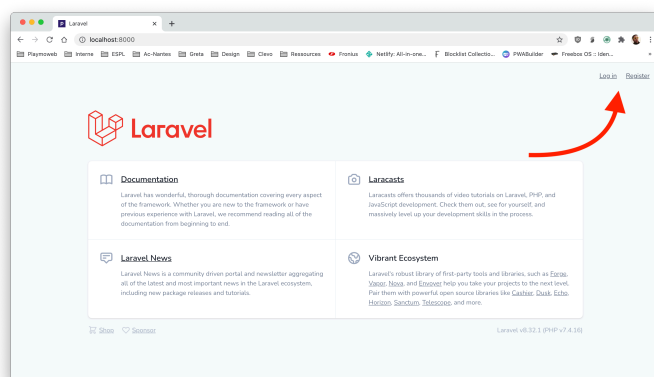
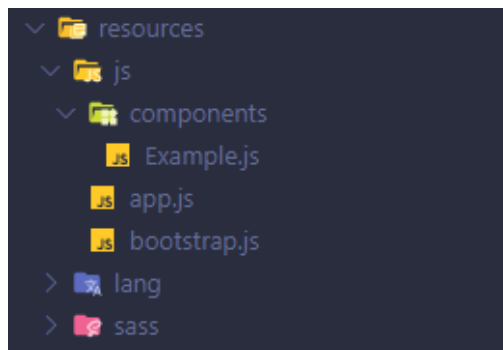


Voici une **présentation claire et structurée** de l'**authentification intégrée via le package laravel/ui** pour les projets **Laravel avec Blade** (auth classique)



Introduction à laravel/ui



Avant l'arrivée de **Laravel Breeze**, **Jetstream** ou **Fortify**, le package officiel **laravel/ui** était (et reste encore) la **solution classique d'authentification avec Blade**.

Il fournit un **scaffold** (structure prête à l'emploi) pour :

- Le **login**
- L'**inscription**
- La **réinitialisation du mot de passe**
- Et la **déconnexion**

... le tout avec un **frontend Blade traditionnel**, sans Inertia ni Livewire.

Installation du package

1. Installer Laravel

```
composer create-project laravel/laravel mon_projet  
cd mon_projet
```

2. Installer le package laravel/ui

```
composer require laravel/ui
```

3. Générer les vues d'authentification (avec Blade)

```
php artisan ui bootstrap --auth
```

Ici bootstrap peut être remplacé par vue ou react si tu veux un front-end différent. Pour du **Blade classique**, on garde bootstrap.

4. Installer les dépendances front-end et compiler les assets

```
npm install  
npm run dev
```

Structure générée

Après la commande `php artisan ui bootstrap --auth`, Laravel crée :

Les routes d'auth

Dans `routes/web.php` :

```
Auth::routes();
```

Cela ajoute automatiquement les routes pour :

- `/login`
- `/register`
- `/password/reset`
- `/logout`
- `/home`

Les contrôleurs

Dans `app/Http/Controllers/Auth/` :

- `LoginController`
- `RegisterController`
- `ForgotPasswordController`
- `ResetPasswordController`

- `VerificationController`

Ces contrôleurs utilisent les **traits** du framework (`Illuminate\Foundation\Auth`) qui contiennent déjà toute la logique d'authentification.

Les vues Blade

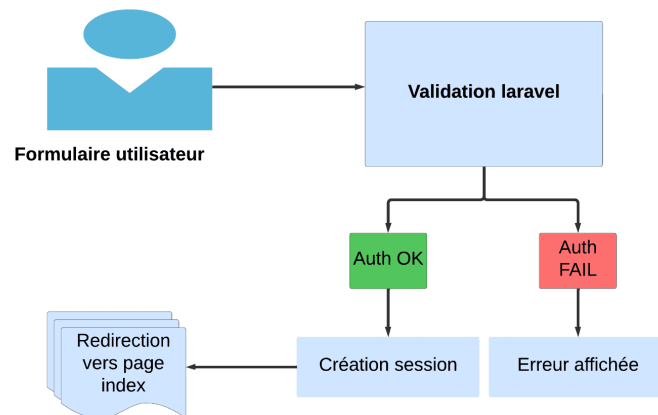
Dans `resources/views/auth/` :

- `login.blade.php`
- `register.blade.php`
- `passwords/email.blade.php`
- `passwords/reset.blade.php`

Et la page d'accueil protégée :

- `resources/views/home.blade.php`

Fonctionnement de l'authentification



Laravel fournit :

- Une **migration** pour la table `users` (avec `email`, `password`, `remember_token`).
- Des **middlewares** :
 - `auth` → protège les routes nécessitant une connexion
 - `guest` → empêche un utilisateur connecté d'accéder à la page de connexion ou d'inscription
- Un **système de sessions** pour gérer la persistance de la connexion.

Exemple :

```
Route::get('/dashboard', function () {  
    return view('dashboard');  
})->middleware('auth');
```

Commandes utiles

Commande	Description
php artisan ui bootstrap --auth	Génère les vues Blade + routes + contrôleurs
php artisan migrate	Crée la table users
php artisan make:auth	Ancienne commande (Laravel < 6), remplacée par ui
php artisan serve	Lance le serveur de dev

Personnalisation

Tu peux facilement modifier :

- Les vues dans `resources/views/auth/`
 - Les redirections dans `LoginController` ou `RegisterController` :

```
protected $redirectTo = '/dashboard';
```
 - Les validations dans la méthode `validator()` du `RegisterController`
 - Le comportement après login ou logout.
-

Avantages de laravel/ui

Avantage	Description
Simple	Parfait pour les apps classiques sans SPA
Rapide à mettre en place	1 commande et tout fonctionne
Basé sur Blade + Bootstrap	Facile à personnaliser
Léger	Pas de Livewire ou Inertia nécessaires

Limites

Limite	Description
Pas d'API	Pas adapté aux front-end SPA (React/Vue modernes)
Ancien système	Supplanté par Breeze et Jetstream depuis Laravel 8+
Peu d'options	Pas de gestion de rôles ou d'authentification avancée

Alternatives modernes

Package	Stack	Description
Laravel Breeze	Blade, Vue, React	Minimal et moderne
Laravel Jetstream	Livewire ou Inertia	Auth + profils + 2FA
Laravel Fortify	Backend-only	API d'authentification configurable