Laravel

Deze les

- Facades
- Mail
- Collections
- Middlewares
- Authentication

Facades

http://laravel.com/docs/5.1/facades

Een facade is een statische wrapper voor “externe” classes. Je heb het gebruik van facades waarschijnlijk al eerder gezien of je hebt er al mee gewerkt.

```php
// Een aantal geregistreerde facades in config/app.php (er zijn nog veel meer...)
'aliases' => [
    'Config'    => Illuminate\Support\Facades\Config::class,
    'DB'        => Illuminate\Support\Facades\DB::class,
    'Mail'      => Illuminate\Support\Facades\Mail::class,
    'Route'     => Illuminate\Support\Facades\Route::class,
    'Schema'    => Illuminate\Support\Facades\Schema::class,
    'Session'   => Illuminate\Support\Facades\Session::class,
    'Validator' => Illuminate\Support\Facades\Validator::class,
],

// Boven je class geef je aan welke facades je gebruikt in een class
// dit moet sowieso bij alle externe classes die je gebruikt in laravel(bijv. models)
use Config;
use DB;

// Vervolgens kun op een facade allerlei statische functies uitvoeren
Config::get('app.aliases');
DB::table('users')->get();
```

Mail

http://laravel.com/docs/5.1/mail

Laravel komt ook met een gemakkelijke mail wrapper. Deze wrapper is gebaseerd op het populaire SwitfMailer. Ook deze wrapper is te benaderen door middel van een facade. De email template die word gebruikt is een blade view en je kan dus ook op die manier data in je email verwerken. Zie de docs voor informatie over CC, BCC, attachments en meer

```php
use Mail;



$user = User::create(['email' => 'leon@noorderpoort.nl', 'name' => 'Leon']);

       Blade view      Data voor blade view        Standaard message object     Userdata

Mail::send('emails.register', ['user' => $user], function ($message) use ($user) {
    $message->from('noreply@noorderpoort.nl', 'Noreply');

    $message->to($user->email, $user->name)->subject('Registration');
});
```

Collections

1/collections

http://laravel.com/docs/5.

Een collection in laravel bied een gemakkelijke wrapper om met array's van data om te gaan. Een eloquent model returned automatisch een Collection(en de database builder vanaf laravel 5.3).

Een aantal voorbeelden van wat je met een collection kan doen:

```php
$locations = array(
      ['city' => 'groningen', 'population' => 800],
      ['city' => 'assen', 'population' => 1]
);
$collection = collect($locations);

$assen = $collection->where('city', 'Assen');

$collection = collect($locations)->map(function ($item) {
   $item['city'] = strtoupper($item['city']);
   return $item;
});

$avg_population = $collection->avg('population'));

dd($collection->toArray());
```

Middlewares

http://laravel.com/docs/5.1/middleware

Een middleware is een stukje code dat word uitgevoerd tussen je “request” en voor code dat in de controller staat. De te gebruiken middlewares defineer je in `app/Http/Kernel.php`. En zoals bij veel modules in laravel kun je middleware ook maken met een console commando.

```php
public function handle($request, Closure $next)
{
    Log::info('user visited from' . $request->ip());

    return $next($request);
}

Route::get('{category}/{post}', ['middleware' => 'log.ip', 'uses' => 'PostsController@routePost' ]);
```

Authentication

http://laravel.com/docs/5.1/authentication

Laravel komt standaard ook met een “module” om Authenticate(login) makkelijk voor je te maken. Standaard meegeleverd worden de Controllers en Migrations. Omdat we alles wat met laravel authentication te maken heeft al een keer besproken en de documentatie over dit onderdeel goed is uitgelegd laat ik het aan jullie over om hier zelf dieper op in te gaan.

http://laravel.com/docs/5.1/authentication

De opdracht is om de calculator die we in les 1 hebben gemaakt te beveiligen met een login zodat deze alleen beschikbaar is voor ingelogde gebruikers.