Les 15 — Docenttekst (normaal)

Fysieke les, 3 uur. Voor uitgebreide voorbereiding + context per slide. Voor letterlijk voorleesbaar script: zie `Les15-Docenttekst-Autocue.md`**.**

Lesvorm en flow

Onderdeel	Indeling
Theorie + 4 live demos	±1 uur klassikaal
Pauze	15 minuten
Hands-on (Polderfest deploy via 3 chats)	±1 uur basis
Extra werk (eigen feature toevoegen)	±30 min als tijd over

Focus: koppelen + gebruiken. Climax: drie chats die GitHub, Polderfest+Supabase, en Vercel MCP samen aan het werk zetten.

Voorbereiding — een week vooraf

Studenten checklist (stuur via Slack):

- Supabase project draait nog (uit Les 11/12)
- Cursor + Pro Student actief
- Node ≥ 18
- Personal Access Tokens **vooraf** aangemaakt:
- GitHub PAT (fine-grained)
- Supabase Personal Access Token (account-level)
- Vercel account (geen token vooraf — OAuth bij eerste gebruik)

Jouw eigen voorbereiding:

- `~/.cursor/mcp.json` met VIER servers vooraf werkend (github + supabase + vercel + polderfest)
- Vercel OAuth-flow vooraf doorlopen (anders kost dat live tijd)
- Polderfest MCP finished `npm install` + `.env.local` gedaan
- Lokaal `npm run dev` getest — `localhost:3000` werkt
- Eén git commit gedaan in polderfest-mcp folder (klaar voor Chat 1)
- Backup productie-URL op Vercel klaar als demo-fallback

Voorbereiding — dag van de les

30 minuten vooraf:

- Cursor open met 4 MCP-indicators groen
- Twee terminals: één in polderfest-mcp folder met `npm run dev`, één leeg
- Browser tabs: github.com (logged in), Supabase dashboard, Vercel dashboard
- Slides in fullscreen

Backup scenario:

- GitHub PAT verlopen → tweede PAT klaar
- Vercel MCP OAuth weigert → val terug op `vercel link` + `vercel --prod` manueel
- MCP-indicator komt niet → herstart Cursor LIVE (laat zien hoe)
- Een chat raakt verstrikt → ALTIJD nieuwe chat openen, niet doorgaan in vervuilde context

Per slide — extra context bij voorbereiding

Slide 1 — Title

Doel: verwachtingen zetten.

Vandaag is anders dan voorgaande lessen: minder code, meer koppelen. Climax in drie chats.

Slide 2 — Vandaag

Doel: roadmap zetten — drie blokken + climax expliciet noemen.

Maak vooraf duidelijk dat de drie chats het hoofdmoment zijn — daar werken we naartoe.

Slide 3-5 — Wat is MCP

Doel: basis-terminologie + waarom.

Niet te diep. Drie rollen + drie primitives noemen, doorgaan.

Slide 6 — Ecosysteem

Doel: context dat het breed is.

Niet alle servers doornemen. Een paar herkenbare voor de klas (GitHub, Slack).

Slide 7 — Vier MCPs vandaag

Doel: zet het demo-blok op.

Belangrijk: noem dat alle vier via HTTP gaan, met dezelfde patroon. Vercel doet OAuth ipv Bearer — uitleggen dat dat veiliger is.

Slide 8 — De mcp.json

Doel: de complete config in één oogopslag.

Echt de hele JSON tonen. Studenten krijgen 'm als template in lesbestanden. Wijs op:

- `type: http` — moderne syntax
- `headers.Authorization` voor Bearer-based

- `vercel` heeft GEEN header — OAuth doet de rest

Slide 9 — Tokens regelen

Doel: student weet hoe hij elk token krijgt.

Belangrijkste valkuilen:

- GitHub PAT: fine-grained, niet classic. Beperkte permissies geven.
- Supabase: ACCOUNT-level token, NIET project keys. `project_ref` uit project URL halen.
- Vercel: geen pre-token. OAuth bij eerste gebruik.

Slide 10 — Live koppelen

Doel: student ziet dat het werkt.

Live doorlopen: open `~/.cursor/mcp.json`, plak, fill tokens, save, Cmd+Q, opnieuw open, indicator groen. Voor Vercel: klik op "needs login", OAuth flow, terug naar Cursor.

Slide 11 — Polderfest MCP

Doel: student weet wat er in de zip zit.

File-tree doorlopen. Benadrukken: één bestand doet het echte werk (`route.ts`). Rest is gewone Next.js.

Slide 12 — Code-tour

Doel: student snapt `server.tool()` patroon.

Eén tool (searchBands) is genoeg. Andere drie volgen hetzelfde patroon.

Slide 13 — Lokaal draaien

Doel: student kan dit straks zelf.

Live: commando's draaien, polderfest in mcp.json toevoegen, herstart Cursor, vraag stellen.

Slide 14 — De drie chats workflow

Doel: climax aankondigen.

Maak expliciet dat het DRIE aparte chats zijn — niet één lange chat. Reden: elke chat heeft één focus, mcp.json kan dan beter de juiste tools aanroepen.

Slide 15 — Chat 1: GitHub MCP

Doel: student ziet GitHub MCP echt aan het werk.

Demo-aanwijzingen:

- Nieuwe chat (Cmd+L, of via "+" knop)
- Plak de exacte prompt van de slide
- Wacht tot Cursor tool-call `create_repository` doet
- Toon github.com om te bevestigen
- Cursor draait daarna `git remote add` + `git push` automatisch in terminal

Backup: als GitHub MCP traag is — gewoon `gh repo create` zelf doen, maar wel commentaar geven dat MCP dit ook had gekund.

Slide 16 — Chat 2: Polderfest features

Doel: zien dat Supabase MCP + Cursor + Polderfest MCP samenwerken in één chat.

Demo-aanwijzingen:

- VOLLEDIG nieuwe chat (Cmd+L) — niet vorige chat hergebruiken
- Plak prompt — Cursor zal eerst Supabase MCP aanroepen (vraagt bevestiging)
- Daarna genereert Cursor de tools in route.ts
- Accept all → save → `git add . && git commit -m "feat: reviews"` + push
- Test in DEZELFDE chat met de twee review-vragen — die gaan via Polderfest MCP

Belangrijke uitleg na demo: drie verschillende MCPs samen aan het werk in één chat. Cursor kiest de juiste.

Slide 17 — Chat 3: Vercel MCP

Doel: zien dat Vercel MCP een complete deploy doet.

Demo-aanwijzingen:

- Nieuwe chat
- Plak prompt met env vars (vooraf in een notitie klaar — niet live tikken want geheim)
- Vercel MCP roept create_project, link_repo, set_env_vars, deploy aan
- Wacht op productie-URL (~45s)
- Open URL in browser om te bevestigen
- Update polderfest in mcp.json naar productie-URL → herstart Cursor → vraag in chat

Belangrijke afsluiting: "Drie chats, vier MCPs, geen terminal-commando's zelf getypt voor github, supabase of vercel."

Slide 18 — Lesopdracht

Doel: overgang naar zelf doen.

Lees de acht stappen voor. Geen deliverables. Wijs op: een mcp.json template staat in lesbestanden — invullen met eigen tokens.

Slide 19 — Extra werk

Doel: snel-werkers bezig houden.

Hetzelfde patroon als Chat 2 uit jouw demo. Studenten kunnen letterlijk de demo kopiëren met een eigen feature-idee.

Slide 20 — Vooruitblik

Doel: voorbereiding Les 16.

Veelvoorkomende vragen + antwoorden

"Werkt OAuth Vercel altijd?" Bij eerste login wel. Als token verloopt verschijnt indicator rood — opnieuw klikken op "needs login".

"Moet Polderfest in dezelfde mcp.json staan als de andere drie?" Ja — zo hebben we 4 indicators tegelijk groen.

"Wat als Supabase MCP read-only is — kan hij dan toch tabellen maken?" Nee, dan zal hij dat weigeren. Zet `read_only=true` op `false` als je migrations wilt doen via MCP. Voor demo: dat is wat we doen.

"Kan dit ook in Claude Desktop?" Ja — `~/Library/Application Support/Claude/claude_desktop_config.json` met dezelfde structuur.

"Wat is het verschil tussen Bearer + OAuth?" Bearer = vooraf token in config. OAuth = browser-flow bij eerste gebruik, token wordt opgeslagen. Vercel doet OAuth om veiligheidsredenen.

Foutmodi tijdens de les

Probleem	Snelle fix
MCP-indicator komt niet	Cmd+Q Cursor en opnieuw openen
mcp.json werkt niet	Valideer JSON op jsonlint.com — één komma fout = ALLE servers vallen
GitHub MCP geeft 401	PAT verlopen of geen rechten
Supabase MCP geeft 403	Token is project-key — heb account-level nodig, OF read_only blokkeert
Vercel "Needs login" blijft hangen	Browser pop-up geblokkeerd? Manueel naar mcp.vercel.com
Vercel MCP kan geen project maken	Vercel account permissies — check je teamselect
Productie-URL geeft 500	Env vars niet (goed) gezet — vercel logs --follow

Vooruitblik Les 16

Volgende les: RAG met embeddings. PDF Q&A; app from scratch. Studenten moeten Polderfest MCP live hebben op Vercel + vier MCP-servers gekoppeld in Cursor.