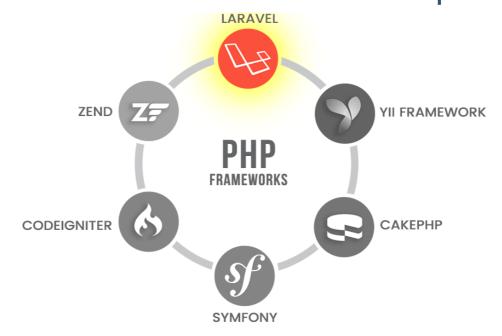
unidad didáctica 7

Laravel - carrito de compra



1. creación de un carrito de compras

Si todavía no tenemos creado un proyecto en Laravel, se deberá crear antes:

- Descargar Laravel:
 - 1 composer create-project --prefer-dist laravel/laravel carrito
- Configurar la base de datos en el archivo **.env** y creamos la base de datos en phpmyadmin como indica el video.

Para que este ejercicio sea rápido y fácil usaremos un plugin llamado darryldecode/cart:

• https://github.com/darryldecode/laravelshoppingcart

• https://github.com/darryldeco

1.1. instalar plugin carrito compra

📜 agregar al carrito

- 1. Instalamos el plugin anteriormente mencionado desde la terminal de *Visual Studio Code* ("altura" proyecto) con composer:
 - 1 sudo docker-compose exec myapp composer require "darryldecode/cart"

1.2. modificar fichero app.php

2. Editamos el archivo config/app.php y en el array **Providers** agregamos:

```
1  //..
2  Darryldecode\Cart\CartServiceProvider::class,
3  //..
```

Luego, en el mismo fichero, en el array Aliases agregamos:

```
1 //...
2 'Cart' => Darryldecode\Cart\Facades\CartFacade::class,
3 //...
```

1.3. crear modelo

3. Crea el Modelo **Producto** y su archivo de migración en la terminal de *Visual Studio Code*:

```
php artisan make:model Producto -m

# ó, si no funciona:

# sudo docker-compose exec myapp php artisan make:model Producto -m
```

4. Agrega los campos para nuestra migración products en database/migrations algo como 2024_01_12_102939_create_productos_table.php:

```
Schema::create('productos', function (Blueprint $table) {
 1
 2
        $table->id();
        $table->string('name')->unique();
 3
        $table->string('slug')->unique();
 4
        $table->string('details')->nullable();
 5
        $table->double('price');
 6
        $table->double('shipping_cost');
 7
        $table->text('description');
 8
        $table->integer('category_id');
 9
        $table->unsignedInteger('brand_id')->unsigned();
10
11
        $table->string('image_path');
        $table->timestamps();
12
13
   });
```

1.4. crear seeder

5. *Alimentar* datos a la base de datos. Ahora podemos introducir algunos datos en la tabla que hemos creado. Laravel proporciona sembradoras o **seeders** para eso, en la terminal y escribe este comando:

```
php artisan make:seed ProductosTableSeeder
  # ó, si no funciona:
  # sudo docker-compose exec myapp php artisan make:seed
  ProductosTableSeeder
```

6. Ahora modifica el archivo de Seeder que está en database\seeders\ProductosTableSeeder.php (en la cabecera importar Product y dentro de public function run():void{..}):

```
use App\Models\Producto;
use Illuminate\Support\Facades\DB;

//...

DB::table('productos')->insert([
          'name' => 'MacBook Pro',
          'slug' => 'macbook-pro',
          'details' => '15 pulgadas, 1TB HDD, 32GB RAM',
```

```
'price' => 2499.99,
9
           'shipping_cost' => 29.99,
           'description' => 'MackBook Pro',
10
11
          'category_id' => 1,
           'brand_id' => 1,
12
13
           'image_path' => 'macbook-pro.png'
14
        1);
15
        DB::table('productos')->insert([
16
          'name' => 'Dell Vostro 3557',
17
18
           'slug' => 'vostro-3557',
19
          'details' => '15 pulgadas, 1TB HDD, 8GB RAM',
          'price' => 1499.99,
20
21
          'shipping_cost' => 19.99,
           'description' => 'Dell Vostro 3557',
22
23
          'category_id' => 1,
           'brand_id' => 2,
24
           'image_path' => 'dell-v3557.png'
25
26
        ]);
27
28
        DB::table('productos')->insert([
          'name' => 'iPhone 11 Pro',
29
30
           'slug' => 'iphone-11-pro',
31
          'details' => '6.1 pulgadas, 64GB 4GB RAM',
           'price' => 649.99,
32
33
          'shipping_cost' => 14.99,
34
           'description' => 'iPhone 11 Pro',
35
          'category_id' => 2,
36
           'brand_id' => 1,
           'image_path' => 'iphone-11-pro.png'
37
38
        ]);
39
40
        DB::table('productos')->insert([
41
          'name' => 'Remax 610D Headset',
42
           'slug' => 'remax-610d',
          'details' => '6.1 pulgadas, 64GB 4GB RAM',
43
44
          'price' => 8.99,
          'shipping_cost' => 1.89,
45
           'description' => 'Remax 610D Headset',
46
47
          'category_id' => 3,
48
           'brand_id' => 3,
           'image path' => 'remax-610d.jpg'
49
50
        ]);
51
52
        DB::table('productos')->insert([
          'name' => 'Samsung LED TV',
53
54
           'slug' => 'samsung-led-24',
55
          'details' => '24 pulgadas, LED Display, Resolución 1366 x 768',
56
           'price' => 41.99,
57
          'shipping_cost' => 12.59,
58
           'description' => 'Samsung LED TV',
59
          'category_id' => 4,
60
           'brand_id' => 4,
           'image_path' => 'samsung-led-24.png'
61
62
        ]);
63
```

```
64
        DB::table('productos')->insert([
65
           'name' => 'Samsung Camara Digital',
           'slug' => 'samsung-mv800',
66
          'details' => '16.1MP, 5x Optical Zoom',
67
           'price' => 144.99,
68
69
          'shipping_cost' => 13.39,
           'description' => 'Samsung Digital Camera',
70
71
          'category_id' => 5,
           'brand_id' => 4,
72
           'image_path' => 'samsung-mv800.jpg'
73
74
        ]);
75
        DB::table('productos')->insert([
76
          'name' => 'Huawei GR 5 2017',
77
           'slug' => 'gr5-2017',
78
79
          'details' => '5.5 pulgadas, 32GB 4GB RAM',
          'price' => 148.99,
80
          'shipping_cost' => 6.79,
81
           'description' => 'Huawei GR 5 2017',
82
83
          'category_id' => 2,
84
           'brand_id' => 5,
           'image_path' => 'gr5-2017.jpg'
85
86
        ]);
    //...
87
```

7. Ahora editamos el archivo database/seeders/DatabaseSeeder.php y le agregamos la clase:

```
1 | $this->call(ProductosTableSeeder::class);
```

8. Ahora crearemos las tablas de la BD y los productos de prueba:

```
php artisan migrate
pho artisan migrate
sudo docker-compose exec myapp pho artisan migrate
```

9. Ejecutar el seeder:

```
php artisan db:seed --class=ProductosTableSeeder
#6, si no funciona:
#sudo docker-compose exec myapp php artisan db:seed --
class=ProductosTableSeeder
```

1.5. crear controlador

10. Crea un controlador para el carrito:

```
php artisan make:controller CartController

# ó, si no funciona:

# sudo docker-compose exec myapp php artisan make:controller

CartController
```

11. Editamos el archivo CartController.php y le pegamos este código:

```
namespace App\Http\Controllers;
 1
 2
   use Illuminate\Http\Request;
 3
    use App\Models\Producto;
 4
 5
 6
    class CartController extends Controller
 7
8
      public function shop() {
 9
        //Para capturar la excepción que pudiera provocar:
        //try {
10
          $productos = Producto::all();
11
        //} catch (\Exception $e) {
12
        // dd($e->getMessage());
13
        //}
14
15
        return view('shop')->withTitle('E-COMMERCE STORE | SHOP')-
    >with(['productos' => $productos]);
16
      }
17
      public function cart() {
18
        $cartCollection = \Cart::getContent();
19
20
        //dd($cartCollection);
        return view('cart')->withTitle('E-COMMERCE STORE | CART')-
21
    >with(['cartCollection' => $cartCollection]);;
22
      }
23
24
      public function remove(Request $request){
25
        \Cart::remove($request->id);
        return redirect()->route('cart.index')->with('success_msg', 'Item
26
    is removed!');
27
      }
28
29
      public function add(Request$request){
30
        \Cart::add(array(
        'id' => $request->id,
31
        'name' => $request->name,
32
        'price' => $request->price,
33
        'quantity' => $request->quantity,
34
        'attributes' => array(
35
        'image' => $request->img,
36
        'slug' => $request->slug
37
38
        )
39
        ));
        return redirect()->route('cart.index')->with('success_msg',
40
    'Producto agregado a tu carrito!');
41
      }
42
43
      public function update(Request $request){
        \Cart::update($request->id,
44
        array(
45
          'quantity' => array(
46
          'relative' => false,
47
          'value' => $request->quantity
48
49
        ),
```

```
50
51
        ));
        return redirect()->route('cart.index')->with('success_msg', 'El
    carro de compra se ha modificado!');
53
      }
54
55
     public function clear(){
56
        \Cart::clear();
        return redirect()->route('cart.index')->with('success_msg',
57
    'Carrito de compra vaicado!');
58
      }
59
60
    }
```

1.6. crear rutas

12. Creamos las rutas en web.php:

```
1  //...
2  use App\Http\Controllers\CartController;
3  //...
4  Route::get('/', [CartController::class, 'shop'])->name('shop');
5  Route::get('/cart', [CartController::class, 'cart'])-
>name('cart.index');
6  Route::post('/add', [CartController::class, 'add'])-
>name('cart.store');
7  Route::post('/update', [CartController::class, 'update'])-
>name('cart.update');
8  Route::post('/remove', [CartController::class, 'remove'])-
>name('cart.remove');
9  Route::post('/clear', [CartController::class, 'clear'])-
>name('cart.clear');
```

13. Ejecutemos el servidor virtual para probar nuestro proyecto en la terminal:

```
php artisan serve
  # ó, si no funciona:
  # sudo docker-compose exec myapp php artisan serve
```

Si se produce un error en la linea:

```
1 | $productos = Producto::all();
```

de CartController.php podremos activar el *try/catch* comentado en las lineas de código.

Además de:

- comprobar que el usuario de la base de datos tiene privilegios sobre la base de datos.
- limpiar la caché de Laravel:

```
php artisan cache:clear
php artisan config:clear
php artisan config:cache

for each of the config and the confi
```

1.7. crear vistas

- 14. Ahora tenemos que crear dos vistas tanto para el carrito como para las páginas de la tienda en la carpeta resources/views:
 - o shop.blade.php y
 - o cart.blade.php
- resources\views\shop.blade.php :

```
1
   @extends('layouts.app')
2
   @section('content')
3
4
   <div class="container" style="margin-top: 80px">
5
     <nav aria-label="breadcrumb">
       6
         <a href="/">Inicio</a>
7
         Tienda
8
9
       </01>
10
     </nav>
     <div class="row justify-content-center">
11
       <div class="col-lg-12">
12
13
         <div class="row">
14
           <div class="col-lg-7">
15
            <h4>Productos</h4>
          </div>
16
17
         </div>
18
         <hr>
19
         <div class="row">
20
          @foreach($productos as $item)
21
          <div class="col-lg-3">
             <div class="card" style="margin-bottom: 20px; height: auto;">
22
              <img src="/images/{{ $item->image_path }}"
23
24
                   class="card-img-top mx-auto"
25
                   style="height: 150px; width: 150px;display: block;"
                   alt="{{ $item->image_path }}"
26
27
              <div class="card-body">
28
29
                <a href=""><h6 class="card-title">{{ $item->name }}</h6></a>
                {{ $item->price }} €
                <form action="{{ route('cart.store') }}" method="POST">
31
32
                  {{ csrf_field() }}
                  <input type="hidden" value="{{ $item->id }}" id="id"
33
   name="id">
```

```
<input type="hidden" value="{{ $item->name }}" id="name"
34
    name="name">
                     <input type="hidden" value="{{ $item->price }}" id="price"
35
    name="price">
                     <input type="hidden" value="{{ $item->image_path }}" id="img"
36
    name="img">
                     <input type="hidden" value="{{ $item->slug }}" id="slug"
37
    name="slug">
                     <input type="hidden" value="1" id="quantity" name="quantity">
38
                     <div class="card-footer" style="background-color: white;">
39
40
                       <div class="row">
                         <button class="btn btn-secondary btn-sm" class="tooltip-</pre>
41
    test" title="add to cart">
                           <i class="fa fa-shopping-cart"></i> agregar al carrito
42
                         </button>
43
                       </div>
44
                     </div>
45
46
                  </form>
                </div>
47
              </div>
48
49
            </div>
50
          @endforeach
51
        </div>
        </div>
52
53
      </div>
54
    </div>
    @endsection
55
```

• resources\views\cart.blade.php :

```
1
   @extends('layouts.app')
 2
 3
   @section('content')
     <div class="container" style="margin-top: 80px">
 4
       <nav aria-label="breadcrumb">
 5
         6
          <a href="/">Tienda</a>
 7
 8
          Cart
 9
        </nav>
10
       @if(session()->has('success_msg'))
11
         <div class="alert alert-success alert-dismissible fade show"</pre>
12
   role="alert">
           {{ session()->get('success_msg') }}
13
14
           <button type="button" class="close" data-dismiss="alert" aria-</pre>
    label="Close">
            <span aria-hidden="true">x</span>
15
          </button>
16
         </div>
17
       @endif
18
19
       @if(session()->has('alert_msg'))
         <div class="alert alert-warning alert-dismissible fade show"</pre>
   role="alert">
           {{ session()->get('alert_msg') }}
21
```

```
<button type="button" class="close" data-dismiss="alert" aria-</pre>
22
    label="Close">
              <span aria-hidden="true">x</span>
23
24
            </button>
25
          </div>
26
        @endif
27
        @if(count($errors) > 0)
28
          @foreach($errors0>all() as $error)
            <div class="alert alert-success alert-dismissible fade show"</pre>
29
    role="alert">
               {{ $error }}
30
               <button type="button" class="close" data-dismiss="alert" aria-</pre>
31
    label="Close">
                <span aria-hidden="true">x</span>
32
33
               </button>
34
            </div>
          @endforeach
35
36
        @endif
        <div class="row justify-content-center">
37
           <div class="col-lg-7">
38
39
              <br>
40
              @if(\Cart::getTotalQuantity()>0)
                <h4>{{ \Cart::getTotalQuantity()}} Producto(s) en el
41
    carrito</h4><br>
42
             @else
43
                <h4>No hay producto(s) en tu carrito</h4><br>
                <a href="/" class="btn btn-dark">Continua en la tienda</a>
44
45
              @endif
46
47
              @foreach($cartCollection as $item)
               <div class="row">
48
49
                  <div class="col-lg-3">
                    <img src="/images/{{ $item->attributes->image }}"
50
    class="img-thumbnail" width="200" height="200">
51
                  </div>
52
                  <div class="col-lg-5">
53
                    >
                      <b><a href="/shop/{{ $item->attributes->slug }}">{{ $item-
54
    >name }}</a></b><br>
55
                      <b>Precio: </b>{{ $item->price }} €<br>
56
                      <b>Subtotal: </b>{{ \Cart::get($item->id)->getPriceSum()
    }} €<br>
                      {{--}}
                                                            <b>With Discount:
57
    </b>${{ \Cart::get($item->id)->getPriceSumWithConditions() }}--}}
58
                    59
                  </div>
                  <div class="col-lg-4">
60
                    <div class="row">
61
                      <form action="{{ route('cart.update') }}" method="POST">
62
63
                        {{ csrf_field() }}
                        <div class="form-group row">
64
                          <input type="hidden" value="{{ $item->id}}" id="id"
65
    name="id">
66
                          <input type="number" class="form-control form-control-</pre>
    sm" value="{{ $item->quantity }}" id="quantity" name="quantity"
    style="width: 70px; margin-right: 10px;">
```

```
67
                             <button class="btn btn-secondary btn-sm"</pre>
     style="margin-right: 25px;"><i class="fa fa-edit"></i></button>
68
                       </div>
69
                     </form>
                     <form action="{{ route('cart.remove') }}" method="POST">
 71
                        {{ csrf_field() }}
                        <input type="hidden" value="{{ $item->id }}" id="id"
 72
     name="id">
 73
                        <button class="btn btn-dark btn-sm" style="margin-</pre>
     right: 10px;"><i class="fa fa-trash"></i></button>
 74
                     </form>
 75
                   </div>
                 </div>
 76
 77
               </div>
               <hr>
             @endforeach
 79
             @if(count($cartCollection)>0)
80
               <form action="{{ route('cart.clear') }}" method="POST">
81
                 {{ csrf_field() }}
82
                 <button class="btn btn-secondary btn-md">Borrar
83
     Carrito</button>
84
               </form>
85
             @endif
86
           </div>
           @if(count($cartCollection)>0)
87
88
             <div class="col-lg-5">
               <div class="card">
89
90
                 <b>Total: </b>${{
91
     \Cart::getTotal() }}
                 92
93
               </div>
               <br>
94
95
               <a href="/" class="btn btn-dark">Continua en la tienda</a>
               <a href="/checkout" class="btn btn-success">Proceder al
96
     Checkout</a>
97
             </div>
98
           @endif
99
         </div>
100
         <br><br><
101
        </div>
102
     @endsection
```

15. A continuación crear, si no lo tienes, el archivo layouts/app.blade.php:

```
1
   <!doctype html>
2
   <html lang="{{ str_replace('_', '-', app()->getLocale()) }}">
3
   <head>
4
       <meta charset="utf-8">
5
       <meta name="viewport" content="width=device-width, initial-scale=1">
       <meta name="csrf-token" content="{{ csrf_token() }}">
6
7
       <title>{{ $title ?? 'E-COMMERCE TIENDA' }}</title>
       <link rel="stylesheet" href={{ url('css/app.css') }}>
8
```

```
<link rel="stylesheet"</pre>
    href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.cs
    s" integrity="sha384-
    ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQU0hcWr7x9JvoRxT2MZw1T"
    crossorigin="anonymous">
10
        <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/font-</pre>
    awesome/4.7.0/css/font-awesome.min.css">
11
        <link href="https://fonts.googleapis.com/css?family=Nunito"</pre>
    rel="stylesheet">
12
    </head>
13
    <body>
    <div id="app">
14
15
        @include('partials.navbar')
16
        <main class="py-4">
            @yield('content')
17
        </main>
18
    </div>
19
20
        <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"</pre>
    integrity="sha384-
    q8i/X+965Dz00rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
    crossorigin="anonymous"></script>
21
    src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.j
    s" integrity="sha384-
    UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1"
    crossorigin="anonymous"></script>
22
    src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"
    integrity="sha384-
    JjSmVgyd0p3pXB1rRibZUAY0IIy60rQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM"
    crossorigin="anonymous"></script>
23
    src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.3.1/jquery.min.js"
    </script>
24 </body>
25 </html>
```

16. Creas esta carpeta y archivo dentro de views/partials/navbar.blade.php. Código para el menú y agregas el código:

```
1
   <nav class="navbar navbar-expand-md navbar-dark fixed-top bg-dark shadow-sm">
2
     <div class="container">
       <a class="navbar-brand" href="{{ url('/') }}">
         E-COMMERCE TIENDA
4
5
       </a>
       <button class="navbar-toggler" type="button" data-toggle="collapse" data-</pre>
    target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-
    expanded="false" aria-label="{{ __('Toggle navigation') }}">
7
         <span class="navbar-toggler-icon"></span>
8
       <div class="collapse navbar-collapse" id="navbarSupportedContent">
9
10
         class="nav-item">
11
             <a class="nav-link" href="{{ route('shop') }}">TIENDA</a>
12
```

```
14
15
             <a id="navbarDropdown" class="nav-link dropdown-toggle"
               href="#" role="button" data-toggle="dropdown"
16
               aria-haspopup="true" aria-expanded="false"
17
18
19
               <span class="badge badge-pill badge-dark">
20
                 <i class="fa fa-shopping-cart"></i> {{
   \Cart::getTotalQuantity()}}
21
               </span>
22
             </a>
23
24
              <div class="dropdown-menu dropdown-menu-right" aria-
   labelledby="navbarDropdown" style="width: 450px; padding: 0px; border-color:
   #9DA0A2">
25
                26
                 @include('partials.cart-drop')
27
28
              </div>
29
          30
31
       </div>
32
     </div>
33
   </nav>
```

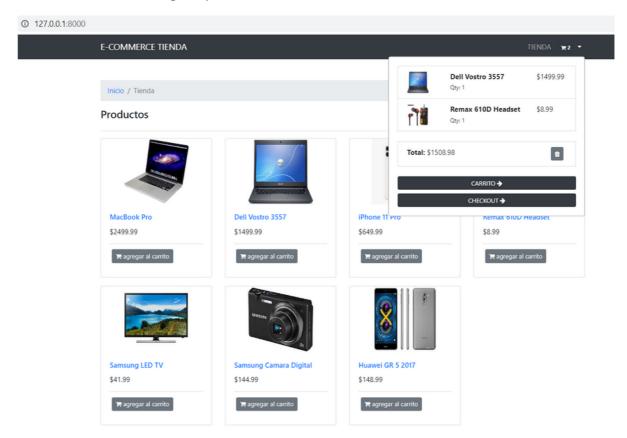
17. partials/cart-drop.blade.php:

```
1
    @if(count(\Cart::getContent()) > 0)
 2
      @foreach(\Cart::getContent() as $item)
       3
         <div class="row">
 4
           <div class="col-lg-3">
 5
             <img src="/images/{{ $item->attributes->image }}"
 6
                  style="width: 50px; height: 50px;"
 7
 8
           </div>
9
           <div class="col-lg-6">
10
             <b>{{$item->name}}</b>
11
12
             <br><small>Qty: {{$item->quantity}}</small>
13
           </div>
           <div class="col-lg-3">
14
             ${{ \Cart::get($item->id)->getPriceSum() }}
15
16
           <hr>
17
         </div>
18
19
       20
     @endforeach
21
      <hr>
     22
23
       <div class="row">
         <div class="col-lg-10">
24
           <b>Total: </b>${{ \Cart::getTotal() }}
25
26
         </div>
         <div class="col-lg-2">
27
           <form action="{{ route('cart.clear') }}" method="POST">
             {{ csrf_field() }}
```

```
<button class="btn btn-secondary btn-sm"><i class="fa fa-trash">
30
    </i></button>
31
           </form>
         </div>
32
33
        </div>
34
     <br>
35
      <div class="row" style="margin: 0px;">
36
37
         <a class="btn btn-dark btn-sm btn-block" href="{{ route('cart.index')}</pre>
    }}">
             CARRITO <i class="fa fa-arrow-right"></i>
38
         </a>
39
         <a class="btn btn-dark btn-sm btn-block" href="">
40
             CHECKOUT <i class="fa fa-arrow-right"></i>
41
42
         </a>
43
     </div>
    @else
44
     Tu carrito esta vacío
45
46
    @endif
```

18. Cargar las imágenes en la carpeta public/images.

Una vez finalizado el código, el producto resultante será:



2. bibliografía

• Enrique Martínez para compucenter33 https://www.youtube.com/c/compucenter33