

Laravel Breeze 2.x

<p align="center"> </p> <p align="center"> </p>






Съдържание

- Относно Breeze
- Характеристики
- Изисквания
- Инсталация
- Налични Стекове
- Използване
- Структура на Проекта
- Разширена Конфигурация
- Тестване
- Security
- Contributing
- Лиценз

Относно Breeze

Laravel Breeze е минималистична и елегантна имплементация на всички Laravel authentication функции, включително login, регистрация, password reset, email verification и profile управление. Breeze е перфектната стартова точка за нови Laravel приложения.

Защо Breeze?

-  **Минималистичен** - Само необходимото, без излишества
-  **Модерен** - Използва съвременни технологии и best practices
-  **Гъвкав** - 6 различни frontend стека на избор
-  **Прост** - Лесен за разбиране и разширяване код
-  **Production-Ready** - Пълна authentication система

✨ Характеристики

Authentication Features

- 🔒 Потребителска регистрация
- 🔑 Login/Logout
- ✉ Email верификация
- 🔄 Password reset
- 👤 Profile управление
- 🛡 Password confirmation за чувствителни операции

Technical Features

- 🎨 6 Frontend Стека: Blade, Livewire (Class/Functional), Vue, React, API
- 🌙 Dark Mode Support: Опционална поддръжка на тъмен режим
- 🛠 TypeScript Support: За React и Vue стековете
- 🎯 SSR Support: Server-Side Rendering за Inertia стековете
- 🧪 Testing Ready: Pest или PHPUnit тестове включени
- 📱 Responsive Design: Mobile-first подход с Tailwind CSS
- ⚡ Vite Integration: Бърз build процес

📋 Изисквания

- PHP \geq 8.2
- Laravel \geq 11.0
- Composer
- Node.js \geq 18.x (за frontend стековете)
- npm, pnpm, yarn, bun или deno (package manager)

🚀 Инсталация

Стъпка 1: Създаване на Laravel проект

```
bash
composer create-project laravel/laravel example-app
cd example-app
```

Стъпка 2: Инсталиране на Breeze

```
bash
```

```
composer require laravel/breeze --dev
```

Стъпка 3: Инсталиране на предпочитан стек

```
bash

# Интерактивен режим (препоръчително)
php artisan breeze:install

# Директна инсталация
php artisan breeze:install blade
php artisan breeze:install livewire
php artisan breeze:install vue
php artisan breeze:install react
php artisan breeze:install api
```

Стъпка 4: Миграция на базата данни

```
bash

php artisan migrate
```

Стъпка 5: Стартиране (за frontend стековете)

```
bash

npm run dev
# или
php artisan serve
```

Налични Стекове

1. Blade Stack

Описание: Traditional server-side rendered приложение с Alpine.js за интерактивност

```
bash

php artisan breeze:install blade
```

Функционалности:

- ✓ Server-side rendering
- ✓ Alpine.js за JavaScript функционалност
- ✓ Tailwind CSS стилизация

- ☒ Традиционен Blade шаблонен engine

Подходящ за: Traditional web приложения, SEO-critical проекти

2. Livewire Stack (Class API)

Описание: Full-stack framework с Livewire Volt Class API

```
bash  
php artisan breeze:install livewire
```

Функционалности:

- ☒ Reactive components без JavaScript
- ☒ Real-time validation
- ☒ Volt Class API
- ☒ Full PHP development

Подходящ за: PHP разработчици, които искат reactive UI без JavaScript

3. Livewire Stack (Functional API)

Описание: Full-stack framework с Livewire Volt Functional API

```
bash  
php artisan breeze:install livewire-functional
```

Функционалности:

- ☒ По-кратък синтаксис с функционален подход
- ☒ Същите предимства като Class API
- ☒ По-малко boilerplate код

Подходящ за: Разработчици, предпочитащи функционално програмиране

4. Vue Stack (Inertia.js)

Описание: Modern SPA с Vue 3 и Inertia.js

```
bash
```

```
# JavaScript
```

```
php artisan breeze:install vue
```

```
# TypeScript + Dark Mode + SSR + ESLint
```

```
php artisan breeze:install vue --typescript --dark --ssr --eslint
```

Функционалности:

- ☒ Vue 3 Composition API
- ☒ Inertia.js за SPA без API
- ☒ TypeScript support (опционално)
- ☒ SSR support (опционално)
- ☒ ESLint + Prettier (опционално)
- ☒ Dark mode (опционално)

Опции:

- `--typescript` - TypeScript вместо JavaScript
- `--ssr` - Server-Side Rendering
- `--dark` - Dark mode поддръжка
- `--eslint` - ESLint и Prettier конфигурация
- `--pest` - Pest вместо PHPUnit

Подходящ за: Vue разработчици, modern SPA приложения

5. React Stack (Inertia.js)

Описание: Modern SPA с React 18 и Inertia.js

```
bash
```

```
# JavaScript
```

```
php artisan breeze:install react
```

```
# TypeScript + всички функции
```

```
php artisan breeze:install react --typescript --dark --ssr --eslint
```

Функционалности:

- ☒ React 18 с Hooks
- ☒ Inertia.js integration

- ☒ HeadlessUI компоненти
- ☒ TypeScript support (опционално)
- ☒ SSR support (опционално)
- ☒ ESLint + Prettier (опционално)

Опции: Същите като Vue стека

Подходящ за: React разработчици, modern SPA приложения

6. API Stack

Описание: API-only authentication със Sanctum

```
bash  
php artisan breeze:install api
```

Функционалности:

- ☒ Laravel Sanctum token authentication
- ☒ API endpoints за authentication
- ☒ CORS конфигурация
- ☒ Ready за mobile или frontend framework

API Endpoints:

```
POST /api/register  
POST /api/login  
POST /api/logout  
POST /api/forgot-password  
POST /api/reset-password  
POST /api/verify-email/{id}/{hash}  
POST /api/email/verification-notification
```

Подходящ за: Mobile apps, отделен frontend, microservices



Използване

Основни Routes

```
php
```

```

// Authentication routes
Route::get('/register', ...);
Route::post('/register', ...);
Route::get('/login', ...);
Route::post('/login', ...);
Route::post('/logout', ...);

// Password reset
Route::get('/forgot-password', ...);
Route::post('/forgot-password', ...);
Route::get('/reset-password/{token}', ...);
Route::post('/reset-password', ...);

// Email verification
Route::get('/verify-email', ...);
Route::get('/verify-email/{id}/{hash}', ...);
Route::post('/email/verification-notification', ...);

// Profile
Route::get('/profile', ...)->middleware('auth');
Route::patch('/profile', ...)->middleware('auth');
Route::delete('/profile', ...)->middleware('auth');

```

Защитаване на Routes

```

php

// Само authenticated потребители
Route::middleware(['auth']->group(function () {
    Route::get('/dashboard', ...);
});

// Verified email изискване
Route::middleware(['auth', 'verified']->group(function () {
    Route::get('/premium', ...);
});

// Password confirmation
Route::middleware(['auth', 'password.confirm']->group(function () {
    Route::delete('/account', ...);
});

```

Customization Examples

Добавяне на полета към регистрацията

1. Добави полета в миграцията:

```
php

// database/migrations/..._create_users_table.php
Schema::create('users', function (Blueprint $table) {
    $table->id();
    $table->string('name');
    $table->string('email')->unique();
    $table->string('phone')->nullable(); // ново поле
    $table->timestamp('email_verified_at')->nullable();
    $table->string('password');
    $table->rememberToken();
    $table->timestamps();
});
```

2. Обнови модела:

```
php

// app/Models/User.php
protected $fillable = [
    'name',
    'email',
    'phone', // добави тук
    'password',
];
```

3. Обнови формата (Blade пример):

```
html

<!-- resources/views/auth/register.blade.php -->
<div>
    <x-input-label for="phone" :value="__('Phone')" />
    <x-text-input id="phone" name="phone" type="text" />
    <x-input-error :messages="$errors->get('phone')" />
</div>
```

4. Обнови контролера:

```
php
```



```
// app/Http/Controllers/Auth/RegisteredUserController.php
$request->validate([
    'name' => ['required', 'string', 'max:255'],
    'email' => ['required', 'string', 'email', 'max:255', 'unique:users'],
    'phone' => ['nullable', 'string', 'max:20'], // добави
    'password' => ['required', 'confirmed', Rules\Password::defaults()],
]);

User::create([
    'name' => $request->name,
    'email' => $request->email,
    'phone' => $request->phone, // добави
    'password' => Hash::make($request->password),
]);
```

Структура на Проекта

```
breeze-2.x/
├── src/
│   ├── BreezeServiceProvider.php    # Service provider регистрация
│   └── Console/
│       ├── InstallCommand.php      # Главна install команда
│       ├── InstallsApiStack.php    # API stack installer
│       ├── InstallsBladeStack.php  # Blade stack installer
│       ├── InstallsInertiaStacks.php # Vue/React installers
│       └── InstallsLivewireStack.php # Livewire installer
├── stubs/                          # Template files за всеки stack
│   ├── api/                        # API stack templates
│   │   ├── app/Http/Controllers/Auth/
│   │   ├── app/Http/Middleware/
│   │   ├── config/
│   │   ├── routes/
│   │   └── tests/
│   ├── default/                    # Blade stack templates
│   │   ├── app/Http/Controllers/
│   │   ├── resources/views/
│   │   ├── routes/
│   │   └── tests/
│   └── inertia-common/             # Споделени Inertia файлове
│       ├── app/Http/Controllers/
│       ├── app/Http/Middleware/
│       └── routes/
```

```

├── inertia-react/           # React (JS) templates
│   └── resources/js/
├── inertia-react-ts/       # React (TS) templates
│   └── resources/js/
├── inertia-vue/           # Vue (JS) templates
│   └── resources/js/
├── inertia-vue-ts/        # Vue (TS) templates
│   └── resources/js/
├── livewire/              # Livewire Class API
│   └── resources/views/livewire/
├── livewire-functional/    # Livewire Functional API
│   └── resources/views/livewire/
└── livewire-common/        # Споделени Livewire файлове
    ├── app/Livewire/
    ├── resources/views/
    └── routes/

```

⚙️ Разширена Конфигурация

Package Manager Detection

Breeze автоматично засича и използва наличния package manager:

```

bash

# Приоритет на проверка:
1. pnpm (ако съществува pnpm-lock.yaml)
2. yarn (ако съществува yarn.lock)
3. bun (ако съществува bun.lock/bun.lockb)
4. deno (ако съществува deno.lock)
5. npm (default fallback)

```

Custom Composer Path

```

bash

php artisan breeze:install vue --composer=/usr/local/bin/composer

```

Environment Variables

След инсталация на API stack, `.env` файлът се обновява:

```
env

APP_URL=http://localhost:8000
FRONTEND_URL=http://localhost:3000
```

Middleware Configuration

Breeze автоматично конфигурира middleware в `bootstrap/app.php`:

```
php

->withMiddleware(function (Middleware $middleware) {
    $middleware->web(append: [
        \App\Http\Middleware\HandleInertiaRequests::class,
    ]);

    $middleware->alias([
        'verified' => \App\Http\Middleware\EnsureEmailsVerified::class,
    ]);
})
```

Тестване

Pest (препоръчително)

```
bash

# Инсталирай с Pest
php artisan breeze:install blade --pest

# Стартирай тестовете
./vendor/bin/pest
```

PHPUnit

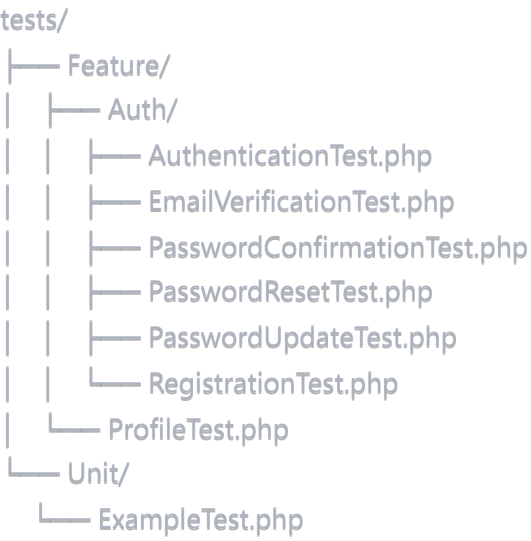
```
bash

# Инсталирай с PHPUnit (по подразбиране)
php artisan breeze:install blade

# Стартирай тестовете
./vendor/bin/phpunit
```

Налични Test Suites

bash



Example Test







php

```
// tests/Feature/Auth/RegistrationTest.php
test('new users can register', function () {
    $response = $this->post('/register', [
        'name' => 'Test User',
        'email' => 'test@example.com',
        'password' => 'password',
        'password_confirmation' => 'password',
    ]);

    $this->assertAuthenticated();
    $response->assertRedirect('/dashboard');
});
```

Security

Текущи Security Features

-  CSRF Protection
-  Password Hashing (bcrypt)
-  Email Verification
-  Rate Limiting
-  SQL Injection Protection
-  XSS Protection

-  Sanctum Token Authentication (API)

Препоръки за Production

1. Environment Variables

```
env

APP_ENV=production
APP_DEBUG=false
APP_KEY=base64:random-32-char-string
```

2. HTTPS

```
php

// app/Providers/AppServiceProvider.php
if ($this->app->environment('production')) {
    URL::forceScheme('https');
}
```

3. Rate Limiting

```
php

// routes/web.php
Route::middleware(['throttle:60,1'])->group(function () {
    // Your routes
});
```

4. Content Security Policy

```
php

// config/sanctum.php
'stateful' => explode(',', env('SANCTUM_STATEFUL_DOMAINS', 'localhost,127.0.0.1')),
```

Докладване на Security Issues

Ако откриете security уязвимост, моля **НЕ** отваряйте публичен issue. Вместо това изпратете email на security@laravel.com.

Contributing

Благодарим че разглеждате възможността да допринесете за Laravel Breeze!

Development Setup

```
bash

# Clone repository
git clone https://github.com/laravel/breeze.git
cd breeze

# Install dependencies
composer install

# Link локално за testing
composer link /path/to/your-test-laravel-app
```

Pull Request Guidelines

- ✔ Следвайте PSR-12 coding standard
- ✔ Добавете тестове за нова функционалност
- ✔ Обновете документацията
- ✔ Един PR = един feature/fix
- ✔ Пишете ясни commit messages

Coding Standards

```
bash

# Run PHP CS Fixer
./vendor/bin/php-cs-fixer fix

# Run tests
./vendor/bin/pest
```



Stack Comparison

Feature	Blade	Livewire	Vue/React	API
Server-side rendering	✔	✔	✗ (SSR опция)	N/A
SPA experience	✗	⚠	✔	N/A
JavaScript required	Minimal	Minimal	Heavy	N/A
SEO friendly	✔	✔	⚠	N/A
Learning curve	Low	Low	Medium	Low
Real-time updates	✗	✔	✔	✔
Mobile app support	✗	✗	⚠	✔

Roadmap

Планирани Features

- ☐ OAuth integration templates
- ☐ Two-factor authentication
- ☐ API rate limiting dashboard
- ☐ Role-based permissions starter
- ☐ Notification preferences UI

Допълнителни Ресурси

Официална Документация

- [Laravel Documentation](#)
- [Breeze Documentation](#)
- [Inertia.js Documentation](#)
- [Livewire Documentation](#)

Video Tutorials

- [Laracasts - Breeze Basics](#)
- [Laravel Daily - Breeze Tutorial](#)

Community

- [Laravel Forums](#)
- [Laravel Discord](#)
- [Laravel Reddit](#)

Changelog

Version 2.x

- ✨ Added Livewire Functional API support
- ✨ TypeScript support for Inertia stacks
- ✨ ESLint + Prettier integration
- ✨ Dark mode support
- ✨ SSR support for Inertia stacks
- ⚡ Improved installation performance
- 🐛 Various bug fixes

Version 1.x

- Initial release with Blade, Livewire, Vue, React, and API stacks

Credits

Laravel Breeze е създаден и поддържан от:

- **Taylor Otwell** - Creator of Laravel
- **The Laravel Team** - Core maintainers
- **Community Contributors** - Bug fixes and improvements

Лиценз

Laravel Breeze е open-source software лицензиран под MIT license.

<p align="center"> Made with ❤️ by the Laravel Team </p> <p align="center"> Laravel • GitHub • Twitter </p>