

Laravel

Sprint 1 i 2



Marc Pallares Pino
2nDAM

Professor: Jordi Vega
Institut : [📍 Institut de l'Ebre](#)

INDEX

Instal·lació Laravel	3
Configuració	4
Sprint 2	9
GITHUB PROJECTE	30
WEBGRAFIA	30

Sprint 1

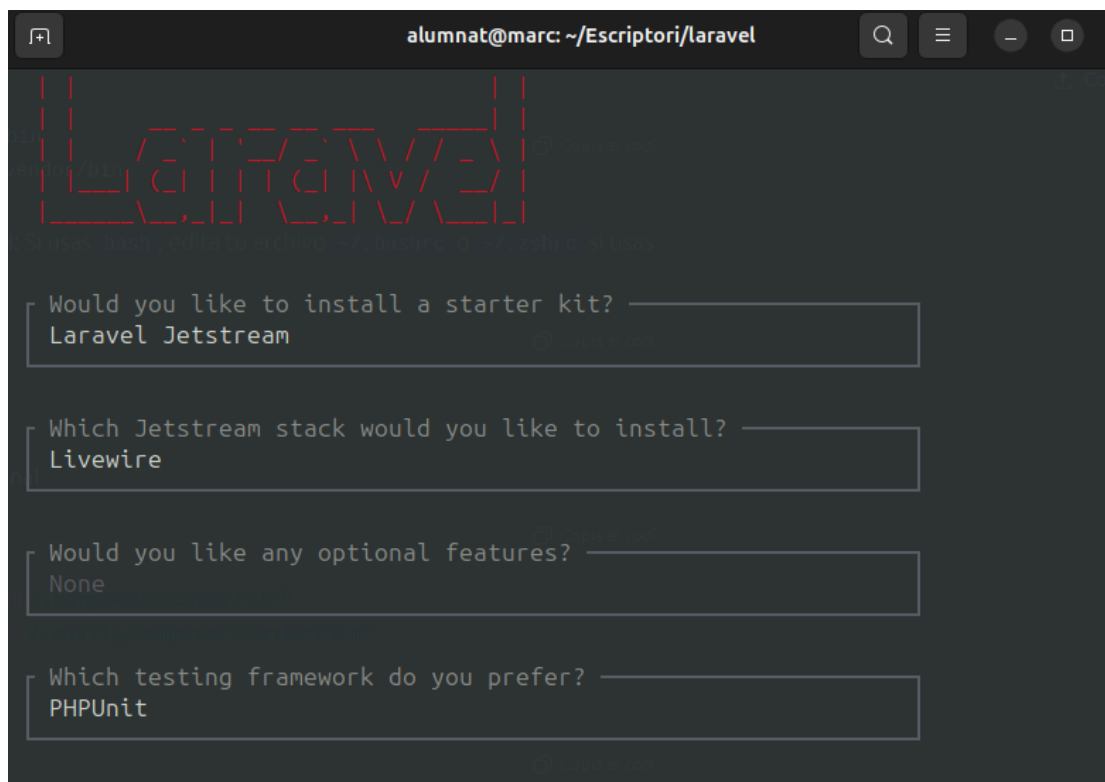
Instal·lació Laravel

Instalem el laravel :

```
/bin/bash -c "$(curl -fsSL https://php.new/install/linux/8.3)"  
composer global require laravel/installer
```

Creem el projecte:

```
laravel new VideosappMarc
```



```
alumnat@marc: ~/Escriptori/laravel
78 packages you are using are looking for funding.
Use the `composer fund` command to find out more!
No security vulnerability advisories found.
> @php -r "file_exists('.env') || copy('.env.example', '.env');"

INFO Application key set successfully.

Which database will your application use? _____
SQLite

Would you like to run the default database migrations? _____
Yes

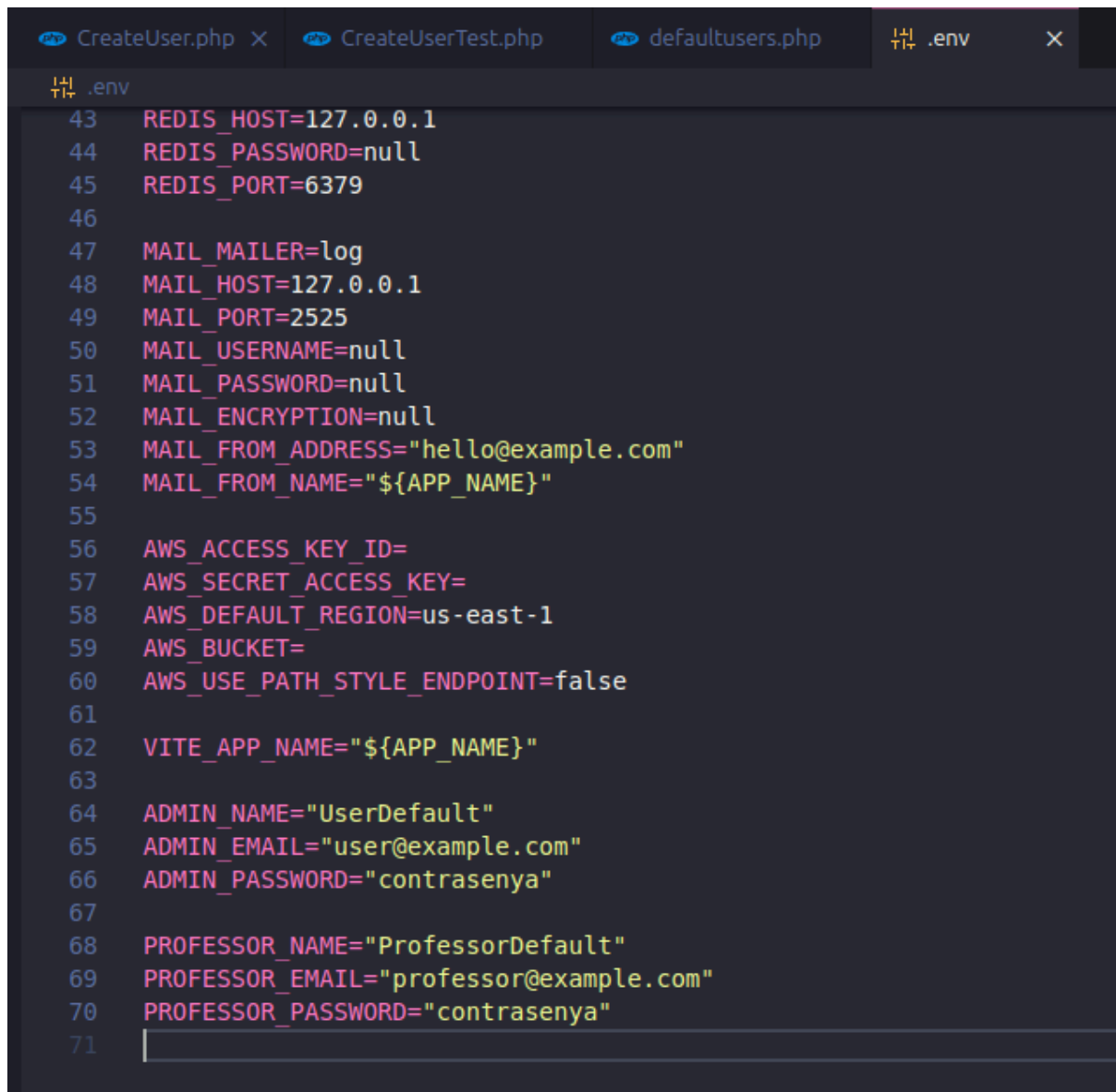
INFO Preparing database.

Creating migration table ..... 4.26ms DONE
```

Configuració

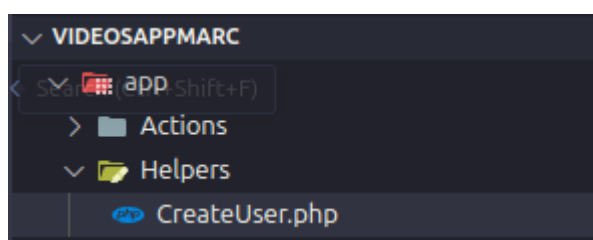
Modifiquem el env





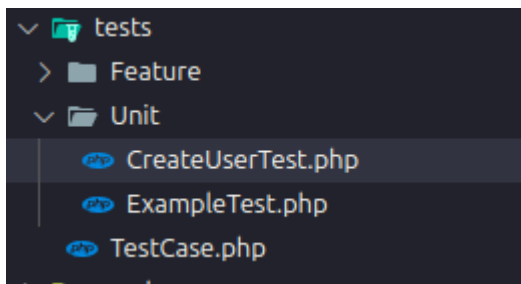
```
43 REDIS_HOST=127.0.0.1
44 REDIS_PASSWORD=null
45 REDIS_PORT=6379
46
47 MAIL_MAILER=log
48 MAIL_HOST=127.0.0.1
49 MAIL_PORT=2525
50 MAIL_USERNAME=null
51 MAIL_PASSWORD=null
52 MAIL_ENCRYPTION=null
53 MAIL_FROM_ADDRESS="hello@example.com"
54 MAIL_FROM_NAME="${APP_NAME}"
55
56 AWS_ACCESS_KEY_ID=
57 AWS_SECRET_ACCESS_KEY=
58 AWS_DEFAULT_REGION=us-east-1
59 AWS_BUCKET=
60 AWS_USE_PATH_STYLE_ENDPOINT=false
61
62 VITE_APP_NAME="${APP_NAME}"
63
64 ADMIN_NAME="UserDefault"
65 ADMIN_EMAIL="user@example.com"
66 ADMIN_PASSWORD="contrasenya"
67
68 PROFESSOR_NAME="ProfessorDefault"
69 PROFESSOR_EMAIL="professor@example.com"
70 PROFESSOR_PASSWORD="contrasenya"
71
```

Creem la carpeta helpers i dintre de ella creem el createUser.php per a fer la funcio per crear l'usuari.



```
CreateUserTest.php x defaultusers.php .env CreateUsers.php x
app > Helpers > CreateUsers.php
1 <?php
2
3 // app/Helpers/CreateUsers.php
4
5 namespace App\Helpers;
6
7 use App\Models\User;
8 use Illuminate\Support\Facades\DB;
9 use Illuminate\Support\Facades\Hash;
10 use Illuminate\Support\Facades\Validator;
11
12 class CreateUsers
13 {
14     public function creacioUsuariDefecte(array $user)
15     {
16         Validator::make($user, [
17             'name' => ['required', 'string', 'max:255'],
18             'email' => ['required', 'string', 'email', 'max:255', 'unique:users,email'],
19             'password' => ['required', 'string', 'min:8'],
20         ])->validate();
21
22         return DB::transaction(function () use ($user) {
23             return tap(User::create([
24                 'name' => $user['name'],
25                 'email' => $user['email'],
26                 'password' => Hash::make($user['password']),
27             ]), function (User $user) {
28                 $this->creacioTeam($user);
29             });
30         });
31     }
32
33     protected function creacioTeam(User $user): void
34     {
35         // Create a team for the user
36         $user->ownedTeams()->create([
37             'name' => "{$user->name}'s Team",
38             'personal_team' => true, // Ensure the personal_team field is set
39         ]);
40     }
41 }
42
```

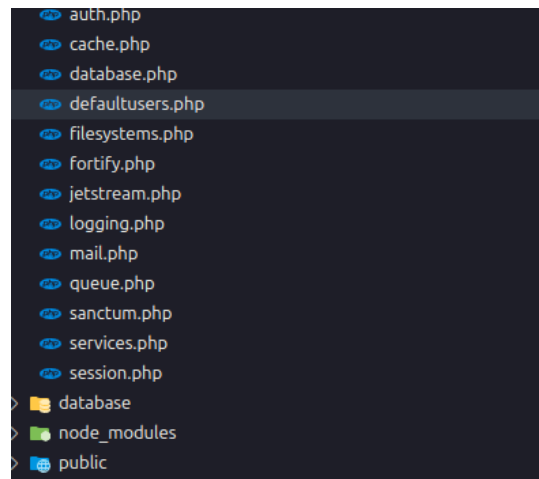
Per crear el test aurem de anar a test , unit i alli dins creem el createusertest.php , per poder comprovar que funciona correctament.



```
CreateUser.php  CreateUserTest.php X  defaultusers.php  .env

tests > Unit > CreateUserTest.php
1  <?php
2
3
4  namespace Tests\Unit;
5
6  use App\Helpers\CreateUsers;
7  use Illuminate\Foundation\Testing\RefreshDatabase;
8  use Tests\TestCase;
9
10 class CreateUserTest extends TestCase
11 {
12     use RefreshDatabase;
13
14     public function testCreacioUsuariDefecte()
15     {
16         $user = (new CreateUsers)->creacioUsuariDefecte([
17             'name' => config('defaultusers.user.name'),
18             'email' => config('defaultusers.user.email'),
19             'password' => config('defaultusers.user.password'),
20         ]);
21
22         $this->assertEquals(config('defaultusers.user.name'), $user->name);
23         $this->assertEquals(config('defaultusers.user.email'), $user->email);
24         $this->assertTrue(\Hash::check(config('defaultusers.user.password'), $user->password));
25         $this->assertCount(1, $user->ownedTeams);
26     }
27
28     public function testCreacioProfessorDefecte()
29     {
30         $user = (new CreateUsers)->creacioUsuariDefecte([
31             'name' => config('defaultusers.professor.name'),
32             'email' => config('defaultusers.professor.email'),
33             'password' => config('defaultusers.professor.password'),
34         ]);
35
36         $this->assertEquals(config('defaultusers.professor.name'), $user->name);
37         $this->assertEquals(config('defaultusers.professor.email'), $user->email);
38         $this->assertTrue(\Hash::check(config('defaultusers.professor.password'), $user->password));
39         $this->assertCount(1, $user->ownedTeams);
40     }
41 }
42
43 // Comprovar que la contra esta encriptada
44 //$this->assertNotEquals('contra', $user->password);
45 //$this->assertNotEquals('contra', $teacher->password);
46
```

creem el config.php per ficar els paràmetres per defecte dels usuaris.



```
config > defaultusers.php
1  <?php
2
3  return [
4      'user' => [
5          'name' => env("USER_NAME", "UserDefault"),
6          'email' => env("USER_EMAIL", "user@example.com"),
7          'password' => env("USER_PASSWORD", "contra1234"),
8      ],
9      'professor' => [
10         'name' => env("PROFESSOR_NAME", "ProfessorDefault"),
11         'email' => env("PROFESSOR_EMAIL", "professor@example.com"),
12         'password' => env("USER_PASSWORD", "contra1234"),
13     ],
14
15
16     'team' => [
17         'name_suffix' => "'s Team",
18     ],
19 ];
20
```


comprovació del funcionament.

```
alumnat@marc:~/Escriptori/laravel/VideosappMarc$ php artisan test
```

```
PASS Tests\Unit\CreateUserTest
```

```
✓ creacio usuari defecte
```

```
✓ creacio professor defecte
```

Sprint 2

Descripció del 2n Sprint:

- Corregir els errors del 1r sprint.

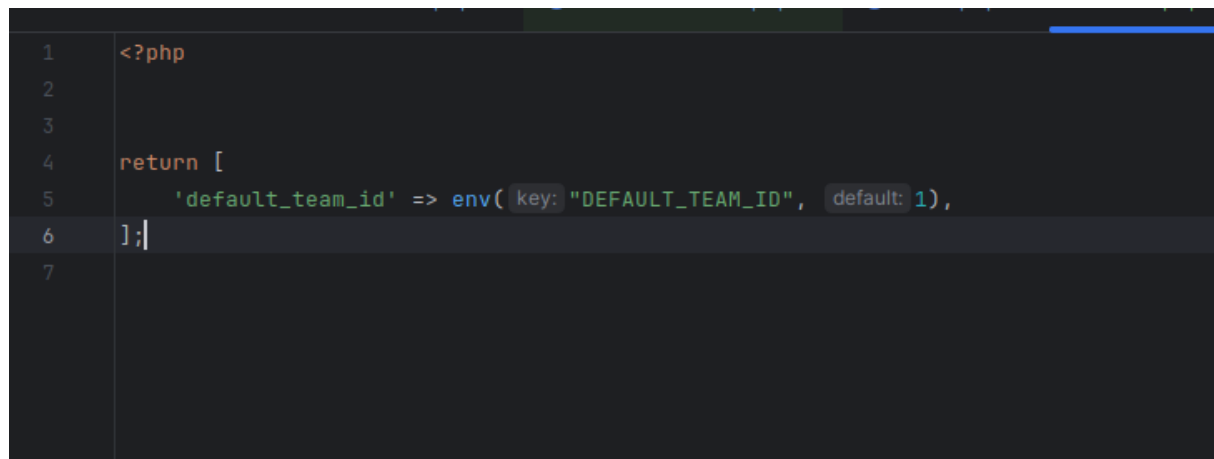
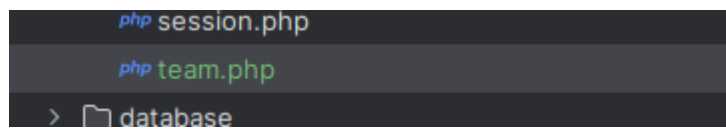
```
1 <?php
2
3 return [
4     'user' => [
5         'name' => env( key: "USER_NAME", default: "UserDefault"),
6         'email' => env( key: "USER_EMAIL", default: "user@example.com"),
7         'password' => env( key: "USER_PASSWORD", default: "contra1234"),
8         'current_team_id' => env( key: "USER_TEAM_ID", default: null),
9     ],
10    'professor' => [
11        'name' => env( key: "PROFESSOR_NAME", default: "ProfessorDefault"),
12        'email' => env( key: "PROFESSOR_EMAIL", default: "professor@example.com"),
13        'password' => env( key: "USER_PASSWORD", default: "contra1234"),
14        'current_team_id' => env( key: "PROFESSOR_TEAM_ID", default: null),
15    ],
16    'team' => [
17        'name_suffix' => "'s Team",
18    ],
19 ];
```

fiquem el test

```
new *
public function testUserHasCurrentTeamId()
{
    $user = User::factory()->create([
        'current_team_id' => config( key: 'team.default_team_id'),
    ]);

    $this->assertNotNull($user->current_team_id);
    $this->assertEquals(config( key: 'team.default_team_id'), $user->current_team_id);
}
```

creem el config



- Al fitxer phpunit descomentar les línies de db_connection i db_database per tal de que els tests utilitzi una base de dades temporal i no afecti a la base de dades real.

Descomentem el db_connection i db_database

```
M+ README.md  php defaultusers.php  CreateUserTest.php  phpunit.xml x
1  <?xml version="1.0" encoding="UTF-8"?>
2  <phpunit xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3      xsi:noNamespaceSchemaLocation="vendor/phpunit/phpunit/phpunit.x
4      bootstrap="vendor/autoload.php"
5      colors="true"
6  >
7      <testsuites>
8          <testsuite name="Unit">
9              <directory>tests/Unit</directory>
10         </testsuite>
11         <testsuite name="Feature">
12             <directory>tests/Feature</directory>
13         </testsuite>
14     </testsuites>
15     <source>
16         <include>
17             <directory>app</directory>
18         </include>
19     </source>
20     <php>
21         <env name="APP_ENV" value="testing"/>
22         <env name="APP_MAINTENANCE_DRIVER" value="file"/>
23         <env name="BCRYPT_ROUNDS" value="4"/>
24         <env name="CACHE_STORE" value="array"/>
25         <env name="DB_CONNECTION" value="sqlite"/> |
26         <env name="DB_DATABASE" value=":memory:"/>
27         <env name="MAIL_MAILER" value="array"/>
28         <env name="PULSE_ENABLED" value="false"/>
29         <env name="QUEUE_CONNECTION" value="sync"/>
30         <env name="SESSION_DRIVER" value="array"/>
31         <env name="TELESCOPE_ENABLED" value="false"/>
32     </php>
33 </phpunit>
34
```

- Crear la migració de videos amb els camps (id, title, description, url, published_at, previous, next, series_id). Per als inserts dels videos a la url podeu agafar una url d'un video de youtube.

Creem la migracio

```
^Calumnat@marc:~/Escriptori/mp09/laravelmarc/VideosappMarc$ php artisan make:migration create_videos_table

INFO Migration [database/migrations/2025_01_23_152042_create_videos_table.php] created successfully.
```

fiquem el codi

```
CreateUserTest.php  php 2025_01_23_152042_create_videos_table.php x
7 class CreateVideosTable extends Migration
9     /**
10      * Run the migrations.
11      *
12      * @return void
13      */
14     public function up()
15     {
16         Schema::create('videos', function (Blueprint $table) {
17             $table->id();
18             $table->string('title');
19             $table->text('description')->nullable();
20             $table->string('url');
21             $table->timestamp('published_at')->nullable();
22             $table->unsignedBigInteger('previous')->nullable();
23             $table->unsignedBigInteger('next')->nullable();
24             $table->unsignedBigInteger('series_id');
25             $table->timestamps();
26
27             $table->foreign('previous')->references('id')->on('table: 'videos')->onDelete('set null');
28             $table->foreign('next')->references('id')->on('table: 'videos')->onDelete('set null');
29             $table->foreign('series_id')->references('id')->on('table: 'series')->onDelete('cascade');
30         });
31     }
32
33     /**
```

comanda per migra

```
alumnat@marc:~/Escriptori/mp09/laravelmarc/VideosappMarc$ php artisan migrate

APPLICATION IN PRODUCTION.
```

Controlador de videos, VideosController amb les funcions testedBy i el show.

creem el videoscontroller

```
Terminal Local x + v
alumnat@marc:~/Escriptori/mp09/laravelmarc/VideosappMarc$ php artisan make:controller VideosController

INFO Controller [app/Http/Controllers/VideosController.php] created successfully.

alumnat@marc:~/Escriptori/mp09/laravelmarc/VideosappMarc$
```

codi:

```
© VideosController.php x
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use App\Models\Video;
6 use Illuminate\Http\Request;
7
8 no usages
9 class VideosController extends Controller
10 {
11     /**
12      * Muestra el video especificado.
13      *
14      * @param int $id
15      * @return \Illuminate\Http\Response
16      */
17     public function show($id)
18     {
19         $video = Video::findOrFail($id);
20         return view('videos.show', compact('video'));
21     }
22
23     /**
24      * Muestra los videos probados por un usuario específico.
25      *
26      * @param int $userId
27      * @return \Illuminate\Http\Response
28      */
29     no usages
30     public function testedBy($userId)
31     {
32         $videos = Video::where('tested_by', $userId)->get();
33         return view('videos.tested_by', compact('videos'));
34     }
35 }
```

- Model de videos, el camp `published_at` ha de ser una data. Ha de tindre les següents funcions per a retornar la data en diferents formats que siguin llegibles a per a l'usuari: `getFormattedPublishedAtAttribute` (retorna una data tipus "13 de gener de 2025"), `getFormattedForHumansPublishedAtAttribute` (retorna una data tipus "fa 2 hores") i `getPublishedAtTimestampAttribute` (retorna el valor Unix timestamp de `published_at`). S'ha d'utilitzar la llibreria Carbon per a manipular les dates i hores.

Creem el model video.php

```
alumnat@marc:~/Escriptori/mp09/laravelmarc/VideosappMarc$ php artisan make:model Video

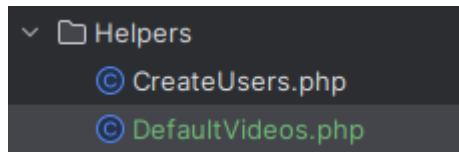
INFO Model [app/Models/Video.php] created successfully.

alumnat@marc:~/Escriptori/mp09/laravelmarc/VideosappMarc$
```

codi:

```
Video.php x
9 class Video extends Model
10 {
11     use HasFactory;
12
13     no usages
14     protected $dates = ['published_at'];
15
16     /**
17      * Retorna la data en format "13 de gener de 2025".
18      *
19      * @return string
20      */
21     no usages
22     public function getFormattedPublishedAtAttribute()
23     {
24         return Carbon::parse($this->published_at)->translatedFormat('d \d\e F \d\e Y');
25     }
26
27     /**
28      * Retorna la data en format "fa 2 hores".
29      *
30      * @return string
31      */
32     no usages
33     public function getFormattedForHumansPublishedAtAttribute()
34     {
35         return Carbon::parse($this->published_at)->diffForHumans();
36     }
37
38     /**
39      * Retorna el valor Unix timestamp de published_at.
40      *
41      * @return int
42      */
43     no usages
44     public function getPublishedAtTimestampAttribute()
45     {
46         return Carbon::parse($this->published_at)->timestamp;
47     }
48 }
```

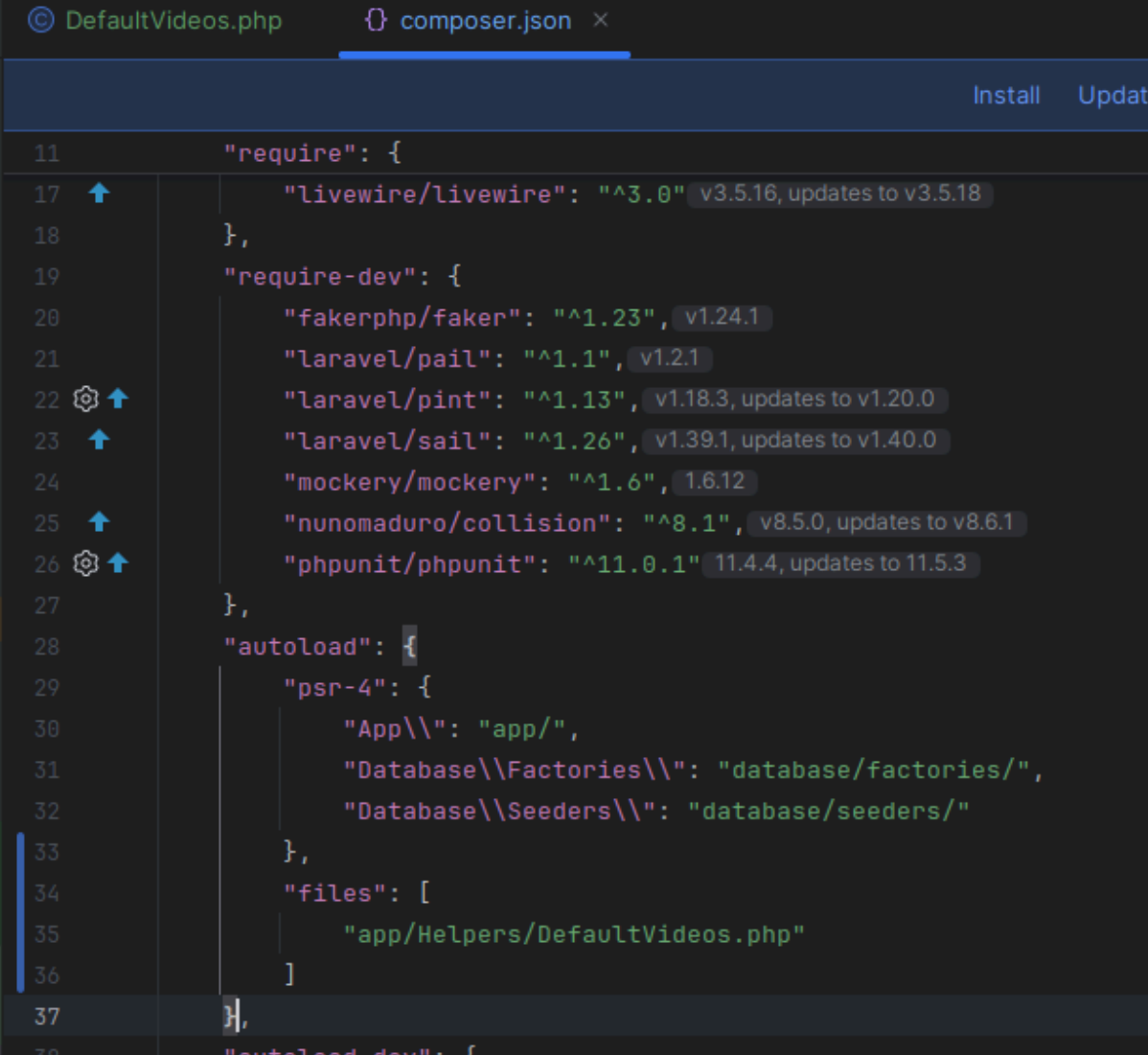
- Crear un helper de default videos.
- creem el helper defaultvideos.php



codi:

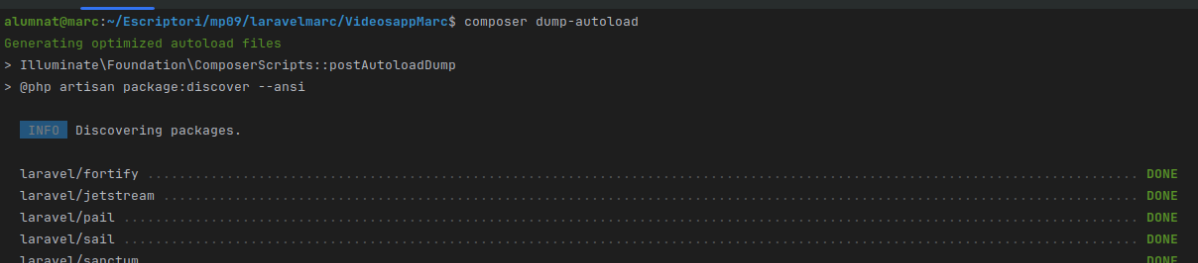
```
1  <?php
2
3  namespace App\Helpers;
4
5  use App\Models\Video;
6  use Carbon\Carbon;
7
8  no usages new *
9  class DefaultVideos
10 {
11     /**
12      * Crea un video por defecto.
13      *
14      * @param array $attributes
15      * @return \App\Models\Video
16      */
17     no usages new *
18     public static function createDefaultVideo(array $attributes = [])
19     {
20         return Video::create(array_merge([
21             'title' => 'Default Title',
22             'description' => 'Default Description',
23             'url' => 'https://www.youtube.com/watch?v=default',
24             'published_at' => Carbon::now(),
25             'previous' => null,
26             'next' => null,
27             'series_id' => 1,
28         ], $attributes));
29     }
30 }
```


ho afegim al json compose



```
11     "require": {
17         "livewire/livewire": "^3.0" v3.5.16, updates to v3.5.18
18     },
19     "require-dev": {
20         "fakerphp/faker": "^1.23", v1.24.1
21         "laravel/pail": "^1.1", v1.2.1
22         "laravel/pint": "^1.13", v1.18.3, updates to v1.20.0
23         "laravel/sail": "^1.26", v1.39.1, updates to v1.40.0
24         "mockery/mockery": "^1.6", 1.6.12
25         "nunomaduro/collision": "^8.1", v8.5.0, updates to v8.6.1
26         "phpunit/phpunit": "^11.0.1" 11.4.4, updates to 11.5.3
27     },
28     "autoload": {
29         "psr-4": {
30             "App\\": "app/",
31             "Database\\Factories\\": "database/factories/",
32             "Database\\Seeders\\": "database/seeders/"
33         },
34         "files": [
35             "app/Helpers/DefaultVideos.php"
36         ]
37     },
38     "autoload-dev": {
```

Per a que reconeixi el helper



```
alumnat@marc:~/Escriptori/mp09/laravelmarc/VideosappMarc$ composer dump-autoload
Generating optimized autoload files
> Illuminate\Foundation\ComposerScripts::postAutoloadDump
> @php artisan package:discover --ansi

INFO Discovering packages.

laravel/fontify ..... DONE
laravel/jetstream ..... DONE
laravel/pail ..... DONE
laravel/sail ..... DONE
laravel/sanctum ..... DONE
laravel/tinker ..... DONE
```

- Posar els usuaris i videos per defecte al DatabaseSeeder.

afegir als seeders de usuarios i videos

```
Terminal Local x + v
alumnat@marc:~/Escriptori/mp09/laravelmarc/VideosappMarc$ php artisan make:seeder UserSeeder

[INFO] Seeder [database/seeders/UserSeeder.php] created successfully.

alumnat@marc:~/Escriptori/mp09/laravelmarc/VideosappMarc$ php artisan make:seeder VideoSeeder

[INFO] Seeder [database/seeders/VideoSeeder.php] created successfully.

alumnat@marc:~/Escriptori/mp09/laravelmarc/VideosappMarc$
```

userSeeder

```
DefaultVideos.php composer.json DatabaseSeeder.php UserSeeder.php x VideoSeeder.p
1 <?php
2
3 namespace Database\Seeders;
4
5 use Illuminate\Database\Seeder;
6 use App\Models\User;
7
8 1 usage
9 class UserSeeder extends Seeder
10 {
11     public function run()
12     {
13         User::create([
14             'name' => 'Default User',
15             'email' => 'default@example.com',
16             'password' => bcrypt( value: 'password'),
17         ]);
18     }
19 }
```

videosseeder

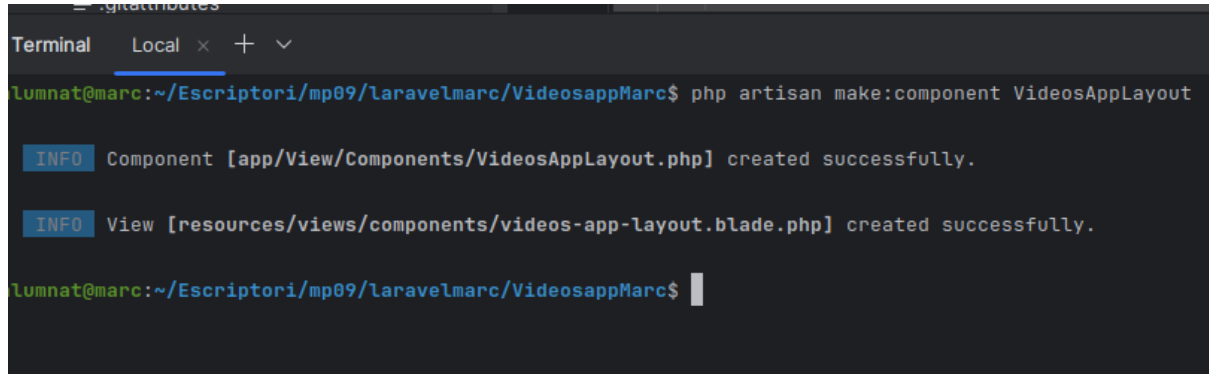
```
© DefaultVideos.php  {} composer.json  © DatabaseSeeder.php  © UserSeeder.php  © VideoSeeder.php x
1  <?php
2
3  namespace Database\Seeders;
4
5  use Illuminate\Database\Seeder;
6  use App\Helpers\DefaultVideos;
7
8  ! usage
9  class VideoSeeder extends Seeder
10 {
11     public function run()
12     {
13         DefaultVideos::createDefaultVideo();
14     }
15 }
```

databaseseeder

```
© DefaultVideos.php  {} composer.json  © DatabaseSeeder.php x  © UserSeeder.php  © VideoSeeder.php
1  <?php
2
3  namespace Database\Seeders;
4
5  use Illuminate\Database\Seeder;
6
7  no usages  ⚙ «Marc» *
8  class DatabaseSeeder extends Seeder
9  {
10     /**
11      * Seed the application's database.
12      */
13     ⚙ «Marc» *
14     public function run(): void
15     {
16         $this->call([
17             UserSeeder::class,
18             VideoSeeder::class,
19         ]);
20     }
21 }
```

- S'ha de crear un layout que es digui VideosAppLayout, ha d'estar tant a app/View/components com a resources/views/layouts.

Creem el layout i la carpeta



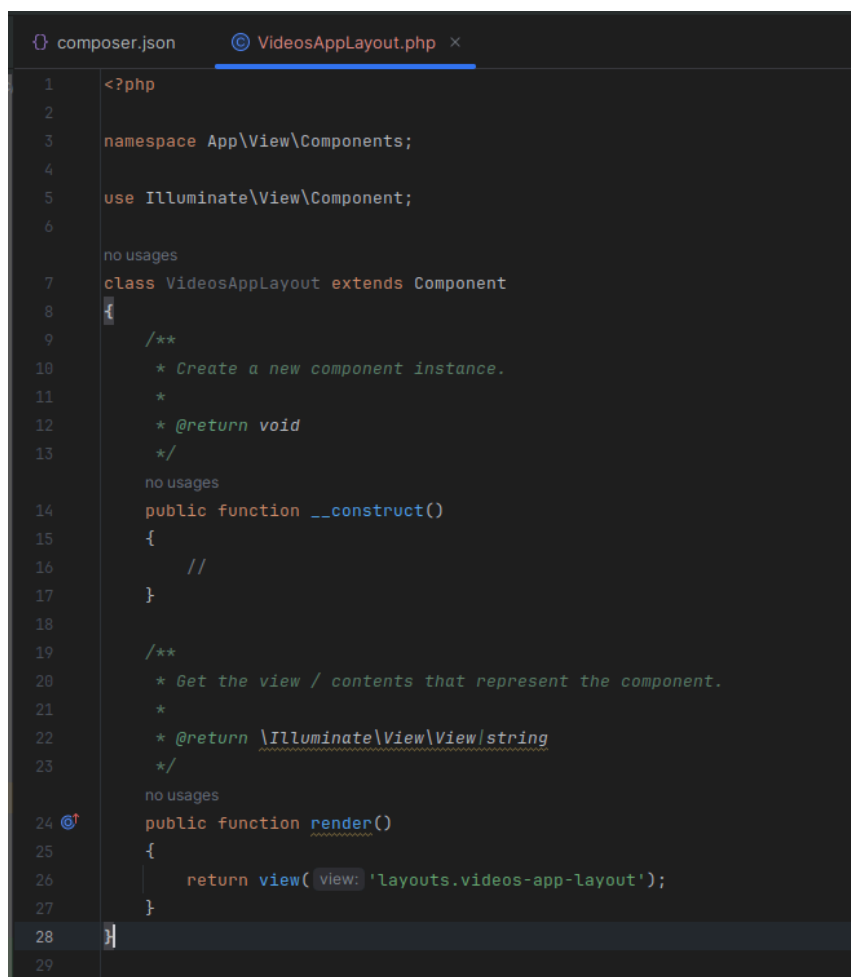
```
Terminal Local x + v
lumnat@marc:~/Escriptori/mp09/laravelmarc/VideosappMarc$ php artisan make:component VideosAppLayout

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

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

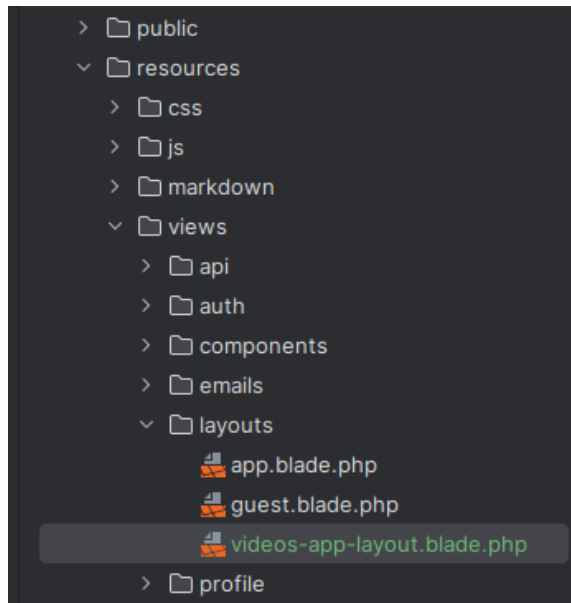
lumnat@marc:~/Escriptori/mp09/laravelmarc/VideosappMarc$
```

codi:



```
composer.json  VideosAppLayout.php x
1  <?php
2
3  namespace App\View\Components;
4
5  use Illuminate\View\Component;
6
7  no usages
8  class VideosAppLayout extends Component
9  {
10     /**
11      * Create a new component instance.
12      *
13      * @return void
14      */
15     no usages
16     public function __construct()
17     {
18         //
19     }
20
21     /**
22      * Get the view / contents that represent the component.
23      *
24      * @return \Illuminate\View\View|string
25      */
26     no usages
27     public function render()
28     {
29         return view('layouts.videos-app-layout');
```

estructura:



codi:

```
composer.json  VideosAppLayout.php  videos-app-layout.blade.php x
1  <!-- resources/views/layouts/videos-app-layout.blade.php -->
2  <!DOCTYPE html>
3  <html lang="en">
4  <head>
5      <meta charset="UTF-8">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0">
7      <title>Videos App</title>
8      <!-- Añade tus enlaces a CSS aquí -->
9  </head>
10 <body>
11 <header>
12     <!-- Contenido del header -->
13 </header>
14
15 <main>
16     {{ $slot }}
17 </main>
18
19 <footer>
20     <!-- Contenido del footer -->
21 </footer>
22 </body>
23 </html>
24
```

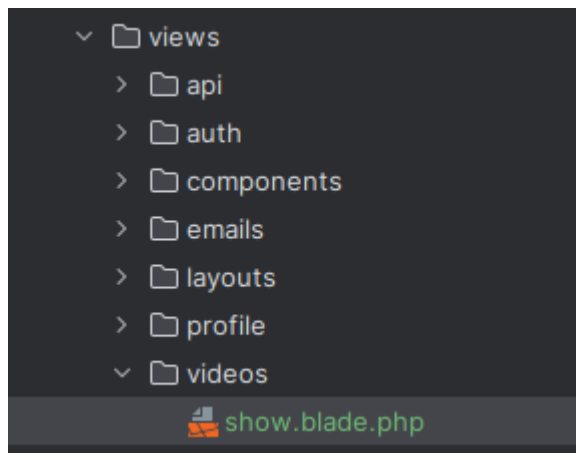
- S'ha de crear la ruta del show de videos.

afegim a web.php la ruta

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

```
© VideosController.php x show.blade.php php web.php
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use App\Models\Video;
6 use Illuminate\Http\Request;
7
8 2 usages 1 «Marc»
9 class VideosController extends Controller
10 {
11     /**
12      * Display the specified video.
13      *
14      * @param int $id
15      * @return \Illuminate\Http\Response
16      */
17     1 «Marc»
18     public function show($id)
19     {
20         $video = Video::findOrFail($id);
21         return view(view: 'videos.show', compact(var_name: 'video'));
22     }
23
24     /**
25      * Display the videos tested by a specific user.
26      *
27      * @param int $userId
28      * @return \Illuminate\Http\Response
29      */
30     no usages 1 «Marc»
31     public function testedBy($userId)
32     {
33         $videos = Video::where('tested_by', $userId)->get();
34         return view(view: 'videos.tested_by', compact(var_name: 'videos'));
35     }
36 }
37
```

Estructura:

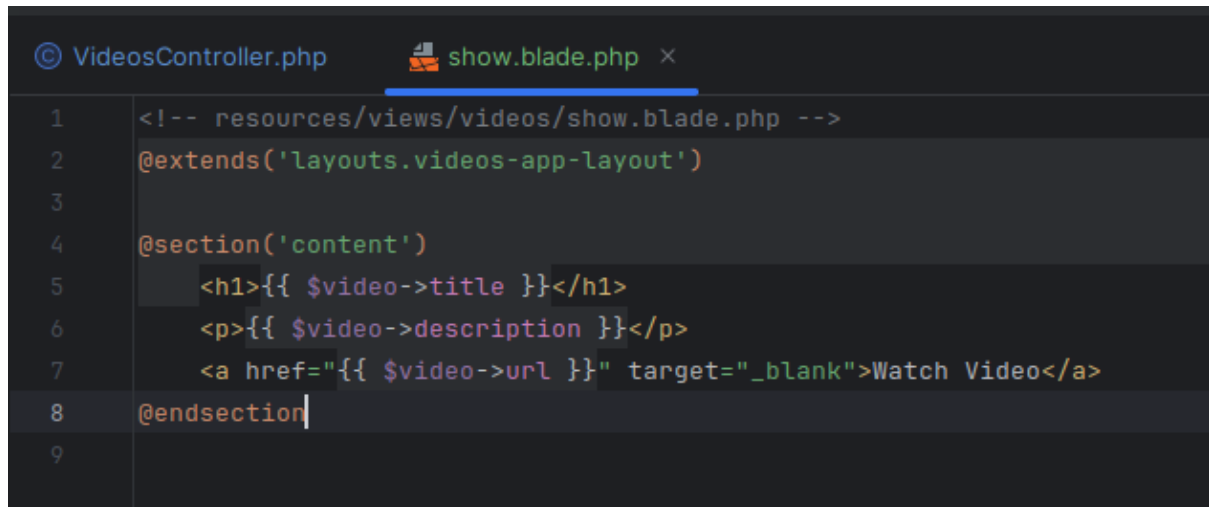


codi:

```
© VideosController.php  show.blade.php ×  php web.php
1  <!-- resources/views/videos/show.blade.php -->
2  @extends('layouts.videos-app-layout')
3
4  @section('content')
5      <h1>{{ $video->title }}</h1>
6      <p>{{ $video->description }}</p>
7      <a href="{{ $video->url }}" target="_blank">Watch Video</a>
8  @endsection
9
```

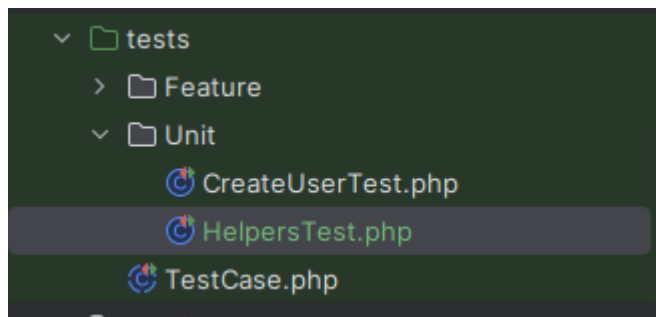
- S'ha de crear la vista del show de videos.

codi de vista:



```
© VideosController.php  show.blade.php x
1 <!-- resources/views/videos/show.blade.php -->
2 @extends('layouts.videos-app-layout')
3
4 @section('content')
5     <h1>{{ $video->title }}</h1>
6     <p>{{ $video->description }}</p>
7     <a href="{{ $video->url }}" target="_blank">Watch Video</a>
8 @endsection
9
```

- Posar el test HelpersTest a la carpeta tests/Unit.
- creem el helperstest a la carpeta unit , el podiem pasar si el tinguem a una altra carpeta per una comanda.



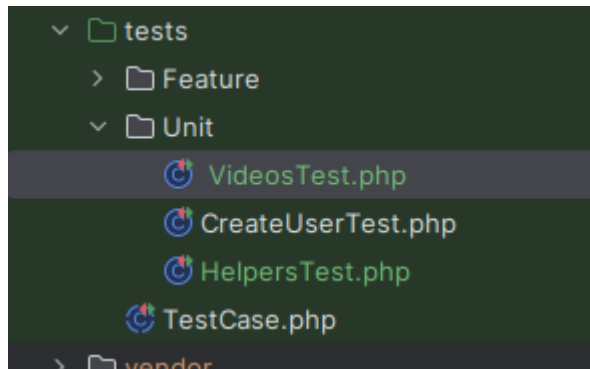
- Afegir el test de la creació dels videos per defecte al fitxer HelpersTest

codi:

```
© VideosController.php  show.blade.php  HelpersTest.php x
1  <?php
2
3  namespace Tests\Unit;
4
5  use App\Models\Video;
6  use PHPUnit\Framework\TestCase;
7
8  new *
9  class HelpersTest extends TestCase
10 {
11     // Otros métodos de prueba
12
13     /** @test */
14     new *
15     public function it_creates_default_video()
16     {
17         // Crear un video por defecto
18         $video = Video::create([
19             'title' => 'Default Title',
20             'description' => 'Default Description',
21             'url' => 'http://example.com/default-video'
22         ]);
23
24         // Aserciones para verificar que el video se creó correctamente
25         $this->assertInstanceOf( expected: Video::class, $video);
26         $this->assertEquals( expected: 'Default Title', $video->title);
27         $this->assertEquals( expected: 'Default Description', $video->description);
28         $this->assertEquals( expected: 'http://example.com/default-video', $video->url);
29     }
30 }
```

- Crear el test VideosTest a la carpeta tests/Unit. 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().

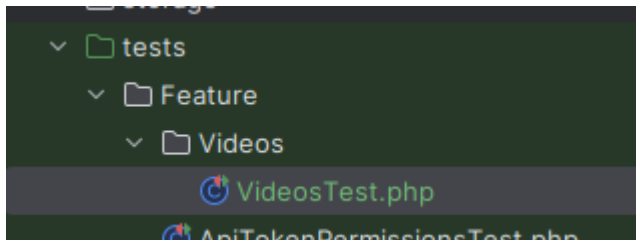
Creem el test



codi:

```
© VideosController.php  show.blade.php  VideosTest.php x
1  <?php
2
3  namespace Tests\Unit;
4
5  use App\Models\Video;
6  use Carbon\Carbon;
7  use PHPUnit\Framework\TestCase;
8
9  new *
10 class VideosTest extends TestCase
11 {
12     /** @test */
13     new *
14     public function can_get_formatted_published_at_date()
15     {
16         // Crear un video con una fecha de publicación
17         $video = new Video(['published_at' => Carbon::parse('2023-01-01 12:00:00')]);
18
19         // Verificar que la fecha de publicación se formatea correctamente
20         $this->assertEquals('01/01/2023', $video->formatted_published_at);
21     }
22
23     /** @test */
24     new *
25     public function can_get_formatted_published_at_date_when_not_published()
26     {
27         // Crear un video sin fecha de publicación
28         $video = new Video(['published_at' => null]);
29
30         // Verificar que la fecha de publicación se formatea correctamente cuando no está publicada
31         $this->assertEquals('No publicado', $video->formatted_published_at);
32     }
33 }
```

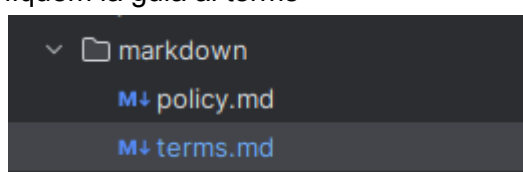
- Crear el test VideosTest a la carpeta tests/Feature/Videos. S'han de crear les funcions `users_can_view_videos()` i `users_cannot_view_not_existing_videos()`.



```
© VideosController.php  show.blade.php  VideosTest.php  VideosTest.php x
1  <?php
2
3  namespace Tests\Feature\Videos;
4
5  use App\Models\Video;
6  use Illuminate\Foundation\Testing\RefreshDatabase;
7  use Tests\TestCase;
8
9  new *
10 class VideosTest extends TestCase
11 {
12     use RefreshDatabase;
13
14     /** @test */
15     new *
16     public function users_can_view_videos()
17     {
18         // Crear un video
19         $video = Video::factory()->create();
20
21         // Hacer una solicitud GET a la ruta del video
22         $response = $this->get(route('videos.show', $video->id));
23
24         // Verificar que la respuesta es exitosa y contiene los datos del video
25         $response->assertStatus(200);
26         $response->assertSee($video->title);
27         $response->assertSee($video->description);
28     }
29
30     /** @test */
31     new *
32     public function users_cannot_view_not_existing_videos()
33     {
34         // Hacer una solicitud GET a una ruta de video inexistente
35         $response = $this->get(route('videos.show', parameters: 999));
36
37         // Verificar que la respuesta es un error 404
38         $response->assertStatus(404);
39     }
40 }
```

• Afegir a resources/markdown/terms una petita guia sobre que tracta el projecte i que heu fet als dos sprints.

fiquem la guia al terms



codi i vista

VideosController.php

show.blade.php

VideosTest.php

VideosTest.php

M+ terms.md

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

Guia del Projecte

Descripció del Projecte

Aquest projecte és una plataforma de vídeos que permet als usuaris pujar,

Sprint 1

En el primer sprint, ens vam centrar en establir la base del projecte:

- Configuració de l'entorn de desenvolupament.

- Creació de l'estructura inicial del projecte.

- Implementació de l'autenticació d'usuaris.

- Creació de les migracions i models per als vídeos.

- Desenvolupament de les primeres vistes per a la gestió de vídeos.

Sprint 2

En el segon sprint, vam ampliar les funcionalitats del projecte:

- Implementació de la pujada i emmagatzematge de vídeos.

- Creació de les vistes per a la visualització de vídeos.

- Desenvolupament de proves unitàries i funcionals per assegurar la quali

- Optimització del rendiment i millores en la interfície d'usuari.

Guia del Projecte

Descripció del Projecte

Aquest projecte és una plataforma de vídeos que permet als usuaris pujar, veure i gestionar vídeos. Els usuaris poden crear comptes, iniciar sessió, i accedir a una varietat de vídeos organitzats per categories.

Sprint 1

En el primer sprint, ens vam centrar en establir la base del projecte:

- Configuració de l'entorn de desenvolupament.
- Creació de l'estructura inicial del projecte.
- Implementació de l'autenticació d'usuaris.
- Creació de les migracions i models per als vídeos.
- Desenvolupament de les primeres vistes per a la gestió de vídeos.

Sprint 2

En el segon sprint, vam ampliar les funcionalitats del projecte:

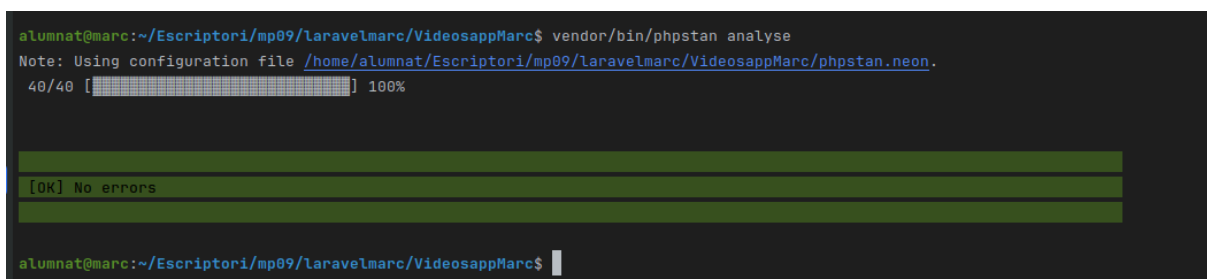
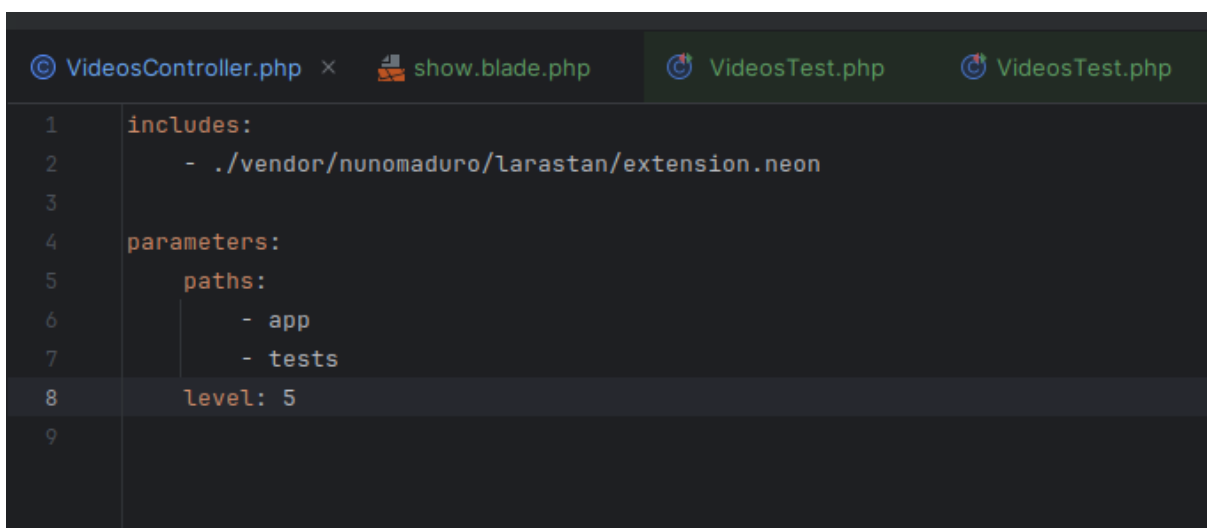
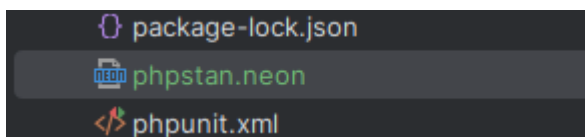
- Implementació de la pujada i emmagatzematge de vídeos.
- Creació de les vistes per a la visualització de vídeos.
- Desenvolupament de proves unitàries i funcionals per assegurar la qualitat del codi.
- Optimització del rendiment i millores en la interfície d'usuari.

- S'ha d'instal·lar i configurar el paquet Larastan, que és una eina per analitzar i detectar el codi en busca d'errors. Una vegada configurat, s'ha d'analitzar el codi en busca d'errors i corregir-los.

instalelem

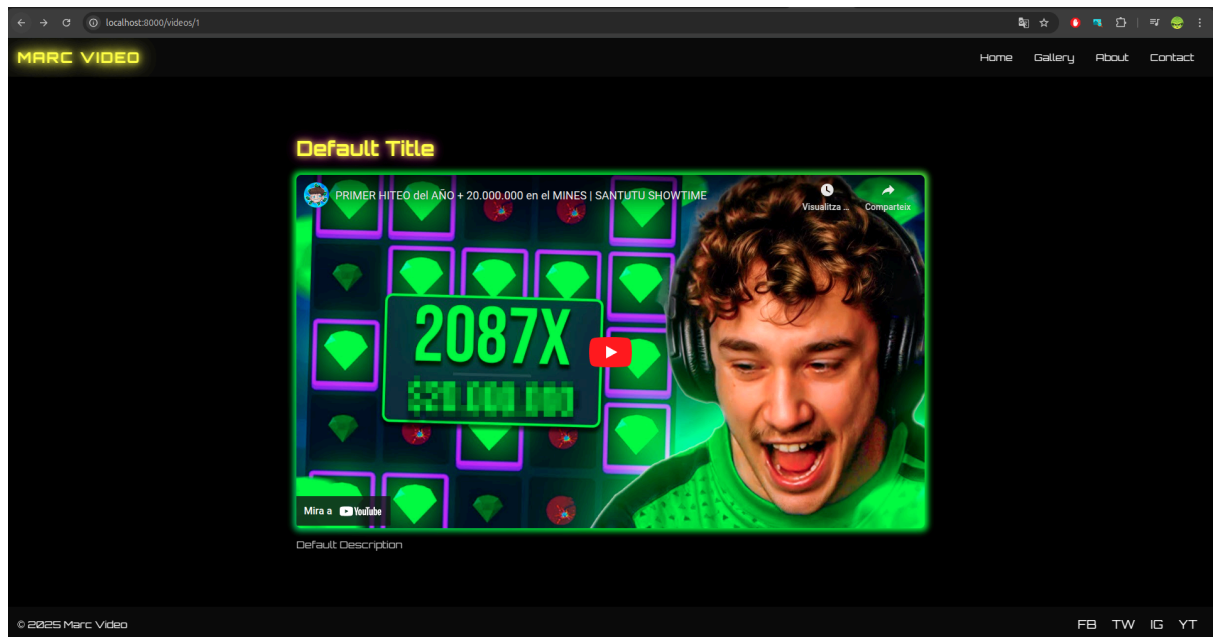
```
alumnat@marc:~/Escriptori/mp09/laravelmarc/VideosappMarc$ composer require --dev nunomaduro/larastan
./composer.json has been updated
Running composer update nunomaduro/larastan
Loading composer repositories with package information
Updating dependencies
Lock file operations: 3 installs, 0 updates, 0 removals
```

Creem el arxiu phpstan.neon i li instalelem els pluguins



🐞 cap error , tenia 2 que els he solucionat era simplement un problema amb una paraula que estava mal escrita

final



GITHUB PROYECTO

<https://github.com/Mpallares1/laravelmarc>

WEBGRAFIA

<https://laravel.com/docs>