Laravel Breeze: A Simple and Elegant Starter Kit

Laravel Breeze is the official starter kit designed for developers who want a simple, minimal, and elegant way to set up authentication in a new Laravel project. It provides the perfect foundation for quickly starting applications without the extra complexity of larger starter kits like Jetstream.

What Is Laravel Breeze?

Laravel Breeze is a lightweight authentication scaffolding package for Laravel. It sets up essential authentication features such as login, registration, password reset, email verification, and logout. Unlike heavier starter kits, Breeze focuses on simplicity, making it an excellent choice for learning or starting projects where you want full control over the front-end stack.

Why Choose Breeze Over Jetstream or Other Kits?

While Laravel Jetstream (see comparison of Laravel Starter Kits) and other starter kits provide advanced features like team management and profile photos, Breeze keeps everything minimal. This is perfect for developers who:

- Prefer simplicity and customization over pre-built features.
- Want to quickly prototype applications with basic authentication.
- Need a starting point for learning Laravel authentication without being overwhelmed.

Laravel Starter Kits

Features of Laravel Breeze

- **Authentication**: Login, registration, password reset, and email verification ready to use.
- **Blade Templates**: By default, Breeze comes with Blade templates styled using Tailwind CSS.
- **Inertia Support**: Optionally install with Vue or React if you prefer a SPA-like front-end.
- **Minimalistic**: No extra bloat, giving developers full flexibility to extend functionality.

Using Breeze in Real Projects

Breeze is great for projects where you want authentication out of the box but prefer to design your own user experience. For example, if you are building a blog, SaaS MVP, or internal dashboard, Breeze gives you a working authentication system in minutes, letting you focus on your application logic.

Laravel Breeze strikes the perfect balance between simplicity and flexibility. It’s easy to install, intuitive to customize, and powerful enough to handle authentication needs for most small to medium-sized applications. If you’re just starting with Laravel or need a lightweight authentication starter kit, Breeze is an excellent choice. it an excellent choice for learning or starting projects where you want full control over the front-end stack.

Installation and Setup

```
composer require laravel/breeze --dev
php artisan breeze:install
npm install && npm run dev
php artisan migrateCode language: JavaScript (javascript)
```

These commands install Breeze, scaffold the authentication views (Blade or Inertia with Vue/React), compile front-end assets, and migrate database tables for user management.

Protecting Routes with Breeze

```
// routes/web.php

use Illuminate\Support\Facades\Route;

Route::get('/dashboard', function () {
    return view('dashboard');
})->middleware(['auth', 'verified'])->name('dashboard');Code language:
PHP (php)
```

The `auth` middleware ensures only authenticated users can access the dashboard. The `verified` middleware adds email verification support if enabled.

Controller Example

```php
// app/Http/Controllers/ProfileController.php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class ProfileController extends Controller
{
    public function __construct()
    {
        $this->middleware('auth'); // Protects all profile routes
    }

    public function index()
    {
        return view('profile.index');
    }
}
```
Code language: PHP (php)

The constructor applies the `auth` middleware to the entire controller, ensuring every method requires a logged-in user.

Blade Example: Showing Auth Links

```
<!-- resources/views/layouts/navigation.blade.php -->

<nav>
    @auth
        <a href="{{ url('/dashboard') }}">Dashboard</a>
        <form method="POST" action="{{ route('logout') }}">
            @csrf
            <button type="submit">Logout</button>
        </form>
    @else
        <a href="{{ route('login') }}">Login</a>
        <a href="{{ route('register') }}">Register</a>
    @endauth
</nav>Code language: HTML, XML (xml)
```

The `@auth` and `@else` directives dynamically show links depending on whether the user is logged in or not.

Inertia + Vue Example

```
// resources/js/Pages/Dashboard.vue

<template>
  <div>
    <h1>Welcome, {{ $page.props.auth.user.name }}</h1>
    <Link href="/logout" method="post" as="button">Logout</Link>
  </div>
</template>

<script setup>
import { Link } from '@inertiajs/vue3'
</script>Code language: HTML, XML (xml)
```

Inertia integrates Laravel’s backend with Vue (or React). The `$page.props.auth.user` object gives direct access to authenticated user data inside the SPA.

Breeze vs Jetstream vs Fortify

Feature	Breeze	Jetstream	Fortify
Purpose	Minimal starter kit with auth scaffolding	Full-featured starter kit with teams, API tokens, and profile management	Backend-only authentication services without views
Front-end	Blade + Tailwind (with optional Inertia Vue/React)	Blade + Tailwind or Inertia Vue/React	None (headless, API-driven)
Complexity	Lightweight, easy to customize	More opinionated, includes advanced features	Requires building your own UI
Best Use Case	Small/medium apps, MVPs, learning projects	SaaS apps, team-based projects, larger apps	Custom applications where UI is built manually
Learning Curve	Beginner-friendly	Intermediate/advanced	Intermediate, requires UI work

Conclusion

Laravel Breeze offers a clean, minimal, and extensible authentication setup. With just a few commands, you get login, registration, and password management ready to use. Whether you prefer Blade templates or a modern SPA approach with Inertia, Breeze provides the perfect foundation for building secure Laravel applications. If you need advanced features out of the box, consider Jetstream, or for maximum flexibility, Fortify may be the better fit.