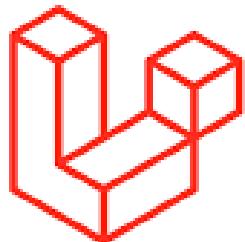


Projecte GLOBAL: VideosApp

2r SPRINT



Professor: Jordi Vega
Assignatures: M7, M8 i M9
Nom: Harvey
Cognoms: John Glover



INDEX:

Corregir els errors del 1r sprint	3
db_connection i db_database	3
Migració de camps	4
Controladors	5
Model de Videos	6
Helper default videos	8
DatabaseSeeder	9
VideosAppLayout	11
Ruta show de videos	13
Vista show de videos	14
Helpers test	18
Guia pas a pas utilitzant la metodologia TDD i AAA	20
VideosTest tests/Unit	22
VideosTest tests/Feature/Videos	25
Crear el terms	29
Larastan tests	30
Instalació	30
Errors:	31
CreateTeam.php:	32
DeleteUser.php:	33
RemoveTeamMember.php:	34
InviteTeamMember.php:	35
Team.php:	36
User.php:	37
console.php:	39
ExampleTest.php:	40
ProfileInformationTest.php:	41
Analisis final:	41



Corregir els errors del 1r sprint

No hi ha cap error ja que la el Sprint 1 estava perfecte.

Retroacció

Qualificació	10,00 / 10,00
Qualificat el	dimarts, 17 de desembre 2024, 15:13
Qualificat per	JV Jordi Vega Tomàs [PROF]
Comentaris de retroalimentació	Molt bon treball. És un test Unit no de Features.

db_connection i db_database

Fitxer abans de canvis:

```
<php>
    <env name="APP_ENV" value="testing"/>
    <env name="APP_MAINTENANCE_DRIVER" value="file"/>
    <env name="BCRYPT_ROUNDS" value="4"/>
    <env name="CACHE_STORE" value="array"/>
    <!-- <env name="DB_CONNECTION" value="sqlite"/> -->
    <!-- <env name="DB_DATABASE" value=":memory:"/> -->
```

Fitxer després de canvis:

```
<php>
    <env name="APP_ENV" value="testing"/>
    <env name="APP_MAINTENANCE_DRIVER" value="file"/>
    <env name="BCRYPT_ROUNDS" value="4"/>
    <env name="CACHE_STORE" value="array"/>
    <env name="DB_CONNECTION" value="sqlite"/>
    <env name="DB_DATABASE" value=":memory:"/>
```

Migració de camps

Crear el fitxer per fer la migració de vídeos.

```
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ php artisan make:migration create_videos_table --create=videos
INFO Migration [database/migrations/2025_01_17_163312_create_videos_table.php] created successfully.

alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$
```

Hauran els camps (id, title, description, url, published_at, previous, next, series_id i el per defecte de timestamps)

```
public function up(): void
{
    Schema::create('videos', function (Blueprint $table) {
        $table->id();
        $table->string('title');
        $table->text('description');
        $table->string('url');
        $table->timestamp('published_at')->nullable();
        $table->integer('previous')->nullable();
        $table->integer('next')->nullable();
        $table->unsignedBigInteger('series_id');
        $table->timestamps();
        // $table->foreign('series_id')->references('id')->on('series');
    });
}
```

Executar la migració de les taules.

```
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ php artisan migrate
INFO Running migrations.

2025_01_17_163312_create_videos_table ..... 9.51ms DONE

alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$
```

Controladors

Començo creant el controlador de Videos.

```
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ php artisan make:controller VideosController
INFO Controller [app/Http/Controllers/VideosController.php] created successfully.

alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$
```

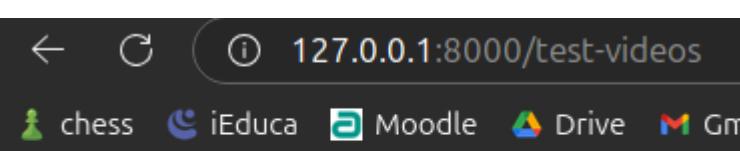
Controlador de videos tindrà les funcions testedBy i el show. Comencem mostrar el testedBy ja que no requereix molt de misteri i per fer una prova inicial.

```
/***
 * Funció de prova per verificar el bon funcionament del controlador.
 *
 * @return string
 */
1 usage
public function testedBy(): string
{
    return "El controlador de vídeos funciona correctament.";
}
```

Al web.php afegirem la següent línia de text per fer la prova a la ruta (/test-videos)

```
Route::get('uri: '/test-videos', [VideosController::class, 'testedBy']);
```

Executem el programa, i veurem que ens mostra el missatge correctament per tant funciona bé i sense problemes.



El controlador de vídeos funciona correctament.

Model de Videos

Comencem creant el model de Video.

```
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ php artisan make:model Video

INFO Model [app/Models/Video.php] created successfully.

alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$
```

Al fitxer de video.php procurarem tenir (@property) per totes les variables necessàries especificant cada un.

```
© Video.php ×

1 <?php
2
3 namespace App\Models;
4
5 use Illuminate\Database\Eloquent\Factories\HasFactory;
6 use Illuminate\Database\Eloquent\Model;
7 use Carbon\Carbon;
8
9
10 /**
11 * App\Models\Video
12 *
13 * @property int $id
14 * @property string $title
15 * @property string $description
16 * @property string $url
17 * @property string|null $previous
18 * @property string|null $next
19 * @property int $series_id
20 * @property Carbon|null $published_at
21 * @property Carbon|null $created_at
22 * @property Carbon|null $updated_at
23 * @property-read string|null $formatted_published_at
24 * @property-read string|null $formatted_for_humans_published_at
25 * @property-read int|null $published_at_timestamp
26 */
```

Aquesta classe Video és un model de Laravel que gestiona la informació dels vídeos, incloent-hi el títol, descripció, URL i data de publicació. Utilitza mètodes per formatar la data de publicació de manera llegible (ex. "22 de gener de 2025"), calcular la diferència de temps respecte a l'actualitat (ex. "fa 3 dies") i obtenir el segell temporal Unix de la data procurant d'usar el Carbon per les dates i info.

```
class Video extends Model
{
    use HasFactory;
    no usages
    protected $fillable = ['title', 'description', 'url', 'previous', 'next', 'series_id', 'published_at'];

    /** @var array<string> */
    no usages
    protected array $dates = ['published_at'];

    /** @return string|null */
    no usages
    public function getFormattedPublishedAtAttribute(): ?string
    {
        if ($this->published_at) {
            return Carbon::parse($this->published_at)->isoFormat('D [de] MMMM [de] YYYY');
        }
        return null;
    }

    /** @return string|null */
    no usages
    public function getFormattedForHumansPublishedAtAttribute(): ?string
    {
        if ($this->published_at) {
            return Carbon::parse($this->published_at)->diffForHumans();
        }
        return null;
    }

    /** @return int|null */
    no usages
    public function getPublishedAtTimestampAttribute(): ?int
    {
        if ($this->published_at) {
            return Carbon::parse($this->published_at)->timestamp();
        }
        return null;
    }
}
```

Helper default videos

Crear un helper de default videos.

```
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ touch app/Helpers/VideoHelper.php
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ ls app/Helpers/
UserHelper.php  VideoHelper.php
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$
```

Dintre del helper creo els vídeos amb la info corresponent, procurant tenir el next i previous ben estructurat amb l'ordre correcte.

```
⑥ VideoHelper.php ×
  4 usages ▲ Wharvey123 *
  7 class VideoHelper
  8 {
  9     /**
10      * Crea vídeos per defecte a la base de dades.
11      * @return void
12     */
13     public static function createDefaultVideos()
14     {
15         $defaultVideos = [
16             [
17                 'title' => 'Introducció a Laravel',
18                 'description' => 'Un vídeo introductori per aprendre els conceptes bàsics de Laravel.',
19                 'url' => 'https://www.youtube.com/embed/PGQxTILBb7M',
20                 'published_at' => now(),
21                 'previous' => null,
22                 'next' => 2,
23                 'series_id' => 1,
24             ],
25             [
26                 'title' => 'Controladors a Laravel',
27                 'description' => 'Aprèn com funcionen els controladors a Laravel i com utilitzar els mètodes GET, POST, PUT, DELETE.',
28                 'url' => 'https://www.youtube.com/embed/0YxgCH2R2bE',
29                 'published_at' => now(),
30                 'previous' => 1,
31                 'next' => 3,
32                 'series_id' => 1,
33             ],
34             [
35                 'title' => 'Models a Laravel',
36                 'description' => 'Exploració dels models a Laravel i com utilitzar els mètodes find, first, where, etc.',
37                 'url' => 'https://www.youtube.com/embed/f-pWNf0Ht1Y',
38                 'published_at' => now(),
39                 'previous' => 2,
40                 'next' => null,
41                 'series_id' => 1,
42             ],
43         ];
44     }

foreach ($defaultVideos as $video) {
    Video::create($video);
}
```

DatabaseSeeder

Posaré els usuaris i vídeos per defecte al DatabaseSeeder.

```
© DatabaseSeeder.php ×
1 <?php
2
3 namespace Database\Seeders;
4
5 use App\Models\User;
6 use Illuminate\Database\Seeder;
7 use App\Helpers\UserHelper;
8 use App\Helpers\VideoHelper;
9
10 no usages ± Wharvey123 *
11 class DatabaseSeeder extends Seeder
12 {
13     /**
14      * Seed the application's database.
15      */
16     ± Wharvey123 *
17     public function run(): void
18     {
19         // Create the default user and professor
20         UserHelper::createDefaultUser();
21         UserHelper::createDefaultProfessor();
22
23         // Crea vídeos per defecte
24         VideoHelper::createDefaultVideos();
25     }
26 }
```

Executare el (db:seed) per aplicar els canvis.

```
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ php artisan db:seed

INFO  Seeding database.

alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$
```

Una vegada fet el seeder la taula és generara amb la info correcta.

The screenshot displays three separate MySQL Workbench windows, each showing the 'videos' table with different columns and data. The top window shows columns: id, title, and description. The middle window shows columns: url, published_at, and previous. The bottom window shows columns: next, series_id, created_at, and updated_at.

1	1	Introducció a Laravel	Un vídeo introductori per aprendre els conce...
2	2	Controladors a Laravel	Aprèn com funcionen els controladors a Larav...
3	3	Models a Laravel	Exploració dels models a Laravel i com interactu...

1	https://www.youtube.com/embed/PGQxIILBb7M	2025-01-22 18:35:57	<null>
2	https://www.youtube.com/embed/0YxgCH2R2bE	2025-01-22 18:35:57	1
3	https://www.youtube.com/embed/f-pWNf0Ht1Y	2025-01-22 18:35:57	2

2		1	2025-01-22 18:35:57
3		1	2025-01-22 18:35:57
<null>		1	2025-01-22 18:35:57

VideosAppLayout

Crear un layout que es digui VideosAppLayout, estara tant a app/View/components com a resources/views/layouts.

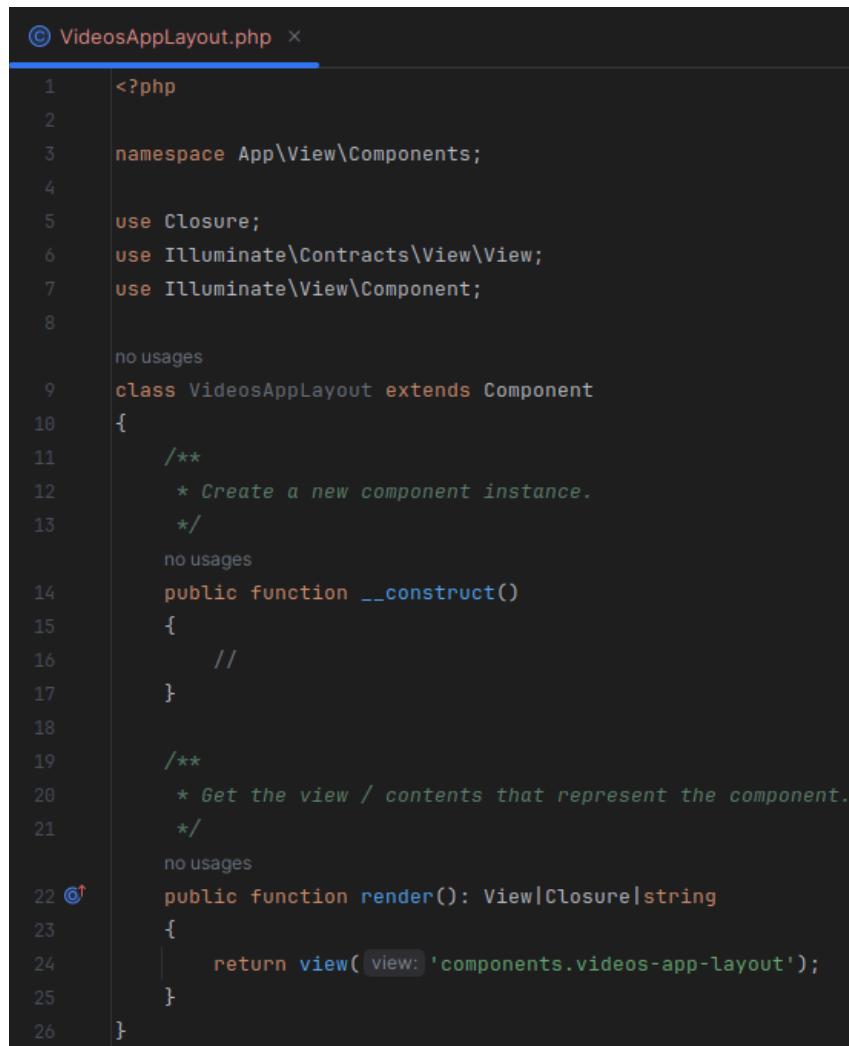
```
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ php artisan make:component VideosAppLayout

[INFO] Component [app/View/Components/VideosAppLayout.php] created successfully.

[INFO] View [resources/views/components/videos-app-layout.blade.php] created successfully.

alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$
```

El VideosAppLayout.php de components per defecte quedara així:



```
① VideosAppLayout.php ×

1  <?php
2
3  namespace App\View\Components;
4
5  use Closure;
6  use Illuminate\Contracts\View\View;
7  use Illuminate\View\Component;
8
9  no usages
10 class VideosAppLayout extends Component
11 {
12     /**
13      * Create a new component instance.
14      */
15     no usages
16     public function __construct()
17     {
18         //
19     }
20     /**
21      * Get the view / contents that represent the component.
22      */
23     no usages
24     public function render(): View|Closure|string
25     {
26         return view('components.videos-app-layout');
```



El layout serà com el següent amb un head i body, on dintre del body haurà el contingut de cada pàgina que usa el layout.

```
videos-app-layout.blade.php ×  
1  <!DOCTYPE html>  
2  <html lang="ca">  
3  <head>  
4      <meta charset="UTF-8">  
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">  
6      <title>@yield('title', 'Video App')</title>  
7      <script src="https://cdn.tailwindcss.com"></script>  
8  </head>  
9  <body class="bg-gray-900 text-gray-200 font-sans leading-relaxed tracking-normal">  
10  
11     <!-- Contingut principal -->  
12     <div class="container mx-auto px-4 py-8">  
13         @yield('content')  
14     </div>  
15  
16     </body>  
17     </html>
```

Ruta show de videos

Crearem la ruta del show de videos al fitxer web.php per a que quan entrem a la ruta (/videos) seguit per un id com ara (1) ens mostrerà el primer video.

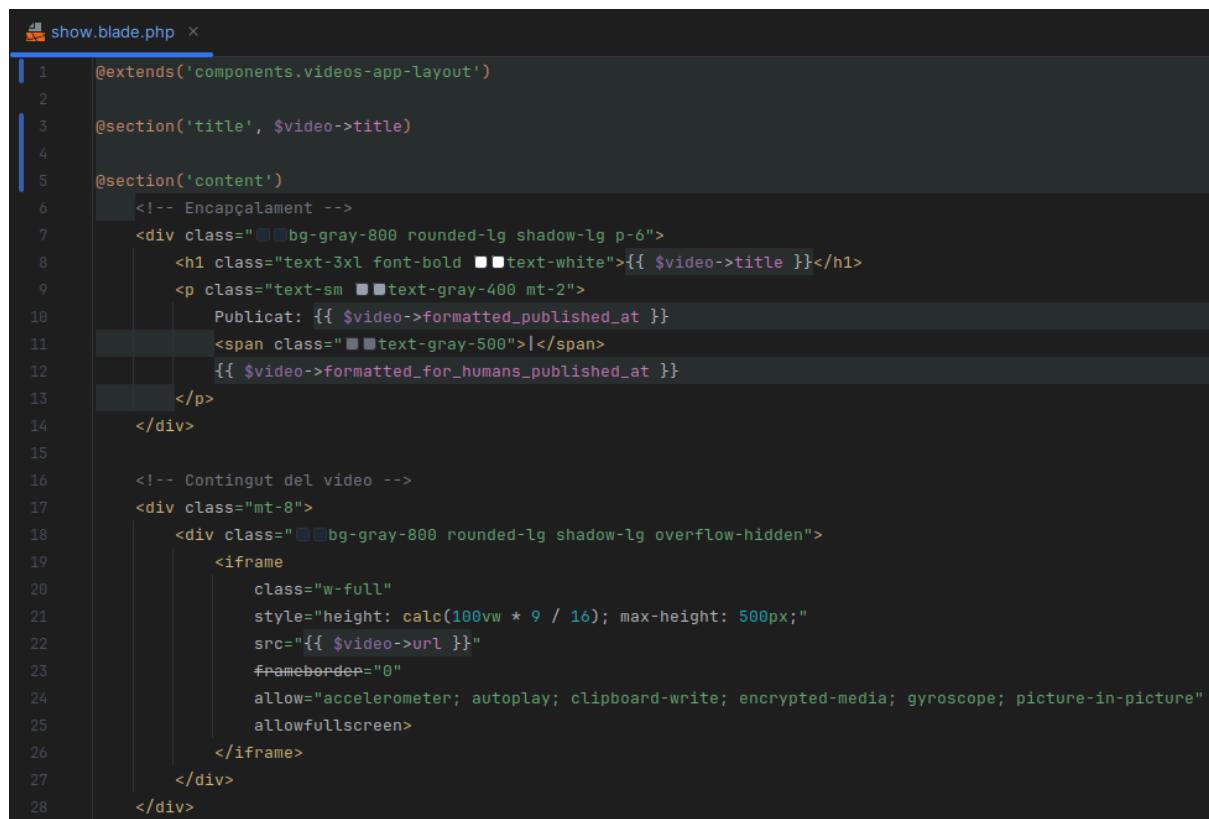
```
php web.php ×
1 <?php
2
3 use Illuminate\Support\Facades\Route;
4 use App\Http\Controllers\VideosController;
5
6 Route::get('/', function () {
7     return view('welcome');
8 });
9
10 Route::middleware([
11     'auth:sanctum',
12     config(key: 'jetstream.auth_session'),
13     'verified',
14 ])->group(function () {
15     Route::get(uri: '/dashboard', function () {
16         return view(view: 'dashboard');
17     })->name(name: 'dashboard');
18 });
19
20 Route::get(uri: '/video/{id}', [VideosController::class, 'show'])->name(name: 'video.show');
21
22 Route::get(uri: '/test-videos', [VideosController::class, 'testedBy']);|
```

Vista show de vídeos

Crear la vista del show de vídeos.

```
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ mkdir -p resources/views/videos
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ touch resources/views/videos/show.blade.php
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ ls -l resources/views/videos/
total 0
-rw-rw-r-- 1 alumnat alumnat 0 de gen. 20 19:58 show.blade.php
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$
```

Aquest codi utilitza el sistema de plantilles Blade de Laravel per mostrar una pàgina de vídeo personalitzada. Hereta l'estructura general del layout videos-app-layout amb @extends i defineix el títol i el contingut específic amb @section. El contingut inclou l'encapçalament amb el títol i la data del vídeo, el reproductor embeddit, la descripció, i la navegació entre vídeos (anterior i següent). Aquest enfocament manté el codi modular i fàcil de gestionar.



```
show.blade.php
1 @extends('components.videos-app-layout')
2
3 @section('title', $video->title)
4
5 @section('content')
6     <!-- Encapçalament -->
7     <div class="bg-gray-800 rounded-lg shadow-lg p-6">
8         <h1 class="text-3xl font-bold text-white">{{ $video->title }}</h1>
9         <p class="text-sm text-gray-400 mt-2">
10             Publicat: {{ $video->formatted_published_at }}
11             <span class="text-gray-500">|</span>
12             {{ $video->formatted_for_humans_published_at }}
13         </p>
14     </div>
15
16     <!-- Contingut del vídeo -->
17     <div class="mt-8">
18         <div class="bg-gray-800 rounded-lg shadow-lg overflow-hidden">
19             <iframe
20                 class="w-full"
21                 style="height: calc(100vw * 9 / 16); max-height: 500px;"
22                 src="{{ $video->url }}"
23                 frameborder="0"
24                 allow="accelerometer; autoplay; clipboard-write; encrypted-media; gyroscope; picture-in-picture"
25                 allowfullscreen>
26             </iframe>
27         </div>
28     </div>
```

```
<!-- Descripció -->
<div class="mt-8 bg-gray-800 rounded-lg shadow-lg p-6">
    <h2 class="text-xl font-semibold text-white">Descripció</h2>
    <p class="text-gray-300 mt-4">{{ $video->description }}</p>
</div>

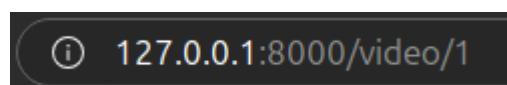
<!-- Navegació entre vídeos -->
<div class="mt-8 flex justify-between items-center text-sm">
    @if($video->previous)
        <a href="{{ url('/video/' . $video->previous) }}"
            class="text-blue-400 hover:text-blue-600 font-medium">
            &larr; Video anterior
        </a>
    @else
        <span class="text-gray-500">No hi ha vídeo anterior</span>
    @endif

    @if($video->next)
        <a href="{{ url('/video/' . $video->next) }}"
            class="text-blue-400 hover:text-blue-600 font-medium">
            Video següent &rarr;
        </a>
    @else
        <span class="text-gray-500">No hi ha vídeo següent</span>
    @endif
</div>
@endsection
```

Inicia l'app.

```
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ php artisan serve
INFO | Server running on [http://127.0.0.1:8000].
```

Entro a la url del video 1.



Com és veu a continuació, em mostra el títol, temps publicat, el video, descripció, i botons per següent i previ.

Veurem que el botó de previ no està actiu ja que és el video número 1.

Introducció a Laravel
Publicat: 20 de January de 2025 | 22 hours ago

Mira a YouTube

Descripció
Un vídeo introductori per aprendre els conceptes bàsics de Laravel.

No hi ha vídeo anterior vídeo següent →

Aquí veurem el segon video que té el previous i next opcions actives.

Controladores a Laravel
Publicat: 19 de January de 2025 | 1 day ago

Mira a YouTube

Descripció
Apren com funcionen els controladors a Laravel i com gestionar les rutes.

← Vídeo anterior vídeo següent →



I finalment el ultim ja no te next perquè no hi han més.

Models a Laravel

Publicat: 22 de January de 2025 | 21 minutes ago

Mira a [YouTube](#)

8. LARAVEL 9 - MODELOS, que son y como usarlos

Laravel

Descripció

Exploració dels models a Laravel i com interactuar amb la base de dades.

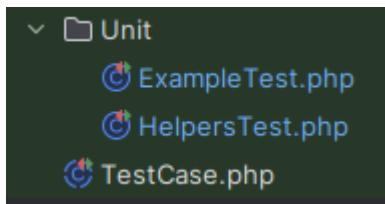
- Vídeo anterior

No hi ha vídeo següent



Helpers test

Posaré el test HelpersTest a la carpeta tests/Unit.



Executo el test per verificar que encara funcioni.

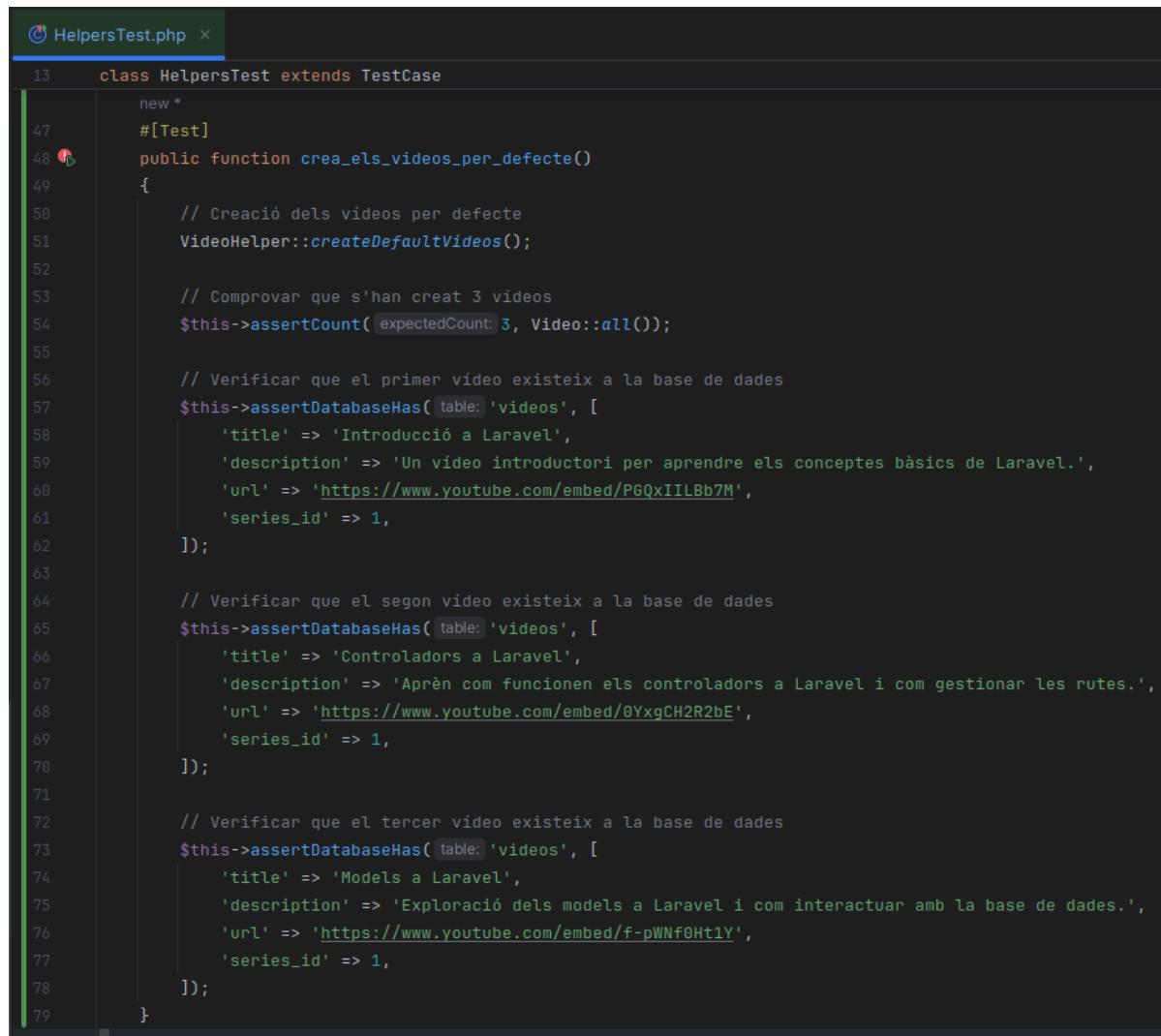
```
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ php artisan test --filter=HelpersTest

  PASS  Tests\Unit\HelpersTest
    ✓ crea els usuaris per defecte

    Tests:    1 passed (6 assertions)
    Duration: 0.14s

alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$
```

Afegire el test de la creació dels vídeos per defecte al fitxer HelpersTest comprovant la info addicional.



```
HelpersTest.php
13 class HelpersTest extends TestCase
14 {
15     new *
16
17     #[Test]
18     public function crea_els_videos_per_defecte()
19     {
20         // Creació dels vídeos per defecte
21         VideoHelper::createDefaultVideos();
22
23         // Comprovar que s'han creat 3 vídeos
24         $this->assertCount( expectedCount: 3, Video::all());
25
26         // Verificar que el primer video existeix a la base de dades
27         $this->assertDatabaseHas( table: 'videos', [
28             'title' => 'Introducció a Laravel',
29             'description' => 'Un vídeo introductori per aprendre els conceptes bàsics de Laravel.',
30             'url' => 'https://www.youtube.com/embed/PGQxIILBb7M',
31             'series_id' => 1,
32         ]);
33
34         // Verificar que el segon video existeix a la base de dades
35         $this->assertDatabaseHas( table: 'videos', [
36             'title' => 'Controladors a Laravel',
37             'description' => 'Aprèn com funcionen els controladors a Laravel i com gestionar les rutes.',
38             'url' => 'https://www.youtube.com/embed/0YxgCH2R2bE',
39             'series_id' => 1,
40         ]);
41
42         // Verificar que el tercer video existeix a la base de dades
43         $this->assertDatabaseHas( table: 'videos', [
44             'title' => 'Models a Laravel',
45             'description' => 'Exploració dels models a Laravel i com interactuar amb la base de dades.',
46             'url' => 'https://www.youtube.com/embed/f-pWNf0Ht1Y',
47             'series_id' => 1,
48         ]);
49     }
50 }
```



Guia pas a pas utilitzant la metodologia TDD i AAA

1. HelpersTest.php - Test per crear usuaris per defecte

Aquest test verifica la creació d'usuaris per defecte amb equips associats.

Pasos TDD:

- **Escrí el test (Fail):**
Primer, creo un test per verificar que es poden crear usuaris per defecte. Executo el test i comprovo que falla perquè els mètodes createDefaultUser i createDefaultProfessor encara no existeixen.
- **Escrí el codi mínim (Pass):**
Implemento els mètodes createDefaultUser i createDefaultProfessor a UserHelper. Faig que aquests mètodes creïn usuaris amb equips per defecte. Executo el test per assegurar-me que passa.
- **Refactoro:**
Optimitzo el codi dels mètodes per assegurar-me que compleixen amb les bones pràctiques i són fàcils de mantenir.

Pasos AAA:

- **Arrange:** Preparo el context, inicialitzo la base de dades i creo les taules necessàries.
- **Act:** Executo els mètodes createDefaultUser i createDefaultProfessor.
- **Assert:** Verifico que:
 - Els usuaris creats són instàncies del model User.
 - Cada usuari té un equip associat amb les dades correctes a la base de dades.



2. HelpersTest.php - Test per crear vídeos per defecte

Aquest test comprova que es poden crear tres vídeos predeterminats a la base de dades.

Pasos TDD:

- **Escric el test (Fail):**
Creo un test que verifiqui la creació de vídeos per defecte. Executo el test i comprovo que falla perquè el mètode createDefaultVideos encara no existeix.
- **Escric el codi mínim (Pass):**
Implemento el mètode createDefaultVideos a VideoHelper i faig que afegeixi tres vídeos predeterminats. Executo el test fins que passi.
- **Refactoro:**
Milloro el codi de createDefaultVideos per fer-lo més lleigible, eficient i mantenible.

Pasos AAA:

- **Arrange:** Configuro el test inicialitzant la base de dades.
- **Act:** Crido el mètode createDefaultVideos.
- **Assert:** Comprovo que:
 - Hi ha exactament tres vídeos a la base de dades.
 - Els vídeos tenen els atributs específics: títol, descripció, URL i series_id.

Executo el test (HelpersTest) de nou i és veu que em fa el test correctament.

```
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ php artisan test --filter=HelpersTest

  PASS  Tests\Unit\HelpersTest
    ✓ crea els usuaris per defecte
    ✓ crea els videos per defecte

  Tests:    2 passed (10 assertions)
  Duration: 0.20s
```

VideosTest tests/Unit

Crear el test VideosTest a la carpeta tests/Unit.

```
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ php artisan make:test VideosTest --unit
INFO Test [tests/Unit/VideosTest.php] created successfully.

alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$
```

Crear les funcions per comprovar la formatació del video:
can_get_formatted_published_at_date() i
can_get_formatted_published_at_date_when_not_published().

```
VideosTest.php ×
1 <?php
2
3 namespace Tests\Unit;
4
5 > use ...
6
7
11 class VideosTest extends TestCase
12 {
13     use RefreshDatabase;
14     public function test_can_get_formatted_published_at_date()
15     {
16         // Configurar idioma a espanyol
17         Carbon::setLocale('es');
18
19         // Crear un video amb una data de publicació
20         $video = Video::factory()->create([
21             'published_at' => Carbon::parse('2025-01-17 14:00:00'),
22         ]);
23
24         // Comprovar que el format és correcte
25         $this->assertEquals('17 de enero de 2025', $video->formatted_published_at);
26     }
27
28     public function test_can_get_formatted_published_at_date_when_not_published()
29     {
30         // Crear un video sense data de publicació
31         $video = Video::factory()->create([
32             'published_at' => null,
33         ]);
34
35         // Comprovar que la propietat retorna null
36         $this->assertNull($video->formatted_published_at);
37     }
38 }
```



Juntament amb el test també hauré de crear el factory amb un return de la info necessaria.

```
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ php artisan make:factory VideoFactory --model=Video
INFO Factory [database/factories/VideoFactory.php] created successfully.

alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$
```

```
© VideoFactory.php ×
1  <?php
2
3  namespace Database\Factories;
4
5  > use ...
6
7  no usages
8
9  class VideoFactory extends Factory
10 {
11     protected $model = Video::class;
12
13     public function definition()
14     {
15         return [
16             'title' => $this->faker->sentence,
17             'description' => $this->faker->paragraph,
18             'url' => $this->faker->word,
19             'published_at' => Carbon::now(),
20             'previous' => null,
21             'next' => null,
22             'series_id' => 1,
23         ];
}
```



VideosTest.php (Unit) - Format de la data de publicació

Aquests tests comproven que la data de publicació d'un vídeo està correctament formatada.

Pasos TDD:

- **Escrí el test (Fail):**
Creo un test que verifiqui el format de la data de publicació d'un vídeo. Executo el test i comprovo que falla perquè la propietat formatted_published_at encara no existeix.
- **Escrí el codi mínim (Pass):**
Implemento la propietat formatted_published_at al model Video. Faig que aquesta propietat retorni el format correcte si published_at té un valor i null en cas contrari.
- **Refactoro:**
Milloro la implementació de la propietat per assegurar eficiència i claredat.

Pasos AAA:

- **Arrange:** Configuro el test creant vídeos amb i sense data de publicació.
- **Act:** Accedeixo a la propietat formatted_published_at.
- **Assert:** Verifico que:
 - La data es mostra amb el format correcte quan published_at no és nul.
 - Retorna null quan published_at és nul.

Executo el test (VideosTest) i és veu que em fa el test correctament.

```
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ php artisan test --filter=VideosTest

  PASS  Tests\Unit\VideosTest
    ✓ can get formatted published at date
    ✓ can get formatted published at date when not published

  Tests:    2 passed (2 assertions)
  Duration: 0.15s
```

VideosTest tests/Feature/Videos

Crear el test VideosTest a la carpeta tests/Feature/Videos.

```
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ php artisan make:test VideosTest
INFO Test [tests/Feature/VideosTest.php] created successfully.

alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$
```

Crear les funcions users_can_view_videos() i users_cannot_view_not_existing_videos() per fer la comprovació correcta del video.

```
VideosTest.php ×
  9  class VideosTest extends TestCase
10  {
11      use RefreshDatabase;
12
13
14      public function test_users_can_view_videos()
15      {
16          // Create a video in the database
17          $video = Video::factory()->create([
18              'title' => 'Test Video',
19              'description' => 'This is a test video.',
20              'published_at' => now(),
21          ]);
22
23
24          // Perform a GET request to the video's route
25          $response = $this->get(route( name: 'video.show', $video->id));
26
27
28          // Assert the response is successful and contains video details
29          $response->assertStatus( status: 200);
30          $response->assertSee($video->title);
31          $response->assertSee($video->description);
32      }
}
```

En cas de que no funcione donara el error 404.

```
 37  public function test_users_cannot_view_not_existing_videos()
 38  {
 39      // Perform a GET request to a non-existing video's route
 40      $response = $this->get(route( name: 'video.show', parameters: 999));
 41
 42      // Assert the response returns a 404 status code
 43      $response->assertStatus( status: 404);
}
```



Visualització de vídeos (VideosTest.php - Feature)

Aquest és el procés que segueixo per comprovar que els usuaris poden visualitzar vídeos existents i que reben un error 404 quan intenten accedir a vídeos inexistentes.

Pasos TDD (Test-Driven Development):

1. Escric el test (Fail):

- Començo escrivint dos tests: un per assegurar-me que un vídeo es pot visualitzar correctament i un altre per comprovar que no es pot accedir a vídeos inexistentes.
- Aquests tests fallen perquè encara no he implementat la ruta ni la vista corresponents.

2. Escric el codi mínim (Pass):

- Implemento la ruta i la vista necessàries per mostrar els vídeos existents.
- Configuro la ruta perquè retorna un error 404 si un vídeo no existeix.
- Executo els tests fins que tots passin.

3. Refactoritzo:

- Milloro el codi de la ruta i la vista, assegurant-me que sigui modular, clar i fàcil de mantenir en el futur.

Pasos AAA (Arrange-Act-Assert):

1. Arrange:

- Creo un vídeo a la base de dades amb les dades necessàries per al test. Si estic provant el cas d'un vídeo inexistent, em garanteixo que no hi hagi cap entrada a la taula de vídeos.

2. Act:

- Faig una petició GET a la ruta corresponent del vídeo que vull provar.

3. Assert:

- Verifico que la resposta inclou el títol i la descripció del vídeo existent quan aquest és accessible.
- També comprovo que la resposta retorna un error 404 quan intento accedir a un vídeo inexistent.

Aquest procés m'ajuda a garantir que el codi es desenvolupa amb funcionalitats provades i fiables.

Executo el test (VideosTest) de nou i veure que em fa els 4 correctament.

```
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ php artisan test --filter=VideosTest

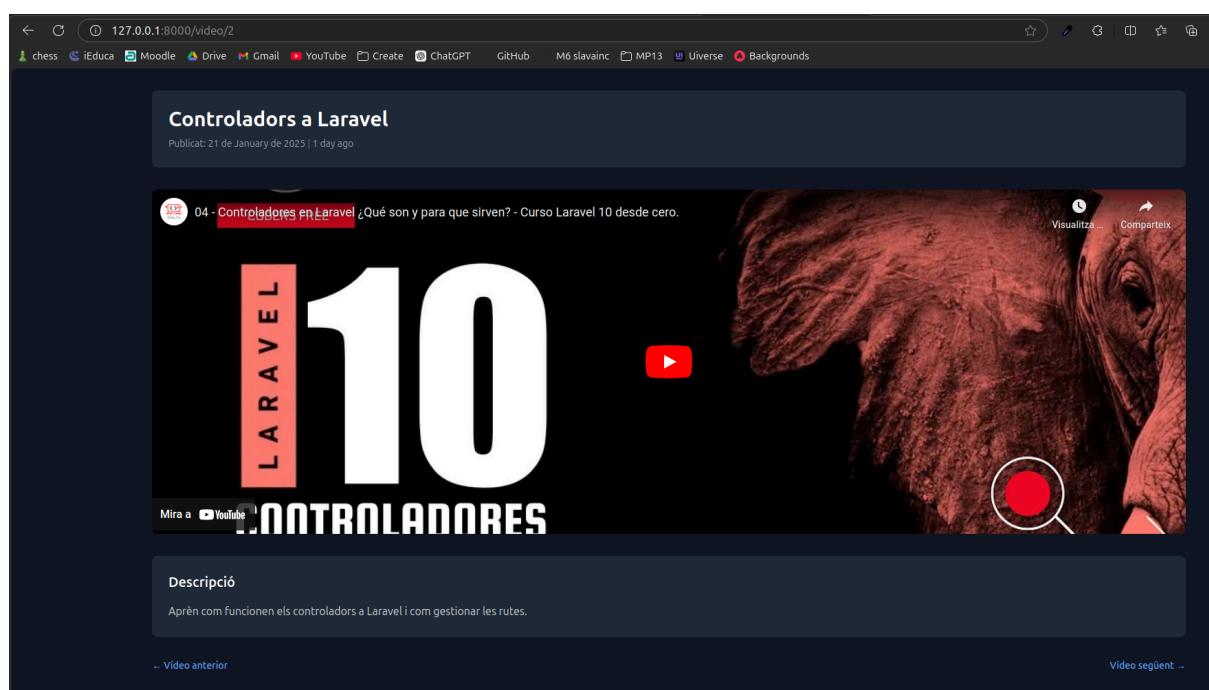
  PASS  Tests\Unit\VideosTest
    ✓ can get formatted published at date
    ✓ can get formatted published at date when not published

  PASS  Tests\Feature\VideosTest
    ✓ users can view videos
    ✓ users cannot view not existing videos

Tests:  4 passed (6 assertions)
Duration: 0.18s

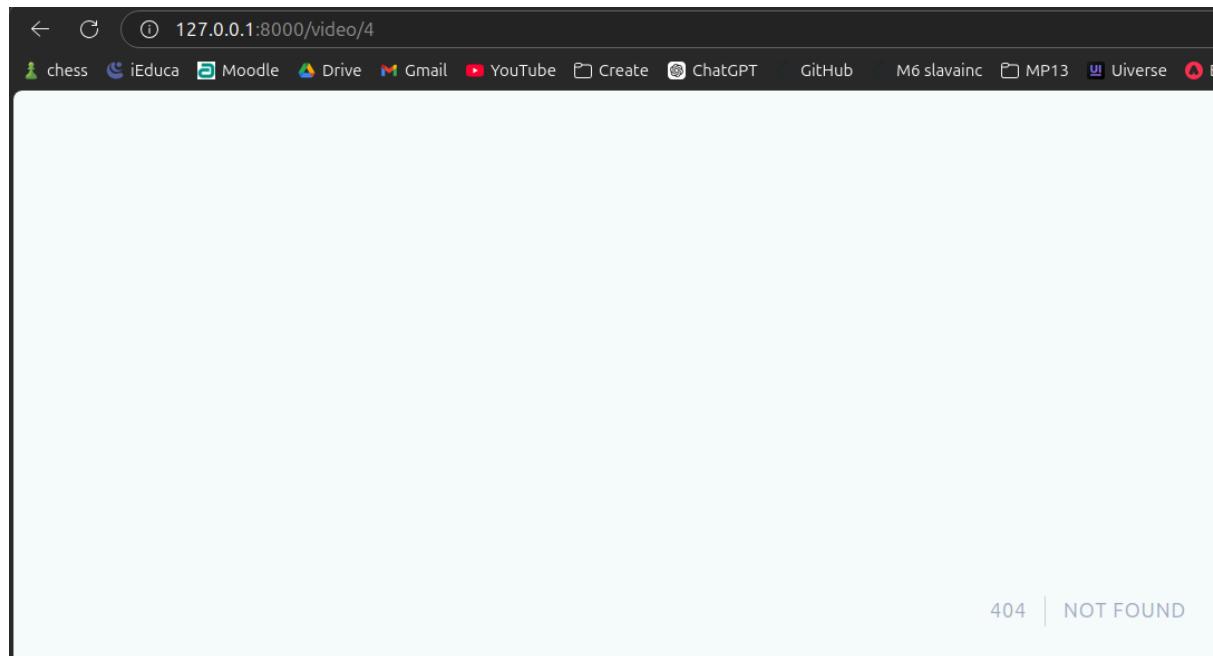
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$
```

Si entro a la pàgina em entra bé.





I si entro amb una id incorrecta o que no existeix ens dona el error 404.



Aquí veurem les peticions.

```
2025-01-22 17:06:25 /video/2 ....  
2025-01-22 17:06:27 /favicon.ico .  
2025-01-22 17:06:29 /video/4 ....  
2025-01-22 17:06:29 /favicon.ico .
```



Crear el terms

Afegirem a resources/markdown/terms una petita guia sobre que tracta el projecte i que he fet als dos sprints.

```
M→ terms.md ×  
1  # Guia del projecte VideosApp  
2  
3  ## Descripció del projecte  
4  **VideosApp** és una aplicació desenvolupada amb Laravel que permet gestionar vídeos educatius.  
5  
6  El projecte està estructurat seguint una metodologia àgil basada en sprints, on cada sprint inclou:  
7  
8  ---  
9  
10 ## Tasques realitzades  
11  
12 ### Sprint 1: Creació inicial del projecte  
13 - **Configuració del projecte**:  
14     - Creació del projecte Laravel amb Jetstream, Livewire, PHPUnit, equips i SQLite.  
15 - **Creació de tests inicials**:  
16     - Test per verificar la creació dels usuaris per defecte (usuari i professor).
```

Guia del projecte VideosApp

Descripció del projecte

VideosApp és una aplicació desenvolupada amb Laravel que permet gestionar vídeos educatius. L'objectiu és oferir una plataforma senzilla però potencialment completa per visualitzar vídeos, amb funcionalitats que inclouen la navegació entre vídeos, la gestió d'usuaris i equips, i una intereficie responsive per a una millor experiència d'usuari.

El projecte està estructurat seguint una metodologia àgil basada en sprints, on cada sprint incorpora funcionalitats específiques al sistema.

Tasques realitzades

Sprint 1: Creació inicial del projecte

- **Configuració del projecte:**
 - Creació del projecte Laravel amb Jetstream, Livewire, PHPUnit, equips i SQLite.
- **Creació de tests inicials:**
 - Test per verificar la creació dels usuaris per defecte (usuari i professor).
- **Helpers:**
 - Creació d'un helper per a la gestió d'usuaris per defecte.

Larastan tests

Instalació

Instal·lem amb la següent comanda

```
alunnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ composer require --dev "larastan/larastan:^3.0"
./composer.json has been updated
Running composer update larastan/larastan
Loading composer repositories with package information
Updating dependencies
Lock file operations: 3 installs, 0 updates, 0 removals
- Locking larastan/larastan (v3.0.2)
- Locking phpmyadmin/sql-parser (5.10.2)
- Locking phpsttan/phpstan (2.1.1)
Writing lock file
Installing dependencies from lock file (including require-dev)
Package operations: 3 installs, 0 updates, 0 removals
- Installing phpsttan/phpstan (2.1.1): Extracting archive
- Installing phpmyadmin/sql-parser (5.10.2): Extracting archive
- Installing Larastan/larastan (v3.0.2): Extracting archive
2 package suggestions were added by new dependencies, use 'composer suggest' to see details.
Generating optimized autoload files
> Illuminate\Foundation\ComposerScripts::postAutoLoadDump
> @php artisan package:discover --ansi
  INFO Discovering packages.

laravel/fortify
laravel/jetstream
laravel/pail
laravel/sail
laravel/sanctum
laravel/tinker
livewire/livewire
nesbot/carbon
numonaduro/collision
numonaduro/termwind
```

Configurem el fitxer per a que comproba les rutes (app, routes i tests), amb un nivell 5 i ignore els 2 errors per defecte que no és poden arreglar.

```
GNU nano 7.2                                     phpsttan.neon
includes:
  - vendor/larastan/larastan/extension.neon
  - vendor/nesbot/carbon/extension.neon

parameters:

paths:
  - app/
  - routes/
  - tests/

# Level 10 is the highest level
level: 5

ignoreErrors:
  # Suppress errors related to accessing undefined properties on Eloquent models
  - '#Access to an undefined property Illuminate\\Database\\Eloquent\\Model::.*#'

  # Suppress errors related to undefined methods on Eloquent models
  - '#Call to an undefined method Illuminate\\Database\\Eloquent\\Model::.*#'
```

Errors:

Executo i començo a arreglar

```
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ ./vendor/bin/phpstan analyse
Note: Using configuration file /home/alumnat/Code/M7-8-9/VideoAppHarvey/phpstan.neon.
56/56 [██████████] 100%
```

Fitxers escanejats.

```
Project ▾
  ▾ VideoAppHarvey [VideoAppHarvey-main] ~/Code/M7-8-9/VideoAppHarvey
    ▾ app
      ▾ Actions
        ▷ Fortify
        ▷ Jetstream
        ▷ Events
      ▷ Helpers
      ▾ Http
        ▷ Controllers
      ▷ Models
      ▷ Policies
      ▷ Providers
      ▷ View
    ▾ routes
      ▷ api.php
      ▷ console.php
      ▷ web.php
    ▾ Unit
      ▷ VideosTest.php
      ▷ ExampleTest.php
      ▷ HelpersTest.php
      ▷ VideosTest.php
```



CreateTeam.php:

El error és un problema amb un return que falta especificar el tipus.

```
Line  app/Actions/Jetstream/CreateTeam.php
-----
35      Method App\Actions\Jetstream\CreateTeam::create() should return
          App\Models\Team but returns Illuminate\Database\Eloquent\Model.
           return.type
-----
```

Error:

```
$user->switchTeam($team = $user->ownedTeams()->create([
    'name' => $input['name'],
    'personal_team' => false,
]));
```

Solució:

Afegirel el tipus a la variable que és retorna.

```
/** @var Team $team */
$team = $user->ownedTeams()->create([
    'name' => $input['name'],
    'personal_team' => false,
]);

$user->switchTeam($team);
```

DeleteUser.php:

El error diu que el paràmetre espera especificar el tipus ja que té una mescla. Ademés del each que no és fa correctament.

```
Line  app/Actions/Jetstream/DeleteUser.php
-----
40  Parameter #1 $callback of method
    Illuminate\Support\Collection<int, Illuminate\Database\Eloquent\Model>::each()
expects callable(Illuminate\Database\Eloquent\Model, int): mixed,
Closure(App\Models\Team): void given.
    argument.type
-----
```

Error:

```
protected function deleteTeams(User $user): void
{
    $user->teams()->detach();

    $user->ownedTeams->each(function (Team $team) {
        $this->deleteTeams->delete($team);
    });
}
```

Solució:

Afegirem la línia del var ademés del each per iterar correctament.

```
// Afegim un PHPDoc per assegurar el tipus
/** @var \Illuminate\Support\Collection<int, Team> $ownedTeams */
$ownedTeams = $user->ownedTeams;

// Iterem sobre els equips
$ownedTeams->each(function (Team $team) {
    $this->deleteTeams->delete($team);
});
```



RemoveTeamMember.php:

El error especifica que no és troba bé el id.

```
Line app/Actions/Jetstream/RemoveTeamMember.php
-----
45 Access to an undefined property Illuminate\Database\Eloquent\Model::$id.
    ➜ property.notFound
💡 Learn more:
    https://phpstan.org/blog/solving-phpstan-access-to-undefined-property
-----
```

Error:

```
protected function ensureUserDoesNotOwnTeam(User $teamMember, Team $team): void
{
    if ($teamMember->id === $team->owner->id) {
        throw ValidationException::withMessages([
            'team' => [__(key: 'You may not leave a team that you created.')],
        ])->errorBag(errorBag: 'removeTeamMember');
}
```

Solució:

Especificare el id correctament.

```
if (! $team->owner instanceof User) {
    throw new \LogicException(message: 'Owner is not a valid User instance.');
}
```



InviteTeamMember.php:

El error diu que la classe no és correcta i el tipus tampoc.

```
Line  app/Actions/Jetstream/InviteTeamMember.php
-----
37      Parameter #1 $invitation of class Laravel\Jetstream\Mail\TeamInvitation
           constructor expects Laravel\Jetstream\TeamInvitation,
           Illuminate\Database\Eloquent\Model given.
           █ argument.type
```

Error:

```
$invitation = $team->teamInvitations()->create([
    'email' => $email,
    'role' => $role,
]);
Mail::to($email)->send(new TeamInvitation($invitation));
```

Solució:

Procurarem importar bé el mòdul, crear la instancia i guardar bé la info.

```
use Laravel\Jetstream\Mail\TeamInvitation; // Importació de la classe Mail de Jetstream
use Laravel\Jetstream\TeamInvitation as JetstreamTeamInvitation; // Importació del model corr

// Crear una instància de TeamInvitation de Jetstream
$invitation = new JetstreamTeamInvitation([
    'email' => $email,
    'role' => $role,
]);

// Guardar la invitació al model TeamInvitation relacionat amb l'equip
$team->teamInvitations()->save($invitation);

// Enviar la invitació per correu
Mail::to($email)->send(new TeamInvitation($invitation));
```



Team.php:

El tipus array no és el adequat per al tipus que se està sobreescrivint.

```
Line   app/Models/Team.php
-----
21     PHPDoc type array<int, string> of property App\Models\Team::$fillable is not
          covariant with PHPDoc type list<string> of overridden property
          Illuminate\Database\Eloquent\Model::$fillable.
          🚫 property.phpDocType
         💡 You can fix 3rd party PHPDoc types with stub files:
          💡 https://phpstan.org/user-guide/stub-files
```

Error:

```
*
* @var array<int, string>
*/
```

Solució:

Canviarem de array a list.

```
*
* @var list<string>
*/
```



User.php:

Hi han errors de array que no és el tipus correcte i un error de foto de perfil que no existeix dins del model.

```
Line    app/Models/User.php
-----
30     PHPDoc type array<int, string> of property App\Models\User::$fillable is not
covariant with PHPDoc type list<string> of overridden property
Illuminate\Database\Eloquent\Model::$fillable.
  📖 property.phpDocType
 💡 You can fix 3rd party PHPDoc types with stub files:
 💡 https://phpstan.org/user-guide/stub-files

41     PHPDoc type array<int, string> of property App\Models\User::$hidden is not
covariant with PHPDoc type list<string> of overridden property
Illuminate\Database\Eloquent\Model::$hidden.
  📖 property.phpDocType
 💡 You can fix 3rd party PHPDoc types with stub files:
 💡 https://phpstan.org/user-guide/stub-files

53     PHPDoc type array<int, string> of property App\Models\User::$appends is not
covariant with PHPDoc type list<string> of overridden property
Illuminate\Database\Eloquent\Model::$appends.
  📖 property.phpDocType
 💡 You can fix 3rd party PHPDoc types with stub files:
 💡 https://phpstan.org/user-guide/stub-files

54     Property 'profile_photo_url' does not exist in model.
  📖 rules.modelAppends
```

Error:

```
*
* @var array<int, string>
*/
```

Solució:

La primera solució del array és reemplaçar amb list.

```
*
* @var list<string>
*/
```

Procurarem cridar bé la foto de perfil.

```
/***
 * Get the profile photo URL.
 */
no usages new
public function getProfilePhotoUrlAttribute(): string
{
    return $this->profile_photo_path
        ? asset( path: 'storage/' . $this->profile_photo_path)
        : $this->defaultProfilePhotoUrl();
}

/***
 * Get the default profile photo URL.
 */
1 usage new
protected function defaultProfilePhotoUrl(): string
{
    return 'https://ui-avatars.com/api/?name=' . urlencode($this->name) . '
```

Actualitzo el composer i ja és arregla tot el tema de foto de perfil.

```
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ composer update
Loading composer repositories with package information
Updating dependencies
Nothing to modify in lock file
Installing dependencies from lock file (including require-dev)
Nothing to install, update or remove
Generating optimized autoload files
> Illuminate\Foundation\ComposerScripts::postAutoloadDump
> @php artisan package:discover --ansi

  [INFO] Discovering packages.

  laravel/fortify ..... DONE
  laravel/jetstream ..... DONE
  laravel/pail ..... DONE
  laravel/sail ..... DONE
  laravel/sanctum ..... DONE
  laravel/tinker ..... DONE
  livewire/livewire ..... DONE
  nesbot/carbon ..... DONE
  nunomaduro/collision ..... DONE
  nunomaduro/termwind ..... DONE
```



console.php:

Hi ha una variable sense definir correctament o que no és crida bé.

```
Line   routes/console.php
-----
7      Undefined variable: $this
      variable.undefined
-----
```

Error:

```
Artisan::command( signature: 'inspire', function () {
    $this->comment(Inspiring::quote());
})->purpose( description: 'Display an inspiring quote')->hourly();
```

Solució:

Imprimirem directe amb un echo per evitar la variable obsoleta.

```
Artisan::command( signature: 'inspire', function () {
    // Imprimir la cita directament amb la funció echo o comandes alternatives
    echo Inspiring::quote();
})->purpose( description: 'Mostrar una cita inspiradora')->hourly();
```



ExampleTest.php:

El errors ens diu algo que és veritat i no podem canviar, true sempre serà true.

```
Line  tests/Unit/ExampleTest.php
-----
14      Call to method PHPUnit\Framework\Assert::assertTrue() with true will always evaluate to true.
      [E] method.alreadyNarrowedType
-----
```

Error:

```
└ Wharvey123
    public function test_that_true_is_true(): void
    {
        $this->assertTrue( condition: true);
    }
}
```

Solució:

Ja que no és un error en si, comentarem el codi.

```
public function test_that_true_is_true(): void
{
    //      $this->assertTrue(true);
}
```

ProfileInformationTest.php:

Hi han 2 errors del estat de les variables que no és troba.

```
Line  tests/Feature/ProfileInformationTest.php
-----
21    Access to an undefined property Livewire\Features\SupportTesting\Testable::$state.
      🔍 property.notFound
      ⚡ Learn more: https://phpstan.org/blog/solving-phpstan-access-to-undefined-property
22    Access to an undefined property Livewire\Features\SupportTesting\Testable::$state.
      🔍 property.notFound
      ⚡ Learn more: https://phpstan.org/blog/solving-phpstan-access-to-undefined-property
-----
```

Error:

```
$this->assertEquals($user->name, $component->state['name']);
$this->assertEquals($user->email, $component->state['email']);
```

Solució:

Eliminarem el estat i les aplicarem directament amb un assertSet.

```
$component = Livewire::test( name: UpdateProfileInformationForm::class)
  ->assertSet( name: 'name', $user->name)
  ->assertSet( name: 'email', $user->email);
```

Analisis final:

```
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ ./vendor/bin/phpstan analyse
Note: Using configuration file /home/alumnat/Code/M7-8-9/VideoAppHarvey/phpstan.neon.
56/56 [██████████] 100%  
  

[OK] No errors  
  

alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$
```