Programación Web Día

LUIS EDIN IBARGUEN MOSQUERA

Taller N°. 3

CONSULTAS

ING. CARLOS ALBERTO LONDOÑO LOAIZA

CORPORACIÓN DE ESTUDIOS

TECNOLÓGICOS

DEL NORTE DEL VALLE

PROGRAMA

CARTAGO

2016

1. Buscar información sobre los siguientes términos
   1. SoftDelete
   2. Seeder
   3. Slug
   4. Middleware

DESARROLLO

**a. SoftDelete**

R/

Además de eliminar los registros de su base de datos, Eloquent también puede borrar suavemente los modelos. Cuando los modelos son borrados suavemente, no se eliminan de su base de datos. En su lugar, un atributo deleted\_at se establece en el modelo y se inserta en la base de datos. Si un modelo tiene un valor deleted\_at no nulo, el modelo se ha suprimido suavemente. Para habilitar supresiones suaves para un modelo, utilice el atributo Illuminate \ Database \ Eloquent \ SoftDeletes en el modelo y agregue la columna deleted\_at a su propiedad $dates:

<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

use Illuminate\Database\Eloquent\SoftDeletes;

class Flight extends Model

{

use SoftDeletes;

/\*\*

\* The attributes that should be mutated to dates.

\*

\* @var array

\*/

protected $dates = ['deleted\_at'];

}

Por supuesto, debe agregar la columna deleted\_at a la tabla de la base de datos. El generador de esquema Laravel contiene un método auxiliar para crear esta columna:

Schema::table('flights', function ($table) {

$table->softDeletes();

});

Ahora, cuando llama al método  delete en el modelo, la columna la deleted\_at se establecerá en la fecha y hora actuales. Y, al consultar un modelo que utiliza borrados suaves, los modelos borrados suavemente se excluirán automáticamente de todos los resultados de la consulta.

Para determinar si una instancia de modelo dado se ha suprimido suavemente, utilice el método trashed:

if ($flight->trashed()) {

//

}

**b. Seeder.**

**R/.**

Conoce los seeders, para alimentar una base de datos y crear datos de prueba o configurar el estado inicial de las tablas para un proyecto.

Seguimos con asuntos relacionados con las bases de datos en el Manual de Laravel 5. En esta ocasión vamos a hablar de los "seeders". Seed es "semilla" en inglés y "seeders" serían algo así como "sembradores", aunque prefiero traducir como "alimentador", ya que sería como yo llamaría a estos sistemas corrientemente.

Los seeders no son más que componentes del framework Laravel que sirven para inicializar las tablas con datos. Así como las migraciones nos permiten especificar el esquema de la base de datos, los seeders nos permiten también por medio de código alimentar las tablas con datos.

Su uso, como decimos puede ser para:

1. Crear datos de prueba con los que trabajar durante el desarrollo de la aplicación
2. Configurar el estado de las tablas que necesita nuestra aplicación para comenzar a trabajar

Para aclarar este segundo caso piensa en el ejemplo de los artículos. Cuando subes un artículo puede tener varios estados: "borrador", "publicado", "borrado", etc. Puede que esos estados estén definidos en una tabla, así luego puedes crear otros estados si lo necesitas, como "obsoleto". Pero quizás no vas a hacer en el backoffice para administrar los cambios sobre esa tabla y prefieres dejar una configuración inicial de los estados que prevees vas a necesitar para que la aplicación funcione según los requisitos pedidos. Entonces usarás el seeder para generar los estados de los artículos iniciales que te han pedido.

**c. Slug**

**R/.**

Los slugs son formas “amigables” de representar el texto de un enlace de tal manera que no se muestren caracteres extraños o variables confusas y que pueda ser leído y entendido por una persona. Desde hace algunos años los buscadores han tomado este factor como un determinante fundamental en el posicionamiento de resultados en búsquedas web, el slug puede contener las palabras clave de un artículo y de esta forma indicarle al lector si la publicación está relacionada con el término de la búsqueda.

En Laravel es muy fácil hacer rutas amigables y existen muchas formas de hacerlo, hoy les explicaré un modo fácil y sencillo de lograr este resultado con un pequeño ejemplo.

Como siempre, antes de empezar debemos tener configurada la conexión a la base de datos desde el archivo .env. Luego crearemos un nuevo modelo llamado Post.

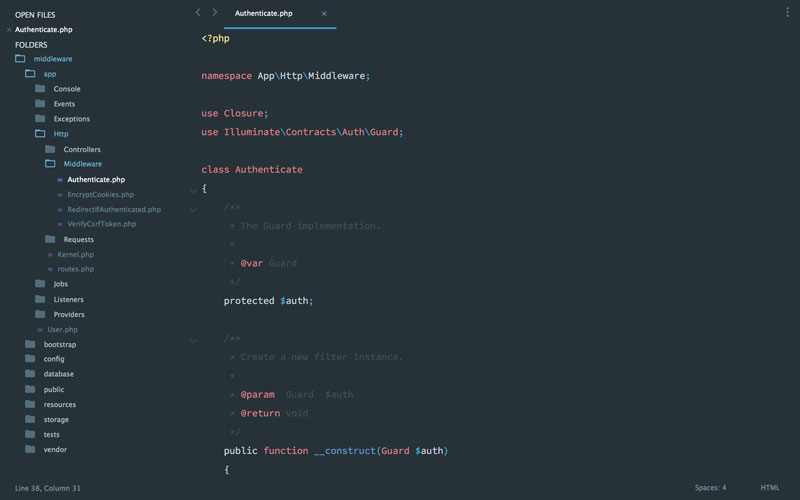
**d. Middleware**

**R/.**

Un middleware es un mecanismo que se utiliza para filtrar las peticiones HTTP en una aplicación. Están disponibles desde la versión 5 de Laravel y te permiten agregar capas adicionales a la lógica de tu aplicación.

El ejemplo más común de un Middleware es el de autenticación. Este nos serviría, por ejemplo, en un sitio en donde es necesario iniciar sesión y tener ciertos privilegios para poder ver el contenido de determinadas páginas.

El Middleware de autenticación es tan común que incluso Laravel 5.1 lo incluye por defecto como lo podemos ver en la siguiente imagen.



*BIBLIOGRAFIA*

[*https://laravel.com/docs/5.3/eloquent#soft-deleting*](https://laravel.com/docs/5.3/eloquent#soft-deleting)

[*http://www.desarrolloweb.com/articulos/seeders-laravel5.html*](http://www.desarrolloweb.com/articulos/seeders-laravel5.html)

[*https://styde.net/friendly-urls-rutas-slugs-en-laravel/*](https://styde.net/friendly-urls-rutas-slugs-en-laravel/)

[*https://platzi.com/blog/middlewares-laravel/*](https://platzi.com/blog/middlewares-laravel/)