

Laravel::DB::ScoreTronic02 - autorisoidut reitit ja omat SQL-kyselyt

Tässä luvussa on tarkoituksena

- syventää käsitystä Laravelin autentikointi- ja autorisointimekanismien toiminnasta erityisesti reittien autorisointiin liittyen
- näyttää kuinka aiemmin hankittua SQL-osaamista voi hyödyntää sellaisenaan ilman Laravelin Query Builderiä

Näkymä harjoitusten listaamiseksi

37.01 Luodaan reitti

routes/web.php

```
1 | ...
2 | Route::get('harkat', 'HarkkaController@list_all');
```

37.02 Luodaan HarkkaController

```
php artisan make:controller HarkkaController
```

ja muokataan luotu tiedosto muotoon

app/Http/Controllers/HarkkaController.php

```
1 | <?php
2 |
3 | namespace App\Http\Controllers;
4 | use Illuminate\Support\Facades\DB;
5 |
6 | //use App\Harkka;
7 |
8 | use Illuminate\Http\Request;
9 |
```

```

10 class HarkkaController extends Controller
11 {
12
13     public function list_all()
14     {
15         $PDO = DB::connection('mysql')->getPdo();
16
17         $sql = <<< SQLEND
18         SELECT *
19         FROM hsarja
20 SQLEND;
21
22         $allsql = $PDO->prepare($sql);
23
24         $allsql->execute();
25
26         // Muista TÄMÄ FETCH_OBJ
27         $harkat = $allsql->fetchAll((\PDO::FETCH_OBJ));
28
29         return view('harkat')->with('harkat', $harkat);
30
31     }
32
33     /*
34     public function list_all2()
35     {
36         $harkat = Harkka::all();
37         return view('harkat')->with('harkat', $harkat);
38     }
39     */
40
41 }

```

Huomaa kommentoidut rivit **Harkka**-mallin käyttämiseksi. Nyt **EI** siis ole luotu mallia `Harkka.php` ja tämä toimii silti. Huomaa myös tulosjoukon hakemisessa `FETCH_OBJ`, jotta voit näkymässä selata tulosjoukkoa!

37.03 Luodaan näkymä harkkojen listaamiseksi

Luodaan seuraava tiedosto

```
1  @extends('layouts.app')
2
3  @section('content')
4      <div class="container">
5          <div class="row">
6
7              <div class="col-md-4 col-md-offset-4">
8
9
10
11                 <div class="panel panel-default">
12                     <div class="panel-heading">Ilmoita harjoituspist
13
14                     <div class="panel-body">
15 <table class="table table-striped">
16     <thead>
17         <tr>
18             <th scope="col">Valitse</th>
19             <th scope="col">Viimeinen palautuspvm</th>
20         </tr>
21     </thead>
22     <tbody>
23
24     @foreach ($harkat as $harkka)
25         <tr class="bg-success">
26             <td><a href="{{ url('/addscoreform') }}"?id={{ $harkka->nrc
27             <td>{{ $harkka->palautuspvm }}</td>
28         </tr>
29     @endforeach
30
31     </tbody>
32 </table>
33
34             </div>
35         </div>
36     </div>
37 </div>
38 </div>
39 @endsection
```

Tämän pitäisi olla toimiva tällaisenaan, koska jo edellisessä kohdassa tälle lisättiin pudotusvalikkoon hyperlinkki. Listauksen näkymä on oheinen

1/public/harkat	
Tronic	/home Demotoiminnot ▾
Ilmoita harjoituspisteet	
Valitse	Viimeinen palautuspvm
Harjoitukset 1	2020-11-11
Harjoitukset 2	2020-11-11

37.04 Harkat.php-malli

Tämä kohta **ei** vaadi toimenpiteitä, mutta voit halutessasi kokeilla seuraavaa.

Jos haluaisit käyttää `hsarja`-taulun listaamiseksi *HarkkaControllerissa* nähtyä (kommentoitua) normaalia muotoa

```
$harkat = Harkka::all();
```

niin tällöin tulisi luoda `Harkka.php`-malli ja sanoa sen käyttävän `hsarja`-taulua. Tällöinkään ei tarvita `php artisan migrate -loitsuja` taulun sisällön listaamiseksi. Huomaa myös kommentoidut kohdat.

Seuraavaa ei siis ole pakollista tehdä, mutta voit kokeilla asiaa silti

app/Harkka.php

```

1  <?php
2
3  namespace App;
4
5  use Illuminate\Database\Eloquent\Model;
6
7  class Harkka extends Model
8  {
9      protected $table = 'hsarja';
10     // id-nimiseltä kentältä voidaan ottaa autoincrement-ominaisuus
11     // pois päältä seuraavasti:
```

```
12 | //public $incrementing = false;
13 | }
```

My Scoreboard - opiskelijan tulostkortti

Luodaan toiminnallisuus, jolla kirjautunut opiskelija voi listata palauttamansa tehtävät ja niistä merkitsemänsä ja hyväksytyt pisteet.

37.05 Luodaan reitti

Luodaan reitti pscores (~ personal scores)

routes/web.php

```
Route::get('/pscores', 'KyselyController@list_personal_scores');
```

37.06 Luodaan KyselyController

Luodaan erillinen KyselyController, jonne sijoitetaan tässä ja jatkossa esitettäviä sekalaisia SQL-kyselyjä.

```
php artisan make:controller KyselyController
```

muokataan tiedosto oheiseen muotoon. Huomaa, että

- kirjautunut käyttäjä haetaan `$student_id = Auth::id();`
- voidaan suorittaa useita SQL-kyselyjä ja eri tulosjoukkot voidaan välittää näkymälle useilla peräkkäisillä `->with->with->with`-viittauksilla
- SQL-kyselyjen perustarkoitus on kommentoituna ohjelmakoodissa. Tässä ei ole pakollista ymmärtää jokaista SQL-kyselyn nyanssia
- kun käytetään omia SQL-kyselyjä PDO-objekteina ilman Laravelin Query Builderiä on myös käytettävä preparing statement -menetelmää SQL-injektoiden ehkäisemiseen

app/Http/Controllers/KyselyController.php

```
1 | <?php
2 |
3 | namespace App\Http\Controllers;
4 | use Illuminate\Support\Facades\DB;
5 | use Illuminate\Http\Request;
```

```

6 use Illuminate\Support\Facades\Auth;
7
8 class KyselyController extends Controller
9 {
10     public function list_personal_scores(Request $request)
11     {
12
13         $student_id = Auth::id();
14
15         $PDO = DB::connection('mysql')->getPdo();
16
17         // $allscorelist
18         // Haetaan lista kaikista kirjautuneen opiskelijan tehtäväpä
19         // ehdotettuine, hyväksyttyine ja maksimipisteineen
20         $sql = <<< SQLEND
21 SELECT users.id, users.name, users.email,
22         tpalautus.hnro, tpalautus.tnro, tpalautus.palautusaika
23         tspec.maxpist,
24         hsarja.palautuspvm, hsarja.kuvaus
25 FROM users
26 INNER JOIN tpalautus
27     ON users.id = tpalautus.student_id
28 INNER JOIN tspec
29     ON tspec.hnro = tpalautus.hnro AND tspec.tnro = tpalautus.
30 INNER JOIN hsarja
31     ON hsarja.nro = tspec.hnro
32     WHERE users.id = :student_id
33     ORDER BY users.id, tpalautus.hnro, tpalautus.tnro
34 SQLEND;
35
36
37     $allscoresql = $PDO->prepare($sql);
38     $allscoresql->bindParam(':student_id', $student_id, \PDO
39
40     $allscoresql->execute();
41     $allscorelist = $allscoresql->fetchAll((\PDO::FETCH_OBJ)); /
42
43     // $sumscorelist
44     // lasketaan ehdotettujen, hyväksyttyjen ja maksimipisteiden
45     // opintojakson harjoitustehtäväärvosanan laskemista varten
46     $sql = <<< SQLEND
47 SELECT sum(tpalautus.initpoints) as ipoints, sum(tpalautus.t
48         sum(tspec.maxpist) as mpoints
49     FROM tspec

```

```

50     INNER JOIN tpalautus
51         ON tspec.hnro = tpalautus.hnro AND tspec.tnro = tpalautus.tnro
52         WHERE tpalautus.student_id = :student_id AND
53             tpalautus.finalpoints IS NOT NULL;
54 SQLEND;
55
56 $sqlresult = $PDO->prepare($sql);
57 $sqlresult->bindParam(':student_id', $student_id, \PDO::PARAM_INT);
58
59
60 $sqlresult->execute();
61 $sumscorelist = $sqlresult->fetchAll(\PDO::FETCH_OBJ); //
62
63
64 // $totalsumscorelist
65 // lasketaan kaikkien tarjolla olevien tehtävien maksimipisteet
66 // opintojakson harjoitustehtäväärvosanan laskemista varten
67 $sql = <<< SQLEND
68 SELECT sum(maxpist) as maxpoints
69     FROM tspec
70 SQLEND;
71 $sqlresult = $PDO->prepare($sql);
72 $sqlresult->execute();
73 $totalsumscorelist = $sqlresult->fetchAll(\PDO::FETCH_OBJ);
74
75 // $pendingsumscorelist
76 // lasketaan ehdotettujen, ei-vielä-hyväksyttyjen ja maksimipisteiden
77 // opintojakson
78 $sql = <<< SQLEND
79 SELECT sum(tpalautus.initpoints) as ipoints, sum(tpalautus.finalpoints) as fpoints,
80         sum(tspec.maxpist) as mpoints
81     FROM tspec
82     INNER JOIN tpalautus
83         ON tspec.hnro = tpalautus.hnro AND tspec.tnro = tpalautus.tnro
84         WHERE tpalautus.student_id = :student_id AND
85             tpalautus.finalpoints IS NULL;
86 SQLEND;
87 $sqlresult = $PDO->prepare($sql);
88 $sqlresult->bindParam(':student_id', $student_id, \PDO::PARAM_INT);
89 $sqlresult->execute();
90 $pendingsumscorelist = $sqlresult->fetchAll(\PDO::FETCH_OBJ);
91
92
93

```

```

94
95         return view('allscorelist')->with('allscorelist', $allsc
96
97         // return $allscorelist;
98
99     }
100 } // class

```

37.07 Luodaan Scoreboard-näkymä

Luodaan allscorelist.blade.php-näkymä tarjolla olevien harjoitusten listaamiseksi.

resources/views/allscorelist.blade.php

Pitkää monipolvista näkymää ei selitellä enempää, huomaa kuitenkin

- `Auth::user()` -käyttö
- kontrollerista useilla peräkkäisillä `->with->with->with`-viittauksilla välitettyjen taulukoiden läpikäynti
- näkymässä on paljon monenlaista yksityiskohtia pisteiden esittämiseen. Tässä ei ole pakollista ymmärtää pisteiden esittämiseen liittyviä nyansseja

```

1  @extends('layouts.app')
2
3  @section('content')
4  <div class="container">
5      <div class="row">
6
7          <div class="col-md-8 col-md-offset-2">
8
9
10
11             <div class="panel panel-default">
12                 <div class="panel-heading">
13                     <h3>My ScoreBoard</h3>
14                 <blockquote>
15                     <b>
16                         @isset(Auth::user()->name)
17                         {{ Auth::user()->name }}
18                         @endisset
19                     </b>
20                     <br>
21                     Kokonaispisteet: {{ $sumscorelist[0]->fpoints + 0 }} / {{ $total

```



```

22
23 @if (floor((0.4 * ($totalsumscorelist[0]->maxpoints + 0)) - ($su
24     Opintojakson hyväksymiseen vaaditaan vielä: {{ floor((0.4 *
25 @else
26     Opintojaksosi on hyväksytty! Lisäpisteet vaikuttavat arvosar
27 @endif
28
29 Harjoitustehtävääarvosana:  {{ number_format(($sumscorelist[0]->
30
31
32
33 </blockquote>
34
35
36 <table class="table table-striped">
37     <thead>
38         <tr>
39             <th scope="col"></th>
40             <th scope="col">Tilanne nyt</th>
41             <th scope="col">Kokonaistilanne</th>
42         </tr>
43     </thead>
44     <tbody>
45
46         <tr class="bg-success">
47             <th>Opettaja tarkistanut<span class="spnTooltip">Opettaja
48             <td>{{ $sumscorelist[0]->fpoints + 0 }} / {{ $sumscorelist
49             <td>{{ $sumscorelist[0]->fpoints + 0 }} / {{ $totalsumscor
50         </tr>
51
52         <tr class="bg-success">
53             <th>Kaikki palautetut</th>
54             <td>{{ number_format($pendingsumscorelist[0]->ipoints + $
55             <td>{{ $pendingsumscorelist[0]->mpoints + $sumscorelist[0]
56         </tr>
57
58         <tr class="bg-success">
59             <th>Arviointia odottaa</th>
60             <td>{{ number_format($pendingsumscorelist[0]->ipoints + 0
61             <td>{{ $pendingsumscorelist[0]->mpoints + 0 }} / {{ $total
62         </tr>
63
64
65     </tbody>

```

[illegible]

```

110         </td>
111     @endif
112 @else
113     <td class="alert alert-danger" role="alert">{{ $sc->fin
114         <span class="spnTooltip">{{ $sc->kommentti }}</s
115     </td>
116 @endif
117 <td>{{ $sc->maxpist }}</td>
118
119
120
121 </tr>
122
123 @endforeach
124
125 </tbody>
126 </table>
127
128         </div>
129     </div>
130 </div>
131 </div>
132 </div>
133 @endsection

```

37.08 Kirjautumista vaativien reittien määrittely

HomeController.php-tiedoston konstruktorissa määriteltiin

```

1 public function __construct()
2 {
3     $this->middleware('auth');
4 }

```

jonka perusteella **kaikki** ko. kontrollerin tarjoamat toiminnot vaativat kirjautumista.

Reittitiedostoon `routes/web.php` voidaan määritellä myös reittitasolla kirjautumista vaativat toiminnot. Esim. muuttamalla **pscores**-reittiä seuraavasti henkilökohtaisen ScoreBoardin voi nähdä vain kirjautunut käyttäjä

Kommentoi ja muuta /pscores-reittimääitykset seuraavasti

`routes/web.php`

```

1  ...
2  //Route::get('/pscores', 'KyselyController@list_personal_scores'
3
4
5  Route::get('/pscores', [
6      'middleware' => 'auth',
7      'uses' => 'KyselyController@list_personal_scores'
8  ]);
9  ...

```

37.08 MyScoreBoard navigointivalikkoon

Lisää navigointivalikon pudotusvalikkoon resources/views/layouts/app.blade.php linkki **My ScoreBoard** -toiminnolle

resources/views/layouts/app.blade.php

```

1  ...
2  <!-- Left Side Of Navbar -->
3  <ul class="nav navbar-nav">
4
5  <li class="dropdown">
6      <a class="dropdown-toggle" data-toggle="dropdown" href="#"
7  <ul class="dropdown-menu" aria-labelledby="download">
8  <li><a href="{{ url('/users') }}">Opiskelijalista</a></li>
9  <li class="divider"></li>
10 <li><a href="{{ url('/harkat') }}">Harkkapalautukset</a></li>
11 <li class="divider"></li>
12 <li><a href="{{ url('/pscores') }}">My ScoreBoard</a></li>
13 <li class="divider"></li>
14 <li><a href="{{ url('/toiminto33') }}">Toiminto 33</a></li>
15
16 </ul>
17 </li>
18 </ul>
19 ...

```

37.09 Testaa

Kirjaudu käyttäjänä Ken Guru ja katso My ScoreBoard. Näkymän tulisi olla oheinen

My ScoreBoard

Guru Ken

Kokonaispisteet: 10 / 18 p.

Opintojaksosi on hyväksytty! Lisäpisteet vaikuttavat arvosanaan!

Harjoitustehtäväärvosana: 2.78

	Tilanne nyt	Kokonaistilanne
Opettaja tarkistanutOpettaja on tarkistanut ja hyväksynyt tämän verran pisteitä	10 / 18	10 / 18
Kaikki palautetut	18 / 18	18 / 18
Arviointia odottaa	0 / 0	0 / 18

Yksittäiset tehtävät

Nimi	Harj/Teht	URL	Ehdotettu	Kirjattu	Max
Guru Ken	H1 T1	Ratkaisu	2	2 ok	2
Guru Ken	H1 T2	Ratkaisu	4	4 ok	4
Guru Ken	H2 T1	Ratkaisu	4	4 ok	4
Guru Ken	H2 T2	Ratkaisu	8 ek	0 ek	8

Admin-käyttäjän kirjautumisen vaatiminen

37.10 Reittien määrittely

Määritellään reitit `users` ja `harkat` saataville vain sille käyttäjälle, jonka email on `admin@student.jamk.fi`. Huolehdi siten ko. reitit ovat määritelty kertaalleen vain tässä tätä käyttötarkoitusta varten eli poista tai kommentoi vanhat merkinnät

routes/web.php

```
1 ...
2 Route::group(['middleware' => ['auth', 'admin']], function() {
3     // your routes
4     Route::get('users', 'UserController@list_all');
```

```

5      Route::get('harkat', 'HarkkaController@list_all');
6  });
7  ...

```

Nyt määritellyt reitit ovat `auth`- ja `admin`-nimisten autorisointikäsittelijöiden vaatimusten mukaan saatavissa.

37.11 app/Http/Kernel.php

Määritellään `admin`-autorisointi middleware-käsittelijänä `app/Http/Kernel.php`-tiedostoon taulukkomuuttujan `$routeMiddleware` uudeksi alkioksi. Lisätään alla näkyvään koodinpätkään viimeinen `'admin'`-alkuinen rivi

app/Http/Kernel.php

```

1  ...
2      protected $routeMiddleware = [
3          'auth' => \App\Http\Middleware\Authenticate::class,
4          'auth.basic' => \Illuminate\Auth\Middleware\AuthenticateWithBasicAuth::class,
5          'bindings' => \Illuminate\Routing\Middleware\SubstituteBindings::class,
6          'cache.headers' => \Illuminate\Http\Middleware\SetCacheHeaders::class,
7          'can' => \Illuminate\Auth\Middleware\Authorize::class,
8          'guest' => \App\Http\Middleware\RedirectIfAuthenticated::class,
9          'signed' => \Illuminate\Routing\Middleware\ValidateSignature::class,
10         'throttle' => \Illuminate\Routing\Middleware\ThrottleRequests::class,
11         'verified' => \Illuminate\Auth\Middleware\EnsureEmailIsVerified::class,
12         'admin' => \App\Http\Middleware\AdminMiddleware::class,
13     ];
14  ...

```

Huomataan, että

- Päätimme itse tämän uuden autorisoinnin nimen `'admin'`
- Päätimme itse, että `admin`-autorisoinnista vastaa `AdminMiddleware`-niminen luokka. Tehdään seuraavana se.

37.12 AdminMiddleware

Luodaan `AdminMiddleware.php`-tiedosto käsittelemään pyyntöjä

```
php artisan make:middleware AdminMiddleware
```

Muokataan AdminMiddleware.php-tiedostoa siten, että vain admin@student.jamk.fi voi käyttää kohdassa 37.10 määriteltyjä reittejä. Muokkaa tiedoston sisällöksi seuraava

app/Http/Middleware/AdminMiddleware.php

```
1  <?php
2
3  namespace App\Http\Middleware;
4
5  use Closure;
6  use Illuminate\Contracts\Auth\Guard;
7
8  class AdminMiddleware
9  {
10     /**
11      * The Guard implementation.
12      *
13      * @var Guard
14      */
15     protected $auth;
16
17     /**
18      * Create a new filter instance.
19      *
20      * @param Guard $auth
21      * @return void
22      */
23     public function __construct(Guard $auth)
24     {
25         $this->auth = $auth;
26     }
27
28     /**
29      * Handle an incoming request.
30      *
31      * @param \Illuminate\Http\Request $request
32      * @param \Closure $next
33      * @return mixed
34      */
35     public function handle($request, Closure $next)
36     {
37         //dd($this->auth->getUser());
```

```
38         if ($this->auth->getUser()->email !== "admin@student.jamk.fi") {
39             return redirect('authproblem');
40             abort(403, 'Unauthorized action.');
```

```
41         }
42
43         return $next($request);
44     }
45 }
```

Huomaa vaatimukset

- ainoastaan käyttäjällä `admin@student.jamk.fi` on pääsy kohdassa 37.10 määritelyihin reitteihin. Osaat varmaan tämän perusteella luoda muitakin kriteereitä
- muut käyttäjät ohjataan reitille 'authproblem'. Tässä materiaalissa ei ole luotu sen reitin päähän mitään näkymää, mutta sinne olisi helppo luoda haluttu authproblem-sivu.

37.13 Testaus

Testaa, että määrittelemäsi reitit ovat vain `admin@student.jamk.fi`-käyttäjänä kirjautuneena saatavilla.