TWEE STAPPENPLANNEN

Vijf stappen voor Claude. *Drie* voor je aanmeldingen.

Deze gids hoort bij twee reels met het trefwoord STAPPEN. Deel 1 zet Claude Code goed neer in vijf commando's, met een skill om te kopiëren. Deel 2 is voor wie software verkoopt en veel aanmeldingen ziet, maar weinig betalende klanten: drie stappen, een meting en een werkblad.

Deel 1 · Claude Code instellen

`/init`, `/memory`, `/fewer-permission-prompts`, een eigen skill en `find-skills`.

Deel 2 · Van aanmelding naar klant

De ene actie vinden, alles ervoor weghalen en meten hoeveel nieuwe gebruikers hem halen.

DEEL 1 · JIJ HEBT CLAUDE NOG NIET INGESTELD? VIJF STAPPEN. KLAAR.

Deze stappen zijn voor **Claude Code**, de versie van Claude die in je projectmap werkt. Je typt de commando's in een sessie: ga in de terminal naar je projectmap en start `claude`.

```
cd ~/pad/naar/je-project
claude
```

01 /init: Claude schrijft zijn eigen handleiding

Wat je typt:

```
/init
```

Wat er gebeurt: Claude bekijkt je project en maakt een bestand `CLAUDE.md` met wat hij vindt: de commando's om te bouwen en te testen, en de afspraken in je code. Staat er al een CLAUDE.md, dan overschrijft `/init` hem niet maar stelt hij verbeteringen voor.

Wil je meer begeleiding? Zet dan eerst `CLAUDE_CODE_NEW_INIT=1` in je terminal en start Claude opnieuw. `/init` vraagt dan wat je wilt opzetten (CLAUDE.md, skills, hooks), stelt vragen over wat hij niet zelf kan vinden en laat een voorstel zien voordat hij iets wegschrijft.

02 /memory: jouw regels erin

Wat je typt:

```
/memory
```

Wat er gebeurt: je ziet een lijst van je CLAUDE.md-bestanden: die van dit project (`./CLAUDE.md`) en je persoonlijke voor al je projecten (`~/.claude/CLAUDE.md`). Kies er een en hij opent in je editor. Bestaat hij nog niet, dan maakt Claude hem eerst aan. In hetzelfde menu zet je het automatische geheugen aan of uit, waarin Claude zelf aantekeningen bijhoudt.

Zet in je CLAUDE.md wat je anders elke sessie opnieuw uitlegt: je toon, wat nooit mag, en hoe Claude test. Claude leest dit bestand aan het begin van elke sessie. Een voorbeeld om aan te passen:

KOPIEER DIT · VOORBEELD VOOR CLAUDE.MD

```
# Regels voor dit project

## Toon
- Teksten voor klanten: Nederlands, "je", korte zinnen.
- Geen vakjargon zonder uitleg.

## Wat nooit mag
- Nooit wachtwoorden of API-sleutels in een bestand zetten.
- Nooit pushen naar main als ik er niet om vraag.
- Niets verzinnen: weet je het niet, vraag het.

## Testen
- Draai `npm test` voordat je zegt dat iets klaar is.
- `npm run build` moet zonder fouten blijven.
```

Schrijf regels die je kunt controleren. "Draai `npm test` voordat je commit" werkt beter dan "test je werk". Anthropic raadt aan om een CLAUDE.md onder de 200 regels te houden: hoe langer het bestand, hoe slechter Claude het volgt. Je kunt ook gewoon in de chat zeggen: "zet dit in CLAUDE.md".

03 /fewer-permission-prompts: minder "mag ik?"

Wat je typt:

```
/fewer-permission-prompts
```

Wat er gebeurt: Claude zoekt in je eerdere sessies naar commando's die veel voorkomen en alleen iets lezen, zoals bestanden bekijken of zoeken. Die zet hij op een toestemmingslijst in `.claude/settings.json` van je project. Daarna vraagt hij minder vaak of hij iets mag doen.

Het gaat alleen om commando's die lezen, niet om commando's die iets veranderen. Open `.claude/settings.json` daarna even en kijk wat erop staat. Dat bestand hoort bij je project: zet je het in git, dan krijgt je team dezelfde lijst.

04 Je eigen skill: /offerte

Een skill is een vaste werkwijze die Claude pakt als hij nodig is. Je maakt een map met daarin één bestand, `SKILL.md`: bovenaan een `description` die zegt wanneer de skill van toepassing is, daaronder je instructies.

Wat je typt, in de terminal in je projectmap:

```
mkdir -p .claude/skills/offerte
```

Zet daarin een bestand `SKILL.md`. Een voorbeeld om te kopiëren en aan te passen:

KOPIEER DIT · .CLAUDE/SKILLS/OFFERTE/SKILL.MD

```
---
description: Stelt een offerte op uit onze vaste prijzen. Gebruik als ik vraag om een
offerte, prijsopgave of begroting voor een klant.
---

# Offerte opstellen

Maak een offerte voor: $ARGUMENTS

1. Gebruik alleen prijzen uit ${CLAUDE_SKILL_DIR}/prijzen.md.
   Staat een prijs er niet in, noem hem dan niet en vraag het mij.
2. Mist er iets, zoals aantallen, het adres of de gewenste datum?
   Vraag het eerst, voordat je begint.
3. Schrijf de offerte in deze volgorde:
   - aanhef met de naam van de klant
   - wat we gaan doen, in gewone taal
   - tabel: omschrijving, aantal, prijs per stuk, totaal
   - totaal zonder en met btw (het tarief staat in prijzen.md)
   - hoe lang de offerte geldig is (staat in prijzen.md)
4. Toon: Nederlands, "je", geen vakjargon.
5. Sla de offerte op als offertes/[klantnaam]-[datum].md
   en laat mij hem lezen voordat hij ergens heen gaat.
```

Zet je prijzen in een tweede bestand in dezelfde map: `.claude/skills/offerte/prijzen.md`. `${CLAUDE_SKILL_DIR}` verwijst naar de map van de skill, en `$ARGUMENTS` wordt vervangen door wat je achter het commando typt.

Wat er gebeurt: de skill heet naar zijn map, dus je start hem met:

```
/offerte badkamer 6 m², familie De Vries
```

Of je vraagt gewoon "maak een offerte voor de familie De Vries": past je vraag bij de description, dan pakt Claude de skill zelf. Claude Code ziet een nieuwe of gewijzigde skill meteen, zonder herstart. Bestond de map `.claude/skills` nog niet toen je de sessie startte, typ dan eerst `/reload-skills`. Wil je de skill in al je projecten, zet hem dan in `~/.claude/skills/offerte/`.

05 find-skills: laat Claude zoeken

Anderen hebben al veel skills gemaakt. `find-skills` is een skill die Claude ze laat vinden in de open bibliotheek van `skills.sh`. Je installeert hem met de skills-tool van Vercel. Daarvoor heb je Node.js nodig, voor `npx`.

KOPIEER DIT · IN DE TERMINAL

```
npx skills add vercel-labs/skills --skill find-skills -a claude-code
```

Wat er gebeurt: `--skill find-skills` kiest die ene skill uit de verzameling, `-a claude-code` zegt dat hij voor Claude Code is. Hij komt in `.claude/skills/` van je project; met `-g` erachter staat hij in al je projecten.

Daarna vraag je in Claude Code bijvoorbeeld:

```
Is er een skill voor het nakijken van mijn teksten op taalfouten?
```

Claude zoekt met `npx skills find` en in de ranglijst op skills.sh, en laat opties zien met de maker, het aantal installaties en het commando. Hij installeert pas als jij ja zegt.

> **Kijk wie een skill gemaakt heeft.** Een skill bevat instructies, en soms scripts, die Claude uitvoert. Open de `SKILL.md` voordat je hem gebruikt. `find-skills` zelf raadt aan om bekende makers te kiezen, voorzichtig te zijn onder de 100 installaties en een repository met minder dan 100 sterren met argwaan te bekijken.

06 Goed om te weten

> - CLAUDE.md is context, geen slot. Claude leest het en volgt het meestal, maar het is geen garantie. Moet iets altijd of nooit gebeuren, dan is daar in Claude Code een hook voor.
> - Een CLAUDE.md en de map `.claude/` in je project kun je met je team delen via git. Persoonlijke voorkeuren horen in `~/.claude/CLAUDE.md` of in `CLAUDE.local.md`, die je niet commit.
> - Typ `/context` om te zien welke CLAUDE.md-bestanden Claude aan het begin heeft geladen, en `/skills` voor de lijst met skills.

DEEL 2 · 500 AANMELDINGEN. 3 BETALENDE KLANTEN.

Het probleem zit meestal niet bij je betaalmuur. De meeste aanmeldingen hebben je product nooit echt gebruikt: account gemaakt, leeg dashboard, en weg. Wie nooit iets aan je product heeft gehad, gaat er ook niet voor betalen.

Grote producten zochten daarom het moment waarop een nieuwe gebruiker het snapt. Chamath Palihapitiya, die bij Facebook het groeiteam leidde, zei in een interview uit 2012 dat het grootste inzicht was om ieder nieuw lid binnen 10 dagen aan 7 vrienden te helpen. Stewart Butterfield, medeoprichter van Slack, vertelde in 2015 dat ze een team dat 2.000 berichten had uitgewisseld zagen als een team dat Slack echt had geprobeerd. Van die teams gebruikte 93 procent Slack nog.

Die getallen horen bij hun product. Het jouwe heeft een eigen moment, en dat zoek je zo.

01 Zoek de ene actie die je betalende klanten vroeg deden

Je zoekt een actie die bijna al je betalende klanten in hun eerste dagen deden, en de meeste afhakers niet. Het moet iets zijn waarbij de gebruiker waarde krijgt, niet een klik die toevallig iedereen doet. Denk aan: de eerste factuur verstuurd, eigen gegevens geïmporteerd, een collega uitgenodigd, een koppeling gemaakt, het eerste rapport gemaakt. "Profiel ingevuld" is meestal geen goede kandidaat.

1. **Maak twee groepen.** Je betalende klanten, en de mensen die zich aanmeldden maar nooit betaalden. De betalende haal je uit je betaalprovider of je database.
2. **Schrijf vijf tot tien kandidaat-acties op.** Alles wat een gebruiker in je product kan doen dat iets oplevert.
3. **Kies een periode.** Bijvoorbeeld de eerste 7 dagen na aanmelding.
4. **Tel per actie.** Welk deel van de betalende klanten deed hem binnen die periode? En welk deel van de afhakers? Je zoekt de actie met het grootste verschil.

Je hoeft hier geen analysepakket voor te hebben. Meestal staat het al in je database: bijna elke tabel heeft een tijdstip waarop iets is aangemaakt. Een voorbeeld in SQL (PostgreSQL). De tabel- en kolomnamen zijn verzonnen; vervang ze door die van jou.

KOPIEER DIT · WELK DEEL DEED DE ACTIE IN WEEK ÉÉN

```
-- per groep: hoeveel gebruikers verstuurden binnen 7 dagen hun eerste factuur
select
  case when u.betaalt then 'betalend' else 'niet betalend' end as groep,
  count(*) as gebruikers,
  count(*) filter (where exists (
    select 1 from facturen f
    where f.gebruiker_id = u.id
      and f.aangemaakt_op <= u.aangemeld_op + interval '7 days'
  )) as deden_het
from gebruikers u
group by 1;
```

Draai dezelfde vraag voor elke kandidaat-actie en zet de uitkomsten in het werkblad hieronder.

Heb je maar een paar betalende klanten? Dan zegt een percentage weinig. Open dan per klant wat hij in zijn eerste dagen deed, op volgorde, en bel of mail ze met één vraag: wanneer wist je dat je zou blijven? Leg dat naast wat je actiefste gratis gebruikers doen.

02 Schrap alles tussen aanmelding en die actie

Maak zelf een nieuw account aan, met een nieuw e-mailadres, en loop naar de actie uit stap 1. Schrijf elk scherm, elk veld en elke klik op. Vraag bij elke stap: is dit nodig om die actie te doen?

WAT JE TEGENKOMT	WAT JE ERMEE KUNT
Velden die je nu nog niet nodig hebt	*Later vragen, op het moment dat ze ertoe doen.*
Een rondleiding langs alle knoppen	*Vervangen door één knop die naar de actie leidt.*
Een leeg dashboard	*Voorbeeldgegevens of een sjabloon, zodat er meteen iets te doen is.*
E-mail bevestigen vóór je verder kunt	*Kijken of het later kan, zonder dat je misbruik toelaat.*
Instellingen kiezen	*Een goede standaard, die je later kunt aanpassen.*

Na de aanmelding hoort de gebruiker op de plek te landen waar hij de actie doet, niet op een overzicht. Verander één ding per keer, zodat je in stap 3 ziet wat het deed.

03 Meet hoeveel nieuwe gebruikers dat moment halen

Eén getal per week: **van alle nieuwe aanmeldingen, welk deel deed de actie in de eerste sessie?** Meet je geen sessies, neem dan een vaste tijd na aanmelding, bijvoorbeeld 30 minuten of dezelfde dag. Houd die keuze daarna vast, anders vergelijk je appels met peren.

KOPIEER DIT · ACTIVATIE PER AANMELDWEEK

```
-- per aanmeldweek: welk deel verstuurde binnen 30 minuten een factuur
select
  date_trunc('week', u.aangemeld_op) as aanmeldweek,
  count(*) as aanmeldingen,
  count(*) filter (where exists (
    select 1 from facturen f
    where f.gebruiker_id = u.id
      and f.aangemaakt_op <= u.aangemeld_op + interval '30 minutes'
  )) as haalden_het_moment
from gebruikers u
```

```
group by 1
order by 1;
```

Deel `haalden_het_moment` door `aanmeldingen` en je hebt je percentage. Kijk na een paar weken ook of de mensen die het moment haalden vaker gingen betalen dan de rest. Is dat niet zo, dan heb je in stap 1 de verkeerde actie gekozen.

04 Het werkblad

Kopieer het, vul het in en houd het elke week bij.

KOPIEER DIT · WERKBLAD

```
STAP 1 · DE ACTIE
Periode na aanmelding: [bijvoorbeeld 7 dagen]
Betalende klanten: [aantal]   Afhakers: [aantal]

Kandidaat-actie        Betalend deed het   Afhakers deden het
[actie 1]              [..] van [..]       [..] van [..]
[actie 2]              [..] van [..]       [..] van [..]
[actie 3]              [..] van [..]       [..] van [..]

Gekozen actie: [...]
Waarom deze: [grootste verschil, en de gebruiker heeft er iets aan]

STAP 2 · DE WEG ERNAARTOE
Scherm/stap na aanmelding     Nodig voor de actie?   Wat ik doe
[1. ...]                      [ja/nee]               [houden/later/weg]
[2. ...]                      [ja/nee]               [houden/later/weg]
[3. ...]                      [ja/nee]               [houden/later/weg]

STAP 3 · DE METING
"Eerste sessie" betekent bij mij: [bijvoorbeeld binnen 30 minuten]

Week      Aanmeldingen   Haalden het moment   Procent   Wat ik veranderde
[week]    [..]           [..]                 [..]%     [één ding]
[week]    [..]           [..]                 [..]%     [één ding]

Controle na [aantal] weken:
Wie het moment haalde, betaalde later: [..]%
Wie het niet haalde, betaalde later:   [..]%
```

05 Eerlijk is eerlijk

- Dat betalende klanten een actie vaker deden, bewijst niet dat die actie ze liet betalen. Misschien waren het gewoon de mensen die al wilden. Daarom meet je in stap 3 of de wijzigingen iets doen.
- Met weinig aanmeldingen per week schommelt het percentage flink. Kijk naar de lijn over meerdere weken, niet naar één week.
- De getallen van Facebook en Slack zijn vuistregels die hun eigen mensen noemden, geen gepubliceerd onderzoek. Neem ze niet over: zoek je eigen moment.
- Een kortere weg naar de actie helpt alleen als de actie zelf iets oplevert. Een product dat het probleem niet oplost, wordt daar niet beter van.

Ondernemer? Laat het voor je bouwen.

Webry bouwt systemen en AI-agents die werk uit je bedrijf overnemen, van leads opvolgen tot offertes klaarzetten. In een half uur aan de telefoon kijken we waar je bedrijf vastloopt en wat een werkbare volgende stap is.

Plan een gesprek

Meer van dit?

Op Instagram laten we zien wat AI nu al kan in het echte werk: tools, workflows en de links erbij.

Volg @webry.nl

Een gids van Webry. Kom je er niet uit met een van de stappen? Stuur ons een bericht op Instagram (@webry.nl) of kijk op webry.nl.

Webry.

Bronnen deel 1: code.claude.com/docs/en/commands, code.claude.com/docs/en/skills en code.claude.com/docs/en/memory (Anthropic), github.com/vercel-labs/skills met de find-skills-skill en docs/agents.md, en skills.sh. Bronnen deel 2: Chamath Palihapitiya in "How we put Facebook on the path to 1 billion users", interview door Gagan Biyani, 2012 (youtube.com/watch?v=raIUQP71SBU); Stewart Butterfield in "From 0 to $1B - Slack's Founder Shares Their Epic Launch Strategy", First Round Review, 27 februari 2015. Gecontroleerd op 2 november 2026. Commando's en opties kunnen veranderen. Webry is niet verbonden aan Anthropic, Vercel, Facebook of Slack.