**To initial project in Laravel [ livewiew ]**

Livewire is a full-stack framework for Laravel that makes building dynamic interfaces simple, without leaving the comfort of Laravel.

/////////////////////////////////////////////////

-composer create-project –prefer-dist laravel/laravel NameOfProject

-composer require livewire/livewire

-php artisan make:livewire NameComponent => [ to make balde.php and controller ]

In controller

Return view ()->layout(‘name of base layout’)

Then make route::get

////////////////////////////// Admin/User=> login/out///////////////////

-**composer require laravel/Jetstream** => for Laravel and includes login, registration, email verification, two-factor authentication, session management, API support via Laravel Sanctum

**-php artisan jetstream:install livewire**

It will build migration in Users table add=> $table->string(“utype”)->default(‘USR’)->comment(‘ ADM for Admin and USR for user or Customer’)

**-php artisan migrate**

In MySql=> change Admin =>utype => ADM

In blade write

 @if (Route::has('login'))

        @auth

              @if (Auth::user()->utype === 'ADM')

                  {{-- Admin --}}

  <a title="My Account" href="#">My Account ({{ Auth::user()->name }})

<I class="fa fa-angle-down" aria-hidden="true"></i></a>

<li class="menu-item"> <a href="{{ route('logout') }}"

  onclick="event.preventDefault(); document.getElementById('logout- form').submit();">Logout</a> </li>

  <form id="logout-form" method="POST" action="{{ route('logout') }}">

            @csrf </form>

  @else

                  {{-- user --}}

  @endif

   @else

       <li class="menu-item"><a title="Register or Login"

 href="{{ route('login') }}">Login</a></li>

        <li class="menu-item"><a title="Register or Login"

          href="{{ route('register') }}">Register</a></li>

    @endif

  @endif

**For Route**

//for user or customer

Route::middleware(['auth:sanctum', 'verified'])->group(function(){

Route::get('/user/dashboard',UserDashboardComponent::class)->name('user.dashboard');

// here put any route to user just who can see it

});

//for Admin

Route::middleware(['auth:sanctum', 'verified','authadmin'])->group(function(){

Route::get('/admin/dashboard',AdminDashboardComponent::class)->name('admin.dashboard');

});

**- php artisan make:middleware AuthAdmin**

**Go to Http/Middlware/AuthAdmin**

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

    {

        if(session('utype') === 'ADM')

        {

            return $next($request);

        }

        else

        {

            session()->flush();

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

        }

        return $next($request);

    }

**Go to Http/kernel.php**

 protected $routeMiddleware = [

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

]

**Go to vendor/laravel/fortify/src/Actions/attemptToAuthentication.php**

if(Auth::user()->utype === 'ADM')

            {

                session(['utype'=>'ADM']);

                return redirect(RouteServiceProvider::HOME);

            }

            else if(Auth::user()->utype === 'USR')

            {

                session(['utype'=>'USR']);

                return redirect(RouteServiceProvider::HOME);

            }

**/////////////////////////////////////////////to make product or category dynamic/////////////**

**-php artisan make:factory CategoryFactory –model=Category**

**-php artisan make:factory ProductFactory –model=Product**

**In database/factories/ProductFactory.php**

 public function definition()

    {

        $product\_name = $this->faker->unique()->words($nb = 4, $asText = true);

        $slug = Str::slug($product\_name);

        return [

            'name' => $product\_name,

            'slug' => $slug,

            'short\_description' => $this->faker->text(200),

            'description' => $this->faker->text(500),

            'regular\_price' => $this->faker->numberBetween(10, 500),

            'SKU' => 'DIGI' . $this->faker->unique()->numberBetween(100, 500),

            'stock\_status' => 'instock',

            'quantity' => $this->faker->numberBetween(100, 200),

            'image' => 'digital\_' . $this->faker->unique()->numberBetween(1, 22) . '.jpg',

            'category\_id' => $this->faker->numberBetween(1, 5)

        ];

**In database/factories/CategoryFactory.php**

public function definition()

    {

        $category\_name = $this->faker->unique()->words($nb=2,$asText=true);

        $slug=Str::slug($category\_name);

        return [

            'name'=>$category\_name,

            'slug'=>$slug

        ];

    }

**Database/seeders/DatabaseSeeder.php**

public function run()

    {

        // \App\Models\User::factory(10)->create();

        \App\Models\category::factory(6)->create();

        \App\Models\product::factory(22)->create();

    }

**-php artisan db:seed**

**//////////////////////////////////////////////////////**

use Livewire\WithPagination;

$products=product::paginate(10);

**//////////////// for cart /////////////////////**

[**https://github.com/hardevine/LaravelShoppingcart**](https://github.com/hardevine/LaravelShoppingcart)

- composer require hardevine/shoppingcart

config/app.php => providers

Gloudemans\Shoppingcart\ShoppingcartServiceProvider::class

And optionally add a new line to the aliases array:

'Cart' => Gloudemans\Shoppingcart\Facades\Cart::class,

**////////////////////////////////////////**

Admin Using WYSIWYG HTML Editor on Product Page

<https://www.tiny.cloud/>

<script>

        $(function(){

    tinymce.init({

      selector: '#short\_description',

      setup:function(editor){

          editor.on('Change',function(e){

              tinyMCE.triggerSave();

              var d\_date=$('#short\_description').val();

              @this.set('short\_description',d\_date);

          });

      }

   });

   tinymce.init({

      selector: '#description',

      setup:function(editor){

          editor.on('Change',function(e){

              tinyMCE.triggerSave();

              var sd\_date=$('#description').val();

              @this.set('description',sd\_date);

          });

      }

   });

        });

      </script>

**To add record in table**

**-php artisan make:migration add\_rstatus\_to\_order\_table --table=order**

**//////////////////////////////////////////////////////////////////**

**Send confirm mail**

**-php artisan make:mail OrderMail**

**/////////////////////////////////////////////**

# Shopping Cart Using Database

php artisan vendor:publish --provider="Gloudemans\Shoppingcart\ShoppingcartServiceProvider" --tag="config"