

Création d'un gestionnaire de membres de club sportif : Front de l'administration


Dans ce TD, nous allons travailler sur la mise en place du FRONT du panneau d'administration de l'application

Table des matières

Modification de l'accueil du Dashboard.....	2
<x-app-layout> et <x-slot>	2
Création des liens dans le tableau de bord.....	6
Gestion des utilisateurs.....	6
Création de la vue utilisateur.....	6
Mise en forme de la page d'édition des profils.....	11
Modification de la page d'accueil.....	12
master.blade.php	12
Modification du menu de navigation navbar.blade.php.....	13
ToDo.....	13

Modification de l'accueil du Dashboard

Rappel : Blade fait le lien entre le contrôleur et les vues, gère les variables et le templating.

Ouvrir le fichier  resources->views->dashboard.blade.php

```
@extends("layout.master")
@section('content')
<x-app-layout>
  <x-slot name="header">
    <h2 class="font-semibold text-xl text-gray-800 leading-tight">
      {{ __('Dashboard') }}
    </h2>
  </x-slot>

  <div class="py-12">
    <div class="max-w-7xl mx-auto sm:px-6 lg:px-8">
      <div class="bg-white overflow-hidden shadow-sm sm:rounded-lg">
        <div class="p-6 bg-white border-b border-gray-200">
          Vous êtes connecté
        </div>
      </div>
    </div>
  </div>
</x-app-layout>
@endsection
```

<x-app-layout> et <x-slot>

Ces "balises" permettent de faire appel, dans ce cas, à un layout. C'est une syntaxe un peu particulière générée par Laravel Breeze grâce aux View Composers.

Depuis Laravel 8, on peut utiliser des layouts grâce aux "composants" et aux View Composers.

Quand on fera notre propre layout, on reverra la syntaxe par "héritage".

Considérons donc simplement que ces balises permettent d'utiliser notre layout !
D'ailleurs, par défaut ce layout, c'est celui dans layouts->**app.blade.php** (cf. page suivante)

resources\views\layouts\app.blade.php

```
<!DOCTYPE html>
<html lang="{{ str_replace('_', '-', app()->getLocale()) }}">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <meta name="csrf-token" content="{{ csrf_token() }}">

    <title>{{ config('app.name', 'Laravel') }}</title>

    <!-- Fonts -->
    <link rel="stylesheet"
href="https://fonts.googleapis.com/css2?family=Nunito:wght@400;600;700&display=s
wap">

    <!-- Styles -->
    <link rel="stylesheet" href="{{ asset('css/app.css') }}">

    <!-- Scripts -->
    <script src="{{ asset('js/app.js') }}" defer></script>
  </head>
  <body class="font-sans antialiased">
    <div class="min-h-screen bg-gray-100">
      @include('layouts.navigation')

      <!-- Page Heading -->
      <header class="bg-white shadow">
        <div class="max-w-7xl mx-auto py-6 px-4 sm:px-6 lg:px-8">
          {{ $header }}
        </div>
      </header>

      <!-- Page Content -->
      <main>
        {{ $slot }}
      </main>
    </div>
  </body>
</html>
```

{{ str_replace('_', '-', app()->getLocale()) }} :

Ici, on utilise `str_replace` de PHP pour remplacer les éventuels `_` par des `-` dans la chaîne `app()->getLocale()`.

`app()` est un helper qui permet d'accéder aux variables de configuration définies dans le fichier `app.php`.

Ici, on va chercher la valeur de "locale". On l'avait remplacée par "fr" au début !

meta name="csrf-token" content="{{ csrf_token() }}" :

Il s'agit ici d'initialiser le TOKEN CSRF, une mesure de protection contre la faille du même nom

{{ config('app.name', 'Laravel') }} :

Ici, on fait appel à un autre helper "config" qui va chercher dans le fichier de configuration spécifié la valeur de la donnée spécifiée.

Si elle n'existe pas, on affiche une valeur par défaut, ici "Laravel". Dans notre cas, on va chercher dans le fichier de configuration `app.php` la valeur de la variable "name".

{{ asset('css/app.css') }} :

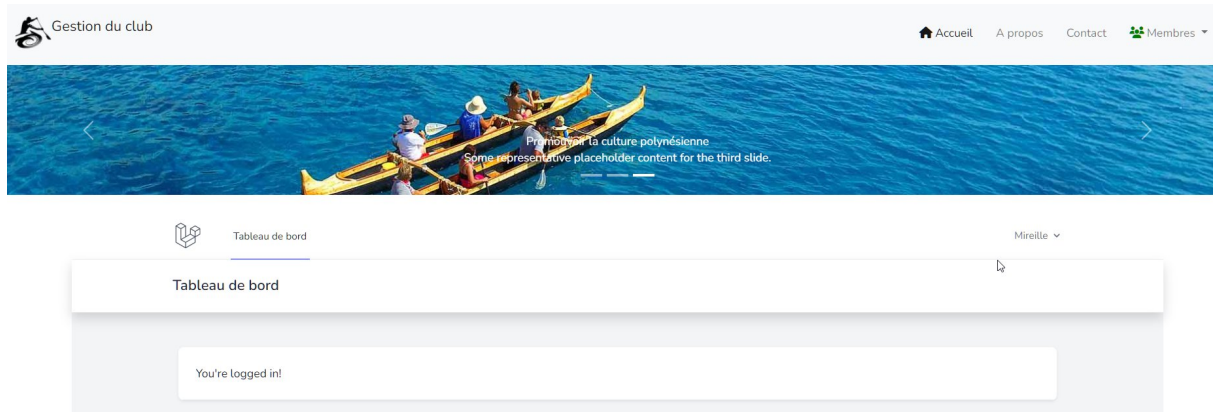
Le helper `asset` nous permet de charger les feuilles de style, images et js qu'on aura mis dans le dossier public. Ici on va chercher le fichier `app.css` dans le dossier `css` du dossier public.

{{ \$header }} :

On affiche la variable nommée "header". Dans notre cas, header est en fait définie dans dashboard.blade.php grâce à x-slot name="header".

Cette syntaxe, encore une fois, est très particulière, on ne la réutilisera pas. On la retrouve aussi avec {{ \$slot }} qui affichera tout le reste de notre contenu dans dashboard.blade.php

🔗 Connectez-vous et atteignez votre tableau de bord : <http://127.0.0.1:8000/dashboard>



Observez le fichier `resources\views\dashboard.blade.php` (j'ai rajouté notre layout master pour garder la navigation principale du site)

```
@extends("layout.master")
@section('content')
<x-app-layout>
  <x-slot name="header">
    <h2 class="font-semibold text-xl text-gray-800 leading-tight">
      {{ __('Dashboard') }}
    </h2>
  </x-slot>

  <div class="py-12">
    <div class="max-w-7xl mx-auto sm:px-6 lg:px-8">
      <div class="bg-white overflow-hidden shadow-sm sm:rounded-lg">
        <div class="p-6 bg-white border-b border-gray-200">
          You're logged in!
        </div>
      </div>
    </div>
  </div>
</x-app-layout>
@endsection
```

On utilise `{{ __('Dashboard') }}`

On pourrait s'attendre à ce que cette portion de code affiche "Dashboard" sur notre page. Pourtant, c'est bien "Tableau de bord" qui est affiché. **Tu saurais pourquoi ?**

En fait, quand on utilise la syntaxe double underscore (`__()`), on dit à Blade que l'on veut afficher le mot entre guillemets dans la langue que renseignée dans le fichier de configuration app.php.

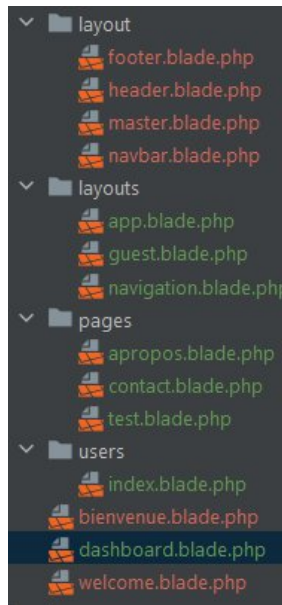
Laravel va alors aller chercher si un fichier nommé `LANGUE.json` existe. Si c'est le cas, il va récupérer la valeur de la clé qui est ici le mot.

Si on regarde dans le fichier `resources/lang/fr.json`, on trouve cette ligne :

"Dashboard": "Tableau de bord",

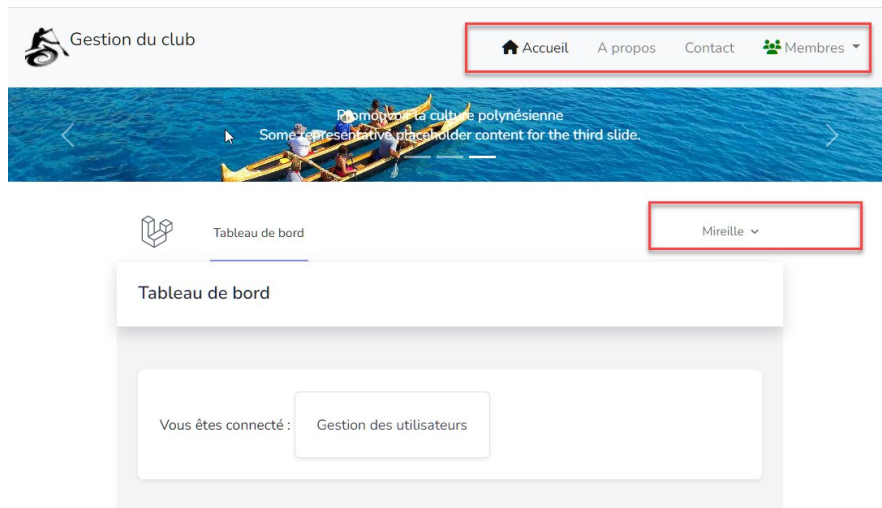
Bilan, on traduit "Dashboard" en "Tableau de bord".

Mais à ce stade, nous nous retrouvons avec plusieurs layouts !!!



Ceux que nous avons créés au tout début de notre projet et ceux créés automatiquement 😞

Il va falloir revoir l'ensemble de nos vues et nos interfaces un peu plus tard 😊



Création des liens dans le tableau de bord

Gestion des utilisateurs

Dans `resources\views\dashboard.blade.php`

```
<div class="p-6 bg-white border-b border-gray-200">
  You're logged in! :
  <a href="{{ route('users') }}" class="inline-flex items-center px-6 py-
4 border border-gray-400 shadow-sm text-base font-medium rounded-md text-
gray-700 bg-white">
    Gestion des utilisateurs
  </a>
</div>
```

Dans `"{{ route('users') }}"`, `route` est un helper...

Cela nous permet de renseigner des liens internes à notre application via le nom des routes.

Dans cet exemple :

```
<?php
Route::get('/users', [UserController::class,
'index'])->middleware(['auth', 'admin'])->name('users');
```

On a donné le nom de "users" à la route. C'est ce nom qui est réutilisé.

Laravel va alors transformer `route('users')` en `monapplication.com/users`.

C'est bien d'utiliser les noms de routes car on est ainsi indépendant du nom de notre site et on peut changer l'URL à tout moment sans devoir changer toutes nos vues.

Création de la vue utilisateur

🔗 Créez un 📁 sous-dossier « users » dans `resources\views`, puis un fichier 📄 `index.blade.php`

On va refaire appel au layout du dashboard et on affiche 2 titres

```
<x-app-layout>
  <x-slot name="header">
    <h2 class="font-semibold text-xl text-gray-800 leading-tight">
      Gestion des utilisateurs
    </h2>
  </x-slot>

  <div class="py-12">
    <div class="max-w-7xl mx-auto sm:px-6 lg:px-8">
      <div class="bg-white overflow-hidden shadow-sm sm:rounded-lg">
        <h3 class="text-3xl mx-4 my-4">Les utilisateurs du
site</h3>

      </div>
    </div>
  </div>
</x-app-layout>
```

Le but va être d'afficher la liste de tous les utilisateurs.

Pour cela on va utiliser un tableau et mettre une ligne par user. On va donc devoir faire un "foreach users"

Blade dispose d'une fonction nommée `@foreach`.

On va faire `@foreach($users as $user)`. Comme en PHP !

```

<x-app-layout>
  <x-slot name="header">
    <h2 class="font-semibold text-xl text-gray-800 leading-tight">
      Gestion des utilisateurs
    </h2>
  </x-slot>

  <div class="py-12">
    <div class="max-w-7xl mx-auto sm:px-6 lg:px-8">
      <div class="bg-white overflow-hidden shadow-sm sm:rounded-lg">
        <h3 class="text-3xl mx-4 my-4">Les utilisateurs du site</h3>
        <div class="flex flex-col">
          <div class="-my-2 overflow-x-auto sm:-mx-6 lg:-mx-8">
            <div class="py-2 align-middle inline-block min-w-full sm:px-6 lg:px-8">
              <div class="shadow overflow-hidden border-b border-gray-200 sm:rounded-lg">
                <table class="min-w-full divide-y divide-gray-200">
                  <thead class="bg-gray-50">
                    <tr>
                      <th scope="col" class="px-6 py-3 text-left text-xs font-medium text-gray-500
uppercase tracking-wider">
                        Nom
                      </th>
                      <th scope="col" class="px-6 py-3 text-left text-xs font-medium text-gray-500
uppercase tracking-wider">
                        Email
                      </th>
                      <th scope="col" class="px-6 py-3 text-left text-xs font-medium text-gray-500
uppercase tracking-wider">
                        Role
                      </th>
                      <th scope="col" class="relative px-6 py-3">
                        <span class="sr-only">Modifier</span>
                      </th>
                    </tr>
                  </thead>

```

```
<tbody class="bg-white divide-y divide-gray-200">
    @foreach($users as $user)
        <tr>
            <td class="px-6 py-4 whitespace-nowrap">
                <div class="text-sm font-medium text-gray-900">
                    {{ $user->name }}
                </div>
            </td>
            <td class="px-6 py-4 whitespace-nowrap">
                <div class="text-sm font-medium text-gray-900">
                    {{ $user->email }}
                </div>
            </td>
            <td class="px-6 py-4 whitespace-nowrap">
                <div class="text-sm font-medium text-gray-900">
                    {{ $user->admin ? 'Administrateur' : 'Utilisateur' }}
                </div>
            </td>
            <td class="px-6 py-4 whitespace-nowrap text-right text-sm font-medium">
                <a href="{{ route('users.edit', $user) }}" class="text-indigo-600
hover:text-indigo-900">Modifier</a>
            </td>
        </tr>
    @endforeach
</tbody>
</table>
</div>
</div>
</div>
</div>
</div>
</x-app-layout>
```

On remarque que l'on affiche les variables avec les « moustaches » `{{ }}`.

A l'intérieur on peut mettre n'importe quelle ligne PHP classique qui affiche du contenu, on peut donc mettre une ternaire.

Enfin, on utilise la `route()` en passant un paramètre à la route qui n'est rien d'autre que `$user`, pour l'implicit binding !

⚠ On a mis un middleware sur les routes `users.edit` et `users.update` (📄 `routes/web.php`) pour que seules les personnes authentifiées y aient accès. Mais... Une fois connecté, on pourrait accéder à cette page et indiquer n'importe quel id user dans l'URL, et donc potentiellement modifier le profil de quelqu'un d'autre que soi !

Il faut corriger cela.

On pourrait utiliser des politiques <https://laravel.com/docs/8.x/authorization> pour palier à ce problème, mais on va rester dans des choses simples.

Nous allons modifier les contrôleurs comme suit : 📄 `UsersController.php`

```
public function edit(User $user) {
    if($user->id == request()->user()->id) {
        abort(403);
    }
    return view('users.edit', compact('user'));
}
```

Il nous reste quand même à autoriser l'administrateur à modifier les profils...

Le contrôleur final :

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Models\User;

class UsersController extends Controller
{
    public function index()
    {
        $users = User::all();
        return view('users.index', compact('users'));
    }

    public function show(User $user) {
        return view('users.show', compact('user'));
    }

    /* V1
    public function edit(User $user) {
        return view('users.edit', compact('user'));
    }*/

    /* V2
    public function edit(User $user) {
        if($user->id == request()->user()->id) {
            abort(403);
        }
        return view('users.edit', compact('user'));
    }

    */

    public function edit(User $user) {
        $this->isAbleToEdit($user);
        return view('users.edit', compact('user'));
    }

    /*
    public function update(Request $request, User $user) {
        // Afficher une valeur :
    */
}
```

```

        dd($request->input('name')); // On considère que notre requête
        contient un champ "name"
        // On valide les données
        $validatedData = $request->validate([
            'name' => ['required', 'min:2', 'max:50'],
            'email' => 'required|email',
        ]);
        $user->update($validatedData);
        return redirect()->back()->with('success', 'Les informations ont
        bien été modifiées');
    }*/
    public function update(Request $request, User $user) {
        $this->isAbleToEdit($user);
        $validatedData = $request->validate([
            'name' => ['required', 'min:2', 'max:50'],
            'email' => 'required|email',
        ]);

        $user->update($validatedData);

        return redirect()->back()->with('success', 'Les informations ont
        bien été modifiées');
    }

    public function destroy(User $user) {
        $user->delete();
        return redirect()->back()->with('success', 'L\'utilisateur a bien
        été supprimé');
    }

    private function isAbleToEdit(User $user) {
        if($user->id == request()->user()->id
and !request()->user()->admin) {
            abort(403);
        }
    }
}

```

Mise en forme de la page d'édition des profils

Création d'un fichier  resources\views\users\edit.blade.php

```
@extends('layout.master')
@section('title', 'Modification de mon profil')
@section('content')
    <div class="bg-white overflow-hidden shadow-sm sm:rounded-lg">
        <h3 class="text-3xl mx-4 my-4">Modification de mon profil</h3>
        @if(session('success'))
            <div class="text-xl text-green-400">
                {{ session('success') }}
            </div>
        @endif
        <form action="{{ route('users.update', $user) }}" method="POST"
class="mx-auto">
            @csrf
            @method('PUT')
            <div class="px-4 py-5 bg-white sm:p-6">
                <div class="py-2">
                    <label for="name" class="block text-sm font-medium
text-gray-700">Prénom</label>
                    <input type="text" name="name" id="name"
autocomplete="given-name" class="mt-1 focus:ring-indigo-500 focus:border-
indigo-500 block w-full shadow-sm sm:text-sm border-gray-300 rounded-md"
value="{{ old('name', $user->name) }}">
                    @error('name')
                        <span class="text-red-600">{{ $message }}</span>
                    @enderror
                </div>
                <div class="py-2">
                    <label for="email" class="block text-sm font-medium
text-gray-700">Email</label>
                    <input type="text" name="email" id="email"
autocomplete="given-name" class="mt-1 focus:ring-indigo-500 focus:border-
indigo-500 block w-full shadow-sm sm:text-sm border-gray-300 rounded-md"
value="{{ old('email', $user->email) }}">
                    @error('email')
                        <span class="text-red-600">{{ $message }}</span>
                    @enderror
                </div>
                <div class="py-2">
                    <input type="submit" class="cursor-pointer inline-flex
items-center w-1/4 py-4 border border-gray-400 shadow-sm text-base font-
medium rounded-md text-gray-700 bg-white justify-center" value="Modifier">
                </div>
            </div>
        </form>
    </div>
@endsection
```

Nous allons maintenant modifier notre page d'accueil 😊

Modification de la page d'accueil

master.blade.php

```
<!DOCTYPE html>
<html lang="{{ str_replace('_', '-', app()->getLocale()) }}">
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <meta name="csrf-token" content="{{ csrf_token() }}">
    <meta name="description" content="@yield('meta', 'Le site des
adhérents')">
    <title>Club Va'a - @yield('title')</title>
    <!-- Styles -->
    <link rel="stylesheet" href="{{ asset('css/app.css') }}">
    <link rel="stylesheet" href="{{ asset('css/bootstrap.css') }}">
    <link rel="stylesheet" href="{{ asset('css/tailwind.css') }}">
</head>
<body>

@include('layout.navbar')
@include('layout.header')

<div class="container">

    @yield('content')

</div>

@include('layout.footer')

@yield('script')

{{--<script src="/js/app.js"></script>--}}
<script src="{{ asset('js/app.js') }}" defer></script>

</body>
</html>
```

J'ai juste fait une modification mineure de l'en-tête pour améliorer le référencement et la sécurité

```
<meta name="csrf-token" content="{{ csrf_token() }}">
```

Il s'agit ici d'initialiser le TOKEN CSRF, une mesure de protection contre la faille du même nom¹.

¹ <https://fr.barracuda.com/glossary/csrf>

Modification du menu de navigation navbar.blade.php

Voici un exemple :

```
<header>
  <a href="/">Accueil</a>
  @auth
    <a href="{{ route('users.edit', Auth::user()) }}">Profil</a>
    <form action="{{ route('logout') }}" method="POST">
      @csrf
      <button type="submit"
href="{{ route('logout') }}">Déconnexion</button>
    </form>
    @if(Auth::user()->admin)
      <a href="{{ route('dashboard') }}">Dashboard</a>
    @endif
  @else
    <a href="{{ route('login') }}">Connexion</a>
  @endauth
</header>
```

@auth est une condition qui vérifie si on est authentifié.

On peut utiliser cette classe pour accéder aux données de l'utilisateur en faisant `Auth::user()`.

Un formulaire et pas un lien pour la déconnexion ?

Il suffit de regarder la méthode supportée par la route qui gère le logout.

Si on fait `php artisan route:list`, on remarque que **Laravel Breeze** qui gère l'authentification ne supporte que le POST pour le logout.

On doit donc utiliser un formulaire et le token CSRF.

À vous de l'adapter à votre navbar 😊

Il sera nécessaire de modifier la page `app.blade.php`... A vous de réfléchir le pourquoi et le comment !

ToDo

Complétez votre application...

Rajout d'un lien inscription (dans « membres ») pour les utilisateurs non connectés

Amélioration des formulaires

Rajout du lien vers l'accueil sur le logo

Permettre la suppression d'un utilisateur non-administrateur (pour les administrateurs seulement !)

Remarques

Laravel 8 ne sera plus maintenu le 24 janvier 2023...

La prochaine étape sera donc la migration d'un site sous Laravel 😊