Practical 6: Laravel Authentication

In this lab, we will explore the concept of Authentication in Laravel. We will explore the concept upon all that we've learned before and on the case of authenticating a user and an administrator using guards. In this case, registered user will be called 'author' and administrator of the web application will be called 'admin'. We will create a new project for this practical purpose.

1 Create Laravel Application

Firstly, create a new Laravel web application using through CLI:

```
composer create-project laravel/laravel:8.* laravelAuth
```

Laravel's **laravel/ui** package provides a quick way to scaffold all of the routes and views needed for authentication. Change to the web application directory **(`\laravelAuth`)** and run the following commands to install the additional dependencies:

```
composer require laravel/ui --dev
php artisan ui vue --auth
npm install && npm run dev
```

If there is a notification saying "`Please run Mix again`**", then run the **`npm run dev`** command once again.

2 Configure Database

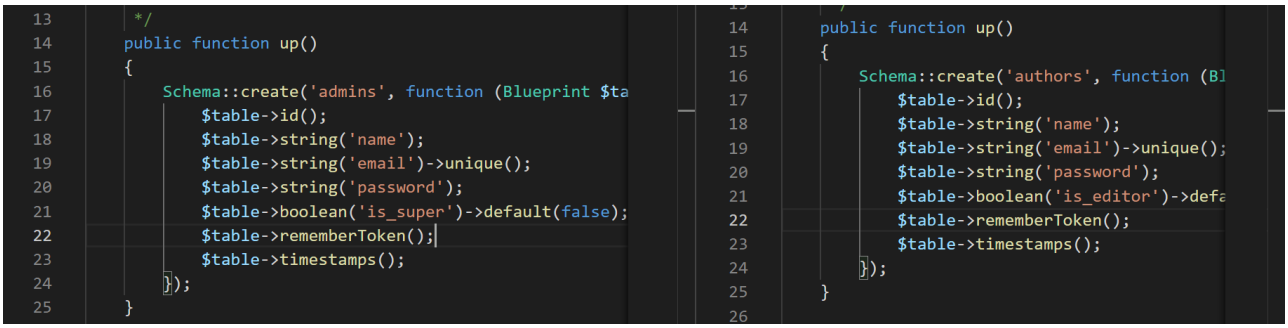
Then, make sure the database configurations in **`.env`** file is correct (double confirm with your MySQL host address and port). Make sure to create the database with appropriate collation in your phpMyAdmin.

2.1 Create Database Migration

Next, create database migration tables for both admins and authors (users) using CLI.

```
php artisan make:migration create_admins_table
php artisan make:migration create_authors_table
```

Then, edit the migrations of both admins and authors table as shown in Figure 1.



```
*/
public function up()
{
    Schema::create('admins', function (Blueprint $ta
        $table->id();
        $table->string('name');
        $table->string('email')->unique();
        $table->string('password');
        $table->boolean('is_super')->default(false);
        $table->rememberToken();
        $table->timestamps();
    });
}
```

```
public function up()
{
    Schema::create('authors', function (Bl
        $table->id();
        $table->string('name');
        $table->string('email')->unique();
        $table->string('password');
        $table->boolean('is_editor')->defa
        $table->rememberToken();
        $table->timestamps();
    });
}
```

Figure 1: Admins and Authors table migrations.

```
Schema::create('admins', function
(Blueprint $table) {
      $table->id();
      $table->string('name');
      $table->string('email')->unique();
      $table->string('password');
      $table->boolean('is_super')-
>default(false);
      $table->rememberToken();
      $table->timestamps();
 });
```

```
Schema::create('authors', function
(Blueprint $table) {
      $table->id();
      $table->string('name');
      $table->string('email')->unique();
      $table->string('password');
      $table->boolean('is_editor')-
>default(false);
      $table->rememberToken();
      $table->timestamps();
 });
```

Now, everything is set. Before migrating the database tables, it is worthy to note that **Laravel 5.4** made a change to the default database character set into "utf8mb4" which includes support for storing emojis. This only affects new applications and as long as the database is **MySQL v5.7.7** and higher, there is nothing to worry about. For those running MariaDB or older versions of MySQL, do proceed to **AppServiceProvider.php** in **App\Providers** to add the lines as shown in Figure 2.

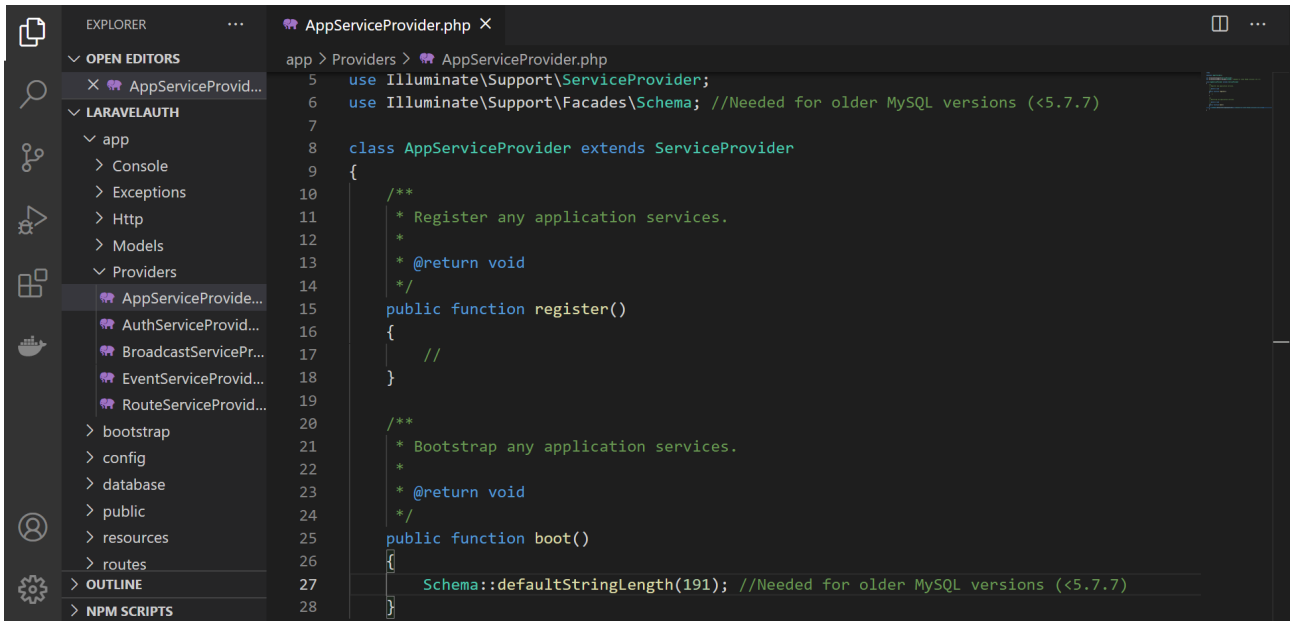


Figure 2: Configure default string length.

```
use Illuminate\Support\Facades\Schema;

public function boot()
```

```
{
    Schema::defaultStringLength(191);
}
```

Then, proceed to migrate the defined database tables through Artisan CLI:

```
php artisan migrate
```

3 Create Models

Now, we have two different tables; admins and authors. To use these different tables to authenticate, we need to define two models for them. Create the models using Artisan CLI.

```
php artisan make:model Admin
php artisan make:model Author
```

Then, edit both models to use guards for Authentication and extend the model to enable authentication to be done as shown in Figure 3.

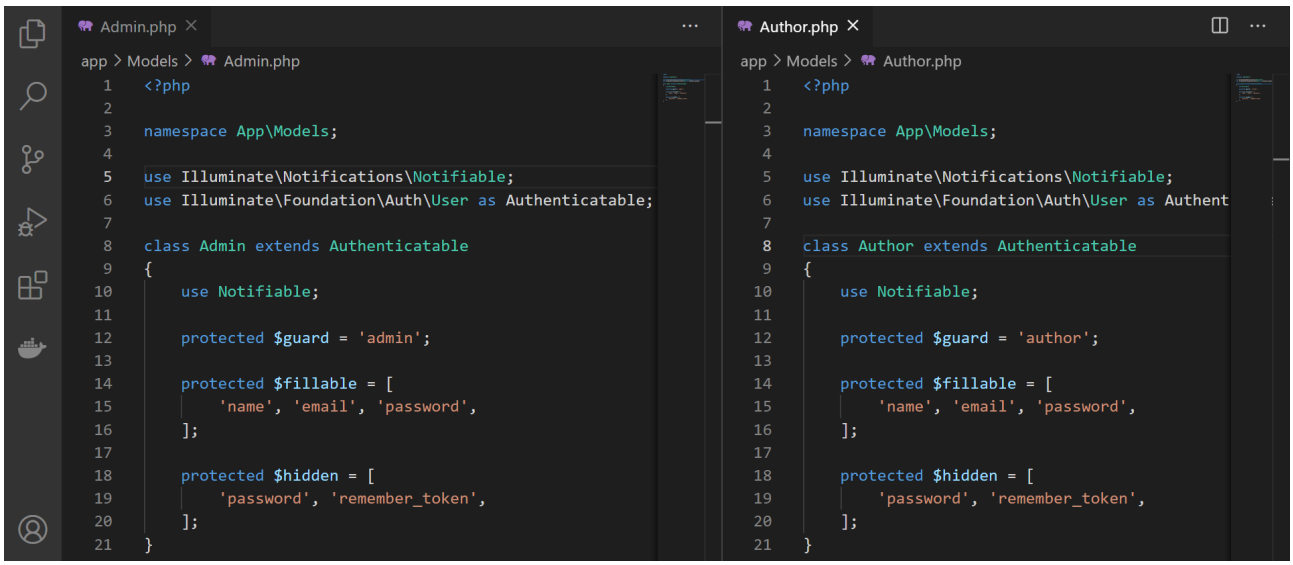


Figure 3: Authenticatable Models.

```
use
Illuminate\Notifications\Notifiable;
use Illuminate\Foundation\Auth\User as
Authenticatable;

class Admin extends Authenticatable
{
    use Notifiable;
    protected $guard = 'admin';
    protected $fillable = [
        'name', 'email', 'password',
    ];
    protected $hidden = [
        'password', 'remember_token',
    ];
}
```

```
use
Illuminate\Notifications\Notifiable;
use Illuminate\Foundation\Auth\User as
Authenticatable;

class Author extends Authenticatable
{
    use Notifiable;
    protected $guard = 'author';
    protected $fillable = [
        'name', 'email', 'password',
    ];
    protected $hidden = [
        'password', 'remember_token',
    ];
}
```

4 Define Guards

With the **guards** declared within the models, let's define/register the guards into authentication configuration file (**config\auth.php**) as shown in Figure 4.

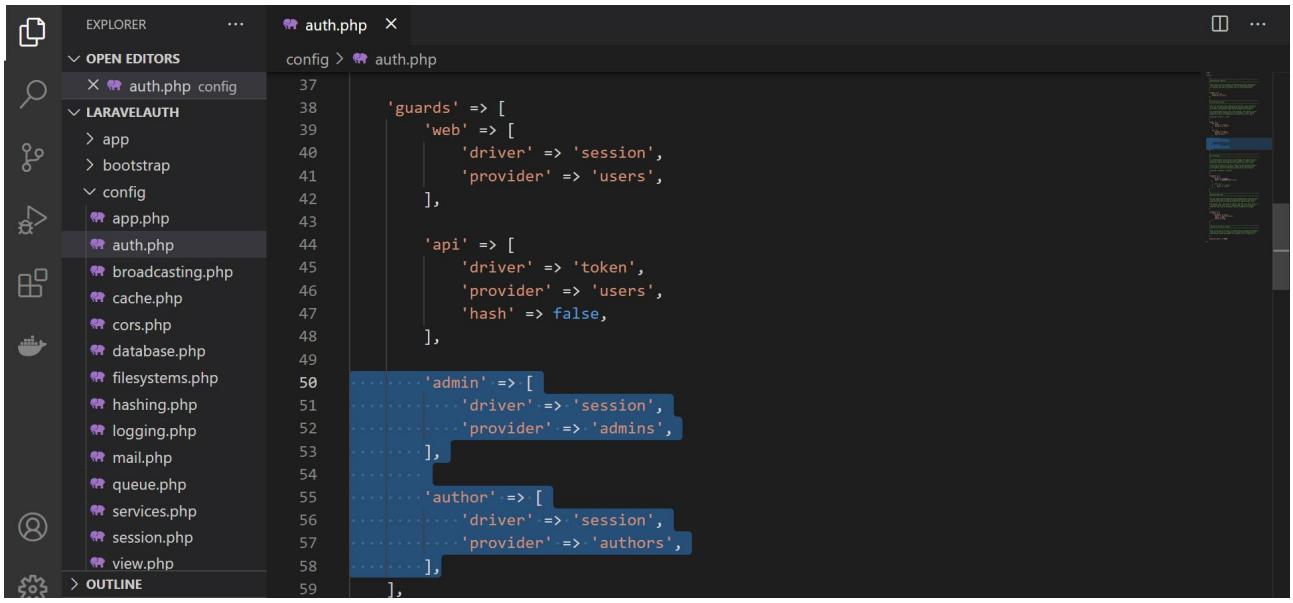


Figure 4: Register newly declared guards.

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

'author' => [
    'driver' => 'session',
    'provider' => 'authors',
],
```

Then, scroll down the file and set the **providers** for the guards as shown in Figure 5. These providers tell Laravel to use authentication or validation when the guard is used.

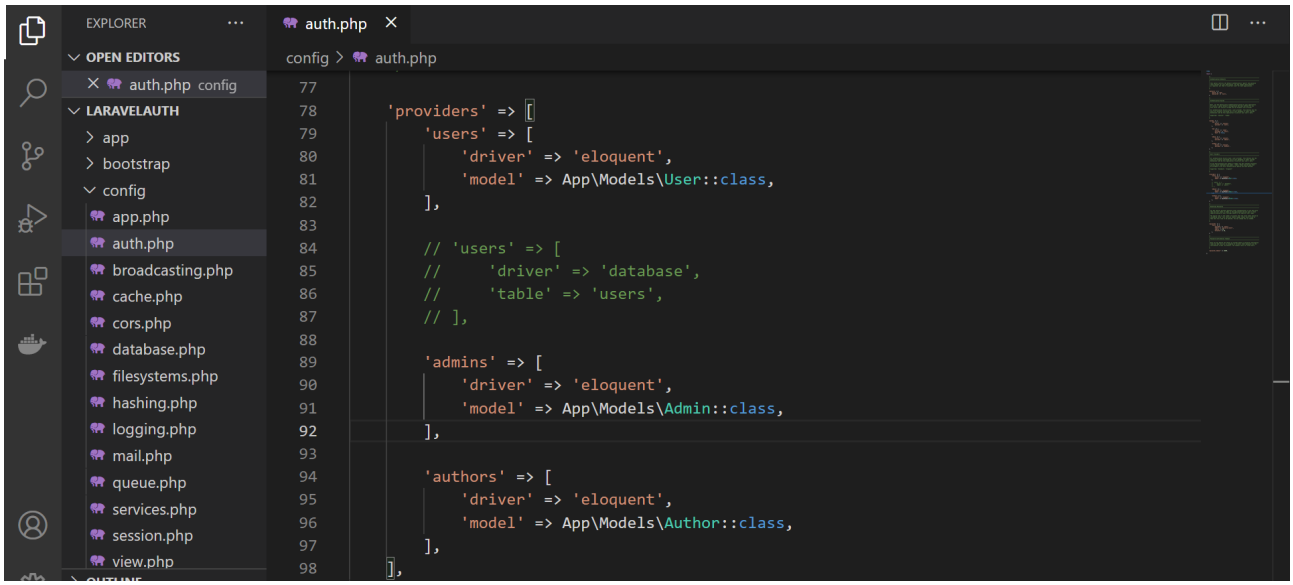


Figure 5: Define providers for the guards.

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

        'authors' => [
            'driver' => 'eloquent',
            'model' => App\Models\Author::class,
        ],
```

5 Define Authentication Logics / Controllers

Now, let's defined the authentication logics for different type of users (**admins** | **authors**) within Users **LoginController** in **App\Http\Controllers\Auth\LoginController.php** as shown below:

```
<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;
use Illuminate\Foundation\Auth\AuthenticatesUsers;
use Illuminate\Http\Request;
use Auth;

class LoginController extends Controller
{
    /*
    |--------------------------------------------------------------------------
    | Login Controller
    |--------------------------------------------------------------------------
    |
    | This controller handles authenticating users for the application and
    | redirecting them to your home screen. The controller uses a trait
    | to conveniently provide its functionality to your applications.
    |
    */

    use AuthenticatesUsers;

    /**
     * Where to redirect users after login.
     *
     * @var string
     */
    protected $redirectTo = '/home';

    /**
     * Create a new controller instance.
     *
     * @return void
     */
    public function __construct()
    {
        $this->middleware('guest')->except('logout');
        $this->middleware('guest:admin')->except('logout');
        $this->middleware('guest:author')->except('logout');
```

```
    }

    public function showAdminLoginForm()
    {
        return view('auth.login', ['url' => 'admin']);
    }

    public function adminLogin(Request $request)
    {
        $this->validate($request, [
            'email'   => 'required|email',
            'password' => 'required|min:6'
        ]);

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

            return redirect()->intended('/admin');
        }
        return back()->withInput($request->only('email', 'remember'));
    }

     public function showAuthorLoginForm()
    {
        return view('auth.login', ['url' => 'author']);
    }

    public function authorLogin(Request $request)
    {
        $this->validate($request, [
            'email'   => 'required|email',
            'password' => 'required|min:6'
        ]);

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

            return redirect()->intended('/author');
        }
        return back()->withInput($request->only('email', 'remember'));
    }
}
```

Then, modify the Users **RegistrationController App\Http\Controllers\Auth\RegisterController.php** as below:

```
<?php

namespace App\Http\Controllers\Auth;

use App\Models\User;
use App\Models\Admin;
use App\Models\Author;
use App\Http\Controllers\Controller;
use Illuminate\Support\Facades\Hash;
use Illuminate\Support\Facades\Validator;
use Illuminate\Foundation\Auth\RegistersUsers;
use Illuminate\Http\Request;

class RegisterController extends Controller
```

```
{
    /*
    |--------------------------------------------------------------------------
    | Register Controller
    |--------------------------------------------------------------------------
    |
    | This controller handles the registration of new users as well as their
    | validation and creation. By default this controller uses a trait to
    | provide this functionality without requiring any additional code.
    |
    */

    use RegistersUsers;

    /**
     * Where to redirect users after registration.
     *
     * @var string
     */
    protected $redirectTo = '/home';

    /**
     * Create a new controller instance.
     *
     * @return void
     */
    public function __construct()
    {
        $this->middleware('guest');
        $this->middleware('guest:admin');
        $this->middleware('guest:author');
    }

    /**
     * Get a validator for an incoming registration request.
     *
     * @param  array  $data
     * @return \Illuminate\Contracts\Validation\Validator
     */
    protected function validator(array $data)
    {
        return Validator::make($data, [
            'name' => ['required', 'string', 'max:255'],
            'email' => ['required', 'string', 'email', 'max:255',
'unique:users'],
            'password' => ['required', 'string', 'min:6', 'confirmed'],
        ]);
    }

    /**
     * @return \Illuminate\Contracts\View\Factory|\Illuminate\View\View
     */
    public function showAdminRegisterForm()
    {
        return view('auth.register', ['url' => 'admin']);
    }

    /**
     * @return \Illuminate\Contracts\View\Factory|\Illuminate\View\View
     */
    public function showAuthorRegisterForm()
    {
```

```
        return view('auth.register', ['url' => 'author']);
    }

    /**
     * @param array $data
     *
     * @return mixed
     */
    protected function create(array $data)
    {
        return User::create([
            'name' => $data['name'],
            'email' => $data['email'],
            'password' => Hash::make($data['password']),
        ]);
    }

    /**
     * @param Request $request
     *
     * @return \Illuminate\Http\RedirectResponse
     */
    protected function createAdmin(Request $request)
    {
        $this->validator($request->all())->validate();
        Admin::create([
            'name' => $request->name,
            'email' => $request->email,
            'password' => Hash::make($request->password),
        ]);
        return redirect()->intended('login/admin');
    }

    /**
     * @param Request $request
     *
     * @return \Illuminate\Http\RedirectResponse
     */
    protected function createAuthor(Request $request)
    {
        $this->validator($request->all())->validate();
        Author::create([
            'name' => $request->name,
            'email' => $request->email,
            'password' => Hash::make($request->password),
        ]);
        return redirect()->intended('login/author');
    }
}
```

6 Define Authentication Pages / Views

Laravel's **laravel/ui** package provides a quick way to scaffold all of the routes and views. Thus, the login page can be easily found with the directory: **resources\views\auth\login.blade.php**
We are checking if we passed a URL parameter to the page when we called it. If we did, we modify the forms action to use the URL parameter. We also modify the header of the form so that it shows the type of user based on login parameter. Let's modify the login form as below:

```
@extends('layouts.app')

@section('content')
<div class="container">
    <div class="row justify-content-center">
        <div class="col-md-8">
            <div class="card">
                <div class="card-header"> {{ isset($url) ? ucwords($url) : ""}}
{{ __('Login') }}</div>

                <div class="card-body">
                    @isset($url)
                    <form method="POST" action='{{ url("login/$url") }}' aria-
label="{{ __('Login') }}">
                    @else
                    <form method="POST" action="{{ route('login') }}" aria-
label="{{ __('Login') }}">
                    @endisset
                        @csrf

                        <div class="form-group row">
                            <label for="email" class="col-md-4 col-form-label
text-md-right">{{ __('E-Mail Address') }}</label>

                            <div class="col-md-6">
                                <input id="email" type="email" class="form-
control @error('email') is-invalid @enderror" name="email"
value="{{ old('email') }}" required autocomplete="email" autofocus>

                                @error('email')
                                    <span class="invalid-feedback"
role="alert">
                                        <strong>{{ $message }}</strong>
                                    </span>
                                @enderror
                            </div>
                        </div>

                        <div class="form-group row">
                            <label for="password" class="col-md-4 col-form-
label text-md-right">{{ __('Password') }}</label>

                            <div class="col-md-6">
                                <input id="password" type="password"
class="form-control @error('password') is-invalid @enderror" name="password"
required autocomplete="current-password">

                                @error('password')
                                    <span class="invalid-feedback"
role="alert">
                                        <strong>{{ $message }}</strong>
                                    </span>
```

```
                                @enderror
                            </div>
                        </div>

                        <div class="form-group row">
                            <div class="col-md-6 offset-md-4">
                                <div class="form-check">
                                    <input class="form-check-input"
type="checkbox" name="remember" id="remember" {{ old('remember') ? 'checked' :
'' }}>

                                    <label class="form-check-label"
for="remember">
                                        {{ __('Remember Me') }}
                                    </label>
                                </div>
                            </div>
                        </div>

                        <div class="form-group row mb-0">
                            <div class="col-md-8 offset-md-4">
                                <button type="submit" class="btn btn-primary">
                                    {{ __('Login') }}
                                </button>

                                @if (Route::has('password.request'))
                                    <a class="btn btn-link"
href="{{ route('password.request') }}">
                                        {{ __('Forgot Your Password?') }}
                                    </a>
                                @endif
                            </div>
                        </div>
                    </form>
                </div>
            </div>
        </div>
    </div>
</div>
@endsection
```

Then, replicate what was done for login page in register page (**resources\views\auth\register.blade.php**).

```
@extends('layouts.app')

@section('content')
<div class="container">
    <div class="row justify-content-center">
        <div class="col-md-8">
            <div class="card">
                <div class="card-header"> {{ isset($url) ? ucwords($url) : ""}}
{{ __('Register') }}</div>

                <div class="card-body">
                    @isset($url)
                    <form method="POST" action='{{ url("register/$url") }}'
aria-label="{{ __('Register') }}">
                    @else
                    <form method="POST" action="{{ route('register') }}" aria-
```

```
label="{{ __('Register') }}">
                    @endisset
                        @csrf



                        <div class="form-group row">
                            <label for="name" class="col-md-4 col-form-label
text-md-right">{{ __('Name') }}</label>

                            <div class="col-md-6">
                                <input id="name" type="text" class="form-
control @error('name') is-invalid @enderror" name="name"
value="{{ old('name') }}" required autocomplete="name" autofocus>

                                    @error('name')
                                        <span class="invalid-feedback"
role="alert">
                                            <strong>{{ $message }}</strong>
                                        </span>
                                    @enderror
                            </div>
                        </div>

                        <div class="form-group row">
                            <label for="email" class="col-md-4 col-form-label
text-md-right">{{ __('E-Mail Address') }}</label>

                            <div class="col-md-6">
                                <input id="email" type="email" class="form-
control @error('email') is-invalid @enderror" name="email"
value="{{ old('email') }}" required autocomplete="email">

                                    @error('email')
                                        <span class="invalid-feedback"
role="alert">
                                            <strong>{{ $message }}</strong>
                                        </span>
                                    @enderror
                            </div>
                        </div>

                        <div class="form-group row">
                            <label for="password" class="col-md-4 col-form-
label text-md-right">{{ __('Password') }}</label>

                            <div class="col-md-6">
                                <input id="password" type="password"
class="form-control @error('password') is-invalid @enderror" name="password"
required autocomplete="new-password">

                                    @error('password')
                                        <span class="invalid-feedback"
role="alert">
                                            <strong>{{ $message }}</strong>
                                        </span>
                                    @enderror
                            </div>
                        </div>

                        <div class="form-group row">
                            <label for="password-confirm" class="col-md-4 col-
```

```
form-label text-md-right">{{ __('Confirm Password') }}</label>

                            <div class="col-md-6">
                                <input id="password-confirm" type="password"
class="form-control" name="password_confirmation" required autocomplete="new-
password">
                            </div>
                        </div>

                        <div class="form-group row mb-0">
                            <div class="col-md-6 offset-md-4">
                                <button type="submit" class="btn btn-primary">
                                    {{ __('Register') }}
                                </button>
                            </div>
                        </div>
                    </form>
                </div>
            </div>
        </div>
    </div>
</div>
@endsection
```

7 Define Authenticated Pages for User Access

Now, let's create the pages that authenticated users will access.
Now that the login and register page are defined, let's make the pages the admins and authors will see when they are authenticated. Create the view files according to the followings:

```
resources/views/layouts/auth.blade.php
resources/views/admin.blade.php
resources/views/author.blade.php
resources/views/home.blade.php
```

Then, insert the following scripts into **auth.blade.php**

```
<!DOCTYPE html>
<html lang="{{ str_replace('_', '-', app()->getLocale()) }}">
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">

    <!-- CSRF Token -->
    <meta name="csrf-token" content="{{ csrf_token() }}">

    <title>{{ config('app.name', 'Laravel') }}</title>

    <!-- Scripts -->
    <script src="{{ asset('js/app.js') }}" defer></script>

    <!-- Fonts -->
    <link rel="dns-prefetch" href="https://fonts.gstatic.com">
    <link href="https://fonts.googleapis.com/css?family=Raleway:300,400,600"
rel="stylesheet" type="text/css">
```

```
    <!-- Styles -->
    <link href="{{ asset('css/app.css') }}" rel="stylesheet">
</head>
<body>
    <div id="app">
        <nav class="navbar navbar-expand-md navbar-light navbar-laravel">
            <div class="container">
                <a class="navbar-brand" href="{{ url('/') }}">
                    {{ config('app.name', 'Laravel') }}
                </a>
                <button class="navbar-toggler" type="button" data-
toggle="collapse" data-target="#navbarSupportedContent" aria-
controls="navbarSupportedContent" aria-expanded="false" aria-
label="{{ __('Toggle navigation') }}">
                    <span class="navbar-toggler-icon"></span>
                </button>

                <div class="collapse navbar-collapse"
id="navbarSupportedContent">
                    <!-- Left Side Of Navbar -->
                    <ul class="navbar-nav mr-auto">

                    </ul>

                    <!-- Right Side Of Navbar -->
                    <ul class="navbar-nav ml-auto">
                        <!-- Authentication Links -->
                       <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>
                                Hi There <span class="caret"></span>
                            </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" style="display: none;">
                                    @csrf
                                </form>
                            </div>
                        </li>
                    </ul>
                </div>
            </div>
        </nav>

        <main class="py-4">
            @yield('content')
        </main>
    </div>
</body>
</html>
```

Now, let's define the page that admin will enter (**admin.blade.php**) once authenticated, as the followings:

```
@extends('layouts.auth')

@section('content')
<div class="container">
    <div class="row justify-content-center">
        <div class="col-md-8">
            <div class="card">
                <div class="card-header">Dashboard</div>
                <div class="card-body">
                    You must be the priviledged administrator of this site!
                </div>
            </div>
        </div>
    </div>
</div>
@endsection
```

Then, define the page that author will enter (**author.blade.php**) once authenticated, as the followings:

```
@extends('layouts.auth')

@section('content')
<div class="container">
    <div class="row justify-content-center">
        <div class="col-md-8">
            <div class="card">
                <div class="card-header">Dashboard</div>
                <div class="card-body">
                    Hi there, awesome author for this site!
                </div>
            </div>
        </div>
    </div>
</div>
@endsection
```

Lastly, define the page that every regular user will visit or the homepage (**home.blade.php**) of the web application as follows:

```
@extends('layouts.auth')

@section('content')
<div class="container">
    <div class="row justify-content-center">
        <div class="col-md-8">
            <div class="card">
                <div class="card-header">Dashboard</div>

                <div class="card-body">
                     Welcome to YLLoo's Web Application.
                </div>
            </div>
        </div>
    </div>
</div>
@endsection
```

8 Define Routes

The web application is almost ready. Let's define the routes to access all the pages created so far. Open the **routes/web.php** file and modify according to the followings:

```
use App\Http\Controllers\Auth\LoginController;
use App\Http\Controllers\Auth\RegisterController;

Route::view('/', 'welcome');
Auth::routes();

Route::get('/login/admin', [LoginController::class, 'showAdminLoginForm']);
Route::get('/login/author', [LoginController::class,'showAuthorLoginForm']);
Route::get('/register/admin',
[RegisterController::class,'showAdminRegisterForm']);
Route::get('/register/author',
[RegisterController::class,'showAuthorRegisterForm']);

Route::post('/login/admin', [LoginController::class,'adminLogin']);
Route::post('/login/author', [LoginController::class,'authorLogin']);
Route::post('/register/admin', [RegisterController::class,'createAdmin']);
Route::post('/register/author', [RegisterController::class,'createAuthor']);

Route::group(['middleware' => 'auth:author'], function () {
    Route::view('/author', 'author');
});

Route::group(['middleware' => 'auth:admin'], function () {

    Route::view('/admin', 'admin');
});

Route::get('logout', [LoginController::class,'logout']);
//Route::get('/home', [App\Http\Controllers\HomeController::class, 'index'])-
>name('home');
```

The **RedirectIfAuthenticated** middleware in **app/Http/Controllers/Middleware/RedirectIfAuthenticated.php** receives the auth guard as a parameter. This middleware is triggered when a user try to visit any page meant for authenticated users. The middleware can then determine the type of authentication the user has and redirect them accordingly. Thus, the middleware need to be modified into the followings:

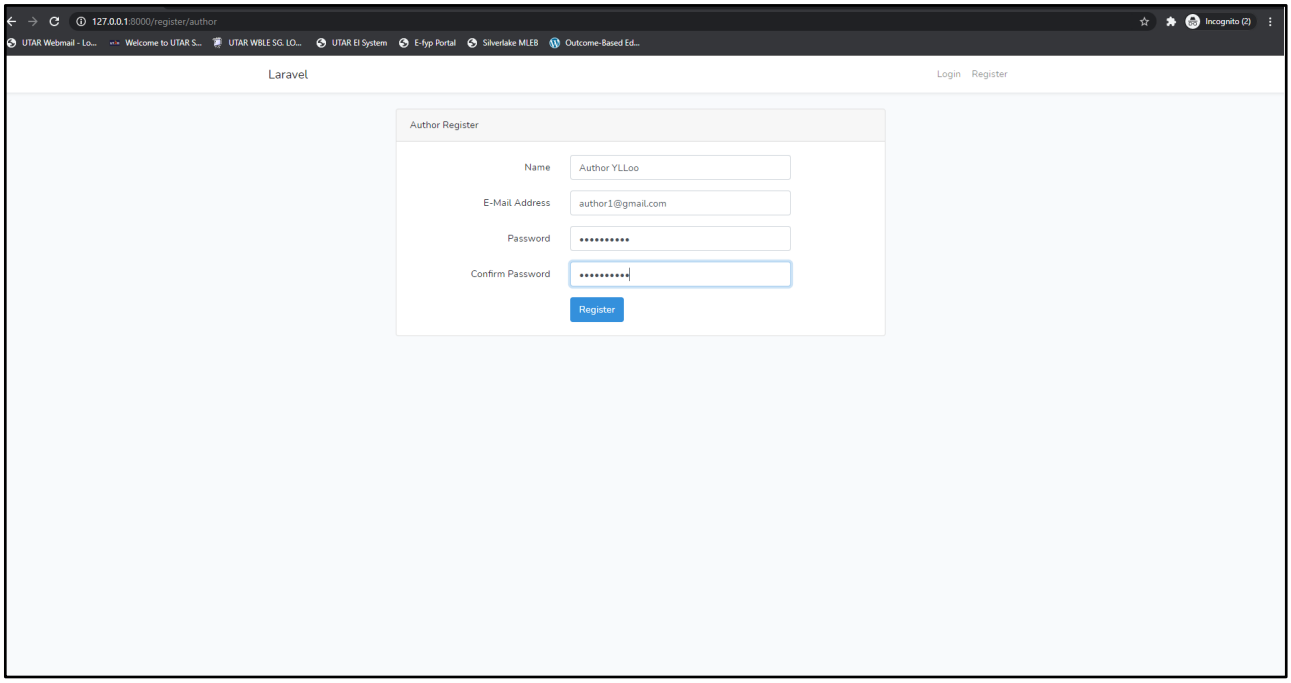
```
<?php

namespace App\Http\Middleware;

use Closure;
use Illuminate\Support\Facades\Auth;

class RedirectIfAuthenticated
{
    public function handle($request, Closure $next, $guard = null)
    {
        if ($guard == "admin" && Auth::guard($guard)->check()) {
            return redirect('/admin');
        }
        if ($guard == "author" && Auth::guard($guard)->check()) {
            return redirect('/author');
        }
        if (Auth::guard($guard)->check()) {
            return redirect('/home');
```

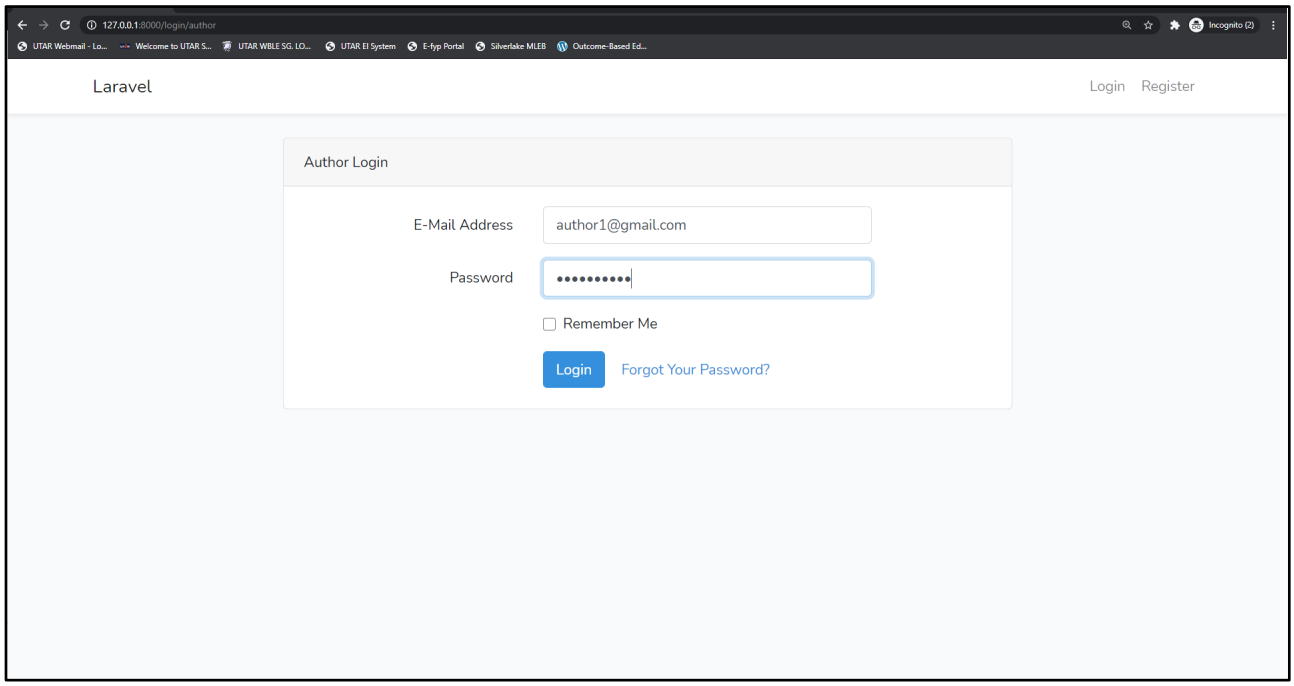
```
        }
        return $next($request);
    }
}
```

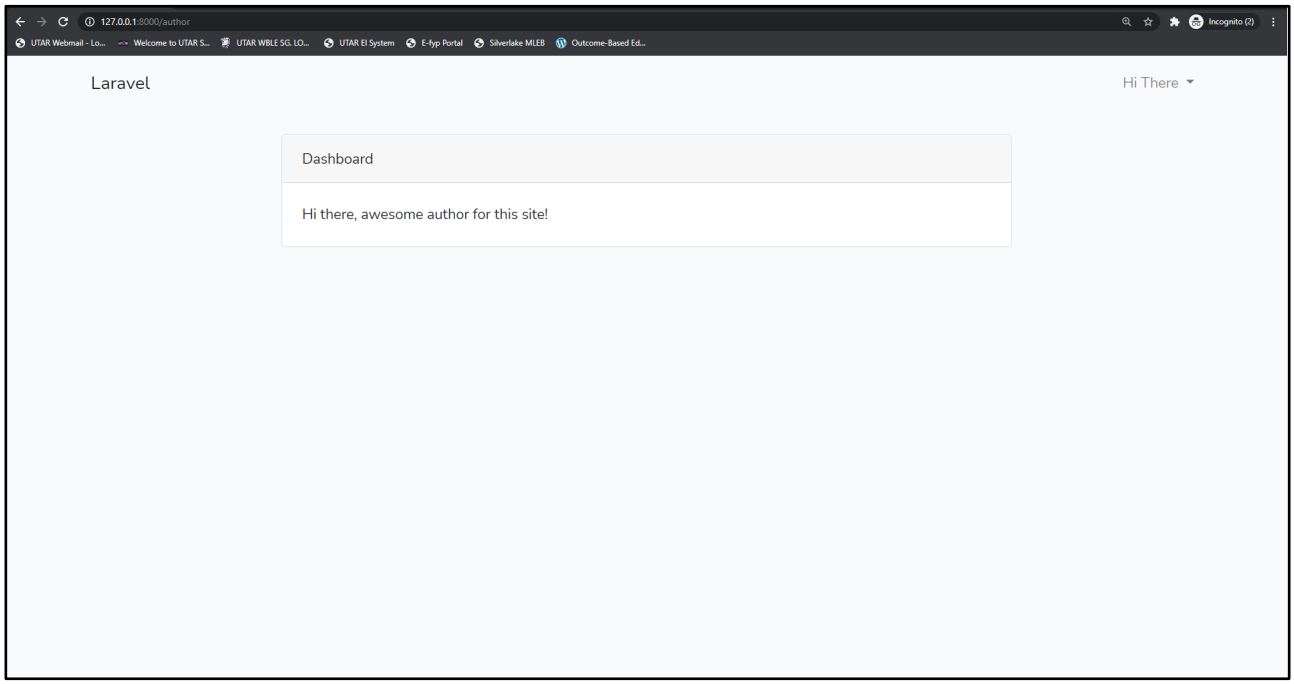
To ensure that when a user tries to visit **/author** then they are redirected to **/login/author** or the same for **/admin**, the exception handler need to be modified. Open the handler file in **app/Exceptions** and modify as the followings:

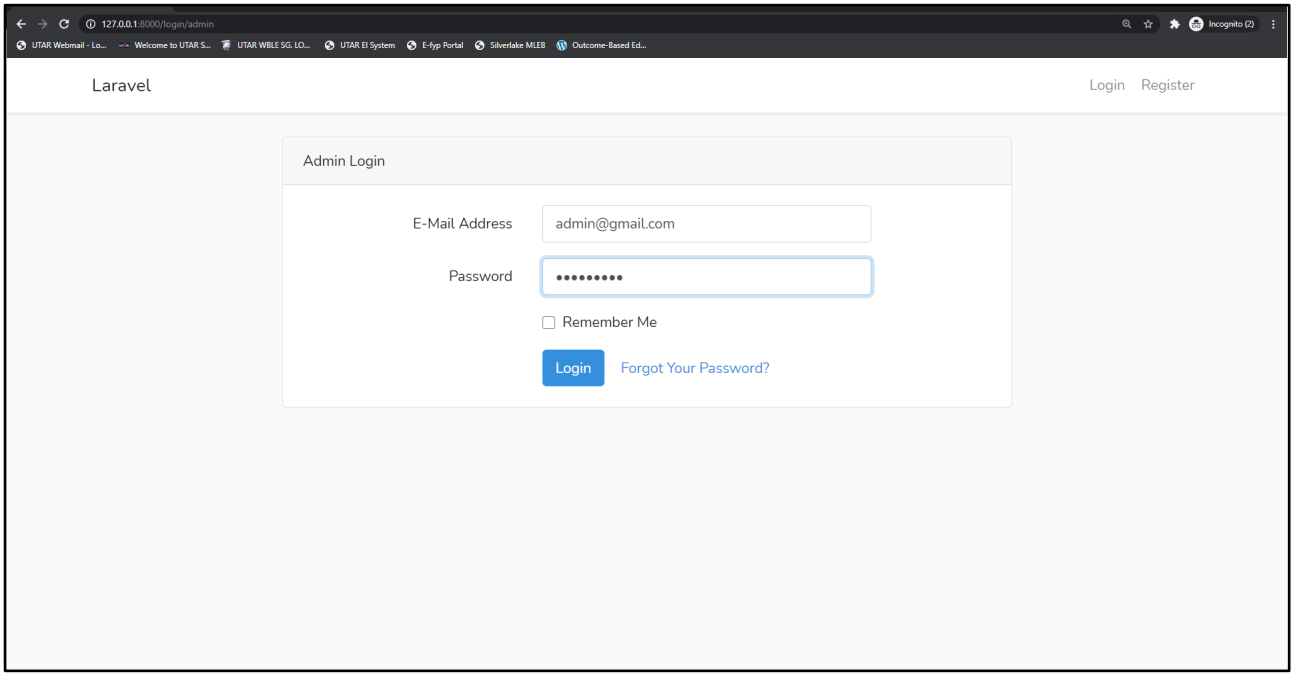
```
<?php

namespace App\Exceptions;

use Exception;
use Illuminate\Foundation\Exceptions\Handler as ExceptionHandler;
use Illuminate\Auth\AuthenticationException;
use Auth;
class Handler extends ExceptionHandler
{
    protected function unauthenticated($request, AuthenticationException
$exception)
    {
        if ($request->expectsJson()) {
            return response()->json(['error' => 'Unauthenticated.'], 401);
        }
        if ($request->is('admin') || $request->is('admin/*')) {
            return redirect()->guest('/login/admin');
        }
        if ($request->is('author') || $request->is('author/*')) {
            return redirect()->guest('/login/author');
        }
        return redirect()->guest(route('login'));
    }
}
```

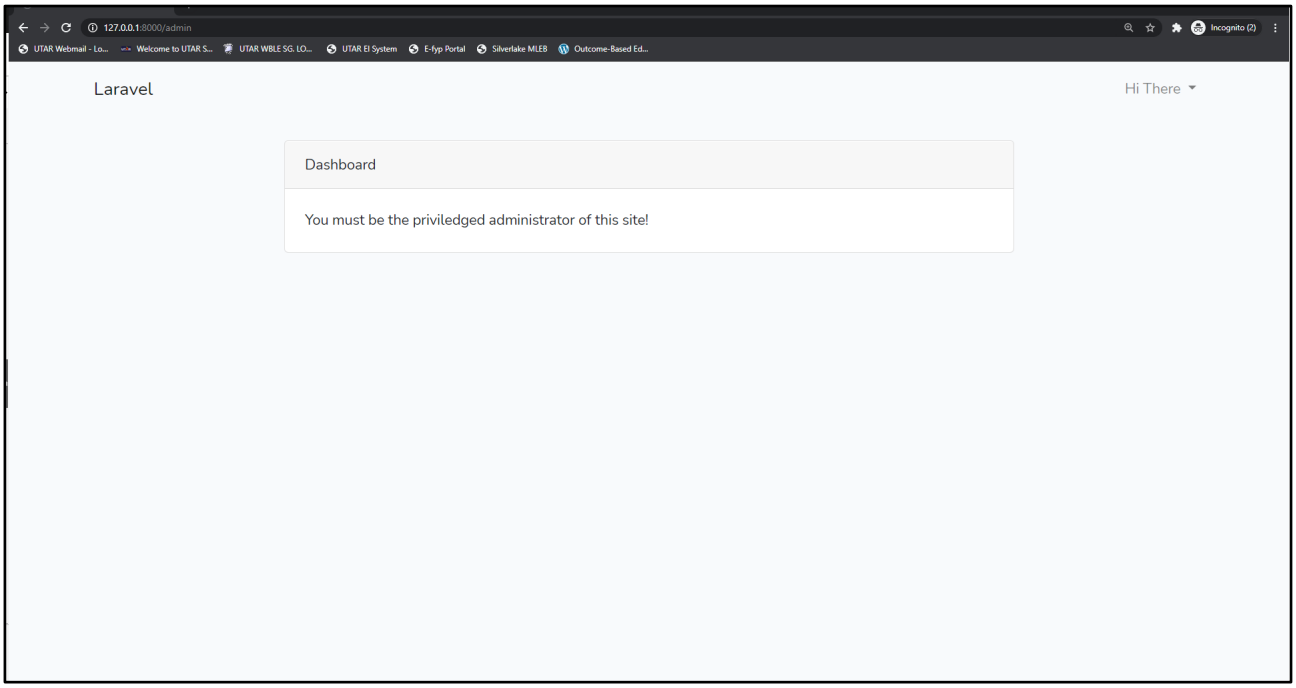
The application is now ready. Host the application in your local server using the Artisan CLI and see how the workings of authentication are executed accordingly using guards and laravel/ui package.




127.0.0.1:8000/login/author
Incognito (2)
UTAR Webmail - Lo...
Welcome to UTAR S...
UTAR WBLE SG. LO...
UTAR EI System
E-fyp Portal
Silverlake MLEB
Outcome-Based Ed...
Laravel
Login
Register
Author Login
E-Mail Address
author1@gmail.com
Password
Remember Me
Login
Forgot Your Password?


127.0.0.1:8000/author
Incognito (2)
UTAR Webmail - Lo...
Welcome to UTAR S...
UTAR WBLE SG. LO...
UTAR EI System
E-fyp Portal
Silverlake MLEB
Outcome-Based Ed...
Laravel
Hi There
Dashboard
Hi there, awesome author for this site!


127.0.0.1:8000/login/admin
Incognito (2)
UTAR Webmail - Lo...
Welcome to UTAR S...
UTAR WBLE SG. LO...
UTAR EI System
E-fyp Portal
Silverlake MLE8
Outcome-Based Ed...
Laravel
Login
Register
Admin Login
E-Mail Address
admin@gmail.com
Password
Remember Me
Login
Forgot Your Password?


127.0.0.1:8000/admin
Incognito (2)
UTAR Webmail - Lo...
Welcome to UTAR S...
UTAR WBLE SG. LO...
UTAR EI System
E-fyp Portal
Silverlake MLE8
Outcome-Based Ed...
Laravel
Hi There
Dashboard
You must be the priviledged administrator of this site!