Technisch ontwerp

Twan Asselbergs - Portfolio

Door: Twan Asselbergs

Datum: 25 oktober 2024

Versie: 1.0

Inhoudsopgave

[1. Inleiding 3](#_Toc181364420)

[1.1. Gekozen software en tools 3](#_Toc181364421)

[1.2. Gekozen programmeertaal en databasesysteem 4](#_Toc181364422)

[1.3. Entiteit Relatie Diagram 4](#_Toc181364423)

[1.4. Technische uitwerking – Posts ophalen met Axios 4](#_Toc181364424)

# Inleiding

Het doel van dit project is om een portfolio-website voor een softwareontwikkelaar te creëren, gebruikmakend van een headless CMS zoals WordPress, Drupal, enzovoort. De frontend is volledig zelf gebouwd, waarbij React.js als framework is gekozen. Voor de backend is gekozen voor een headless WordPress-oplossing. De website laadt onder andere projectinformatie met aanpasbare content vanuit WordPress en heeft een minimalistische, moderne uitstraling.

## Gekozen software en tools

**Frontend Framework**: **React.js**

* **Waarom**: Ik heb React.js gekozen vanwege de component-gebaseerde structuur en omdat het een industrie-standaard is. Het is een waardevolle technologie om te leren en biedt veel flexibiliteit.

**CSS Framework**: **Tailwind CSS**

* **Waarom**: Ik heb voor Tailwind CSS gekozen omdat het een efficiëntere en modernere benadering biedt dan standaard CSS. Tailwind heeft ingebouwde templates voor kleuren en afmetingen, waardoor het eenvoudig is om een consistent design te creëren.

**Backend CMS**: **WordPress (Headless CMS)**

* **Waarom**: WordPress leek me een leuke uitdaging omdat het, net zoals React.js, veel wordt gebruikt. Ik vond het voor dit project leuk om ervaring op te doen met de populairste technologieën. Door WordPress als headless CMS te gebruiken, wordt de frontend volledig losgekoppeld, wat mij meer vrijheid geeft om zelf een design te bouwen.

**REST API Requests**: **Axios**

* **Waarom**: Ik gebruik Axios voor REST API-aanroepen omdat het eenvoudiger in gebruik is dan de native fetch API en betere foutafhandeling biedt.

## Gekozen programmeertaal en databasesysteem

**Programmeertalen:**

* **JavaScript**: Ik heb JavaScript gebruikt voor de frontend (React.js) omdat het de standaard is voor dynamische websites en goed werkt in alle browsers.
* **PHP**: Ik heb PHP gebruikt voor de backend (WordPress) omdat WordPress hierop gebaseerd is en veel mogelijkheden biedt via plugins en uitbreidingen.

**Databasesysteem: MySQL**

* **Waarom**: WordPress gebruikt MySQL als databasesysteem. Het is krachtig, werkt goed samen met PHP en is betrouwbaar voor het efficiënt verwerken van data.

## Entiteit Relatie Diagram

* **Posts**: De primaire entiteit waarin elk projectpost wordt opgeslagen.
* **Users**: Gebruikers van het CMS met verschillende rollen zoals Admin en Editor.
* **Custom Fields**: Specifieke velden toegevoegd aan de standaard Post voor extra content, zoals featured\_image\_url, github\_link, of icons (Deze gebruik ik zelf ook in mijn portfolio).

## Technische uitwerking – Posts ophalen met Axios

In de ProjectsSection-component maak ik gebruik van de useEffect-hook in combinatie met Axios om een HTTP GET-verzoek naar de WordPress-API te sturen. Dit verzoek haalt een lijst van projecten op als JSON-data. Deze data wordt vervolgens opgeslagen in de posts-state en in de component weergegeven en gestyled met Tailwind CSS.

Hier een stukje code uit mijn ProjectsSection.jsx:

useEffect(() => {

let url = `${process.env.REACT\_APP\_API\_ROOT}/posts`;

axios.get(url).then((res) => {

setPosts(res.data);

});

}, []);