

Routing

¿Qué pasa exactamente cuando hacemos una petición?

Antes de profundizar en los aspectos prácticos de laravel, veamos primero qué sucede exactamente cuando llega una solicitud.

Imaginemos que administra un restaurante de entrega de pizzas.

Cuando una persona visita URL de la página del restaurante antes de que se muestren las páginas, primero llegamos al servidor y cargamos su laravel. Inmediatamente, su aplicación se da cuenta de que, esta es una solicitud para la página de inicio.

Ahora en laravel, es realmente fácil asociar cualquier URI con una respuesta

Para notar en este caso si el usuario realiza una solicitud para la página de inicio, cargaremos un controlador y un controlador es la siguiente pieza del rompecabezas.

El controlador recibe una solicitud y proporciona una respuesta, por lo que en este caso, la solicitud es cargar la página de inicio de Larry's Pizza.

Cómo el controlador genera esa respuesta puede tomar muchas formas y hablaremos un poco sobre esto, pero por ahora digamos que el controlador debe delegar para obtener la información necesaria de su base de datos puede ser todas las pizzas, todos los especiales, todos los temas que podemos utilizar, lo que se conoce como un modelo elocuente. Para esto ahora, no solo el modelo proporciona una buena API para realizar cualquier cantidad de consultas SQL en su base de datos, sino que también es una ubicación para almacenar cualquier lógica de dominio o lógica de negocios específica.

Una vez que el controlador ha delegado en el modelo, el siguiente paso es cargar la vista.

Son las hojas del usuario en última instancia. Es la parte HTML de su código base, define la estructura, carga un archivo CSS, las referencias de JavaScript ahora lo más importante si recibió los datos del controlador.

Ahora lo más importante es que la vista recibe los datos del controlador y luego los procesa para el usuario.

Resumen

El controlador delega y carga toda la información necesaria para proporcionar una respuesta que recibe los datos del controlador y luego genera la estructura HTML para el usuario.

Setup a Database Connection

En la carpeta del proyecto se encuentra el archivo .env Aquí es donde almacena todas las claves y valores de configuración importantes, lo primero que debe entender son todas estas claves de base de datos, todos ellos serán referenciados dentro del archivo de configuración correspondiente.

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=blog_alexamith
DB_USERNAME=root
DB_PASSWORD=
```

Cambio los valores a los cuales se adapte con lo que voy a trabajar.

Paso 1

Voy a la carpeta config

Paso 2

Abro el archivo database.php

Paso 3

Elijo y configuro la base de datos que vaya a usar

- Conexión de graves: estamos usando nuestra reconexión usando un adaptador MySQL o sqlite o postgres. Cuando este caso va a leer esa clave del archivo de entorno y si no existe, cambiaremos de forma predeterminada a mySQL.

```
/*
|-----
| Default Database Connection Name
|-----
|
| Here you may specify which of the database connections below you
| to use as your default connection for all database work. Of course
| you may use many connections at once using the Database library.
|
*/

'default' => env('DB_CONNECTION', 'mysql'),

/*
```

- Aquí están todas las conexiones entre las que puede elegir, así que imaginemos que desea usar sqlite en su lugar, de acuerdo, volvería a su archivo de entorno y cambiaría la conexión, así que ahora está usando sqlite.

Crear la base de datos mySql

En la terminal lanzo sigo los siguientes pasos.

- Me conecto a mysql con el comando

```
sudo mysql -u root
```

- Creo la bd con el nombre que coloqué en el archivo .env

```
create database laravel;
```

- Me conecto a la bd que **laravel**

```
use laravel
```

- Creo una tabla por ejemplo así.

```
CREATE TABLE persona (nombre VARCHAR(30), edad INT);
```

- Comprobar la tabla creada.

```
DESCRIBE persona;
```

- Añadir algunos registros.

```
INSERT INTO persona VALUES ('Jose Sanchez', 22);
```

- Comprobar los datos.

```
SELECT * FROM persona;
```