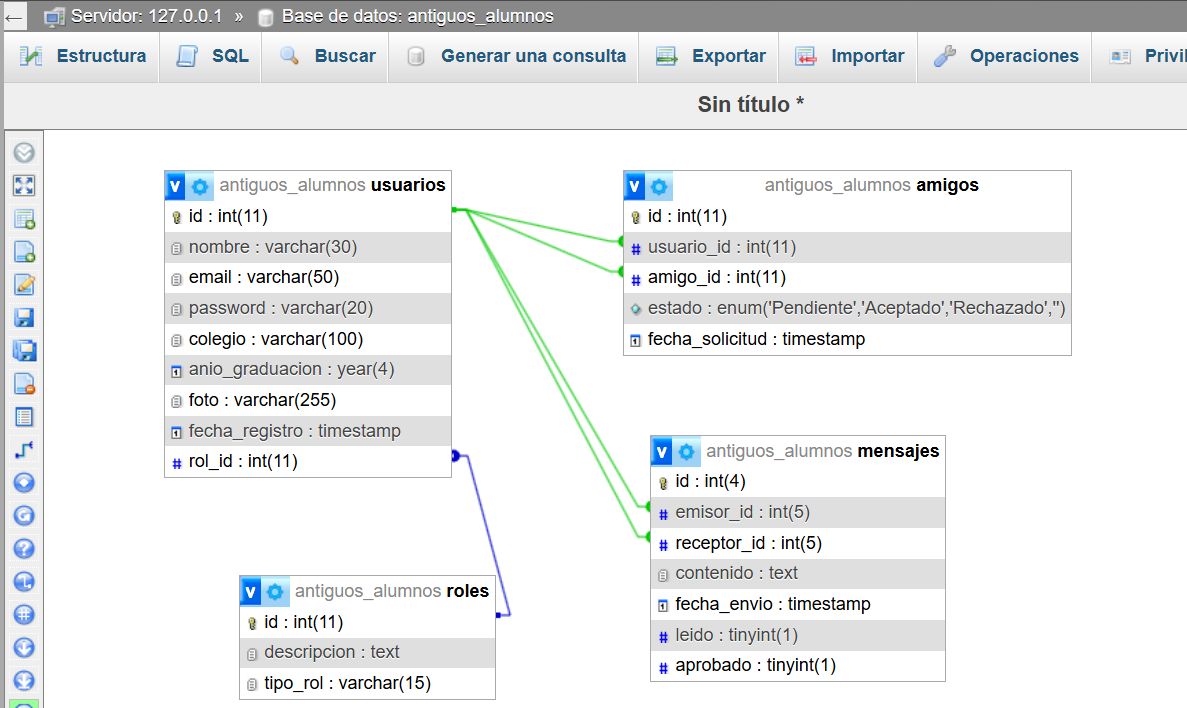
PROYECTO ANTIGUOS ALUMNOS

# Primeros pasos en el diseño

1. Diseño de la base de datos



Comienzo el proyecto del tfg consistente en el desarrollo y diseño de una aplicación web, a través de la cual se puedan encontrar compañeros de colegios e institutos.

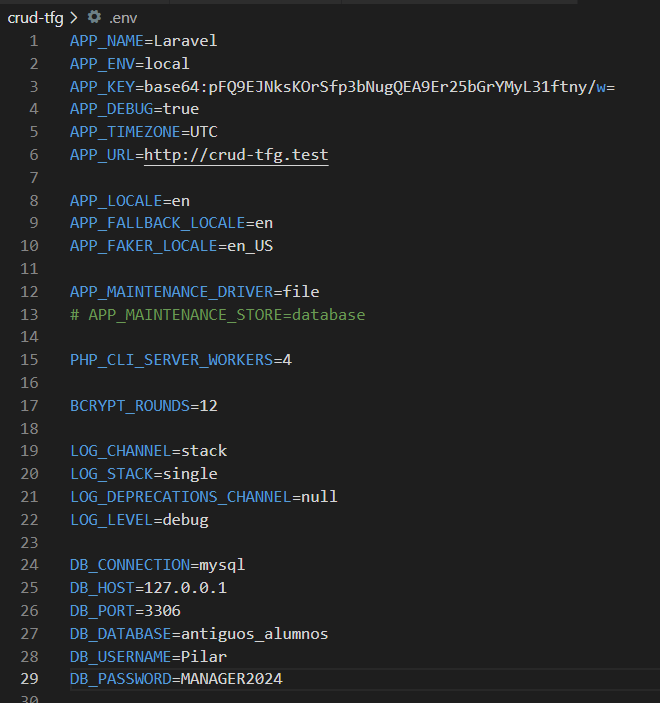
1. Entorno de desarrollo

El proyecto lo voy a realizar usando el framework Laravel. Laravel Herd es una herramienta de entorno de desarrollo creada por el equipo oficial de Laravel, que admite el cambio de versiones de PHP y Node. Después de instalar Herd, sirve a todos los sitios de tu máquina a través de dominios \*.test: es como Laravel Valet, pero no tiene dependencias y no necesita Homebrew. Viene con sus propios binarios precompilados, lo que hace que sea increíblemente rápido de instalar y usar.

Instalé Herd y comencé a trabajar con VSC desde sus archivos.

## Rutas

Los valores de configuración importantes en Laravel se definen mediante el archivo .env que existe en la raíz de nuestra aplicación. Así lo hice y quedó de la siguiente manera.



Donde entre otras cosas, señalé la conexión a la bbdd, datos de usuario y pasword.

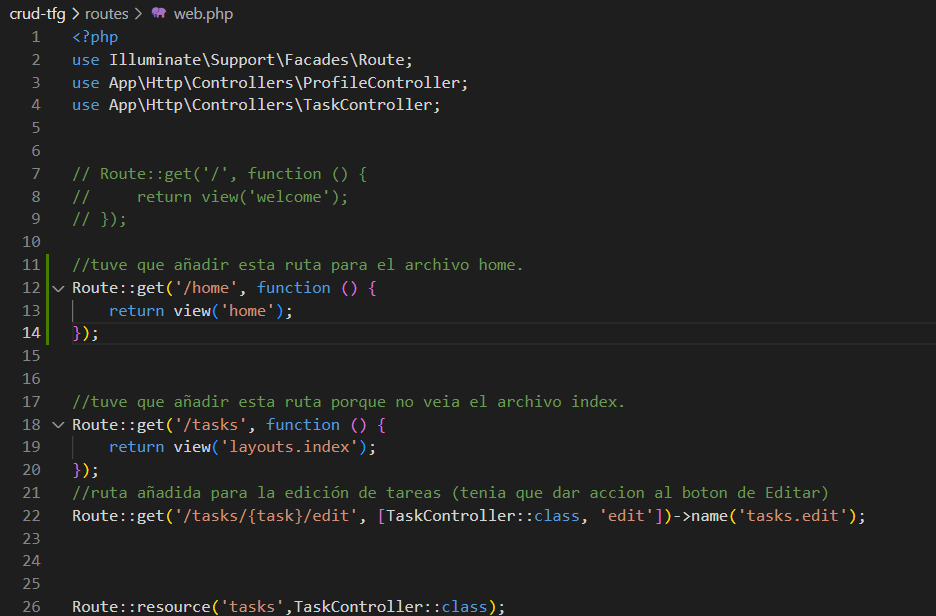
Una vez que ya tuve configurada la base de datos en MySql, pude ejecutar las migraciones de base de datos de mi aplicación. Que se harán mediante el código -php artisan migrate-

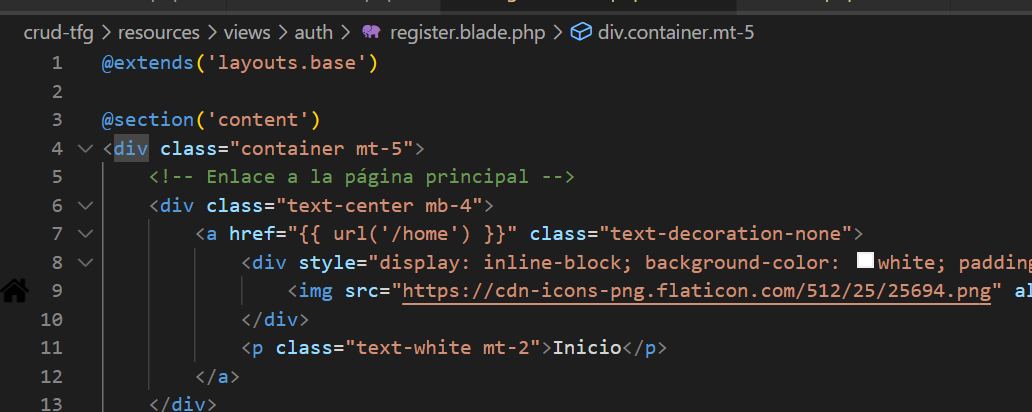
Lo primero que tuve que familiarizarme fue con el diseño de las rutas en Laravel. Todas las rutas de Laravel se definen en tus archivos de ruta, que se encuentran en el directorio routes. Estos archivos se cargan automáticamente mediante el App\Providers\RouteServiceProvider de tu aplicación. El archivo routes/web.php define rutas para tu interfaz web. Estas rutas se asignan al grupo de middleware web, que proporciona funciones como el estado de sesión y la protección CSRF.

Para la mayoría de las aplicaciones, comenzarás definiendo rutas en tu archivo routes/web.php. Las rutas definidas en routes/web.php se pueden acceder ingresando la URL definida de la ruta en tu navegador.

Nota: Cuando defines varias rutas que comparten la misma URI, las rutas que utilizan los métodos get, post, put, patch, delete y options deben definirse antes que las rutas que utilizan los métodos any, match y redirect. Esto asegura que la solicitud entrante coincida con la ruta correcta.

Comencé a definir las primeras rutas tal y como detallo en la imagen siguiente:

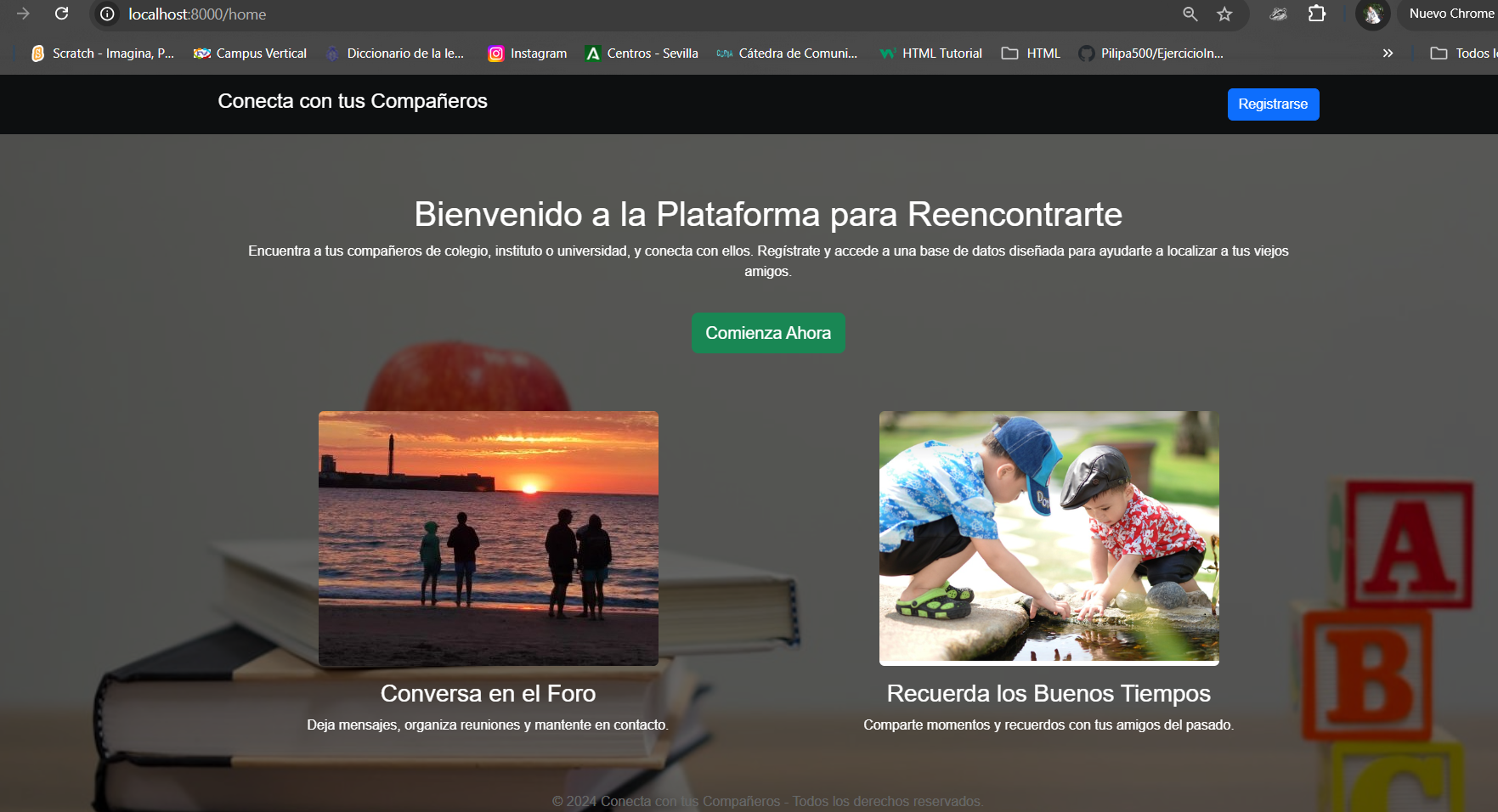


En Laravel se puede usar la directiva de Blade @extends para especificar qué diseño debe “heredar” la vista secundaria. En mi caso diseñé un archivo base.blade.php que me servía como de plantilla para el resto de vistas. Las vistas que extienden un diseño de Blade pueden inyectar contenido en las secciones del diseño mediante directivas @section.

Y realicé el archivo que sería mi pantalla inicial de la aplicación [home.blade.php](http://home.blade.php).

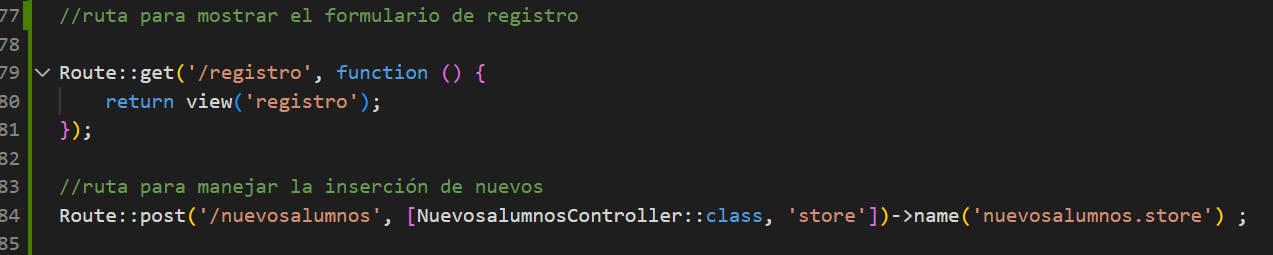
Blade es el motor de plantillas simple pero potente incluido con Laravel. A diferencia de algunos motores de plantillas PHP, Blade no te restringe a usar código PHP simple en tus plantillas. De hecho, todas las plantillas de Blade se compilan en código PHP simple y se almacenan en caché hasta que se modifican, lo que significa que Blade agrega prácticamente cero sobrecarga a tu aplicación. Los archivos de plantilla de Blade usan la extensión de archivo .blade.php y generalmente se almacenan en el directorio resources/views.

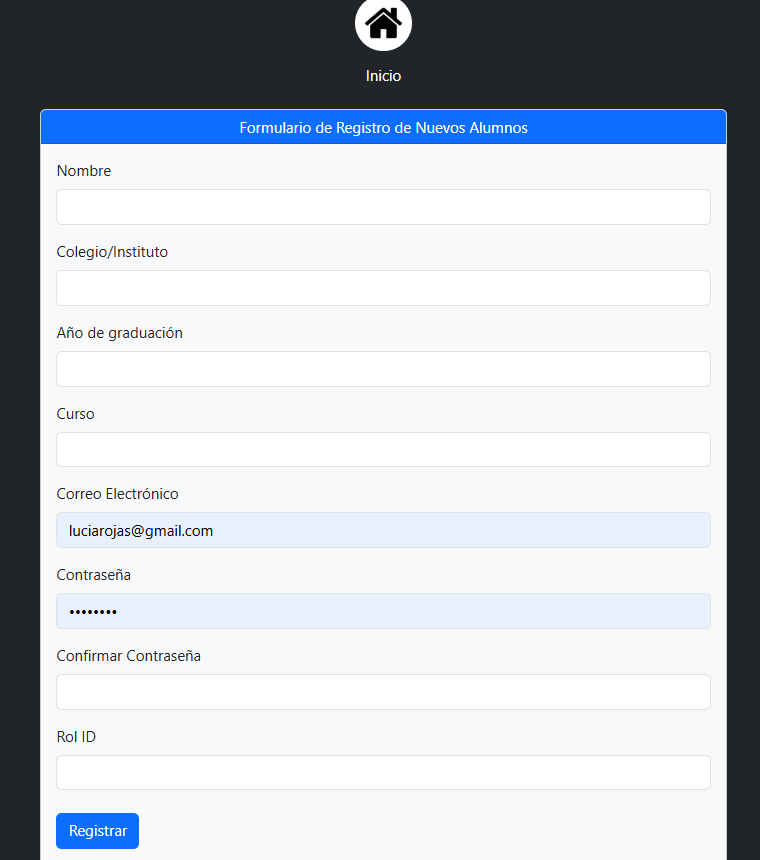
En esa dirección generé el archivo [home.blade.php](http://home.blade.php) donde se detallaba el “porqué” de esta web.



En esta primera página se describe brevemente como desde esta app podremos encontrar a los compañeros con lo que hubiera estados unidos en el pasado, por medio de nuestros centros de estudios. Con un registro inicial, estaremos insertándonos en la base de datos, desde la que podremos formar parte de esta comunidad de amigos.

Los dos botones que hemos puesto aquí, tanto el de “Registrarse” como el de “Comienza Ahora”, dirigen al formulario de registro, donde se iniciará el proceso de acceso.

Antes de editar el formulario de registro, nos vamos al archivo de web.php donde, como ya hemos explicado más arriba detallamos las rutas de todo nuestro proyecto 

En este caso damos las rutas para que un usuario invitado pueda crear y almacenar su registro, rellenando el formulario al que se dirige desde el home. 

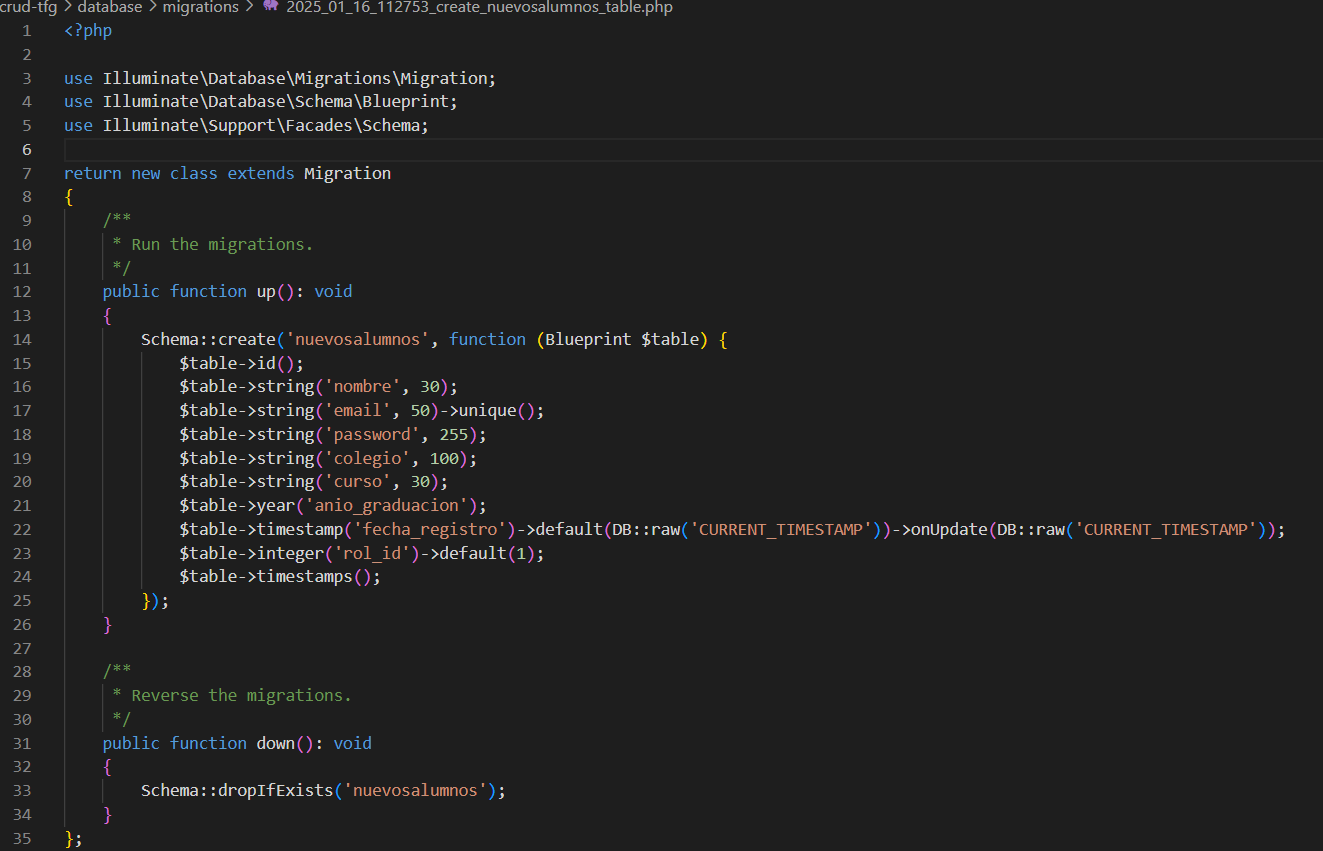
(debo añadir el archivo del formulario-registro y su código, con el enlace a la lista de alumnos)

Como se pudo observar en la imagen del diseño de la bbdd que tiene este proyecto, había creado la “tabla nuevosalumnos” con la que alineamos las entradas del formulario, para que se insertaran correctamente.

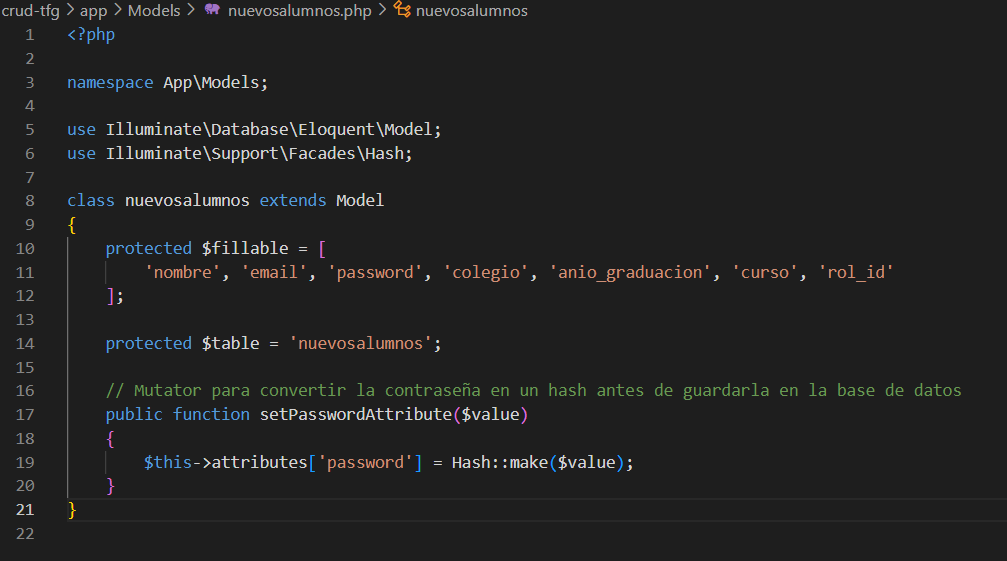
Me aseguré de que la migración de la tabla usuarios, estuviera correctamente definida en Laravel. Y cree una migración con el comando php artisan make: migration create\_nuevosalumnos\_table

Abrí el archivo de migración recién creado en database/migrations/.

Y dentro del método up, definí las columnas según la estructura de la base de datos. Aquí dejo el ejemplo basado en la imagen del diseño de la bbdd:



y definí las columnas según la estructura de la base de datos en App/Models creando el archivo nuevosalumnos.php.



**Ejecutar la Migración**:

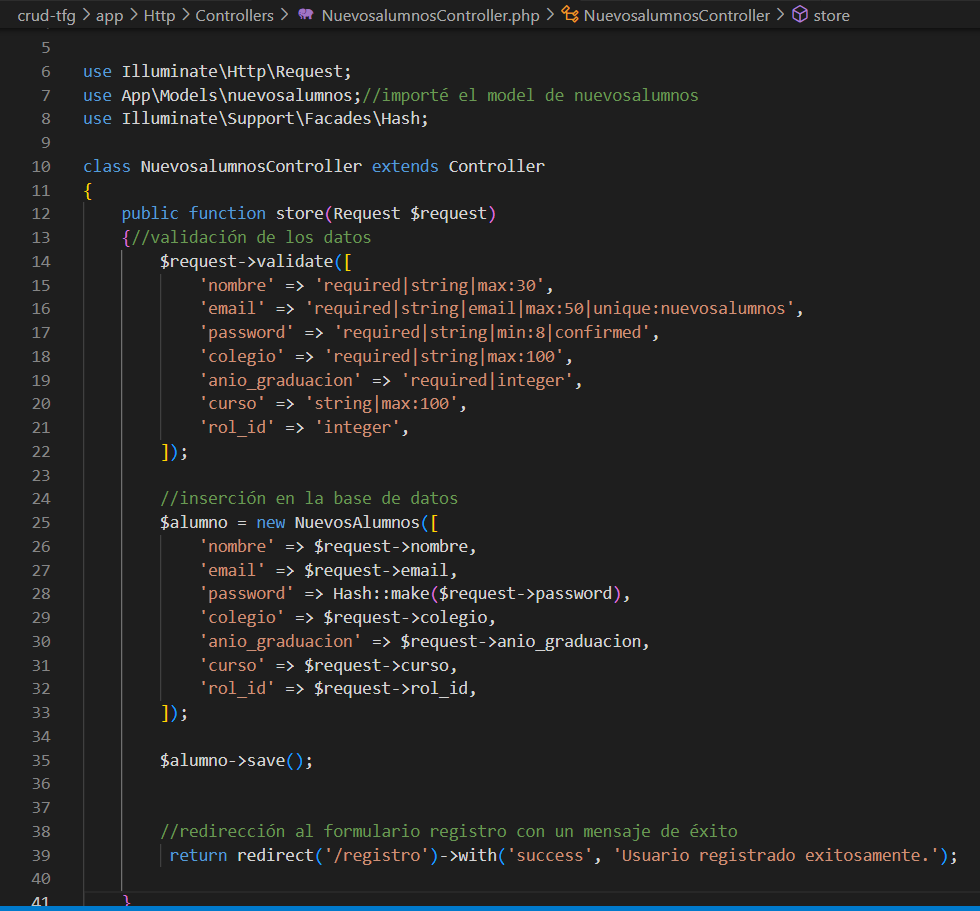
Una vez que he definido las columnas, ejecuto la migración con el siguiente comando: php artisan migrate

Esto creará la tabla **nuevosalumnos** en la base de datos con las columnas especificadas.

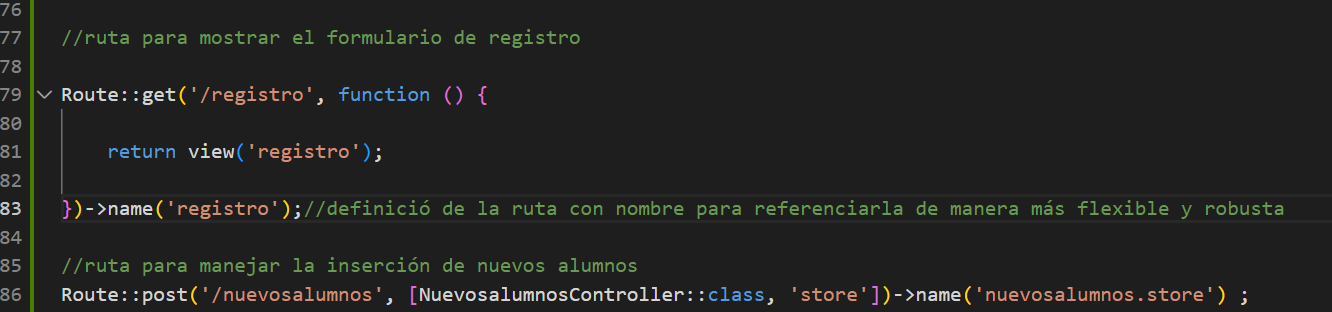
De igual manera, tendré que hacer lo mismo para las demás tablas que vaya a usar. Como serán las que se ven en la bbdd de más arriba. Como son: Amigos, roles y mensajes. Así que voy a generar ya los archivos con el comando php artisan make: migration create\_ejemplotabla\_table y los tendré ya escritos para cuando los vaya a usar.

Controlador:

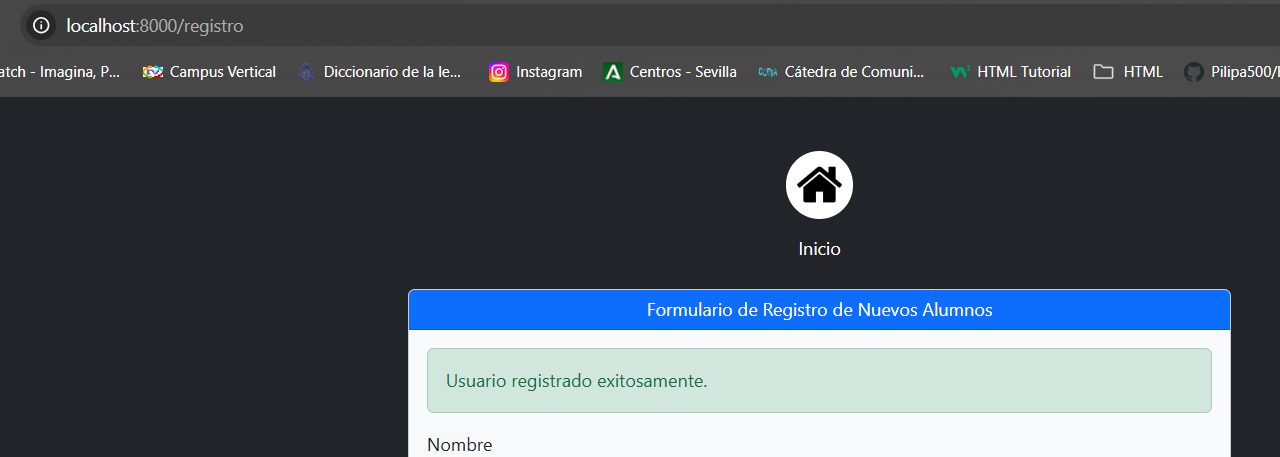
A continuación creamos un controlador para manejar el registro de usuarios con el comando php artisan make:controller NuevosalumnosController. En el método de registro, valida y guarda los datos del formulario:

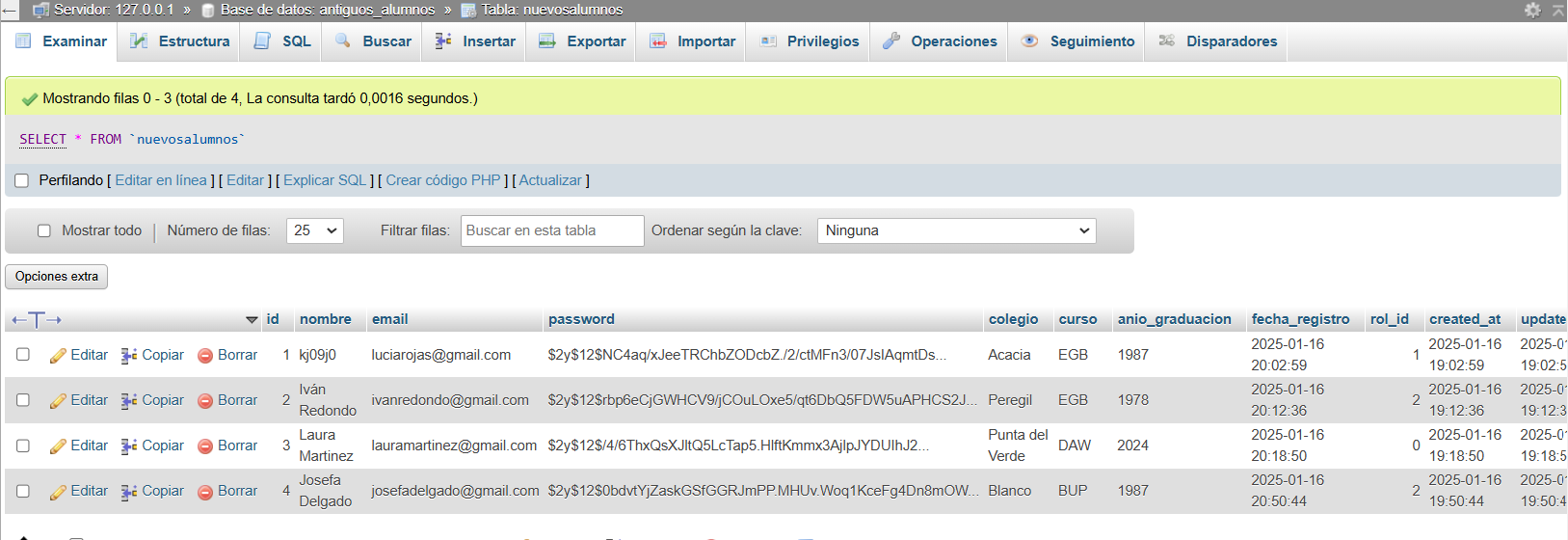


A continuación se definen las rutas necesarias en web.php



Añadí varios alumnos en el formulario registro y me salieron insertados correctamente en la bbdd de MySQL





Y vemos en la imagen anterior como se han insertado los cuatro alumnos que registré.

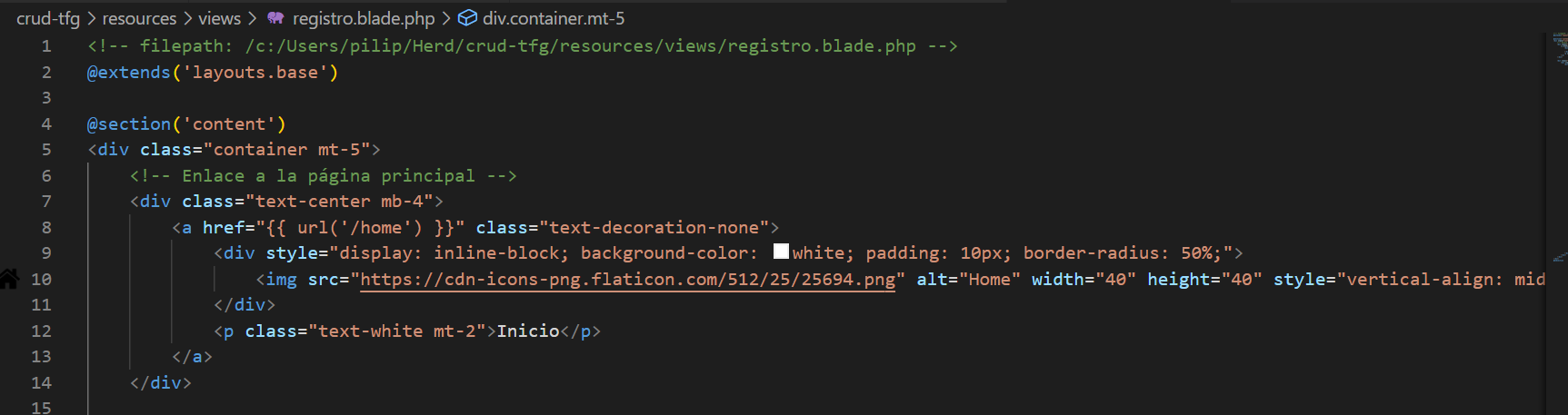
Hasta aquí todo va correctamente- Ahora voy a ver por donde sigo:

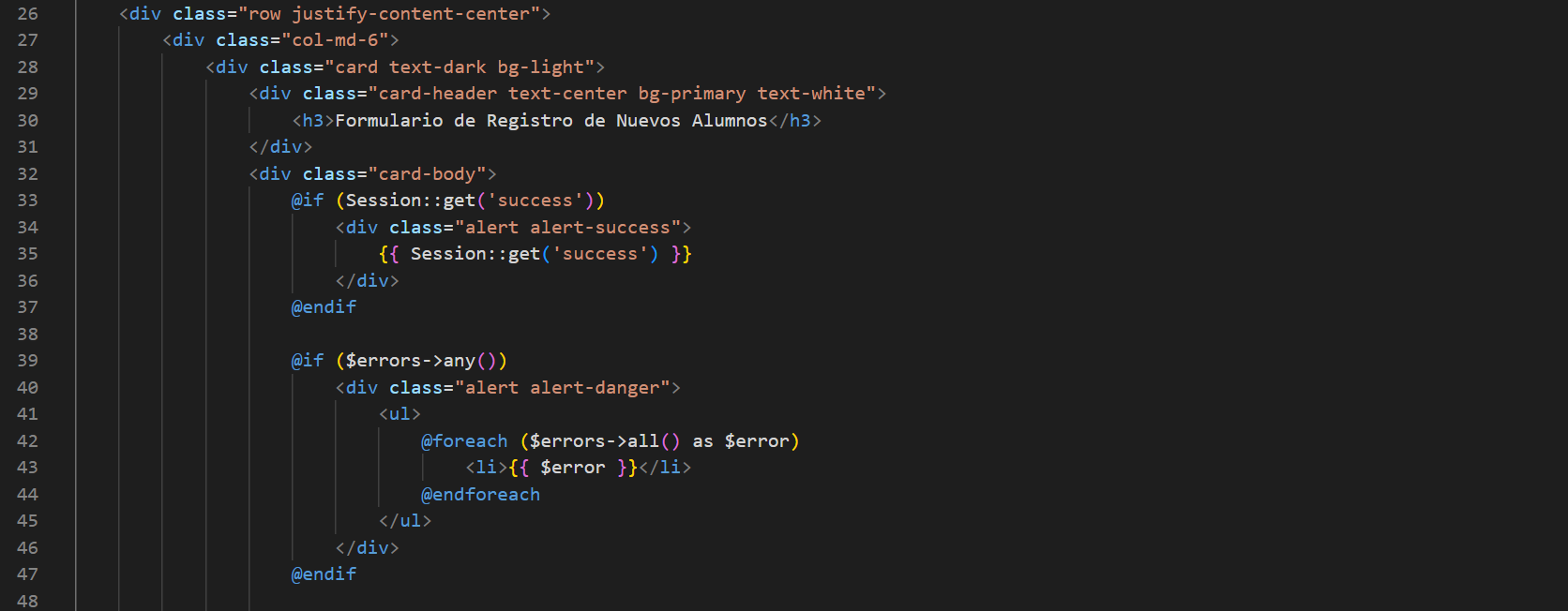
**Desde aquí -----------------------------------------------------------------------------**

**Búsqueda de alumnos**

Me plantee la posibilidad de hacer búsquedas por los alumnos registrados, una vez que se muestre una lista con los alumnos que hay. Dicha búsqueda podrá ser por nombres, colegios, cursos, años… Añadiendo un enlace, quizás desde la misma página de registro, una vez que ya fue correctamente registrado el alumno.

La imagen siguiente es el formulario registro que tengo hasta ahora, que no mostré más arriba…



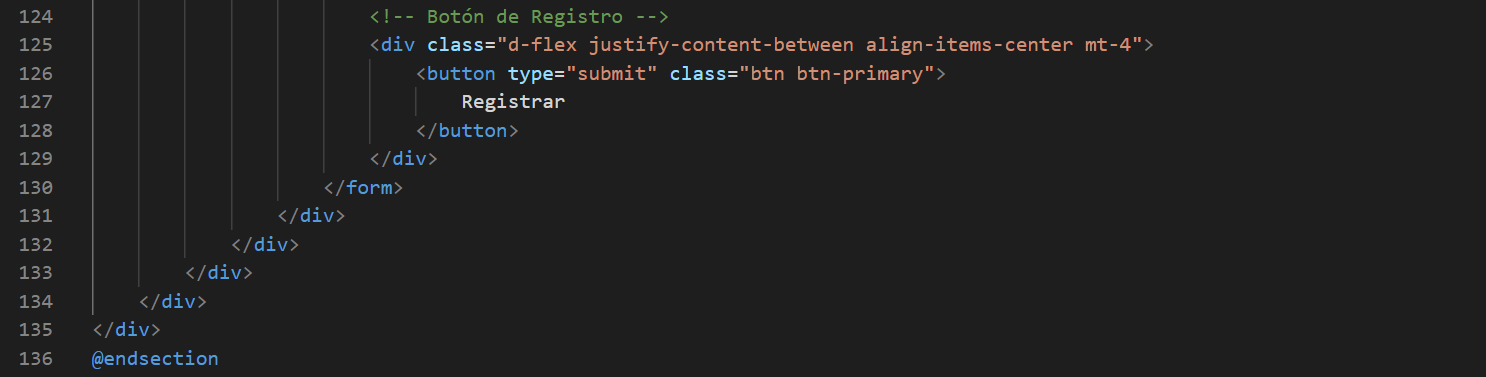






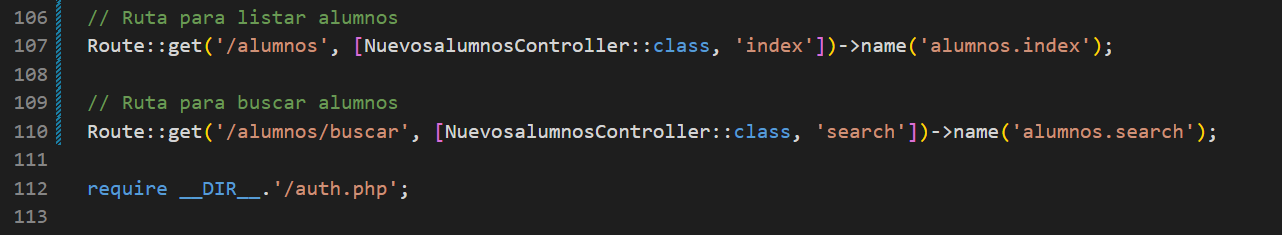






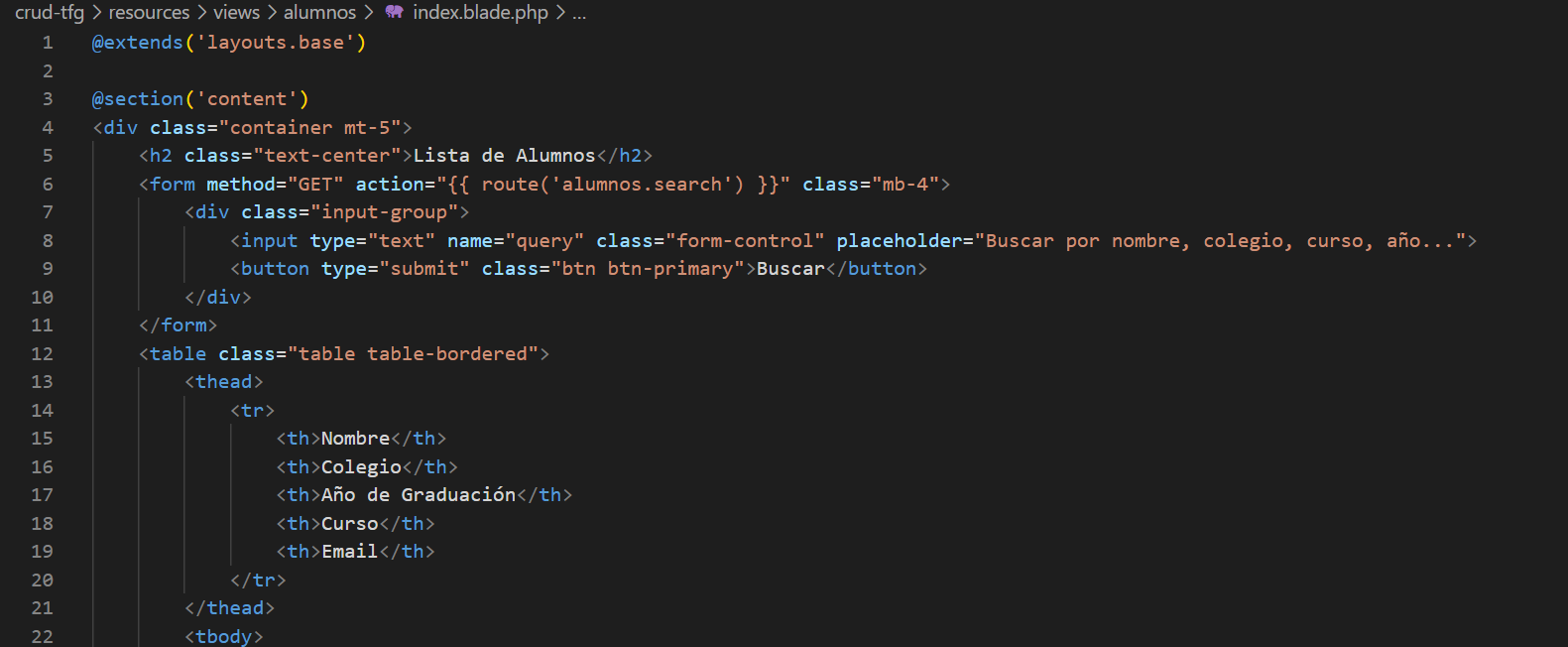
Comencé, como siempre por pensar en cómo crear una vista para listar los alumnos registrados:

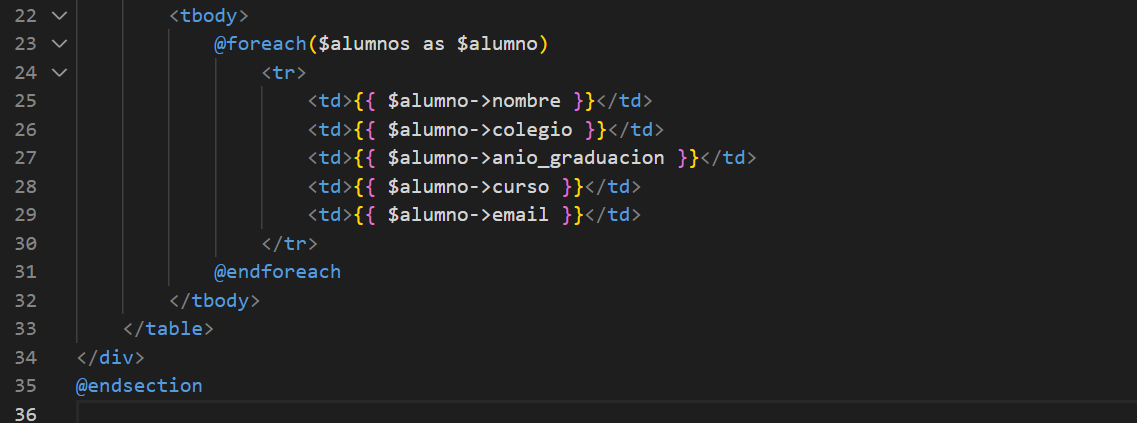
1. Añadir rutas para listar y buscar alumnos.



Para poder acceder a esas rutas, tuve que hacer una vista nueva que se llama alumnos y un index nuevo con la tabla en la que se listará la búsqueda de los alumnos.

1. Añadí funcionalidad de búsqueda en la lista de alumnos





1. Añadir métodos adicionales en el NuevosalumnosController para manejar la creación, visualización y edición de alumnos.
2. Añadir enlaces desde la página registro y la página de inicio para acceder a la lista de alumnos.

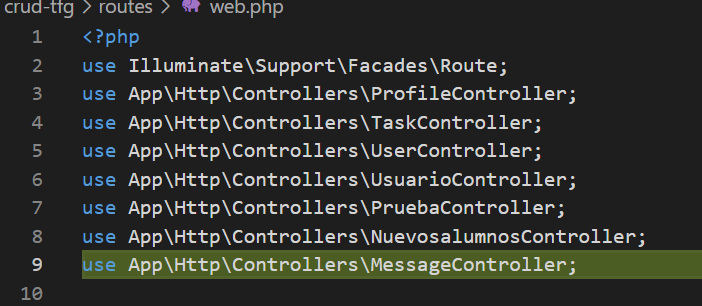
Hasta aquí---------------------------------------------- intenté insertar el sistema de búsqueda de alumnos, pero no salió. Al parecer el alumnos.index no se enrutó bien. Me pasé dos días repasando las rutas y aparentemente estaban bien. También repasé todos los enlaces donde los había insertado y nada. Así que anulé todo lo editado y volví al principio. Borré también la vista/alumnos con su index. Osea el último archivo al que hice pantallazo aquí.

A continuación como me atasqué con eso seguí con el sistema de mensajería..

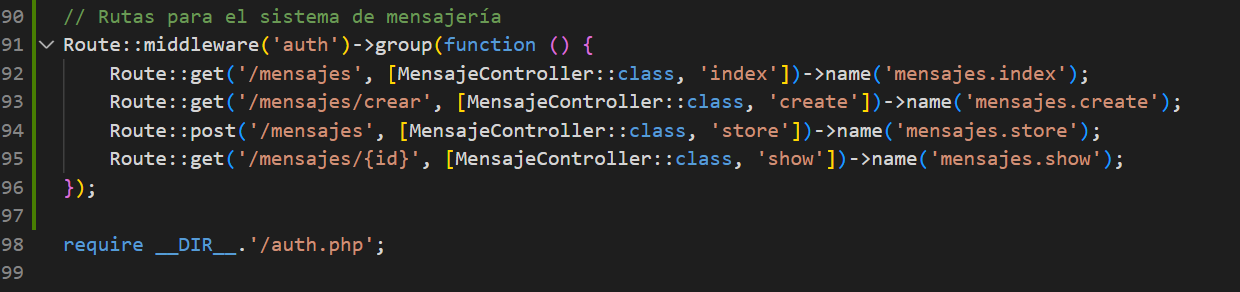
Primero intentaré el sistema de mensajeria

1. En routes/web.php

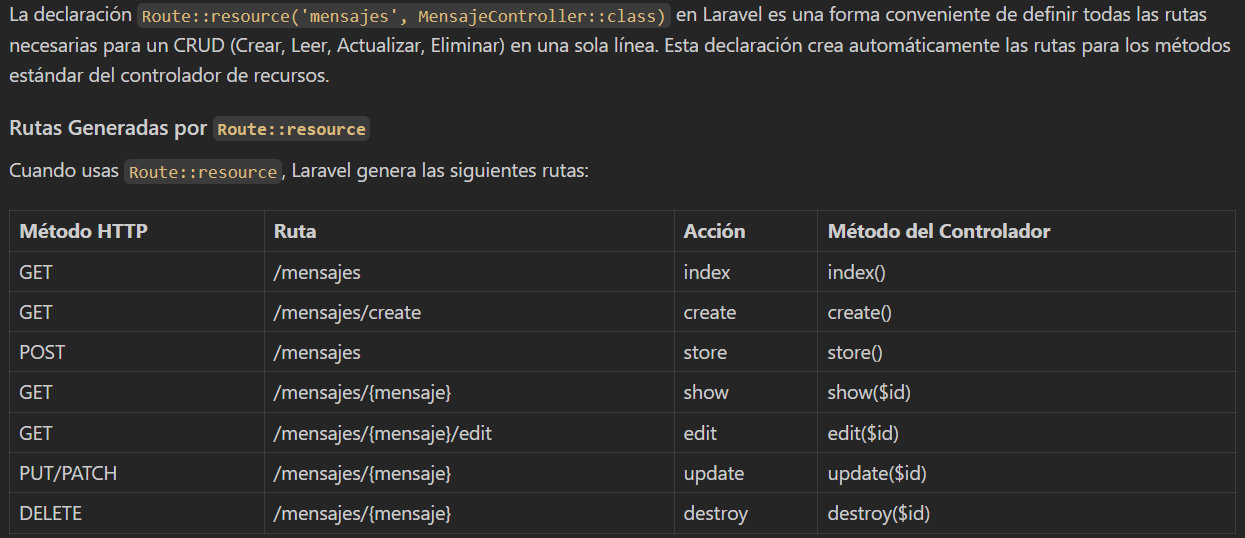
Añado el controlador y la ruta al enrutador

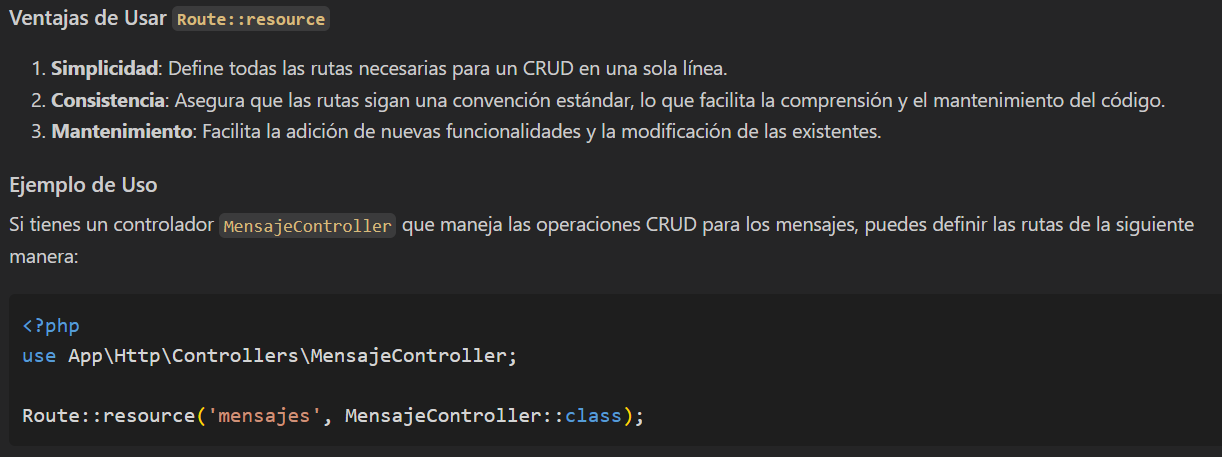


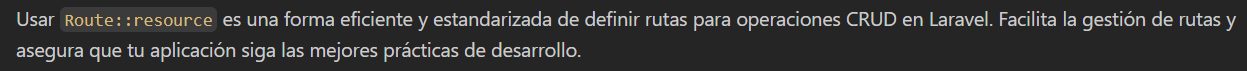




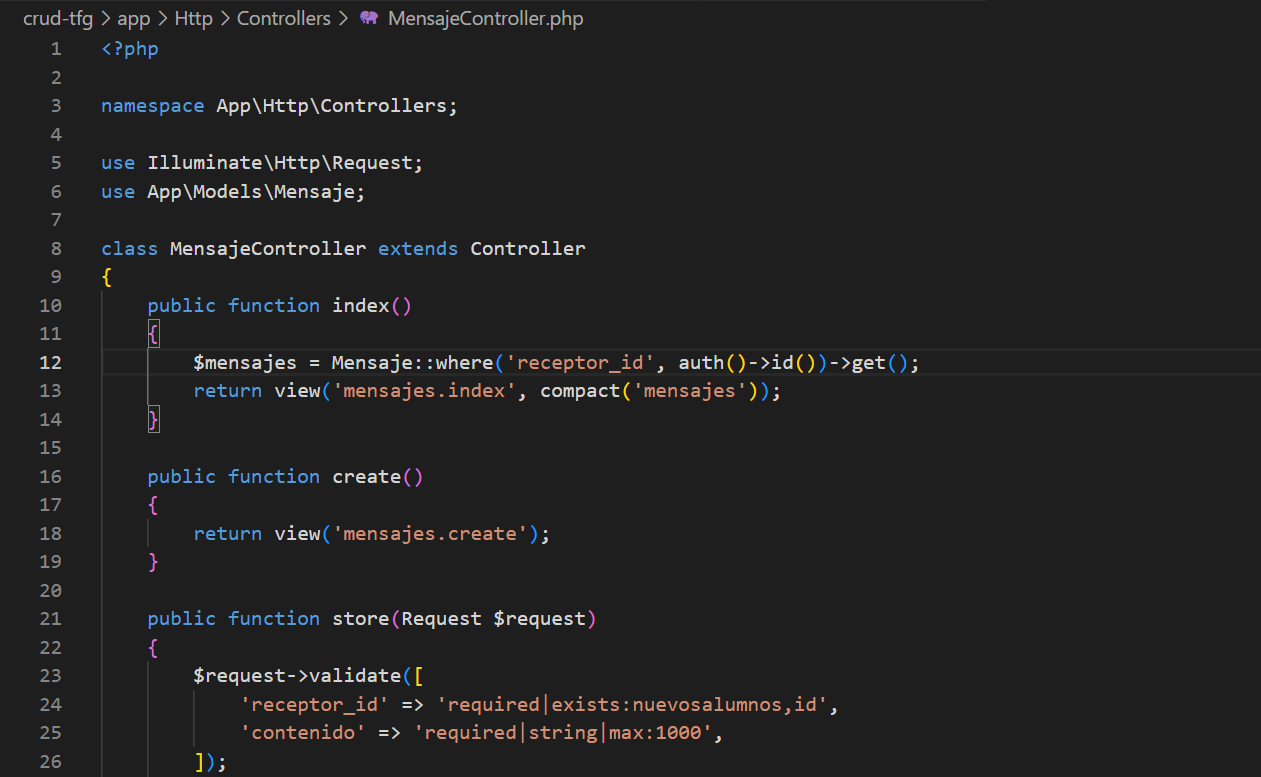
En este punto recordé que había visto como al seguir la “guía” que estudié para el CRUD de TASK, para iniciar este proyecto había una ruta que solo escribía una línea para todas la funciones. Y se trata de la declaración Route::resource

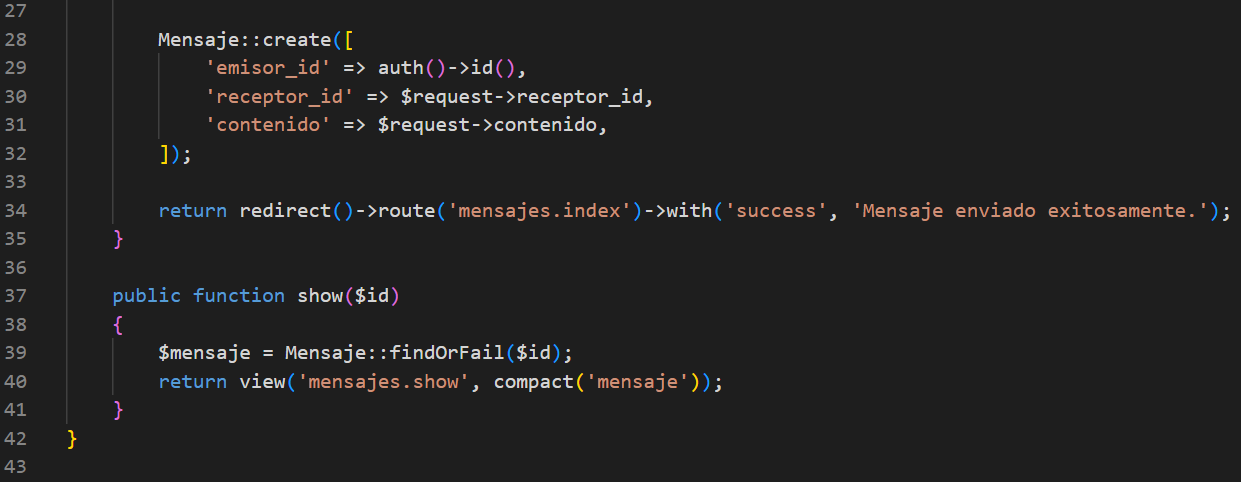




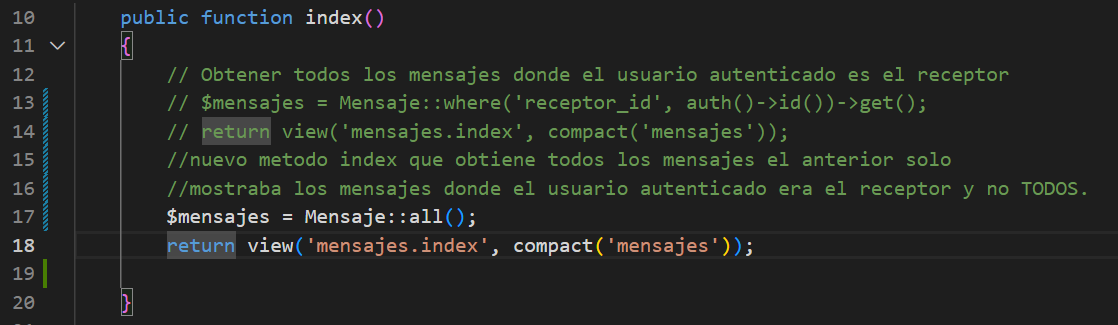


1. Luego crearía el Controlador para la mensajería

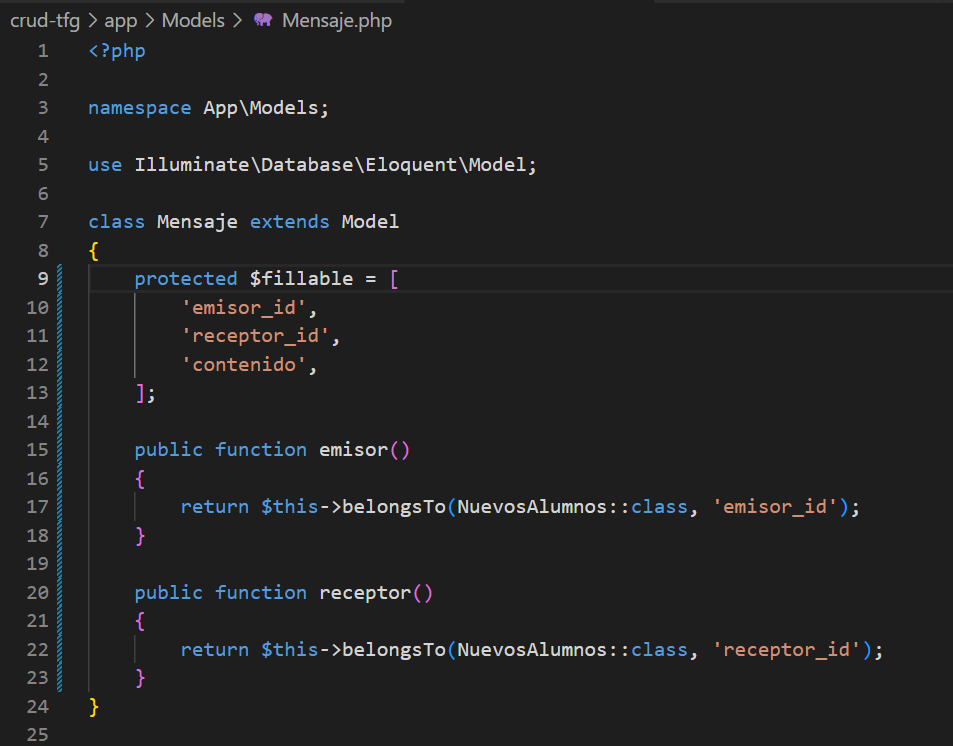




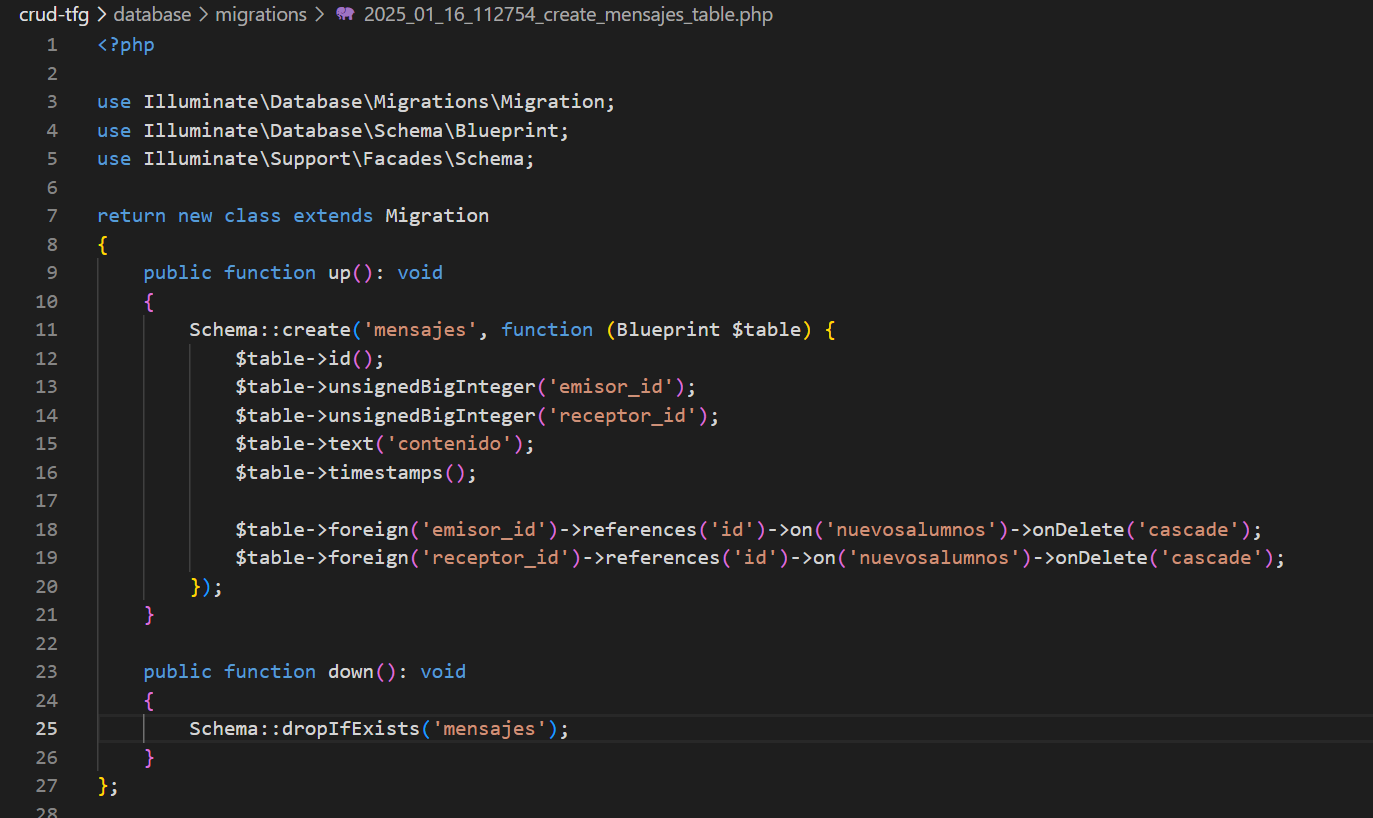
[Tuve que modificar la función index porque solo se listaba un mensaje y no todos. Con esta modificación salían todos](#ControladorMensajes). Indistintamente de quien lo enviaba



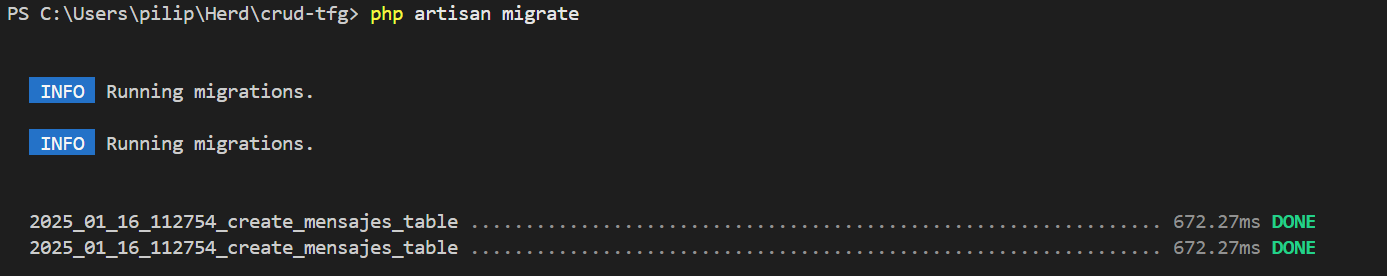
1. Hice un modelo para los mensajes



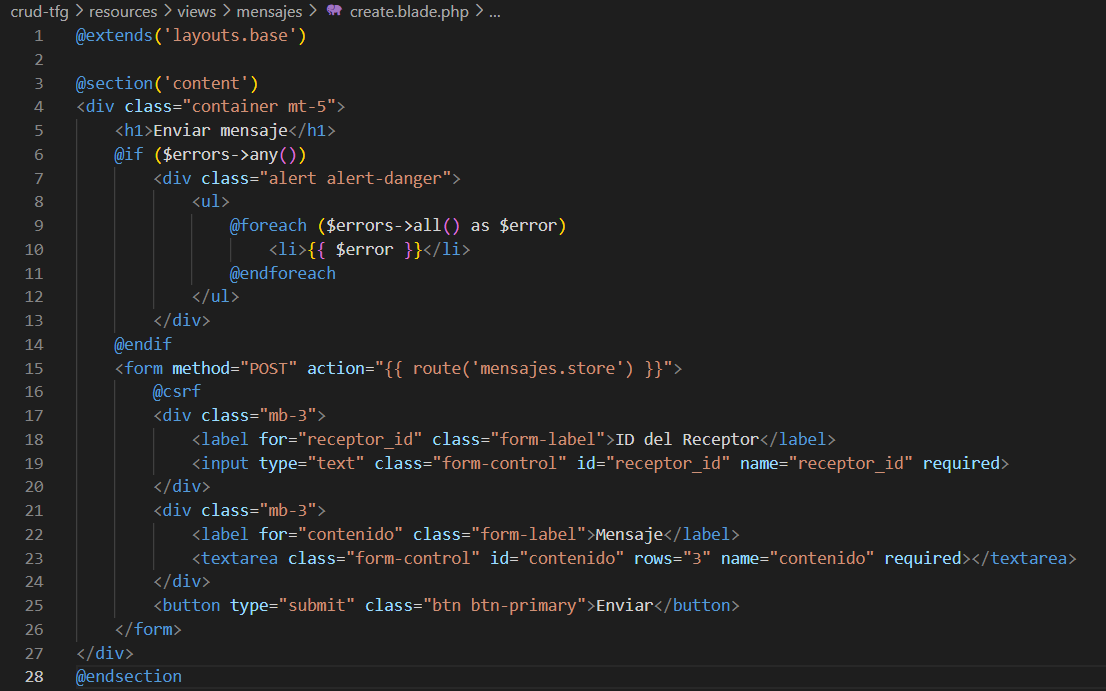
1. Crear una migración para la tabla de mensajes. Aquí debo tener cuidado con la tabla a la que apunta el id, porque deberá ser la **tabla nuevosalumnos.**



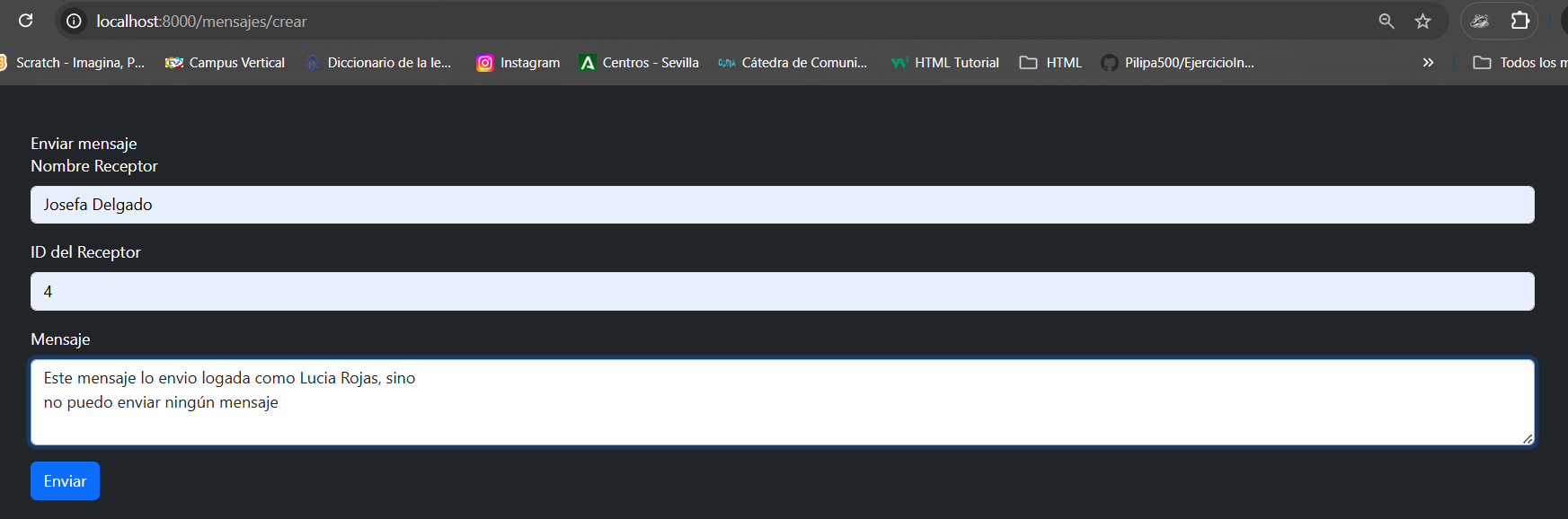
Una vez que ya estaba diseñada la tabla ejecuté la migración. No sé por qué razón salió dos veces en la pantalla del terminal.



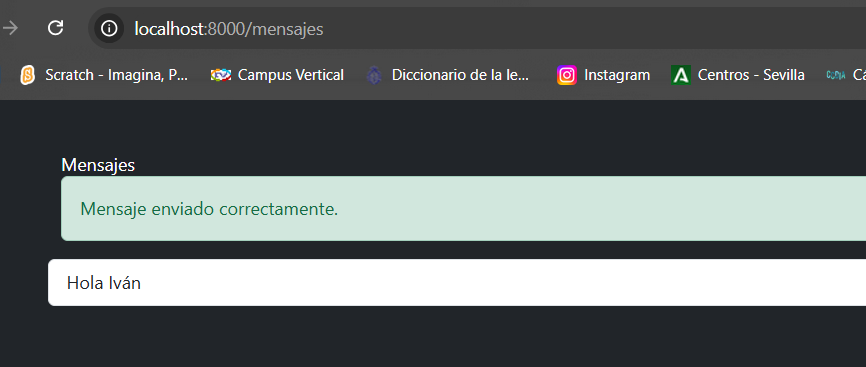
1. Crear una vista para enviar nuevos mensajes



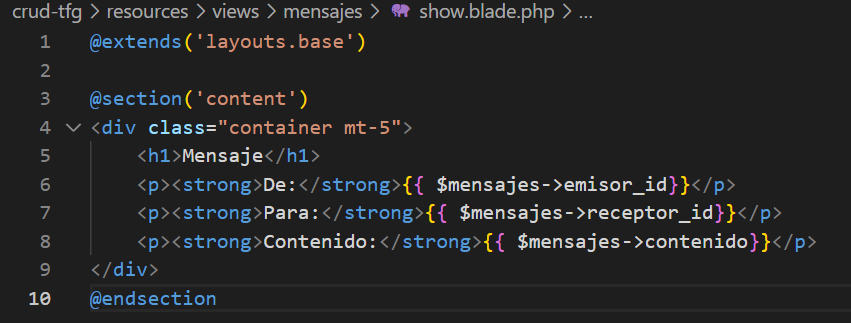
Este se vería en el navegador así:



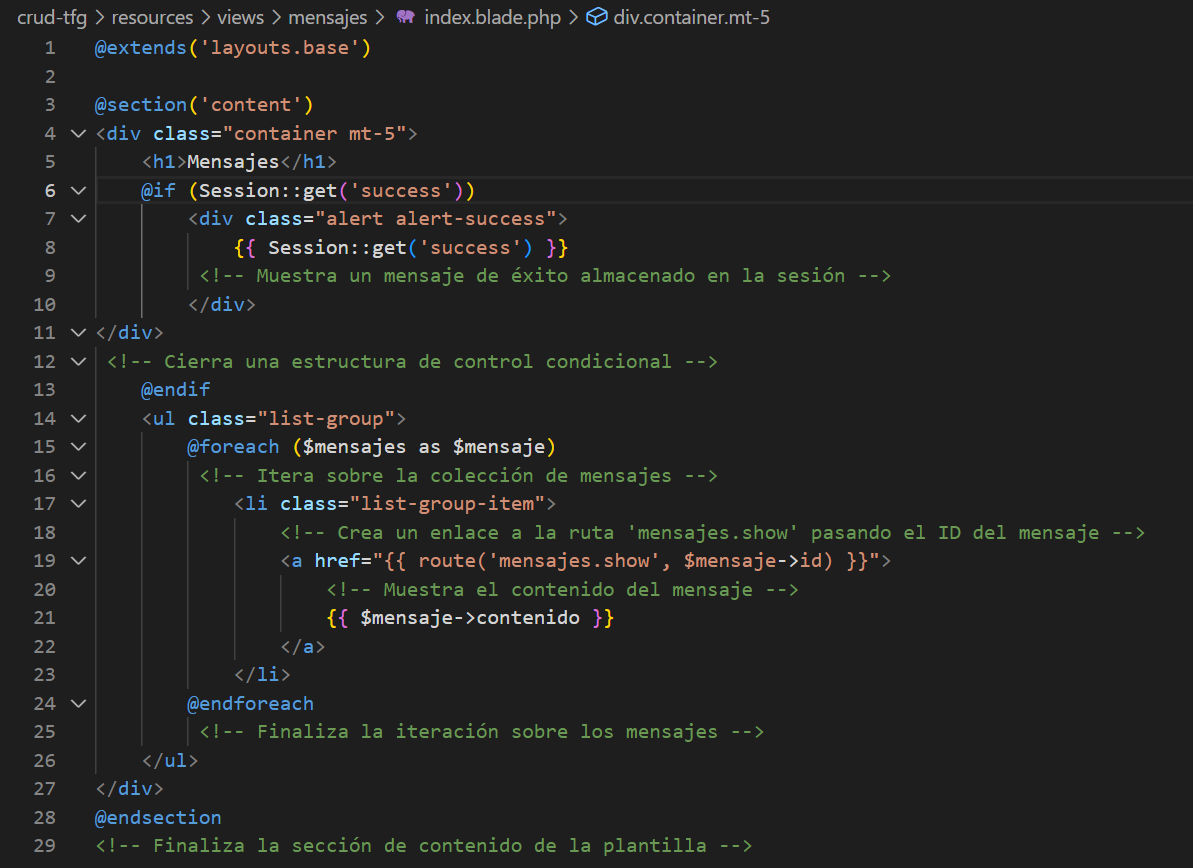
No se vería esta página, hasta que el alumno no estuviera logado… Y si se pulsa en enviar, sale el mensaje de enviado correctamente y sólo se lista el primero



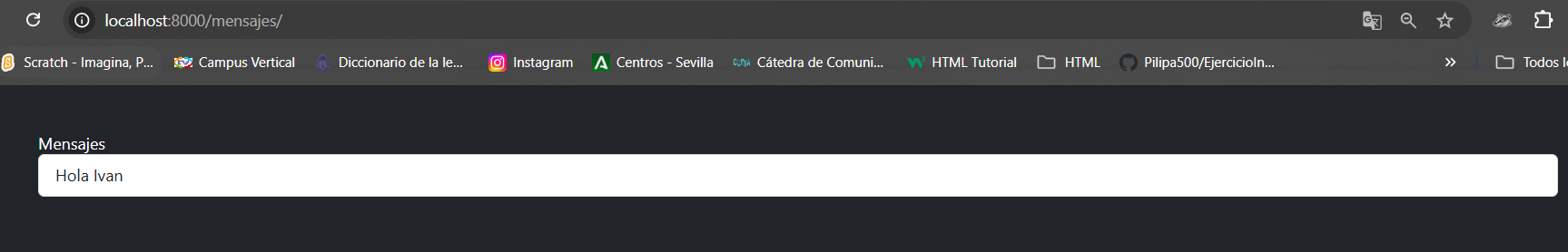
1. Crear una vista para mostrar un mensaje específico



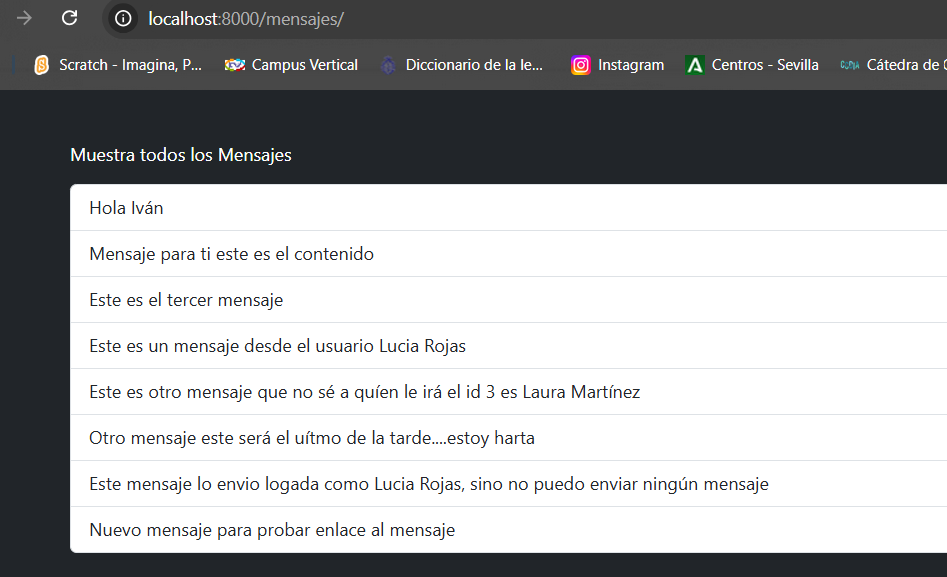
1. Crear una vista para listar los mensajes. Puse los comentarios a este código, está en el VSC, no En el pantallazo.



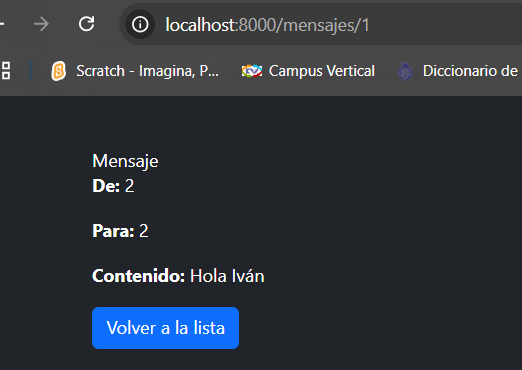
Por ahora solo se ve un mensaje y no se listan los 4 o 5 que hay generados



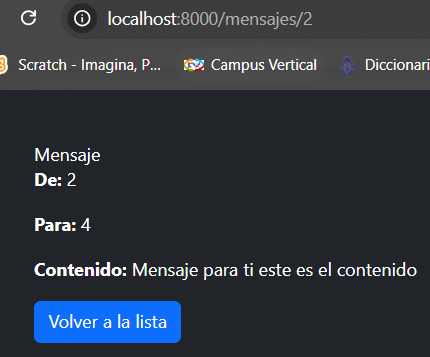
Ya si se listan todos (como se ve en la imagen de abajo), después de haber cambiado la función index. En el controlador de mensajes

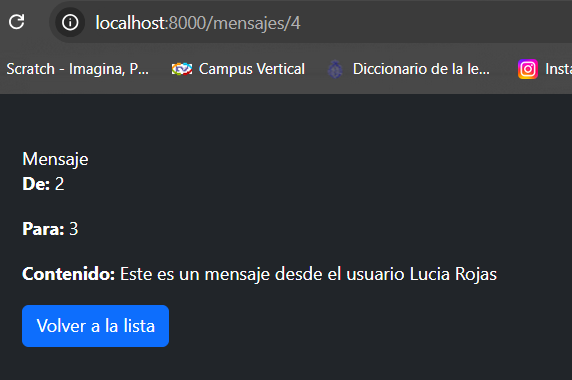


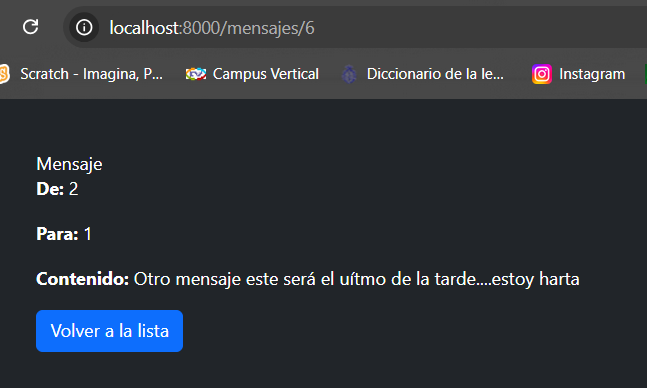
Si se pincha en el mensaje, puse un link, para que mostrara el mensaje en este caso es el 1



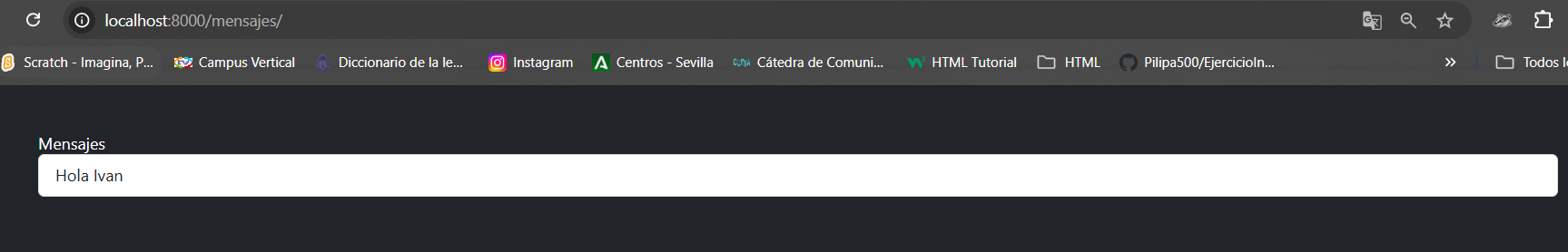
Si cambio en el navegador y pongo el 2 sale el segundo

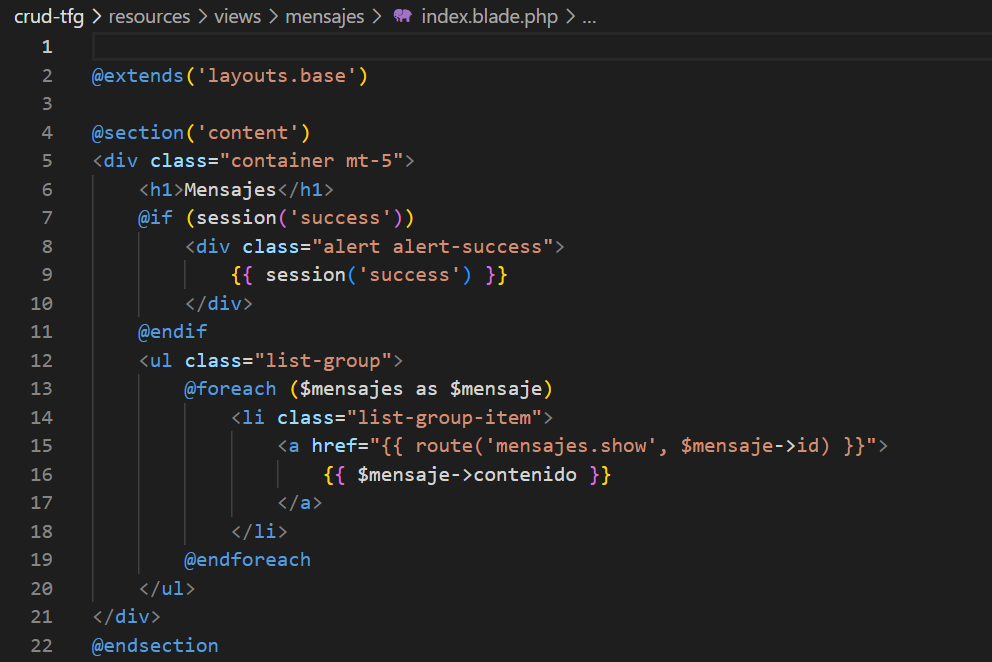






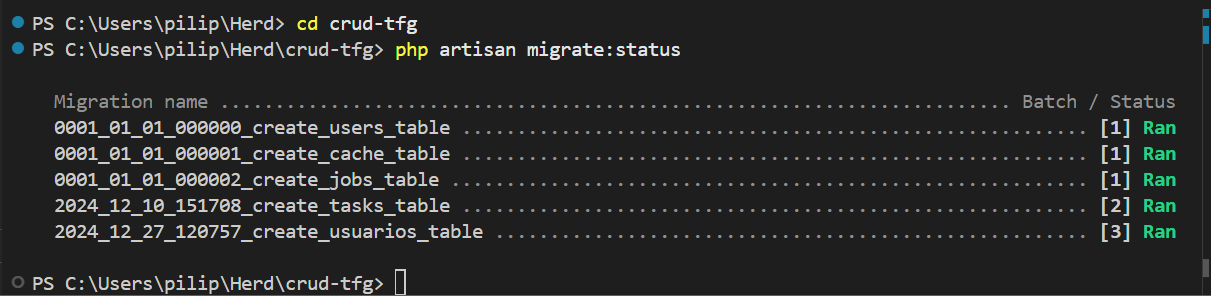
Si pincho en Volver a la lista el único que sale en el listado es el 1





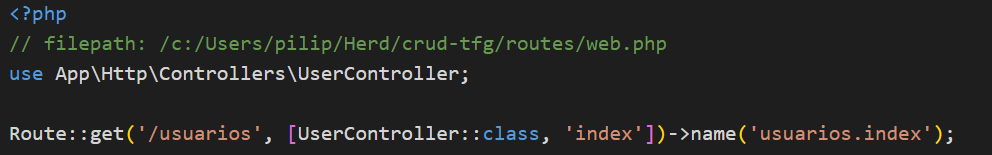
Por aquí me quedé

Al añadir usuarios registrados, observé que “aparentemente”, parecían guardarse, pero no lo veía añadido en la bbdd de MySQl. Así que seguí, comprobando que la conexión con la bbdd.

 Que aunque aparecía “corriendo” en el terminal, no lo veía claro. No salían los nuevos usuarios en la tabla de MySQL.

Así que decidí, **crear una nueva vista (nuevo archivo también)** para listar los usuarios que se van añadiendo a la plataforma de AAA.

1. Creé una ruta para la vista de usuarios:

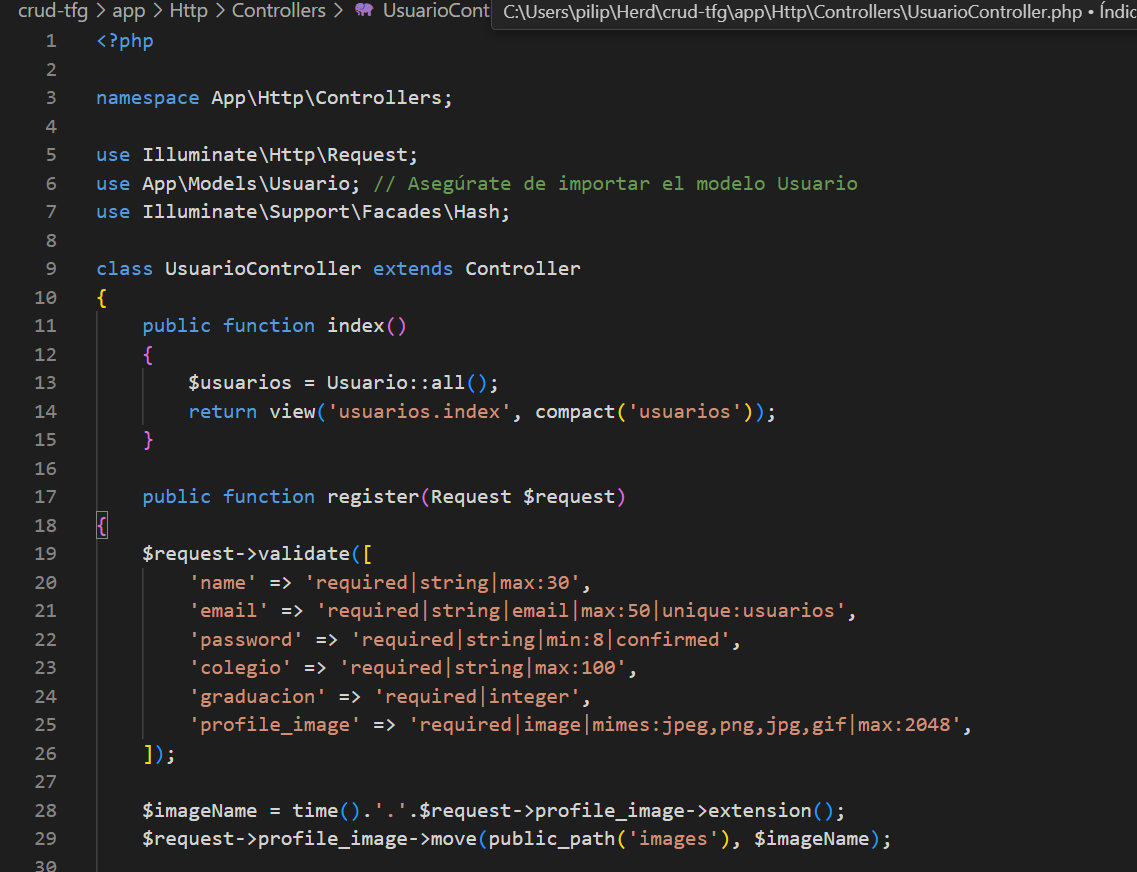


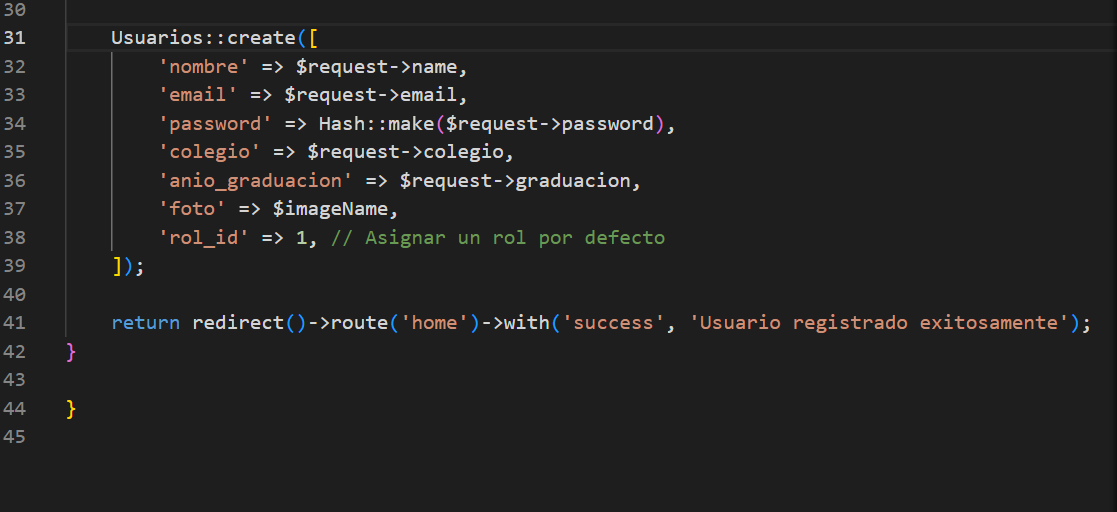
El archivo completo está así, actualmente.



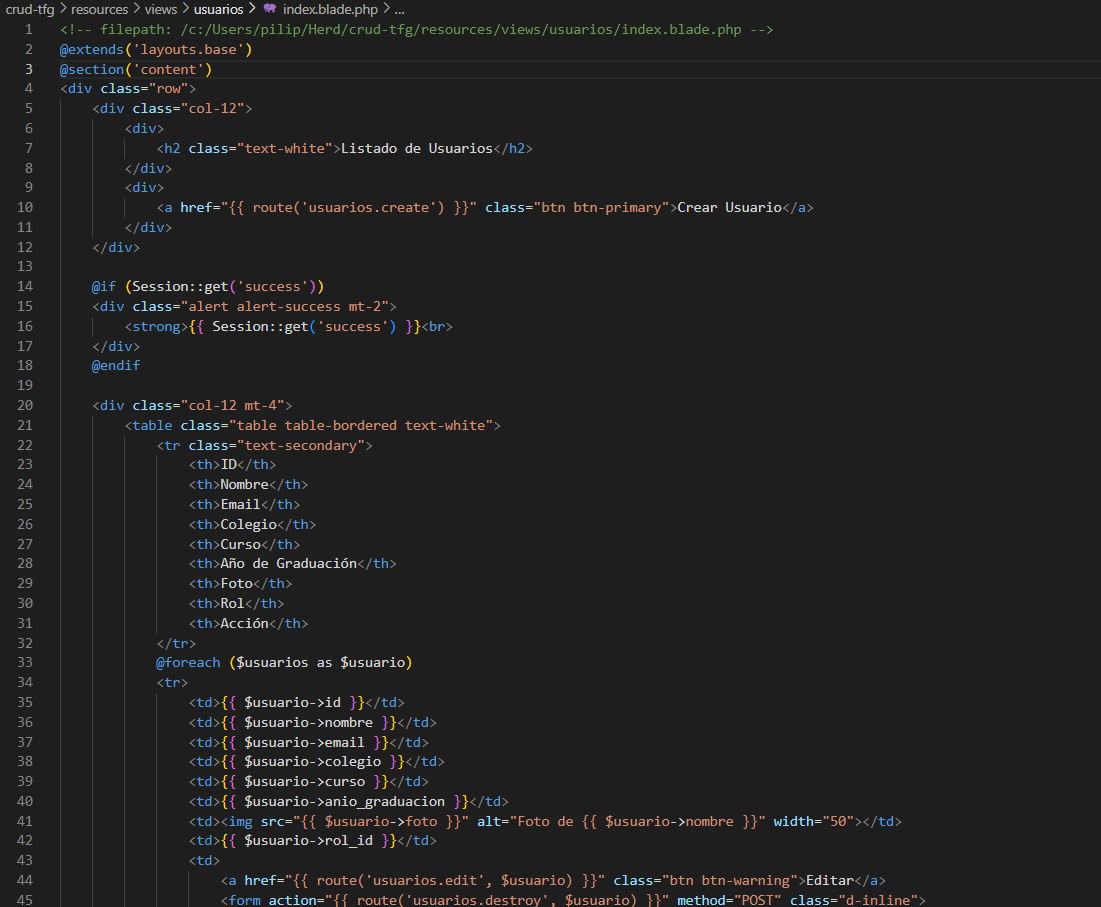


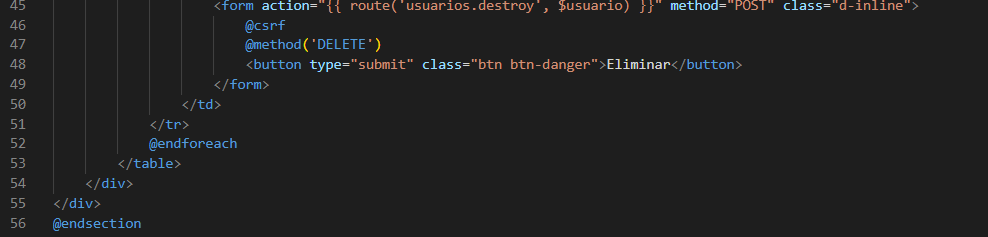
1. Utilicé el mismo `UsuarioController` para añadir el código necesario para listar los usuarios. Solo necesite añadir el método `index` y asegurarme de que el modelo `Usuario` esté importado correctamente. Aquí puse como lo hice:





Crear la vista para listar los usuarios. Hice un nuevo archivo de vista index.blade.php en la carpeta resources/views/usuarios.

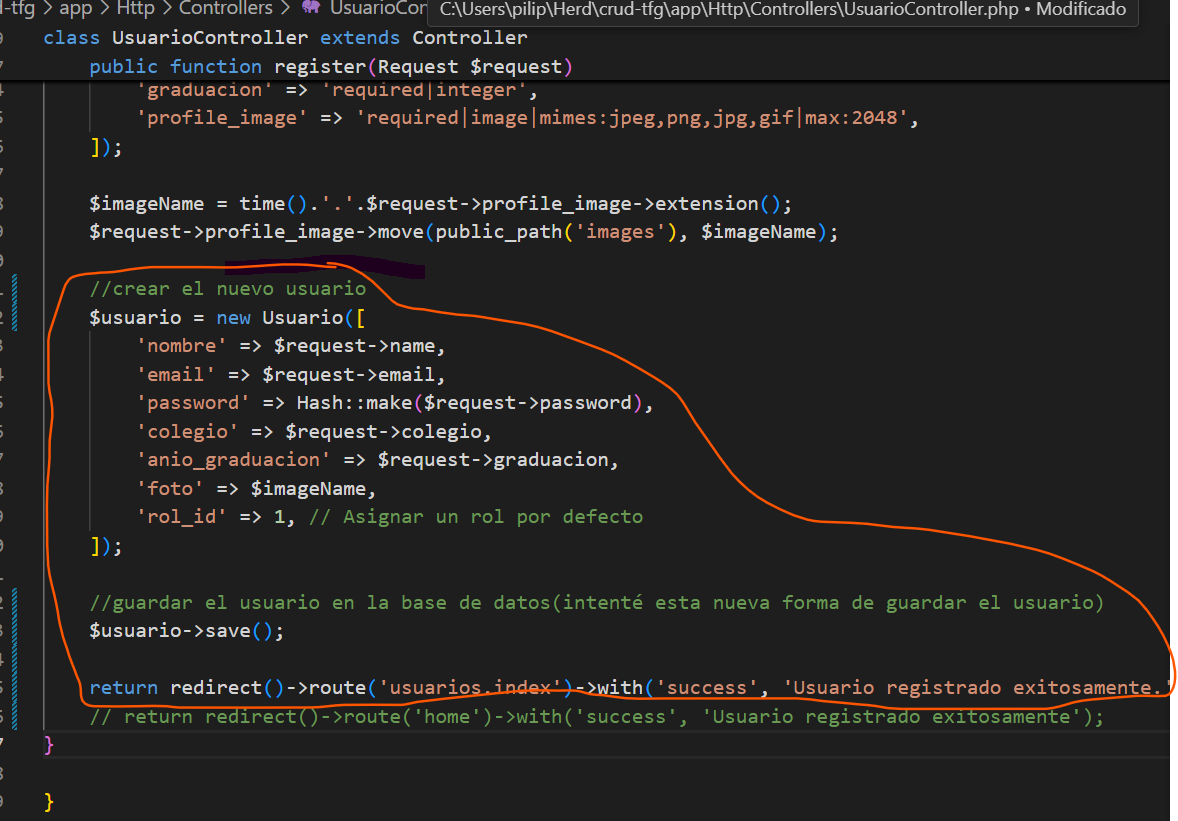




Ahora debería poder acceder a la vista de usuarios visitando la URL /usuarios en mi navegador.

PERO NO ES ASÍ!!! ALGO ESTÁ PASANDO Y NO SÉ QUE ES!!!

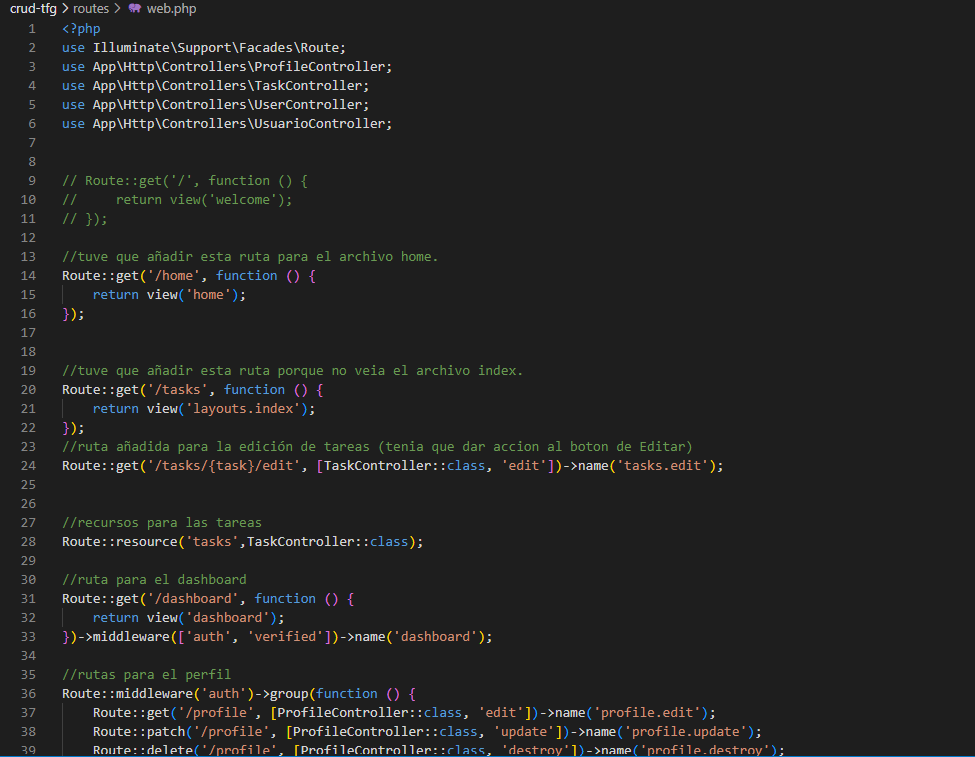
Se están guardando los usuarios en la tabla user de la bbdd y no en la de usuarios, que es la que destiné para el registro. A pesar de que está corriendo y activada, según el terminal. Pero al intentar abrir la url /usuarios no se abre el listado que he hecho.

Cambié el controlador en esta parte del código

Y volví activar el servidor local. La realidad es que a cada poco debo hacerlo. En el terminal ejecuto en la dirección del proyecto

php -S localhost:8000 -t public y reactivo el servidor.

Pensé que para poder ver la vista de usuarios, debía añadir en las dirección web, también su vista y toda la definición de las rutas para el CRUD de usuarios, por el administrador. Con lo que el archivo quedó así en C:\Users\pilip\Herd\crud-tfg\routes\web.php:





Al pensar hacer un CRUD, parecido al de “task” guardé las posibles rutas que iría creando, para el perfil del “administrador” de la plataforma.

Ahora modifiqué el archivo de UsuarioController.php. Donde tuve que añadir los métodos correspondientes para manejar las rutas, que acababa de crear:

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use App\Models\Usuarios;

use Illuminate\Support\Facades\Hash;

class UsuarioController extends Controller

{

public function index()

{

$usuarios = Usuarios::all();

return view('usuarios.index', compact('usuarios'));

}

public function create()

{

return view('usuarios.create');

}

public function store(Request $request)

{

$request->validate([

'name' => 'required|string|max:30',

'email' => 'required|string|email|max:50|unique:usuarios',

'password' => 'required|string|min:8|confirmed',

'colegio' => 'required|string|max:100',

'graduacion' => 'required|integer',

'profile\_image' => 'required|image|mimes:jpeg,png,jpg,gif|max:2048',

]);

$imageName = time().'.'.$request->profile\_image->extension();

$request->profile\_image->move(public\_path('images'), $imageName);

$usuario = new Usuarios([

'nombre' => $request->name,

'email' => $request->email,

'password' => Hash::make($request->password),

'colegio' => $request->colegio,

'anio\_graduacion' => $request->graduacion,

'foto' => $imageName,

'rol\_id' => 1,

]);

$usuario->save();

return redirect()->route('usuarios.index')->with('success', 'Usuario registrado exitosamente.');

}

public function show($id)

{

$usuario = Usuarios::findOrFail($id);

return view('usuarios.show', compact('usuario'));

}

public function edit($id)

{

$usuario = Usuarios::findOrFail($id);

return view('usuarios.edit', compact('usuario'));

}

public function update(Request $request, $id)

{

$request->validate([

'name' => 'required|string|max:30',

'email' => 'required|string|email|max:50|unique:usuarios,email,'.$id,

'password' => 'nullable|string|min:8|confirmed',

'colegio' => 'required|string|max:100',

'graduacion' => 'required|integer',

'profile\_image' => 'nullable|image|mimes:jpeg,png,jpg,gif|max:2048',

]);

$usuario = Usuarios::findOrFail($id);

if ($request->hasFile('profile\_image')) {

$imageName = time().'.'.$request->profile\_image->extension();

$request->profile\_image->move(public\_path('images'), $imageName);

$usuario->foto = $imageName;

}

$usuario->nombre = $request->name;

$usuario->email = $request->email;

if ($request->password) {

$usuario->password = Hash::make($request->password);

}

$usuario->colegio = $request->colegio;

$usuario->anio\_graduacion = $request->graduacion;

$usuario->save();

return redirect()->route('usuarios.index')->with('success', 'Usuario actualizado exitosamente.');

}

public function destroy($id)

{

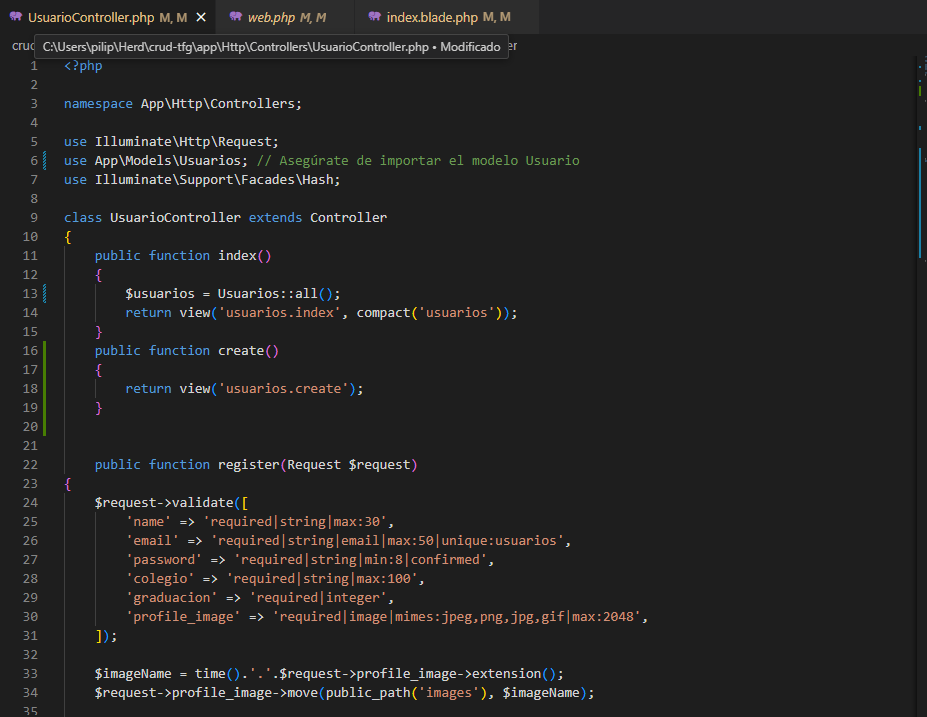
$usuario = Usuarios::findOrFail($id);

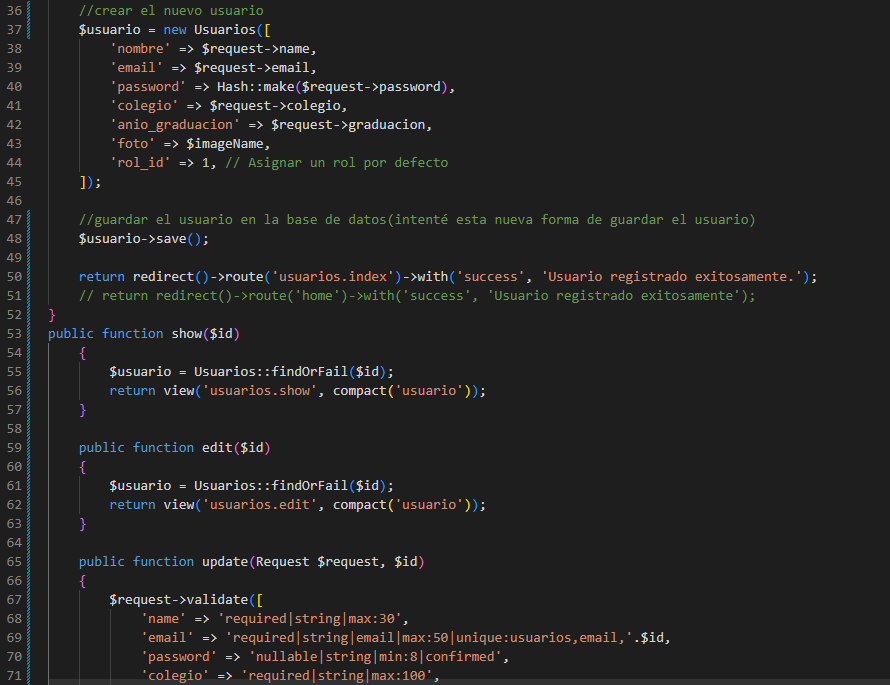
$usuario->delete();

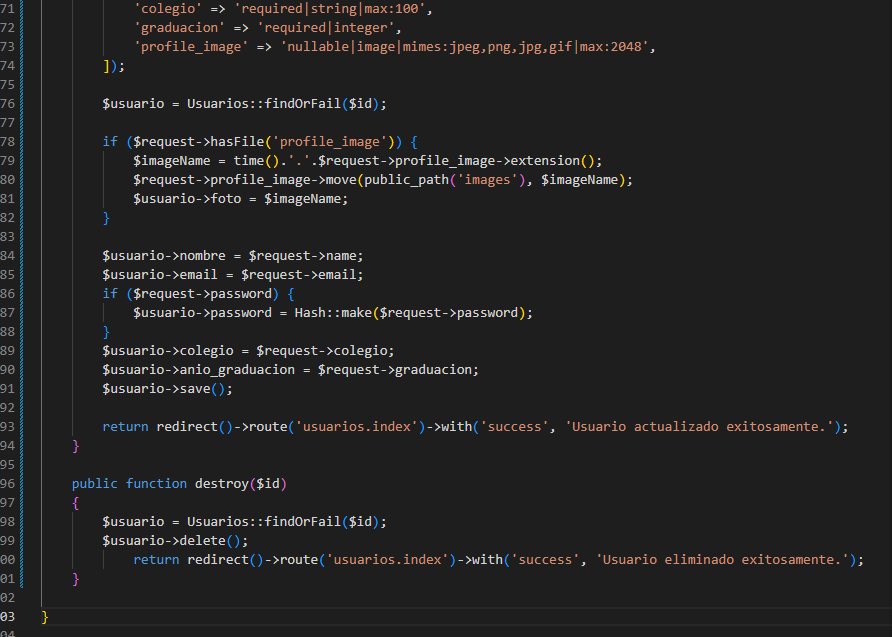
return redirect()->route('usuarios.index')->with('success', 'Usuario eliminado exitosamente.');

}

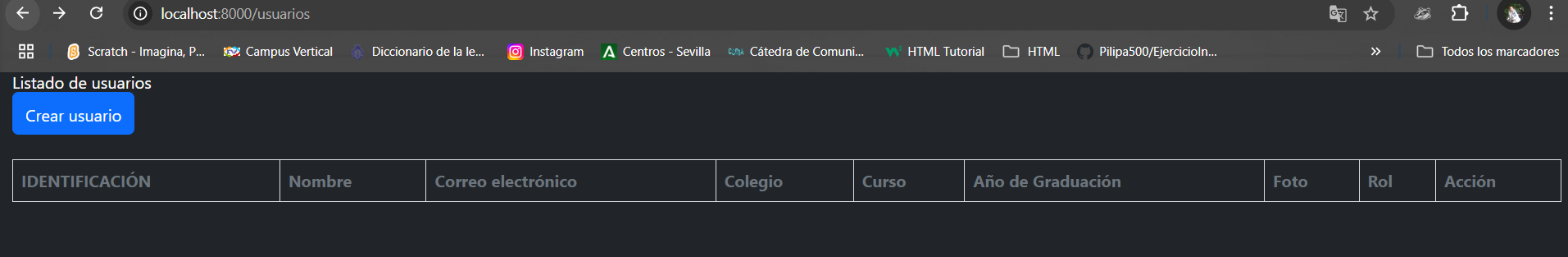
}







Una vez que hice esto si que conseguí “ver” la página que cree para usuarios, tal que así:



Con esto “pretendo” que cada columna se rellenen con los datos de las bbdd de los usuarios que se vayan registrando. Observo que la dimensión de cada columna, no están proporcionadas a los datos que contendrán. Por ejemplo: Identificación, tiene muuucho espacio y nombre, muy poco. Año de graduación es muy largo y foto, muy poco. Voy a intentar, solucionar este tema.

PENDIENTE!!!

Ahora tendré que asegurarme que las vistas correspondientes a (index.blade.php, create.blade.php, edit.blade.php, etc) estén en directorio resources/views/usuarios.

PENDIENTE!!!

He vuelto a repasar la razón por la que no se cargan los datos de registro de la bbdd de usuarios, sino los de la bbdd de users. Osea cuando se rellena el formulario de register.blade.php y se crea correctamente el usuario, esos datos que se guardan lo hacen en MySQL en la tabla users y no en usuarios, que es donde supuestamente debían estar. Es lo que pretendía rellenar en esta página de index.blade.php que se encuentra en la views/usuarios usando los datos de la inserción de los usuarios en el registro.

Voy a comenzar por:

1. Verificar el model Usuarios: asegurándome que el modelo Usuarios esté configurado correctamente para apuntar a la tabla usuarios. Además, asegurarme de que el UsuarioController esté utilizando el modelo Usuarios para las operaciones de creación y almacenamiento.