Read

Inleiding

We hebben een database aangemaakt.

We hebben via een migration een tabel aangemaakt.

En we hebben een model aangemaakt zodat Laravel 'weet' waar de data staat en hoe de tabel heet.

We gaan nu de Read functie maken. We gaan daarvoor de **controller**, **view** en de **routing** opzetten.

Controller

Als de gebruiker van de webapplicatie de applicatie gebruikt dan gaat hij altijd eerst naar de controller. De controller gebruikt dan het model om de database te raadplegen en de output wordt view de view aan de gebruiker getoond. Dit is de MVC-structuur waarmee Laravel een applicatie opbouwt.

Met het volgende commando maken we een controller.

```
php artisan make:controller StockController --resource
```

Dit commando maakt een file app/Http/Conttrollers/StockController.php

image-1667035179108.png

Open deze file en plaats in de index functie de volgende code

```
public function index()
{
    $stocks = Stock::all();

    return view('stocks.index', compact('stocks')); // ->
resources/views/stocks/index.blade.php
}
```

Regel 3 haalt alle records uit de database en regel 5 opent de view (die we nog moeten maken).

Voeg bovenaan in de controller (op regel 6 bijvoorbeeld) de volgende regel code toe.

```
use App\Models\Stock;
```

Hiermee 'kent' de controller het model van stock.

View

In Laravel staan alle views in resources/view en alle view files hebben de file extensie blade.php

Waarom precies wordt later uitgelegd.

We maken een nieuwe file in resources/view/stocks/stocks en noemen die file index.blade.php. De folder stocks moeten we ook zelf aanmaken!

We plaatsen de volgende code in resources/view/stocks/index.blade.php

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/css/bootstrap.min.css"
integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
crossorigin="anonymous">
    <title>Stocks</title>
</head>
<body>
  <div class="container" style="margin:40px;">
    <table class="table">
      <thead class="thead-light">
          <tr>
            <th>ID</th>
            <th>Stock Name</th>
            <th>Stock Ticket</th>
            <th>Stock Value</th>
            <th>Updated at</th>
          </tr>
      </thead>
```

```
      <tbody>
          @foreach($stocks as $stock)
          <tr>
              <td>{{$stock->id}}</td>
              <td>{{$stock->stock_name}} </td>
              <td>{{$stock->ticket}}</td>
              <td>{{$stock->value}}</td>
              <td>{{$stock->updated_at}}</td>
          </tr>
          @endforeach
      </tbody>
    </table>
  </div>

</body>
</html>
```

Op regel 7 laden we Bootstrap (CSS framework) en gebruiken dat voor de vormgeving. We gaan daar in deze lessen verder niet op in.

De rest is standaard HTML code waarbij de resultaten in een tabel worden afgedrukt.

De *for-each* loop die we gebruiken (regel 23 en 31) ziet er iets anders uit als we gewend zijn in PHP en in Yii.

Dit is de Laravel-manier en hoort bij *blade* templates. Hierover later meer.

Routing

De laatste stap is de routing. In Yii ging dit volgens een vast patroon. In Laravel moeten we de routing zelf coderen.

De routing staat in de file routes/web.php

```
Route::get('stocks', [App\Http\Controllers\StockController::class, 'index']);
```

Dit koppelt de url */stock* aan de functie *index()* in de *StockController* class.

Testen

Om goed te kunnen testen moeten we de database vullen met wat data. Dat kun je doen door met de onderstaande query in phpmyadmin, 5 regels in de database toe te voegen.

```
INSERT INTO `stocks` (`id`, `stock_name`, `ticket`, `value`, `created_at`, `updated_at`)
VALUES
(1, 'Apple Inc.', 'AAPL', '1200.00', NULL, NULL),
(2, 'NVIDIA Corporation', 'NVDA', '830.00', NULL, NULL),
(3, 'Tesla Inc', 'TSLA', '1950.00', NULL, NULL),
(4, 'Netflix Inc', 'NFLX', '300.00', NULL, NULL),
(5, 'Amazon.com, Inc.', 'AMZN', '1640.00', NULL, NULL);
```

Zorg dat de Laravel ontwikkel-omgeving draait (php artisan serve) en ga naar localhost:8000/stocks. Als het goed is dan zie je het volgende overzicht:

image-1667044390059.png

De laatste kolom *Updated at* is (nog) leeg omdat de records via een query zijn aangemaakt. Als we straks via de browser records aanmaken zul je zien dat deze kolom automatisch wordt gevuld.

Revision #2
Created 2022-10-29 15:33:16 UTC by Max
Updated 2022-10-29 15:36:18 UTC by Max