LFTS - Comments

52. Write the Markup for a Post Comment

Creamos la vista para los commentarios, de la misma manera de la que lo hemos venido haciendo, utilizando blade

resources > views > components > post-comment.blade.php > article.flex.bg-gray-100.border.border-gray-200.p-6.rounded-xl.space-x-4 > div >

```
<article class="flex bg-gray-100 border border-gray-200 p-6 rounded-xl space-x-4">
    <div class="flex-shrink-0">
        <img src="https://i.pravatar.cc/100" alt="" width="60" height="60" class="rounded-xl">
    </div>

    <div>
        <header class="mb-4">
            <h3 class="font-bold">Kim Wexler</h3>
            <p class="text-xs">
                Posted
                <time>8 months ago</time>
            </p>
        </header>
        <p>
            Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore
        </p>
    </div>
</article>
```

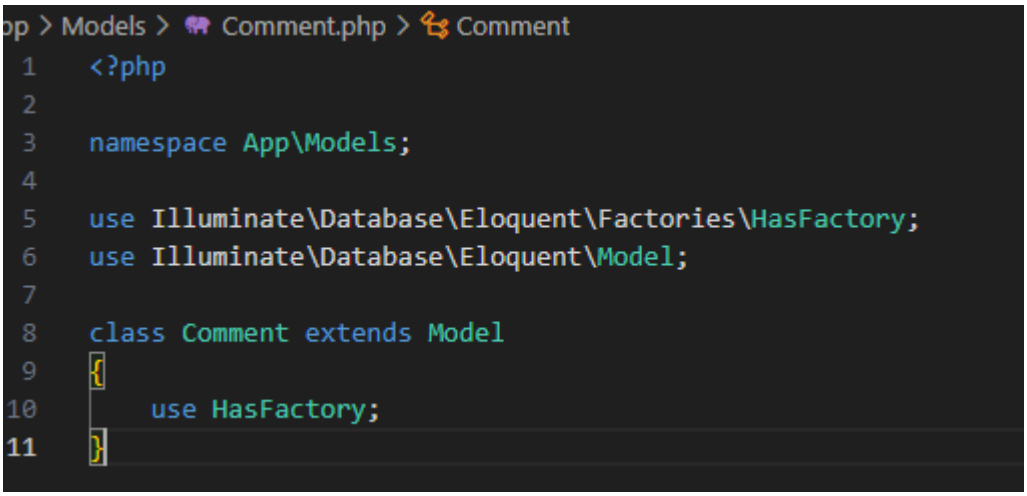
```
        <section class="col-span-8 col-start-5 mt-10 space-y-6">
            <x-post-comment />
        </section>
    </section>
```

53. Table Consistency and Foreign Key Constraints

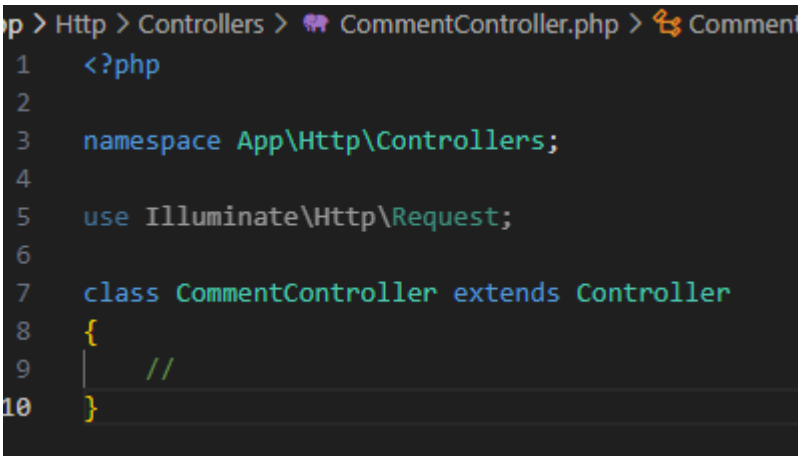
Ahora que creamos la vista para los comentarios tenemos que agregar la tabla y darle consistencia para poder usarla igual que las otras tablas que hemos creado

Para crear todo esto a la vez utilzamos el php artisan

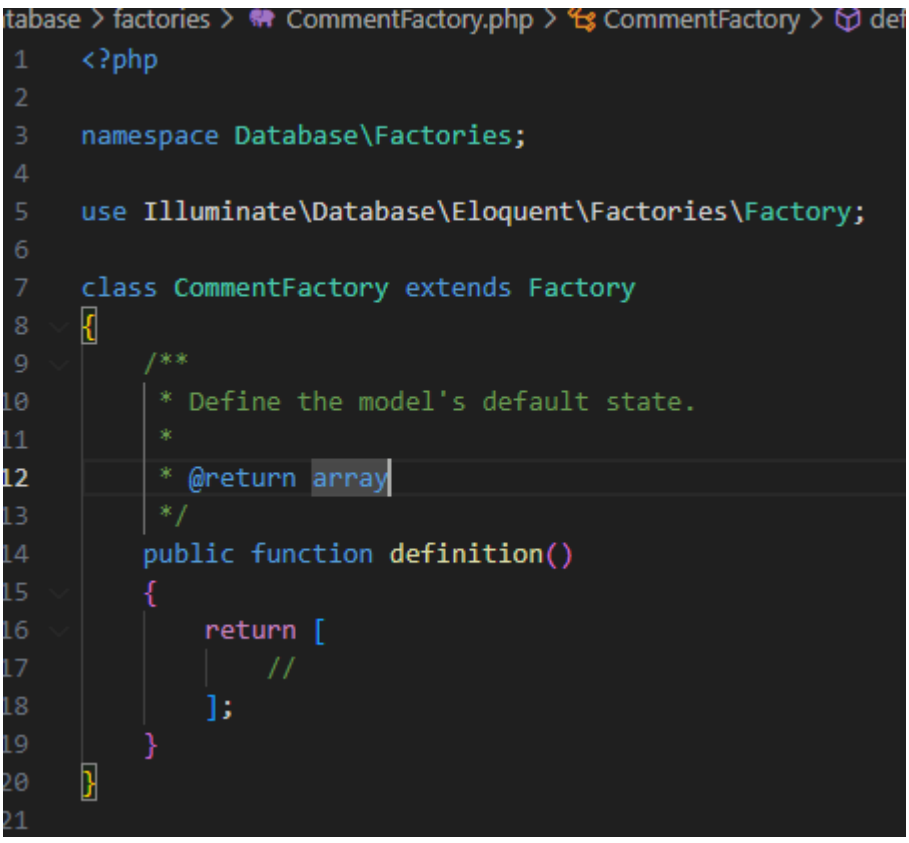
```
php artisan make:model -cfm
```



```
pp > Models > Comment.php > Comment
1   <?php
2
3   namespace App\Models;
4
5   use Illuminate\Database\Eloquent\Factories\HasFactory;
6   use Illuminate\Database\Eloquent\Model;
7
8   class Comment extends Model
9   {
10      use HasFactory;
11  }
```



```
p > Http > Controllers > CommentController.php > Comment
1   <?php
2
3   namespace App\Http\Controllers;
4
5   use Illuminate\Http\Request;
6
7   class CommentController extends Controller
8   {
9       //
10  }
```



```
tabase > factories > CommentFactory.php > CommentFactory > def
1   <?php
2
3   namespace Database\Factories;
4
5   use Illuminate\Database\Eloquent\Factories\Factory;
6
7   class CommentFactory extends Factory
8   {
9       /**
10       * Define the model's default state.
11       *
12       * @return array
13       */
14      public function definition()
15      {
16          return [
17              //
18          ];
19      }
20  }
21
```

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateCommentsTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('comments', function (Blueprint $table) {
            $table->id();
            $table->foreignId('post_id')->constrained()->cascadeOnDelete();
            $table->foreignId('user_id')->constrained()->cascadeOnDelete();
            $table->text('body');
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('comments');
    }
```

```php
            $table->foreignId('user_id')->constrained()->cascadeOnDelete();
            $table->foreignId('category_id')->constrained()->cascadeOnDelete();
            $table->string('slug')->unique();
```

54. Make the Comments Section Dynamic

Como el titulo indica en este episodio hacemos los titulos para que funcionen de manera dinamica

```php
            'user_id' => User::factory(),
            'post_id' => Post::factory(),
            'body' => $this->faker->paragraph()
```

```php
namespace App\Models;

use App\Models\Post;
use App\Models\User;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Database\Eloquent\Factories\HasFactory;

class Comment extends Model
{
    use HasFactory;

    public function post()
    {
        //hasOne, hasMany, belongsTo, belongsToMany
        return $this->belongsTo(Post::class);
    }

    public function author()
    {
        //hasOne, hasMany, belongsTo, belongsToMany
        return $this->belongsTo(User::class, 'user_id');
    }
}
```

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

```php
    {
        $user = User::factory()->create([
            'name' => 'Tommy Shelby',
            'username' => 'TommyShelby',
        ]);
        Comment::factory(10)->create([
            'user_id' => $user->id,
        ]);
    }
```

```blade
@props(['comment'])

<article class="flex bg-gray-100 border border-gray-200 p-6 rounded-xl spac
    <div class="flex-shrink-0">
        <img src="https://i.pravatar.cc/60?u={{$comment->id}}" alt="" width
    </div>

    <div>
        <header class="mb-4">
            <h3 class="font-bold">{{$comment->author->username}}</h3>
            <p class="text-xs">
                Posted
                <time>{{$comment->created_at->diffForHumans()}}</time>
            </p>
        </header>
        <p>
            {{$comment->body}}
        </p>
```

```blade
        <section class="col-span-8 col-start-5 mt-10 space-y-6">
            @foreach ($post->comments as $comment)
            <x-post-comment :comment="$comment" />
            @endforeach
        </section>
```

55. Design the Comment Form

> Creamos un formulario para que los usuarios puedan crear los commentarios

```blade
<x-panel class="bg-gray-50">
    <article class="flex space-x-4">
        <div class="flex-shrink-0">
            <img src="https://i.pravatar.cc/60?u={{$comment->id}}" alt="" width="60" height="60" class="round
        </div>

        <div>
            <header class="mb-4">
                <h3 class="font-bold">{{$comment->author->username}}</h3>
                <p class="text-xs">
                    Posted
                    <time>{{$comment->created_at->diffForHumans()}}</time>
                </p>
            </div>
    </article>
</x-panel>
```

```blade
<section class="col-span-8 col-start-5 mt-10 space-y-6">
    <x-panel>
        <form action="#" method="POST">
            @csrf
            <header class="flex items-center">
                <img src="https://i.pravatar.cc/40?u={{auth()->id()}}" alt="" width="40" height="40"
                <h2 class="ml-4">Want to participate?</h2>
            </header>
            <div class="mt-5">
                <textarea name="body" id="body" rows="5" class="w-full text-sm focus:outline-none foc
            </div>
            <div class="flex justify-end mt-5 border-t border-gray-200 pt-5">
                <button type="submit" class="bg-blue-500 text-white uppercase font-semibold text-xs p
            </div>
        </form>
    </x-panel>
```

resources > views > components > panel.blade.php > div

```blade
<div {{$attributes(['class' => 'border border-gray-200 p-6 rounded-xl'])}}>
    {{$slot}}
</div>
```

56. Activate the Comment Form

Ahora que tenemos creado el formulario vamos a activarlo y convertirlo de algo dummy a algo realmente funcional

```php
Route::get('/', [PostController::class, 'index'])->name('home');
Route::get('/posts/{post:slug}', [PostController::class, 'show'])->name('post');
Route::post('/posts/{post:slug}/comments', [CommentController::class, 'store']);

```

```blade
<section class="col-span-8 col-start-5 mt-10 space-y-6">
@auth
                <x-panel>
                    <form action="/posts/{{$post->slug}}/comments" method="POST">
                        @csrf
                        <header class="flex items-center">
                            <img src="https://i.pravatar.cc/40?u={{auth()->id()}}" alt="" width="40"
                            <h2 class="ml-4">Want to participate?</h2>
                        </header>
                        <div class="mt-5">
                            <textarea name="body" id="body" rows="5" class="w-full text-sm focus:outl
                        </div>
                        <div class="flex justify-end mt-5 border-t border-gray-200 pt-5">
                            <button type="submit" class="bg-blue-500 text-white uppercase font-semibo
                        </div>
                    </form>
                </x-panel>
            @else
                <p class="font-semibold">
                    <a href="/regsiter" class="hover:underline">Register</a> or <a href="/login" clas
                </p>
            @endauth
```

```blade
<div class="flex-shrink-0">
<img src="https://i.pravatar.cc/60?u={{$comment->user_id}}" alt="" width="60" height="60" class="r
</div>

<div>
    <header class="mb-4">
        <h3 class="font-bold">{{$comment->author->username}}</h3>
        <p class="text-xs">
            Posted
            <time>{{$comment->created_at->format('F j, Y, g:i a')}}</time>
        </p>
    </header>
```

```php
        Paginator::useTailwind();
        //Model::unguard();
    }
```

```php

    protected $guarded = [];

```

```php
{
    public function store(Post $post)
    {

        request()->validate([
            'body' => 'required',
        ]);

        $post->comments()->create([
            'body' => request('body'),
            'user_id' => request()->user()->id,
        ]);

        return back();
    }
```

57. Some Light Chapter Clean Up

> En este episodio hacemos unos pequeños ajuste para que se vea mejor y sea un poco mas amigable con el usuario

resources > views > components > submit-button.blade.php > button.bg-blue-500.text-white.uppercase.font-semibold.text-xs.py-2.px-10.rounded

```blade
<button type="submit" class="bg-blue-500 text-white uppercase font-semibold text-xs py-2 px-10 rounded-2xl h
    {{$slot}}
</button>
```

```blade
@auth
    <x-panel>
        <form action="/posts/{{$post->slug}}/comments" method="POST">
            @csrf
            <header class="flex items-center">
                <img src="https://i.pravatar.cc/40?u={{auth()->id()}}" alt="" width="40" height="40" class="r
                <h2 class="ml-4">Want to participate?</h2>
            </header>
            <div class="mt-5">
                <textarea name="body" id="body" rows="5" class="w-full text-sm focus:outline-none focus:ring"
                @error('body')
                    <span class="text-xs text-red-500">{{$message}}</span>
                @enderror
            </div>
            <div class="flex justify-end mt-5 border-t border-gray-200 pt-5">
                <x-submit-button>Post</x-submit-button>
            </div>
        </form>
    </x-panel>
@else
    <p class="font-semibold">
        <a href="/regsiter" class="hover:underline">Register</a> or <a href="/login" class="hover:underline">
    </p>
@endauth
```

```blade
        <section class="col-span-8 col-start-5 mt-10 space-y-6">
            @include('posts._add-comment-form')

            @foreach ($post->comments as $comment)
                <x-post-comment :comment="$comment" />
```