

# Génération d'URL:

## 1.Introduction :

Laravel fournit plusieurs aides pour vous aider à générer des URL pour votre application. Ces aides sont principalement utiles pour créer des liens dans vos modèles et vos réponses API, ou pour générer des réponses de redirection vers une autre partie de votre application.

## 2.Les bases

### *a. Génération d'URLs*

Le helper `url` peut être utilisé pour générer des URL arbitraires pour votre application. L'URL générée utilisera automatiquement le schéma (HTTP ou HTTPS) et l'hôte de la requête en cours de traitement par l'application :

```
@php
    $student=App\Models\Student::find(66);
@endphp
<a href="{{url('/students/$student->id')}}">show</a>
```

### *b. Accéder à l'URL courante*

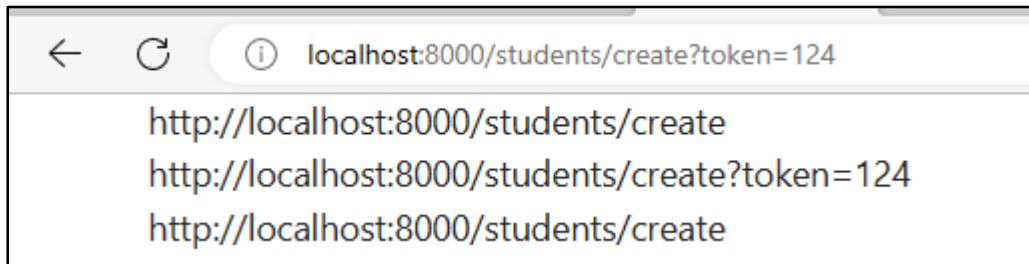
Si aucun chemin d'accès n'est fourni à l'aide `url`, une instance `Illuminate\Routing\UrlGenerator` est renvoyée, ce qui vous permet d'accéder aux informations sur l'URL actuelle :

```
{{{-- // Get the current URL without the query string... --}}
{{url()->current()}}

{{-- // Get the current URL including the query string... --}}
{{url()->full()}}

{{-- // Get the full URL for the previous request... --}}
{{url()->previous()}}
```

//output



### 3. URLs pour les routes nommées

Le helper **route** peut être utilisé pour générer des URL vers des routes nommées. Les routes nommées vous permettent de générer des URL sans être couplé à l'URL réelle définie sur la route. Par conséquent, si l'URL de la route change, aucune modification ne doit être apportée à vos appels à la fonction **route**. Par exemple, imaginez que votre application contient une route définie comme suit :

```
Route::get('/students/{student}',[StudentController::class,'show'])
->name('students.show');
```

Pour générer une URL vers cette route, vous pouvez utiliser le helper **route** comme suit :

```
<a href="{{route('students.show',$student)}}"><h1>{{$student->name}}</h1> </a>
```

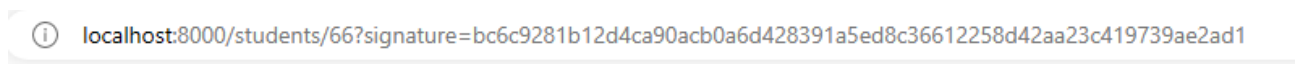
#### *a. URLs signées*

Laravel vous permet de créer facilement des URL "signées" vers des routes nommées. Ces URL ont un hachage de "signature" ajouté à la chaîne de requête, ce qui permet à Laravel de vérifier que l'URL n'a pas été modifiée depuis sa création. Les URL signées sont particulièrement utiles pour les routes qui sont accessibles au public mais qui nécessitent une couche de protection contre la manipulation des URL.

Par exemple, vous pouvez utiliser des URL signées pour mettre en place un lien public de "profile" qui est envoyé par courriel à vos clients. Pour créer une URL signée vers un itinéraire nommé, utilisez la méthode `signedRoute` de la façade URL :

```
<a href="{{URL::signedRoute('students.show',$student)}}"><h1>{{$student->name}}</h1> </a>
```

Le code ci-dessus retournera une chaîne de caractères comme ceci :



localhost:8000/students/66?signature=bc6c9281b12d4ca90acb0a6d428391a5ed8c36612258d42aa23c419739ae2ad1

http://localhost:8000/students/66?signature=bc6c9281b12d4ca90acb0a6d428391a5ed8c36612258d42aa23c419739ae2ad1

Si vous souhaitez générer une URL de route signée temporaire qui expire après un laps de temps déterminé, vous pouvez utiliser la méthode `temporarySignedRoute`. Lorsque Laravel valide une URL de route signée temporaire, il s'assure que l'horodatage d'expiration encodé dans l'URL signée n'est pas écoulé :

```
<a href="{{URL::temporarySignedRoute('students.show', now()->addMinutes(1),$student)}}"><h1>{{$student->name}}</h1> </a>
```

Maintenant que nous avons l'URL signée prête à être utilisée, nous devons valider les demandes entrantes et pour ce faire, nous avons deux options :

#### Avec un middleware

Laravel propose un middleware signé que nous pouvons utiliser. Tout d'abord, nous devons nous assurer que nous incluons le middleware dans `App\Http\Kernel`

```
protected $routeMiddleware = [
    //...
    'signed' => \App\Http\Middleware\ValidateSignature::class,
];
```

Maintenant nous pouvons l'utiliser dans notre fichier routes comme ceci :

```
Route::get('/students/{student}',[StudentController::class,'show'])->name('students.show')->middleware('signed');
```

Utilisation de la méthode `hasValidSignature()` sur l'objet de requête

Ou nous pouvons vérifier que la demande entrante a une signature valide à l'intérieur de l'action du contrôleur:

```
public function show(Request $request,Student $student)
{
    if (! $request->hasValidSignature()) {
        abort(401);
    }
    return view('students.profile', ['student' => $student]);
}
```

## 4.URLs pour les actions du contrôleur

La fonction **action** génère une URL pour l'action de contrôleur donnée :

```
<a href="{{ action([App\Http\Controllers\StudentController::class,'index']) }}">Return to students list</a>
```

Si la méthode du contrôleur accepte des paramètres de route, vous pouvez transmettre un tableau associatif de paramètres de route comme deuxième argument de la fonction :

```
<a href="{{ action([App\Http\Controllers\StudentController::class,'show'],[
'student'=>$student]) }}"><h1>{{$student->name}}</h1> </a>
```