

1. Prérequis

Avant d'installer Laravel, assurez-vous d'avoir :

- **PHP** \geq 8.1 installé
- **Composer** installé : <https://getcomposer.org>
- Un serveur local (facultatif mais recommandé) comme **XAMPP**, **Laragon**, ou **MAMP**

2. Installer Laravel via Composer

a) Installer l'installateur global de Laravel:

composer global require laravel/installer

b) Créer un nouveau projet Laravel :

laravel new mon_projet

OU, si vous n'avez pas installé l'installateur global :

composer create-project laravel/laravel mon_projet

3. Lancer le serveur Laravel

Une fois dans le dossier de votre projet :

```
cd mon_projet  
php artisan serve
```

Cela démarre un serveur de développement sur :

<http://localhost:8000>

4. Structure globale

Quand vous ouvrez un projet **Laravel**, vous verrez principalement ces dossiers et fichiers :

```
mon_projet/  
├── app/  
├── bootstrap/  
├── config/  
├── database/  
├── public/  
├── resources/  
├── routes/  
├── storage/  
└── tests/  
├── .env  
└── artisan  
└── composer.json  
...
```

app/

Contient **la logique métier** de l'application.

- **Console/** : commandes artisan personnalisées
- **Exceptions/** : gestion des erreurs/exceptions
- **Http/** :
 - **Controllers/** : les contrôleurs qui gèrent les requêtes HTTP
 - **Middleware/** : filtres de requêtes
 - **Requests/** : classes de validation
- **Models/** : les modèles Eloquent qui représentent les tables de base de données

bootstrap/

Dossier de **démarrage de l'application**.

- **app.php** initialise l'application
- Le dossier **cache/** contient les fichiers de cache générés

config/

Contient tous les fichiers de **configuration** du framework et des packages :

- **app.php, database.php, mail.php, auth.php**, etc.

Chaque fichier correspond à une partie de l'application configurable.

database/

Tout ce qui concerne la **base de données** :

- **migrations/** : fichiers pour créer/modifier les tables
- **factories/** : pour générer des données fictives (fake)
- **seeders/** : pour insérer des données de test dans la BDD

public/

Le **point d'entrée** de votre application.

- Contient **index.php** (le routeur principal)
- Contient les **fichiers accessibles publiquement** : CSS, JS, images...

resources/

Contient toutes les **ressources non compilées** :

- **views/** : les fichiers Blade (.blade.php)
- **lang/** : fichiers de traduction

- `css/`, `js/` (si vous utilisez Laravel Mix ou Vite)

routes/

Contient tous les fichiers de **routes**.

- `web.php` : routes pour les pages web (avec vues)
- `api.php` : routes pour les API REST
- `console.php` : pour les commandes artisan personnalisées
- `channels.php` : pour les notifications en temps réel

storage/

Contient les fichiers générés par l'application :

- `logs/` : fichiers de log
- `app/` : fichiers uploadés
- `framework/` : cache, sessions, vues compilées...

 **Important** : ce dossier doit être **accessible en écriture** par le serveur.

tests/

Contient les **tests automatisés** (PHPUnit).

- `Feature/` : tests plus larges, simulent des requêtes HTTP
- `Unit/` : tests unitaires classiques

.env

Fichier de **configuration des variables d'environnement**.

Exemple :

```
APP_NAME=Laravel
APP_ENV=local
DB_DATABASE=laravel
DB_USERNAME=root
DB_PASSWORD=
```

artisan

Le **terminal interne de Laravel**.

Permet de lancer des commandes :

```
php artisan make:controller
php artisan migrate
php artisan serve
```

composer.json

Définit les **dépendances PHP** de votre projet (packages), les scripts, l'autoload, etc.

Lab_1 :

Objectif : tester l'installation et créer votre premier projet Laravel pour afficher un message de bienvenue.

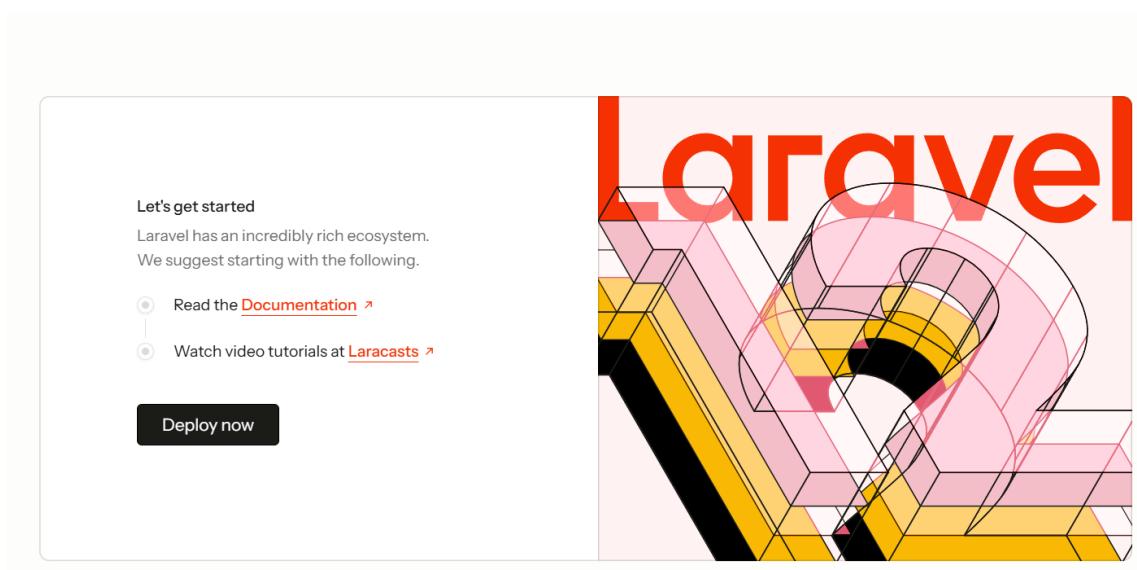
Suivre les étapes suivantes :

- 1- Pour éviter tout problème de téléchargement, désactiver votre antivirus pendant l'installation.
- 2- Créez un dossier Projet_affichage
- 3- Cmd : laravel new crud

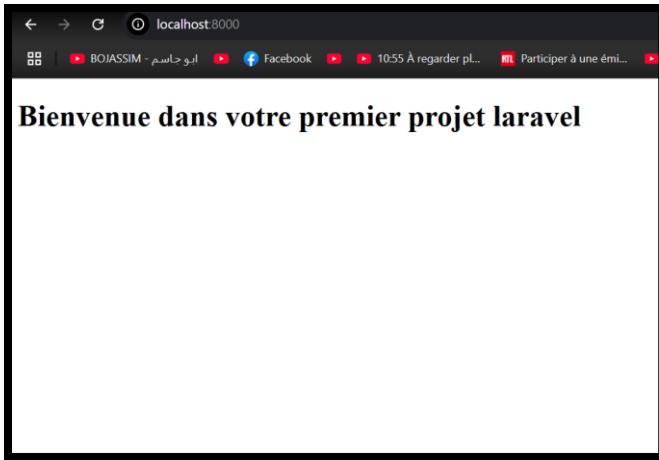
On va appeler le projet crud.

```
Laravel [ ]  
Would you like to install a starter kit? [No starter kit]:  
[none] No starter kit  
[breeze] Laravel Breeze  
[jetstream] Laravel Jetstream
```

- 4- Tapez entrer pour passer à l'étape suivante
 - 5- Pour les tests on va accepter l'option par défaut qui est Pest, donner entrer pour passer
- ```
Which testing framework do you prefer? [Pest]:
[0] Pest
[1] PHPUnit
```
- 6- L'étape suivante consiste à choisir le type de base de données, laissez par défaut sqlite.
  - 7- A la fin de l'installation, faites cd crud et lancer le serveur par **php artisan serve**
  - 8- La page de laravel 12 s'affiche, donc tout va bien !



- 9- Dans vscode, vous aurez l'arborescence avec le fichier **welcome.blade.php**, toutes les vues ont du type .blade.php, changez le nom de cette vue en **index.blade.php**
- 10- Effacer le contenu de cette vue et remplacez là par un affichage de bienvenue.
- 11- Comme le nom du fichier a changé, il faut changer le nom dans route/web.php



## II. Création d'un premier modèle de données et afficher les données sur une vue

### 1. Créer un modèle + migration

Dans votre terminal, tapez :

```
php artisan make:model Produit -m
```

Cela crée :

- `app/Models/Produit.php` → le **modèle**
- `database/migrations/xxxx_create_produits_table.php` → la **migration**

modifiez le fichier `Modles/Produit.php` en

```
<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Model;

class Produit extends Model
{
 //
 protected $fillable = [
 'nom',
 'prix',
];
}
```

### 2. Modifier la migration

Ouvrez le fichier dans `database/migrations/` et ajoutez des colonnes par exemple :

```
public function up()
{
 Schema::create('produits', function (Blueprint $table) {
 $table->id();
 $table->string('nom');
 $table->decimal('prix', 8, 2);
 $table->timestamps();
 });
}
```

```
 });
}
```

### 3. Lancer la migration

```
php artisan migrate
```

Cela crée la table `produits` dans la base de données.

### 4. Ajouter des données (en dur ou via un seeder)

#### Dans un contrôleur

Ouvrez ou créez `AccueilController.php`, puis :

Pour créer le Controller : `php artisan make:controller AccueilController`

```
use App\Models\Produit;
public function index()
{
 // Ajouter des produits si la table est vide
 if (Produit::count() == 0) {
 Produit::create(['nom' => 'Ordinateur', 'prix' => 1200]);
 Produit::create(['nom' => 'Souris', 'prix' => 15]);
 }

 // Récupérer tous les produits
 $produits = Produit::all();

 // Envoyer à la vue
 return view('index', compact('produits'));
}
```

### 5. Afficher les données dans `resources/views/index.blade.php`

```
<!DOCTYPE html>
<html>
<head>
 <title>Liste des Produits</title>
</head>
<body>
 <h1>Produits disponibles :</h1>

 @foreach ($produits as $produit)
 {{ $produit->nom }} - {{ $produit->prix }} €
 @endforeach

</body>
</html>
```

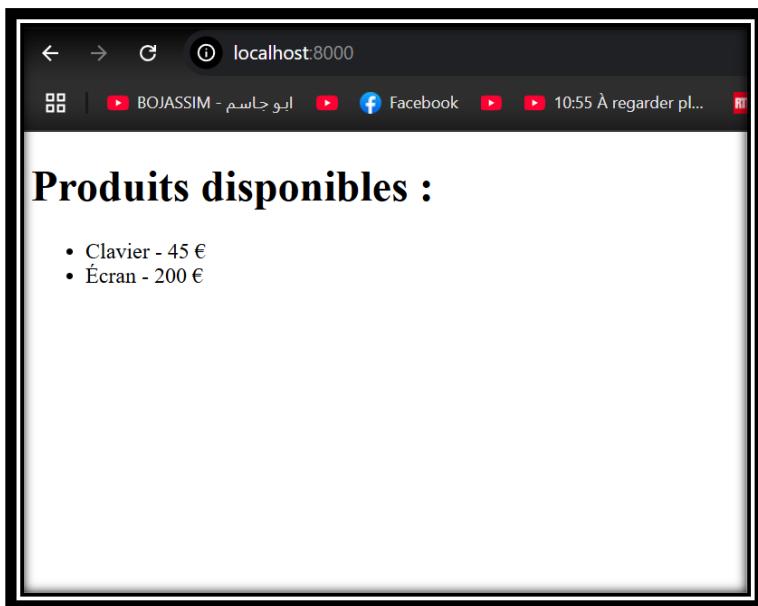
### 6. Tester

Lancez : `php artisan serve`

Puis visitez : `http://localhost:8000`

Vous verrez la liste des produits !

Et vous aurez le résultat suivant :



- Clavier - 45 €
- Écran - 200 €