composer create-project --prefer-dist laravel/laravel laravelRolePermission

cd laravelRolePermission

composer require laravel/ui

php artisan ui bootstrap

php artisan ui bootstrap –auth

[npm install && npm run dev

npm install && npm run dev // again

npm run watch // if need  
  
composer require laravelcollective/html

$ php artisan make:model Role -mcr

$ php artisan make:model Permission -mcr

$ php artisan make:model Role\_has\_permission -m

$ php artisan make:controller AdminController

$ php artisan make: controller DeveloperController

$ php artisan make: controller ExecutiveController

**Edit all table**

User table

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

            $table->id();

            $table->string('name');

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

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

            $table->string('password');

            $table->tinyInteger('role\_id')->nullable();

            $table->tinyInteger('status\_id')->nullable();

            $table->rememberToken();

            $table->timestamps();

        });

Role table

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

            $table->id();

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

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

            $table->timestamps();

        });

Permission table

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

            $table->id();

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

            $table->timestamps();

        });

Role has permission table

Schema::create('role\_has\_permissions', function (Blueprint $table) {

            $table->bigIncrements('id');

            $table->unsignedBigInteger('role\_id')->nullable();

            $table->unsignedBigInteger('permission\_id')->nullable();

            $table->foreign('role\_id')->references('id')->on('roles')->onDelete('cascade');

            $table->foreign('permission\_id')->references('id')->on('permissions')->onDelete('cascade');

            $table->timestamps();

        });

**All model**   
===========

User model

 protected $fillable = [

        'name',

        'email',

        'password',

        'role\_id',

        'status\_id',

    ];

    public function roles(){

        return $this->belongsTo('App\Role');

    }

Role model

  protected $fillable=['name','slug'];

    public function users(){

            return $this->hasMany('App\User');

    }

Permission model

 protected $fillable=['name'];

    public function role(){

        return $this->hasMany('App\Role');

    }

Role has permission model

 protected $fillable=['role\_id','permission\_id'];

    public function roles()

        {

             return $this->belongsToMany(Role::class);

        }

    public function permissions()

        {

            return $this->hasMany('App\Permission');

        }

**Some command**  
==================

$ php artisan migrate

$ php artisan make:seed UsersTableSeeder

$ php artisan make:seed PermissionsTableSeeder

$ php artisan make:seed RolesTableSeeder

**Create seeder**  
================

Role seeder  
---------------

use DB;

   public function run()

    {

           DB::table('roles')->insert([

            'name' => 'admin',

            'slug' => 'admin',

        ]);

        DB::table('roles')->insert([

            'name' => 'executive',

            'slug' => 'executive',

        ]);

        DB::table('roles')->insert([

            'name' => 'developer',

            'slug' => 'developer',

        ]);

    }

Permission seeder  
-------------------------

use App\Models\Permission;

use DB;

  $data = [

            'user-list',

            'user-create',

            'user-edit',

            'user-delete',

            'role-list',

            'role-create',

            'role-edit',

            'role-delete',

            'permission-list',

            'permission-create',

            'permission-edit',

            'permission-delete',

        ];

        foreach ($data as $permission) {

             Permission::create(['name' => $permission]);

        }

User seeder

use DB;

 DB::table('users')->insert([

            'role\_id' => '1',

            'status\_id' => '1',

            'name' => 'Admin',

            'email' => 'admin@gmail.com',

            'password' => bcrypt('123456789'),

        ]);

        DB::table('users')->insert([

            'role\_id' => '2',

            'status\_id' => '1',

            'name' => 'Executive',

            'email' => 'executive@gmail.com',

            'password' => bcrypt('123456789'),

        ]);

        DB::table('users')->insert([

            'role\_id' => '3',

            'status\_id' => '1',

            'name' => 'developer',

            'email' => 'developer@gmail.com',

            'password' => bcrypt('123456789'),

        ]);

Database seeder

use DB;

        $this->call(PermissionsTableSeeder::class);

        $this->call(RolesTableSeeder::class);

        $this->call(UsersTableSeeder::class);

$php artisan db:seed // php artisan db:seed --class= PermissionsTableSeeder

**Create Middleware**   
=====================

$ php artisan make:middleware Admin

$ php artisan make:middleware Executive

$ php artisan make:middleware Developer

$ php artisan make:middleware PriventBackHistory

**app\Http\Middleware**  
------------------------------

Admin middleware

use Auth;

if(!Auth::check()){

            return redirect()->route('login');

        }

        // Role\_id=1 (admin)

        if(Auth::check() && Auth::user()->role\_id==1){

            return $next($request);

        }

        // Role\_id=2 (Executive)

        if(Auth::check() && Auth::user()->role\_id == 2){

            return redirect()->route('executive');

        }

        // Role\_id = 3 (Developer)

        if(Auth::check() && Auth::user()->role\_id == 3 ){

            return redirect()->route('developer');

        }

Executive Middleware

-----------------------------

use Auth;

  if(!Auth::check()){

            return redirect()->route('login');

        }

        // Role\_id=2 (Executive)

        if(Auth::check() && Auth::user()->role\_id == 2){

            return $next($request);

        }

Developer middleware

-----------------------------

use Auth;

      if(!Auth::check()){

            return redirect()->route('login');

        }

        // Role\_id = 3 (Developer)

        if(Auth::check() && Auth::user()->role\_id == 3 ){

            return $next($request);

        }

PriventBackHistory middleware

-------------------------------------------

use Auth;

     return $next($request);

        return $response->header('Cache-Control', 'private,no-cache,no-store,max-age=0,must-revalidate,pre-check=0,post-check=0')

                        ->header('Pragme','no-cashe')

                        ->header('Expires','Sun, 02 Jan 1990 00:00:00 GMT');

RedirectIfAuthenticated middleware (Just edit)   
---------------------------------------------------

  public function handle(Request $request, Closure $next, ...$guards)

    {

        if (Auth::guard($guards)->check() && Auth::user()->role\_id == 1) {

            return redirect()->route('admin.index');

        } elseif(Auth::guard($guards)->check() && Auth::user()->role\_id == 2){

            return redirect()->route('executive.index');

        } elseif(Auth::guard($guards)->check() && Auth::user()->role\_id == 3){

            return redirect()->route('developer.index');

        }else {

            return $next($request);

      }

        // $guards = empty($guards) ? [null] : $guards;

        // foreach ($guards as $guard) {

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

        //         return redirect(RouteServiceProvider::HOME);

        //     }

        // }

        // return $next($request);

    }

**Kernel.php (Added last lines)**   
=================================

**app\Http**  
-------------

 protected $routeMiddleware = [

    'admin' => \App\Http\Middleware\Admin::class,

          'executive' => \App\Http\Middleware\Executive::class,

          'developer' => \App\Http\Middleware\Developer::class,

          'priventBackHistory' => \App\Http\Middleware\PriventBackHistory::class,

    ];

**LoginController**  
==================

**app\Http\Controllers\Auth**

---------------------------------------

use Auth;

// protected $redirectTo = RouteServiceProvider::HOME;

    public function redirectTo(){

        switch(Auth::user()->role\_id){

            case 1:

                $this->redirectTo = '/admin';

                return $this->redirectTo;

            break;

            case 2:

                $this->redirectTo = '/executive';

                return $this->redirectTo;

            break;

            case 3:

                $this->redirectTo = '/developer';

                return $this->redirectTo;

            break;

            default;

                $this->redirectTo = '/login';

                return $this->redirectTo;

            break;

        }

    }

**RegisterController**  
=====================  
**app\Http\Controllers\Auth**--------------------------------------

  public function \_\_construct()

    {

        $this->middleware('auth');

    }

**Route**  
=======

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

    return view('welcome');

});

// Auth::routes();

Auth::routes(['register' => false]);

Route::get('login', [App\Http\Controllers\CustomAuthController::class, 'index'])->name('login');

Route::post('custom-login', [App\Http\Controllers\CustomAuthController::class, 'customLogin'])->name('login.custom');

Route::get('registration', [App\Http\Controllers\CustomAuthController::class, 'registration'])->name('register-user');

Route::post('custom-registration', [App\Http\Controllers\CustomAuthController::class, 'customRegistration'])->name('register.custom');

Route::get('signout', [App\Http\Controllers\CustomAuthController::class, 'signOut'])->name('signout');

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

    Route::get('/', [App\Http\Controllers\AdminController::class, 'index'])->name('admin');

    // Users

    Route::get('users', [App\Http\Controllers\AdminController::class, 'user'])->name('admin.users');

    Route::get('/users.create',  [App\Http\Controllers\AdminController::class, 'usercreate'])->name('admin.users.create');

    Route::post('users.store', [App\Http\Controllers\AdminController::class, 'userstore'])->name('admin.users.store');

    Route::get('users.show.{id}', [App\Http\Controllers\AdminController::class, 'usershow'])->name('admin.users.show');

    Route::get('users.edit.{id}', [App\Http\Controllers\AdminController::class, 'useredit'])->name('admin.users.edit');

    Route::get('users.publish.{id}', [App\Http\Controllers\AdminController::class, 'userpublish'])->name('admin.users.publish');

    Route::get('users.unpublish.{id}', [App\Http\Controllers\AdminController::class, 'userunpublish'])->name('admin.users.unpublish');

    Route::patch('users.update.{id}', [App\Http\Controllers\AdminController::class, 'userupdate'])->name('admin.users.update');

    Route::delete('/users.destroy.{id}', [App\Http\Controllers\AdminController::class, 'userdestroy'])->name('admin.users.destroy');

    // Admin role

    Route::get('/roles', [App\Http\Controllers\AdminController::class, 'role'])->name('admin.roles');

    Route::get('/roles.create', [App\Http\Controllers\AdminController::class, 'rolecreate'])->name('admin.roles.create');

    Route::post('/roles.store', [App\Http\Controllers\AdminController::class, 'rolestore'])->name('admin.roles.store');

    Route::get('/roles.show.{id}', [App\Http\Controllers\AdminController::class, 'roleshow'])->name('admin.roles.show');

    Route::get('/roles.edit.{id}', [App\Http\Controllers\AdminController::class, 'roleedit'])->name('admin.roles.edit');

    Route::patch('/roles.update.{id}', [App\Http\Controllers\AdminController::class, 'roleupdate'])->name('admin.roles.update');

    Route::delete('/roles.destroy.{id}', [App\Http\Controllers\AdminController::class, 'roledelete'])->name('admin.roles.destroy');

    // Admin permission

    Route::get('/permissions', [App\Http\Controllers\AdminController::class, 'permission'])->name('admin.permissions');

    Route::get('/permissions.create', [App\Http\Controllers\AdminController::class, 'permissioncreate'])->name('admin.permissions.create');

    Route::post('/permissions.store', [App\Http\Controllers\AdminController::class, 'permissionstore'])->name('admin.permissions.store');

    Route::get('/permissions.show.{id}', [App\Http\Controllers\AdminController::class, 'permissionshow'])->name('admin.permissions.show');

    Route::get('/permissions.edit.{id}', [App\Http\Controllers\AdminController::class, 'permissionedit'])->name('admin.permissions.edit');

    Route::patch('/permissions.update.{id}', [App\Http\Controllers\AdminController::class, 'permissionupdate'])->name('admin.permissions.update');

    Route::delete('/permissions.destroy.{id}', [App\Http\Controllers\AdminController::class, 'permissiondelete'])->name('admin.permissions.destroy');

});

Route::group(['prefix' => 'executive','middleware' => ['auth','executive','priventBackHistory']], function() {

    Route::get('/', [App\Http\Controllers\ExecutiveController::class, 'index'])->name('executive');

    // Users

    Route::get('users', [App\Http\Controllers\ExecutiveController::class, 'user'])->name('executive.users');

    Route::get('/users.create',  [App\Http\Controllers\ExecutiveController::class, 'usercreate'])->name('executive.users.create');

    Route::post('users.store', [App\Http\Controllers\ExecutiveController::class, 'userstore'])->name('executive.users.store');

    Route::get('users.show.{id}', [App\Http\Controllers\ExecutiveController::class, 'usershow'])->name('executive.users.show');

    Route::get('users.edit.{id}', [App\Http\Controllers\ExecutiveController::class, 'useredit'])->name('executive.users.edit');

    Route::get('users.publish.{id}', [App\Http\Controllers\ExecutiveController::class, 'userpublish'])->name('executive.users.publish');

    Route::get('users.unpublish.{id}', [App\Http\Controllers\ExecutiveController::class, 'userunpublish'])->name('executive.users.unpublish');

    Route::patch('users.update.{id}', [App\Http\Controllers\ExecutiveController::class, 'userupdate'])->name('executive.users.update');

    Route::delete('/users.destroy.{id}', [App\Http\Controllers\ExecutiveController::class, 'userdestroy'])->name('executive.users.destroy');

    // Admin role

    Route::get('/roles', [App\Http\Controllers\ExecutiveController::class, 'role'])->name('executive.roles');

    Route::get('/roles.create', [App\Http\Controllers\ExecutiveController::class, 'rolecreate'])->name('executive.roles.create');

    Route::post('/roles.store', [App\Http\Controllers\ExecutiveController::class, 'rolestore'])->name('executive.roles.store');

    Route::get('/roles.show.{id}', [App\Http\Controllers\ExecutiveController::class, 'roleshow'])->name('executive.roles.show');

    Route::get('/roles.edit.{id}', [App\Http\Controllers\ExecutiveController::class, 'roleedit'])->name('executive.roles.edit');

    Route::patch('/roles.update.{id}', [App\Http\Controllers\ExecutiveController::class, 'roleupdate'])->name('executive.roles.update');

    Route::delete('/roles.destroy.{id}', [App\Http\Controllers\ExecutiveController::class, 'roledelete'])->name('executive.roles.destroy');

    // Admin permission

     Route::get('/permissions', [App\Http\Controllers\ExecutiveController::class, 'permission'])->name('executive.permissions');

    Route::get('/permissions.create', [App\Http\Controllers\ExecutiveController::class, 'permissioncreate'])->name('executive.permissions.create');

    Route::post('/permission.store', [App\Http\Controllers\ExecutiveController::class, 'permissionstore'])->name('executive.permissions.store');

    Route::get('/permissions.show.{id}', [App\Http\Controllers\ExecutiveController::class, 'permissionshow'])->name('executive.permissions.show');

    Route::get('/permissions.edit.{id}', [App\Http\Controllers\ExecutiveController::class, 'permissionedit'])->name('executive.permissions.edit');

    Route::patch('/permissions.update.{id}', [App\Http\Controllers\ExecutiveController::class, 'permissionupdate'])->name('executive.permissions.update');

    Route::delete('/permissions.destroy.{id}', [App\Http\Controllers\ExecutiveController::class, 'permissiondelete'])->name('executive.permissions.destroy');

});

Route::group(['prefix' => 'developer','middleware' => ['auth','developer','priventBackHistory']], function() {

    Route::get('/', [App\Http\Controllers\DeveloperController::class, 'index'])->name('developer');

    Route::get('users', [App\Http\Controllers\DeveloperController::class, 'user'])->name('developer.users');

    Route::get('/users.create',  [App\Http\Controllers\DeveloperController::class, 'usercreate'])->name('developer.users.create');

    Route::post('users.store', [App\Http\Controllers\DeveloperController::class, 'userstore'])->name('developer.users.store');

    Route::get('users.show.{id}', [App\Http\Controllers\DeveloperController::class, 'usershow'])->name('developer.users.show');

    Route::get('users.edit.{id}', [App\Http\Controllers\DeveloperController::class, 'useredit'])->name('developer.users.edit');

    Route::get('users.publish.{id}', [App\Http\Controllers\DeveloperController::class, 'userpublish'])->name('developer.users.publish');

    Route::get('users.unpublish.{id}', [App\Http\Controllers\DeveloperController::class, 'userunpublish'])->name('developer.users.unpublish');

    Route::patch('users.update.{id}', [App\Http\Controllers\DeveloperController::class, 'userupdate'])->name('developer.users.update');

    Route::delete('/users.destroy.{id}', [App\Http\Controllers\DeveloperController::class, 'userdestroy'])->name('developer.users.destroy');

    // Admin role

    Route::get('/roles', [App\Http\Controllers\DeveloperController::class, 'role'])->name('developer.roles');

    Route::get('/roles.create', [App\Http\Controllers\DeveloperController::class, 'rolecreate'])->name('developer.roles.create');

    Route::post('/roles.store', [App\Http\Controllers\DeveloperController::class, 'rolestore'])->name('developer.roles.store');

    Route::get('/roles.show.{id}', [App\Http\Controllers\DeveloperController::class, 'roleshow'])->name('developer.roles.show');

    Route::get('/roles.edit.{id}', [App\Http\Controllers\DeveloperController::class, 'roleedit'])->name('developer.roles.edit');

    Route::patch('/roles.update.{id}', [App\Http\Controllers\DeveloperController::class, 'roleupdate'])->name('developer.roles.update');

    Route::delete('/roles.destroy.{id}', [App\Http\Controllers\DeveloperController::class, 'roledelete'])->name('developer.roles.destroy');

    // Admin permission

    Route::get('/permissions', [App\Http\Controllers\DeveloperController::class, 'permission'])->name('developer.permissions');

    Route::get('/permissions.create', [App\Http\Controllers\DeveloperController::class, 'permissioncreate'])->name('developer.permissions.create');

    Route::post('/permissions.store', [App\Http\Controllers\DeveloperController::class, 'permissionstore'])->name('developer.permissions.store');

    Route::get('/permissions.show.{id}', [App\Http\Controllers\DeveloperController::class, 'permissionshow'])->name('developer.permissions.show');

    Route::get('/permissions.edit.{id}', [App\Http\Controllers\DeveloperController::class, 'permissionedit'])->name('developer.permissions.edit');

    Route::patch('/permissions.update.{id}', [App\Http\Controllers\DeveloperController::class, 'permissionupdate'])->name('developer.permissions.update');

    Route::delete('/permissions.destroy.{id}', [App\Http\Controllers\DeveloperController::class, 'permissiondelete'])->name('developer.permissions.destroy');

});

**Admin Controller**

===================

use App\Models\Admin;

use App\Models\User;

use App\Models\Role;

use App\Models\Permission;

use App\Models\Role\_has\_permission;

use DB;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Auth;

use Illuminate\Support\Facades\Http;

use App\Http\Controllers\RoleController;

use App\Http\Controllers\PermissionController;

use App\Http\Controllers\UserController;

use App\Http\Controllers\ProjetController;

class AdminController extends Controller

{

    public function index(){

        return view('admin.index');

    }

    // ================== User=============

    public function user(){

        $user  = new UserController;

        $data = $user->index();

        return view('admin.users.index', compact('data'));

    }

     public function usercreate(){

        return view('admin.users.create');

    }

    public function userstore(Request $request){

        $user  = new UserController;

        $user->store($request);

        return redirect()->route('admin.users')

            ->with('success', 'User created successfully.');

    }

    public function usershow($id){

        $user  = new UserController;

        $urs = $user->show($id);

        $users = $urs[0];

        return view('admin.users.show', compact('users'));

    }

    public function useredit($id){

        $users  = new UserController;

        $urs = $users->edit($id);

        $user = $urs[0];

        return view('admin.users.edit', compact('user'));

    }

    public function userpublish($id){

        $publish =  User::find($id);

        $publish->status\_id = 0;

        $publish->save();

        return redirect('admin/users');

    }

    public function userunpublish($id){

        $publish =  User::find($id);

        $publish->status\_id = 1;

        $publish->save();

        return redirect('admin/users');

    }

    public function userupdate(Request $request, $id) {

        $user  = new UserController;

        $user->update($request, $id);

        return redirect()->route('admin.users')

            ->with('success', 'User updated successfully.');

    }

    public function userdestroy($id)

    {

        $user  = new UserController;

        $urs = $user->destroy($id);

        return redirect()->route('admin.users')

            ->with('success', 'User deleted successfully.');

    }

    // ================== Role=============

    public function role(){

        $role  = new RoleController;

        $data = $role->index();

        return view('admin.roles.index', compact('data'));

    }

     public function rolecreate(){

        $role  = new RoleController;

        $roles = $role->create();

        $permission = $roles[0];

        $users = $roles[1];

        return view('admin.roles.create', compact(['permission','users']));

    }

    public function rolestore(Request $request){

        $role  = new RoleController;

        $role->store($request);

        return redirect()->route('admin.roles')

            ->with('success', 'Role created successfully.');

    }

    public function roleshow($id){

        $role  = new RoleController;

        $roles = $role->show($id);

        $role = $roles[0];

        $rolePermissions = $roles[1];

        return view('admin.roles.show', compact('role', 'rolePermissions'));

    }

    public function roleedit($id){

        $role  = new RoleController;

        $roles = $role->edit($id);

        $role = $roles[0];

        $permission = $roles[1];

        $rolePermissions = $roles[2];

        return view('admin.roles.edit', compact('role', 'permission', 'rolePermissions'));

    }

    public function roleupdate(Request $request, $id) {

        $role  = new RoleController;

        $role->update($request, $id);

        return redirect()->route('admin.roles')

            ->with('success', 'Role updated successfully.');

    }

    // ================== permission=============

    public function permission(){

        $permission  = new PermissionController;

        $data = $permission->index();

        return view('admin.permissions.index', compact('data'));

    }

     public function permissioncreate(){

        return view('admin.permissions.create');

    }

    public function permissionstore(Request $request){

        $permissions  = new PermissionController;

        $permissions->store($request);

        return redirect()->route('admin.permissions')

            ->with('success', 'Permission created successfully.');

    }

    public function permissionshow($id){

        $permission  = new PermissionController;

        $prms = $permission->show($id);

        $permissions = $prms[0];

        return view('admin.permissions.show', compact('permissions'));

    }

    public function permissionedit($id){

        $permission  = new PermissionController;

        $permission = $permission->edit($id);

        $permissions = $permission[0];

        return view('admin.permissions.edit', compact('permissions'));

    }

    public function permissionupdate(Request $request, $id) {

        $role  = new PermissionController;

        $role->update($request, $id);

        return redirect()->route('admin.permissions')

            ->with('success', 'Permission updated successfully.');

    }

    public function permissiondelete($id)

    {

        $permission  = new PermissionController;

        $permission->destroy($id);

        return redirect()->route('admin.permissions')

            ->with('success', 'Pemission deleted successfully.');

    }

    //============================ Projects ===============

    public function projects(){

          $data = Project::orderBy('id', 'desc')->paginate(5);

        // return view('admin.project.index', compact('data'));

        return view('admin.project.index');

    }

    public function projectscreate(){

        return view('admin.project.create');

    }

     public function projectsstore(Request $request)

    {

        $role\_id = $request->input('role\_id');

        $permissions = $request->input('permission');

        //  return redirect()->route('admin.porjects')

        //     ->with('success', 'Permission created successfully.');

    }

}

**Executive Controller**  
----------------------------------------

use App\Models\User;

use App\Models\Executive;

use Illuminate\Http\Request;

use App\Models\Role;

use App\Models\Permission;

use App\Models\Role\_has\_permission;

use DB;

use App\Http\Controllers\RoleController;

use App\Http\Controllers\PermissionController;

use App\Http\Controllers\UserController;

class ExecutiveController extends Controller

{

    public function index()

    {

    return view('executive.index');

    }

    // ================== User=============

    public function user(){

        $user  = new UserController;

        $data = $user->index();

        return view('executive.users.index', compact('data'));

    }

     public function usercreate(){

        return view('executive.users.create');

    }

    public function userstore(Request $request){

        $user  = new UserController;

        $user->store($request);

        return redirect()->route('executive.users')

            ->with('success', 'User created successfully.');

    }

    public function usershow($id){

        $user  = new UserController;

        $urs = $user->show($id);

        $users = $urs[0];

        return view('executive.users.show', compact('users'));

    }

    public function useredit($id){

        $users  = new UserController;

        $urs = $users->edit($id);

        $user = $urs[0];

        return view('executive.users.edit', compact('user'));

    }

    public function userupdate(Request $request, $id) {

        $user  = new UserController;

        $user->update($request, $id);

        return redirect()->route('executive.users')

            ->with('success', 'User updated successfully.');

    }

     public function userpublish($id){

        $publish =  User::find($id);

        $publish->status\_id = 0;

        $publish->save();

        return redirect('executive/users');

    }

    public function userunpublish($id){

        $publish =  User::find($id);

        $publish->status\_id = 1;

        $publish->save();

        return redirect('executive/users');

    }

    public function userdestroy($id)

    {

        $user  = new UserController;

        $urs = $user->destroy($id);

        return redirect()->route('executive.users')

            ->with('success', 'User deleted successfully.');

    }

    // ================== Role=============

    public function role(){

        $role  = new RoleController;

        $data = $role->index();

        return view('executive.roles.index', compact('data'));

    }

     public function rolecreate(){

        $role  = new RoleController;

        $roles = $role->create();

        $permission = $roles[0];

        $users = $roles[1];

        return view('executive.roles.create', compact(['permission','users']));

    }

    public function rolestore(Request $request){

        $role  = new RoleController;

        $role->store($request);

        return redirect()->route('executive.roles')

            ->with('success', 'Role created successfully.');

    }

    public function roleshow($id){

        $role  = new RoleController;

        $roles = $role->show($id);

        $role = $roles[0];

        $rolePermissions = $roles[1];

        return view('executive.roles.show', compact('role', 'rolePermissions'));

    }

    public function roleedit($id){

        $role  = new RoleController;

        $roles = $role->edit($id);

        $role = $roles[0];

        $permission = $roles[1];

        $rolePermissions = $roles[2];

        return view('executive.roles.edit', compact('role', 'permission', 'rolePermissions'));

    }

    public function roleupdate(Request $request, $id) {

        $role  = new RoleController;

        $role->update($request, $id);

        return redirect()->route('executive.roles')

            ->with('success', 'Role updated successfully.');

    }

    // ================== permission=============

    public function permission(){

        $permission  = new PermissionController;

        $data = $permission->index();

        return view('executive.permissions.index', compact('data'));

    }

     public function permissioncreate(){

        return view('executive.permissions.create');

    }

    public function permissionstore(Request $request){

        $permissions  = new PermissionController;

        $permissions->store($request);

        return redirect()->route('executive.permission')

            ->with('success', 'Permission created successfully.');

    }

    public function permissionshow($id){

        $permission  = new PermissionController;

        $prms = $permission->show($id);

        $permissions = $prms[0];

        return view('executive.permissions.show', compact('permissions'));

    }

    public function permissionedit($id){

        $permission  = new PermissionController;

        $permission = $permission->edit($id);

        $permissions = $permission[0];

        return view('executive.permissions.edit', compact('permissions'));

    }

    public function permissionupdate(Request $request, $id) {

        $role  = new PermissionController;

        $role->update($request, $id);

        return redirect()->route('executive.permissions')

            ->with('success', 'Permission updated successfully.');

    }

}

**Developer Controller**-------------------------------------------

use App\Models\Developer;

use Illuminate\Http\Request;

class DeveloperController extends Controller

{

  public function index()

    {

         return view('developer.index');

    }

   // ================== User=============

    public function user(){

        $user  = new UserController;

        $data = $user->index();

        return view('developer.users.index', compact('data'));

    }

     public function usercreate(){

        return view('developer.users.create');

    }

    public function userstore(Request $request){

        $user  = new UserController;

        $user->store($request);

        return redirect()->route('developer.users')

            ->with('success', 'User created successfully.');

    }

    public function usershow($id){

        $user  = new UserController;

        $urs = $user->show($id);

        $users = $urs[0];

        return view('developer.users.show', compact('users'));

    }

    public function useredit($id){

        $users  = new UserController;

        $urs = $users->edit($id);

        $user = $urs[0];

        return view('developer.users.edit', compact('user'));

    }

    public function userpublish($id){

        $publish =  User::find($id);

        $publish->status\_id = 0;

        $publish->save();

        return redirect('developer/users');

    }

    public function userunpublish($id){

        $publish =  User::find($id);

        $publish->status\_id = 1;

        $publish->save();

        return redirect('developer/users');

    }

    public function userupdate(Request $request, $id) {

        $user  = new UserController;

        $user->update($request, $id);

        return redirect()->route('developer.users')

            ->with('success', 'User updated successfully.');

    }

    public function userdestroy($id)

    {

        $user  = new UserController;

        $urs = $user->destroy($id);

        return redirect()->route('developer.users')

            ->with('success', 'User deleted successfully.');

    }

    // ================== Role=============

    public function role(){

        $role  = new RoleController;

        $data = $role->index();

        return view('developer.roles.index', compact('data'));

    }

     public function rolecreate(){

        $role  = new RoleController;

        $roles = $role->create();

        $permission = $roles[0];

        $users = $roles[1];

        return view('developer.roles.create', compact(['permission','users']));

    }

    public function rolestore(Request $request){

        $role  = new RoleController;

        $role->store($request);

        return redirect()->route('developer.roles')

            ->with('success', 'Role created successfully.');

    }

    public function roleshow($id){

        $role  = new RoleController;

        $roles = $role->show($id);

        $role = $roles[0];

        $rolePermissions = $roles[1];

        return view('developer.roles.show', compact('role', 'rolePermissions'));

    }

    public function roleedit($id){

        $role  = new RoleController;

        $roles = $role->edit($id);

        $role = $roles[0];

        $permission = $roles[1];

        $rolePermissions = $roles[2];

        return view('developer.roles.edit', compact('role', 'permission', 'rolePermissions'));

    }

    public function roleupdate(Request $request, $id) {

        $role  = new RoleController;

        $role->update($request, $id);

        return redirect()->route('developer.roles')

            ->with('success', 'Role updated successfully.');

    }

    // ================== permission=============

    public function permission(){

        $permission  = new PermissionController;

        $data = $permission->index();

        return view('developer.permissions.index', compact('data'));

    }

     public function permissioncreate(){

        return view('developer.permissions.create');

    }

    public function permissionstore(Request $request){

        $permissions  = new PermissionController;

        $permissions->store($request);

        return redirect()->route('developer.permissions')

            ->with('success', 'Permission created successfully.');

    }

    public function permissionshow($id){

        $permission  = new PermissionController;

        $prms = $permission->show($id);

        $permissions = $prms[0];

        return view('developer.permissions.show', compact('permissions'));

    }

    public function permissionedit($id){

        $permission  = new PermissionController;

        $permission = $permission->edit($id);

        $permissions = $permission[0];

        return view('developer.permissions.edit', compact('permissions'));

    }

    public function permissionupdate(Request $request, $id) {

        $role  = new PermissionController;

        $role->update($request, $id);

        return redirect()->route('developer.permissions')

            ->with('success', 'Permission updated successfully.');

    }

    public function permissiondelete($id)

    {

        $permission  = new PermissionController;

        $permission->destroy($id);

        return redirect()->route('developer.permissions')

            ->with('success', 'Pemission deleted successfully.');

    }

}

**Permission controller**  
---------------------------------------------

use App\Models\Permission;

use DB;

use Illuminate\Http\Request;

class PermissionController extends Controller

{

    public function index()

    {

       $data = Permission::orderBy('id','DESC')->paginate(5);

       return $data;

    //    return view('permissions.index', compact('data'));

    }

    public function create()

    {

         return view('permissions.create');

    }

    public function store(Request $request)

    {

        $this->validate($request, [

            'name' => 'required|unique:permissions,name',

        ]);

        Permission::create(['name' => $request->input('name')]);

        return ($request->all());

        // return redirect()->route('permissions.index')

        //     ->with('success', 'Permission created successfully.');

    }

    public function show( $id)

    {

         $permission = Permission::find($id);

        return array($permission);

        // return view('permissions.show', compact('permission'));

    }

    public function edit($id)

    {

         $permission = Permission::find($id);

         return array($permission);

        // return view('permissions.edit', compact('permission'));

    }

    public function update(Request $request,  $id)

    {

       $this->validate($request, [

            'name' => 'required'

        ]);

        $permission = Permission::find($id);

        $permission->name = $request->input('name');

        $permission->save();

        return $request->all();

        // return redirect()->route('permissions.index')

        //     ->with('success', 'Permission updated successfully.');

    }

    public function destroy($id)

    {

           $permission = Permission::find($id)->delete();

            return array($permission);

        // return redirect()->route('permissions.index')

        //     ->with('success', 'Permission deleted successfully');

    }

**Role Controller**-------------------------------

use App\Models\Role;

use App\Models\Permission;

use App\Models\Role\_has\_permission;

use DB;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\View;

class RoleController extends Controller

{

  public function index()

    {

        $data = Role::orderBy('id','ASC')->paginate(5);

        return $data;

        // return View::first(['admin.roles.index', 'executive.roles.index'], compact('data'));

        // return view('admin.roles.index', compact('data'));

    }

    public function create()

    {

        $permission = Permission::get();

        $users = DB::table('roles')

                ->select('\*')

                ->get();

       return array($permission, $users);

        // return view('roles.create', compact(['permission','users']));

    }

    public function store(Request $request)

    {

        $role\_id = $request->input('role\_id');

        $permissions = $request->input('permission');

        foreach ($permissions as $value) {

               $postmeta = New Role\_has\_permission();

               $postmeta->role\_id =  $role\_id;

               $postmeta->permission\_id =  $value;

               $postmeta->save();

           }

           return $request->all();

        // return redirect()->route('roles.index')

        //     ->with('success', 'Role created successfully.');

    }

    public function show($id)

    {

        $role = Role::find($id);

        $rolePermissions = Permission::join('role\_has\_permissions', 'role\_has\_permissions.permission\_id', 'permissions.id')

            ->where('role\_has\_permissions.role\_id',$id)

            ->get();

        return array($role, $rolePermissions);

        // return view('roles.show', compact('role', 'rolePermissions'));

  }

    public function edit($id)

    {

        $role = Role::find($id);

        $permission = Permission::get();

        $rolePermissions = DB::table('role\_has\_permissions')

            ->where('role\_has\_permissions.role\_id', $id)

            ->get();

        return array($role, $permission, $rolePermissions);

        // return view('roles.edit', compact('role', 'permission', 'rolePermissions'));

    }

    public function update(Request $request, $id)

    {

        $role\_id = DB::table('role\_has\_permissions')

            ->where('role\_has\_permissions.role\_id', $id)

            ->first();

        $dbprmsID='';

        $permissions = $request->input('permission');

        $unpermission = $request->input('unpermission');

        if(isset($permissions) && isset($unpermission)){

                $results=array\_diff($unpermission,$permissions);

                foreach($unpermission as $uncheck) {

                    foreach ($permissions as $value) {

                           $dbprmsID = $value;

                                    DB::table('role\_has\_permissions')->where('permission\_id', $value)->updateOrInsert([

                                        'role\_id'=> $role\_id->role\_id,

                                        'permission\_id' => $value,

                                    ],

                                    [   'role\_id'=> $role\_id->role\_id,

                                        'permission\_id' => $value,

                                        'updated\_at'=>date('Y-m-d H:i:s')

                                    ]);

                                }

                            }

                    foreach($results as $result){

                            Role\_has\_permission::where('permission\_id',  $result)->where('role\_id', $role\_id->role\_id)->delete();

                    }

                }

                 return $request->all();

        // return redirect()->route('roles.index')

        //     ->with('success', 'Role updated successfully.');

    }

    public function destroy($id)

    {

        Role::find($id)->delete();

        return redirect()->route('roles.index')

            ->with('success', 'Role deleted successfully');

    }

}

**User Controller**  
---------------------------------

use DB;

use Hash;

use App\Models\User;

use App\Models\Role;

use App\Models\Permission;

use App\Models\role\_has\_permissions;

use Illuminate\Http\Request;

use Arr;

class UserController extends Controller

{

    public function index()

    {

        $data = User::orderBy('id', 'desc')->paginate(5);

        return $data;

        // return view('users.index', compact('data'));

    }

    public function create()

    {

        return view('users.create');

    }

    public function store(Request $request)

    {

        $this->validate($request, [

            'name' => 'required',

            'email' => 'required|email|unique:users,email',

            'password' => 'required|confirmed',

        ]);

        $input = $request->all();

        // print\_r($input);

        // die();

        $input['password'] = Hash::make($input['password']);

        $user = User::create($input);

        return $request->all();

        // return redirect()->route('users.index')

        //     ->with('success', 'User created successfully.');

    }

    public function show($id)

    {

        $user = User::find($id);

        return array($user);

        // return view('users.show', compact('user'));

    }

    public function edit($id)

    {

        $user = User::find($id);

        return array($user);

        // return view('users.edit', compact('user'));

    }

    public function update(Request $request, $id)

    {

        $this->validate($request, [

            'name' => 'required',

            'email' => 'required|email|unique:users,email,'.$id,

            'password' => 'confirmed',

        ]);

        $input['name'] = $request->name;

        $input['email'] = $request->email;

        if(!empty($input['password'])) {

            $input['password'] = Hash::make($input['password']);

        } else {

            $input = Arr::except($input, array('password'));

        }

        $input['role\_id'] = $request->role\_id;

        $input['status\_id'] = $request->status\_id;

        $user = User::find($id);

        $user->update($input);

        // die();

        return $request->all();

        // return redirect()->route('users.index')

        //     ->with('success', 'User updated successfully.');

    }

    public function destroy($id)

    {

        $user =  User::find($id)->delete();

        return array($user);

        // return redirect()->route('users.index')

        //     ->with('success', 'User deleted successfully.');

    }

}

Navbar ([Note: Every menu assign in Navbar.blade.php like menu-project])

-------------

<ul class="nav side-menu">

                         @php

                             $rhps = DB::table('role\_has\_permissions')->get();

                             $permissions = DB::table('permissions')->get();

                             $roles = DB::table('roles')->get();

                         @endphp

                         @foreach ($roles as $role)

                             @foreach ($rhps as $rhp)

                                 @foreach ($permissions as $permission)

                                     @if ($role->id == Auth::user()->role\_id && $role->id == $rhp->role\_id)

                                         @if ($rhp->permission\_id == $permission->id)

                                             @php

                                                 $name = $permission->name;

                                             @endphp

                                             @if (stristr($name, 'menu'))

                                                 @php

                                                     $value = substr(strstr($name, '-'), 1);

                                                 @endphp

                                                 <li>

                                                     @if ($value == 'users')

                                                         <a class="btn btn-primary"

                                                          href="{{ route(Auth::user()->name . '.' . $value)}}" style="text-transform: uppercase;">{{ $value }}</a>

                                                     @elseif ($value == 'roles')

                                                         <a class="btn btn-primary"

                                                             href="{{ route(Auth::user()->name . '.' . $value)}}" style="text-transform: uppercase;">{{ $value }}</a>

                                                     @elseif ($value == 'permissions')

                                                         <a class="btn btn-primary"

                                                             href="{{ route(Auth::user()->name . '.' . $value)}}" style="text-transform: uppercase;">{{ $value }}</a>

                                                     @elseif ($value == 'projects')

                                                         <a class="btn btn-primary"

                                                             href="{{ route(Auth::user()->name . '.' . $value)}}" style="text-transform: uppercase;">{{ $value }}</a>

                                                     @else

                                                     @endif

                                                 </li>

                                             @endif

                                         @endif

                                     @endif

                                 @endforeach

                             @endforeach

                         @endforeach

                     </ul>