AI geeft, levert direct het gewenste resultaat op. Soms krijg je:

- een onvolledig antwoord,
- verkeerde code,
- te algemene uitleg,
- of net niet wat je bedoelde.

In plaats van gefrustreerd te raken, kun je deze momenten gebruiken om **van je fout te leren**:

**□□ Laat de AI zelf uitleggen waarom de prompt niet goed werkte, en hoe je die beter kunt formuleren.**

Zo leer je hoe kleine verschillen in formulering grote impact kunnen hebben.

# Voorbeeld

**Slechte prompt:**
*Maak een mooie website*

**Verbeterde prompt:**
*Maak een eenvoudige, moderne HTML-pagina met een grote titel, een navigatiebalk bovenaan, en drie contentblokken onder elkaar. Gebruik alleen HTML en CSS, zonder externe libraries.*

**Prompt debugging:**
*Waarom gaf de AI een vage reactie op "maak een mooie website"?*
*Wat zou ik anders moeten vragen?*

# Opdracht

Je oefent met het herkennen en verbeteren van een slechte prompt.

1. **Schrijf eerst bewust een vage of slechte prompt**, zoals:
   - "Maak een website"
   - "Schrijf een formulier"
   - "Geef uitleg over PHP"
2. **Vraag vervolgens aan de AI:**
   - Waarom deze prompt niet optimaal is
   - Wat er beter zou kunnen aan deze prompt
   - Hoe je de prompt kunt herschrijven
3. **Schrijf daarna een verbeterde versie** van je prompt, en test het resultaat.
4. **Reflecteer:**
   - Wat is het verschil in output?
   - Wat leer je hiervan over duidelijkheid en specificiteit in prompts?

# Inleveren

1. De oorspronkelijke (slechte) prompt
2. De analyse van de AI
3. De verbeterde prompt
4. De nieuwe output
5. Korte reflectie: wat is het belangrijkste inzicht?

---

Revision #10
Created 13 May 2025 18:32:43 by Max
Updated 4 June 2025 18:15:05 by Max