

Atlier 8: Ajouter un nouveau produit dans Ecommerce

1- Creation du Resource controller RProduitController :

```
● hodaifa@HodaifaPC:~/everything/laravel/Ecommerce$ php artisan make:controller RProduitController --resource  
INFO Controller [app/Http/Controllers/RProduitController.php] created successfully.  
❖ hodaifa@HodaifaPC:~/everything/laravel/Ecommerce$
```

2- Create des routes dans web.php :

```
22  
23 | Route::resource('produits', RProduitController::class);  
24  
25 | Route::get('/produits/create',[RProduitController::class,'create']->name('create'));
```

3- Modifications dans .env pour Cloudinary :

```
67  
68 CLOUDINARY_CLOUD_NAME=dq3ujv1w5  
69 CLOUDINARY_API_KEY=488627473343982  
70 CLOUDINARY_API_SECRET=iBfR5kTwgmCZqqXp9K9ZJ_cSGTk
```

4- Installer le SDK Cloudinary pour PHP :

```
● hodaifa@HodaifaPC:~/everything/laravel/Ecommerce$ php artisan config:clear  
INFO Configuration cache cleared successfully.  
  
● hodaifa@HodaifaPC:~/everything/laravel/Ecommerce$ composer require cloudinary/cloudinary_php  
./composer.json has been updated  
Running composer update cloudinary/cloudinary_php  
Loading composer repositories with package information  
Updating dependencies  
Lock file operations: 2 installs, 0 updates, 0 removals  
- Locking cloudinary/cloudinary_php (3.1.2)  
- Locking cloudinary/transformation-builder-sdk (2.1.2)  
Writing lock file  
Installing dependencies from lock file (including require-dev)  
Package operations: 2 installs, 0 updates, 0 removals
```

5- Form dans addProduit.blade.php (/produits/create) :

```
<div class="container">
  <div class="row justify-content-center">
    <div class="col-md-8">
      <div class="card">
        <div class="card-body">
          <form method="POST" action="/produits" enctype="multipart/form-data">
            @csrf

            <div class="form-group">
              <label for="nom">{{ __('Nom du produit') }}</label>
              <input id="nom" type="text" class="form-control" name="nom">
              @error('nom')
                {{ $message }}
              @enderror
            </div>

            <div class="form-group">
              <label for="prix">{{ __('Prix du produit') }}</label>
              <input id="prix" type="text" class="form-control" name="prix">
              @error('prix')
                {{ $message }}
              @enderror
            </div>

            <div class="form-group">
              <label for="categorie">{{ __('Catégorie du produit') }}</label>
              <input id="categorie" type="text" class="form-control" name="categorie">
              @error('categorie')
                {{ $message }}
              @enderror
            </div>

            <div class="form-group">
              <label for="image">{{ __('Image du produit') }}</label>
              <input id="image" type="file" class="form-control-file" name="image">
            </div>

            <div class="form-group mb-0">
              <button type="submit" class="btn btn-primary">
                {{ __('Ajouter le produit') }}
              </button>
            </div>
          </form>
        </div>
      </div>
    </div>
  </div>
</div>
```

6- Dans view/incs/flash.blade.php :

```
@if($message=Session::get('success'))

  <div class="alert alert-success alert-block">
    <button type="button" class="close" data-dismiss="alert">X</button>
    <strong> {{ $message }}</strong>
  </div>

@endif
```

7- Creation de reqest 'AddProductRequest.php' :

```
● houdaifa@HoudaifaPC:~/everything/laravel/Ecommerce$ php artisan make:request AddProductRequest.php

INFO Request [app/Http/Requests/AddProductRequest.php] created successfully.

❖ houdaifa@HoudaifaPC:~/everything/laravel/Ecommerce$
```

8- Modifications des rules et des messages dans 'AddProductRequest' :

```
public function rules(): array
{
    return [
        'nom'=>'required|min:5',
        'prix'=>'required',
        'categorie'=>'required|min:5'
    ];
}

// msg error
public function messages()
{
    return [
        'nom.required'=>'name is required',
        'nom.min'=>'too short enter more',
        'prix.required'=>'price is required',
        'categorie.min'=>'too short enter more',
        'categorie.required'=>'category is required'
    ];
}
```

9- Validation, upload au Cloudinary, et enregistrement dans la BDD :

```
public function store(Request $request)
{
    $request->validated();
    // 2) Upload to Cloudinary
    $cloudinary = new Cloudinary([
        'cloud' => [
            'cloud_name' => env('CLOUDINARY_CLOUD_NAME'),
            'api_key' => env('CLOUDINARY_API_KEY'),
            'api_secret' => env('CLOUDINARY_API_SECRET'),
        ],
    ]);

    $result = $cloudinary->uploadApi()->upload(
        $request->file('image')->getRealPath(),
        [
            'folder' => 'produits',
        ]
    );

    $imageUrl = $result['secure_url'];

    // 3) Save en DB
    $produit = new Produit();
    $nom=$request->input('nom');
    $prix=$request->input('prix');
    $categorie=$request->input('categorie');

    $produit->nom=$nom;
    $produit->prix=$prix;
    $produit->categorie=$categorie;

    // adapte le nom de colonne selon ta DB :
    // ex: image, image_url, photo, etc.
    $produit->image = $imageUrl;

    $produit->save();

    // 4) Redirect
    return back()->with('success','You have successfully added a new Product.');
```