Hybrid Topic

**Next-auth**

* **Door**: Catherine Tahon
* **Demo datum**: 9 maart 2025
* **Aspecten**:
  + Authenticatie creëren (github)
* **Code url github**:
  + <https://github.com/CatherineTahon/test-auth.git>

# Korte uitleg topic

Ik wil graag authenticatie maken op mijn webapp. Deze authenticatie maakt gebruik van verschillende providers zoals bijvoorbeeld GitHub, maar ook een email met passwoord is mogelijk. In deze topic ga ik **enkel** gebruik maken van de GitHub authenticatie mogelijkheid. Maar weet dat er nog veel meer mogelijkheden zijn. <https://next-auth.js.org/providers/>

# Aspect 1: Authenticatie

## Korte beschrijving

Ik leg uit hoe de authenticatie werkt via de applicatie maar ook hoe je client-id en client-secrets ophaalt van GitHub om zo deze accounts te koppelen aan de lokale webapp en aan de production webapp.

## Prerequisites

Wat moet al gekend zijn voor je aan dit topic begint:

* Hoe je een terminal opent
* Eigen naam en passwoord github

Welke tools moeten geinstalleerd zijn voor je aan dit topic kan beginnen:

* NextAuth
* Een bestaand project waar NextAuth op mag en kan gecodeerd worden.

Info om deze tools te installeren (als het gaat om iets dat we **niet** in de lessen gezien hebben):

* In het project command: npm install next-auth

## Stappen

* **Client id en secret (Github)**
  + **Login op github en ga naar volgende link** [**https://github.com/settings/developers**](https://github.com/settings/developers)
  + **Rechts zie je een knop New OAuth app – klik daarop**
  + **Afbeelding met Lettertype, tekst, schermopname, Graphics

    Door AI gegenereerde inhoud is mogelijk onjuist.**
  + **Vul de naam van je app in**
  + **Homepage url** 
    - **!Letop in dev mode ga je gebruik maken van jouw local address =** [**http://localhost:3000/api/auth/callback/github**](http://localhost:3000/api/auth/callback/github)
    - **Wanneer je dit wil testen in prod mode ga je dit aanpassen naar jouw domein =** [**https://catherine-tahon-eindopdracht-catherines-projects-11df764d.vercel.app/api/auth/callback/github**](https://catherine-tahon-eindopdracht-catherines-projects-11df764d.vercel.app/api/auth/callback/github)
  + **Dan klik je op de knop onderaan genaamd Register Application Afbeelding met tekst, Lettertype, schermopname, groen

    Door AI gegenereerde inhoud is mogelijk onjuist.**
  + **Nu vind je jouw client-id**
  + **Klik op de button generate a new client secret**
  + **Houd zowel de client-id als de client secret bij de hand deze heb je straks nodig**
* **AuthSecret**
  + **Open jouw terminal op je pc**
  + **Type** Node **en druk op enter om Node.js shell te openen**
  + **Je krijg dit te zien:** Welcome to Node.js v20.12.2.
  + **Daarna type je** console.log(require('crypto').randomBytes(32).toString('base64')); **en druk enter**
  + **Je krijgt dan jouw secret key** NEXTAUTH\_SECRET=your-random-secret
  + **Deze mag je even bewaren, we gaan deze straks gebruiken**
* **Authenticatie in webapp**
* **Creëer een file waarin je duidelijk maakt welke providers gebruik maken van deze authenticatie op de website**

**Folder: C:\Codes\test-auth\src\app\api\auth\[...nextauth]\route.ts**

import NextAuth from "next-auth";

import GitHubProvider from "next-auth/providers/github";

export const authOptions = {

providers: [

GitHubProvider({

clientId: process.env.GITHUB\_ID ?? "",

clientSecret: process.env.GITHUB\_SECRET ?? "",

}),

],

};

export const handler = NextAuth(authOptions);

export { handler as GET, handler as POST };

* **We gaan gebruik maken van useSession() hiermee kan je makkelijk zien of er een gebruiker is ingelogd of niet. We zetten deze import in een aparte file**

**Folder: C:\Codes\test-auth\src\components\SessionProvider.tsx**

"use client";

import { SessionProvider } from "next-auth/react";

export default SessionProvider

* **In de frontend ga je ervoor zorgen dat er een controle is of de user wel ingelogd is of niet Folder: C:\Codes\test-auth\src\components\NavMenu.tsx**

"use client";

import Link from "next/link";

import { signIn, signOut, useSession } from "next-auth/react";

import { usePathname } from "next/navigation";

function AuthButton() {

const { data: session } = useSession();

return (

<div className="text-center mb-4">

{session ? (

<>

<p className="font-bold text-lg">Welcome, {session?.user?.name}</p>

<button

onClick={() => signOut()}

className="mt-2 text-black rounded-full"

>

Sign out

</button>

</>

) : (

<>

<p className="font-semibold text-lg">Not signed in</p>

<button

onClick={() => signIn()}

className="mt-2 text-black rounded-full"

>

Sign in

</button>

</>

)}

</div>

);

}

export default function NavMenu() {

const pathname = usePathname();

return (

<div className="flex flex-col items-center">

<AuthButton />

<hr />

<ul className="flex flex-col gap-2">

<Link href="/">

<li>

Home

</li>

</Link>

<Link href="/protected">

<li>

Protected Route

</li>

</Link>

</ul>

</div>

);

}

* **We gaan naar de layout folder in de root van onze applicatie om de SessionProvider dan ook beschikbaar te maken op onze volledige applicatie**

**Folder: C:\Codes\test-auth\src\app\layout.tsx**

import { getSession } from "next-auth/react";

import SessionProvider from "../components/SessionProvider";

import NavMenu from "@/components/NavMenu";

import "bootstrap/dist/css/bootstrap.min.css";

export default async function RootLayout({ children }: { children: React.ReactNode }) {

return (

<html lang="en">

<body>

<SessionProvider>

<main>

<NavMenu />

{children}

</main>

</SessionProvider>

</body>

</html>

);

}

* **Simpele Homepage**

**Folder: C:\Codes\test-auth\src\app\page.tsx**

export default function Page() {

return (

<div className="items-center justify-center p-4">

<h1>Home Page</h1>

<p>Welcome to the Home Page!</p>

</div>

);

}

* **We maken een protectedpage aan om te kijken wat er gebeurd als de user niet ingelogd is. Hier gaan we dus gebruik maken van session.**

**Folder: C:\Codes\test-auth\src\app\protected\page.tsx**

import { getServerSession } from "next-auth";

import { redirect } from "next/navigation";

export default async function ProtectedRoute() {

const session = await getServerSession();

if (!session || !session.user) {

redirect("/api/auth/signin");

}

return (

<div className="items-center justify-center p-4">

<h1>Protected Page</h1>

<p>Welcome to the protected content!</p>

</div>

);

}

* **Autorisatie lokaal draaien**
  + **De client id en client secret die we in het begin hebben aangemaakt gaan we nu gebruiken. We maken een .env file aan waar we die informatie in plaatsen. Dit moet gebeuren om een connectie te kunnen maken met GitHub.**

**Folder: C:\Codes\test-auth\.env**

GITHUB\_ID=eigen github user id uit stap 1

GITHUB\_SECRET=eigen github secret uit stap 1

NEXTAUTH\_SECRET=eigen secret uit stap 1

NEXTAUTH\_URL=http://localhost:3000

* **Autorisatie in production draaien**
  + **De client id en client secret die we in het begin hebben aangemaakt met daarin de homepage-url van jouw webapp die op vercel staat gaan we nu gebruiken.**
  + **Je gaat deze id en secret dit keer niet in de .env file zetten want deze word toch niet gecommit omdat dit secrets zijn. Je zal deze bij vercel zelf moeten invoeren.**
    - **Ga naar jouw vercel project**
    - **Klik dan op settings Afbeelding met tekst, software, Multimediasoftware, schermopname

      Door AI gegenereerde inhoud is mogelijk onjuist.**
    - **Klik op environment variables**
    - **Afbeelding met tekst, schermopname, Lettertype

      Door AI gegenereerde inhoud is mogelijk onjuist.**
    - **Hier ga je jouw key-value pairs ingeven Afbeelding met tekst, schermopname, software, Multimediasoftware

      Door AI gegenereerde inhoud is mogelijk onjuist.**
    - **Afbeelding met tekst, schermopname

      Door AI gegenereerde inhoud is mogelijk onjuist.**
    - **Druk daarna op save**

## Debug

Ik had zelf moeite met tew eten of mijn id en secrets juist aankwamen in mijn route.ts daarom maakte ik gebruik van een console.log. Als ik dan mijn code uitvoerde kon ik in mijn terminal in Visual Studio Code (server-side) terug vinden of mijn client-id en client-secret effectief ingevuld waren of dat deze “undefined” waren.

*import* NextAuth *from* "next-auth";

*import* GitHubProvider *from* "next-auth/providers/github";

*export* const authOptions = {

  providers: [

    GitHubProvider({

      clientId: process.env.GITHUB\_ID ?? "",

      clientSecret: process.env.GITHUB\_SECRET ?? "",

    }),

  ],

};

console.log("GitHub ID:", process.env.GITHUB\_ID);

console.log("GitHub Secret:", process.env.GITHUB\_SECRET);

*export* const handler = NextAuth(authOptions);

*export* { handler *as* GET, handler *as* POST };

## Referenties

* [**https://next-auth.js.org/getting-started/example**](https://next-auth.js.org/getting-started/example)
* [**https://www.youtube.com/watch?v=md65iBX5Gxg**](https://www.youtube.com/watch?v=md65iBX5Gxg) **!heel erg interessant!**

# Conclusie

Ik vond dit een interessante topic, zeker omdat ik een authenticatie in mijn applicatie wou gebruiken. Nu weet ik hoe dit werkt en hoe ik andere providers eraan koppel (bv GitHub). Allemaal doormiddel van een Client-id en Client-secret.