Producten laten zien vanuit de database en de kracht van Eloquent

Nu je de database gemaakt hebt en producten erin hebt gezet, moet je nog de producten ophalen vanuit de database en tonen op het scherm.

Hoe haal je jouw producten op uit de database?

Zoals in eerdere blokken aangegeven, kan je communiceren met de database via Models. Om aan te geven welk model (dus welk tabel) data wil ophalen, moet je die aangeven in de Controller.

Data opvragen doe je in de Controller

Weet je nog wat de controller doet? Voor recap:
Controllers zorgt voor de logica en verbindt de **Model** met de **View**. In de uitleg van het **MVC model** was de **Controller** de ober. Die moet ervoor zorgen dat de bestellingen van de klant (de view) goed wordt doorgegeven aan de chef (model).

Maar hoe geef je dan aan dat je jouw producten wil via de `Product` model? Daarvoor hebben we **Eloquent**.

Wat is Eloquent?

Eloquent zorgt ervoor dat je **CRUD** acties aansturen met jouw **Models**. In dit geval voor jouw producten, kan je dus **Eloquent** gebruiken bij de **Product** model. Zo kan je nu bijvoorbeeld jouw producten ophalen (C**R**UD van **Read)**

Hoe haal je data op uit een tabel?

Om data op te halen uit een tabel moet je dus aansturen wat de Model moet doen. Alle aansturingen en logica doe je dus in jouw controller. Hieronder is er een voorbeeld hoe je auto's ophaalt:

Carcontroller.php
Stel dat dit een controller is die je hebt gemaakt:

```
namespace App\Http\Controllers;

class CarController extends Controller
  public function showCars(){

    return view('listcars');
  }
}
```

In deze functie wil je alle auto's ophalen, en de data in de view meenemen (daarom heet de functie ook showCars). Daarvoor moet je dus eerst aangeven welk Model je wil aansturen. Voor dit voorbeeld heb ik een `Car` model gemaakt. Om aan te geven welk Model je wil aansturen geef je eerst aan in de controller welk model je wil gebruiken met **use**:

```
use App\Models\Car;
```

Dit stukje code zet je tussen de namespace en class in:

```
namespace App\Http\Controllers;
use App\Models\Car;

class CarController extends Controller
  public function showCars(){

    return view('listcars');
  }
}
```

Vervolgens geef je in de functie aan met Eloquent wat je wil aansturen met de Car Model. In dit geval wil je dus ophalen met:

```
$allCars = Cars::all();
```

Dit stukje code zet je in de functie, voordat je de view laat zien zoals hieronder is weergegeven:

```
namespace App\Http\Controllers;
use App\Models\Car;

class CarController extends Controller
  public function showCars(){

  $allCars = Car::all();
```

```
    return view('listcars');
  }
}
```

Opgehaalde data meenemen in een view

Nu heb je alle auto's opgehaald met een model via Eloquent. Die heb je in de variabele `$allCars` gezet. Om de auto's op de view te laten zien, moet je de variabele meenemen waar je jouw view returnt. Dat is in dit geval `return view('listcars');`

De functie `view()` kan je twee parameters geven:

- Paramater 1: jouw view die je wil tonen.
- Parameter 2: data die je in de view wil meenemen (zoals in dit geval de auto's)

Voor parameter 2, kan je op verschillende manieren data (in dit geval de auto's) meenemen. Zo kan je bijvoorbeeld data meenemen hoe de variabelenaam (in dit geval $allCars) heet: `compact('allCars')`

Wat doet `compact()`?

Met compact() kan je een bestaande variabele met data in de controller meenemen naar de view en daarin je dezelfde variabele kan gebruiken.

Dit voeg je dus toe in de `return view()`:

```
namespace App\Http\Controllers;
use App\Models\Car;

class CarController extends Controller
  public function showCars(){

  $allCars = Car::all();
    return view('listcars', compact('allCars'));
  }
}
```

Je kan dus nu in jouw view `$allCars` gebruiken om de data op te halen en te gebruiken (door bijvoorbeeld te tonen)!

Meegenomen data in de view tonen

Met behulp van de compact() functie, kan je dus de variabele $allCars ook gebruiken in jouw view! Zoals je al eerder weet, kan je met Blade PHP code schrijven.

Stel: dit is de view van `listcars`:

```
@extends('reuse')
@section('content')
<div class="container">
      <h1> Mijn Auto's </h1>
      <div class="row">
      <div class="col-4">
          <h3>Porsche 911</h3>
          <p>Supervette auto!</p>
        </div>
      <div class="col-4">
          <h3>Volkswagen Polo</h3>
          <p>Goede prijs-kwaliteit auto!</p>
        </div>
      </div>
</div>
@endsection
```

Zoals je ziet zijn er twee auto's hard-coded gemaakt. Dit gaan we dus dynamisch maken door de data te laten zien uit de database met behulp van Blade Templates:

Deze code:

```
<div class="col-4">
  <h3>Porsche 911</h3>
  <p>Supervette auto!</p>
</div>
<div class="col-4">
  <h3>Volkswagen Polo</h3>
  <p>Goede prijs-kwaliteit auto!</p>
</div>
```

Zetten we nu om naar:

```
@foreach($allCars as $car)
<div class="col-4">
  <h3>{{$car->title}}</h3>
```

```
  <p>{{$car->description}}</p>
</div>
@endforeach()
```

Wat staat er in de code?

- `@foreach` / `@endforeach`: zorgt ervoor dat je jouw array (in dit geval `$allCars`) geloopt wordt. Elke auto kan je aanroepen met `$car`. De variabele kan je zelf noemen.
  - In de loop, loop je ook een `<div class="col-4">`. Dus er passen maximaal 3 kolommen met autogegevens per rij (er passen 3x col-4 per rij).
    - In de kolommen heb je de data die je toont van `$car`. Je kan een specifieke data tonen met behulp van de kolomnaam uit de Model zoals `$car->title` en `$car->description`

Opdracht: data ophalen met Eloquent en op een view laten zien (10p)

Om data van een model (in dit geval jouw **Product** model) op te halen moet je eerst jouw model in jouw controller toevoegen:

Stappenplan:

1. Open jouw `ProductController.php` file.
2. Om jouw model toe te voegen, voeg je deze code toe: `use App\Models\Modelnaam;`. Jouw *Modelnaam* is dus de naam van jouw Model en dat is `Product`
3. Zoek in jouw `ProductController` naar de functie die jouw `view` returnt naar de productpagina.
   1. Maak in de functie code om gegevens op te halen via Product Model met Eloquent en stop die in een variabele.
   2. Maak gebruik met compact() om data mee te nemen in de view
4. Ga in de aangewezen view een toon daar jouw gegevens uit de database met behulp van Blade Templates

Wat lever je in?

- Jouw ProductController.php file
- Jouw Productpagina view

Revision #1
Created 2022-10-27 09:15:55 UTC by Max

Updated 2022-10-27 09:15:55 UTC by Max