**9 Tips to Set Up Multiple Authentication in Laravel**

Published Aug 06, 2017Last updated Aug 10, 2017

**What was I doing?**

I was building a system that required users, doctors, and admins to register and have different authentications. I needed to set up different logins and tables for them and I wanted to make use of the Laravel App\User. I got access to the default Eloquent authentication driver and started digging. I came across a few tips that got me on the road to success by setting up custom guards.

Here's what I did:

**How I Approached Setting Up Multiple Authentications**

1. First off, you should have Laravel auth set up. If you don't, you can run:  
   php artisan make:auth
2. php artisan migrate

This will scaffold the entire authentication system.

1. Create the admin and doctors model and migration.  
   php artisan make:model Models/Admins -m

This creates a migration file from the admins model.(/app/Models/)

To make it easier, you can just duplicate the 'create\_users\_table' migration file but remember to make the necessary modifications:

The Admin Model:

**<?php**

**namespace** **App**\**Models**;

**use** **Illuminate**\**Foundation**\**Auth**\**User** **as** **Authenticatable**;

**class** **Admin** **extends** **Authenticatable**

{

**protected** $guard = 'admin';

**protected** $fillable = [

'firstname', 'midname', 'lastname', 'email', 'address', 'password',

];

**protected** $hidden = [

'password', 'remember\_token',

];

}

The Admins Migration file:

**<?php**

**use** **Illuminate**\**Support**\**Facades**\**Schema**;

**use** **Illuminate**\**Database**\**Schema**\**Blueprint**;

**use** **Illuminate**\**Database**\**Migrations**\**Migration**;

**class** **CreateAdminsTable** **extends** **Migration**

{

**public** **function** **up**()

{

Schema::create('admins', **function** (Blueprint $table) {

$table->increments('id');

$table->string('firstname');

$table->string('midname');

$table->string('lastname');

$table->string('email')->unique();

$table->string('address')->nullable();

$table->string('password');

$table->rememberToken();

$table->timestamps();

});

}

**public** **function** **down**()

{

Schema::dropIfExists('admins');

}

}

1. In the config/auth.php file, set up the custom guards and providers for doctors, users, and admins.

'guards' =>

[

'web' => [ 'driver' => 'session', 'provider' => 'users', ],

'api' => [ 'driver' => 'token', 'provider' => 'users', ],

'admin' => [ 'driver' => 'session', 'provider' => 'admins',],

'admin-api' => [ 'driver' => 'token', 'provider' => 'admins', ],

'doctor' => [ 'driver' => 'session', 'provider' => 'doctors', ],

'doctor-api' => [ 'driver' => 'token', 'provider' => 'doctors', ],

],

'providers' =>

[

'users' => [ 'driver' => 'eloquent', 'model' => App\Models\User::class, ],

'admins' => [ 'driver' => 'eloquent', 'model' => App\Models\Admin::class, ],

'doctors' => [ 'driver' => 'eloquent', 'model' => App\Models\Doctor::class, ],

],

1. Now we have our guards set up, we'll set up our admin login view.

<form id="sign\_in\_adm" method="POST" action="{{ route('admin.login.submit') }}">

{{ csrf\_field() }}

<div **class**="input-group"> <span **class**="input-group-addon"> <i **class**="zmdi zmdi-account"></i> </span>

<div **class**="form-line">

<input **type**="email" **class**="form-control" name="email" placeholder="Email Address" value="{{ old('email') }}" required autofocus>

</div>

**@if** ($errors->has('email'))

<span **class**="text-danger"><strong>{{ $errors->first('email') }}</strong></span>

**@endif**

</div>

<div **class**="input-group"> <span **class**="input-group-addon"> <i **class**="zmdi zmdi-lock"></i> </span>

<div **class**="form-line">

<input **type**="password" **class**="form-control" name="password" placeholder="Password" required>

</div>

</div>

<div>

<div **class**="">

<input **type**="checkbox" name="remember" {{ old('remember') ? 'checked' : '' }} id="rememberme" **class**="filled-in chk-col-pink">

<label **for**="rememberme">**Remember** **Me**</label>

</div>

<div **class**="text-center">

<button **type**="submit" **class**="btn btn-raised waves-effect g-bg-cyan">**SIGN** **IN**</button>

</div>

</div>

</form>

</div>

The admin login is almost the same as the user login, but we'll change the action of the form to a route, which we will set up later in our routes file.

1. We have our login views set up, so we can create the AdminController now  
   php artisan make:controller Auth/AdminController

**<?php**

**namespace** **App**\**Http**\**Controllers**;

**use** **Illuminate**\**Http**\**Request**;

**class** **AdminController** **extends** **Controller**

{

**public** **function** **\_\_construct**()

{

**$this**->middleware('auth:admin');

}

**public** **function** **index**()

{

**return** view('admin');

}

}

1. And the AdminLoginController,  
   php artisan make:controller auth/AdminLoginController

**<?php**

**namespace** **App**\**Http**\**Controllers**\**Auth**;

**use** **Illuminate**\**Http**\**Request**;

**use** **App**\**Http**\**Controllers**\**Controller**;

**use** **Auth**;

**use** **Route**;

**class** **AdminLoginController** **extends** **Controller**

{

**public** **function** **\_\_construct**()

{

**$this**->middleware('guest:admin', ['except' => ['logout']]);

}

**public** **function** **showLoginForm**()

{

**return** view('auth.admin\_login');

}

**public** **function** **login**(Request $request)

{

**$this**->validate($request, [

'email' => 'required|email',

'password' => 'required|min:6'

]);

**if** (Auth::guard('admin')->attempt(['email' => $request->email, 'password' => $request->password], $request->remember)) {

*// if successful, then redirect to their intended location*

**return** redirect()->intended(route('admin.dashboard'));

}

**return** redirect()->back()->withInput($request->only('email', 'remember'));

}

**public** **function** **logout**()

{

Auth::guard('admin')->logout();

**return** redirect('/admin');

}

}

1. Now we will set our routes; routes/web.php

Route::get('/', **function** () {

**return** view('welcome');

});

Auth::routes();

Route::get('/home', 'HomeController@index');

Route::prefix('admin')->group(**function**() {

Route::get('/login',

'Auth\AdminLoginController@showLoginForm')->name('admin.login');

Route::post('/login', 'Auth\AdminLoginController@login')->name('admin.login.submit');

Route::get('logout/', 'Auth\AdminLoginController@logout')->name('admin.logout');

Route::get('/', 'AdminController@index')->name('admin.dashboard');

})

1. With the routes setup, we have to set up the different login views for the guards. We effect this in the app/Exceptions/Handler.php

**protected** **function** **unauthenticated**($request, AuthenticationException $exception)

{

**if** ($request->expectsJson()) {

**return** response()->json(['error' => 'Unauthenticated.'], 401);

}

$guard = array\_get($exception->guards(), 0);

**switch** ($guard) {

**case** 'admin':

$login = 'admin.login';

**break**;

**case** 'doctor':

$login = 'doctor.login';

**break**;

**default**:

$login = 'login';

**break**;

}

**return** redirect()->guest(route($login));

}

1. Finally, If the login is successful, redirect to the dashboard of the specific guard. This is done in the RedirectIfAuthenticated.php

**public** **function** **handle**($request, Closure $next, $guard = null)

{

**switch** ($guard) {

**case** 'admin':

**if** (Auth::guard($guard)->check()) {

**return** redirect()->route('admin.dashboard');

}

**break**;

**case** 'doctor':

**if** (Auth::guard($guard)->check()) {

**return** redirect()->route('doctor.dashboard');

}

**break**;

**case** 'consultant':

**if** (Auth::guard($guard)->check()) {

**return** redirect()->route('consultant.dashboard');

}

**break**;

**default**:

**if** (Auth::guard($guard)->check()) {

**return** redirect('/');

}

**break**;

}

**return** $next($request);

}