## **🌌 Hell Week – Word een Jedi van de Laravel AI-kracht**

**Amine Ben Slimene, Didier Vanassche, Robin Caulier**

Laatste update: 05-05-2025



Welkom jonge Padawan 🧑‍🚀, Vandaag begint jouw training in de oude kunsten van Laravel en Livewire. Je missie? Maak een magische knop die met behulp van AI automatisch productbeschrijvingen genereert – en dat alles in stijl... met de kracht van de Force ⚔️

Pak je lightsaber (terminal), roep de kracht aan (Livewire), en neem de controle over je CMS zoals een ware Jedi-meester.

## **🗓️ Dag 1 – De Start van het Avontuur: Project Setup**

## **🎯 Wat is het doel?**

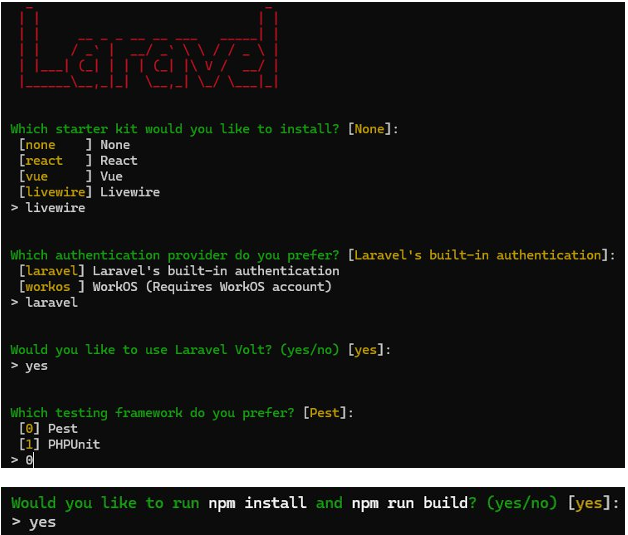
Elke Jedi heeft zijn eigen lichtzwaard nodig... en jij hebt jouw eigen Laravel-project nodig. In deze eerste stap leer je hoe je je eigen Laravel-sterrenstelsel creëert met Livewire, Laravel Volt en ingebouwde authenticatie. Geen kloons of oude projecten deze keer — jij bouwt het helemaal zelf, vers uit de oven van de Artisan Forge. Klaar? Bouw de basis van je Force!

## **✅ Actie 1: Een Nieuw Laravel-Project Aanmaken**

Gebruik deze krachtlijn in je terminal:

livewire new laravelApp

Je krijgt dan een **interactieve wizard** zoals op de afbeelding . Kies **exact** deze instellingen:



1. Which starter kit would you like to install?  
    👉 **Livewire**
2. Which authentication provider do you prefer?  
    👉 **Laravel's built-in authentication**
3. Would you like to use Laravel Volt?  
    👉 yes
4. Which testing framework do you prefer?  
    👉 0 (**Pest**)
5. Would you like to run npm install and npm run build?  
    👉 yes

⚠️ Deze keuzes zorgen ervoor dat je project **volledig klaar staat** met Livewire, Tailwind CSS, auth, Laravel Volt en tooling.

## **✅ Actie 2: Open je project in PhpStorm**

Open je project in PhpStorm met deze stappen:

1. Start PhpStorm
2. Kies **"Open"**
3. Navigeer naar je map laravelApp
4. Klik op **Open**

💡 *Tip: stel in PhpStorm Laravel & Blade support in voor betere auto-completion en highlighting.*

### **⚡ Open een nieuw terminalvenster en voer uit:**

composer run dev

**🗓️ Dag 2 – Creatie van Producten: Bouw de Toekomst van je App**

**🎯 Wat is het doel?**

Een Jedi zonder voorraad is slechts een monnik met een cape. In deze stap leer je hoe je producten toevoegt aan je applicatie: van het bouwen van modellen en migraties tot het gebruik van componenten die je winkelgalaxy doen draaien.

### **🧱 1) De products-tabel aanroepen 📦**

Gebruik de Jedi-spreuk om een nieuwe tabel in de database te roepen:

php artisan make:migration create\_products\_table

### **📜 2) Definieer de tabelstructuur in de migration 🧾**

Schema::create('products', function (Blueprint $table) {  
 $table->id();  
 $table->string('name');  
 $table->text('description')->nullable();  
 $table->decimal('price', 8, 2)->default(0);  
 $table->timestamps();  
});

🧠 *Waarom?* – Omdat een Jedi altijd structuur kent. Zonder velden geen data. En zonder data... geen beschrijving.

### **🧙‍♂️ 3) Roep het model tot leven ✨**

php artisan make:model Product

### **🔧 4) Activeer het model met de juiste velden ⚙️**

class Product extends Model  
{  
 protected $fillable = ['name', 'description', 'price'];  
}

🧠 *Waarom?* – fillable bepaalt wat een Padawan mag vullen. Zonder dit zal de Force je formulier niet verstaan.

### **🔁 5) Voer de migratie uit – En geef vorm aan de database 🌌**

php artisan migrate:fresh --seed

🧠 *Waarom?* – Elke Jedi heeft een tempel. Dit is die van jouw producten.

### **🛠️ 6) Bouw het overzicht – De ShowProducts-component 📊**

php artisan make:livewire products.show-products

🧠 *Waarom?* – Want een Jedi moet kunnen kijken naar zijn creaties. Een lijst van producten, jouw eerste holokron.

### **🔗 7) Verbind het pad van de Force – Maak de route actief 🗺️**

use App\Livewire\Products\ShowProducts;  
Route::get('/products', ShowProducts::class)->name('products.index');

🧠 *Waarom?* – Zonder route, geen toegang. Je moet de weg weten naar de products-pagina.

## **⚙️ 8) Logica van de component – Code van ShowProducts**

<?php  
namespace App\Livewire\Products;  
use Livewire\Component;  
use App\Models\Product;  
class ShowProducts extends Component  
{  
 public $products;  
 public function mount()  
 {  
 $this->products = Product::all();  
 }  
 public function render()  
 {  
 return view('livewire.products.show-products');  
 }  
}

### **✨ Wat doet deze code?**

* **mount()** : wordt automatisch uitgevoerd wanneer de component wordt geladen. Hier haalt hij **alle producten** uit de database.
* **render()** : retourneert de Livewire view die de productenlijst toont.

## **🖼️ 9) De Livewire View – resources/views/livewire/products/show-products.blade.php**

<div class="p-4">  
 <h2 class="text-xl font-bold mb-4">Productenlijst</h2>  
 <div class="flex justify-end mb-4">  
 <a href="{{ route('products.create') }}"  
 class="bg-green-500 text-white px-4 py-2 rounded hover:bg-green-600">  
 Nieuw Product  
 </a>  
 </div>  
  
 <table class="table-auto w-full border">  
 <thead>  
 <tr class="bg-gray-200">  
 <th class="border px-4 py-2">ID</th>  
 <th class="border px-4 py-2">Naam</th>  
 <th class="border px-4 py-2">Beschrijving</th>  
 <th class="border px-4 py-2">Prijs</th>  
 </tr>  
 </thead>  
 <tbody>  
 @forelse($products as $product)  
 <tr>  
 <td class="border px-4 py-2">{{ $product->id }}</td>  
 <td class="border px-4 py-2">{{ $product->name }}</td>  
 <td class="border px-4 py-2">{{ $product->description }}</td>  
 <td class="border px-4 py-2">  
 € {{ number\_format($product->price, 2, ',', '.') }}  
 </td>  
 </tr>  
 @empty  
 <tr>  
 <td colspan="4" class="border px-4 py-2 text-center">  
 Geen producten gevonden.  
 </td>  
 </tr>  
 @endforelse  
 </tbody>  
 </table>  
</div>

### **📋 Wat toont de view?**

* Een titel en een knop **“Nieuw Product”**.
* Een tabel met de kolommen: ID, Naam, Beschrijving, Prijs.
* Als er geen producten zijn: een duidelijke melding “Geen producten gevonden”.

### **🧭 9) Zet de link in je navigatie – Zodat de Jedi weet waar te klikken 🧭**

<flux:navlist.item  
 icon="cube"  
 :href="route('products.index')"  
 :current="request()->routeIs('products.index')"  
 wire:navigate  
>  
 {{ \_\_('Producten') }}  
</flux:navlist.item>

🧠 *Waarom?* – Een Jedi zonder navigatie is als een droid zonder richting. De link leidt je naar het overzicht van de producten.

### **✅ 10) TEST: Navigeer als een jonge Jedi naar de lijst van producten ⚔️**

Open /products in je browser. Als de lijst verschijnt, is je kracht al gegroeid...

### **🥋 Dag 3 – De Vorming van Creatie: Producten maken met de Kracht ✍️**

De Jedi kan niet alleen observeren, hij moet ook scheppen. Tijd om de pagina voor het creëren van producten te maken – mét een magische AI-knop die beschrijvingen schrijft zoals een meester-copywriter.

“Niet enkel observeren, een Jedi moet ook scheppen. Code schrijven, jij zult.” – Meester Yoda

### **🔨 10) Maak de Livewire-component voor creatie van producten ✍️**

Gebruik deze Force-opdracht:

php artisan make:livewire products.create-product

🧠 *Waarom?* – Omdat zelfs een jonge Jedi producten moet kunnen creëren. Deze component toont een formulier voor nieuwe items.

### **🧠 11) Codeer de CreateProduct-component (de controlekamer van je formulier)**

Open app/Livewire/Products/CreateProduct.php en voeg toe:

namespace App\Livewire\Products;  
  
use Livewire\Component;  
use App\Models\Product;  
use Illuminate\Support\Facades\Http;  
  
class CreateProduct extends Component  
{  
 public $form = [  
 'name' => '',  
 'description' => '',  
 'price' => '',  
 ];  
  
 public function save()  
 {  
 Product::create($this->form);  
 session()->flash('success', 'Product succesvol aangemaakt.');  
 return redirect()->route('products.index');  
 }  
  
 public function render()  
 {  
 return view('livewire.products.create-product');  
 }  
}

🧠 *Waarom?* – Deze component beheert het formulier en de logica om een product op te slaan in je database.

### **🖼️ 12) Bouw de HTML-view create-product.blade.php 🧾**

<div class="p-6">  
 <h2 class="text-2xl font-bold mb-4">Nieuw Product</h2>  
  
 <form wire:submit.prevent="save" class="space-y-4 max-w-xl">  
 <div>  
 <label for="name" class="block font-semibold">Naam</label>  
 <input type="text" wire:model.defer="form.name" id="name" class="w-full border rounded p-2" />  
 </div>  
  
 <div>  
 <label for="description" class="block font-semibold">Beschrijving</label>  
 <textarea wire:model.defer="form.description" id="description" class="w-full border rounded p-2"></textarea>  
 </div>  
  
 <div>  
 <label for="price" class="block font-semibold">Prijs (€)</label>  
 <input type="number" wire:model.defer="form.price" step="0.01" id="price" class="w-full border rounded p-2" />  
 </div>  
  
 <button type="submit" class="bg-blue-500 text-white px-4 py-2 rounded">Opslaan</button>  
 </form>  
</div>

🧠 *Waarom?* – Dit is het visuele gedeelte van je creatiepagina. Het gebruikt Livewire om alle data te koppelen aan de component.

### **🌐 13) Voeg de route toe aan je web.php 📡**

use App\Livewire\Products\CreateProduct;  
  
Route::get('/products/create', CreateProduct::class)->name('products.create');

🧠 *Waarom?* – Zonder route, geen toegang. Jedi’s moeten weten waar hun creatie-tempel zich bevindt.

### **✨ 14) Voeg een knop toe in de lijst om een nieuw product te maken**

In show-products.blade.php, boven je tabel:

<div class="flex justify-end mb-4">  
 <a href="{{ route('products.create') }}" class="bg-green-500 text-white px-4 py-2 rounded hover:bg-green-600">  
 Nieuw Product  
 </a>  
</div>

### **🔙 15) Voeg een knop "← Terug naar lijst" toe aan create-product.blade.php**

<a href="{{ route('products.index') }}" class="inline-block mb-4 text-blue-600 hover:underline">  
 ← Terug naar lijst  
</a>

🧠 *Waarom?* – Zelfs een Jedi keert terug naar zijn oorsprong. Met stijl.

### **✅ 16) Test: Gebruik de pagina en creëer je eerste product 🚀**

➡️ Navigeer naar /products/create  
 ➡️ Vul het formulier in  
 ➡️ Klik op **Opslaan**  
 ➡️ Word begroet met succes ✨

## **🥋 Dag 4 – De Kracht Ontketenen: Beschrijvingen genereren met AI 🤖✨**

“Sterk is de AI in deze... Maar niet voor iedere hand bestemd.” – Meester Yoda

Nu komt de magie. We gaan een **AI-knop** toevoegen die automatisch productbeschrijvingen genereert via de **gratis Groq API**.

### **🔑 17) Vraag je gratis Groq API key aan**

1. Ga naar: <https://console.groq.com/keys>
2. Meld je aan met Google of GitHub
3. Klik op “Create API Key”
4. Kopieer je sleutel (ze ziet eruit als gsk\_xxxxxxxxxx)

### **⚙️ 18) Voeg de sleutel toe aan .env**

GROQ\_API\_KEY=gsk\_xxxxxxxxxxx

In config/services.php, voeg toe:

'groq' => [  
 'key' => env('GROQ\_API\_KEY'),  
],

### **💬 19) Voeg de AI-magie toe in je component**

In CreateProduct.php, voeg een nieuwe methode toe:

public function generateDescription()  
{  
 if (empty($this->form['name'])) {  
 $this->addError('form.name', 'Geef eerst een productnaam op.');  
 return;  
 }  
  
 $response = Http::withToken(config('services.groq.key'))  
->withOptions(['verify' => false]) // <-- disable SSL check  
->post('https://api.groq.com/openai/v1/chat/completions', [  
 'model' => 'llama3-70b-8192',  
 'messages' => [  
 ['role' => 'system', 'content' => 'Je bent een copywriter voor een webshop.'],  
 ['role' => 'user', 'content' => 'Schrijf een korte, aantrekkelijke beschrijving voor het product: ' . $this->form['name']],  
 ],  
 'temperature' => 0.7,  
 ]);  
  
 $data = $response->json();  
 $this->form['description'] = $data['choices'][0]['message']['content'] ?? 'Geen beschrijving gevonden.';  
}

### **🧪 20) Voeg de knop toe in create-product.blade.php**

Onder het veld name, zet dit:  
<div class="flex items-center gap-2 mt-2">  
 <button type="button"  
 wire:click="generateDescription"  
 class="bg-purple-600 text-white px-3 py-1 rounded hover:bg-purple-700">  
 Genereer Beschrijving  
 </button>  
 <span class="text-sm text-gray-500">Gebruik de AI om automatisch een beschrijving te maken.</span>  
</div>

🧠 *Waarom?* – Want met grote kracht komt grote verantwoordelijkheid. Alleen de Jedi-meesters (admins) mogen klikken.

## **✅ 23) Test alles**

* 🧪 Log in
* 🧪 Navigeer naar /products/create
* 🧪 Vul de naam in en klik op **Genereer Beschrijving**
* 🧪 De AI vult de beschrijving automatisch in 🤖

## **🌟 Conclusie – De Laatste Stap van de Padawan**

Gefeliciteerd, jonge Padawan! 🧑‍🚀  
 Je hebt de Force van Laravel, Livewire én AI onder de knie gekregen.  
 Je hebt een magische knop gebouwd die schrijft met de kracht van Groq 🤯

Je hebt bewezen dat je niet zomaar een ontwikkelaar bent...  
 **je bent een Laravel Jedi!** ⚔️

Dus sluit je laptop, pak je lightsaber (of een koffie),  
 en onthoud:

“Code, commit, en push. Rusten, een Jedi daarna mag.” – Meester Yoda ☕️

