

Ejercicio del Módulo Formativo 2: Construcción de un Blog con Laravel y Tailwind

Este ejercicio es algo más complejo ya que se utilizarán dependencias y componentes más avanzados en la creación de un blog como si fuera un CMS. El tutorial que se ha seguido comienza en este [enlace](#); se divide en varias partes, por lo que se ha seguido esa misma organización y estructura para su desarrollo:

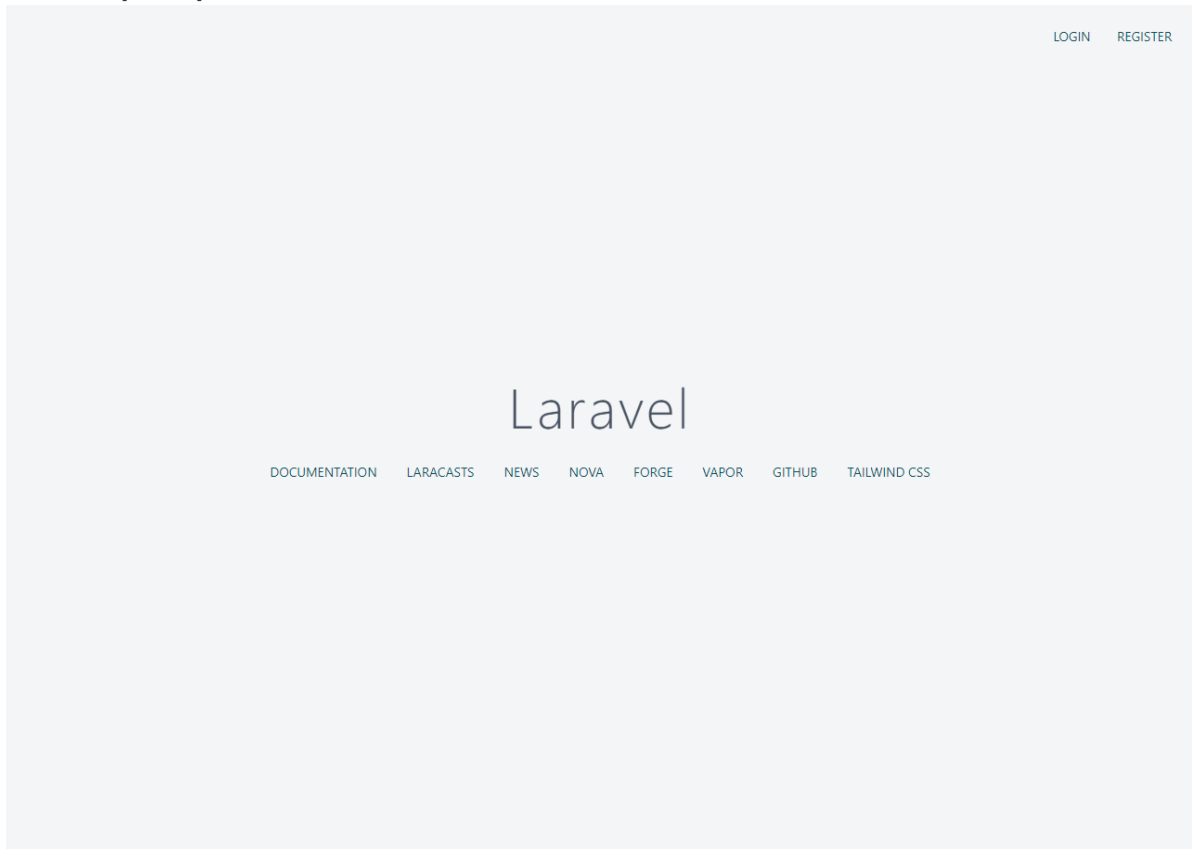
Parte 1: Creación del proyecto e instalación de dependencias

Esta primera parte es sencilla, solo es necesario instalar por consola (de laragon) lo que vamos a usar en este proyecto de blog:

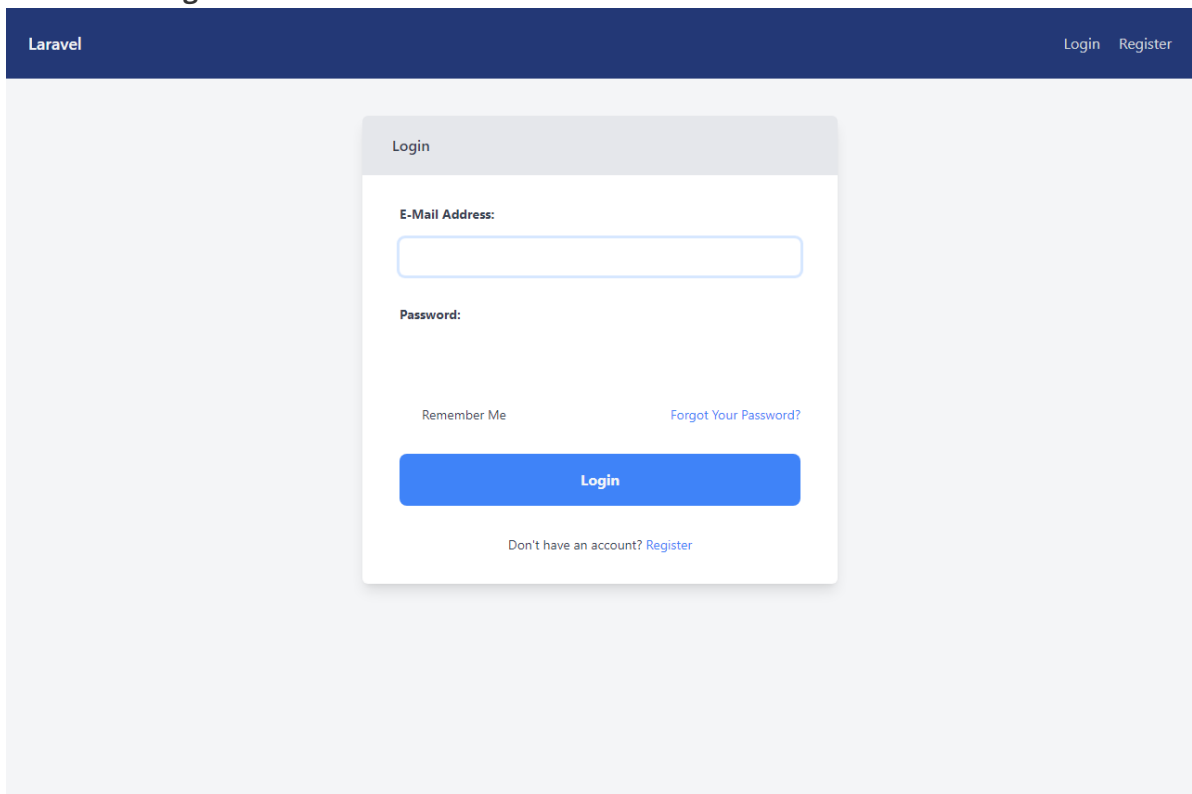
- `composer global require laravel/installer` (si no se tiene ya)
- `laravel new laravel-tailwind`
- `composer require laravel/ui --dev` (sistema de autenticación)
- `composer require laravel/frontend-presets/tailwindcss --dev` (librería de tailwind)
- `php artisan ui tailwindcss --auth` (conexión entre las dos)
- `npm install && npm run dev`
- `npm audit fix --force` (porque da algunos errores de instalación)
- `npm install -g npm@latest` (actualización de la versión que tenía antes, esto da errores por lo que se han modificado los permisos de toda la carpeta laragon, igualmente el comando no funciona)
- `php artisan serve` (inicia el servidor)
- Por último se ha hecho: `npm install laravel-mix@latest --save-dev && npm run dev` (no da errores)

Tras estos pasos, si se va a la url que inicia el servidor se obtiene las siguientes pantallas funcionales:

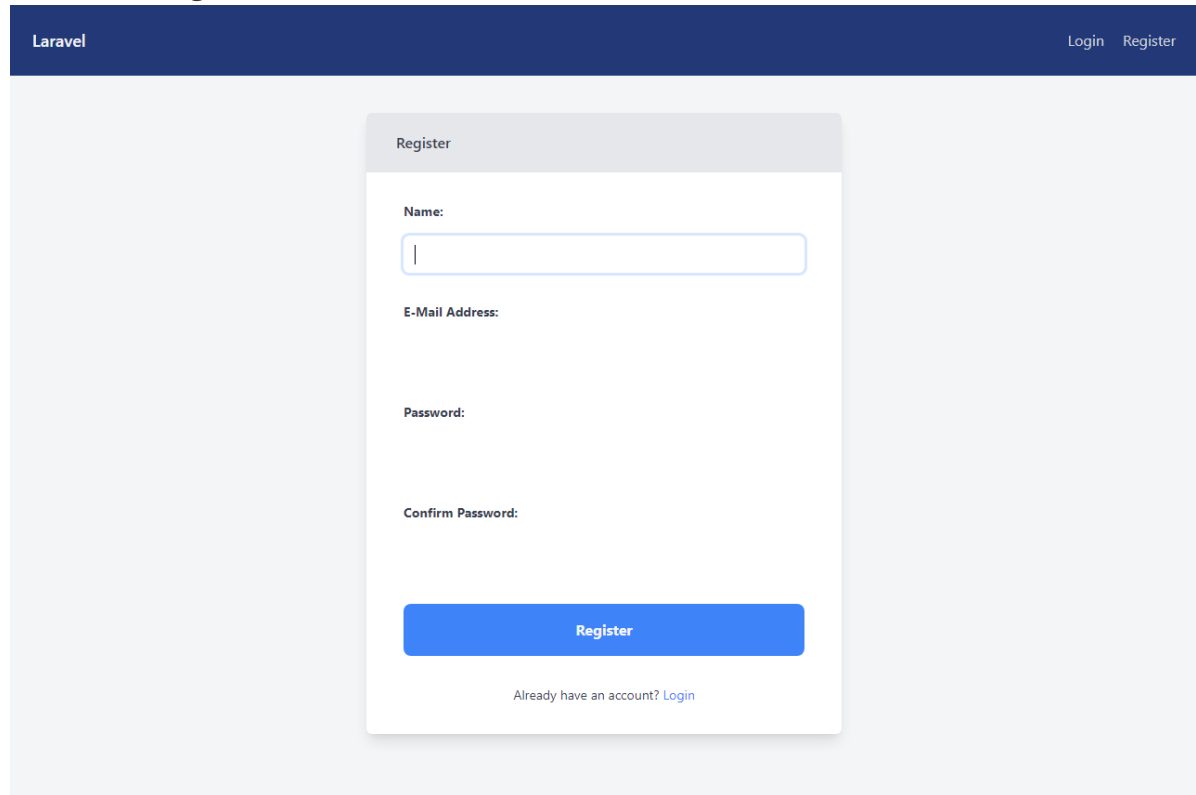
Pantalla principal:



Pantalla de login:



Pantalla de registro:



Parte 2: Base de datos y Modelos

Tras todas las instalaciones, es necesario crear la base de datos en phpmyadmin, la llamamos igual que al proyecto y modificamos el archivo .env para que se produzca la conexión entre proyecto y servidor.

El siguiente paso es la creación de modelos. Como se vio en el ejercicio anterior se pueden crear todos los archivos de golpe pero en esta ocasión lo haremos por partes.

Primero realizamos los comandos que crearán modelos y migraciones de los mismos:

- "php artisan make:model Post -m"
- "php artisan make:model Comment -m"

En un momento editaremos estos archivos, pero primero vamos a modificar el modelo User que viene por defecto en larvel:

- por un lado, la migración (tabla), para incluir columnas en la función up() que digan si el usuario es staff o admin
- por otro lado, el modelo (User.php), para incluir las funciones:
 - isAdmin()
 - isStaff()
 - authorizeRoles()
 - hasAnyRole()

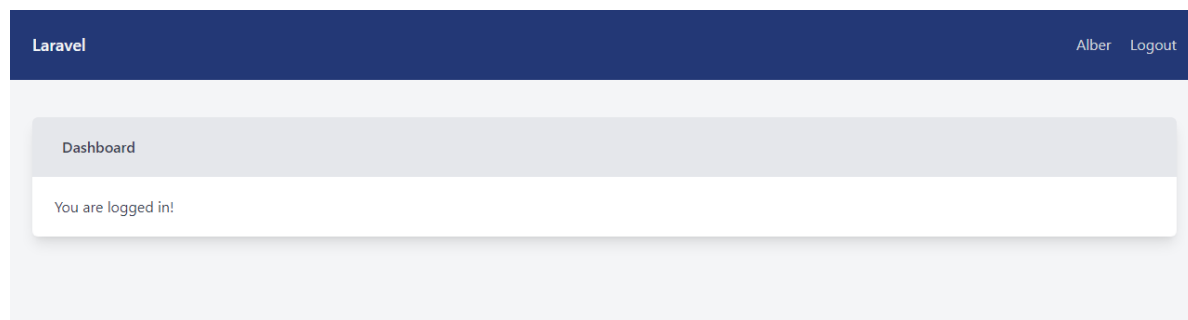
Completado este inciso, editamos los archivos correspondientes de los modelos creados:

- migraciones (tablas) de posts y comments (function up).
- modelos Post.php y Comment.php (columnas y public functions con los métodos de las relaciones entre clases).
- además instalamos una dependencia para que se generen automáticamente las urls de los posts con el comando "composer require cviebrock/eloquent-sluggable".
- hacemos **php artisan migrate** para que todas las tablas se creen en nuestra base de datos.

Por último vamos a crear algunos datos de prueba en nuestras tablas, para ello tenemos que crear las factories y seeders de nuestros modelos con los siguientes comandos y pasos:

- "php artisan make:factory PostFactory -m Post".
- editar PostFactory.php y DatabaseSeeder.php para crear 50 posts.
- "php artisan db:seed" = se puebla la tabla de registros (en realidad se poblarían todas pero solo se ha especificado parámetros para posts y un usuario para Users).

Las pantallas de la aplicación continúan siendo las mismas, pero a mayores ya nos podemos logear con el usuario que se ha creado y tenemos una **pantalla de inicio** que luce así:



Como se puede ver tenemos por defecto la opción de hacer logout en el extremo superior derecho, junto al nombre del perfil que todavía no es funcional. En el otro lado, si clicamos en "Laravel" tenemos la misma pantalla principal de la primera parte, pero al estar logueados en vez de las opciones login/register aparece un único enlace a nuestra Home.

Parte 3: Página de Inicio y Posts



En esta nueva sección se crearán Controladores, Rutas y Plantillas para personalizar nuestra aplicación de blog. Por orden:

- Controlador de los Posts: "php artisan make:controller PostController --resource" (el parámetro final crea un sistema CRUD en esta clase controlador, que se usará más adelante).
- Edición de PostController: se añaden las funciones home() y detail(), que mostrarán la página principal y un post individual respectivamente, e importamos la llamada al modelo Post para consultar a la base de datos (es importante hacer siempre esta llamada para evitar errores).
- Por supuesto estos métodos tendrán sus respectivas vistas en resources, con el formato "blade.php" que ya conocemos: es el sistema de plantillas que utiliza laravel y que contiene tanto html como php. Pero antes de esas páginas modificaremos algún archivo más:
 - Abrimos resources/views/layouts/**app.blade.php** para configurar el estilo de base que tendrá la web, y cambiamos el código por defecto por el que proporciona el tutorial. Podemos ver que ya se ha añadido un link a una hoja de estilos que funcionará en toda la aplicación, y también que hay rutas declaradas a distintas funciones del proyecto.
 - Creamos, dentro de resources/views/ un directorio /posts, y a su vez dentro de él un archivo "home.blade.php". con el contenido dado por el tutorial. Esto mostrará la home page del blog con una cabecera, un acerca de y los últimos 6 posts escritos; y se puede ver que se extiende al principio el layout de antes.
 - En la misma carpeta, creamos un archivo "post.blade.php" y también introducimos el código. En esta página se mostrarán los detalles de un post individual así como los comentarios de los usuarios (también extiende el layout de app).
- Por último necesitamos editar el archivo de rutas "web.php", para que todo esté conectado y funcione correctamente. Sólo hay que tener cuidado en dejar "Auth::routes()" intacto, ya que estp se encarga del login, logout y registro de usuarios. Al final tendremos dos rutas:

- o Una hacia la home page
- o Otra hacia el post individual

Ordenamos "php artisan serve" y podemos observar las siguientes pantallas:

Home:

HOME Alber  

My Laravel Blog

About me



Lorem ipsum dolor sit, amet consectetur adipisicing elit. Voluptate necessitatibus ullam commodi perferendis accusamus sint error sequi, dolore nam, vel praesentium dignissimos nostrum quod fuga corporis asperiores laudantium, possimus veniam! Lorem ipsum dolor sit amet, consectetur adipisicing elit. Consectetur iure cumque qui impedit quod earum dolores nisi nemo totam vero natus aperiam, libero consequuntur nesciunt atque officia exercitationem rerum. Veritatis! Lorem ipsum dolor sit amet consectetur adipisicing elit. Voluptas in hic ratione recusandae nostrum, saepe aliquam alias ipsum? Asperiores rerum numquam officia harum atque, impedit perspiciatis facilis nobis tempora est!

Lasts posts

Non quis molestias reiciendis et. Nostrum qui delectus in excepturi excepturi. Aut aliquam porro molestiae repellat exercitationem aperiam. Qui ut harum repudiandae officia n... Read more	Maxime quas consequuntur qui. Autem sint velit qui laudantium. Sequi voluptatem recusandae alias qui sed maxime cumque labore. Alias dolores doloribus laudantium et. Quas... Read more	Expedita sapiente voluptatem eligendi deserunt eos autem officiis. Sequi est eum aut et corrupti repudiandae. Minus et omnis itaque eum. Quia fugiat sapiente quia dignissimos vero exercitationem qui. Nemo vo... Read more	Eligendi occaecati quasi dignissimos deleniti hic ea. Ut eligendi laborum vero consequatur ut ullam delectus. Eos voluptatum ipsam ratione voluptas autem. Et et iste possimus magni nihil cum mag... Read more
Quae minima perferendis sed autem dolores. Ipsam qui eaque aut enim. Cum aspernatur recusandae culpa vitae a magni et. Veritatis omnis non tempore aut ullam earum qui. Voluptatem numq... Read more			Occaecati debitis necessitatibus expedita. Dolorum nostrum explicabo enim cum ducimus facilis. Quas eum sint dignissimos accusantium eos vitae cumque. Molestiae quo ratione exercitati... Read more

@MyLaravelBlog By Alberto Ramírez

Post (al darle a leer más):

HOME Alber  

My Laravel Blog

Maxime quas consequuntur qui.

Autem sint velit qui laudantium. Sequi voluptatem recusandae alias qui sed maxime cumque labore. Alias dolores doloribus laudantium et. Quasi iste voluptatem placeat ea itaque autem dignissimos ut. Tempora fugiat voluptas itaque aut repellendus voluptates. Maxime et vitae iusto veniam ut facere autem fuga. Assumenda autem vitae consequuntur et maiores labore eaque. Maxime omnis beatae blanditiis assumenda. Officiis dignissimos magni aut delectus et rerum error. Ipsa est vel est. Assumenda aut magni est accusamus. Qui enim ipsa vel provident. Vero ea et ducimus aut. Eveniet harum sint vero dolorem. Perspiciatis itaque eaque et qui beatae. Nihil sed quia reprehenderit est sed. Optio voluptas cupiditate eligendi soluta sit qui voluptas. Dolorem vel illo rerum voluptatem veniam. Omnis magni excepturi corporis dicta eveniet delectus qui recusandae. Ullam id aut distinctio aut. Illum quo tenetur nulla voluptatibus debitis. Dolores nihil ullam qui dolor. Est corporis ad nisi quos qui laudantium perferendis. Vel ea ut est quaerat. Iusto voluptatibus animi molestias alias est rem inventore impedit. Mollitia esse unde eum quia nisi perferendis autem. Non dignissimos et est vel enim tempora. Cupiditate dolor necessitatibus in. Ullam impedit porro culpa neque nobis. Quia ducimus soluta in minima possimus. Minima voluptas est veniam est corrupti earum omnis. Ut delectus est ducimus deserunt molestias. Neque et eligendi maxime aut excepturi vero voluptatem voluptatem. Non ut exercitationem recusandae adipisci. Delectus vero odio voluptate et minus optio molestiae. Voluptatem quos mollitia est. Voluptatem molestiae esse quos voluptate aut. Sunt laborum explicabo perferendis quidem et sint consequatur qui. Adipisci sed impedit ex qui ipsum officia. Voluptatum quisquam iusto commodi harum. Quia sed quo facilis aperiam. Et sit sint aut aliquid est molestiae consequuntur. Minus debitis cupiditate perferendis ut omnis quibusdam. Et perferendis ab provident vel totam assumenda ipsam. Dolor quidem est et quaerat nobis aliquid. Autem aut iste porro. Quasi voluptas odit fugit dolorem perspiciatis. Commodi voluptatem quam placeat nulla. Mollitia aperiam doloribus consequatur beatae ut. Consectetur reprehenderit saepe natus placeat. Ullam assumenda eveniet odit cumque nihil. Et aut tempora itaque exercitationem. Maiores dolores rem fugit officia odit dolor distinctio. Voluptatem sequi id labore ea quae. Deserunt facilis optio eum molestiae praesentium asperiores. Laboriosam corporis et sed consequuntur. Eos nisi laborum qui voluptatibus dolore ab deleniti. Nulla incidunt exercitationem nam. Laudantium ut architecto est. Odit expedita aliquam repellat doloremque. Corrupti ipsam voluptate veritatis quasi ea. Minima at est quos sequi tempore sunt.

Comments

@MyLaravelBlog By Alberto Ramírez

Parte 4: Formularios y Validación de datos

Nota: se ha cambiado "public const HOME = '/';" en app/Providers/RouteServiceProvider.php para que cuando un usuario se loguee entre directamente en el home del blog creado antes (sino iría al home de fuera, el dashboard).

Hecho esto pasamos a la creación de comentarios, para lo que necesitamos una clase custom request que se extiende de la clase de formulario y sirve para validar la información enviada por un user. Entonces:

- Hacemos el comando "php artisan make:request CommentRequest", que crea en app/Http/Requests/ el archivo CommentRequest.php
- Abrimos ese archivo y sustituimos el código por el proporcionado por el tutorial. Se han añadido métodos para comprobar la autorización del usuario e indicar las reglas de los parámetros, y también para personalizar los comentarios.
- Creamos el controlador de los comentarios: "php artisan make:controller CommentController", y editamos el código interno, que básicamente contiene una función de guardar los datos.
- Nos movemos a resources/views/posts/post.blade.php (vista de un post) y añadimos el bloque de código que muestra el formulario para crear comentarios (siempre que un usuario esté logueado).
- Finalmente añadimos a web.php la ruta necesaria para generar los comentarios (mediante la conexión con el controlador y el método store).

Ahora la **página de Post** se muestra así:

HOME Alber    

My Laravel Blog

Expedita sapiente voluptatem eligendi deserunt eos autem officiis.

Sequi est eum aut et corrupti repudiandae. Minus et omnis itaque eum. Quia fugiat sapiente quia dignissimos vero exercitationem qui. Nemo voluptate et dolores culpa quas odit ut doloremque. Et est mollitia repellat. Mollitia tenetur corporis ex ipsam. Est iste voluptatibus quod asperiores voluptatem. Aut eos dolor qui incidunt sed. Non et in quo et cum. Totam dicta temporibus et ducimus ipsam sit. Similique voluptates labore odio doloremque rem. Iusto iure sit quam a et praesentium voluptate. Aliquid eum libero aspernatur et necessitatibus quis sed. Excepturi aut repellendus et praesentium officia qui in doloremque. Consectetur quisquam voluptas animi est. Et et nam sit numquam. Doloremque id ea iste laboriosam dolore voluptas qui. Laboriosam non libero quos velit sunt. Voluptatum fugit aut deserunt beatae cum voluptatibus. Laudantium omnis ex numquam ea quidem recusandae deserunt. Cum possimus quod inventore ut inventore. Quod et cupiditate cumque et voluptas aut. Sed nihil sequi quos odit. Laudantium debitis omnis porro. Sunt voluptas distinctio error a. Quia perspicatis laborum officiis voluptas. Commodi eligendi in ut est dicta eveniet. Modi nulla occaecati a consequuntur molestiae deleniti. Eius facere eum repellendus vel possimus dicta quia. Quo doloribus pariatur aut non sit. Delectus minus occaecati excepturi quis sit. Architecto similique aperiam amet ea repellat aut. Quis expedita facilis maxime ullam. Voluptatem architecto omnis enim voluptas rerum eum. Qui sit magni inventore aut est maxime. Dolorem esse est adipisci voluptatibus quis doloribus velit. Quisquam ea quo repudiandae itaque facilis porro eum. Amet doloremque enim fuga quis ut facilis vel enim. Sint fugit quos non cupiditate aut. Voluptate molestiae facere eaque est eos maxime voluptatem. Nihil ut et et asperiores. Dignissimos consectetur amet distinctio quas ut corrupti sit. Velit nostrum sit modi enim sunt nostrum. Nobis tempore autem culpa earum enim dolorem ducimus. Accusamus consectetur possimus voluptates sapiente commodi. Itaque molestias sit est. Dolor a quis est aut est quo in. Rerum consectetur perferendis exercitationem fugiat. Sit sequi nobis non itaque ut consequatur voluptatum. Iusto exercitationem consequatur possimus et sunt reiciendis dignissimos. Reiciendis repellat omnis consectetur officiis omnis blanditiis tempore. Dolores ut saepe tenetur et. Perferendis repudiandae dolores eum. Asperiores assumenda dolorum maxime fugit facere sint. Repudiandae perferendis accusamus incidunt nostrum dolore. Est ut libero harum dolor quos. Veniam aut ut dolores odit amet. Voluptatibus velit aut suscipit laborum. Doloribus sit quia excepturi excepturi ut rerum consequuntur aliquid. Explicabo quae voluptates quas asperiores minima. Qui nobis voluptatem occaecati vitae qui consectetur quae. Voluptatum voluptas qui enim quibusdam ut consequatur. Error tempora cumque ea ut architecto et. Explicabo velit error beatae ab. Ut sed ipsa dolores quos. Ut ipsum quia quis et.

Comments

Write your comment here

SEND

@MyLaravelBlog By Alberto Ramírez

Nota: en este momento del proyecto se ha cambiado el nombre y las credenciales del usuario, por lo que se han vuelto a migrar y poblar las tablas.



Parte 5: CRUD para administrar los Posts

CRUD son las siglas en inglés de Crear, Leer, Actualizar y Borrar, las funciones principales que puede tener un modelo en Laravel. En este caso vamos a completar este sistema para la clase Post.

- En primer lugar, añadimos en web.php las rutas que necesitará el framework para cada función. Se podría elaborar una ruta por cada elemento del CRUD pero en este caso aprovechamos el método resource que nos permite establecerlas en una sola línea.

- Si se quieren ver todas las rutas desglosadas se utiliza el comando "php artisan route:list"
- Después se completa el código de las funciones en el PostController.php, que nos proporciona el tutorial, y a su vez se crean las vistas relacionadas con cada una de ellas (y también se completan). Es decir:
 - función index() -> muestra el listado de posts -> vista resources/views/posts/list.blade.php
 - función create() -> crea el post -> vista resources/views/posts/create.blade.php
 - función store() -> guarda el post -> no tiene vista propia
 - función edit() -> modifica el post -> vista resources/views/posts/edit.blade.php
 - función update() -> actualiza el post -> no tiene vista propia
 - función destroy() -> borra el post -> no tiene vista propia
- Además, se ha añadido en resources/views/layouts/app.blade.php un nuevo enlace en la barra de navegación, de forma que el usuario podrá ir al panel de administración de posts desde el menú.
- También se ha hecho un nuevo Request: "php artisan make:request PostRequest" para la creación de Posts.
- Ahora desde la página de listado de posts podemos acceder a la creación de uno nuevo, a la opción de editar uno ya hecho, y también a borrar cualquiera (previo aviso por pantalla de si estamos seguros de hacerlo).

Lista de Posts:

HOME Noe  



My Laravel Blog



List Post

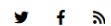
Create new post

Title	Creation	Author	Status	Actions
Optio magnam ipsa maxime nihil.	25 May, 2022	Noe	Published	Edit Delete
Deleniti rerum est dicta excepturi nemo.	25 May, 2022	Noe	Published	Edit Delete
Nesciunt voluptas natus et.	25 May, 2022	Noe	Published	Edit Delete
Odit omnis repellat doloribus earum sed.	25 May, 2022	Noe	Published	Edit Delete
Nam voluptatibus et dolores veritatis autem.	25 May, 2022	Noe	Published	Edit Delete
Adipisci quia aut nobis et et qui.	25 May, 2022	Noe	Published	Edit Delete
Ducimus et tenetur harum maiores exercitationem voluptatem.	25 May, 2022	Noe	Published	Edit Delete
Incidunt molestiae eius ipsum sit rerum aliquam.	25 May, 2022	Noe	Published	Edit Delete
Soluta illo voluptatem ullam illo mollitia.	25 May, 2022	Noe	Published	Edit Delete
Earum sit quia nihil.	25 May, 2022	Noe	Published	Edit Delete
Debitis id aut commodi ut inventore.	25 May, 2022	Noe	Published	Edit Delete
Earum magnam veritatis magni magni.	25 May, 2022	Noe	Published	Edit Delete
Qui soluta ut voluptatem non.	25 May, 2022	Noe	Published	Edit Delete
Repudiandae culpa recusandae eum alias.	25 May, 2022	Noe	Published	Edit Delete
Unde commodi sunt voluptas et.	25 May, 2022	Noe	Published	Edit Delete
Commodi et sit in commodi vel occaecati.	25 May, 2022	Noe	Published	Edit Delete
Vel sequi quod quia eius.	25 May, 2022	Noe	Published	Edit Delete
Sit dolorum voluptas et in doloreque qui eaque.	25 May, 2022	Noe	Published	Edit Delete
Inventore excepturi aut alias iure.	25 May, 2022	Noe	Published	Edit Delete
Explicabo debitis ut rerum sed aut.	25 May, 2022	Noe	Published	Edit Delete
Voluptas debitis quibusdam reiciendis ipsa dolorem cum eos doloreque.	25 May, 2022	Noe	Published	Edit Delete
Eligendi ab aut natus.	25 May, 2022	Noe	Published	Edit Delete
Culpa perspiciatis nesciunt quo aliquid quaerat laborum voluptatem.	25 May, 2022	Noe	Published	Edit Delete
Beatae dicta velit provident fuga consequatur et.	25 May, 2022	Noe	Published	Edit Delete
Aliquid quibusdam commodi veniam exercitationem.	25 May, 2022	Noe	Published	Edit Delete
Magni expedita incidunt porro consequatur quas excepturi sed.	25 May, 2022	Noe	Published	Edit Delete
Omnis et ut quasi provident consequatur quibusdam est.	25 May, 2022	Noe	Published	Edit Delete
Est suscipit autem eos inventore.	25 May, 2022	Noe	Published	Edit Delete
Voluptatem iste voluptate qui dolor.	25 May, 2022	Noe	Published	Edit Delete
Nihil similique occaecati sed quae animi.	25 May, 2022	Noe	Published	Edit Delete
Fuga odio voluptatum eveniet nulla reiciendis.	25 May, 2022	Noe	Published	Edit Delete
Ut quia totam et possimus.	25 May, 2022	Noe	Published	Edit Delete
Quae culpa cupiditate autem dolores voluptatum saepe.	25 May, 2022	Noe	Published	Edit Delete
Quae repellendus deserunt vel.	25 May, 2022	Noe	Published	Edit Delete
Quis fugiat repudiandae iste dolorem velit dolorem expedita.	25 May, 2022	Noe	Published	Edit Delete
Debitis sunt voluptas dolores ratione quos dicta est.	25 May, 2022	Noe	Published	Edit Delete
Magni facere architecto iste ratione ut corporis.	25 May, 2022	Noe	Published	Edit Delete
Nesciunt ut modi illum numquam unde.	25 May, 2022	Noe	Published	Edit Delete
Animi aut quasi sint.	25 May, 2022	Noe	Published	Edit Delete
Officiis omnis ut ea aperiam enim.	25 May, 2022	Noe	Published	Edit Delete
Harum ipsum cumque nihil natus tempora inventore qui.	25 May, 2022	Noe	Published	Edit Delete
Qui culpa vel iste magni sapiente fugit iste.	25 May, 2022	Noe	Published	Edit Delete
Eaque saepe dicta est quia reprehenderit et voluptates.	25 May, 2022	Noe	Published	Edit Delete
Aut atque aliquid placeat deleniti ut.	25 May, 2022	Noe	Published	Edit Delete
Expedita repellendus quia velit nesciunt architecto cupiditate.	25 May, 2022	Noe	Published	Edit Delete
Dolorem illum facere assumenda eum velit quasi quas est.	25 May, 2022	Noe	Published	Edit Delete
Vero nobis est repudiandae porro veniam dignissimos qui dolor.	25 May, 2022	Noe	Published	Edit Delete
Aliquam expedita ipsa id velit doloribus.	25 May, 2022	Noe	Published	Edit Delete
Dicta quasi quibusdam sunt ut odio ea laudantium quidem.	25 May, 2022	Noe	Published	Edit Delete
Ratione expedita neque totam et qui sequi eum.	25 May, 2022	Noe	Published	Edit Delete

@MyLaravelBlog By Noe Fdez

Crear nuevo Post:

HOME Noe  



My Laravel Blog

Create Post

Write the title of the post

Write your post here

☐ Is draft?

SEND

@MyLaravelBlog By Noe Fdez

My Laravel Blog

Edit Post

Earum magnam veritatis magni magni.

Qui hic non tenetur dicta ratione illo molestiae.
Sequi in hic qui quod et unde. Numquam aut

☐ Is draft?

SEND

@MyLaravelBlog By Noe Fdez

Parte 6: Test unitarios

Los test unitarios se utilizan en producción para comprobar que la web funciona correctamente. Básicamente son scripts que se generan para verificar nuestro código. Laravel trae por defecto el soporte de PHPUnit, mediante la carpeta "tests/" del árbol de directorios, que se divide en Unit/ (tests específicos de frgamentos de código) y en Feature/ (tests más complejos).

Es necesario hacer un par de cambios de configuración antes de testear la web:

- Duplicar el archivo .env y renombrarlo como .env.testing
- Crear una nueva BB.DD de prueba y vincularla al nuevo .env.testing (en mi caso es laravel-tailwind-test)
- Modificar el archivo database/factories/UserFactory.php, en concreto el array de la función definition(), para que los usuarios que se creen tengan permisos de admin y staff

Hecho esto podemos hacer el el primer test con el comando: "php artisan make:test CommentTest", lo que creará el archivo Comment.Test dentro de tests/Feature/, y debemos reemplazar su código por el que nos da el tutorial. En el segundo test haríamos lo mismo: "php artisan make:test PostTest" y editar el código del archivo Post.Test con el que nos dan.

Aquí nos encontramos con un error de código donde el framework no reconoce la variable \$user declarada anteriormente en estos archivos, esto se debe a que el Test esperaba encontrar una llamada específica desde el modelo de User que actualmente no está declarada. Para arreglar este error y que desaparezcan los fallos he incluido "/* @var User */" en las funciones necesarias, antes de cada declaración de variable \$user o similares.

Nota: previo al comando de test debemos migrar las tablas a la base de prueba con "php artisan migrate:fresh --env=testing".

Una vez solucionado podemos hacer el comando "php artisan test" y si todo está correcto la prueba se pasaría sin problema. En mi caso todos los archivos pasan el test a excepción de una sola función de PostTest, que devuelve un error aunque el archivo en sí no presenta fallos visibles.

```
PASS Tests\Unit\ExampleTest
✓ example

PASS Tests\Feature\CommentTest
✓ create comment

PASS Tests\Feature\ExampleTest
✓ example

FAIL Tests\Feature\PostTest
✓ main page
✓ create post
✗ edit post
✓ delete post
✓ edit post staff member
✓ delete post staff member

---

• Tests\Feature\PostTest > edit post
Failed asserting that null matches expected 'Post has been updated sucessfully'.

at C:\laragon\www\laravel-tailwind\tests\Feature\PostTest.php:99
```

Parte 7: Personalización propia de la web (tailwind)

Como al principio de todo se han descargado las dependencias de tailwind, cuando abrimos resources/css/app.css no encontramos con el archivo iniciado por 3 líneas que proporcionan el estilo actual:

- @import "tailwindcss/base";
- @import "tailwindcss/components";
- @import "tailwindcss/utilities";

A partir de ellas y de los conocimientos de hojas de estilos podemos cambiar el layout de nuestras páginas y de sus elementos a nuestro gusto.

Repositorio del proyecto

El repositorio que contiene todo el proyecto es accesible en [Github](#), en caso de querer revisar o comprobar el código.

Noelia Figueiras Fernández | 39488208-Z