

# ***Práctica 2***

## ***Creación de Modelos, Vistas y Controladores***

## Índice

Índice.....	2
1.Creación del Modelo de Libros: Genera un modelo y, un controlador para esta tabla.....	4
2. Controlador de Libros: En el controlador de libros, añade funciones para el CRUD (BUSCAR, INSERTAR, BORRAR, ACTUALIZAR) de libros. ....	5
3. Vistas de Libros: Crea vistas usando blade para mostrar todos los libros, ver detalles de un libro, añadir un nuevo libro y editar un libro existente.....	6
4. Creación del Modelo de Préstamos: Genera un modelo y un controlador. ....	14
5. Controlador de Préstamos: En el controlador de préstamos, añade funciones para el CRUD de préstamos. Recuerda que debes validar que el libro esté disponible antes de permitir el préstamo y actualizar el estado del libro cuando se realice un préstamo o una devolución. PISTA: Añadir un nuevo campo de si esta devuelto o no. ....	15
6. Vistas de Préstamos: Crea vistas para mostrar todos los préstamos, ver detalles de un préstamo, añadir un nuevo préstamo y finalizar un préstamo existente. (El usuario sabremos cual es en la siguiente practica).....	16
7. Rutas: Añade rutas en tu archivo web.php para cada acción en los controladores de libros y préstamos. ....	27



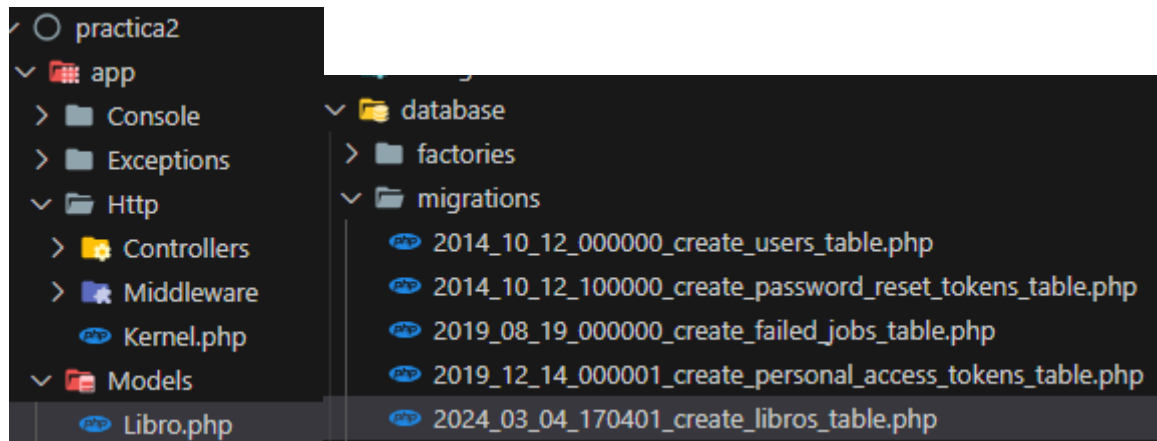
## 1.Creación del Modelo de Libros: Genera un modelo y, un controlador para esta tabla.

Esta primera parte se hizo en la práctica 1, pero repasamos.

Se crea el modelo desde la terminal de VSC:

```
C:\Users\nerea\OneDrive\Escritorio\practica2>php artisan make:model Libro -m
```

Con lo que se genera:

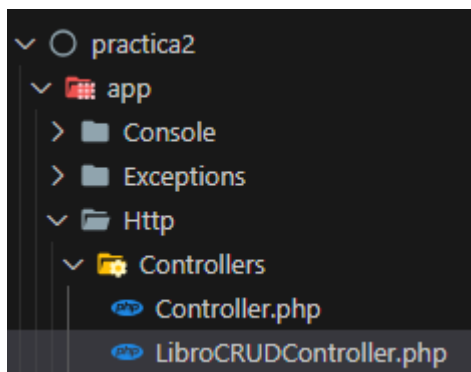


Dentro de Libros voy se crea el contenido de la tabla:

```
public function up(): void
{
    Schema::create('libros', function (Blueprint $table) {
        $table->id();
        $table->string('titulo');
        $table->string('autor');
        $table->date('año_publicacion');
        $table->string('genero');
        $table->string('disponible');
        $table->timestamps();
    });
}
```

Y ahora se genera el controlador:

```
C:\Users\nerea\OneDrive\Escritorio\practica2>php artisan make:controller LibroCRUDController
```



2. Controlador de Libros: En el controlador de libros, añade funciones para el CRUD (BUSCAR, INSERTAR, BORRAR, ACTUALIZAR) de libros.

En Libro.php:

- **Buscar**

```
1 reference | 0 overrides
public static function getLibroID($id){ //Buscamos libro por su id
    return Libro::find($id); //Find solo me sirve para buscar por la clave primaria
}

0 references | 0 overrides
public static function getLibroTitulo($titulo){ //Buscamos libro por su titulo
    return Libro::where('titulo','=', $titulo); //
}
```

- **Insertar**

```
2 references | 0 overrides
public static function saveLibro($titulo,$autor,$fechaPublicacion){
    //OPCION 1
    Libro::create([
        'titulo' => $titulo,
        'autor' => $autor,
        'fechaPublicacion' => $fechaPublicacion,
    ]);

    //OPCIÓN 2 Es Mejor
    $libro = new Libro;
    $libro -> titulo = $titulo;
    $libro -> autor = $autor;
    $libro -> fechaPublicacion = $fechaPublicacion;
    $libro -> save();

    return $libro;
}
```

- **Borrar**

```
0 references | 0 overrides
public static function deleteLibro ($id){
    $libro = Libro::find($id);
    if (isset($libro)){
        $libro -> delete();
        return "OK";
    }
    return "No existe ese ID del libro";
}
```

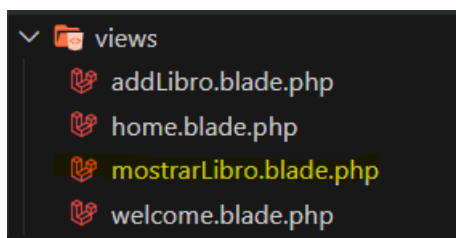
- **Actualizar**

```
0 references | 0 overrides
public static function updateLibro ($id, $titulo, $autor, $fechaPublicacion){
    $libro = Libro::find($id);
    if (isset ($libro)){
        $libro -> titulo = $titulo;
        $libro -> autor= $autor;
        $libro -> fechaPublicacion = $fechaPublicacion;
        $libro -> save();
        return $libro -> id;
    }
    return "No existe ese ID del libro";
}
```

3. Vistas de Libros: Crea vistas usando blade para mostrar todos los libros, ver detalles de un libro, añadir un nuevo libro y editar un libro existente.

- **Mostrar todos los libros**

Se crea la vista mostrarLibro.blade:



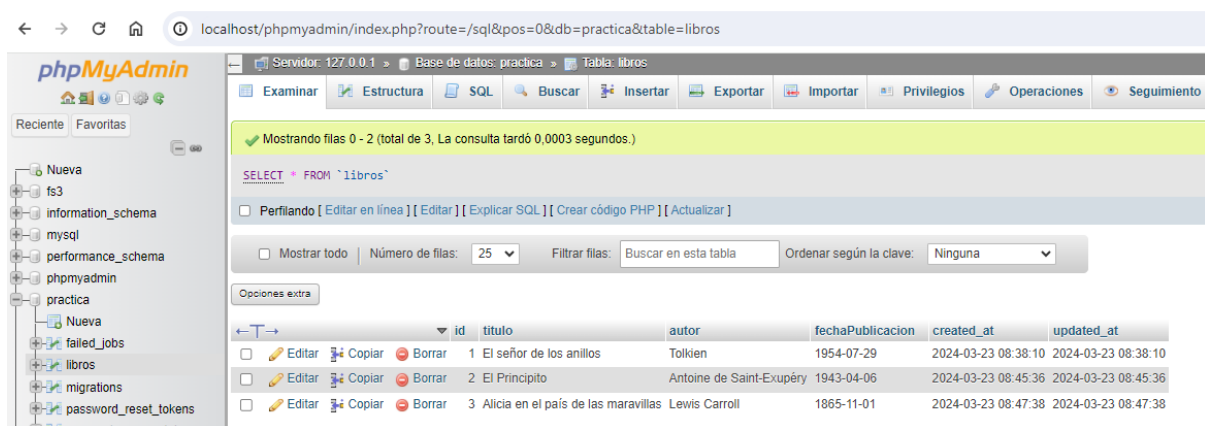
Esta sería la vista en VSC:

```
mostrarLibro.blade.php x
practica2 > resources > views > mostrarLibro.blade.php > a > p#({$libro -> id})
1 @extends('home')
2
3 @section('title', 'Mostrar Libros')
4
5
6 @section('content')
7     @if (count($libros) > 0)
8         @foreach ($libros as $libro)
9             <a href="/borrarLibro/{{$libro->id}}">
10                 <p id="{{$libro -> id}}">{{$libro -> titulo}} // {{$libro->autor}} // {{$libro->fechaPublicacion}}</p>
11             </a>
12         @endforeach
13     @endif
14 @endsection
15
```

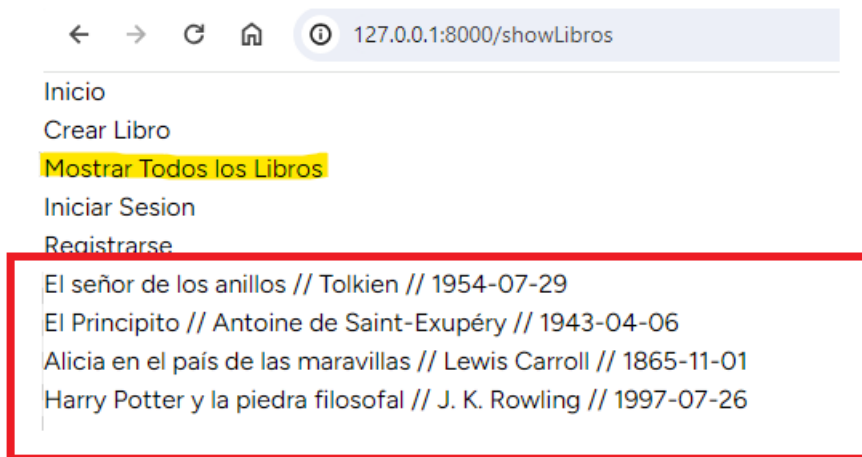
Cuando se abre el navegador sale el listado de los libros que se han registrado:



Los cuales coinciden con los que están en phpMyAdmin:

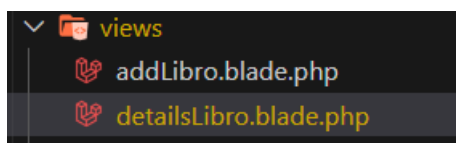


Se ha creado el botón “Mostrar todos los libros” que cuando se hace click se muestra el listado:



- **Ver detalles de un libro**

Se crea la vista:



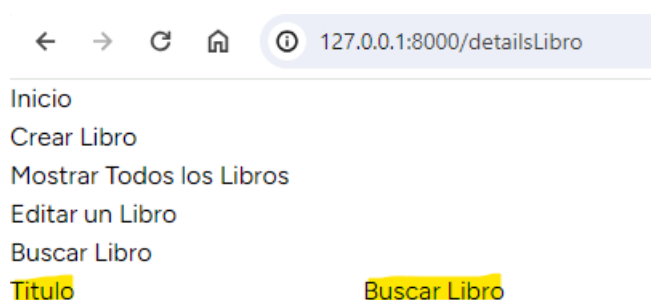
En VSC vamos a buscar un Libro por su título y mostrar todos los detalles:



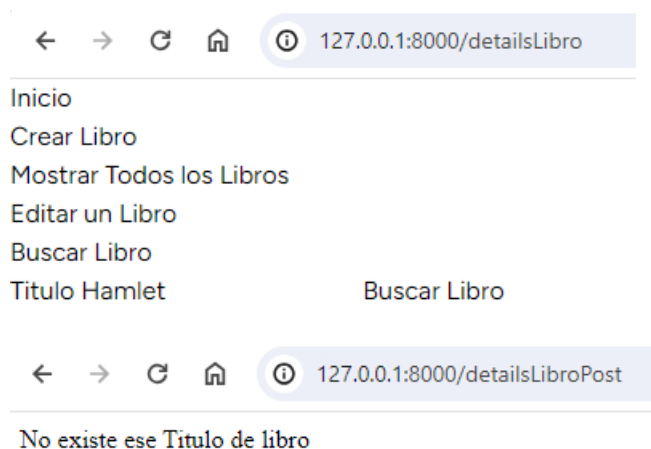


```
1 reference | 0 overrides
public function detailsLibro(Request $datosEnviados)
{
    $libro = Libro::getLibroTitulo($datosEnviados->input('titulo'))->first();
    if ($libro) {
        return $libro;
    }
    return "No existe ese Titulo de libro";
}
```

Cuando se abre el navegador y se hace click en Buscar Libro, te sale el form para escribir el Titulo y el botón de Buscar Libro:



Si se introduce un título que no figura en la base de datos nos devuelve el siguiente mensaje:

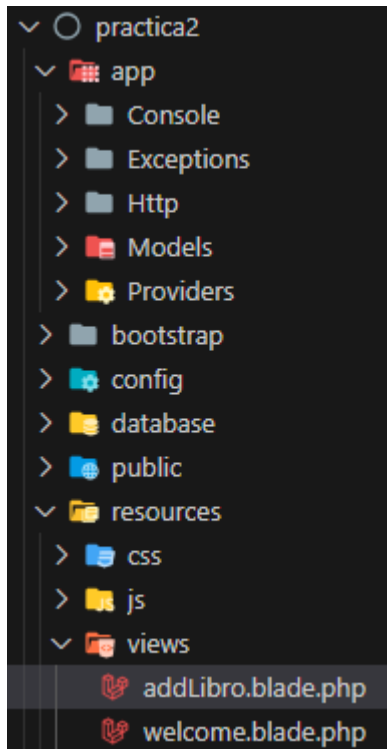


Y si se introduce uno que si está en la base de datos nos devuelve:

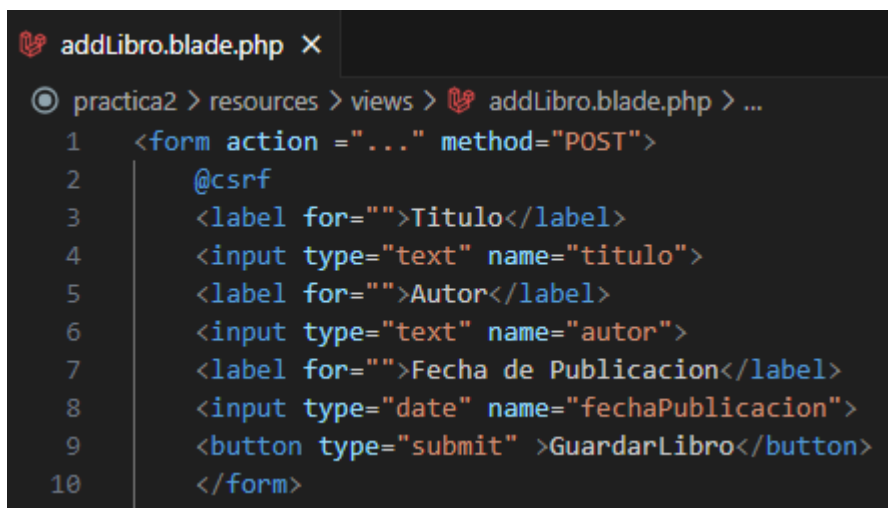


- **Añadir un nuevo libro**

Para la creación de la vista, se crea el archivo addLibro.blade.php:



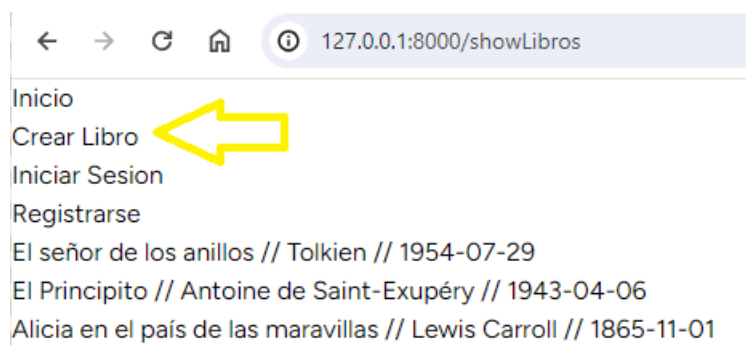
La nueva vista que creamos es un formulario:



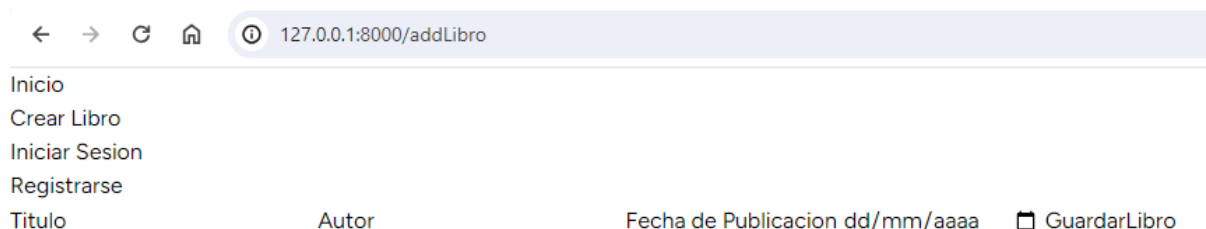
Esta sería la vista en VSC:

```
1 reference | 0 overrides
public function addLibro(Request $datosEnviados)
{
    return $this->libro->saveLibro(
        $datosEnviados->input('titulo'),
        $datosEnviados->input('autor'),
        $datosEnviados->input('fechaPublicacion')
    );
}
```

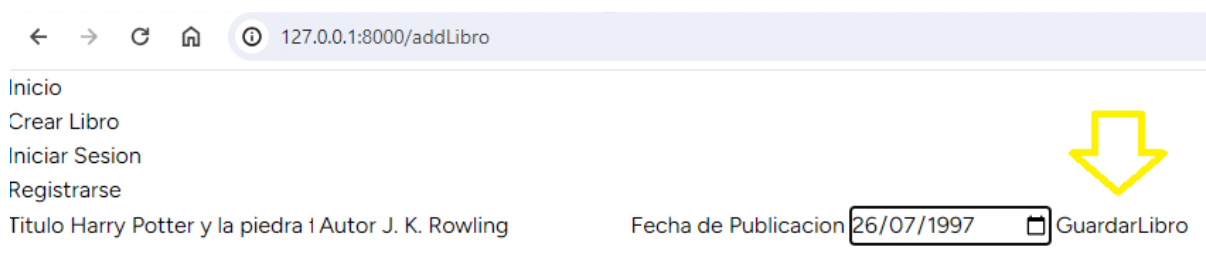
Cuando se abre el navegador, puedo añadir un libro desde la ventana de showLibros haciendo click en Crear Libro



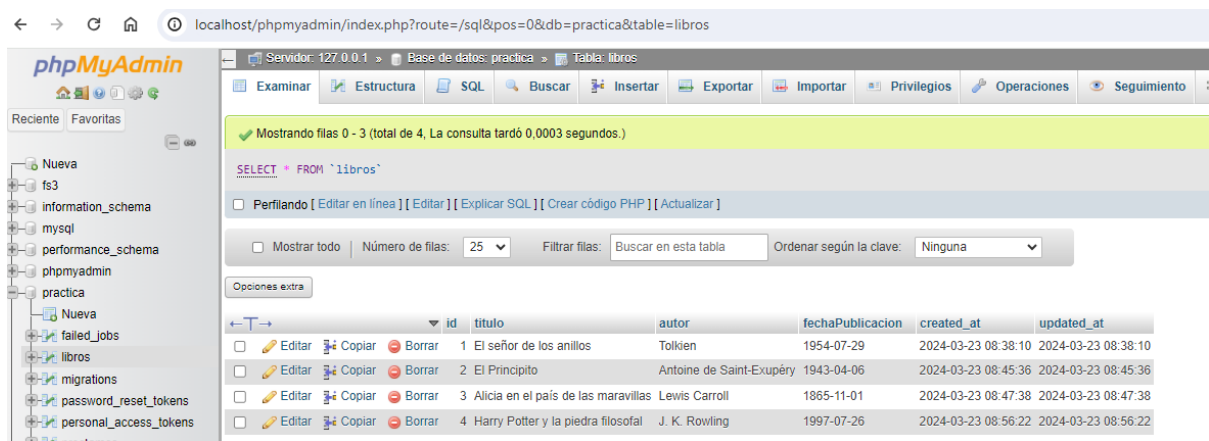
O abrir la url addLibro:



Se añade la información y click en Guardar Libro:



Comprobamos en phpMyAdmin que la información del libro se ha actualizado:

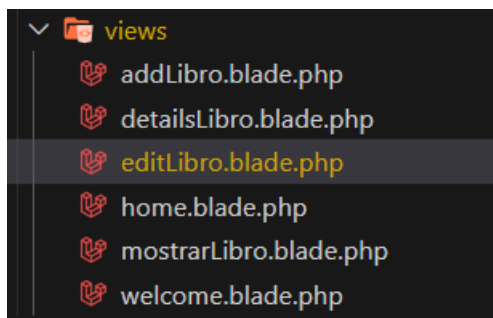


The screenshot shows the phpMyAdmin interface. The left sidebar displays the database structure, including a 'practica' database with a 'libros' table. The main area shows the 'libros' table with 4 records. The table has columns: id, titulo, autor, fechaPublicacion, created\_at, and updated\_at. The records are:

	id	titulo	autor	fechaPublicacion	created_at	updated_at
<input type="checkbox"/>	1	El señor de los anillos	Tolkien	1954-07-29	2024-03-23 08:38:10	2024-03-23 08:38:10
<input type="checkbox"/>	2	El Principito	Antoine de Saint-Exupéry	1943-04-06	2024-03-23 08:45:36	2024-03-23 08:45:36
<input type="checkbox"/>	3	Alicia en el país de las maravillas	Lewis Carroll	1865-11-01	2024-03-23 08:47:38	2024-03-23 08:47:38
<input type="checkbox"/>	4	Harry Potter y la piedra filosofal	J. K. Rowling	1997-07-26	2024-03-23 08:56:22	2024-03-23 08:56:22

- **Editar uno existente**

Se crea la vista, editLibro.blade.php:



La nueva vista que creamos es un formulario:

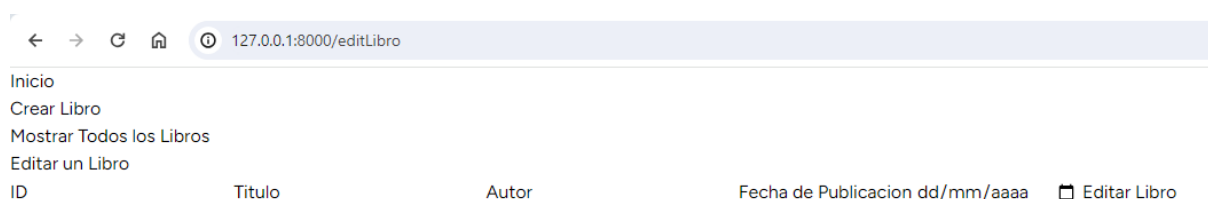
```
practica2 > resources > views > editLibro.blade.php > form > label
1  @extends('home')
2
3  @section('title', 'Editar Libro')
4
5
6  @section('content')
7  <form action="{{route('editLibro')}}" method="POST">
8      @csrf
9      <label for="id">ID</label>
10     <input type="number" name="id">
11     <label for="titulo">Titulo</label>
12     <input type="text" name="titulo">
13     <label for="autor">Autor</label>
14     <input type="text" name="autor">
15     <label for="fechaPublicacion">Fecha de Publicacion</label>
16     <input type="date" name="fechaPublicacion">
17     <button type="submit">Editar Libro</button>
18 </form>
19 @endsection
```

Esta sería la vista en VSC:

```
1 reference | 0 overrides
public static function updateLibro($id, $titulo, $autor, $fechaPublicacion)
{
    $libro = Libro::find($id);
    if (isset($libro)) {
        $libro->titulo = $titulo;
        $libro->autor = $autor;
        $libro->fechaPublicacion = $fechaPublicacion;
        $libro->save();
        return $libro;
    }
    return "No existe ese ID del libro";
}
```

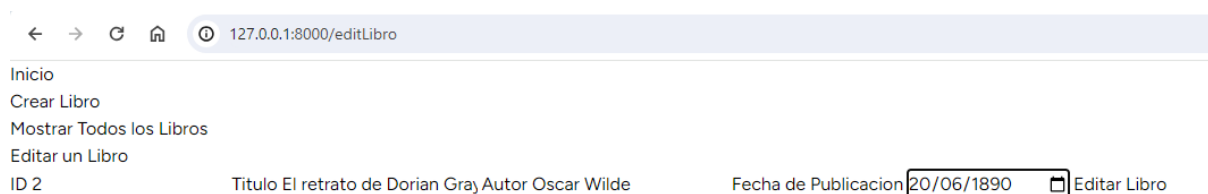
```
1 reference | 0 overrides
public function editLibro(Request $datosEnviados)
{
    return $this->libro->updateLibro(
        $datosEnviados->input('id'),
        $datosEnviados->input('titulo'),
        $datosEnviados->input('autor'),
        $datosEnviados->input('fechaPublicacion')
    );
}
```

Cuando se abre el navegador y se le da al botón de Editar un Libro:



ID	Titulo	Autor	Fecha de Publicacion dd/mm/aaaa	Editar Libro
----	--------	-------	---------------------------------	--------------

Se rellena el formulario con la información del libro, la cual se actualiza en la base de datos:



ID 2	Titulo El retrato de Dorian Gray	Autor Oscar Wilde	Fecha de Publicacion 20/06/1890	Editar Libro
------	----------------------------------	-------------------	---------------------------------	--------------

Si se revisa la base de datos vemos el libro que estaba en el ID 2 se ha modificado:



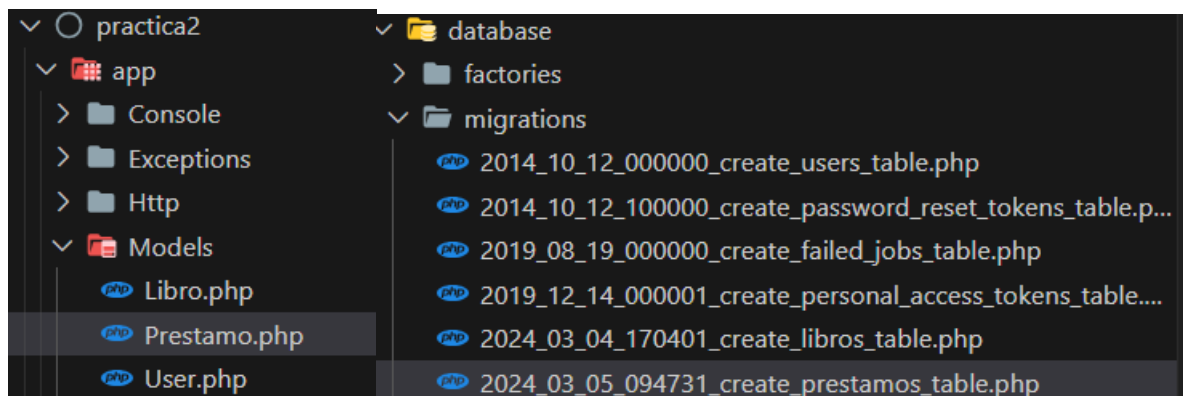
#### 4. Creación del Modelo de Préstamos: Genera un modelo y un controlador.

Esta primera parte se hizo en la práctica 1, pero repasamos.

Se crea el modelo desde la terminal de VSC:

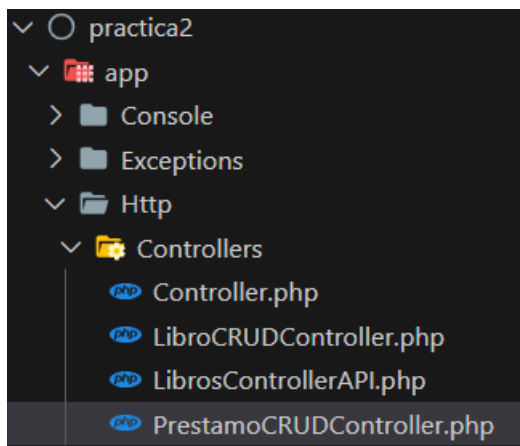
```
C:\Users\nerea\OneDrive\Escritorio\practica2>php artisan make:model Prestamo -m
```

Con lo que se genera:



Y ahora se genera el controlador:

```
C:\Users\nerea\OneDrive\Escritorio\practica2>php artisan make:controller PrestamoCRUDController
```



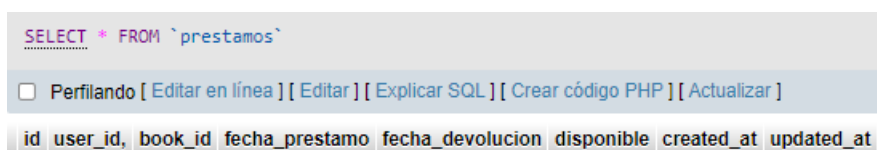
5. Controlador de Préstamos: En el controlador de préstamos, añade funciones para el CRUD de préstamos. Recuerda que debes validar que el libro esté disponible antes de permitir el préstamo y actualizar el estado del libro cuando se realice un préstamo o una devolución. PISTA: Añadir un nuevo campo de si esta devuelto o no.

Primero se define que campos va a tener préstamo:

- El usuario se va a dejar para poderlo usar en las próximas prácticas
- El campo fecha devolución se ha puesto que pueda ser nulo, ya que cuando se hace un préstamo no se sabe aún la fecha de devolución.

```
public function up(): void
{
    Schema::create('prestamos', function (Blueprint $table) {
        $table->id();
        // $table->string('user_id');
        $table->string('titulo');
        $table->date('fecha_prestamo');
        $table->date('fecha_devolucion')->nullable();
        $table->boolean('disponible')->default(true); //Campo para saber si el libro esta disponible para prestar
        $table->timestamps();
    });
}
```

Y esto será lo que se muestre en phpMyAdmin:



Se crean las funciones en Prestamo.php para:

- Libro disponible para permitir el préstamo

Cuando se realiza una devolución de un libro que ha estado prestado, buscamos por book\_id:

```
0 references | 0 overrides
public static function updatePrestamo($book_id, $fecha_devolucion)
{
    //Actualizamos una devolución
    $prestamo = Prestamo::find($book_id);
    if (isset($prestamo)) {
        $prestamo -> fecha_devolucion = $fecha_devolucion; //Se actualiza la fecha devolución
        $prestamo -> disponible = true; //El libro pasa a estar disponible
        $prestamo -> save();
        return $prestamo;
    }
    return "No existe ese ID de prestamo";
}
```

- Actualizar el estado del libro cuando se realiza préstamo:

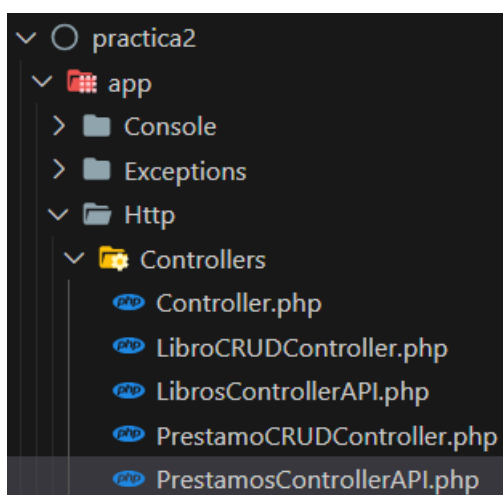
```
0 references | 0 overrides
public static function savePrestamo($titulo, $fecha_prestamo)
{
    //Guardamos un prestamo

    $prestamo = new Prestamo;
    $prestamo->titulo = $titulo;
    $prestamo->fecha_prestamo = $fecha_prestamo;
    $prestamo->disponible = false; //El libro pasa a no estar disponible
    $prestamo->save();

    return $prestamo;
}
```

6. Vistas de Préstamos: Crea vistas para mostrar todos los préstamos, ver detalles de un préstamo, añadir un nuevo préstamo y finalizar un préstamo existente. (El usuario sabremos cual es en la siguiente practica).

Se crea la API de Prestamos:





- **Añadir un nuevo préstamo**

```
1 reference | 0 overrides
public static function savePrestamo($titulo, $fecha_prestamo)
{
    //Guardamos un préstamo

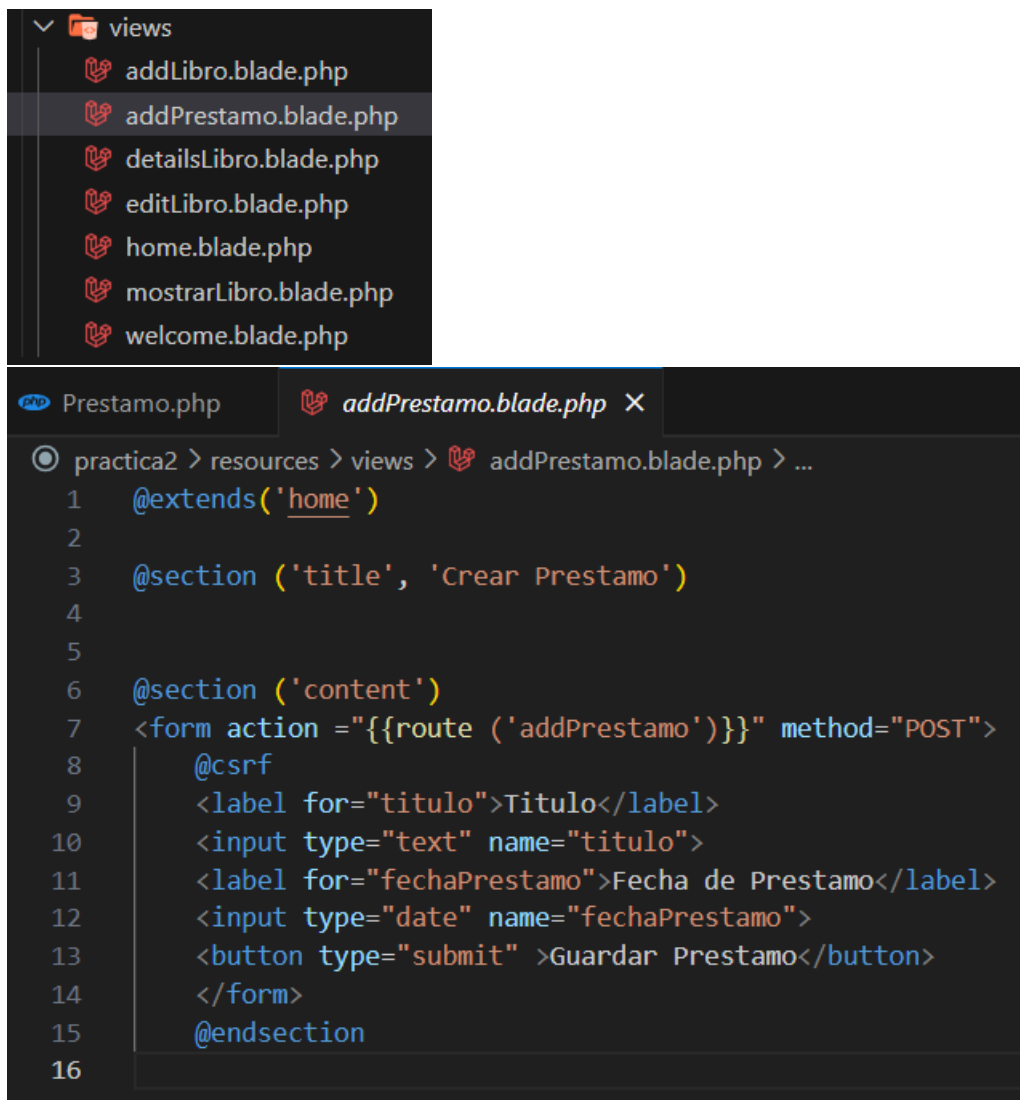
    $prestamo = new Prestamo;
    $prestamo->titulo = $titulo;
    $prestamo->fecha_prestamo = $fecha_prestamo;
    $prestamo->disponible = false; //El libro pasa a no estar disponible
    $prestamo->save();

    return $prestamo;
}
```

```
1 reference | 0 overrides
public function mostrarFormularioPrestamo()
{
    return view('addPrestamo'); //Es el addPrestamo.blade.php
}
```

```
1 reference | 0 overrides
public function addPrestamo(Request $datosEnviados)
{
    return $this->prestamo->savePrestamo(
        $datosEnviados->input('titulo'),
        $datosEnviados->input('fechaPrestamo'),
        $datosEnviados->input('disponible')
    );
}
```

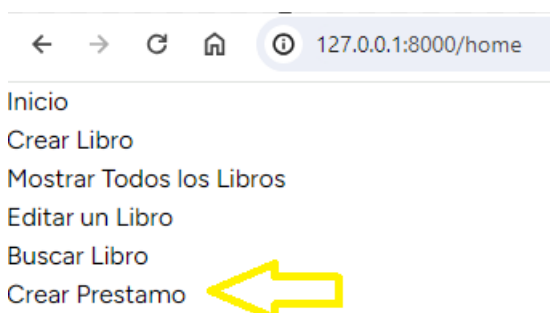
Se ha creado la vista addPrestamo.blade.php:



The image shows a code editor interface. On the left, a file explorer displays a folder named 'views' containing several Blade template files: addLibro.blade.php, addPrestamo.blade.php, detailsLibro.blade.php, editLibro.blade.php, home.blade.php, mostrarLibro.blade.php, and welcome.blade.php. The file 'addPrestamo.blade.php' is selected. The main editor area shows the code for this file, which extends the 'home' view and creates a section for adding a loan. The code includes a form with fields for 'titulo' (title) and 'fechaPrestamo' (loan date), and a 'Guardar Prestamo' (Save Loan) button.

```
practica2 > resources > views > addPrestamo.blade.php > ...
1  @extends('home')
2
3  @section('title', 'Crear Prestamo')
4
5
6  @section('content')
7      <form action="{{route('addPrestamo')}}" method="POST">
8          @csrf
9          <label for="titulo">Titulo</label>
10         <input type="text" name="titulo">
11         <label for="fechaPrestamo">Fecha de Prestamo</label>
12         <input type="date" name="fechaPrestamo">
13         <button type="submit">Guardar Prestamo</button>
14     </form>
15     @endsection
16
```

Y cuando se abre el navegador:



← → ↺ 🏠 ⓘ 127.0.0.1:8000/addPrestamo

Inicio  
Crear Libro  
Mostrar Todos los Libros  
Editar un Libro  
Buscar Libro  
Crear Prestamo

Título Fecha de Prestamo dd/mm/aaaa

Y cuando se añade la información se vuelca correctamente en la base de datos:

SELECT \* FROM `prestamos`

☐ Perfilando [ Editar en línea ] [ Editar ] [ Explicar SQL ] [ Crear código PHP ] [ Actualizar ]

☐ Mostrar todo | Número de filas: 25 | Filtrar filas:

Opciones extra

	id	titulo	fecha_prestamo	fecha_devolucion	disponible	created_at	updated_at
<input type="checkbox"/>	1	El señor de los anillos	2024-03-01	NULL	0	2024-03-26 17:57:59	2024-03-26 17:57:59

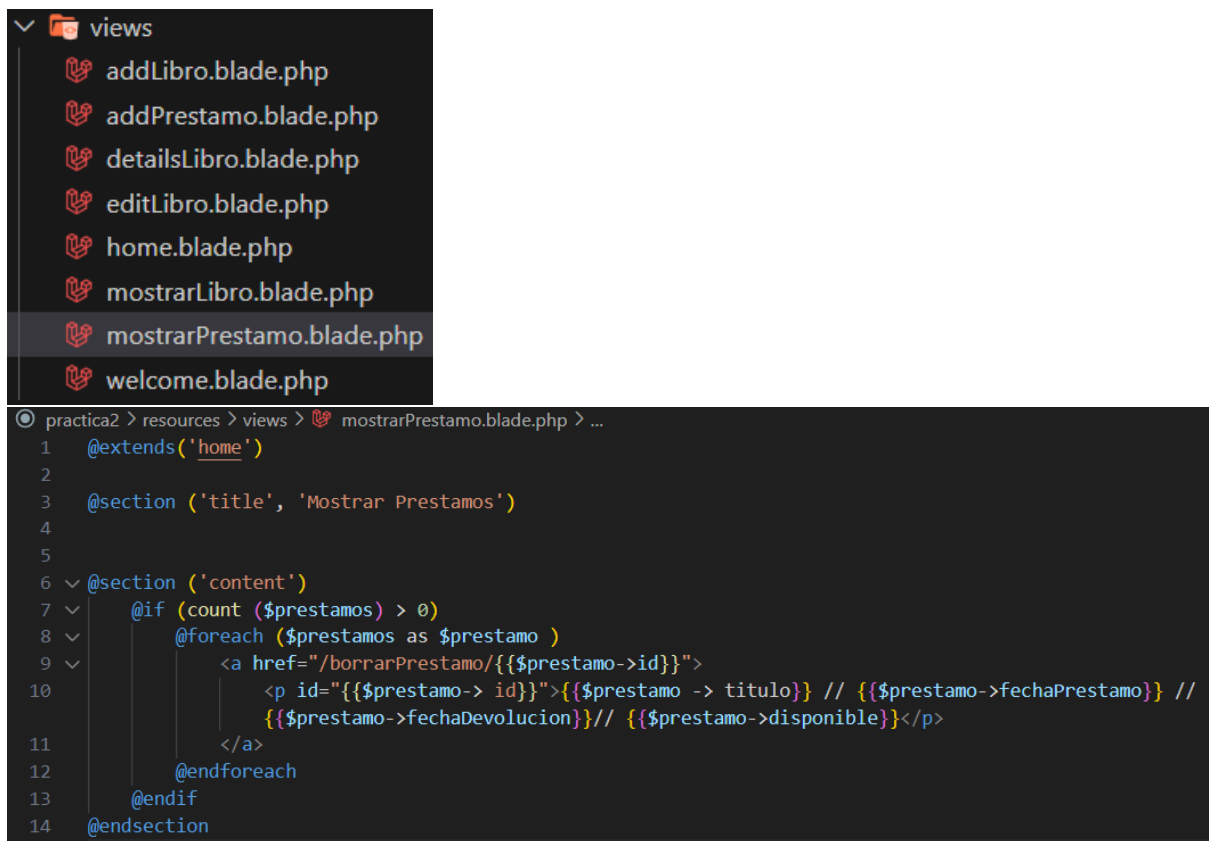
Editar Copiar Borrar

#### - Mostrar todos los préstamos

Se crea el botón en Home:

```
<ul>
  <li><a href="{{route('home')}}">Inicio</a></li>
  <li><a href="{{route('FormularioAddLibro')}}">Crear Libro</a></li>
  <li><a href="{{route('mostrarFormularioAdd')}}">Mostrar Todos los Libros</a></li>
  <li><a href="{{route('FormularioEditLibro')}}">Editar un Libro</a></li>
  <li><a href="{{route('FormularioDetailsLibro')}}">Buscar Libro</a></li>
  <li><a href="{{route('FormularioPrestamoLibro')}}">Crear Prestamo</a></li>
  <li><a href="{{route('mostrarFormularioPrestamo')}}">Mostrar Todos los Prestamos</a></li>
</ul>
```

Ahora, se necesita crear la vista mostrarPrestamo.blade.php:



The screenshot shows a file explorer on the left with the 'views' directory expanded, listing several Blade templates. The main editor displays the content of 'mostrarPrestamo.blade.php'.

```
views
├── addLibro.blade.php
├── addPrestamo.blade.php
├── detailsLibro.blade.php
├── editLibro.blade.php
├── home.blade.php
├── mostrarLibro.blade.php
├── mostrarPrestamo.blade.php
└── welcome.blade.php
```

```
practica2 > resources > views > mostrarPrestamo.blade.php > ...
1  @extends('home')
2
3  @section('title', 'Mostrar Prestamos')
4
5
6  @section('content')
7      @if (count($prestamos) > 0)
8          @foreach ($prestamos as $prestamo)
9              <a href="/borrarPrestamo/{{ $prestamo->id }}">
10                 <p id="{{ $prestamo->id }}">{{ $prestamo->titulo }} // {{ $prestamo->fechaPrestamo }} //
11                    {{ $prestamo->fechaDevolucion }} // {{ $prestamo->disponible }}</p>
12                 </a>
13             @endforeach
14         @endif
15     @endsection
```

Y se crea la ruta web:

```
//Rutas para Prestamo
Route::get('/addPrestamo', [PrestamoCRUDController::class, 'mostrarFormularioPrestamo']->name('FormularioPrestamoLibro'));

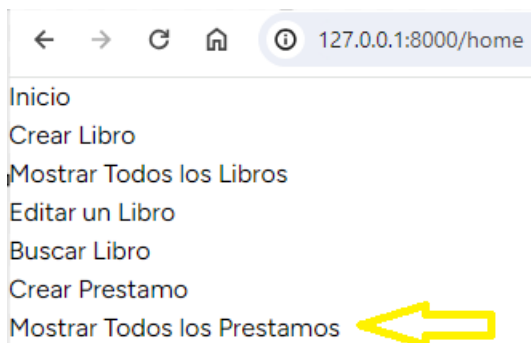
Route::post('/addPrestamoPost', [PrestamoCRUDController::class, 'addPrestamo']->name('addPrestamo'));

Route::get('/showPrestamos', [PrestamoCRUDController::class, 'showPrestamos']->name('mostrarFormularioPrestamo'));
```

Ya sólo queda definir la función:

```
1 reference | 0 overrides
public function showPrestamos()
{
    $allPrestamos = $this->prestamo->obtenerPrestamos();
    return view('mostrarPrestamo', ['prestamos' => $allPrestamos]);
}
```

Si se abre el navegador:



Se nos muestra lo mismo que tenemos en la base de datos:

Mostrar Todos los Prestamos  
El señor de los anillos // // // 0  
Don Quijote de la Mancha // // // 0

SELECT \* FROM `prestamos`

☐ Perfilando

[\[ Editar en línea \]](#)

[\[ Editar \]](#)

[\[ Explicar SQL \]](#)

[\[ Crear código PHP \]](#)

[\[ Actualizar \]](#)

☐ Mostrar todo

Número de filas: 25

Filtrar filas:

Ordenar según la clave: Ninguna

Opciones extra

			id	titulo	fecha_prestamo	fecha_devolucion	disponible	created_at	updated_at	
<input type="checkbox"/>	<a href="#">Editar</a>	<a href="#">Copiar</a>	<a href="#">Borrar</a>	1	El señor de los anillos	2024-03-01	NULL	0	2024-03-26 17:57:59	2024-03-26 17:57:59
<input type="checkbox"/>	<a href="#">Editar</a>	<a href="#">Copiar</a>	<a href="#">Borrar</a>	3	Don Quijote de la Mancha	2024-03-18	NULL	0	2024-03-28 09:35:59	2024-03-28 09:35:59

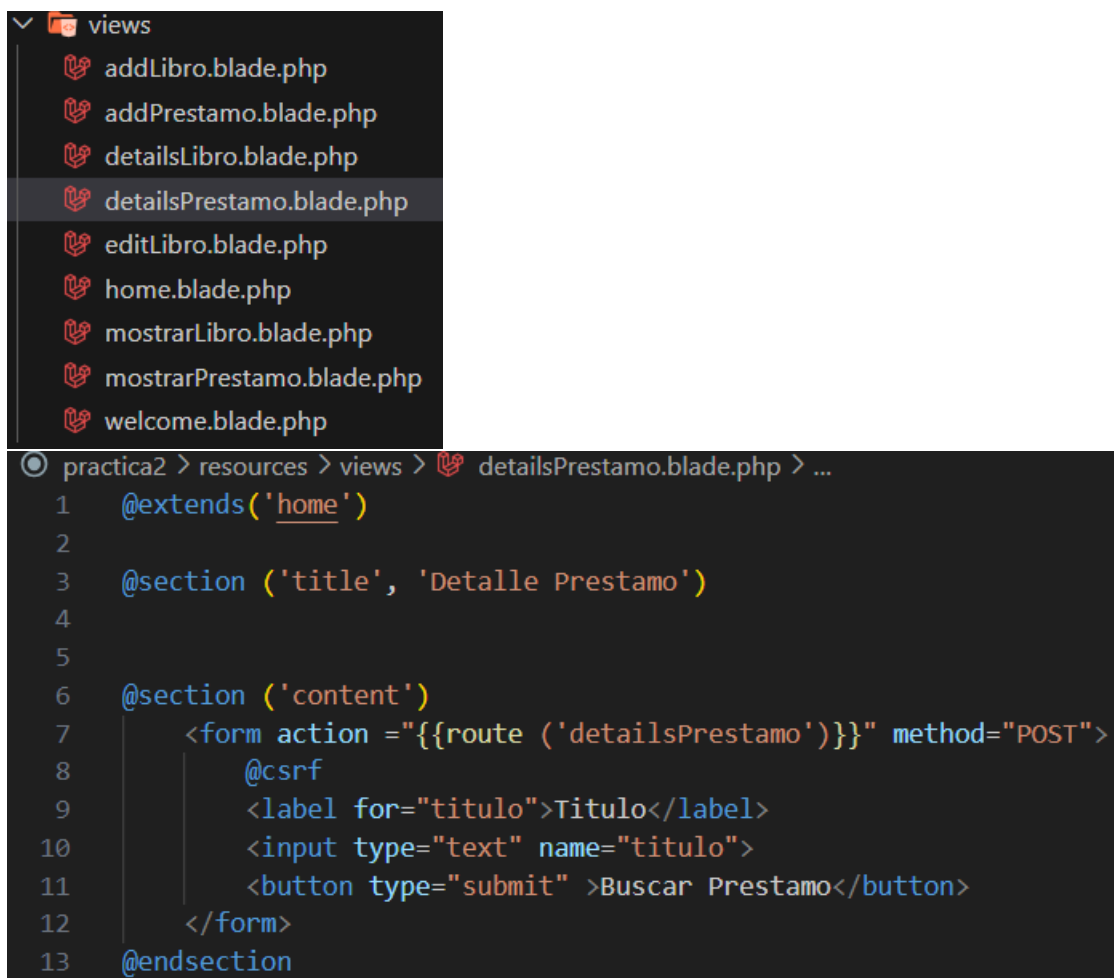
#### - Ver detalles de un préstamo

Para la búsqueda de un préstamo se tiene que hacer en dos pasos: primero mostrar el formulario para que se inserte la información del título del préstamo a buscar y luego que se muestre la información obtenida (los detalles del préstamo si está prestado o un mensaje que diga que no está prestado).

Se crea el botón de Home:

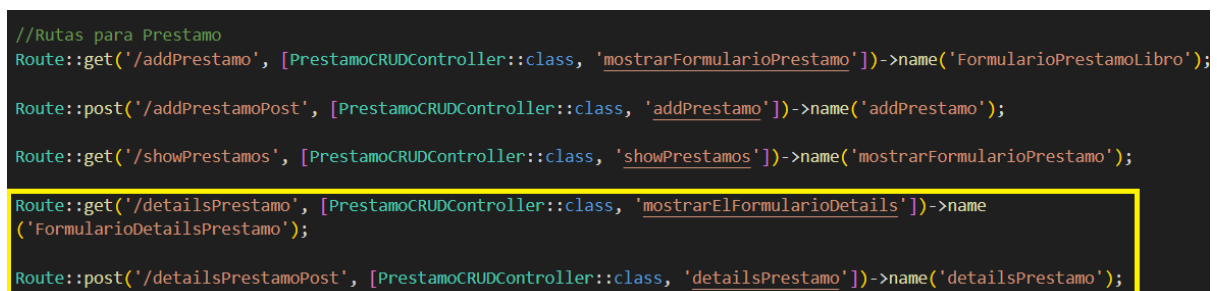
```
<ul>
  <li><a href="{{route('home')}}">Inicio</a></li>
  <li><a href="{{route('FormularioAddLibro')}}">Crear Libro</a></li>
  <li><a href="{{route('mostrarFormularioAdd')}}">Mostrar Todos los Libros</a></li>
  <li><a href="{{route('FormularioEditLibro')}}">Editar un Libro</a></li>
  <li><a href="{{route('FormularioDetailsLibro')}}">Buscar Libro</a></li>
  <li><a href="{{route('FormularioPrestamoLibro')}}">Crear Prestamo</a></li>
  <li><a href="{{route('mostrarFormularioPrestamo')}}">Mostrar Todos los Prestamos</a></li>
  <li><a href="{{route('FormularioDetailsPrestamo')}}">Buscar un Prestamo</a></li>
</ul>
```

Ahora, se necesita crear la vista detailsPrestamo.blade.php:



```
practica2 > resources > views > detailsPrestamo.blade.php > ...
1  @extends('home')
2
3  @section ('title', 'Detalle Prestamo')
4
5
6  @section ('content')
7      <form action="{{route ('detailsPrestamo')}}" method="POST">
8          @csrf
9          <label for="titulo">Titulo</label>
10         <input type="text" name="titulo">
11         <button type="submit" >Buscar Prestamo</button>
12     </form>
13 @endsection
```

Y se crea la ruta web, que como se dijo anteriormente, son dos pasos:



```
//Rutas para Prestamo
Route::get('/addPrestamo', [PrestamoCRUDController::class, 'mostrarFormularioPrestamo'])->name('FormularioPrestamoLibro');

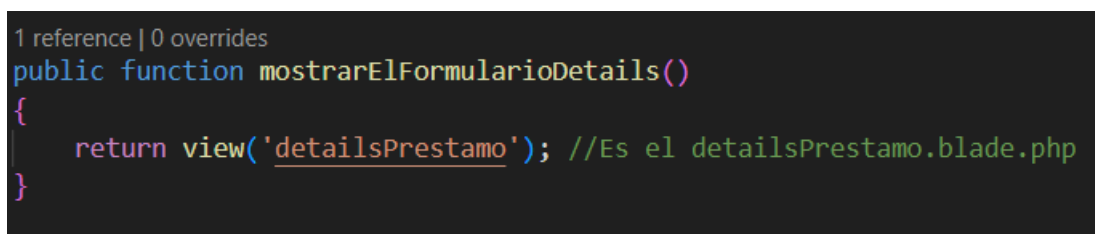
Route::post('/addPrestamoPost', [PrestamoCRUDController::class, 'addPrestamo'])->name('addPrestamo');

Route::get('/showPrestamos', [PrestamoCRUDController::class, 'showPrestamos'])->name('mostrarFormularioPrestamo');

Route::get('/detailsPrestamo', [PrestamoCRUDController::class, 'mostrarElFormularioDetails'])->name('FormularioDetailsPrestamo');

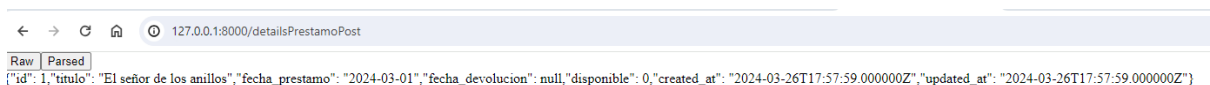
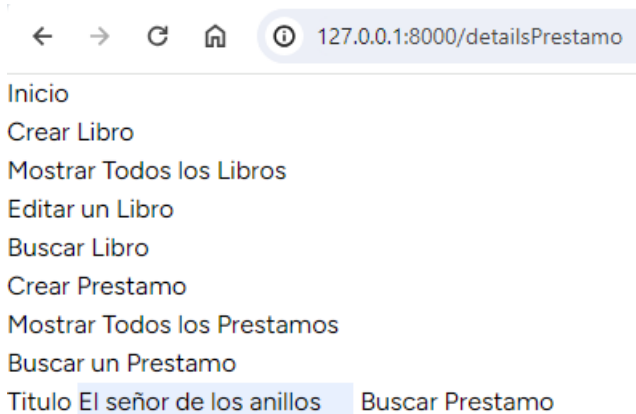
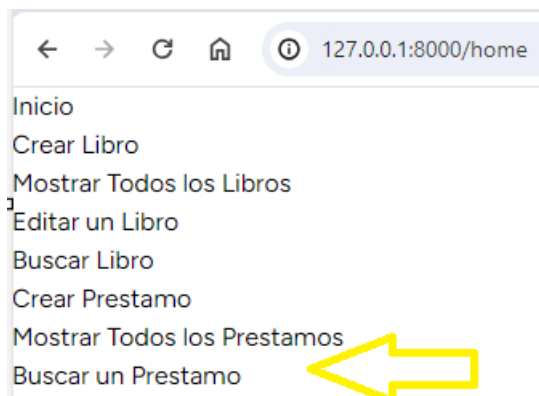
Route::post('/detailsPrestamoPost', [PrestamoCRUDController::class, 'detailsPrestamo'])->name('detailsPrestamo');
```

Ya sólo queda definir la función:



```
1 reference | 0 overrides
public function mostrarElFormularioDetails()
{
    return view('detailsPrestamo'); //Es el detailsPrestamo.blade.php
}
```

Si se abre el navegador:

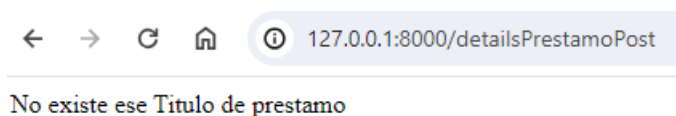


Se nos muestra lo mismo que tenemos en la base de datos:



Y si se introduce un título que no se encuentra en la base de datos:

Buscar un Prestamo  
Titulo El Principito      Buscar Prestamo



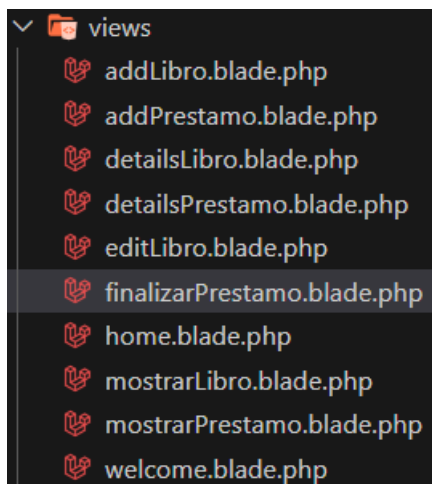
- **Finalizar un préstamo existente**

Igual que el caso anterior, tenemos dos pasos: primero se muestra el formulario para pedirnos el ID del libro de la devolución y la fecha, y luego ya se ejecuta la devolución, actualizando la base de datos, como en el caso anterior, si no se encontrara el ID nos devuelve un mensaje de error.

Se crea el botón de Home:

```
<ul>
<li><a href="{{route('home')}}">Inicio</a></li>
<li><a href="{{route('FormularioAddLibro')}}">Crear Libro</a></li>
<li><a href="{{route('mostrarFormularioAdd')}}">Mostrar Todos los Libros</a></li>
<li><a href="{{route('FormularioEditLibro')}}">Editar un Libro</a></li>
<li><a href="{{route('FormularioDetailsLibro')}}">Buscar Libro</a></li>
<li><a href="{{route('FormularioPrestamoLibro')}}">Crear Prestamo</a></li>
<li><a href="{{route('mostrarFormularioPrestamo')}}">Mostrar Todos los Prestamos</a></li>
<li><a href="{{route('FormularioDetailsPrestamo')}}">Buscar un Prestamo</a></li>
<li><a href="{{route('FormularioFinalizarPrestamo')}}">Finalizar un Prestamo</a></li>
</ul>
```

Ahora, se necesita crear la vista finalizarPrestamo.blade.php:





```
practica2 > resources > views > finalizarPrestamo.blade.php > ...
1  @extends('home')
2
3  @section('title', 'Finzalizar Prestamo')
4
5
6  @section('content')
7      <form action="{{route('finalizarPrestamo')}}" method="POST">
8          @csrf
9          <label for="id">ID</label>
10         <input type="number" name="id">
11         <label for="titulo">Titulo</label>
12         <input type="text" name="titulo">
13         <label for="fechaDevolucion">Fecha de Devolucion</label>
14         <input type="date" name="fechaPrestamo">
15         <button type="submit">Guardar Devolucion</button>
16     </form>
17 @endsection
```

Y se crea la ruta web, que como se dijo anteriormente, son dos pasos:

```
Route::get('/finalizarPrestamo', [PrestamoCRUDController::class, 'mostrarFormularioFinalizar'])->name('FormularioFinalizarPrestamo');

Route::post('/finalizarPrestamoPost', [PrestamoCRUDController::class, 'finalizarPrestamo'])->name('finalizarPrestamo');
```

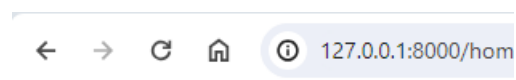
Ya sólo queda definir la función:

```
1 reference | 0 overrides
public function mostrarFormularioFinalizar()
{
    return view('finalizarPrestamo'); //Es el finalizarPrestamo.blade.php
}
```

```
1 reference | 0 overrides
public function finalizarPrestamo(Request $datosEnviados)
{
    return $this->prestamo->updatePrestamo(
        $datosEnviados->input('id'),
        $datosEnviados->input('titulo'),
        $datosEnviados->input('fechaDevolucion'),
    );
}
```

```
1 reference | 0 overrides
public static function updatePrestamo($book_id, $titulo, $fecha_devolucion)
{
    //Actualizamos una devolución
    $prestamo = Prestamo::find($book_id);
    if (isset($prestamo)) {
        $prestamo->titulo = $titulo;
        $prestamo->fecha_devolucion = $fecha_devolucion; //Se actualiza la fecha devolución
        $prestamo->disponible = true; //El libro pasa a estar disponible
        $prestamo->save();
        return $prestamo;
    }
    return "No existe ese ID de prestamo";
}
```

Si se abre el navegador:



Inicio

Crear Libro

Mostrar Todos los Libros

Editar un Libro

Buscar Libro

Crear Prestamo

Mostrar Todos los Prestamos

Buscar un Prestamo

Finalizar un Prestamo



Finalizar un Prestamo

ID 1

Titulo Alicia en el país de las m

Fecha de Devolucion 28/03/2024

Guardar Devolucion

127.0.0.1:8000/finalizarPrestamoPost

Raw | Parsed

```
{"id":1,"titulo":"Alicia en el país de las maravillas","fecha_prestamo":"2024-03-01","fecha_devolucion":"2024-03-28","disponible":true,"created_at":"2024-03-26T17:57:59.000000Z","updated_at":"2024-03-28T11:23:26.000000Z"}
```

Se nos muestra lo mismo que tenemos en la base de datos:

SELECT \* FROM `prestamos`

☐ Perfilando [ [Editar en línea](#) ] [ [Editar](#) ] [ [Explicar SQL](#) ] [ [Crear código PHP](#) ] [ [Actualizar](#) ]

☐ Mostrar todo | Número de filas: 25 | Filtrar filas:  Ordenar según la clave: Ninguna

Opciones extra

		id	titulo	fecha_prestamo	fecha_devolucion	disponible	created_at	updated_at
<input type="checkbox"/>	Editar							
		1	Alicia en el país de las maravillas	2024-03-01	2024-03-28	1	2024-03-26 17:57:59	2024-03-28 11:23:26
<input type="checkbox"/>	Editar							
		3	Don Quijote de la Mancha	2024-03-18	NULL	1	2024-03-28 09:35:59	2024-03-28 11:03:35

Y si se introduce un título que no se encuentra en la base de datos:

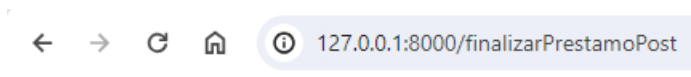
Finalizar un Prestamo

ID 2

Titulo Harry Potter y la piedra

Fecha de Devolucion 27/03/2024

Guardar Devolucion



Importante, en este caso estamos haciendo la búsqueda por ID, nos verifica el ID del préstamo, no el título.

## 7. Rutas: Añade rutas en tu archivo web.php para cada acción en los controladores de libros y préstamos.

Para mostrar todas las url de las vistas que hemos mostrado en los apartados 3 y 6 tenemos el siguiente archivo web.php:

- Las rutas de Libro:

```
Route::get('/home', [Controller::class, 'home'])->name('home');

Route::get('/addLibro', [LibroCRUDController::class, 'mostrarFormularioAdd'])->name('FormularioAddLibro');
Route::post('/addLibroPost', [LibroCRUDController::class, 'addLibro'])->name('addLibro');
Route::get('/showLibros', [LibroCRUDController::class, 'showLibros'])->name('mostrarFormularioAdd');
Route::get('/editLibro', [LibroCRUDController::class, 'mostrarFormularioEdit'])->name('FormularioEditLibro');
Route::post('/editLibroPost', [LibroCRUDController::class, 'editLibro'])->name('editLibro');
Route::get('/detailsLibro', [LibroCRUDController::class, 'mostrarFormularioDetails'])->name('FormularioDetailsLibro');
Route::post('/detailsLibroPost', [LibroCRUDController::class, 'detailsLibro'])->name('detailsLibro');
```

- Las rutas de Préstamo:

```
//Rutas para Prestamo
Route::get('/addPrestamo', [PrestamoCRUDController::class, 'mostrarFormularioPrestamo'])->name('FormularioPrestamoLibro');
Route::post('/addPrestamoPost', [PrestamoCRUDController::class, 'addPrestamo'])->name('addPrestamo');
Route::get('/showPrestamos', [PrestamoCRUDController::class, 'showPrestamos'])->name('mostrarFormularioPrestamo');
Route::get('/detailsPrestamo', [PrestamoCRUDController::class, 'mostrarElFormularioDetails'])->name('FormularioDetailsPrestamo');
Route::post('/detailsPrestamoPost', [PrestamoCRUDController::class, 'detailsPrestamo'])->name('detailsPrestamo');
Route::get('/finalizarPrestamo', [PrestamoCRUDController::class, 'mostrarFormularioFinalizar'])->name('FormularioFinalizarPrestamo');
Route::post('/finalizarPrestamoPost', [PrestamoCRUDController::class, 'finalizarPrestamo'])->name('finalizarPrestamo');
```