

# Introdução ao Laravel Breeze

## O que é o Laravel Breeze?

O Laravel Breeze é uma implementação **minimalista e simples** do sistema de autenticação do Laravel, fornecendo uma estrutura funcional e pronta para uso.

Ele utiliza o **Blade**, o mecanismo de templates nativo do Laravel, para criar todas as interfaces de autenticação necessárias.

## Características principais:



Sistema completo de login e registro



Recuperação de senha



Verificação de e-mail



Proteção de rotas com middleware

### Pré-requisito:

Projeto Laravel 12 já criado e banco de dados configurado no arquivo .env

# Instalação do Laravel Breeze

1

## Instalar o pacote Laravel Breeze

Adicione o Laravel Breeze como uma dependência de desenvolvimento ao seu projeto:

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

Terminal

### Dica:

O Laravel Breeze é uma solução leve e minimalista para autenticação, ideal para projetos que precisam de um sistema simples e personalizável.

Diferente do Laravel Jetstream, o Breeze não inclui recursos avançados como autenticação de dois fatores ou gerenciamento de equipes.



# Geração dos Arquivos de Autenticação

2

## Gerar os arquivos de autenticação

Execute o comando Artisan para instalar o Breeze com o stack Blade:

```
php artisan breeze:install blade
```

Terminal

## Opções de stack disponíveis:

### blade

Utiliza o sistema de templates Blade do Laravel

### react

Utiliza React com Inertia.js

### vue

Utiliza Vue.js com Inertia.js

### api

Configuração para autenticação via API tokens

# Componentes Adicionados pelo Breeze (1/2)

O Laravel Breeze adiciona diversos componentes ao seu projeto para facilitar a implementação de autenticação:

## Views de Autenticação

Telas de login, registro e redefinição de senha

```
resources/views/auth/
```

## Layout Base

Template principal para todas as páginas

```
resources/views/layouts/app.blade.php
```

## Rotas de Autenticação

Definições de rotas para login, registro, etc.

```
routes/auth.php
```

# Componentes Adicionados pelo Breeze (2/2)

## Controladores de Autenticação

Classes que gerenciam o fluxo de autenticação

```
app/Http/Controllers/Auth/
```

## Componentes Blade Reutilizáveis

Inputs, labels e outros elementos de formulário

```
resources/views/components/
```

## Dashboard Inicial

Página inicial para usuários autenticados

```
resources/views/dashboard.blade.php
```

### Importante:

Todos estes componentes são totalmente personalizáveis e podem ser adaptados às necessidades específicas do seu projeto.



# Compilando os Assets

Após a instalação do Laravel Breeze, é necessário compilar os arquivos CSS e JavaScript para garantir o funcionamento correto da interface:

```
npm install  
npm run dev
```

Terminal



**npm install**

Instala dependências



**Vite**

Processa arquivos



**Assets**

CSS e JS compilados



**Pronto**

Interface funcional

# Criando as Tabelas no Banco de Dados

O Laravel já vem com migrations padrão para o sistema de autenticação. Para criar as tabelas, execute:

```
php artisan migrate
```

Terminal

## Tabelas criadas pelas migrations:

Tabela	Finalidade
<b>users</b>	Armazenar dados dos usuários
<b>password_reset_tokens</b>	Tokens de redefinição de senha
<b>personal_access_tokens</b>	Autenticação via API (tokens)

### ⚠ Importante:

Verifique se o banco de dados no arquivo `.env` existe e está com as credenciais corretas.

# Iniciando o Servidor

## Iniciar o servidor de desenvolvimento



Execute o  
comando



Servidor inicia



Acesse no  
navegador

```
php artisan serve
```

Terminal

O servidor estará disponível no endereço:

 <http://127.0.0.1:8000>

✓ Verificação:



# Rotas de Autenticação Disponíveis

Após a instalação do Laravel Breeze, as seguintes rotas de autenticação estarão disponíveis:



**/register**

Tela de cadastro de novos usuários



**/login**

Tela de login para usuários existentes



**/forgot-password**

Redefinição de senha para usuários



**/dashboard**


Rota protegida, acessível apenas após login

# Estrutura de Arquivos Criada (1/2)

## Views e Templates

### **resources/views/**

#### **auth/**

 login.blade.php

 register.blade.php

 forgot-password.blade.php


 reset-password.blade.php

 dashboard.blade.php

#### **layouts/**

 app.blade.php

#### **components/**

 button.blade.php

# Estrutura de Arquivos Criada (2/2)

## Controladores e Rotas

### **routes/**


 web.php

 auth.php

### **app/Http/Controllers/Auth/**

 AuthenticatedSessionController.php

 RegisteredUserController.php

 PasswordResetLinkController.php

 NewPasswordController.php

### **Dica:**

Todos estes arquivos podem ser modificados livremente para atender às necessidades específicas do seu projeto. O Laravel Breeze fornece apenas um ponto de partida.



# Protegendo Rotas com Autenticação

## Middleware de Autenticação

O middleware **auth** verifica se o usuário está autenticado antes de permitir o acesso à rota.

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

PHP

## Grupo de Rotas Protegidas:

```
Route::middleware(['auth'])->group(function () {  
    Route::get('/perfil', function () {  
        return view('profile');  
    });  
});
```

PHP

**Dica:** Combine múltiplos middlewares para proteção extra: **['auth', 'verified']**

# Implementando o Logout

## Formulário de Logout

O Laravel Breeze já configura a funcionalidade de logout. Para implementá-la, crie um formulário POST:

```
<form method="POST" action="{{ route('logout') }}">
  @csrf
  <button type="submit" class="btn">Sair</button>
</form>
```

HTML



Clique no  
botão



POST para  
/logout



Sessão  
encerrada



**Segurança:** O Laravel utiliza o token CSRF (@csrf) para proteger contra ataques de falsificação de requisição.

# Customizando as Views de Autenticação (1/2)

As views de autenticação geradas pelo Laravel Breeze podem ser facilmente personalizadas para atender às necessidades específicas do seu projeto.

## Personalização Visual

Modifique o HTML e CSS das views para adequar ao design do seu projeto:

```
resources/views/auth/
```

## Adicionando Novos Campos

Para adicionar campos como nome completo, telefone, CPF, etc.:

- Crie uma migration para adicionar os campos à tabela users
- Atualize o formulário em `register.blade.php`
- Modifique o `RegisteredUserController.php`

# Customizando as Views de Autenticação (2/2)