

Les 11 — Vercel AI SDK

Praat met je eigen data — Polderfest 2027 demo

Vak:	AI-Assisted Development
Opleiding:	NOVI Hogeschool Utrecht
Klas:	Klas A (3 uur fysiek, demo-driven)
Demo-app:	Polderfest 2027 — fictief festival, 500 records

Leerdoelen

- Weten wat de Vercel AI SDK is en welke 4 kern-functies hij heeft
- Modellen-landschap kunnen plaatsen + kosten inschatten
- Een nieuwe Next.js app + Supabase + AI SDK aan elkaar koppelen
- Een seed-script lezen en runnen voor 500 dummy records
- Chat-route + chat-UI met streamText + useChat implementeren
- Begrijpen waarom data + AI samen krachtiger is dan beide apart

Inhoud

1. Wat is de Vercel AI SDK?
2. Het modellen-landschap
3. De vier kern-functies
4. Project setup van A tot Z
5. Seed script: dummy data in Supabase
6. Chat-route + chat-UI implementeren
7. Waarom data + AI samen krachtig zijn
8. Best practices & valkuilen
9. Wat komt hierna? Tool Calling teaser
10. Bronnen

1

Wat is de Vercel AI SDK?

Open-source TypeScript library waarmee je AI-features in je webapplicatie bouwt. Gemaakt door Vercel — daarom perfecte integratie met Next.js, Server Components, Server Actions.

Wat krijg je?

- **Unified API** — zelfde code voor elk model (40+ providers)
- **Streaming out-of-the-box** — geen WebSocket-setup
- **React hooks** — `useChat`, `useCompletion` voor instant chat UI
- **Tool Calling** — AI roept functies aan die jij definieert (volgende les)
- **Structured output** — type-safe data via `generateObject` + `Zod`
- **Multi-step agents** — via `maxSteps`

Eerste indruk

```
import { generateText } from "ai";
import { openai } from "@ai-sdk/openai";

const { text } = await generateText({
  model: openai("gpt-4o-mini"),
  prompt: "Wat de Polderfest 2027 line-up samen",
});
```

Naar Anthropic? Eén regel veranderen — ``openai(...)`` → ``anthropic(...)``. Dat is de waarde.

2

Het modellen-landschap

Provider	Model	Sterke punten	Prijs (in/out per 1k tokens)
OpenAI	gpt-4o-mini	Snel + goedkoop — default	\$0.15 / \$0.60
OpenAI	gpt-4o	Multimodaal (vision)	\$2.50 / \$10
OpenAI	gpt-4.1	Reasoning, agents	\$2 / \$8
Anthropic	claude-sonnet-4	Coding, lange context (200k)	\$3 / \$15
Google	gemini-2.5-flash	Multimodaal, ultra goedkoop	\$0.075 / \$0.30
Groq	llama-3.3-70b	Ultra-fast inference	\$0.59 / \$0.79

Vuistregel

Start met `gpt-4o-mini`. Upgrade pas als nodig.

Onze Polderfest demo: <2 cent

500 bands \approx 30k tokens per call · gpt-4o-mini · 50 vragen \approx €0.25. Zelfde met gpt-4o = €4. Daarom: start mini.

3

De vier kern-functies

generateText — Antwoord ophalen

Wacht tot AI antwoord klaar is — returnt string.

```
const { text } = await generateText({
  model: openai("gpt-4o-mini"),
  prompt: "Geef 3 koffie-poll opties",
});
```

Wanneer: korte antwoorden, server-only, niet-interactief.

streamText — Streaming antwoord

Streamt karakter voor karakter. Vandaag gebruiken we dit.

```
const result = streamText({
  model: openai("gpt-4o-mini"),
  messages,
});
return result.toDataStreamResponse();
```

useChat — React hook

Client-kant: complete chat UI in 10 regels.

```
"use client";
import { useChat } from "ai/react";

const { messages, input, handleInputChange, handleSubmit }
  = useChat();
```

Werkt alleen met streamText API endpoint.

generateObject — Gestructureerde data

Type-safe data terug, gevalideerd met Zod.

```
const { object } = await generateObject({
  model: openai("gpt-4o-mini"),
  schema: z.object({
    question: z.string(),
    options: z.array(z.string()).length(4),
  }),
  prompt: "Maak een poll over koffie",
});
```

Vandaag niet — komt terug in latere lessen.

4 Project setup van A tot Z

Voor je eigen project (lesopdracht/huiswerk): zelfde stappen als wat Tim live deed.

Stap 1 — Next.js scaffolden

```
npx create-next-app@latest mijn-thema \
  --typescript --tailwind --app --eslint --no-src-dir --turbo
cd mijn-thema
```

Stap 2 — Nieuwe Supabase project

- supabase.com/dashboard → New Project
- Wacht ~2 min op deploy
- Settings → API → kopieer URL + anon key + service_role key

Stap 3 — Schema in SQL Editor

```
create table items (
  id          bigserial primary key,
  name        text not null,
  category    text,
  rating      int,
  description text,
  created_at  timestamp default now()
);
```

Stap 4 — .env.local

```
NEXT_PUBLIC_SUPABASE_URL=https://xxx.supabase.co
NEXT_PUBLIC_SUPABASE_ANON_KEY=...
SUPABASE_SERVICE_ROLE_KEY=...
OPENAI_API_KEY=sk-proj-...
```

Belangrijk

- **NEXT_PUBLIC_*** = client-leesbaar (alleen voor anon key)
- **SUPABASE_SERVICE_ROLE_KEY** = server-only, voor seed
- **OPENAI_API_KEY** = NOOIT als NEXT_PUBLIC_, server-only

Stap 5 — Packages

```
npm install @supabase/supabase-js ai @ai-sdk/openai zod
npm install --save-dev tsx dotenv
```

Stap 6 — Supabase client

```
// lib/supabase.ts
import { createClient } from "@supabase/supabase-js";

export const supabase = createClient(
  process.env.NEXT_PUBLIC_SUPABASE_URL!,
  process.env.NEXT_PUBLIC_SUPABASE_ANON_KEY!,
);
```

5

Seed script: dummy data in Supabase

Een seed-script vult je tabel eenmalig met dummy data. Geen handmatige inserts — procedureel gegenereerd.

Het Polderfest-script

Het volledige `seed-polderfest.ts` zit als bijlage bij deze les. Kernidee:

```
import { createClient } from "@supabase/supabase-js";
import "dotenv/config";

const supabase = createClient(
  process.env.SUPABASE_URL!,
  process.env.SUPABASE_SERVICE_ROLE_KEY!,
);

const adjectives = ["Lost", "Velvet", "Iron", "Neon", ...];
const nouns = ["Tigers", "Wolves", "Mirrors", ...];

function generateBand(i) {
  return {
    name: `${pick(adjectives)} ${pick(nouns)}`,
    genre: pick(genres), /* ... */
  };
}

// Insert in batches van 100
for (let i = 0; i < bands.length; i += 100) {
  await supabase.from("bands").insert(bands.slice(i, i+100));
}
```

Runnen

```
npx tsx scripts/seed-polderfest.ts
```

Waarom procedureel?

- 500 hard-coded records = 500 regels handmatig — mind-numbing
- Combinaties van 30 adjectives × 30 nouns = 900 namen mogelijk
- Seed-random = reproduceerbaar bij re-run

Pro tip: AI als seed-script writer

Open OpenCode/Cursor. Plak Polderfest script + jouw schema. Vraag: 'pas dit aan voor mijn thema'. AI doet 't in 30 sec. Daarna jij review.

6

Chat-route + chat-UI implementeren

Dit is wat **nieuw** is. Next.js + Supabase kennen jullie al.

De chat-route

```
// app/api/chat/route.ts
import { streamText } from "ai";
import { openai } from "@ai-sdk/openai";
import { supabase } from "@/lib/supabase";

export async function POST(req: Request) {
  const { messages } = await req.json();

  // 1. Haal data op
  const { data: bands } = await supabase.from("bands").select("*");

  // 2. Format als context
  const context = bands!.map((b) =>
    `- ${b.name} (${b.genre}, ${b.tier}, ${b.day})`
  ).join("\n");

  // 3. System prompt
  const system = `Je bent een festival-assistent...
${context}
Verzin niets – gebruik alleen bovenstaande data.`;

  const result = streamText({
    model: openai("gpt-4o-mini"),
    system,
    messages,
  });
  return result.toDataStreamResponse();
}
```

De chat-pagina

```
// app/chat/page.tsx
"use client";
import { useChat } from "ai/react";

export default function ChatPage() {
  const { messages, input, handleInputChange, handleSubmit }
    = useChat();
  return (
    <main>
      {messages.map((m) => (
        <div key={m.id}>{m.role}: {m.content}</div>
      ))}
      <form onSubmit={handleSubmit}>
        <input value={input} onChange={handleInputChange} />
      </form>
    </main>
  );
}
```

Voorbeeld-vragen die we live stelden

Vraag	Wat AI doet
Welke bands spelen zaterdag op de Beach Stage?	Filtert door context
3 headliners met meeste popularity	Sorteert + select top
Hoeveel jazz fusion acts totaal?	Telt
Vat de electronic-scene samen	Samenvatting (alleen AI kan dit)
Wie was hoofdact van Polderfest 2025?	Eerlijk: 'weet ik niet'

7

Waarom data + AI samen krachtig zijn

Data alleen

- SQL queries: filter, sort, select
- Geen interpretatie, geen taal, geen samenvatting
- Gebruiker moet zelf SQL kunnen

AI alleen

- Kennis is generiek (training data)
- Verzint vaak (hallucinatie)
- Geen toegang tot live of private data

Data + AI

- AI filtert via reasoning op tekst-context
- Antwoorden in natuurlijke taal
- Samenvattingen en interpretatie
- Domein-kennis = jouw data, AI redeneert erover

Quote om mee weg te lopen

"Een LLM zonder jouw data is een gewone chatbot."

"Een LLM mét jouw data is een product."

8

Best practices & valkuilen

Doen

- Begin met gpt-4o-mini — upgrade pas als nodig
- System prompt is essentieel — 'gebruik alleen onze data'
- Stream alles — streamText voelt 5x sneller
- AI calls altijd server-side — keys blijven veilig
- Loading state — AI duurt 1-5 sec, zonder feedback voelt het stuk
- Foutafhandeling: try/catch rond elke AI-call

Niet doen

- Geen NEXT_PUBLIC_OPENAI_API_KEY — wordt zichtbaar in client
- Niet de output blind vertrouwen — AI hallucineert
- Niet alle data altijd meesturen — werkt voor 500, niet voor 50k (volgende les)
- Niet gpt-4o als default — 15x duurder dan mini

Veelvoorkomende fouten

Fout	Oorzaak	Oplossing
OPENAI_API_KEY is not defined	.env.local niet geladen	Dev server herstarten
Seed: permission denied	Anon key i.p.v. service role	SUPABASE_SERVICE_ROLE_KEY
AI antwoordt in Engels	Niet om NL gevraagd	System prompt aanpassen
AI verzint feiten	System prompt te zwak	'Verzin niets, gebruik alleen onze data'
Chat laadt niet	useChat zonder streamText	toDataStreamResponse() retourneren

9

Wat komt hierna? Tool Calling teaser

Het schaalprobleem

Vandaag sturen we alle 500 bands mee als context bij elke request. ~30k tokens. Werkt prima voor 500. Werkt **niet** voor 5.000 of 50.000.

Volgende les: Tool Calling (Les 12)

AI besluit zelf welke functie aan te roepen:

```
const { text } = await generateText({
  model: openai("gpt-4o-mini"), messages,
  tools: {
    searchBands: tool({
      description: "Zoek bands op dag, stage, genre",
      parameters: z.object({
        day: z.string().optional(),
        stage: z.string().optional(),
      }),
      execute: async ({ day, stage }) => {
        const { data } = await supabase.from("bands")
          .select("*").eq("day", day);
        return data;
      },
    }),
  },
  maxSteps: 5,
});
```

Workflow: user vraagt → AI roept searchBands aan → krijgt 60 bands → antwoordt. Schaalbaar, slim, multi-step.

Daarna in deze leerlijn

- **Les 13:** Agents + maxSteps (autonome multi-step)
- **Les 14:** RAG + embeddings (semantic search, grote datasets)
- **Les 15-16:** Testing + Deployment + Performance
- **Les 17-18:** Eindopdracht-werkdagen + Pitch

10 Bronnen

Vercel AI SDK

- Hoofdpagina: ai-sdk.dev/docs/introduction
- streamText: ai-sdk.dev/docs/reference/ai-sdk-core/stream-text
- useChat: ai-sdk.dev/docs/reference/ai-sdk-ui/use-chat
- Tool Calling (volgende les): ai-sdk.dev/docs/foundations/tools

Supabase

- JS Client: supabase.com/docs/reference/javascript
- Row Level Security: supabase.com/docs/guides/auth/row-level-security
- Server-side usage: supabase.com/docs/guides/auth/server-side

Inspiratie

- v0.dev — Generative UI
- chat.vercel.ai — Officiële demo
- Vercel templates AI: vercel.com/templates?type=ai