

Atelier 7 : Version 2 de l'application e-commerce (store en ligne) en utilisant Laravel, Vercel et AlwaysData.

1. Introduction

Ce projet consiste à développer un site web e-commerce en utilisant le framework **Laravel**. L'objectif principal est de mettre en pratique l'architecture **MVC (Model – View – Controller)**, la gestion d'une base de données, l'utilisation des seeders, ainsi que l'affichage dynamique des données à l'aide de Blade.

Le site permet d'afficher une collection de produits électroniques classés par catégories telles que : Phones, Laptops, Accessories et Components.

2. Technologies utilisées

- **Langage backend** : PHP
- **Framework** : Laravel
- **Base de données** : MySQL
- **Frontend** : Blade, HTML, CSS
- **ORM** : Eloquent
- **Outils** : Artisan (Laravel CLI)

3. Architecture du projet (MVC)

Le projet suit l'architecture MVC :

- **Model** : Représente les tables de la base de données et permet l'interaction avec les données.

- **Controller** : Contient la logique métier et récupère les données depuis les modèles.
- **View (Blade)** : Affiche les données dynamiquement à l'utilisateur.

Cette architecture permet une bonne organisation du code et facilite la maintenance du projet.

4. Base de données

La base de données contient une table principale nommée **products**.

Champs principaux :

- nom
- prix
- categorie
- image
- solde

Les données sont manipulées à l'aide du modèle **Product**.

5. Seeder et gestion des données

Un seeder nommé **ProductSeeder** est utilisé pour remplir la table **products**.

Le seeder utilise la méthode `updateOrCreate()` afin de :

- Insérer les produits s'ils n'existent pas
- Mettre à jour les produits existants sans créer de doublons

Cela permet de modifier facilement les informations (prix, images, etc.) en relançant le seeder.

Commande utilisée :

```
php artisan db:seed --class=ProductSeeder
```

6. Routing

Les routes sont définies dans le fichier routes/web.php.

Elles permettent de :

- Relier les URL aux méthodes des contrôleurs
- Afficher les pages du site
- Charger les produits depuis la base de données

7. Contrôleur

Le contrôleur (exemple : ProductController) gère :

- La récupération des produits depuis la base de données
- Le regroupement des produits par catégorie
- L'envoi des données vers les vues Blade

Les données sont transmises aux vues à l'aide de compact().

8. Vues Blade (Frontend)

Les vues Blade permettent :

- L'affichage dynamique des produits
- L'utilisation de boucles @foreach
- La réutilisation du layout principal avec @extends

Un layout principal est utilisé pour garantir une structure cohérente sur toutes les pages du site.

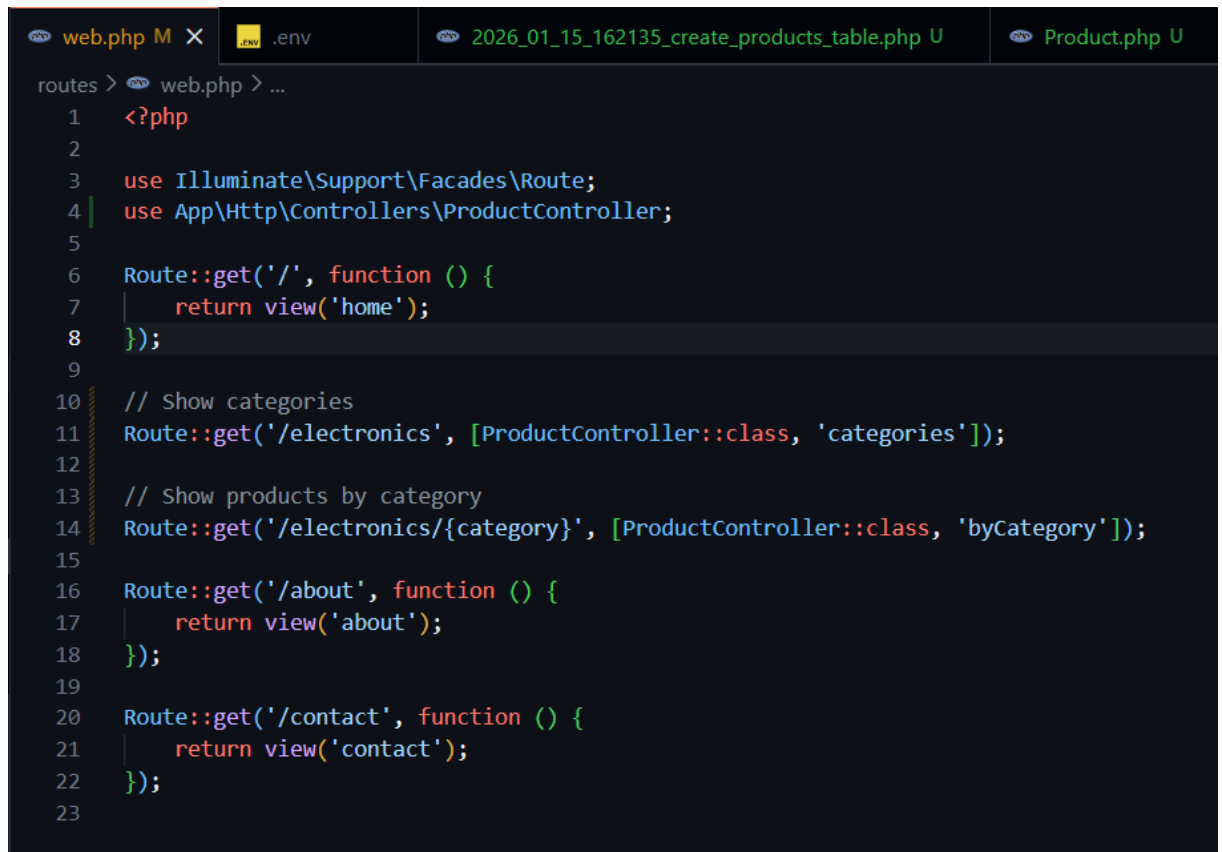
9. Fonctionnement global du site

1. L'utilisateur accède à une URL
2. La route appelle une méthode du contrôleur
3. Le contrôleur récupère les données via le modèle
4. Les données sont envoyées à une vue Blade
5. La page est affichée dynamiquement à l'utilisateur

10. Captures d'écran à inclure dans le rapport

A. Code source

- routes/web.php



```
routes > web.php > ...
1  <?php
2
3  use Illuminate\Support\Facades\Route;
4  use App\Http\Controllers\ProductController;
5
6  Route::get('/', function () {
7      return view('home');
8  });
9
10 // Show categories
11 Route::get('/electronics', [ProductController::class, 'categories']);
12
13 // Show products by category
14 Route::get('/electronics/{category}', [ProductController::class, 'byCategory']);
15
16 Route::get('/about', function () {
17     return view('about');
18 });
19
20 Route::get('/contact', function () {
21     return view('contact');
22 });
23
```

- Modèle Product.php

```
Product.php U X ProductSeeder.php U DatabaseSeeder.php M Control

app > Models > Product.php > ...
1  <?php
2
3  namespace App\Models;
4
5  use Illuminate\Database\Eloquent\Model;
6
7  class Product extends Model
8  {
9      protected $fillable = [
10         'nom',
11         'prix',
12         'categorie',
13         'image',
14         'solde',
15     ];
16 }
17
```

- Seeder ProductSeeder.php

```
ProductSeeder.php U X DatabaseSeeder.php M Controller.php ProductController.php U electronics-categories.blade.php U ele

database > seeders > ProductSeeder.php > ProductSeeder > run
1  <?php
2
3  namespace Database\Seeders;
4
5  use Illuminate\Database\Seeder;
6  use App\Models\Product;
7
8  class ProductSeeder extends Seeder
9  {
10     /**
11      * Run the database seeds.
12      */
13     public function run(): void
14     {
15         $products = [
16             // Phones
17             [
18                 'nom' => 'Samsung Galaxy S25 FE',
19                 'prix' => 700,
20                 'categorie' => 'Phones',
21                 'image' => 'https://images.samsung.com/is/image/samsung/p6pim/n_africa/sm-s731bzkvmwd/gallery/n-africa-gal',
22                 'solde' => 0,
23             ],
24             [
25                 'nom' => 'iPhone 17 Pro',
26                 'prix' => 1099,
27                 'categorie' => 'Phones',
28                 'image' => 'https://istyle.ma/cdn/shop/files/IMG-18064329_m_jpeg_1_9fdfedc9-6392-46f2-a032-dd53e151cfc4.jpg',
29                 'solde' => 0,
30             ],
31             [
32                 'nom' => 'Samsung Galaxy S25 Ultra',
33                 'prix' => 1200,
34                 'categorie' => 'Phones',
35                 'image' => 'https://images.samsung.com/is/image/samsung/p6pim/lb/2501/gallery/lb-galaxy-s25-s938-534365-sm',
36                 'solde' => 0,
37             ],
38         ];
39     }
40 }
```

```

107         ];
108     }
109     foreach ($products as $product) {
110         Product::updateOrCreate(
111             ['nom' => $product['nom']],
112             $product
113         );
114     }
115 }
116 }
117

```

- Contrôleur (exemple : ProductController.php)

```

DatabaseSeeder.php M Controller.php ProductController.php U X electronics-categories.blade.php U
app > Http > Controllers > ProductController.php > ProductController
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use App\Models\Product;
6
7  class ProductController extends Controller
8  {
9      /**
10       * Show all categories
11       */
12     public function categories()
13     {
14         $categories = Product::select('categorie')
15             ->distinct()
16             ->get();
17
18         return view('electronics-categories', [
19             'categories' => $categories
20         ]);
21     }
22
23     /**
24      * Show products by category
25      */
26     public function byCategory($category)
27     {
28         $products = Product::where('categorie', $category)->get();
29
30         return view('electronics-products', [
31             'products' => $products,
32             'category' => $category
33         ]);
34     }
35 }
36

```

- Layout principal Blade (master.blade.php)

```
DatabaseSeeder.php M ProductController.php U master.blade.php X electronics-categories.blade.php U electronics-products.blade.php U
resources > views > layout > master.blade.php > html > body.page-layout > script
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <meta name="description" content="Premium electronics store - High-quality electronic products for modern lifestyles">
7
8   <title>@yield('title', 'MSONE') - Premium Electronics</title>
9
10  <link rel="stylesheet" href="{{ asset('css/style.css') }}">
11 </head>
12 <body class="page-layout">
13
14   @include('partials.menu')
15
16   <main class="page-content">
17     <div class="container">
18       @yield('content')
19     </div>
20   </main>
21
22   @include('partials.footer')
23
24   <script>
25     // Navbar scroll effect
26     window.addEventListener('scroll', function() {
27       const navbar = document.querySelector('.navbar');
28       if (navbar && window.scrollY > 10) {
29         navbar.classList.add('scrolled');
30       } else if (navbar) {
31         navbar.classList.remove('scrolled');
32       }
33     });
34   </script>
35 </body>
36 </html>
37
```

B. Base de données





































- Structure de la table products

```

2026_01_15_162135_create_products_table.php X Product.php U ProductSeeder.php U DatabaseSeeder.php M
database > migrations > 2026_01_15_162135_create_products_table.php > class > up > Closure
1  <?php
2
3  use Illuminate\Database\Migrations\Migration;
4  use Illuminate\Database\Schema\Blueprint;
5  use Illuminate\Support\Facades\Schema;
6
7  return new class extends Migration
8  {
9      /**
10       * Run the migrations.
11       */
12     public function up(): void
13     {
14         Schema::create('products', function (Blueprint $table) {
15             $table->id();
16             $table->string('nom', 200);
17             $table->double('prix');
18             $table->string('categorie', 200);
19             $table->text('image')->nullable();
20             $table->boolean('solde')->nullable();
21             $table->timestamps();
22         });
23     }
24
25     /**
26      * Reverse the migrations.
27      */
28     public function down(): void
29     {
30         Schema::dropIfExists('products');
31     }
32 };
33

```

- Données présentes dans la table

			id	nom	prix	categorie	image	solde	created_at	updated_at	
<input type="checkbox"/>	 Edit	 Copy	 Delete	1	Samsung Galaxy S25 FE	700	Phones	https://images.samsung.com/is/image/samsung/p6pim/...	0	NULL	2026-01-16 07:50:42
<input type="checkbox"/>	 Edit	 Copy	 Delete	2	iPhone 17 Pro	1099	Phones	https://istyle.ma/cdn/shop/files/IMG-18064329_m.jp...	0	NULL	NULL
<input type="checkbox"/>	 Edit	 Copy	 Delete	3	Samsung Galaxy S25 Ultra	1200	Phones	https://images.samsung.com/is/image/samsung/p6pim/...	0	NULL	2026-01-16 07:48:59
<input type="checkbox"/>	 Edit	 Copy	 Delete	4	Dell Inspiron 16 7640 2-in-1	1300	Laptops	https://m.media-amazon.com/images/I/51fpCMmuoML.jp...	0	NULL	NULL
<input type="checkbox"/>	 Edit	 Copy	 Delete	5	Samsung Galaxy Book5 Pro 360	1600	Laptops	https://images.samsung.com/is/image/samsung/p6pim/...	0	NULL	NULL
<input type="checkbox"/>	 Edit	 Copy	 Delete	6	MacBook Pro M3 Max	3199	Laptops	https://store.storeimages.cdn-apple.com/1/as-image...	0	NULL	2026-01-15 23:19:28
<input type="checkbox"/>	 Edit	 Copy	 Delete	7	Logitech MX Master 3S	140	Accessories	https://resource.logitech.com/c_fill,q_auto,f_auto...	0	NULL	NULL
<input type="checkbox"/>	 Edit	 Copy	 Delete	8	Bose SoundLink Micro Bluetooth Speaker	89	Accessories	https://assets.bosecreative.com/transform/ecc6a491...	0	NULL	NULL
<input type="checkbox"/>	 Edit	 Copy	 Delete	9	Samsung Galaxy Buds 3	149	Accessories	https://media.electroplanet.ma/media/catalog/produ...	0	NULL	NULL
<input type="checkbox"/>	 Edit	 Copy	 Delete	10	Samsung SSD 980 M.2 PCIe NVMe 1TB	110	Components	https://www.ultrapc.ma/16764-large_default/samsung...	0	NULL	2026-01-15 23:22:14
<input type="checkbox"/>	 Edit	 Copy	 Delete	11	Asus Laptop Fan	60	Components	https://5.imimg.com/data5/ZB/GQ/HY/SELLER-98115099...	0	NULL	NULL
<input type="checkbox"/>	 Edit	 Copy	 Delete	12	Dell Laptop Charger 65W USB Type C	40	Components	https://wpsitesdata.blob.core.windows.net/pctintas...	0	NULL	2026-01-15 23:48:09

11. Conclusion

Ce projet a permis de comprendre le fonctionnement d'une application Laravel complète, de la base de données jusqu'à l'interface utilisateur.

L'utilisation des seeders et de l'architecture MVC a facilité l'organisation du code et la gestion des données.

Le site est fonctionnel, évolutif et respecte les bonnes pratiques du framework Laravel.