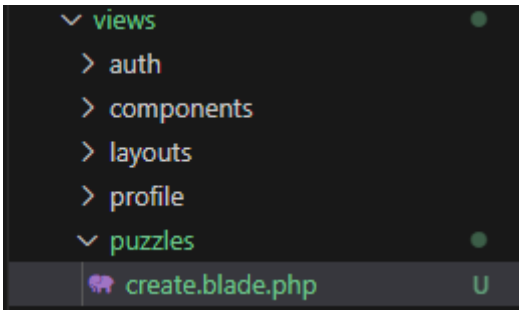
Développement Web
Laravel 10
CRUD

Create

La création d'un nouveau Puzzle nécessite 2 étapes : le renseignement des données du puzzle (via un formulaire) puis l'enregistrement en base. Nous allons voir ensemble ces 2 étapes.

Formulaire

Ont créé la vue create



On y renseigne ce code

create.blade.php U

resources > views > puzzles > create.blade.php > x-app-layout > x-puzzles-card > form

```
<x-app-layout>
    <!-- Définition de l'en-tête de la page -->
    <x-slot name="header">
        <h2 class="font-semibold text-xl text-gray-800 leading-tight">
            {{ __('Create a puzzle') }}
        </h2>
    </x-slot>

    <x-puzzles-card>
        <!-- Message de réussite -->
        <!-- Si une session contient un message (comme un succès après la soumission du formulaire), celui-ci est affiché -->
        @if (session()->has('message'))
            <div class="mt-3 mb-4 list-disc list-inside text-sm text-green-600">
                {{ session('message') }}
            </div>
        @endif

        <!-- Formulaire pour la création d'un puzzle -->
        <form action="{{ route('puzzles.store') }}" method="POST">
            @csrf

            <!-- Champ pour le nom du puzzle -->
            <div>
                <x-input-label for="nom" :value="__('Nom')" />
                <x-text-input id="nom" class="block mt-1 w-full" type="text" name="nom" :value="old('nom')" required autofocus />
                <x-input-error :messages="$errors->get('nom')" class="mt-2" />
            </div>

            <!-- Champ pour la catégorie du puzzle -->
            <div>
                <x-input-label for="categorie" :value="__('Catégorie')" />
                <select id="categorie" name="categorie" class="block mt-1 w-full" required autofocus>
                    <option value="" disabled selected>Choisissez une catégorie</option>
                    <option value="3D">3D</option>
                    <option value="Art">Art</option>
                    <option value="Nature">Nature</option>
                    <option value="Architecture">Architecture</option>
                <!-- Ajoutez d'autres options selon vos catégories -->
                </select>
                <x-input-error :messages="$errors->get('categorie')" class="mt-2" />
            </div>


            <!-- Champ pour la description du puzzle -->
            <div>
                <x-input-label for="description" :value="__('Description')" />
                <x-text-input id="description" class="block mt-1 w-full" type="text" name="description" :value="old('description')" required autofocus />
                <x-input-error :messages="$errors->get('description')" class="mt-2" />
            </div>
```

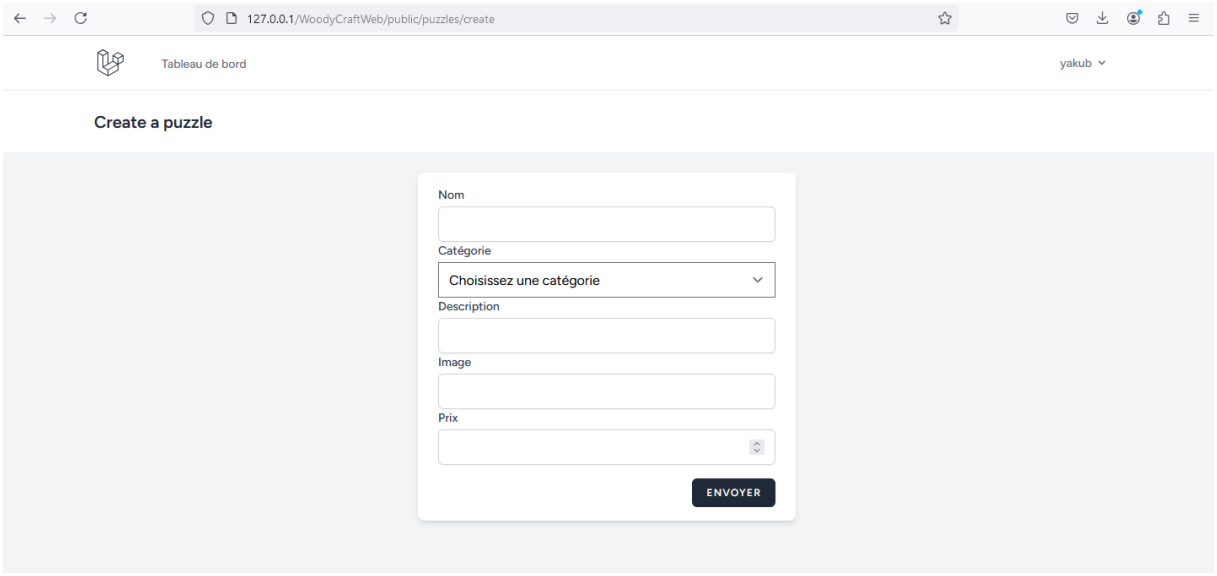
–> Selon vous, à quoi sert ce code ?

C'est la view du formulaire

On va compléter le code de la fonction create du contrôleur PuzzleControleur pour appeler la vue :

```php
/**
 * Afficher le formulaire pour créer une nouvelle ressource (puzzle).
 */
public function create()
{
    return view('puzzles.create');
}
```

Maintenant avec l'url http://localhost/woodycraftweb/public/puzzles/create on obtient la page avec le formulaire :



→Que se passe-t-il si vous remplissez les champs et que vous cliquez sur "Envoyer" ?

Rien

Enregistrement

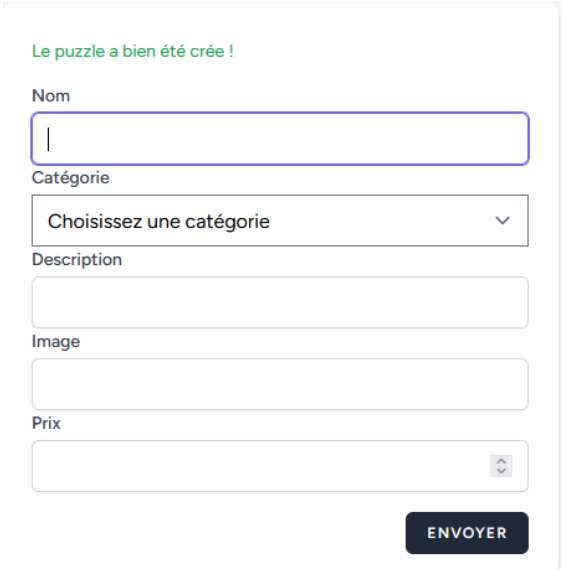
Nous devons maintenant enregistrer ce nouveau puzzle dans la base de données.

Pour cela, nous allons compléter la méthode store() du contrôleur.

```php
/**
 * Enregistrer une nouvelle ressource dans la base de données.
 */
public function store(Request $request)
{
    $date = $request->validate([
        'nom' => 'required|max:100',
        'categorie' => 'required|max:500',
        'description' => 'required|max:500',
        'image' => 'required|max:100',
        'prix' => 'required|numeric|between:0,99,99',
    ]);

    $puzzles = new Puzzle();
    $puzzles->nom = $request->nom;
    $puzzles->categorie = $request->categorie;
    $puzzles->description = $request->description;
    $puzzles->image = $request->image;
    $puzzles->prix = $request->prix;
    $puzzles->save();
    return back()->with('message', "Le puzzle a bien été crée !");
}
```

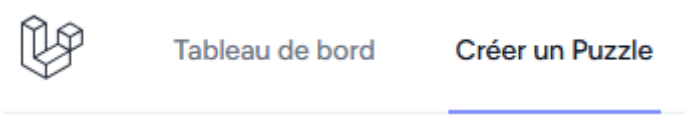
Vérifiez que ça fonctionne (côté application et dans la base).



yakub.puzzles: 1 ligne(s) au total (environ) Suivant

id	nom	categorie	description	image	prix	created_at	updated_at
1	puzz 1	Art	puzz 1	puzz 1	2,13	2024-09-23 09:47:28	2024-09-23 09:47:28

Nous souhaitons maintenant pouvoir accéder à ce formulaire directement depuis le menu. Pour cela, ajoutez un nouveau lien de navigation dans la vue views/layouts/navigation.blade.php.



Update

edit.blade.php U | PuzzleController.php M | create.blade.php U

resources > views > puzzles > edit.blade.php

```blade
<x-app-layout>
    <x-slot name="header">
        <h2 class="font-semibold text-xl text-gray-800 leading-tight">
            {{ __('Éditer un puzzle') }}
        </h2>
    </x-slot>

    <x-puzzles-card>
        <!-- Message de réussite -->
        @if (session()->has('message'))
            <div class="mt-3 mb-4 text-sm text-green-600">
                {{ session('message') }}
            </div>
        @endif

        <form action="{{ route('puzzles.update', $puzzle->id) }}" method="POST">
            @csrf
            @method('PUT')

            <!-- Nom -->
            <div>
                <x-input-label for="nom" :value="__('Nom')" />
                <x-text-input id="nom" class="block mt-1 w-full" type="text"
                              name="nom" :value="old('nom', $puzzle->nom)" required autofocus />
                <x-input-error :messages="$errors->get('nom')" class="mt-2" />
            </div>

            <!-- Catégorie -->
            <div>
                <x-input-label for="categorie" :value="__('Catégorie')" />
                <select id="categorie" name="categorie" class="block mt-1 w-full" required>
                    <option value="" disabled {{ old('categorie', $puzzle->categorie) === null ? 'selected' : '' }}>Choisissez une catégorie</option>
                    <option value="3D" {{ old('categorie', $puzzle->categorie) == '3D' ? 'selected' : '' }}>3D</option>
                    <option value="Art" {{ old('categorie', $puzzle->categorie) == 'Art' ? 'selected' : '' }}>Art</option>
                    <option value="Nature" {{ old('categorie', $puzzle->categorie) == 'Nature' ? 'selected' : '' }}>Nature</option>
                    <option value="Architecture" {{ old('categorie', $puzzle->categorie) == 'Architecture' ? 'selected' : '' }}>Architecture</option>
                <!-- Ajoutez d'autres options selon vos catégories -->
                </select>
                <x-input-error :messages="$errors->get('categorie')" class="mt-2" />
            </div>


            <!-- Description -->
            <div class="mt-4">
                <x-input-label for="description" :value="__('Description')" />
                <x-text-input id="description" class="block mt-1 w-full" type="text"
                              name="description" :value="old('description', $puzzle->description)" required />
                <x-input-error :messages="$errors->get('description')" class="mt-2" />
            </div>
```

Dans le contrôleur, on met à jour la méthode edit() :

```php
/**
 * Afficher le formulaire pour éditer une ressource spécifique (puzzle).
 */
public function edit(Puzzle $puzzle)
{
    return view('puzzles.edit',compact('puzzle'));
}
```

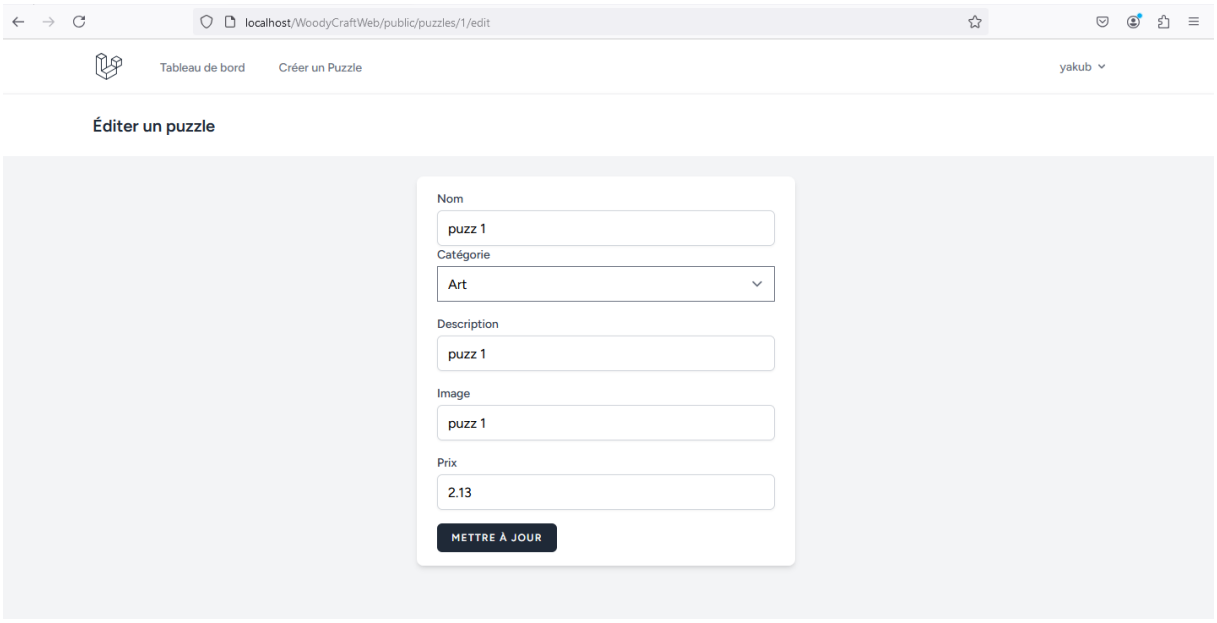
Enregistrement

```php
/**
 * Mettre à jour une ressource spécifique dans la base de données.
 */
public function update(Request $request, Puzzle $puzzle)
{
    // Validation des données
    $data = $request->validate([
        'nom' => 'required|max:100',
        'categorie' => 'required|max:100',
        'description' => 'required|max:500',
        'image' => 'required|max:500',
        'prix' => 'required|numeric|between:0,99.99',
    ]);

    // Mise à jour des attributs du puzzle
    $puzzle->nom = $request->nom;
    $puzzle->categorie = $request->categorie;
    $puzzle->description = $request->description;
    $puzzles->image = $request->image;
    $puzzle->prix = $request->prix;

    // Sauvegarder les changements
    $puzzle->save();

    // Redirection ou retour avec un message de succès
    return redirect()->route('puzzles.index')->with('message', 'Puzzle mis à jour avec succès.');
}
```



Read

show.blade.php

resources > views > puzzles > show.blade.php

```blade
<x-app-layout>
    <x-slot name="header">
        <h2 class="font-semibold text-xl text-gray-800 leading-tight">
            @lang('Afficher un puzzle')
        </h2>
    </x-slot>

    <x-puzzles-card>
        <h3 class="font-semibold text-xl text-gray-800"> @lang('Nom') </h3>
        <p>{{ $puzzle->nom }}</p>

        <h3 class="font-semibold text-xl text-gray-800 pt-2"> @lang('Catégorie') </h3>
        <p>{{ $puzzle->categorie }}</p>

        <h3 class="font-semibold text-xl text-gray-800 pt-2"> @lang('Description') </h3>
        <p>{{ $puzzle->description }}</p>

        <h3 class="font-semibold text-xl text-gray-800 pt-2"> @lang('image') </h3>
        <p>{{ $puzzle->image }}</p>

        <h3 class="font-semibold text-xl text-gray-800 pt-2"> @lang('prix') </h3>
        <p>{{ $puzzle->prix }}</p>

        <h3 class="font-semibold text-xl text-gray-800 pt-2"> @lang('Date de création') </h3>
        <p>{{ $puzzle->created_at->format('d/m/Y') }}</p>

        @if ($puzzle->created_at != $puzzle->updated_at)
            <h3 class="font-semibold text-xl text-gray-800 pt-2"> @lang('Dernière mise à jour') </h3>
            <p>{{ $puzzle->updated_at->format('d/m/Y') }}</p>
        @endif
    </x-puzzles-card>
</x-app-layout>

```