Laravel 5.7 par la pratique – L'authentification

Dans ce chapitre nous allons poursuivre le développement de l'application de galerie photos. On l'a laissée en chantier avec un layout adapté mais il faut maintenant qu'on s'occupe de l'apparence des formulaires de l'authentification pour les harmoniser avec l'ensemble.

Comme on va avoir des répétitions de code on va un peu mutualiser ça avec ce que nous offre Blade comme possibilités, en l'occurrence les inclusions de vues et les composants.

On va aussi créer un helper pour rendre actif l'item en cours du menu dans la barre de navigation. Nous ajouterons la vérification de l'email et on terminera en mettant les textes des emails de notification en français.

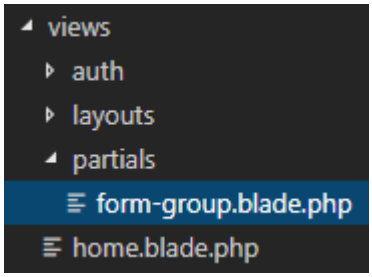
Les formulaires de Bootstrap 4

Si on regarde la documentation de Bootstrap 4 on a une page consacrée aux formulaires. On voit que chaque contrôle est équipé de ce genre de code :

```
<div class="form-group">
  <label for="exampleInputEmail1">Email address</label>
  <input type="email" class="form-control" id="exampleInputEmail1"
aria-describedby="emailHelp" placeholder="Enter email">
  <small id="emailHelp" class="form-text text-muted">We'll never
share your email with anyone else.</small>
</div>
```

On a aussi tout un passage concernant la validation.

Plutôt que de le reproduire plusieurs fois le même code on va créer une vue partielle à inclure :



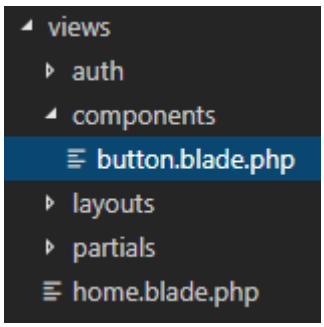
En prévoyant toutes les variables nécessaires avec la validation :

```
<div class="form-group">
    <label for="{{ $name }}">{{ $title }}</label>
     <input id="{{ $name }}" type="{{ $type }}" class="form-
control{{ $errors->has($name) ? ' is-invalid' : '' }}" name="{{
$name }}" value="{{ old($name, isset($value) ? $value : '') }}" {{
$required ? 'required' : ''}}>

    @if ($errors->has($name))
        <div class="invalid-feedback">
            {{ $errors->first($name) }}
        </div>
    @endif
</div>
```

Dans tous les formulaires on va aussi avoir un bouton de soumission. La syntaxe que nous donne la documentation est celle-ci :

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

Là on va créer un composant :



Avec ce code :

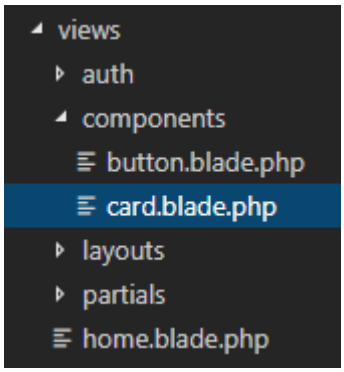
```
<button type="submit" class="btn @isset($color){{ ' btn-' . $color
}}@else btn-primary @endisset float-right">
    {{ $slot }}
```

```
</button>
```

Enfin pour faire plus joli on va inclure les formulaires dans un composant card de Bootstrap. On voit dans la documentation la syntaxe de ce composant :

```
<div class="card">
  <div class="card-header">
    Featured
  </div>
  <div class="card-body">
    <h4 class="card-title">Special title treatment</h4>
     <p class="card-text">With supporting text below as a natural lead-in to additional content.</p>
    <a href="#" class="btn btn-primary">Go somewhere</a>
  </div>
</div>
```

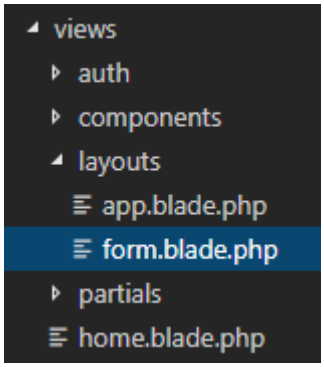
On va en faire un composant pour notre application :



Avec ce code :

```
<div class="card text-white bg-dark mb-3">
    <h4 class="card-header">
        {{ $title }}
    </h4>
    <div class="card-body">
        {{ $slot }}
    </div>
</div>
```

Pour compléter la panoplie on va aussi créer un layout pour tous les formulaires :




```
@extends('layouts.app')

@section('content')
    <div class="container py-5">
        <div class="row">
            <div class="col-md-6 offset-md-3">
                @yield('card')
            </div>
        </div>
    </div>
@endsection
```

Maintenant nous sommes bien équipés pour modifier les formulaires existant…

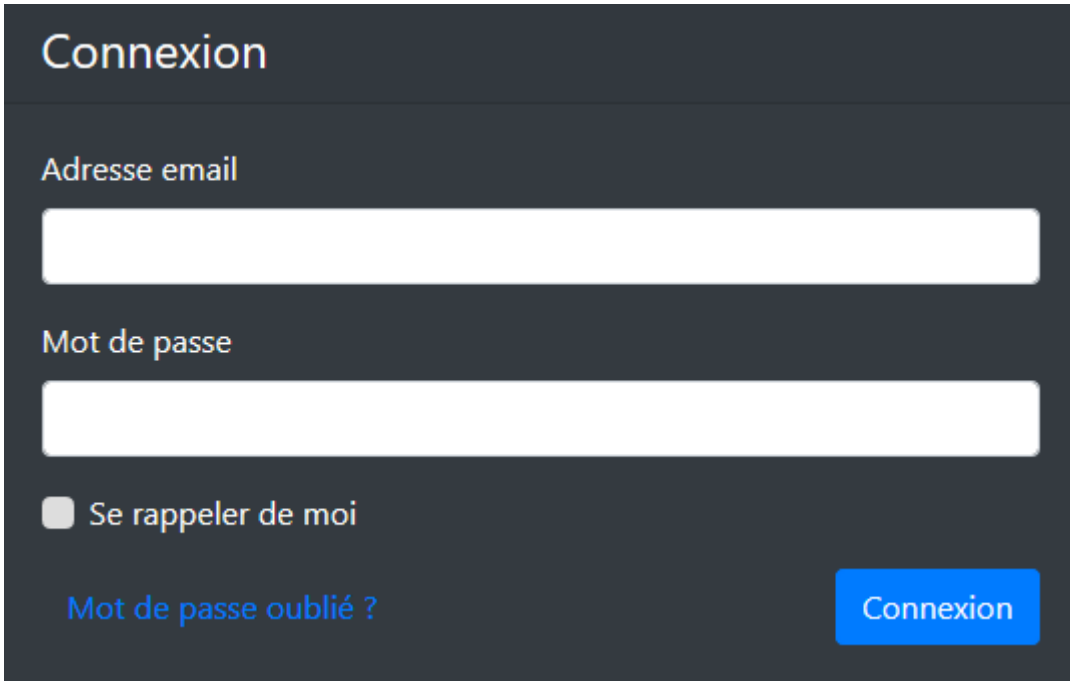
La connexion

Voici le nouveau code pour la vue de connexion (**auth/login**) :

```
@extends('layouts.form')

@section('card')

    @component('components.card')

        @slot('title')
            @lang('Connexion')
        @endslot

        <form method="POST" action="{{ route('login') }}">
            {{ csrf_field() }}

            @include('partials.form-group', [
```

```
                'title' => __('Adresse email'),
                'type' => 'email',
                'name' => 'email',
                'required' => true,
                ])

            @include('partials.form-group', [
                'title' => __('Mot de passe'),
                'type' => 'password',
                'name' => 'password',
                'required' => true,
                ])

            <div class="custom-control custom-checkbox">
                  <input type="checkbox" class="custom-control-
input" id="remember" name="remember" {{ old('remember') ?
'checked' : '' }}>
                      <label class="custom-control-label"
for="remember"> @lang('Se rappeler de moi')</label>
            </div>

            @component('components.button')
                @lang('Connexion')
            @endcomponent

                      <a class="btn btn-link" href="{{
route('password.request') }}">
                @lang('Mot de passe oublié ?')
            </a>

        </form>

    @endcomponent

@endsection
```

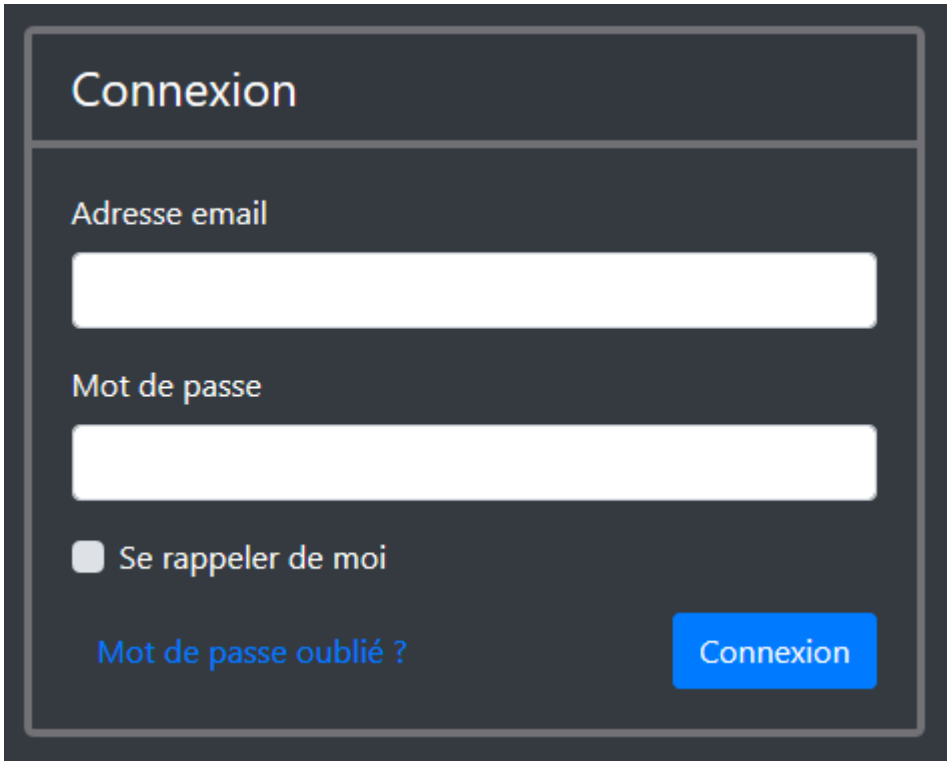
Ce qui donne cet aspect :



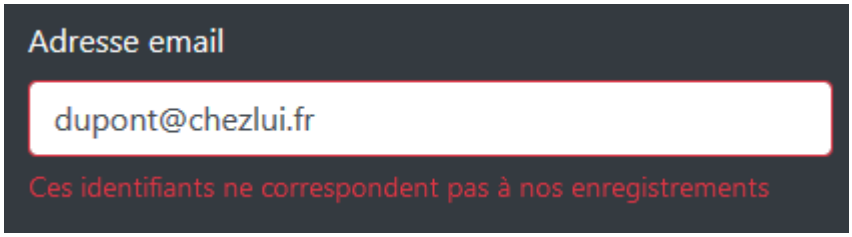
Pour bien délimiter le formulaire on va dessiner la bordure avec le CSS. Dans le fichier resources/sass/_variables on va modifier les variables concernées pour le composant. Au passage on va supprimer tout ce qui concerne la typographie et les couleurs que Laravel prévoit de base. Donc on va n'avoir plus que ça dans ce fichier :

```
// Body
$body-bg: #343a40;

// Card
$card-border-width: 4px;
$card-border-radius: .3rem;
$card-border-color: rgba(255, 242, 242, .3);
```

On régénère avec **npm run dev** et on obtient maintenant cet aspect :



On vérifie que la validation apparaît bien :

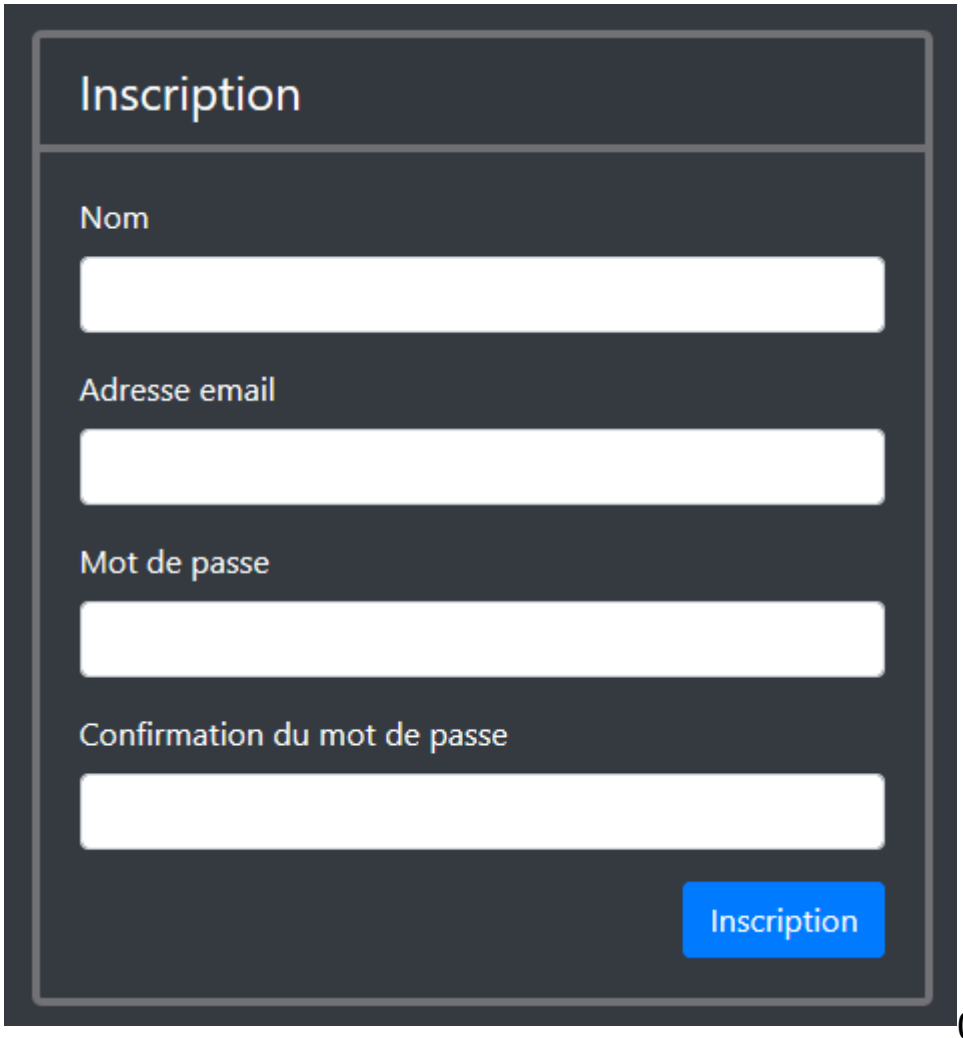


L’inscription

De la même manière on va construire la vue pour l’inscription (**auth/register**) :

```
@extends('layouts.form')

@section('card')

    @component('components.card')

        @slot('title')
            @lang('Inscription')
        @endslot

        <form method="POST" action="{{ route('register') }}">
            {{ csrf_field() }}
```

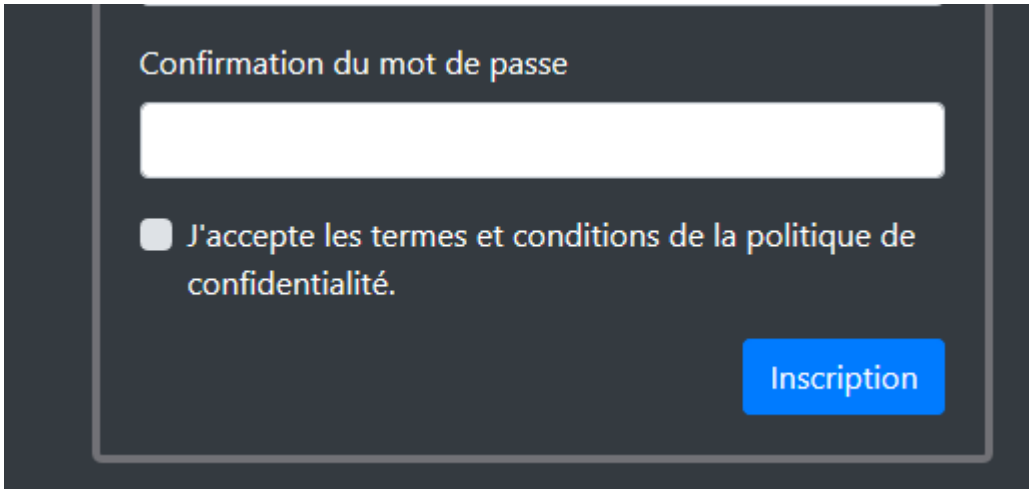
```
            @include('partials.form-group', [
                'title' => __('Nom'),
                'type' => 'text',
                'name' => 'name',
                'required' => true,
                ])

            @include('partials.form-group', [
                'title' => __('Adresse email'),
                'type' => 'email',
                'name' => 'email',
                'required' => true,
                ])

            @include('partials.form-group', [
                'title' => __('Mot de passe'),
                'type' => 'password',
                'name' => 'password',
                'required' => true,
                ])

            @include('partials.form-group', [
                'title' => __('Confirmation du mot de passe'),
                'type' => 'password',
                'name' => 'password_confirmation',
                'required' => true,
                ])

            @component('components.button')
                @lang('Inscription')
            @endcomponent

        </form>

    @endcomponent

@endsection
```



On va ajouter une case à cocher à ce formulaire pour s'assurer que les personnes qui s'inscrivent ont bien lu la politique de confidentialité. Vous avez sans doute remarqué que le RGPD est entré en vigueur et complique un peu la vie des développeurs (entre autres). Si vous n'êtes pas au courant je vous conseille d'aller consulter le site de la CNIL.

On va donc prévoir une case à cocher avant le bouton de confirmation en prévoyant **required** :

```
<div class="form-group">
    <div class="custom-control custom-checkbox">
          <input type="checkbox" class="custom-control-input" id="ok" name="ok" required>
            <label class="custom-control-label" for="ok">@lang('J\'accepte les termes et conditions de la politique de confidentialité.')</label>
    </div>
</div>
```



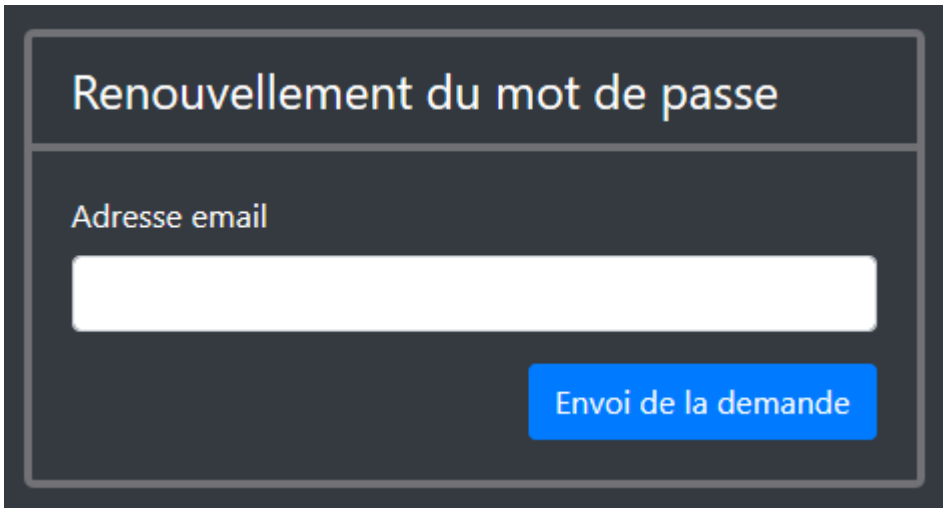
On ne va pas aller jusqu’à ajouter ça à la validation côté serveur et nous contenter de celle côté client.

Le renouvellement du mot de passe

On continue avec la vue pour la demande de renouvellement du mot de passe (**auth/passwords/email**) :

```
@extends('layouts.form')

@section('card')

    @if (session('status'))
        <div class="alert alert-success" role="alert">
            {{ session('status') }}
        </div>
    @endif

    @component('components.card')

        @slot('title')
            @lang('Renouvellement du mot de passe')
        @endslot

         <form method="POST" action="{{ route('password.email') }}">
            {{ csrf_field() }}
```

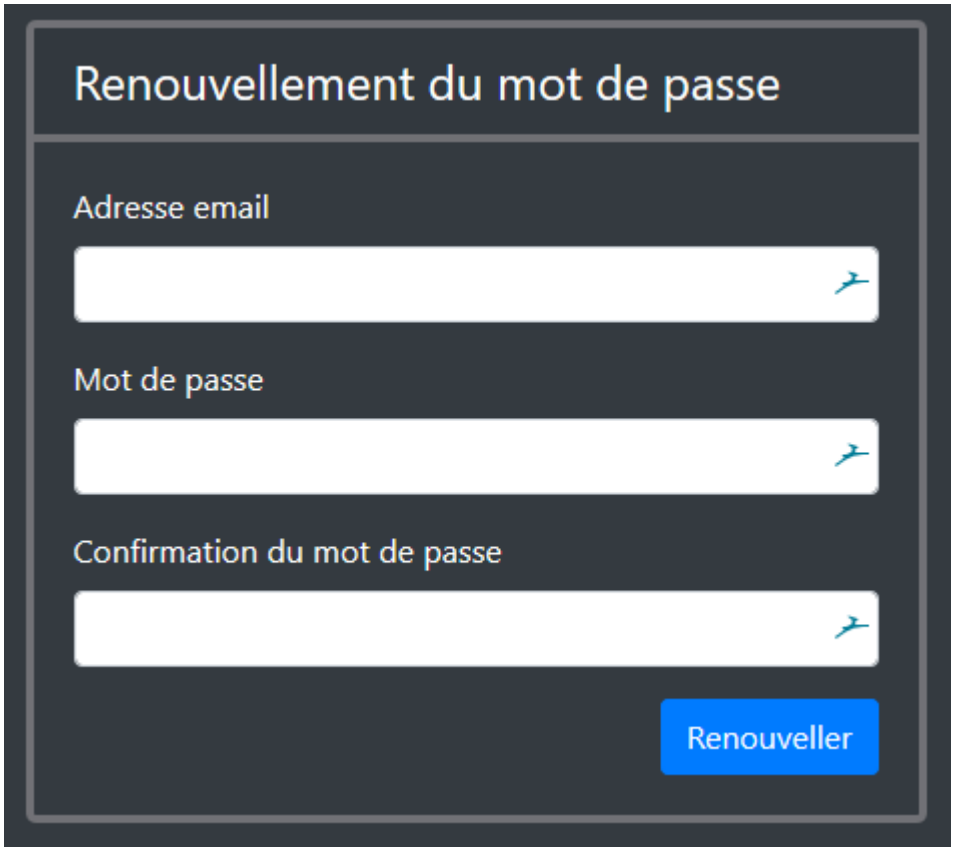
```
            @include('partials.form-group', [
                'title' => __('Adresse email'),
                'type' => 'email',
                'name' => 'email',
                'required' => true,
                ])

            @component('components.button')
                @lang('Envoi de la demande')
            @endcomponent
        </form>

    @endcomponent

@endsection
```



Et enfin la vue pour le renouvellement (**auth/passwords/reset**) :

```
@extends('layouts.form')

@section('card')

    @component('components.card')

        @slot('title')
            @lang('Renouvellement du mot de passe')
        @endslot

         <form method="POST" action="{{ route('password.request')
}}">
            {{ csrf_field() }}
```

```
            <input type="hidden" name="token" value="{{ $token
}}">

            @include('partials.form-group', [
                'title' => __('Adresse email'),
                'type' => 'email',
                'name' => 'email',
                'required' => true,
                ])

            @include('partials.form-group', [
                'title' => __('Mot de passe'),
                'type' => 'password',
                'name' => 'password',
                'required' => true,
                ])

            @include('partials.form-group', [
                'title' => __('Confirmation du mot de passe'),
                'type' => 'password',
                'name' => 'password_confirmation',
                'required' => true,
                ])

            @component('components.button')
                @lang('Renouveller')
            @endcomponent

        </form>

    @endcomponent

@endsection
```



Pour faire vos essais vous aurez besoin d'un gestionnaire d'emails efficace. Je vous conseille Mailtrap. Vous pouvez créer gratuitement un compte avec une boîte de réception, 50 messages et 2 messages par seconde. Vous récupérez vos identifiants et les reportez dans le **fichier .env**. Si vous avez des soucis en local avec l'url supprimez la valeur de la variable **APP_URL** toujours dans le fichier **.env**.

L'item actif dans le menu

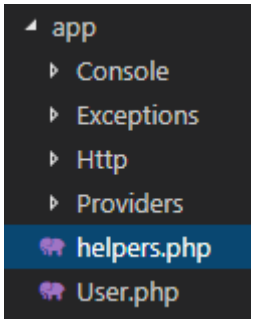
On a bien avancé mais il y a une chose pas très jolie au niveau de la barre de navigation : rien n'indique quel item est actif. On va maintenant arranger ça…

Quand on regarde la documentation de Bootstrap 4 on voit qu'il suffit d'ajouter la classe **active** à un item pour le distinguer visuellement :

```
<li class="nav-item active">
```

Il faut donc qu'on s'arrange pour ajouter cette classe en fonction de la requête en cours. Pour l'occasion on va créer un helper. On

commence par créer un fichier (**app/helpers.php**) :



Et on lui ajoute cette fonction :

```
<?php

if (!function_exists('currentRoute')) {
    function currentRoute(...$routes)
    {
        foreach($routes as $route) {
            if(request()->url() == $route) {
                return ' active';
            }
        }
    }
}
```

On prend toujours la précaution de vérifier que la fonction n'existe pas déjà. Pour que Laravel soit au courant que notre fonction existe on va ajouter la référence dans **composer.json** :

```
"autoload": {
    "classmap": [
        "database/seeds",
        "database/factories"
    ],
    "files": [
        "app/helpers.php"
    ],
    "psr-4": {
        "App\\": "app/"
    }
},
```

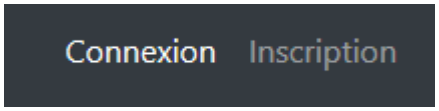
On relance composer :

```
composer dumpautoload
```

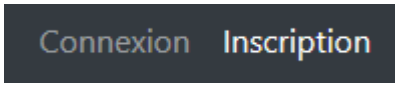
Maintenant il ne reste plus qu'à modifier le code de la barre de navigation dans la vue **layouts/app** :

```
<li class="nav-item{{ currentRoute(route('login')) }}"><a class="nav-link" href="{{ route('login') }}">@lang('Connexion')</a></li>
<li class="nav-item{{ currentRoute(route('register')) }}"><a class="nav-link" href="{{ route('register') }}">@lang('Inscription')</a></li>
```

Maintenant quand je vais sur la vue de connexion l'item correspondant s'illumine :



Et ça fonctionne aussi pour l'inscription :



Il n'y a que pour le renouvellement du mot de passe qu'on a rien à rendre actif…

La vérification de l'email

Avec la version 5.7 de Laravel on dispose de la possibilité d'ajouter facilement la vérification de l'email des personnes qui s'inscrivent. Tant que l'email n'est pas confirmé on interdit certaines routes.

Il faut que le modèle **App\User** implémente **Illuminate\Contracts\Auth\MustVerifyEmail** :

```
class User extends Authenticatable implements MustVerifyEmail
```

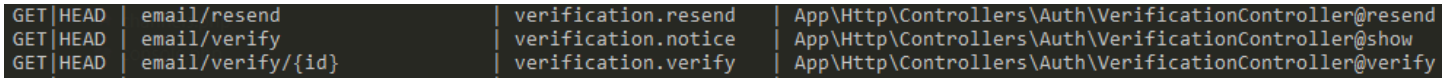
Au niveau de la base le champ **email_verified_at** existe déjà par défaut, donc rien à faire.

Au niveau des routes (**routes/web.php**) il faut modifier cette ligne

:

```
Auth::routes(['verify' => true]);
```

Ce qui a pour effet d’ajouter les routes correspondantes. On peut le vérifier avec **php artisan route:list** :



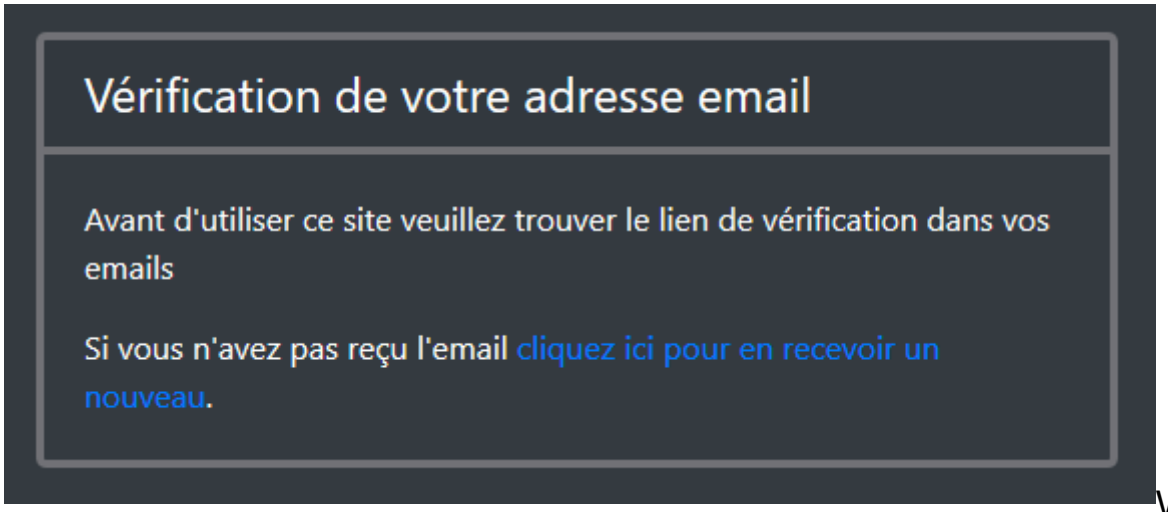
Le contrôleur **VerificationController** existe déjà.

Il ne reste plus qu’à protéger les routes à réserver aux utilisateurs dont l’email est vérifié en ajoutant le middleware **verified**. Pour le moment on ne va rien protéger puisqu’on a rien à protéger puisqu’on a juste la page d’accueil !

Il existe déjà la vue **resources/views/auth/verify.blade.php**. Mais on va la relooker pour notre site :

```
@extends('layouts.form')

@section('card')

    @component('components.card')

        @slot('title')
            @lang('Vérification de votre adresse email')
        @endslot

        @if (session('resent'))
            <div class="alert alert-success" role="alert">
                 @lang("Un nouveau lien de vérification a été envoyé à votre adresse email.")
            </div>
        @endif

         <p>@lang("Avant d'utiliser ce site veuillez trouver le lien de vérification dans vos emails")</p>
         @lang("Si vous n'avez pas reçu l'email ") <a href="{{ route('verification.resend') }}">@lang("cliquez ici pour en recevoir un nouveau")</a>.
```

```
    @endcomponent

@endsection
```



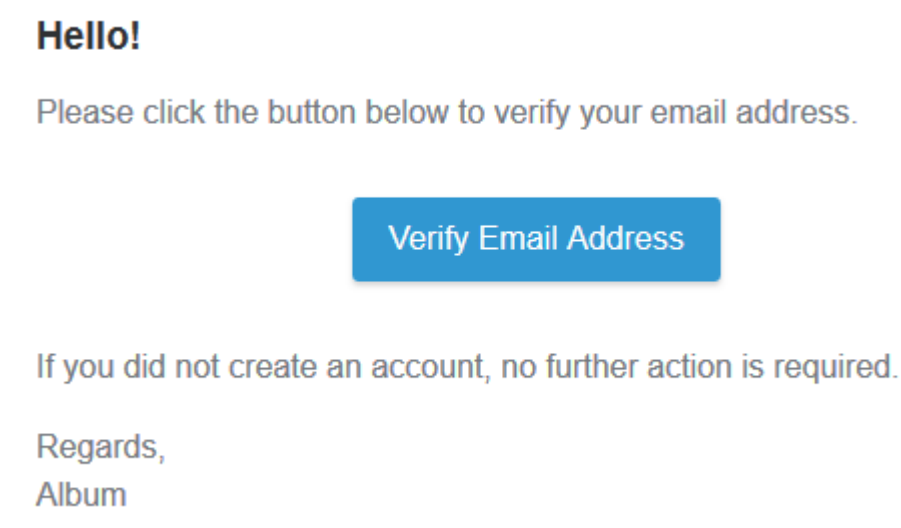
Vous pouvez tester cette vue en ajoutant le middleware dans la route de base :

```
Route::get('/', 'HomeController@index')->name('home')->middleware('verified');
```

Avec ce middleware maintenant ne sont plus accessibles que la connexion et l'enregistrement.

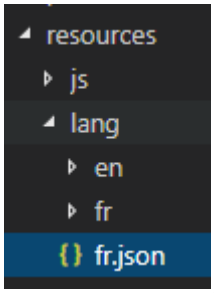
Les emails en français

Vous avez sans doute remarqué que les emails pour le renouvellement du mot de passe et de la vérification de l'email ont par défaut des textes en anglais :



Pour les traductions

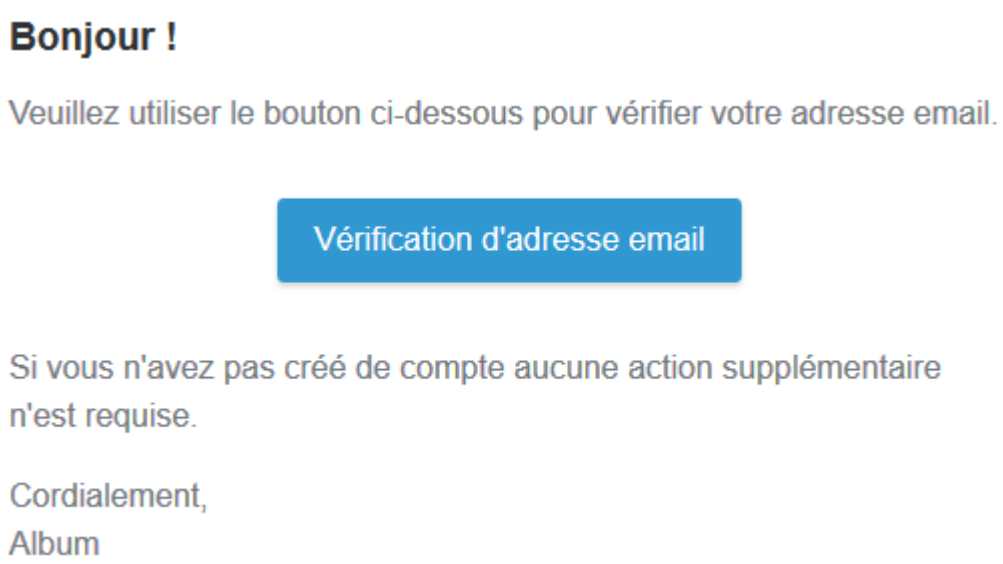
on peut utiliser des fichiers **JSON** placés dans le dossier **resources/lang**, on va en créer un pour le français :



Avec ce contenu :

```
{
    "Verify Email Address": "Vérification d'adresse email",
    "Please click the button below to verify your email address." : "Veuillez utiliser le bouton ci-dessous pour vérifier votre adresse email.",
     "If you did not create an account, no further action is required." : "Si vous n'avez pas créé de compte aucune action supplémentaire n'est requise.",
     "Reset Password Notification": "Renouvellement du mot de passe",
    "You are receiving this email because we received a password reset request for your account.": "Vous recevez ce message parce que nous avons reçu une demande de renouvellement du mot de passe pour votre compte.",
    "Reset Password": "Renouvellement",
    "If you did not request a password reset, no further action is required." : "Si vous n'avez pas fait cette demande, aucune action complémentaire n'est requise.",
    "Hello!": "Bonjour !",
    "Have a good day": "Bonne journée !",
    "Regards": "Cordialement"
}
```

Et voilà le résultat :



Conclusion

Dans ce chapitre on a vu :

- comment créer une vue à inclure et des composants pour faciliter l'écriture des vues de formulaire
- comme styliser les formulaires
- comment créer un helper pour rendre actif l'item en cours dans la barre de navigation
- comment ajouter la vérification de l'email
- comment avoir des emails de notification en français.

Pour vous simplifier la vie vous pouvez charger le projet dans son état à l'issue de ce chapitre.