

Les 13 — Huiswerk

Agent uitbreiden + prepareStep + custom stop + AGENT.md

Vak:	AI-Assisted Development
Deadline:	Voor Les 14 — RAG + embeddings
Inleveren:	GitHub repo + AGENT.md in root
Tijd:	~2 uur

Doel

Bouwt voort op de lesopdracht (research-agent met 3 tools + stepCountIs(10)). In dit huiswerk maak je de agent slimmer: extra tools, dynamic step-control, een eigen stop-conditie, en documenteer je het geheel.

A

Extra tool — listReports (verplicht)

Voeg een 4e tool toe om opgeslagen rapporten op te halen.

```
export const listReports = tool({
  description:
    "Toon eerder opgeslagen onderzoeksrapporten. " +
    "Gebruik dit als de gebruiker vraagt naar bestaand onderzoek " +
    "voordat je nieuw onderzoek doet.",
  inputSchema: z.object({
    limit: z.number().int().min(1).max(20).default(5),
  }),
  execute: async ({ limit }) => {
    const { data, error } = await supabase
      .from("research_reports")
      .select("id, query, created_at")
      .order("created_at", { ascending: false })
      .limit(limit);
    if (error) return { error: error.message };
    return data;
  },
});
```

Eisen:

- listReports werkt

-
- System prompt aangepast — agent weet wanneer te checken
 - Minstens 1 test-query gedraaid die listReports triggert

B

prepareStep toevoegen (verplicht)

Kies minimaal één use case:

Optie B1 — Dynamic model

```
prepareStep: async ({ stepNumber, messages }) => {
  if (stepNumber > 4 || messages.length > 12) {
    return { model: openai("gpt-4o") };
  }
  return {};
}
```

Optie B2 — Fase-gebaseerde tools

```
prepareStep: async ({ stepNumber }) => {
  if (stepNumber <= 2) {
    return { activeTools: ["listReports", "webSearch"] };
  }
  if (stepNumber <= 5) {
    return { activeTools: ["readPage"] };
  }
  return { activeTools: ["saveReport"], toolChoice: "required" };
}
```

Optie B3 — Context-trimming

```
prepareStep: async ({ messages }) => {
  if (messages.length > 15) {
    return {
      messages: [messages[0], ...messages.slice(-10)],
    };
  }
  return {};
}
```

Eisen:

- Minimaal 1 prepareStep use-case geïmplementeerd
- Effect zichtbaar in console-logs
- In AGENT.md uitleggen welke optie + waarom

C

Custom stop-conditie (verplicht)

Schrijf een eigen StopCondition. Twee voorbeelden:

Optie C1 — Token-budget

```
import type { StopCondition } from "ai";

const tokenBudget: StopCondition<typeof tools> = ({ steps }) => {
  const total = steps.reduce(
    (sum, s) => sum + (s.usage?.totalTokens ?? 0),
    0
  );
  return total > 30_000;
};
```

Optie C2 — Genoeg sources verzameld

```
const enoughSources: StopCondition<typeof tools> = ({ steps }) => {
  const readPages = steps.flatMap((s) =>
    s.toolCalls?.filter((tc) => tc.toolName === "readPage") ?? []
  );
  return readPages.length >= 5;
};
```

Combineren met andere condities:

```
stopWhen: [
  stepCountIs(30),
  tokenBudget,
  hasToolCall("saveReport"),
]
```

Eisen:

- Custom StopCondition werkt
- Gecombineerd met stepCountIs(N) voor veiligheid
- In AGENT.md uitleggen wat de conditie doet

D

AGENT.md documentatie (verplicht)

In repo-root, vijf secties:

Sectie 1 — Tools

Tabel met al je tools (Read/Write).

Sectie 2 — Stop-strategie

- `stepCountIs(N)` — safety cap, $N = \dots$
- Custom — uitleg
- Combinatie waarom

Sectie 3 — `prepareStep` keuze

- Welke optie (B1/B2/B3)
- Waarom
- Verschil console-logs met/zonder

Sectie 4 — Voorbeeld-run

Pak een echte query, draai 'm, log alle steps:

```
Query: "Wat zijn de belangrijkste AI-trends in 2026?"
```

```
Step 0: webSearch({ query: "AI trends 2026" })
```

```
-> 5 resultaten
```

```
Step 1: readPage({ url: "..." })
```

```
-> 4500 char tekst
```

```
Step 2: readPage({ url: "..." })
```

```
Step 3: webSearch(...)
```

```
Step 4: readPage(...)
```

```
Step 5: saveReport({ ... })
```

```
-> reportId: 42
```

```
Finale rapport: (200-300 woorden)
```

Sectie 5 — Eén observatie

- Onverwacht goed gedrag?
- Loop die te lang duurde?
- Tool die niet werd gekozen?
- Effect van `prepareStep` zichtbaar?

Vorm: max 700 woorden totaal. Concrete logs, mag informeel.

E

Bonus (optioneel)

Bonus 1 — UI met live steps

Gebruik `agent.stream()` in plaats van `generate()`. Client toont elke step terwijl agent draait.

Bonus 2 — Sub-agent voor samenvatten

```
const summarizer = new ToolLoopAgent({
  model: openai("gpt-4o-mini"),
  system: "Vat &eacute;&eacute;n pagina samen in 5 bullets.",
  tools: { readPage },
  stopWhen: stepCountIs(3),
});

const summarizePage = tool({
  description: "Vat een URL samen",
  inputSchema: z.object({ url: z.string().url() }),
  execute: async ({ url }) => {
    const r = await summarizer.generate({
      prompt: `Samenvat: ${url}`,
    });
    return { summary: r.text };
  },
});
```

Bonus 3 — Done-tool pattern

Vervang `hasToolCall('saveReport')` door een expliciete done-tool zonder `execute`, met `toolChoice: 'required'`. Documenteer in `AGENT.md` wat het verschil was.

F

Inleveren + Beoordeling

Inleveren:

1. GitHub repo URL in Brightspace
2. `AGENT.md` in repo-root
3. Updated `lib/tools.ts` met `listReports`
4. Updated `lib/agent.ts` met `prepareStep` + custom stop
5. Schema-update in `schema.sql` (mocht je nieuwe tabel hebben)

Beoordeling (10 pt — voldoende 6+):

Criterium	Punten
A — listReports werkt + agent gebruikt 'm correct	2
B — prepareStep + zichtbaar effect	2
C — Custom StopCondition werkt	2
D — AGENT.md compleet met 5 secties	3
Agent draait end-to-end (geen broken runs)	1
Totaal	10

Veelvoorkomende valkuilen

- prepareStep returnt niks — je moet {} returnen, niet undefined
- Custom stop returnt nooit true — log per stap je conditie-waarde
- Agent stopt te vroeg — check stopWhen array niet te restrictief
- Sub-agent recursie — limit op depth om stack-overflow te vermijden
- Token-cost out of control — log per stap usage + cost-cap

Tips

- Log per stap — console.log({ step, toolCalls, usage }) is de beste debug-tool
- Test je StopCondition met fake steps eerst (isolated)
- Houd AGENT.md bij tijdens werken — niet aan eind
- Eindrapport kwaliteit ligt 80% aan system prompt — niet aan tools
- Volgende les: RAG + embeddings. Combo: RAG-tool in een agent.