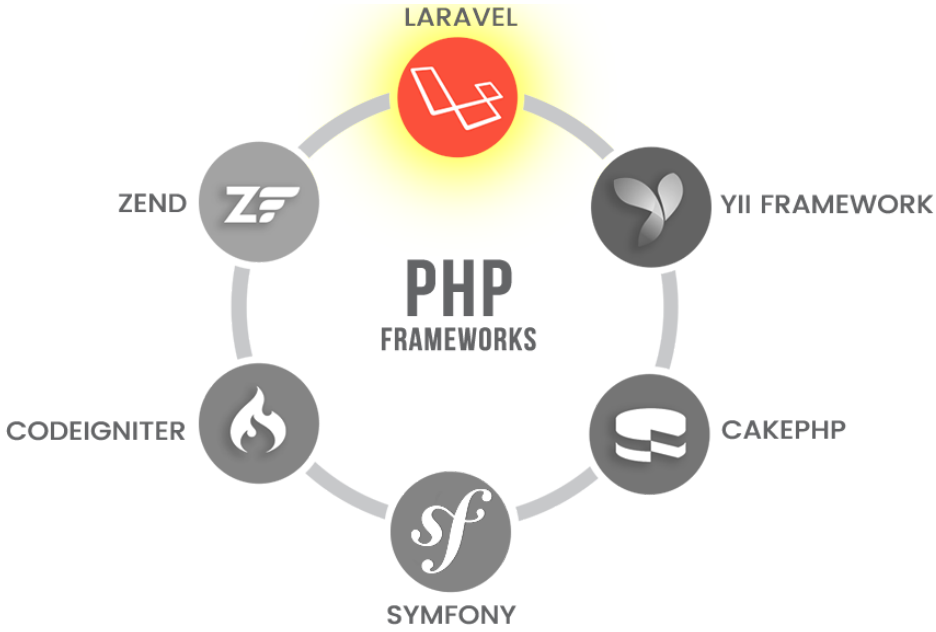
unidad didáctica 7

Laravel - carrito de compra



1. creación de un carrito de compras

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

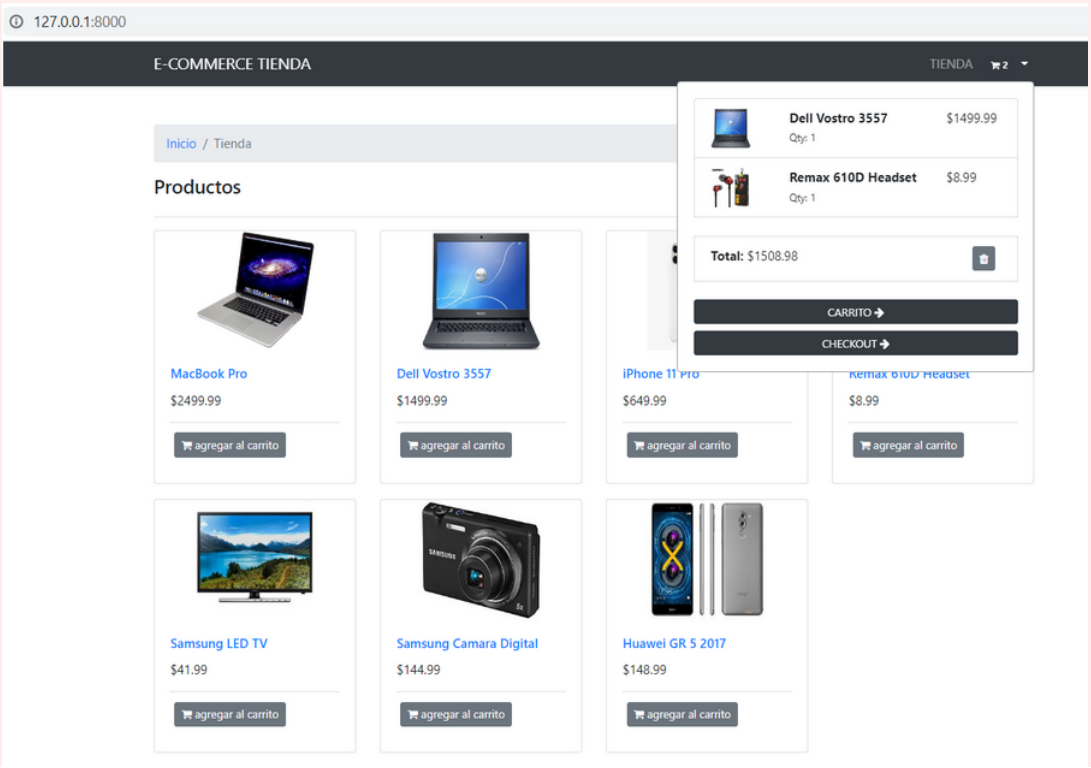
- Descargar Laravel:

```
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



1.1. instalar plugin carrito compra

1. Instalamos el plugin anteriormente mencionado desde la terminal de *Visual Studio Code* ("altura" proyecto) con composer:

```
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:

```
//..
Darryldecode\Cart\CartServiceProvider::class,
//..
```

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

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

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) {
    $table->id();
    $table->string('name')->unique();
    $table->string('slug')->unique();
    $table->string('details')->nullable();
    $table->double('price');
    $table->double('shipping_cost');
    $table->text('description');
    $table->integer('category_id');
    $table->unsignedInteger('brand_id')->unsigned();
    $table->string('image_path');
    $table->timestamps();
});
```

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,
      'shipping_cost' => 29.99,
      'description' => 'MackBook Pro',
      'category_id' => 1,
      'brand_id' => 1,
      'image_path' => 'macbook-pro.png'
    ]);

    DB::table('productos')->insert([
      'name' => 'Dell Vostro 3557',
      'slug' => 'vostro-3557',
      'details' => '15 pulgadas, 1TB HDD, 8GB RAM',
      'price' => 1499.99,
      'shipping_cost' => 19.99,
      'description' => 'Dell Vostro 3557',
      'category_id' => 1,
      'brand_id' => 2,
      'image_path' => 'dell-v3557.png'
    ]);

    DB::table('productos')->insert([
      'name' => 'iPhone 11 Pro',
      'slug' => 'iphone-11-pro',
      'details' => '6.1 pulgadas, 64GB 4GB RAM',
      'price' => 649.99,
      'shipping_cost' => 14.99,
      'description' => 'iPhone 11 Pro',
      'category_id' => 2,
      'brand_id' => 1,
      'image_path' => 'iphone-11-pro.png'
    ]);

    DB::table('productos')->insert([
      'name' => 'Remax 610D Headset',
      'slug' => 'remax-610d',
      'details' => '6.1 pulgadas, 64GB 4GB RAM',
      'price' => 8.99,
      'shipping_cost' => 1.89,
      'description' => 'Remax 610D Headset',
      'category_id' => 3,
      'brand_id' => 3,
      'image_path' => 'remax-610d.jpg'
    ]);

    DB::table('productos')->insert([
      'name' => 'Samsung LED TV',
      'slug' => 'samsung-led-24',
      'details' => '24 pulgadas, LED Display, Resolución 1366 x 768',
      'price' => 41.99,
      'shipping_cost' => 12.59,
      'description' => 'Samsung LED TV',
      'category_id' => 4,
      'brand_id' => 4,
      'image_path' => 'samsung-led-24.png'
    ]);

```

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

```
1  php artisan migrate
2  # ó, si no funciona:
3  # sudo docker-compose exec myapp php artisan migrate
```

9. Ejecutar el seeder:

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

1.5. crear controlador

10. Crea un controlador para el carrito:

```
1  php artisan make:controller CartController
2  # ó, si no funciona:
3  # sudo docker-compose exec myapp php artisan make:controller
   CartController
```

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

```php
namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Models\Producto;

class CartController extends Controller
{
  public function shop() {
    //Para capturar la excepción que pudiera provocar:
    //try {
      $productos = Producto::all();
    //} catch (\Exception $e) {
    //  dd($e->getMessage());
    //}
    return view('shop')->withTitle('E-COMMERCE STORE | SHOP')->with(['productos' => $productos]);
  }

  public function cart() {
    $cartCollection = \Cart::getContent();
    //dd($cartCollection);
    return view('cart')->withTitle('E-COMMERCE STORE | CART')->with(['cartCollection' => $cartCollection]);;
  }

  public function remove(Request $request){
    \Cart::remove($request->id);
    return redirect()->route('cart.index')->with('success_msg', 'Item is removed!');
  }

  public function add(Request$request){
    \Cart::add(array(
    'id' => $request->id,
    'name' => $request->name,
    'price' => $request->price,
    'quantity' => $request->quantity,
    'attributes' => array(
    'image' => $request->img,
    'slug' => $request->slug
    )
    ));
    return redirect()->route('cart.index')->with('success_msg', 'Producto agregado a tu carrito!');
  }

  public function update(Request $request){
    \Cart::update($request->id,
    array(
      'quantity' => array(
      'relative' => false,
      'value' => $request->quantity
    ),
```

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

```
1  php artisan serve
2  # ó, si no funciona:
3  # 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

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

- reiniciar los servidores.

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`:
    - `shop.blade.php` y
    - cart.blade.php

- `resources\views\shop.blade.php` :

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

@section('content')
<div class="container" style="margin-top: 80px">
  <nav aria-label="breadcrumb">
    <ol class="breadcrumb">
      <li class="breadcrumb-item"><a href="/">Inicio</a></li>
      <li class="breadcrumb-item active" aria-current="page">Tienda</li>
    </ol>
  </nav>
  <div class="row justify-content-center">
    <div class="col-lg-12">
      <div class="row">
        <div class="col-lg-7">
          <h4>Productos</h4>
        </div>
      </div>
      <hr>
      <div class="row">
        @foreach($productos as $item)
        <div class="col-lg-3">
          <div class="card" style="margin-bottom: 20px; height: auto;">
            <img src="/images/{{ $item->image_path }}"
                 class="card-img-top mx-auto"
                 style="height: 150px; width: 150px;display: block;"
                 alt="{{ $item->image_path }}"
            >
            <div class="card-body">
              <a href=""><h6 class="card-title">{{ $item->name }}</h6></a>
              <p>{{ $item->price }} €</p>
              <form action="{{ route('cart.store') }}" method="POST">
                {{ csrf_field() }}
                <input type="hidden" value="{{ $item->id }}" id="id"
name="id">
```

```
                <input type="hidden" value="{{ $item->name }}" id="name"
name="name">
                <input type="hidden" value="{{ $item->price }}" id="price"
name="price">
                <input type="hidden" value="{{ $item->image_path }}" id="img"
name="img">
                <input type="hidden" value="{{ $item->slug }}" id="slug"
name="slug">
                <input type="hidden" value="1" id="quantity" name="quantity">
                <div class="card-footer" style="background-color: white;">
                  <div class="row">
                    <button class="btn btn-secondary btn-sm" class="tooltip-
test" title="add to cart">
                      <i class="fa fa-shopping-cart"></i> agregar al carrito
                    </button>
                  </div>
                </div>
              </form>
            </div>
          </div>
        </div>
      @endforeach
    </div>
    </div>
  </div>
</div>
@endsection
```

- `resources\views\cart.blade.php` :

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

@section('content')
  <div class="container" style="margin-top: 80px">
    <nav aria-label="breadcrumb">
      <ol class="breadcrumb">
        <li class="breadcrumb-item"><a href="/">Tienda</a></li>
        <li class="breadcrumb-item active" aria-current="page">Cart</li>
      </ol>
    </nav>
    @if(session()->has('success_msg'))
      <div class="alert alert-success alert-dismissible fade show"
role="alert">
        {{ session()->get('success_msg') }}
        <button type="button" class="close" data-dismiss="alert" aria-
label="Close">
          <span aria-hidden="true">×</span>
        </button>
      </div>
    @endif
    @if(session()->has('alert_msg'))
      <div class="alert alert-warning alert-dismissible fade show"
role="alert">
        {{ session()->get('alert_msg') }}
```

```
        <button type="button" class="close" data-dismiss="alert" aria-label="Close">
          <span aria-hidden="true">×</span>
        </button>
      </div>
    @endif
    @if(count($errors) > 0)
      @foreach($errors0>all() as $error)
        <div class="alert alert-success alert-dismissible fade show" role="alert">
          {{ $error }}
          <button type="button" class="close" data-dismiss="alert" aria-label="Close">
            <span aria-hidden="true">×</span>
          </button>
        </div>
      @endforeach
    @endif
    <div class="row justify-content-center">
       <div class="col-lg-7">
         <br>
         @if(\Cart::getTotalQuantity()>0)
           <h4>{{ \Cart::getTotalQuantity()}} Producto(s) en el carrito</h4><br>
         @else
           <h4>No hay producto(s) en tu carrito</h4><br>
           <a href="/" class="btn btn-dark">Continua en la tienda</a>
         @endif

         @foreach($cartCollection as $item)
           <div class="row">
             <div class="col-lg-3">
               <img src="/images/{{ $item->attributes->image }}" class="img-thumbnail" width="200" height="200">
             </div>
             <div class="col-lg-5">
               <p>
                 <b><a href="/shop/{{ $item->attributes->slug }}">{{ $item->name }}</a></b><br>
                 <b>Precio: </b>{{ $item->price }} €<br>
                 <b>Subtotal: </b>{{ \Cart::get($item->id)->getPriceSum() }} €<br>
                 {{--                                <b>With Discount: </b>${{ \Cart::get($item->id)->getPriceSumWithConditions() }}--}}
               </p>
             </div>
             <div class="col-lg-4">
               <div class="row">
                 <form action="{{ route('cart.update') }}" method="POST">
                   {{ csrf_field() }}
                   <div class="form-group row">
                     <input type="hidden" value="{{ $item->id}}" id="id" name="id">
                     <input type="number" class="form-control form-control-sm" value="{{ $item->quantity }}" id="quantity" name="quantity" style="width: 70px; margin-right: 10px;">
```

```
                        <button class="btn btn-secondary btn-sm"
style="margin-right: 25px;"><i class="fa fa-edit"></i></button>
                    </div>
                  </form>
                  <form action="{{ route('cart.remove') }}" method="POST">
                     {{ csrf_field() }}
                     <input type="hidden" value="{{ $item->id }}" id="id"
name="id">
                     <button class="btn btn-dark btn-sm" style="margin-
right: 10px;"><i class="fa fa-trash"></i></button>
                  </form>
                </div>
              </div>
            </div>
            <hr>
          @endforeach
          @if(count($cartCollection)>0)
            <form action="{{ route('cart.clear') }}" method="POST">
              {{ csrf_field() }}
              <button class="btn btn-secondary btn-md">Borrar
Carrito</button>
            </form>
          @endif
        </div>
        @if(count($cartCollection)>0)
          <div class="col-lg-5">
            <div class="card">
              <ul class="list-group list-group-flush">
                <li class="list-group-item"><b>Total: </b>${{
\Cart::getTotal() }}</li>
              </ul>
            </div>
            <br>
            <a href="/" class="btn btn-dark">Continua en la tienda</a>
            <a href="/checkout" class="btn btn-success">Proceder al
Checkout</a>
          </div>
        @endif
      </div>
      <br><br>
    </div>
@endsection
```

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

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

```
    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.cs
s" integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/font-
awesome/4.7.0/css/font-awesome.min.css">
    <link href="https://fonts.googleapis.com/css?family=Nunito"
rel="stylesheet">
</head>
<body>
<div id="app">
    @include('partials.navbar')
    <main class="py-4">
        @yield('content')
    </main>
</div>
    <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
integrity="sha384-
q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
crossorigin="anonymous"></script>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.j
s" integrity="sha384-
UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1"
crossorigin="anonymous"></script>
    <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"
integrity="sha384-
JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM"
crossorigin="anonymous"></script>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.3.1/jquery.min.js"
</script>
</body>
</html>
```

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

```
<nav class="navbar navbar-expand-md navbar-dark fixed-top bg-dark shadow-sm">
  <div class="container">
    <a class="navbar-brand" href="{{ url('/') }}">
      E-COMMERCE TIENDA
    </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">
      <ul class="navbar-nav ml-auto">
        <li class="nav-item">
          <a class="nav-link" href="{{ route('shop') }}">TIENDA</a>
        </li>
```

```
        <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"
           >
             <span class="badge badge-pill badge-dark">
               <i class="fa fa-shopping-cart"></i> {{
\Cart::getTotalQuantity()}}
             </span>
           </a>

            <div class="dropdown-menu dropdown-menu-right" aria-
labelledby="navbarDropdown" style="width: 450px; padding: 0px; border-color:
#9DA0A2">
              <ul class="list-group" style="margin: 20px;">
                @include('partials.cart-drop')
              </ul>
            </div>
        </li>
      </ul>
    </div>
  </div>
</nav>
```

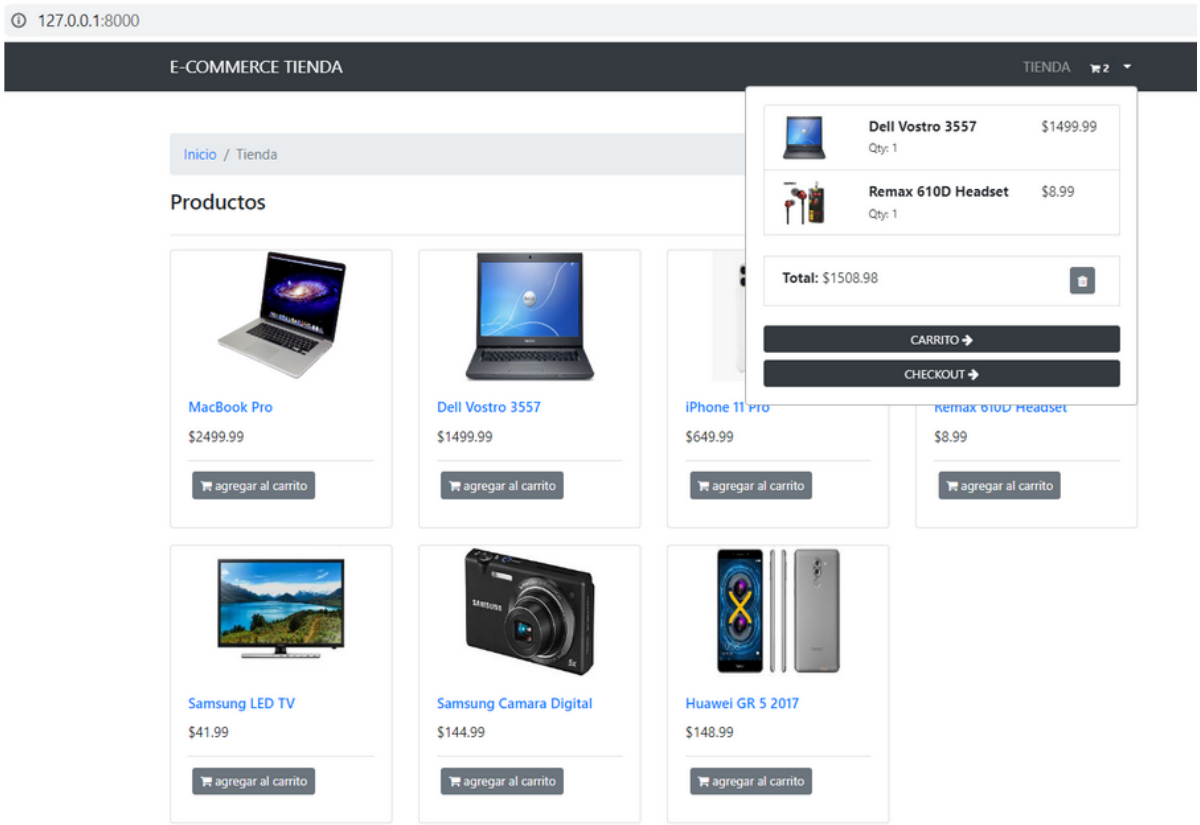
17. `partials/cart-drop.blade.php`:

```
@if(count(\Cart::getContent()) > 0)
  @foreach(\Cart::getContent() as $item)
    <li class="list-group-item">
      <div class="row">
        <div class="col-lg-3">
          <img src="/images/{{ $item->attributes->image }}"
               style="width: 50px; height: 50px;"
          >
        </div>
        <div class="col-lg-6">
          <b>{{$item->name}}</b>
          <br><small>Qty: {{$item->quantity}}</small>
        </div>
        <div class="col-lg-3">
          <p>${{ \Cart::get($item->id)->getPriceSum() }}</p>
        </div>
        <hr>
      </div>
    </li>
  @endforeach
  <br>
  <li class="list-group-item">
    <div class="row">
      <div class="col-lg-10">
        <b>Total: </b>${{ \Cart::getTotal() }}
      </div>
      <div class="col-lg-2">
        <form action="{{ route('cart.clear') }}" method="POST">
          {{ csrf_field() }}
```

```
          <button class="btn btn-secondary btn-sm"><i class="fa fa-trash">
</i></button>
        </form>
      </div>
    </div>
  </li>
  <br>
  <div class="row" style="margin: 0px;">
      <a class="btn btn-dark btn-sm btn-block" href="{{ route('cart.index')
}}">
          CARRITO <i class="fa fa-arrow-right"></i>
      </a>
      <a class="btn btn-dark btn-sm btn-block" href="">
          CHECKOUT <i class="fa fa-arrow-right"></i>
      </a>
  </div>
@else
  <li class="list-group-item">Tu carrito esta vacío</li>
@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