

# Proyecto de laravel con debian

---

[Link de repositorio remoto](#)

## Episodio 5

Explicacion de rutas

Aplicar negrita a la vista de welcome

Funciones que se pueden poner dentro del return de las rutas

[Codigo](#)

## Episodio 6

Cambiar welcome page

Agregar CSS

Crear alertas con JS

[Codigo](#)

## Episodio 7

Agregar mas CSS

Primeros blog Posts

[Codigo](#)

## Episodio 8

Agregar las rutas y archivos para obtener el contenido de los post de manera dinamica

[Codigo](#)

## Episodio 9

Atrapar los errores posibles que puede tener una ruta, aprender de expresiones regulares

[Codigo](#)

## Episodio 10

Utilizar cache para economizar recursos de peticiones al servidor

[Codigo](#)

## Episodio 11

Utilizar clases para mejorar el codigo, de manera que encontrar los posts sea mas facil y no ensucie tanto el codigo de rutas

Utilizar el FileSystem para obtener de manera dinamica el numero y el contenido de los posts existentes

Introduccion a la metadata

[Codigo](#)

## Episodio 12

Utilizar librerias externas para manejar metadata (Spazie YamlFrontMatter)

Utilizar collections para iterar y mapear de mejor manera

[Codigo](#)

## Episodio 13

Sortear por fecha para acomodar los post

Cachear el resultado de los post para no hacer futuras peticiones al servidor de informacion constante

[Codigo](#)

## Episodio 14

Introduccion a Blade

Diferentes sintaxis para mejorar el rendimiento del codigo

[Codigo](#)

## Episodio 15

Introduccion e implementacion de layout files para facilitar el reciclaje de codigo en los views

[Codigo](#)

## Episodio 16

Implementar un findOrFail para atrapar el error cuando una ruta de post no existe

[Codigo](#)

## Episodio 17

Introduccion a la conexion a base de datos

Se cambian parametros en el archivo .env para poder conectar a nuestra instancia de MySql

[Codigo](#)

## Episodio 18

Introduccion a las migraciones, estructura de la base de datos y rollbacks

Tip: Laravel bloquea las migraciones para limpiar la base de datos si el ambiente es de produccion `php artisan migrate:fresh`

[Codigo](#)

## Episodio 19

Introduccion a los modelos, creacion de usuarios e interaccion con tinkler

Comandos como:

```
$user-> new User;  
$user -> save();  
$user -> Find::all();
```

[Codigo](#)

## Episodio 20

Creacion de modelos y migraciones utilizando PHP artisan

```
php artisan make:migration create_posts_table
```

Crear modelo

```
php artisan make:model Post
```

[Codigo](#)

## Episodio 21

Consultas a la base de datos para actualizar el primer post

```
$post = App\Models\Post::first();  
  
$post->body = new body;  
$post->save();
```

[Codigo](#)

## Episodio 22

Crear post con el siguiente codigo:

```
Post::create(['title'=>'My second post','excerpt'=>'Lorem Ipsum',  
'body'=>'Lorem ipsum Lorem ipsum Lorem Ipsum']);
```

Agregar el \$fillable en el modelo de Post para que el codigo anterior corra de manera correcta

[Codigo](#)

## Episodio 23

Mejorar la busqueda del post mediante el cambio de enviar un **Post** **\$post** en lugar de un **\$id** en el request del Endpoint

```
Route::get('/posts/{post}', function (Post $post) {
```

```
// $post = Post::findOrFail($id);

return view('post',['post' => $post]);

});
```

[Codigo](#)

## Episodio 24

Crear migracion y modelo para agregar categorias a los blog posts

```
php artisan make:model Category -m
```

Se crea una relacion entre cada post y categoria mediante una llave foranea llamada **category\_id** en el modelo de *post* [Codigo](#)

## Episodio 25

Crear la relacion Post - Categoria pero desde el modelo de *Category* mediante este codigo:

```
public function posts(){

    return $this->hasMany(Post::class);

}
```

Para poder consultar cada categoria y traer todos los posts que pertenecen

[Codigo](#)

## Episodio 26

Solucionar el llamado 'N+1 problem'

Instalar la libreria 'Clockwork'

```
composer require itsgoingd/clockwork
```

Tambien se puede agregar la extension del navegador de manera opcional

[Codigo](#)

## Episodio 27

Agregar creador de cada post mediante una relacion similar a cada post con su categoria

Crear un seeder para poblar la base de datos de manera automatica

```
php artisan db:seed
```

[Codigo](#)

## Episodio 28

Utilizar factories para crear contenido en la base de datos de manera aleatoria y de forma rapida

```
php artisan make:factory PostFactory  
php artisan make:factory CategoryFactory
```

[Codigo](#)

## Episodio 29

Ver todos los posts relacionados al autor similar al proceso realizado con las categorias

[Codigo](#)

## Episodio 30

Solucionar el N+1 previamente detallado pero en las categorias de cada post mediante el siguiente codigo en Post

```
protected $with = ['category','author'];
```

[Codigo](#)

## Episodio 31

Convertir el html y CSS a blade utilizando tambien el complemento de Tailwind, ademas de utilizar componentes para reutilizar codigo en las vistas.

[Codigo](#)

## Episodio 32

Inyectar la informacion de los posts de la pagina principal de manera dinamica

[Codigo](#)

## Episodio 33

Similar al episodio anterior, cargar los post en su pagina individual de manera dinamica

[Codigo](#)

## Episodio 34

Hacer un dropdown para mostrar las categorias en el homepage utilizando JavaScript

[Codigo](#)

## Episodio 35

Extraer el dropdown como un componente para reciclar ese codigo

[Codigo](#)

## Episodio 36

Limpieza y mejoras de codigo

[Codigo](#)

## Episodio 37

Funcion de busqueda

[Codigo](#)

## Episodio 38

Mejorar funcion de busqueda de posts

[Codigo](#)

## Episodio 39

Mejorar el query similar a la busqueda de posts pero ahora por categoria

[Codigo](#)

## Episodio 40

Extraer el codigo del dropdown de categorias en su propio componente para mejorar aun mas y limpiar el codigo de las vistas

[Codigo](#)

## Episodio 41

Implementar un filtro por autor

[Codigo](#)

## Episodio 42

Unir el filtro para categoria y busqueda de texto

[Codigo](#)

## Episodio 43

Arreglar un pequeno bug en las busquedas

[Codigo](#)

## Episodio 44

Paginar la informacion que se muestra en las vistas cuando son muchos posts

Hacer codigo de vendors publico para poder actualizar o cambiar vistas

[Codigo](#)

## Episodio 45

Implementar una pagina de registro de usuario, autenticacion, autorizacion

[Codigo](#)

## Episodio 46

Hashear las contraseñas para guardarlas de manera segura en la base de datos

[Codigo](#)

## Episodio 47

Implementar validacion de errores y mensajes de error

[Codigo](#)

## Episodio 48

Mostrar un mensaje de exito flash cuando el registro es completado de manera exitosa

[Codigo](#)

## Episodio 49

Implementar la funcion de iniciar y cerrar sesion

Introduccion al middleware

[Codigo](#)

## Episodio 50

Crear la vista de inicio de sesion y manejar los posibles errores de validacion

[Codigo](#)

## Episodio 51

Demo de laravel breeze, una libreria de autenticacion que genera las vistas y toda la logica relacionada con autenticacion y autorizacion de usuarios

[Laravel breeze](#)

```
php artisan breeze:install
```

```
php artisan migrate  
npm install  
npm run dev
```