

Les 3 — Lesopdracht

Cursor in de praktijk

Vak: AI Fundamentals
Cursus: AI Developer
Volgende les: Les 4 — TypeScript

In deze les leer je hoe je **Cursor** effectief gebruikt om snel professionele componenten te bouwen. Je maakt een **NIEUW** Next.js project met Cursor Composer, genereert AI-gegenereerde cursorrules, en oefent met Tab Completion, Chat, Inline Edit en Composer.

Eindresultaat: Een werkende Next.js applicatie met Hero component en Feature Cards, volledig gebouwd met Cursor.

Timing

Totaal: 85 minuten

- Setup & Git: 10 min
- Skills & cursorrules: 10 min
- Hero Component (Composer): 20 min
- Feature Cards (Composer): 20 min
- Inline Edit styling: 15 min
- Git commit: 5 min
- Buffer: 5 min

Vorbereiding

Benodigde software

- **Cursor** (gedownload en geïnstalleerd van cursor.com)
- **Node.js** 18+ en npm
- **Git** (voor versiecontrole)
- **Terminal/Command Prompt**

Stap 1: Project aanmaken met Git (10 minuten)

Stap 1.1: Nieuwe folder en Git init

Open Terminal/Command Prompt op je computer (niet in Cursor nog):

```
mkdir les3-cursor-project
cd les3-cursor-project
git init
```

Stap 1.2: Open in Cursor

```
cursor .
```

Dit opent je lege project in Cursor.

Checkpoint 1: Project is open in Cursor met lege folder.

Stap 1.3: Create Next.js app in Cursor Terminal

Open de Terminal in Cursor met **Ctrl+`** (backtick):

```
npx create-next-app@latest . --typescript --tailwind --app
```

Beantwoord alle vragen met **Yes (y)**:

- TypeScript? → **Yes**
- ESLint? → **Yes**
- Tailwind CSS? → **Yes**
- src/ directory? → **No**
- App Router? → **Yes**
- Default import alias? → **Yes**

Dit duurt ~2-3 minuten. Wacht tot het klaar is.

Checkpoint 2: Next.js project is aangemaakt. Je ziet `app/`, `components/`, `package.json`, etc. in de file tree.

Stap 2: Cursor Skills & Docs (10 minuten)

Stap 2.1: Open Settings

Klik op het  **Settings** icoon linksonder in Cursor.

Stap 2.2: Activeer Docs

Ga naar **Features** → **Docs**

Voeg deze drie documenten toe (zoek en click Add):

1. **Next.js** (officiële docs)
2. **React** (officiële docs)
3. **Tailwind CSS** (officiële docs)

Dit helpt Cursor veel beter code te genereren omdat het toegang heeft tot de juiste documentatie.

Stap 2.3: Controleer Tab Completion

Ga naar **Features** → **Completion**

- Zorg dat **Tab Completion** is ingeschakeld (dit is gratis!)

Checkpoint 3: Alle drie docs zijn zichtbaar in je Docs sidebar rechts. Tab Completion is enabled.

Stap 3: Generate cursorrules met Chat (1 request)

Stap 3.1: Open Cursor Chat

Druk **Ctrl+L** (Windows/Linux) of **Cmd+L** (Mac).

Een chat panel opent aan de zijkant.

Stap 3.2: Prompt voor .cursorrules

Typ deze prompt en druk Enter:

```
Je bent een expert Next.js developer. Genereer een .cursorrules bestand voor een Next.js project met TypeScript, Tailwind CSS, en App Router.
```

```
Het bestand moet guidelines bevatten voor:
```

- Functional components en React hooks
- TypeScript best practices
- Tailwind CSS utilities (nooit inline styles)
- Bestandsnaamgeving (PascalCase voor components, camelCase voor utils)
- Folder structuur (/app, /components, /lib)
- Comments in het Nederlands
- Keine `any` types

```
Output het volledige .cursorrules bestand klaar om te copy-pasten.
```

Wacht op Cursor's antwoord.

Stap 3.3: Copy de output

Cursor geeft je het complete .cursorrules bestand.

Selecteer alle code (Ctrl+A in de chat output) en kopieer het.

Stap 3.4: Maak .cursorrules bestand

1. Rechtsklik op de root folder in de left sidebar

2. Klik "New File"
3. Noem het `.cursorrules` (met punt aan het begin!)
4. Plak de content

Save het bestand (Ctrl+S).

Checkpoint 4: `.cursorrules` bestand bestaat in je project root met goede guidelines.

Request Budget: 1 van ~7 used

Stap 4: Hero Component met Composer (20 minuten)

Stap 4.1: Maak component file

1. Rechtsklik op de app folder
2. "New File"
3. Noem het `app/components/Hero.tsx`

(Dit maakt automatisch de `components` subfolder)

Stap 4.2: Open Composer

Met het `Hero.tsx` file open, druk **Ctrl+Shift+I** (Windows/Linux) of **Cmd+Shift+I** (Mac).

De Composer panel opent.

Stap 4.3: Geef instructie aan Composer

In Composer, typ:

Maak een Hero component (TypeScript, React functional component) met:

- Grote h1 heading: "Welkom bij Cursor"
- Subheading met beschrijvende tekst: "Bouw websites snel met AI"
- Een CTA button: "Aan de slag" (link naar `#features`)
- Responsive design: Tailwind CSS (`md: breakpoint`)
- Flexbox voor verticale centrering
- Dark mode compatible (`dark: classes`)
- Gradient background (blauw naar paars)
- Geen external dependencies

Geef alleen de component code, geen imports of exports statements voor buiten.

Cursor genereert het component.

Stap 4.4: Accept of refine

Review de getoonde code. Klik **Accept All** (groene knop) om het in je bestand in te voegen.

Checkpoint 5: `Hero.tsx` file bevat een werkend Hero component met juiste structuur.

Request Budget: 2 van ~7 used

Stap 5: Feature Cards Component met Composer (20 minuten)

Stap 5.1: Maak component file

1. Rechtsklik op app folder
2. "New File"
3. Noem het app/components/FeatureCards.tsx

Stap 5.2: Open Composer

Druk **Ctrl+Shift+I** in dit nieuwe bestand.

Stap 5.3: Composer prompt

```
Maak een FeatureCards component (TypeScript, React) dat:  
- Een grid van 3 feature cards toont  
- Responsive: 1 kolom op mobile, 3 kolommen op desktop (Tailwind grid)  
- Elke card heeft:  
  * Een icon uit lucide-react (Package, Zap, Layers)  
  * Een h3 title  
  * Een description text  
- Hover effect: transform scale-105 en shadow toename  
- Cards hebben border en padding (Tailwind)  
- Dark mode ready  
- lucide-react is geïmporteerd bovenaan  
  
Geef alleen de component code.
```

Composer genereert het component.

Stap 5.4: Accept

Klik **Accept All**.

Stap 5.5: Install lucide-react

In je Cursor Terminal (Ctrl+`):

```
npm install lucide-react
```

Wacht tot npm klaar is.

Checkpoint 6: FeatureCards.tsx werkt, lucide-react is geïnstalleerd.

Request Budget: 3 van ~7 used

Stap 6: Integreer in Homepage (5 minuten)

Stap 6.1: Open `app/page.tsx`

Dit is je homepage.

Stap 6.2: Update de content

Vervang de bestaande content (of voeg toe aan het bestand) met:

```
import Hero from '../components/Hero';
import FeatureCards from '../components/FeatureCards';

export default function Home() {
  return (
    <main className="min-h-screen">
      <Hero />
      <section className="py-20 bg-gray-50 dark:bg-gray-900">
        <div className="container mx-auto px-4">
          <h2 className="text-4xl font-bold mb-12 text-center">Features</h2>
          <FeatureCards />
        </div>
      </section>
    </main>
  );
}
```

Save (Ctrl+S).

Stap 6.3: Test in browser

In Cursor Terminal:

```
npm run dev
```

Open <http://localhost:3000> in je browser.

Je ziet nu: Hero → Features section met 3 cards.

Checkpoint 7: Website draait lokaal en ziet er netjes uit.

Stap 7: Styling Refinement met Inline Edit (15 minuten)

Nu gebruiken we **Inline Edit** (Ctrl+K) voor kleine styling tweaks.

Stap 7.1: Hero button styling

1. Open `app/components/Hero.tsx`
2. Klik ergens in de button JSX
3. Druk **Ctrl+K**
4. Type:

Voeg een hover effect toe aan de button:

- Scale slightly bigger on hover
- Change background color smoothly
- Add smooth transition

1. Review de suggestie, klik Accept

Stap 7.2: Card shadows verbeteren

1. Open `app/components/FeatureCards.tsx`
2. Selecteer een card div
3. Druk **Ctrl+K**
4. Type:

Voeg een mooier shadow effect toe aan de cards:

- Larger shadow on hover
- Smooth transition
- Better depth perception

1. Accept

Stap 7.3: Typography verbeteren

1. Open `app/page.tsx`
2. Druk **Ctrl+K**
3. Type:

Maak de "Features" heading groter en mooier:

- Groter font size
- Betere kleur gradient

1. Accept

Refresh je browser (F5) om de changes te zien.

Checkpoint 8: Styling ziet er gepolijst uit, hover effects werken.

Request Budget: 4-5 van ~7 used

Stap 8: Git Commit (5 minuten)

Stap 8.1: Open Source Control

Druk **Ctrl+Shift+G** in Cursor.

Source Control panel opent links.

Stap 8.2: Stage alle bestanden

Klik op het + icoon bij "Changes" of druk **Ctrl+Shift+A**.
Alle bestanden worden staged (groen).

Stap 8.3: Write commit message

In het message veld, typ:

```
feat: Build Hero and FeatureCards components with Cursor Composer
```

Stap 8.4: Commit

Druk **Ctrl+Enter** of klik de ✓ button.
Je commit is gemaakt!
Checkpoint 9: Git commit is succesvol (geen errors).

Keyboard Shortcuts Reference

Actie	Windows	Mac
Cursor Chat	Ctrl+L	Cmd+L
Composer	Ctrl+Shift+I	Cmd+Shift+I
Inline Edit (quick fix)	Ctrl+K	Cmd+K
Terminal	Ctrl+`	Cmd+`
Command Palette	Ctrl+Shift+P	Cmd+Shift+P
Source Control	Ctrl+Shift+G	Cmd+Shift+G
Save	Ctrl+S	Cmd+S
Tab Completion	Tab	Tab

Request Budget Tracking

Je hebt ~7 requests beschikbaar in deze les:

Fase	Type	Requests
cursorrules generatie	Chat	1
Hero Component	Composer	1

Fase	Type	Requests
FeatureCards Component	Composer	1
Button styling tweak	Inline Edit	0.5
Card styling tweak	Inline Edit	0.5
Typography tweak	Inline Edit	0.5
Buffer/extra	-	2.5

■ **Pro tip:** Gebruik **Tab Completion** (gratis!) voor:

- Imports toevoegen
- Props schrijven
- Repetitieve code (zelfde classNames herhalen)

Troubleshooting

Probleem: "Module not found: lucide-react"

Oplossing:

```
npm install lucide-react
npm run dev
```

(Herstart je dev server na npm install)

Probleem: Composer stelt rare dingen voor

Oplossing: Wees specifiek in je prompt. Verwijs naar bestaande code:

- "Voeg lucide-react icons toe" (voldoende duidelijk)
- "Zorg dat het met TypeScript werkt" (specificeer)

Probleem: Inline Edit accepteert niet

Oplossing: Klik de groene ✓ knop of druk Ctrl+Enter

Probleem: `npm run dev` geeft error

Stappen:

1. Controleer dat je in de juiste folder bent: `pwd`
2. Controleer node versie: `node --version` (moet 18+ zijn)
3. Verwijder en herinstalleer dependencies:

```
rm -rf node_modules package-lock.json
npm install
npm run dev
```

Probleem: Port 3000 al in gebruik

```
npm run dev -- -p 3001
# Openen: http://localhost:3001
```

Probleem: Git werkt niet

Controleer:

- Ben je in de juiste folder? `pwd` moet je project folder zijn
- Bestaat `.git` folder? `ls -la .git`
- Zijn je bestanden getracked? `git status`

Checklist - Vinkje alles af voor je klaar bent

- ☐ Git init en create-next-app uitgevoerd
- ☐ Cursor Skills (Next.js, React, Tailwind) geactiveerd
- ☐ `.cursorrules` bestand gegenereerd en in root geplaatst
- ☐ `app/components/Hero.tsx` aangemaakt met Composer
- ☐ `app/components/FeatureCards.tsx` aangemaakt met Composer
- ☐ `lucide-react` geïnstalleerd via npm
- ☐ Componenten geïmporteerd in `app/page.tsx`
- ☐ `npm run dev` draait zonder errors
- ☐ Website ziet er goed uit op `http://localhost:3000`
- ☐ Styling verfijnd met Inline Edit (minimaal 2 aanpassingen)
- ☐ Git commit gemaakt met duidelijke message
- ☐ Alle bestanden zijn saved (geen rode stippen in sidebar)

Volgende stappen

Voor volgende les:

- Zorg dat `npm run dev` nog werkt
- Download het huiswerk zip bestand van Teams (les3-debug-challenge.zip)
- Voorkom dat je project files verwijdert/opslaanslecht

Veel succes! ■