

FRAMEWORKS PHP

Manuel Alejandro Romero Miguel



ÍNDICE DE CONTENIDOS

○ Paginación

○ Validando
formularios



LARAVEL: BLADE, EL GESTOR DE PLANTILLAS



Separamos la vista del resto
*Mantenmos relación. Ficheros diferentes pero
pudiendose comunicar*











<https://laravel.com/docs/10.x/pagination>

Consiste en restringir el número de filas de la colección que obtenemos al realizar una consulta

```
$alumnos = Alumno::paginate(10);
```

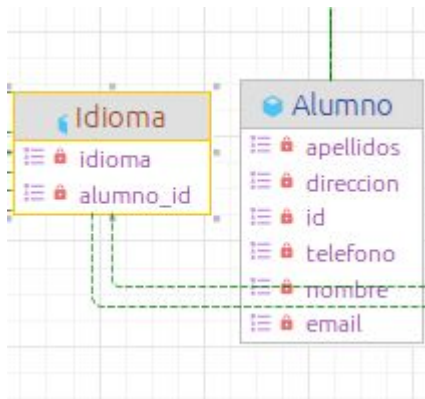
En este caso obtendremos todos los alumnos, pero los vamos a recorrer de 10 en 10
Para visualizarlos, \$alumnos tiene un método llamado **links()**, que añadiremos en la vista

```
<div class="bg-green-200">
    {{ $alumnos->links() }}
</div>
```

NUEVO ALUMNO					
Listado de alumnos					
Nombre	Email	Edad	Teléfono	Opciones	
Ana Isabel Salcedo	sdejesus@yahoo.es	26	961-284512		
Antonio Ybarra	jaime.noelia@yahoo.com	35	990-54-5263		
Luis Villalobos Tercero	wmuniz@latinmail.com	55	+34 953 753417		
Lic. Lola Dueñas Segundo	wvazquez@muniz.com	29	908 985822		
Nayara Toledo Tercero	enrique54@hotmail.es	222	+34 923 83 1762		
Nombre	Email	Edad			
Mostrando 36 al 40 de 40 resultados			< 1 2 3 4 5 6 7 8 >		

Especificando clave extranjera en una tabla

Para obtener una clave foránea debemos especificar el método **foreignkey()**.
Se puede hacer de dos maneras



Estableciendo los métodos correspondientes

```
$table->foreign('alumno_id')->references('id')->on('alumnos');
```

01 Especificando clave extranjera en una tabla

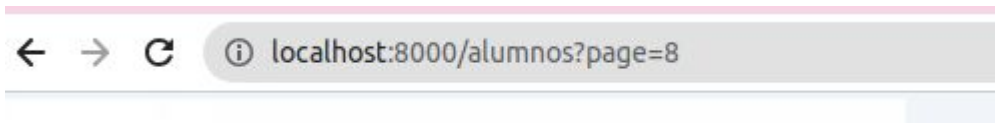
Siguiendo la convención de nombrado de **laravel**:

nombre_tabla_singular_id

```
$table->foreignId("alumno_id")->constrained();
```

01 La paginación

Ahora, a través de un atributo llamado **page** laravel sabe qué página mostrar de todos



Al realizar las diferentes acciones del crud, debemos de guardar y pasar el número de página, para que después de la acción, nos quedemos en la página concreta y no volvamos al principio

El listado: Recupero el número de página como el valor de una variable de tipo **get**

```
use Illuminate\Support\Facades\Request;

public function index()
{
    $alumnos = Alumno::paginate(5);
    $page = Request::get('page')??1;
    return view("alumnos.listado", compact("alumnos","page"));
}
//
```

A green curved arrow originates from the line '\$page = Request::get('page')??1;' in the code block and points towards the 'page=8' part of the URL 'localhost:8000/alumnos?page=8' shown in the image above.

01 La paginación en la vista

Revisamos nuestro fichero de configuración `tailwind.config.js` que tenga esta línea (si no, hay que añadirla)

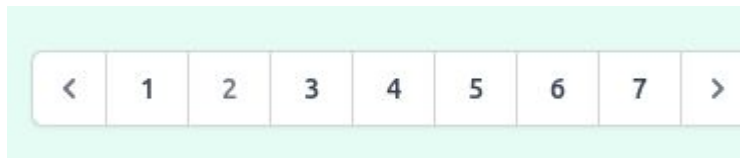
```
content: [  
  
  './vendor/laravel/framework/src/Illuminate/Pagination/resources/views/*.blade.php',  
  './storage/framework/views/*.php',  
  './resources/views/**/*.blade.php',  
],
```


01 La paginación en la vista

Simplemente añadimos al final de la vista

```
{{ $alumnos->links() }}
```

Con ello conseguiremos un menú de navegación para movernos entre diferentes páginas



Podemos publicar la vista para modificarla

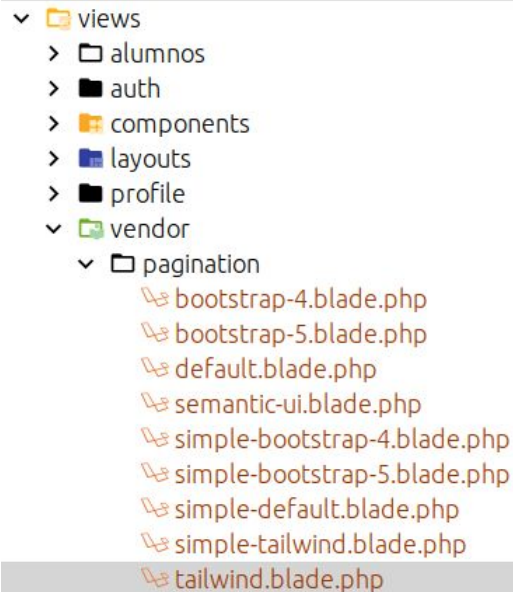
01 La paginación en la vista

Para publicar las vistas

```
php artisan vendor:publish --tag=laravel-pagination
```

Y nos aparecerán en **view**.

En nuestro caso la vista que se muestra es
tailwind.blade.php



01 La paginación

Editar

```
<td>
    <a href="{{route('alumnos.edit', [$alumno->id, 'page'=>$page]) }}">
```

listado.blade.php

```
public function edit(Alumno $alumno){
    $page = Request::get("page");
    return view ("alumnos.edit", compact ("alumno", "page"));
}
```

AlumnoController.blade.php

edit.blade.php

```
<form action="{{route('alumnos.update', [$alumno->id, 'page'=>$page]) }}" method="post"
class="space-y-4">_____
```

```
public function update(UpdateAlumnoRequest $request, Alumno $alumno)
{
    $page =$request->input('page');
    $datos nuevos = $request->input();
    $alumno->updateOrFail ($datos_nuevos);
    return redirect ("alumnos?page= $page");
}
```

AlumnoController.blade.php

01 La paginación

Crear nuevo alumno

listado.blade.php

```
<a class="btn btn-primary" href="{{route('alumnos.create', ['page'=>$page])}}">Nuevo Alumno</a>
```

AlumnoController.blade.php

```
public function create() {  
    $page = Request::get('page');  
    return view("alumnos.create", compact('page'));  
}
```

create.blade.php

```
<form action="{{route('alumnos.store', ['page'=>$page])}}" method="post" class="space-y-4">
```

AlumnoController.blade.php

```
public function store(StoreAlumnoRequest $request) {  
    $page = Request::get('page');  
    $alumno = new Alumno($request->input());  
    $alumno->saveOrFail();  
    return redirect(route('alumnos.index', ['page'=>$page]));  
}
```

01 La paginación

Borrar un alumno

listado.blade.php

```
<form action="{{ route('alumnos.destroy', $alumno->id) }}" method="POST">
```

AlumnoController.blade.php

```
public function destroy (Alumno $alumno)
{
    $alumno->delete();
    return back();
    // return redirect (route("alumnos.index"));
    //
}
```

Validando formularios

<https://laravel.com/docs/10.x/validation>

Siempre en el servidor hay que validar formularios

Método **validate** del objeto Request. Habrá que validar en el método **store** y **update**
Este método recibe un **array con las reglas de validación**

```
public function store (StoreAlumnoRequest $request)
{
    $request->validate ([
        'nombre'=>'required|size:5 '
    ]);
    ...
}
```

AlumnoController.blade.php

Si la validación no se cumple, no se entrega la response, sino que nos quedamos en la misma página

Validando formularios

mostrando datos en el formulario

Disponemos de la directiva **@errors** la cual mostrará mensajes en el caso de que se haya producido algún tipo de error.

Podemos recoger todos los errores y mostrarlos

```
@if ($errors->any())  
    <div class="alert alert-danger" >  
        <ul>  
            @foreach ($errors->all() as $error)  
                <li>{{ $error }}</li>  
            @endforeach  
        </ul>  
    </div>  
@endif
```

create.blade.php

Esta directiva la podemos personalizar, especificando el campo en el cual se ha producido un error,

```
@error("nombre")  
    <div class="text-red-500 text-xs" >{{ $message }}</div>  
@enderror
```

Form Request

Normalmente para realizar la validación escribiremos el código en un **Form Request**, Una vez creado la invocación es implícita a la acción correspondiente (store o update).
Cuando creamos el entorno del modelo con **--all** se crearon FormRequest
Esta clase tiene dos métodos importantes: **authorize y rules**
La **validación** se realiza en el método **rules**

```
public function rules(): array
{
    return [
        'nombre'=>"required|min:4|string" ,
        'telefono'=>"required|integer|size:9" ,
        'email'=>"required|email"
        //
    ];
}
```


Form Request

Puede ser que queramos mostrar cada error debajo de la caja de texto dónde se produjo. Disponemos de la variable **\$message** para mostrar el texto del error

```
<div>
  <label for="">Nombre</label>
  <input name="nombre" type="text" placeholder="nombre"
    class="input input-bordered input-primary w-full max-w-xs" />
</div>

@error("nombre")
  <div class="text-red-500 text-xs" >{{ $message }}</div>
@enderror
```

Adaptar el idioma

En laravel 10, no existe una carpeta con idiomas, hay que instalarla
Lo primero que debemos hacer es establecer la carpeta de idiomas
<https://laravel-lang.com/installation/>

```
php artisan lang:publish
```

Para la instalación de otros directorios traducidos accedemos a

<https://laravel-lang.com/installation/>

Siguiendo las instrucciones procedemos a la instalación

```
composer require laravel-lang/common --dev  
php artisan lang:add es  
php artisan lang:update
```

Adaptar el idioma

Ahora ya solo queda acceder a la configuración del fichero en config/app.php y actualizar la configuración de nuestra aplicación

```
'locale' => 'es',
```

Datos nuevo alumno

Ahora el mensaje estará en castellano

Nombre

El campo nombre es obligatorio.