CRUD-toiminnallisuus RESTful-reitein

Laravelin *RESTful Resource controller* asettaa reittitiedoston määrityksellä

```
Route::resource('customers', 'CustomerController');
```

muutamia vakioreittejä samalla nimeämällä ne. Tässä luvussa ei käytetä tuota toiminnallisuutta sellaisenaan vaan reitit ja niiden toiminnot kirjoitetaan itse reittitiedostoon näkyviin. Esityksessä kuitenkin noudatetaan samaa reittien ja käsittelymetodien nimeämiskäytäntöä. Oheisena nämä reitit ovat kirjoitettuna. Huomaat varmaan, että rivien 1-6 toiminnot toteutettiin sellaisenaan jo edellisessä luvussa

```
1  Route::get('/customers', 'CustomerController@index');
2
3  Route::get('/customers/create', 'CustomerController@create');
4  Route::post('/customers', 'CustomerController@store');
5
6  Route::get('/customers/{id}', 'CustomerController@show');
7
8  Route::get('/customers/{id}/edit', 'CustomerController@edit')
9  Route::patch('/customers/{id}', 'CustomerController@update');
10
11 Route::delete('/customers/{id}', 'CustomerController@destroy'
```

Toiminnot riveittäin ovat

- Rivi 1: Listataan kaikki asiakkaat
- Rivi 3: Näytetään lisäyslomake asiakkaan lisäämiseksi
- Rivi 4: Tallennetaan lomakkeelle kirjoitetut tiedot tietokantaan
- Rivi 6: Näytetään yksittäisen asiakkaan tiedot {id}-numeron perusteella
- Rivi 8: Näytetään asiakkaan {id} tiedot muokkauslomakkeella
- Rivi 9: Päivitetään muokkauslomakkeelle muutetut tiedot tietokantaan
- Rivi 11: Poistetaan asiakkaan {id} tiedot tietokannasta.

Muokkauslomakkeen näyttämistä varten lisätään reittitiedostoon reitti

routes/web.php

```
Route::get('/customers/{id}/edit', 'CustomerController@ed
```

ja kontrolleriin `edit()`-metodi

```
public function edit($id) {
   $customer = Customer::find($id);
   return view('customers/edit')->with('customer', $customer)
}
```

Lomake `resources/views/customers/edit.blade.php` itsessään on pieni muunnos tyhjästä lomakkeesta oheisella tavalla. Huomaa seuraavat asiat

- lähetysmetodi on `POST`, mutta Laravelin helper-metodi `method_field('PATCH')` generoi lomakkeeseen piilokentän `<input type="hidden" name="_method" value="PATCH">`, jonka perusteella Laravel tietää etsiä `PATCH`-metodia käytettäväksi. Tämä tehdään siksi, koska web-selaimia ei ole speksattu lähettämään `PATCH`-pyyntöjä.
- action-määritteessä käytetään `url()`-helper-metodia oikean URL-polun muodostamiseen. `url('/')` generoi URL-polun http://192.168.1.126/~testi/projekti01/public. Näin vältytään ../../-viittauksien käyttämiseltä

resources/views/customers/edit.blade.php

```
<!DOCTYPE html>
<html>

  <head>
    <title>Edit Customer</title>
  </head>

  <body>
    <h1>Edit Customer</h1>

    <form method="POST" action="{{ url('/') }}/customers/{{ $

      {{ method_field('PATCH') }}
```

```
14        {{ csrf_field() }}
15
16        <div>
17          <input type="text" name="name" value="{{ $customer->n
18        </div>
19
20        <div>
21          <input type="text" name="birth_date" value="{{ $custo
22        </div>
23
24         <div>
25           <button type="submit">Save changes</button>
26         </div>
27
28       </form>
29
30    </body>
31
32 </html>
```

30.02 Tietueen päivittäminen

Varsinainen päivitys tapahtuu määrittelemällä aluksi vaadittu "*patch-reitti*" `routes/web.php`-tiedostoon

```
Route::patch('/customers/{id}', 'CustomerController@updat
```

ja lisäämällä kontrolleriin `update()`-metodi

```
1  public function update($id) {
2
3      $customer = Customer::find($id);
4
5      $customer->name = request('name');
6      $customer->birth_date = request('birth_date');
7
8      $customer->save();
```

```
9       return redirect('/customers');
10  }
11
```

Testaus: Lataa haluttu asiakastietue muokkauslomakkeelle osoitteella http://192.168.1.126/~testi/projekti01/public/customers/1/edit ja tallenna tekemäsi muutokset.

30.03 Tietueen poistaminen

Luodaan poistamista varten reitti

```
Route::delete('/customers/{id}', 'CustomerController@dest
```

ja lisätään muokkauslomakkeen näkymään **erillinen toinenkin** HTML-lomake poistamista varten. Huomaa, että poistossa

- käytetään POST-metodia, mutta piilokentällä (tässä lyhennetty merkintä `@method('DELETE')`) Laraveliä ohjeistetaan käyttämään reittitiedostoon merkittyä DELETE-metodia.
- CSRF-token merkitään myös lyhennettynä `@csrf`

resources/views/customers/edit.blade.php

```
1   ...
2       <form method="POST" action="{{ url('/') }}/customers/{{ $
3
4         @method('DELETE')
5         @csrf
6
7          <div>
8            <button type="submit">Delete Customer</button>
9          </div>
10
11       </form>
12  ...
```

Lopuksi tarvitaan vielä kontrolleriin varsinainen `destroy()`-metodi:

```php
public function destroy($id) {
    Customer::find($id)->delete();
    return redirect('/customers');
}
```

Testaus: Lataa haluttu asiakastietue muokkauslomakkeelle osoitteella http://192.168.1.126/~testi/projekti01/public/customers/1/edit ja poista se.