**Multi authentication** (customer/staff area)

Tạo project customerStaff bằng dòng lệnh bên dưới

composer create-project --prefer-dist laravel/laravel=8.5.9 customerStaff

Tạo domain customer.staff.authentication.laravel.com và trỏ vào thư mục customerStaff (dùng virtual host)

Tạo database customer-staff-authentication-laravel-k37 (lưu ý k37 sẽ thay đổi theo khóa học)

Chỉ định database trong file .env

DB\_CONNECTION=mysql

DB\_HOST=127.0.0.1

DB\_PORT=3306

DB\_DATABASE=customer-staff-authentication-laravel-k37

DB\_USERNAME=root

DB\_PASSWORD=

**Xác thực khu vực user**

Để tạo xác thực đơn giản, có hỗ trợ bootstrap ta dùng gói laravel/ui

composer require laravel/ui

php artisan ui vue --auth

Tạo table customers có cấu trúc tương tự như table users. Tạo table customers bằng dòng lệnh bên dưới.

php artisan make:migration create\_customers\_table --create=customers

Dòng lệnh trên sinh ra file dùng để migrate tại thư mục: database\migrations

Mở file <datetime>\_create\_customers\_table.php và chỉnh sửa function up()

 public function up()

    {

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

            $table->id();

            $table->string('name');

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

            $table->timestamp('email\_verified\_at')->nullable();

            $table->string('password');

            $table->rememberToken();

            $table->timestamps();

        });

    }

Chạy dòng lệnh php artisan migrate để hoàn tất import các tables vào trong database

Tạo model Customer tương tự như model User (app/Models/Customer.php)

<?php

namespace App\Models;

use Illuminate\Contracts\Auth\MustVerifyEmail;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Foundation\Auth\User as Authenticatable;

use Illuminate\Notifications\Notifiable;

class Customer extends Authenticatable

{

    use HasFactory, Notifiable;

    /\*\*

     \* The attributes that are mass assignable.

     \*

     \* @var array

     \*/

    protected $fillable = [

        'name',

        'email',

        'password',

    ];

    /\*\*

     \* The attributes that should be hidden for arrays.

     \*

     \* @var array

     \*/

    protected $hidden = [

        'password',

        'remember\_token',

    ];

    /\*\*

     \* The attributes that should be cast to native types.

     \*

     \* @var array

     \*/

    protected $casts = [

        'email\_verified\_at' => 'datetime',

    ];

}

Vào file config/auth.php, chỉnh sửa lại model trong providers của guard web cho phù hợp

'providers' => [

        'users' => [

            'driver' => 'eloquent',

            'model' => App\Models\Customer::class,

        ],

        // 'users' => [

        //     'driver' => 'database',

        //     'table' => 'users',

        // ],

    ],

Sửa lại file app\Http\Controllers\Auth\RegisterController.php

use App\Models\Customer;

protected function create(array $data)

    {

        return Customer::create([

            'name' => $data['name'],

            'email' => $data['email'],

            'password' => Hash::make($data['password']),

        ]);

    }

Trong thư mục copy đính kèm, copy thư mục public và resources\views\layouts

**Xác thực khu vực admin**

Phần trên là hỗ trợ cho xác thực ở phía người dùng (customer)

Để tạo xác thực phía admin ta tạo thêm middleware và guard tương ứng

Đầu tiên tạo table staffs có cấu trúc gần tương tự như table users. Tạo table staffs bằng dòng lệnh bên dưới.

php artisan make:migration create\_staffs\_table --create=staffs

Dòng lệnh trên sinh ra file dùng để migrate tại thư mục: database\migrations

Mở file <datetime>\_create\_staffs\_table.php và chỉnh sửa function up()

 public function up()

    {

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

            $table->id();

            $table->string('name');

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

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

            $table->timestamp('email\_verified\_at')->nullable();

            $table->string('password');

            $table->rememberToken();

            $table->timestamps();

        });

    }

Chạy dòng lệnh php artisan migrate để hoàn tất import các tables vào trong database

Tạo model Staff tương tự như model User (app/Models/Staff.php)

<?php

namespace App\Models;

use Illuminate\Contracts\Auth\MustVerifyEmail;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Foundation\Auth\User as Authenticatable;

use Illuminate\Notifications\Notifiable;

class Staff extends Authenticatable

{

    use HasFactory, Notifiable;

    /\*\*

     \* The attributes that are mass assignable.

     \*

     \* @var array

     \*/

    protected $fillable = [

        'name',

'username',

        'email',

        'password',

    ];

    /\*\*

     \* The attributes that should be hidden for arrays.

     \*

     \* @var array

     \*/

    protected $hidden = [

        'password',

        'remember\_token',

    ];

    /\*\*

     \* The attributes that should be cast to native types.

     \*

     \* @var array

     \*/

    protected $casts = [

        'email\_verified\_at' => 'datetime',

    ];

}

Tiếp theo tạo middleware AuthAdmin

php artisan make:middleware AuthAdmin

Cập nhật file app\Http\Middleware\AuthAdmin.php

<?php

namespace App\Http\Middleware;

use Illuminate\Auth\Middleware\Authenticate as Middleware;

class AuthAdmin extends Middleware

{

    protected function redirectTo($request)

    {

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

            return route('admin.login');

        }

    }

}

Khai báo middleware AuthAdmin trong app\Http\Kernel.php

protected $routeMiddleware = [

        'auth' => \App\Http\Middleware\Authenticate::class,

        'auth.basic' => \Illuminate\Auth\Middleware\AuthenticateWithBasicAuth::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,

        'authAdmin' => \App\Http\Middleware\AuthAdmin::class,

    ];

Khai báo guard admin để xác thực trong file config/auth.php

'guards' => [

        'web' => [

            'driver' => 'session',

            'provider' => 'users',

        ],

        'api' => [

            'driver' => 'token',

            'provider' => 'users',

            'hash' => false,

        ],

        'admin' => [

            'driver' => 'session',

            'provider' => 'admins',

        ],

    ],

'providers' => [

        'users' => [

            'driver' => 'eloquent',

            'model' => App\Models\Customer::class,

        ],

        'admins' => [

            'driver' => 'eloquent',

            'model' => App\Models\Staff::class,

        ],

        // 'users' => [

        //     'driver' => 'database',

        //     'table' => 'users',

        // ],

    ],

Tạo các routes hỗ trợ cho phần admin (routes/web.php)

use Illuminate\Support\Facades\Auth;

// Admin area

Route::prefix('admin')->namespace('App\Http\Controllers\Admin')->group(function () {

    Route::get('login', 'Auth\LoginController@showLoginForm')->name('admin.login');

    Route::post('login', 'Auth\LoginController@login');

    Route::post('logout', 'Auth\LoginController@logout')->name('admin.logout');

    Route::get('password/confirm', 'Auth\ConfirmPasswordController@showConfirmForm')->name('admin.password.confirm');

    Route::post('password/confirm', 'Auth\ConfirmPasswordController@confirm')->name('admin.password.confirm');

    Route::post('password/email', 'Auth\\ForgotPasswordController@sendResetLinkEmail')->name('admin.password.email');

    Route::get('password/reset', 'Auth\ForgotPasswordController@showLinkRequestForm')->name('admin.password.request');

    Route::post('password/reset', 'Auth\ResetPasswordController@reset')->name('admin.password.update');

    Route::get('password/reset/{token}', 'Auth\ResetPasswordController@showResetForm')->name('admin.password.reset');

    Route::get('register', 'Auth\RegisterController@showRegistrationForm')->name('admin.register');

    Route::post('register', 'Auth\RegisterController@register');

    Route::middleware(['authAdmin:admin'])->group(function () {

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

    });

 });

Copy controller, view từ thư mục đính kèm (thư mục app trong thư mục copy đính kèm)

Chỉnh sửa file app\Http\Controllers\Admin\Auth\RegisterController.php

use App\Models\Staff;

protected function validator(array $data)

    {

        return Validator::make($data, [

            'name' => ['required', 'string', 'max:255'],

            'email' => ['required', 'string', 'email', 'max:255', 'unique:staffs'],

            'username' => ['required', 'string', 'max:255', 'unique:staffs,username'],

            'password' => ['required', 'string', 'min:8', 'confirmed'],

        ]);

    }

protected function create(array $data)

    {

        return Staff::create([

            'name' => $data['name'],

            'email' => $data['email'],

'username' => $data['username'],

            'password' => Hash::make($data['password']),

        ]);

    }

Chỉnh sửa file app\Http\Controllers\Admin\Auth\LoginController.php

/\*\*

     \* Get the login username to be used by the controller.

     \*

     \* @return string

     \*/

    public function username()

    {

        return 'username';

    }

Chỉnh sửa view resources\views\admin\auth\register.blade.php, thêm field nhập liệu username

<div class="form-group row">

                            <label for="username" class="col-md-4 col-form-label text-md-right">{{ \_\_('Username') }}</label>

                            <div class="col-md-6">

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

                                @error('username')

                                    <span class="invalid-feedback" role="alert">

                                        <strong>{{ $message }}</strong>

                                    </span>

                                @enderror

                            </div>

                        </div>

Chỉnh sửa file resources\views\admin\auth\login.blade.php, từ email đổi sang username

<div class="form-group row">

                            <label for="username" class="col-md-4 col-form-label text-md-right">{{ \_\_('Username') }}</label>

                            <div class="col-md-6">

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

                                @error('username')

                                    <span class="invalid-feedback" role="alert">

                                        <strong>{{ $message }}</strong>

                                    </span>

                                @enderror

                            </div>

                        </div>

Để phân biệt 2 area login (admin/user). Ta cần chỉnh sửa trong hàm login của app\Http\Controllers\**Admin**\Auth\LoginController.php

public function \_\_construct()

    {

        $this->middleware('guest:admin')->except('logout');

    }

Và file app\Http\Controllers\**Admin**\Auth\RegisterController.php

public function \_\_construct()

    {

        $this->middleware('guest:admin');

    }

Để logout của area này không ảnh hưởng đến area của vùng kia thì tắt 2 dòng code như bên dưới trong file vendor\laravel\ui\auth-backend\AuthenticatesUsers.php

public function logout(Request $request)

    {

        $this->guard()->logout();

        //$request->session()->invalidate();

        //$request->session()->regenerateToken();

        if ($response = $this->loggedOut($request)) {

            return $response;

        }

        return $request->wantsJson()

            ? new JsonResponse([], 204)

            : redirect('/');

    }

Cũng remove hàm regenerate() session (chưa làm)

/\*\*

     \* Send the response after the user was authenticated.

     \*

     \* @param  \Illuminate\Http\Request  $request

     \* @return \Illuminate\Http\RedirectResponse|\Illuminate\Http\JsonResponse

     \*/

    protected function sendLoginResponse(Request $request)

    {

        // $request->session()->regenerate();

        $this->clearLoginAttempts($request);

        if ($response = $this->authenticated($request, $this->guard()->user())) {

            return $response;

        }

        return $request->wantsJson()

                    ? new JsonResponse([], 204)

                    : redirect()->intended($this->redirectPath());

    }