CAPÍTULO 10

codea
Cursos de programación con proyectos

Gestión de Posts

Desarrollo de un Blog Corporativo con Laravel
Crear y gestionar publicaciones en un blog corporativo.

Objetivos

- Crear el modelo de **Post** y su relación con **Category**.
- Implementar la gestión de publicaciones en el panel de administración.
- Introducir el concepto de **SEO para Posts** con **slugs** y **meta descripciones**.
- Implementación exclusiva para el rol **Admin**.

📁 Estructura del módulo

```
laratech/
│
├── app/
│   ├── Models/
│   │   └── Post.php
│   └── Http/
│       └── Controllers/
│           └── PostController.php
│
├── database/
│   └── migrations/
│       └── xxxx_xx_xx_create_posts_table.php
│
├── routes/
│   └── web.php
├── resources/
│   └── views/
│       └── posts/
│           ├── index.blade.php
│           ├── create.blade.php
│           ├── edit.blade.php
│           ├── show.blade.php
│           └── _form.blade.php
│
├── public/
│   └── uploads/
│       └── posts/
│
└── .env
```

Creación del Modelo Post | Tabla Posts

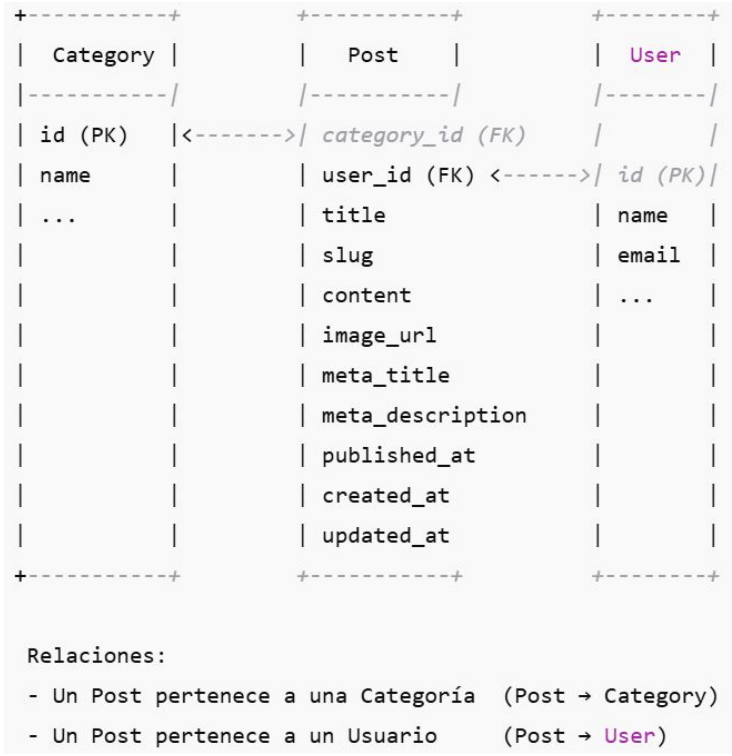
Pasos para crear el modelo:

- **Generación del modelo y migración:**
  `php artisan make:model Post -m`
- **Definición de los campos del modelo:**
  - `title`: Título de la publicación.
  - `slug`: Slug amigable para SEO.
  - `content`: Contenido completo.
  - `category_id`: Relación con la categoría.
  - `user_id`: Autor del post.
  - `image_url`: Imagen destacada.
  - `meta_title`: Título SEO.
  - `meta_description`: Descripción SEO.
  - `published_at`: Fecha de publicación.

📌 ***Importante:***

El slug debe ser único y generado automáticamente.

Establecer la relación entre los Modelos



🔁 Relaciones en Laravel

Modelo Post : pertenece a

```php
public function category()
{
    return $this->belongsTo(Category::class);
}

public function user()
{
    return $this->belongsTo(User::class);
}
```

Modelo Category : tiene muchos posts

```php
public function posts()
{
    return $this->hasMany(Post::class);
}
```

Modelo User : tiene muchos post

```php
public function posts()
{
    return $this->hasMany(Post::class);
}
```

Controlador PostController

Acciones:

- Lista de posts —----- index(){...}
- Formulario create — create(){...}
- Insertar post —------- store(Request $request){...}
- Formulario edit —---- edit(Post $post){...}
- Actualizar post —---- update(Request $request, Post $post){...}
- Borrar post —-------- destroy(Post $post){...}

Panel de Administración - Vista de Posts

Todas las vistas del CRUD de servicios **extienden un layout principal** ubicado en:

`resources/views/layouts/admin.blade.php`

- **Index (Lista de Posts):**

Mostrar todos los posts con título, fecha de publicación y categoría. Botones para editar y eliminar.

- **Create (Crear Post):**

Formulario para crear un nuevo post con campos de título, contenido, categoría, imagen, y SEO.

- **Edit (Editar Post):**

Formulario similar a Create, pero con los datos del post cargados para editar.

📌 *Vistas organizadas en `admin/post/`.*

Resumen y Conclusiones

- Hemos creado el modelo de **Post** con su relación a **Category**.
- Implementamos un **slug estático** para SEO.
- Se implementó el CRUD completo para gestionar los posts en el panel de administración.
- El rol **Admin** tiene acceso exclusivo para crear y editar los posts.

codea
Cursos de programación con proyectos

FIN

Curso completo : Laravel Página Web Administrable