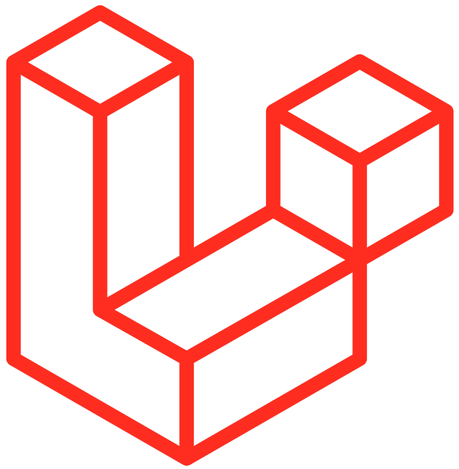
Programação para a WEB - servidor (server-side)

Sérgio da Silva Nogueira

PHP

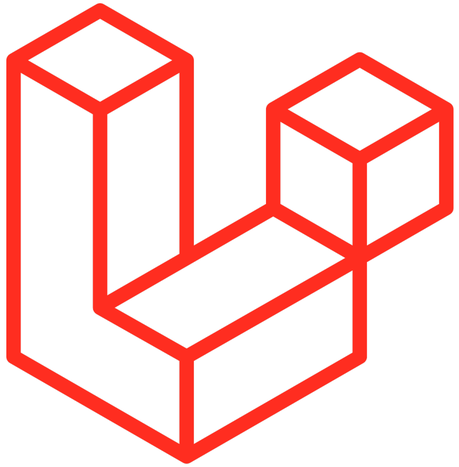
Middleware



Middleware é software que reside entre um sistema operativo e as aplicações que são executadas no mesmo. Funcionando essencialmente como uma camada de tradução oculta, o middleware possibilita a comunicação e a gestão de dados para aplicações distribuídas. É, por vezes denominado de “plumbing” (canalização), uma vez que liga duas aplicações para que os dados e as bases de dados possam ser facilmente passados pelo “pipe” (cano). A utilização de middleware permite aos utilizadores efetuarem pedidos, como submeter formulários num browser ou permitir que o servidor Web devolva páginas Web dinâmicas com base no perfil de um utilizador.

P H P

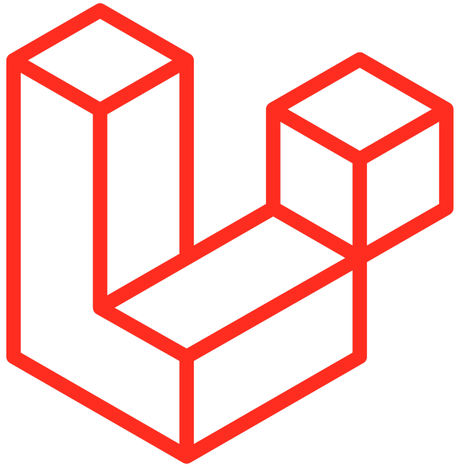
Middleware



Os exemplos comuns de middleware incluem middleware de bases de dados, middleware de servidor de aplicações, middleware orientado para mensagens, middleware Web e monitores de processamento de transações. Normalmente, cada programa fornece serviços de mensagens para que aplicações diferentes consigam comunicar através de estruturas de mensagens, como o protocolo SOAP (simple object access protocol), serviços Web, REST (representational state transfer) e JSON (JavaScript object notation). Enquanto todo o middleware efetua funções de comunicação, o tipo escolhido por uma empresa para utilização depende do serviço que está a ser utilizado e do tipo de informação que é necessário comunicar. Isto poderá incluir autenticação de segurança, gestão de transações, filas de mensagens, servidores de aplicações, servidores Web e diretórios. O middleware também pode ser utilizado para processamento distribuído com ações que estão a ocorrer em tempo real, em vez de enviar os dados para trás e para a frente.

P H P

Auth - Middleware

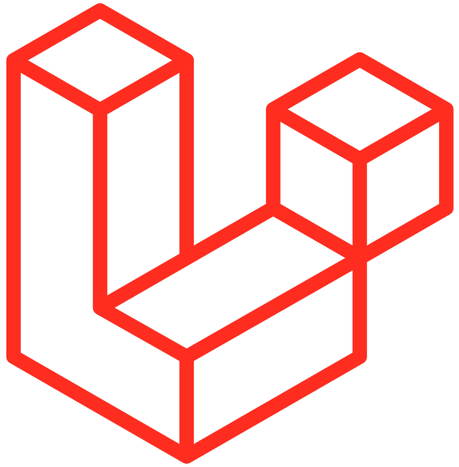


Para atribuir middleware a rotas específicas, primeiro é necessário atribuir ao middleware uma chave no ficheiro **app/Http/Kernel.php** de seu aplicativo. Por padrão, a propriedade $routeMiddleware desta classe contém entradas para o middleware incluídas no Laravel. E possível adicionar o próprio middleware a esta lista e atribuir a ele uma chave de sua escolha:

```php
/**
 * The application's route middleware.
 *
 * These middleware may be assigned to groups or used individually.
 *
 * @var array
 */
protected $routeMiddleware = [
    'auth' => \App\Http\Middleware\Authenticate::class,
    'auth.basic' => \Illuminate\Auth\Middleware\AuthenticateWithBasicAuth::class,
    'bindings' => \Illuminate\Routing\Middleware\SubstituteBindings::class,
    'cache.headers' => \Illuminate\Http\Middleware\SetCacheHeaders::class,
    'can' => \Illuminate\Auth\Middleware\Authorize::class,
    'guest' => \App\Http\Middleware\RedirectIfAuthenticated::class,
    'password.confirm' => \Illuminate\Auth\Middleware\RequirePassword::class,
    'signed' => \Illuminate\Routing\Middleware\ValidateSignature::class,
    'throttle' => \Illuminate\Routing\Middleware\ThrottleRequests::class,
    'verified' => \Illuminate\Auth\Middleware\EnsureEmailIsVerified::class,
];
```

P H P

Auth - Middleware

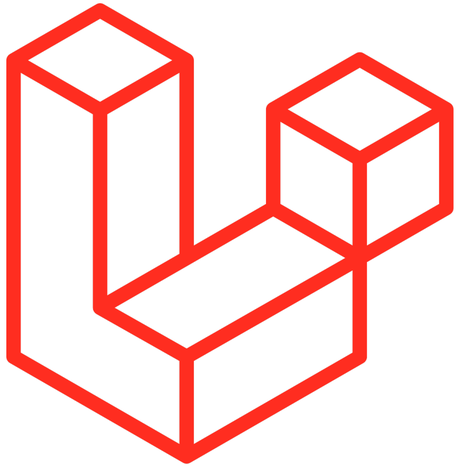


O Laravel fornece uma rota padrão para autenticação, para a utilizar devemos declarar no ficheiro de gestão de rotas qual o grupo de rotas que devem ser afetas por esse middleware. Vejamos o seguinte exemplo, no qual apenas permitimos apenas que as rotas de visualização de informação fiquem disponíveis para qualquer utilizar mesmo que não esteja autenticado

```
Auth::routes();
Route::get('/', 'PlayerController@index');
Route::get('/home', 'HomeController@index')->name('home');
Route::prefix('players')->group(function () {

    Route::get('', 'PlayerController@index');

    // Auth Middleware
    Route::group(['middleware' => 'auth'], function () {
        Route::get('create', 'PlayerController@create');
        Route::post('', 'PlayerController@store');
        Route::get('{player}/edit', 'PlayerController@edit');
        Route::put('{player}', 'PlayerController@update');
        Route::delete('{player}', 'PlayerController@destroy');
    });

    Route::get('{player}', 'PlayerController@show');
});
```

P H P

Auth - Middleware

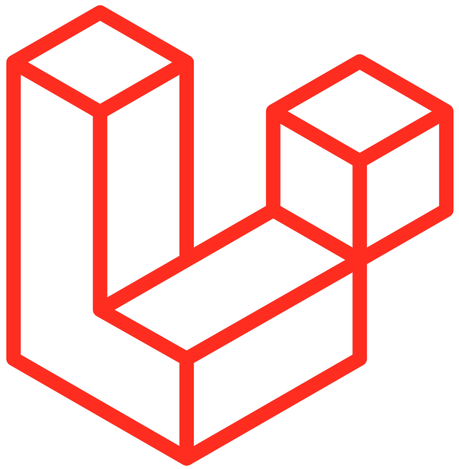


Após o login o user é redirecionado para a rota /home, podemos alterar este redirecionamento para “por exemplo” redirecionar o utilizar para a nossa rota de players. Para fazer essa alteração, devemos aceder ao ficheiro: **app/Providers/RouteServiceProvider.php** e alterar a constante Home:

```
/**
 * The path to the "home" route for your application.
 *
 * @var string
 */
public const HOME = '/players';
```

PHP

Auth - Middleware



Para finalizar, vamos esconder os links que só estão disponíveis quando o utilizador está autenticado utilizando as tags blade:

```
@auth
    // The user is authenticated...
@endauth

@guest
    // The user is not authenticated...
@endguest
```

Ou até mesmo:

```
@if (Auth::guest())

@else

@endif
```

PHP

Auth - Middleware

Após as alterações propostas, a nossa aplicação está mais próxima de uma solução de software a ser utilizada no mundo real. Para melhorar esta experiência vamos adiciona os links de **Login** e **Register** na nossa Nav.

```
@guest
    <li class="nav-item">
        <a class="nav-link" href="{{ route('login') }}">{{ __('Login') }}</a>
    </li>
    @if (Route::has('register'))
        <li class="nav-item">
            <a class="nav-link" href="{{ route('register') }}">{{ __('Register') }}</a>
        </li>
    @endif
@else
    <li class="nav-item dropdown">
        <a id="navbarDropdown" class="nav-link dropdown-toggle" href="#" role="button" data-toggle="dropdown" aria-haspopup="true" aria-expanded="false" v-pre>
            {{ Auth::user()->name }}
        </a>

        <div class="dropdown-menu dropdown-menu-right" aria-labelledby="navbarDropdown">
            <a class="dropdown-item" href="{{ route('logout') }}"
               onclick="event.preventDefault();
                             document.getElementById('logout-form').submit();">
                {{ __('Logout') }}
            </a>

            <form id="logout-form" action="{{ route('logout') }}" method="POST" class="d-none">
                @csrf
            </form>
        </div>
    </li>
@endguest
```

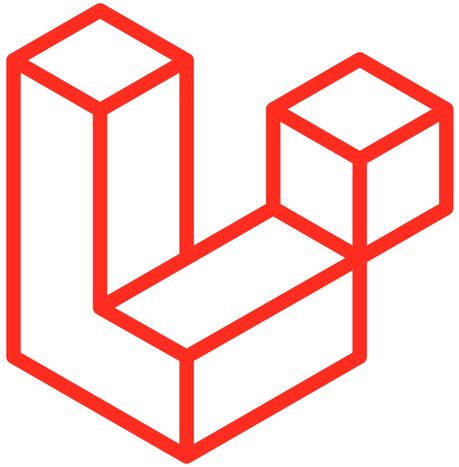
PHP

Auth - Middleware

```
<ul class="navbar-nav mr-auto">
    @guest
        <li class="nav-item">
            <a class="nav-link" href="{{ route('login') }}">{{ __('Login') }}</a>
        </li>
        @if (Route::has('register'))
            <li class="nav-item">
                <a class="nav-link" href="{{ route('register') }}">{{ __('Register') }}</a>
            </li>
        @endif
    @else
        <li class="nav-item dropdown">
            <a id="navbarDropdown" class="nav-link dropdown-toggle" href="#" role="button" data-toggle="dropdown" aria-haspopup="true" aria-expanded="false" v-pre>
                {{ Auth::user()->name }}
            </a>

            <div class="dropdown-menu dropdown-menu-right" aria-labelledby="navbarDropdown">
                <a class="dropdown-item" href="{{ route('logout') }}"
                   onclick="event.preventDefault();
                                                 document.getElementById('logout-form').submit();">
                    {{ __('Logout') }}
                </a>

                <form id="logout-form" action="{{ route('logout') }}" method="POST" class="d-none">
                    @csrf
                </form>
            </div>
        </li>
    @endguest
</ul>
<ul class="navbar-nav mr-auto">
    <li class="nav-item @if(Request::is('players')) active @endif">
        <a class="nav-link" href="{{ url('players') }}">Players List <span class="sr-only">(current)</span></a>
    </li>
    <li class="nav-item @if(Request::is('players/create')) active @endif">
        <a class="nav-link" href="{{ url('players/create') }}">Add Player</a>
    </li>
</ul>
```

P H P

Auth - Middleware



Após o login o user é redirecionado para a rota /home, podemos alterar este redirecionamento para “por exemplo” redirecionar o utilizar para a nossa rota de players. Para fazer essa alteração, devemos aceder ao ficheiro: **app/Providers/RouteServiceProvider.php** e alterar a constante Home:

```
/**
 * The path to the "home" route for your application.
 *
 * @var string
 */
public const HOME = ‘/players';
```