

# Update

## Inleiding

Voor de update functie gaan we een knopje maken in ons overzicht.

## Index View

We beginnen met het aanpassen van de index view. Plaats deze code als extra kolom in de tabel en vergeet niet ook een extra `<th></th>` te plaatsen. Het aantal kolommen in de tabel moet in alle regels gelijk zijn!

```
<td><a href="/stocks/edit/id={{ $stock->id }}" class="btn btn-primary">Edit</a></td>
```

## Edit Route

We voegen een route toe in de web.php

```
Route::get('stocks/edit/{id}', [App\Http\Controllers\StockController::class, 'edit']);
```

Zodra we op de knop edit drukken in het overzicht (in de index.blade.php view) dan wordt de edit() function in deStockController aangeroepen. Het \$id wordt bovendien meegegeven.

## Controller edit()

In de controller zetten we de volgende code in de edit() function.

```
public function edit($id)
{
    $stock = Stock::find($id);
    return view('stocks.edit',['stock'=>$stock]); // -> resources/views/stocks/edit.blade.php
}
```

De edit() functie in de controller haalt het record met het id \$is op en het record, \$stock wordt aan de view edit.blade.php in de stocks folder meegegeven.

# Edit View

We plaatsen de volgende code in de `edit.blade.php` file in de stocks view. Deze file moeten we aanmaken.

```
<!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;">

    <h1 class="display-3">Stocks</h1>

    <form method="post" action="/stocks/update/{{ $stock->id }}">
      @csrf
      <div class="form-group">

        <label for="stock_name">Stock Name:*</label>
        <input type="text" class="form-control" name="stock_name" value="{{ $stock->stock_name }}" />
      </div>

      <div class="form-group">
        <label for="ticket">Stock Ticket:*</label>
        <input type="text" class="form-control" name="ticket" value="{{ $stock->ticket }}" />
      </div>

      <div class="form-group">
        <label for="value">Stock Value:</label>
        <input type="text" class="form-control" name="value" value="{{ $stock->value }}" />
      </div>
    </form>
  </div>
</body>
</html>
```

```

        </div>

        <button type="submit" class="btn btn-primary">Update</button>

    </form>

</div>

</body>
</html>

```

Test de code en als het goed is, verschijnt het edit form als er op de editknop wordt gedrukt.

[image1667129791204.png](#)

Wanneer de data wordt aangepast in dit edit form dan wordt er een post gedaan `/stocks/update/{{$stock-id}}`. Ben je bijvoorbeeld record met id 1 aan het aanpassen dan wordt het formulier gepost naar `/stocks/update/1`.

Hiervoor moeten we nu een route aanmaken.

## Update Route

We voegen de volgende route in `web.php`.

```
Route::post('stocks/update/{id}', [App\Http\Controllers\StockController::class, 'update']);
```

Nu moeten we de `update()` functie in de `StockController` aanpassen zodat het aangepaste record naar de database wordt weggeschreven.

## Controller update()

plaats de volgende code in de `update()` functie in de `StockController`.

```

public function update(Request $request, $id)
{
    // Validation for required fields (and using some regex to validate our numeric value)
    $request->validate([
        'stock_name'=>'required',
        'ticket'=>'required',
        'value'=>'required|max:10|regex:/^-?[0-9]+(?:\.[0-9]{1,2})?$/
    ]);

    $stock = Stock::find($id);

```

```
// Getting values from the blade template form
$stock->stock_name = $request->get('stock_name');
$stock->ticket = $request->get('ticket');
$stock->value = $request->get('value');
$stock->save();

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

In het eerste gedeelte regel 4 t/m 8 worden de velden gevalideerd. Dit is hetzelfde als bij de *create*.

Daarna wordt op regel 9 t/m 14 het juiste record uit de database geladen en worden de nieuwe waarden weggeschreven.

Op regel 16 wordt naar de index view gesprongen. Dit is het standaard overzicht.

## Samengevat

We beginnen dus met een update vanuit de index view.

Dan gaan we via de route `stocks/edit/{id}` naar de `edit()` functie in de `StockController`.

De `StockController` haalt de data op en laat deze data via de edit view zien.

Vanuit de edit view worden nieuwe waarden via een post naar de route `stocks/update/{id}` gestuurd.

De route `stocks/update/{id}` verwijst naar de `update()` functie in de `StockController`.

Deze functie voert de update uit en gaat daarna (terug) naar de index view.

Of in het kort, ziet het rondje er als volgt uit.

```
index view -> (route) stocks/edit/{id} -> edit() functie in StockController -> (view)
update -> (route) stocks/update/{id} -> update() functie in StockController -> index
view
```

--