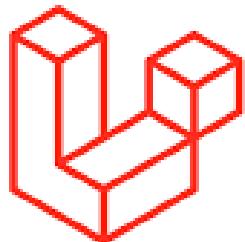


Projecte GLOBAL: VideosApp

6e SPRINT



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



INDEX:

Corregir els errors del 5e sprint.

3

Corregir els errors del 5e sprint.

Retroacció

Qualificació	10,00 / 10,00
Qualificat el	dijous, 27 de març 2025, 15:48
Qualificat per	JV Jordi Vega Tomàs [PROF]
Comentaris de retroalimentació	<p>+ Molt bon treball. Al crear l'usuari fa falta crear el team de l'usuari. Estaria bé que es pogués assignar els permisos al crear i editar ...</p>

Procuro tenir els 3 rols per assignar.

	id	name	guard_name
1	1	superadmin	web
2	2	regular	web
3	3	videomanager	web

UserHelper.php

Assigno bé els rols a cada ususari.

```
// Comprovar si el rol 'regular' existeix, si no, crear-lo
$regularRole = Role::firstOrCreate(['name' => 'regular']);

// Assignar el rol de 'regular' al user
$user->assignRole($regularRole);
```

UserManagerController

Procuro crear, editar i guardar bé la info.

```
// Mostra el formulari per crear un nou usuari
// Wharvey123 *
public function create(): View|Factory|Application
{
    if (! Gate::allows(ability: 'manage-users')) {
        abort( code: 403);
    }

    $roles = Role::all(); // Afegir aquesta línia

    return view( view: 'users.manage.create', compact( var_name: 'roles')); // Passar $roles a la vista
}

public function store(Request $request): RedirectResponse
{
    if (! Gate::allows(ability: 'manage-users')) {
        abort( code: 403);
    }

    $validated = $request->validate([
        'name' => 'required|string|max:255',
        'email' => 'required|email|unique:users',
        'password' => 'required|min:6',
        'roles' => 'array'
    ]);

    // Crear l'usuari
    $user = User::create($validated);

    // Crear equip personal automàticament
    $team = $user->createPersonalTeam();
    $user->current_team_id = $team->id;
    $user->save();

    // Assignar rols
    $user->syncRoles($request->input( key: 'roles', []));

    return redirect()->route( route: 'users.manage.index')->with('success', 'Usuari creat correctament.');
}

// Mostra el formulari d'edició per un usuari concret
// Wharvey123 *
public function edit($id): View|Factory|Application
{
    if (! Gate::allows(ability: 'manage-users')) {
        abort( code: 403);
    }

    $user = User::with( relations: 'currentTeam')->findOrFail($id);
    $teams = Team::all(); // Obtenir tots els equips
    $roles = Role::all(); // Obtenir tots els rols

    return view( view: 'users.manage.edit', compact( var_name: 'user', ...var_names: 'teams', 'roles'));
}
```

User.php

Les relacions adequades.

```
// Relació amb l'equip actual
new *
public function currentTeam(): BelongsTo
{
    return $this->belongsTo(related: Team::class, foreignKey: 'current_team_id');
}

// Relació amb tots els equips de l'usuari
new *
public function teams(): BelongsToMany
{
    return $this->belongsToMany(related: Team::class);
}
```

UserFactory.php

Al factory configuro el següent.

```
public function configure(): Factory|UserFactory
{
    return $this->afterCreating(function (User $user) {
        $user->createPersonalTeam();
    });
}
```

create.blade.php i edit.blade.php

```
<!-- Desplegable de Rols -->


<label for="role" class="block text-gray-200 font-semibold">Rol</label>
    <div class="relative">
        <select name="role" id="role" class="w-full p-3 bg-gray-800 text-white border border-gray-600 rounded-lg appearance-none">
            @foreach($roles as $role)
                <option value="{{ $role->name }}" {{ strtolower($role->name) === 'regular' ? 'selected' : '' }}>
                    {{ $role->name }}
                </option>
            @endforeach
        </select>
        <!-- Dropdown arrow -->
        <div class="pointer-events-none absolute inset-y-0 right-0 flex items-center pr-3 text-gray-400">
            <svg class="fill-current h-4 w-4" xmlns="http://www.w3.org/2000/svg" viewBox="0 0 20 20">
                <path d="M5.516 7.548l4.484 4.482 4.482-4.482"/>
            </svg>
        </div>
    </div>
</div>


```

index.blade.php

```
<td class="px-6 py-4 text-gray-800">
    @foreach($user->roles as $role)
        @php
            // Assignem colors segons el nom del rol
            $colorClass = 'bg-gray-500';
            if(strtolower($role->name) == 'regular'){
                $colorClass = 'bg-blue-500';
            } elseif(strtolower($role->name) == 'videomanager'){
                $colorClass = 'bg-green-500';
            } elseif(strtolower($role->name) == 'superadmin'){
                $colorClass = 'bg-red-500';
            }
        @endphp
        <span class="inline-block {{ $colorClass }} text-white text-xs font-semibold mr-2 px-3 py-1 rounded-tl rounded-br">
            {{ $role->name }}
        </span>
    @endforeach
</td>
```

 INSTITUT DE L'EBRE TORTOSA	GLOBAL M7, M8 i M9	DAM 2n
--	---------------------------	-------------------------

Vista final amb els canvis del Sprint 5 arreglats.

Gestió d'usuaris				
ID	Nom	Email	Rols	Accions
1	alumne	alumne@gmail.com	regular	Editar Eliminar
2	harvey	harvey@gmail.com	superadmin	Editar Eliminar
3	Super Admin	superadmin@videosapp.com	superadmin	Editar Eliminar
4	Regular	regular@videosapp.com	regular	Editar Eliminar
5	Video Manager	videosmanager@videosapp.com	videomanager	Editar Eliminar

Al crear o editar un usuari és mostra un desplegable per triar un rol.

Crear Usuari

Nom

Email

Contrasenya

Rol
 ▼

[Crear Usuari](#) [Tornar](#)

Editar Usuari

Nom

Email

Nova Contrasenya (deixa en blanc per no canviar)

Equip
 ▼

Rol
 ▼

[Actualitzar Usuari](#) [Tornar](#)

Arreglant tests de sprints previs

En cas que al modificar el codi falla algun test d'un sprint anterior, s'han d'arreglar.

Procuro establir tots els rols adequats per cada usuari als testos.

```
public function user_with_permissions_can_store_users()
{
    // Arrange: Autenticar com a superadmin (té permisos per emmagatzemar usuaris)
    $this->loginAsSuperAdmin();

    $newUserData = [
        'name' => 'Nou Usuari',
        'email' => 'nou@example.com',
        'password' => 'password',
        'roles' => ['superadmin']
    ];
}
```

Mels pasa sense problemes.

```
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ php artisan test tests/Feature/Users/UsersManageControllerTest.php

  PASS Tests\Feature\Users\UsersManageControllerTest
    ✓ login as video manager
    ✓ login as super admin
    ✓ login as regular user
    ✓ user with permissions can see add users
    ✓ user without users manage create cannot see add users
    ✓ user with permissions can store users
    ✓ user without permissions cannot store users
    ✓ user with permissions can destroy users
    ✓ user without permissions cannot destroy users
    ✓ user with permissions can see edit users
    ✓ user without permissions cannot see edit users
    ✓ user with permissions can update users
    ✓ user without permissions cannot update users
    ✓ user with permissions can manage users
    ✓ regular users cannot manage users
    ✓ guest users cannot manage users
    ✓ superadmins can manage users

  Tests:  17 passed (70 assertions)
  Duration: 0.81s
```

Modificar videos per poder assignar el video a les sèries.

A les vistes procuro especificar el camp de seleccionar la sèrie

```
<option value="">No pertany a una sèrie</option>
@foreach($series as $serie)
    <option value="{{ $serie->id }}" {{ (isset($seriesId) && $seriesId == $serie->id) ? 'selected' : '' }}>{{ $serie->title }}</option>
@endforeach
<select>
```

Faig la relació al model.

```
public function serie(): BelongsTo
{
    return $this->belongsTo(related: Serie::class, foreignKey: 'series_id');
```

Als controladors aplicare el mateix canvi

```
public function create(Request $request): View\Factory\Application
{
    if (!auth()->user()->can(abilities: 'videos.create')) {
        abort(code: 403, message: 'Unauthorized');
    }
    // Obtenir totes les sèries disponibles per el selector
    $series = Serie::all();
    // Recollir l'id de la sèrie preseleccionada (si s'ha passat per paràmetre)
    $seriesId = $request->query(key: 'series_id');
    return view(view: 'videos.manage.create', compact(var_name: 'series', ...var_names: 'seriesId'));
}
```

Els regular users han de poder crear vídeos.

Per tant a VideoController afegirem les funcions del crud per als regular. A la vista de vídeos s'haurà de crear els botons per al crud.

Al index posaré el boto

```
@section('content')
    <div class="container mx-auto px-4">
        <!-- Títol principal -->
        <h1 class="text-3xl font-bold text-white my-6">VideosApp</h1>
        @auth
            @can('videos.create')
                <div class="mb-6">
                    <a href="{{ route('videos.manage.create') }}">
                        class="bg-blue-500 text-white px-4 py-2 rounded hover:bg-blue-700 transition-colors">
                            Afegir Nou Video
                    </a>
                </div>
            @endcan
        @endauth
```

Modifico el store al controlador

```
public function store(Request $request): RedirectResponse
{
    if (!auth()->user()->can(abilities: 'videos.create')) {
        abort(code: 403, message: 'Unauthorized');
    }

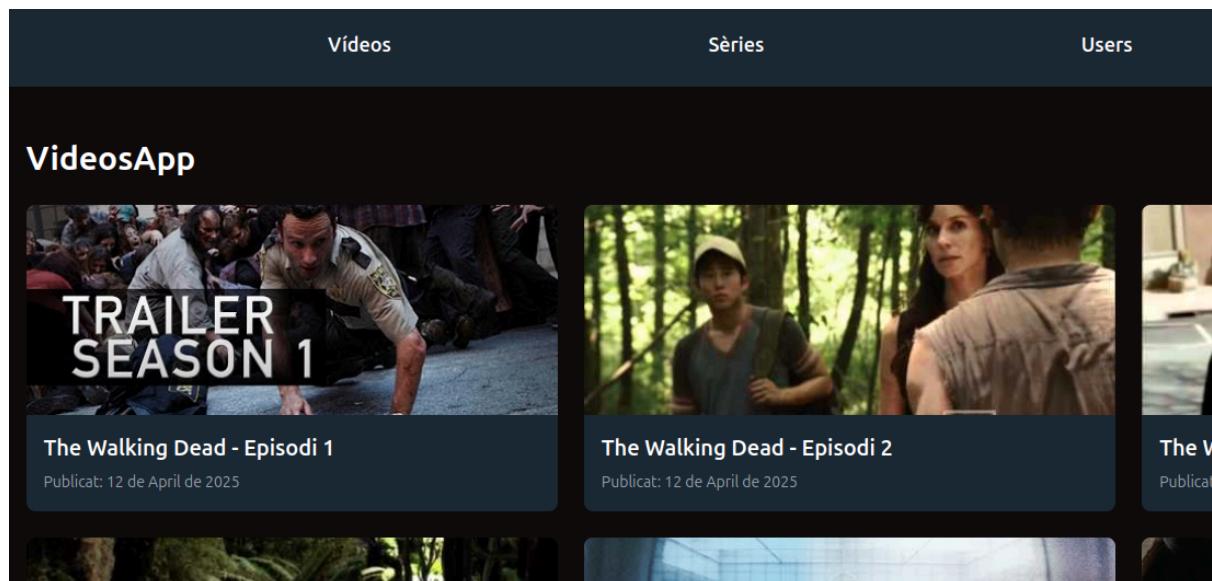
    $data = $request->validate([
        'title'      => 'required|string|max:255',
        'description' => 'required|string',
        'url'         => 'required|url',
        'published_at' => 'nullable|date',
        'previous'    => 'nullable|integer',
        'next'        => 'nullable|integer',
        'series_id'   => 'nullable|integer|exists:series,id',
    ]);

    $data['published_at'] = now();
    $data['user_id'] = auth()->id();

    // Create video and store reference for redirection
    $video = Video::create($data);

    // Redirect to the public show page for the created video
    return redirect()->route(route: 'video.show', $video->id)
        ->with('success', 'Video creat correctament.');
}
```

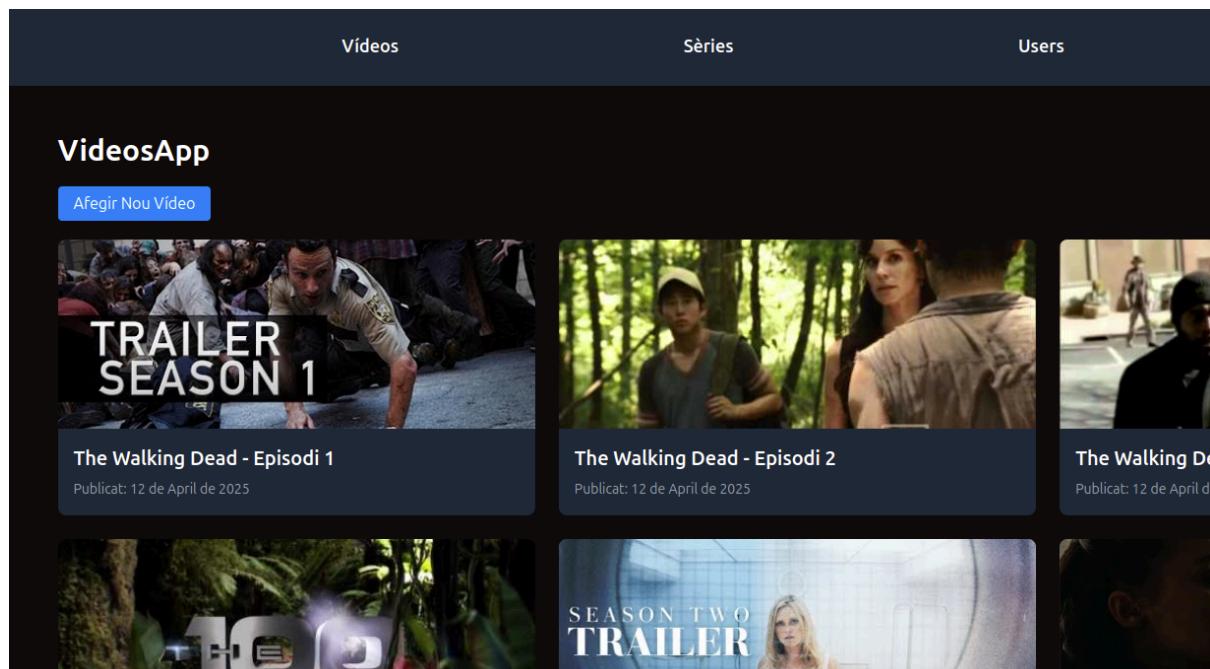
Vista previa de logejar



Entro amb regular.

The login form is contained within a light gray rounded rectangle. It features a blue header bar with the word 'Email' in white. Below it is a text input field with a blue border containing the email address 'regular@videosapp.com'. Underneath is a password input field with a blue border and a redacted content placeholder. To the left of the password field is the text 'Password'. Below the password field is a 'Remember me' checkbox followed by the text 'Remember me'. At the bottom left is a link 'Forgot your password?'. On the bottom right is a dark blue 'LOG IN' button with white text.

Apareix el botó.



Ja puc crear!

Afegir Nou Vídeo

Títol

Descripció

URL

Assigna a una sèrie

Tornar Enrere

Desar Vídeo

Migració de series

Crear migració series. Ha de tindre els camps id, title, description, image, user_name, user_photo_url, published_at.

```
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ php artisan make:migration create_series_table
[INFO] Migration [database/migrations/2025_04_10_073650_create_series_table.php] created successfully.

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

```
php 2025_04_10_073650_create_series_table.php ×
1  <?php
2
3  > use ...
4
5
6
7  return new class extends Migration {
8      /**
9       * Executa les migracions.
10      */
11     public function up(): void
12     {
13         Schema::create('series', function (Blueprint $table) {
14             $table->id();
15             $table->string('title');
16             $table->text('description');
17             $table->string('image')->nullable();
18             $table->string('user_name');
19             $table->string('user_photo_url')->nullable();
20             $table->timestamp('published_at')->nullable();
21             $table->timestamps();
22         });
23     }
24
25     /**
26      * Desa les migracions.
27      */
28     public function down(): void
29     {
30         Schema::dropIfExists('series');
31     }
32 };
```

Model de series

Crear el model de series. Ha de tindre les funcions testedby, videos (per fer la relació 1:N), getFormattedCreatedAtAttribute, getFormattedForHumansCreatedAtAttribute, getCreatedAtTimestampAttribute.

Crear el model, posaré els fillables i les funcions que necesite.

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

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

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

Estableixo els camps, i les funcions que requereixo.

```
Serie.php ×
1 namespace App\Models;
2
3 use ...;
4
5
6 class Serie extends Model
7 {
8     use HasFactory;
9
10    // Tell Laravel which factory to use.
11    no usages
12
13    protected static function newFactory(): SeriesFactory
14    { return SeriesFactory::new(); }
15
16    no usages
17    protected $fillable = ['title', 'description', 'image', 'user_name', 'user_photo_url', 'published_at'];
18
19    no usages
20    public function testedBy(): string
21    { return self::class . 'Test'; }
22
23    public function videos(): HasMany
24    { return $this->hasMany(related: Video::class, foreignKey: 'series_id'); }
25
26    no usages
27    public function getFormattedCreatedAtAttribute(): ?string
28    { return $this->created_at ? Carbon::parse($this->created_at)->isoFormat('D [de] MMMM [de] YYYY') : null; }
29
30    no usages
31    public function getFormattedForHumansCreatedAtAttribute(): ?string
32    { return $this->created_at ? Carbon::parse($this->created_at)->diffForHumans() : null; }
33
34    no usages
35    public function getCreatedAtTimestampAttribute(): ?int
36    { return $this->created_at ? Carbon::parse($this->created_at)->timestamp : null; }
}
```

Al model de vídeos afegire la relació 1:N.

```
© Video.php ×  
35 class Video extends Model  
79  
80     /**  
81      * Relació 1:N amb la Serie.  
82      */  
83      no usages new *  
84      public function serie(): BelongsTo  
85      {  
86          return $this->belongsTo( related: Serie::class, foreignKey: 'series_id');  
87      }  
88  }
```

SeriesManageController

Crear SeriesManageController en les funcions testedby, index, store, edit, update, delete i destroy.

Crear la migració de la serie ha de tenir els camps, id, title, description, image (nullable), user_name, user_photo_url (nullable), published_at (nullable).

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

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

Aquest controlador gestiona la creació, edició, actualització i eliminació de sèries, assegurant la validació de dades i la gestió dels vídeos associats.

```
② SeriesManageController.php ×
10 class SeriesManageController extends Controller
11
12     public function index(): View
13     {
14         $series = Serie::all();
15         return view('series.manage.index', compact('var_name: 'series'));
16     }
17
18     public function create(): View
19     {
20         return view('series.manage.create');
21     }
22
23     public function store(Request $request): RedirectResponse
24     {
25         // Validar els camps obligatoris
26         $data = $request->validate([
27             'title'      => 'required|string|max:255',
28             'description' => 'required|string',
29             'image'       => 'required|url',
30         ]);
31
32         // Assignar la data actual de publicació
33         $data['published_at'] = now();
34
35         // Assignar la informació de l'usuari: nom i foto de perfil
36         // Assegura't que el camp utilitzat coincideixi amb el nom definit al model d'usuari.
37         $data['user_name'] = auth()->check() ? auth()->user()->name : 'Usuari Desconegut';
38         $data['user_photo_url'] = auth()->check() ? auth()->user()->profile_photo_url : null;
39     }
40
41 }
```

SeriesController

Crear SeriesController en les funcions index i show.

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

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

Aquest controlador gestiona la visualització de sèries, permetent la cerca per títol i mostrant els detalls d'una sèrie amb els seus vídeos associats.

```
② SeriesController.php ×
1  <?php
2
3  namespace App\Http\Controllers;
4
5  > use ...
6
7  3 usages
8
9  class SeriesController extends Controller
10 {
11     public function index(Request $request): View
12     {
13         // Si hi ha paràmetre de cerca, filtra per títol
14         $query = Serie::query();
15         if ($request->filled('key: search')) {
16             $query->where('column: title', 'operator: LIKE', 'value: %' . $request->search . '%');
17         }
18         $series = $query->get();
19
20         return view('series.index', compact('var_name: series'));
21     }
22
23     public function show($id): View
24     {
25         $serie = Serie::with('videos')->findOrFail($id);
26         return view('series.show', compact('var_name: serie'));
27     }
28 }
```

SeriesHelper

A helpers crear la funció `create_series()` ha d'haver 3 sèries.

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

Aquest ajudant crea automàticament tres sèries per defecte, assegurant que es registrin correctament a la base de dades si no existeixen prèviament.

```
SeriesHelper.php ×
  7   class SeriesHelper
11   /*
12   * Usage
13   * public static function create_series(): void
14   *
15   * $defaultSeries = [
16   *     // Sèrie 1: The Walking Dead
17   *     [
18   *         'title'          => 'The Walking Dead',
19   *         'description'    => 'Una sèrie post-apocalíptica que segueix un grup de supervivents en un món ple de zombis.',
20   *         // Portada de The Walking Dead:
21   *         'image'          => 'https://cdn.hobbyconsolas.com/sites/navi.axelspringer.es/public/media/image/2018/02/walking-dead-8.jpg?tf=3840x',
22   *         'user_name'       => 'Rick Grimes',
23   *         'user_photo_url' => 'https://hips.hearstapps.com/hmg-prod/images/the-walking-dead-rick-grimes-regreso-teoria-temporada-10-1553704149.jp
24   *         'published_at'   => now()
25   *     ],
26   *     // Sèrie 2: The 100
27   *     [
28   *         'title'          => 'The 100',
29   *         'description'    => 'Una sèrie on un grup de joves tornen a la Terra per avaluar si és habitable després d'un apocalipsi nuclear.',
30   *         // Portada de The 100:
31   *         'image'          => 'https://www.casaspammer.com/wp-content/uploads/2015/01/qdfc7pzsrm-market_maxres.jpg',
32   *         'user_name'       => 'Lexa',
33   *         'user_photo_url' => 'https://static.wikia.nocookie.net/thehundred/images/6/67/S3_episode_4_-Lexa.jpg',
34   *         'published_at'   => now()
35   *     ],
36   *     // Sèrie 3: The Witcher
37   *     [
38   *         'title'          => 'The Witcher',
39   *         'description'    => 'Una sèrie de fantasia basada en els contes d'un caçador de monstres en un món fosc i perillós.',
40   *         // Portada de The Witcher:
41   *         'image'          => 'https://cdn.mos.cms.futurecdn.net/zV8oTyn3AEfXibuuFLkez8-1200-80.jpg',
42   *         'user_name'       => 'Geralt',
43   *         'user_photo_url' => 'https://static.wikia.nocookie.net/witcher/images/5/51/Netflix_geralt_shirt.jpg',
44   *         'published_at'   => now()
45   *     ];

```

Vistes per al CRUD

Crear les vistes per al CRUD que només poden veure-ho els que tinguin els permisos adients:

```
'features' => [
    // Features::termsAndPrivacyPolicy(),
    Features::profilePhotos(),
    // Features::api(),
    Features::teams(['invitations' => true]),
    Features::accountDeletion(),
],
```

[manage/index.blade.php](#)

Aquesta vista gestiona la presentació i administració de sèries, mostrant informació rellevant i permetent accions com editar o eliminar-les.

```
index.blade.php
@extends('layouts.VideosAppLayout')
@section('content')
    

# Gestió de Sèries


    <div class="flex justify-center mb-6">
        <a href="{{ route('series.manage.create') }}" class="bg-gray-800 text-white px-6 py-3 rounded hover:bg-gray-700 transition-all">
            Afegir Sèrie
        </a>
    </div>
    <div class="flex justify-center">
        <div class="w-full lg:w-3/4 px-4 py-6">
            <div class="bg-gray-100 rounded-lg shadow-xl overflow-x-auto">
                <table class="min-w-full divide-y divide-gray-300">
                    <thead class="bg-gray-800">
                        <tr>
                            <th class="px-6 py-3 text-left text-white font-semibold">Portada</th>
                            <th class="px-6 py-3 text-left text-white font-semibold">Nom de la Sèrie</th>
                            <th class="px-6 py-3 text-center text-white font-semibold">Foto Perfil</th>
                            <th class="px-6 py-3 text-left text-white font-semibold">Creador</th>
                            <th class="px-6 py-3 text-center text-white font-semibold">Accions</th>
                        </tr>
                    </thead>
                    <tbody class="bg-white divide-y divide-gray-200">
                        @foreach($series as $serie)
                            <tr class="hover:bg-gray-50 transition-colors">
                                <!-- Portada de la sèrie amb imatge lleugerament més gran i rectangular -->
                                <td class="px-6 py-4">
                                    @if($serie->image)
                                        
                                    @else
                                        <span class="text-gray-500">Sense imatge</span>
                                    @endif
                                </td>
                                <!-- Títol de la sèrie -->
                                <td class="px-6 py-4 text-gray-900 font-medium">{{ $serie->title }}</td>
                                <!-- Foto de perfil del creador -->
                                <td class="px-6 py-4 text-center">
                                    @if($serie->user_photo_url)
                                        
                                    @else
                                        <span class="text-gray-500">Sense foto</span>
                                    @endif
                                </td>
                            </tr>
                        @endforeach
                    </tbody>
                </table>
            </div>
        </div>
    </div>

```

[manage/create.blade.php](#)

Aquesta vista proporciona un formulari perquè els usuaris puguin crear una nova sèrie, validant la introducció de dades essencials com títol, descripció i imatge.

```
create.blade.php ✘

1 @extends('layouts.VideosAppLayout')
2
3 @section('content')
4     <div class="max-w-xl mx-auto bg-gray-800 p-6 rounded-lg shadow-lg">
5         <h1 class="text-2xl font-bold mb-4 text-white">Crear Nova Sèrie</h1>
6         <form action="{{ route('series.manage.store') }}" method="POST">
7             @csrf
8             <div class="mb-4">
9                 <label for="title" class="block text-sm font-medium text-white mb-1">Titol</label>
10                <input type="text" name="title" id="title" placeholder="Introdueix el títol"
11                    class="w-full p-2 bg-gray-700 text-white border border-gray-600 rounded focus:outline-none focus:border-blue-400" required>
12            </div>
13            <div class="mb-4">
14                <label for="description" class="block text-sm font-medium text-white mb-1">Descripció</label>
15                <textarea name="description" id="description" placeholder="Introdueix una descripció" rows="4"
16                    class="w-full p-2 bg-gray-700 text-white border border-gray-600 rounded focus:outline-none focus:border-blue-400" required>
17            </div>
18            <div class="mb-4">
19                <label for="image" class="block text-sm font-medium text-white mb-1">Portada (URL de la imatge)</label>
20                <input type="url" name="image" id="image" placeholder="https://exemple.com/imatge.jpg"
21                    class="w-full p-2 bg-gray-700 text-white border border-gray-600 rounded focus:outline-none focus:border-blue-400" required>
22            </div>
23            <div class="flex justify-end space-x-3 mt-4">
24                <a href="{{ route('series.manage.index') }}">
25                    <button class="bg-gray-500 text-white px-4 py-2 rounded hover:bg-gray-700 transition-colors">
26                        Tornar Enrere
27                    </button>
28                    <button type="submit"
29                        class="bg-blue-500 text-white px-4 py-2 rounded hover:bg-blue-700 transition-colors">
30                        Desar Sèrie
31                    </button>
32                </div>
33            </form>
34        </div>
35    @endsection
```

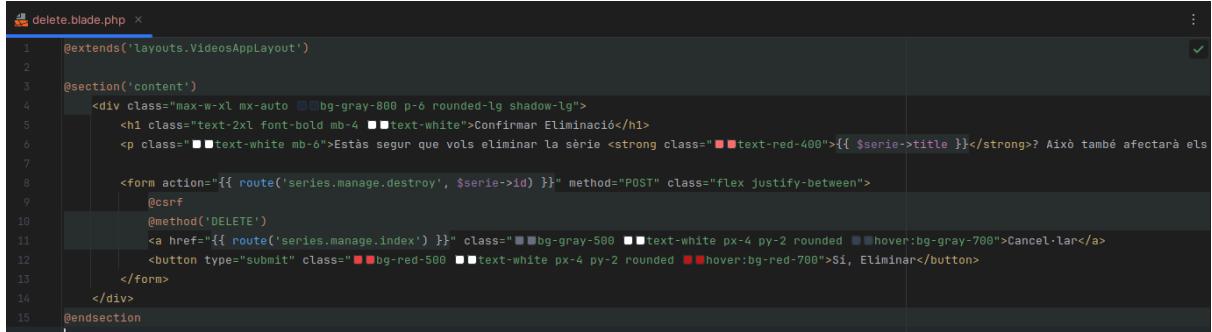
[manage/edit.blade.php](#)

Aquesta vista proporciona un formulari perquè els usuaris puguin editar una sèrie existent, permetent la modificació del títol, la descripció i la portada.

```
edit.blade.php ×
1 @extends('Layouts.VideosAppLayout')
2
3 @section('content')
4     <div class="max-w-xl mx-auto bg-gray-800 p-6 rounded-lg shadow-lg">
5         <h1 class="text-2xl font-bold mb-4 text-white">Editar Sèrie</h1>
6         <form action="{{ route('series.manage.update', $serie->id) }}" method="POST">
7             @csrf
8             @method('PUT')
9             @if($serie->image)
10                 <div class="mb-4 text-center">
11                     
13                 </div>
14             @endif
15             <div class="mb-4">
16                 <label for="title" class="block text-sm font-medium text-white mb-1">Títol</label>
17                 <input type="text" name="title" id="title" value="{{ $serie->title }}"
18                         class="w-full p-2 bg-gray-700 text-white border border-gray-600 rounded focus:outline-none focus:border-blue-400" required>
19             </div>
20             <div class="mb-4">
21                 <label for="description" class="block text-sm font-medium text-white mb-1">Descripció</label>
22                 <textarea name="description" id="description" rows="4"
23                         class="w-full p-2 bg-gray-700 text-white border border-gray-600 rounded focus:outline-none focus:border-blue-400" required>
24             </div>
25             <div class="mb-4">
26                 <label for="image" class="block text-sm font-medium text-white mb-1">Nova Portada (URL)</label>
27                 <input type="url" name="image" id="image" value="{{ $serie->image }}"
28                         placeholder="https://exemple.com/nova_imatge.jpg"
29                         class="w-full p-2 bg-gray-700 text-white border border-gray-600 rounded focus:outline-none focus:border-blue-400">
30             </div>
31             <div class="flex justify-end space-x-3 mt-4">
32                 <a href="{{ route('series.manage.index') }}"
33                     class="bg-gray-500 text-white px-4 py-2 rounded hover:bg-gray-700 transition-colors">
34                     Tornar Enrere
35                 </a>
36                 <button type="submit"
37                     class="bg-blue-500 text-white px-4 py-2 rounded hover:bg-blue-700 transition-colors">
38                     Actualitzar Sèrie
39                 </button>
40             </div>
41         </form>
```

[manage/delete.blade.php](#)

Aquesta vista mostra un formulari de confirmació per eliminar una sèrie, advertint sobre l'impacte en els vídeos associats i oferint opcions per cancel·lar o procedir amb l'eliminació.



The screenshot shows a code editor with the file 'delete.blade.php' open. The code is written in Blade PHP template syntax. It extends a layout and contains a section for content. Inside the section, there is a confirmation message and a form with two buttons: 'Cancelar' and 'Eliminar'. The code uses Bootstrap classes for styling.

```
delete.blade.php
1  @extends('layouts.VideosAppLayout')
2
3  @section('content')
4      <div class="max-w-xl mx-auto bg-gray-800 p-6 rounded-lg shadow-lg">
5          <h1 class="text-2xl font-bold mb-4 text-white">Confirmar Eliminació</h1>
6          <p class="text-white mb-6">Estàs segur que vols eliminar la sèrie <strong class="text-red-400">{{ $serie->title }}</strong>? Això també afectarà els
7
8              <form action="{{ route('series.manage.destroy', $serie->id) }}" method="POST" class="flex justify-between">
9                  @csrf
10                 @method('DELETE')
11                 <a href="{{ route('series.manage.index') }}" class="bg-gray-500 text-white px-4 py-2 rounded hover:bg-gray-700">Cancelar</a>
12                 <button type="submit" class="bg-red-500 text-white px-4 py-2 rounded hover:bg-red-700">$1, Eliminar</button>
13             </form>
14         </div>
15     @endsection
```

index.blade.php

Aquesta vista mostra totes les sèries disponibles, permetent la cerca per títol i organitzant-les en una graella visual atractiva.

[show.blade.php](#)

Aquesta vista presenta la informació detallada d'una sèrie, incloent el creador, la descripció i la llista d'episodis disponibles amb enllaços als vídeos.

```
show.blade.php x
1  @php use Illuminate\Support\Str; @endphp
2  @extends('layouts.VideosAppLayout')
3
4  @section('content')
5      <div class="container mx-auto px-4 sm:px-6 lg:px-8">
6          <!-- Banner de la sèrie: imatge limitada en amplada -->
7          <div class="relative w-full max-w-4xl mx-auto my-6">
8              title }}" class="w-full h-48 object-cover rounded-lg shadow">
9              <div class="absolute inset-0 bg-black bg-opacity-50 flex flex-col justify-center items-center rounded-lg">
10                  <h1 class="text-3xl font-bold text-white text-center px-4">{{ $serie->title }}
```

PermissionHelper

A helpers crear els permisos de gestió de les sèries per al crud i assignar-los als usuaris superadmin.

Aquest ajudant defineix i gestiona permisos per a la gestió de sèries i vídeos, assignant-los a rols específics com 'superadmin' i 'videomanager'.

```
⑤ PermissionHelper.php ×
9  class PermissionHelper
10 {
11     /** Defineix totes les portes d'accés (Gates) relacionades amb la gestió de sèries, vídeos i usuaris.*/
12     6 usages ▾ Wharvey123 *
13     public static function define_gates(): void
14     {
15         // Gestions generals
16         Gate::define(ability: 'manage-series', fn($user) => $user->hasPermissionTo('manage series') || $user->isSuperAdmin());
17         Gate::define(ability: 'manage-videos', fn($user) => $user->hasPermissionTo('manage videos') || $user->isSuperAdmin());
18         Gate::define(ability: 'manage-users', fn($user) => $user->hasPermissionTo('manage users') || $user->isSuperAdmin());
19
20         // Permisos específics d'usuaris
21         Gate::define(ability: 'create-users', fn($user) => $user->hasPermissionTo('users.create') || $user->isSuperAdmin());
22         Gate::define(ability: 'edit-users', fn($user) => $user->hasPermissionTo('users.edit') || $user->isSuperAdmin());
23         Gate::define(ability: 'delete-users', fn($user) => $user->hasPermissionTo('users.delete') || $user->isSuperAdmin());
24
25         // Permisos específics de sèries
26         Gate::define(ability: 'create-series', fn($user) =>
27             | $user->hasPermissionTo('series.create') || $user->hasPermissionTo('manage series') || $user->isSuperAdmin()
28         );
29         Gate::define(ability: 'edit-series', fn($user) =>
30             | $user->hasPermissionTo('series.edit') || $user->hasPermissionTo('manage series') || $user->isSuperAdmin()
31         );
32         Gate::define(ability: 'delete-series', fn($user) =>
33             | $user->hasPermissionTo('series.delete') || $user->hasPermissionTo('manage series') || $user->isSuperAdmin()
34     }
35 }
```

```
6 usages new *
public static function create_series_management_permissions(): void
{
    $seriesPermissions = ['manage series', 'series.create', 'series.edit', 'series.delete',];
    $videoPermissions = ['manage videos', 'videos.create', 'videos.edit', 'videos.delete',];

    // Crear permisos si no existeixen
    foreach (array_merge($seriesPermissions, $videoPermissions) as $permission) {
        Permission::firstOrCreate(['name' => $permission]);
    }

    // Assignar permisos al rol superadmin
    $superadminRole = Role::firstOrCreate(['name' => 'superadmin']);
    $superadminRole->syncPermissions(array_merge($seriesPermissions, $videoPermissions));

    // Assignar només permisos de sèries al rol videomanager
    $videomanagerRole = Role::firstOrCreate(['name' => 'videomanager']);
    $videomanagerRole->syncPermissions($seriesPermissions);
}
```

SeriesFactory

Aquesta fàbrica genera dades fictícies per a sèries, incloent títol, descripció, imatges i informació del creador, facilitant la creació de registres per a proves.

```
© SeriesFactory.php x
1  <?php
2
3  namespace Database\Factories;
4
5  > use ...
6
7
8  class SeriesFactory extends Factory
9  {
10     protected $model = Serie::class;
11
12     public function definition(): array
13     {
14         return [
15             'title'      => $this->faker->sentence,
16             'description' => $this->faker->paragraph,
17             'image'       => $this->faker->imageUrl( width: 150, height: 150),
18             'user_name'   => $this->faker->name,
19             'user_photo_url' => $this->faker->imageUrl( width: 50, height: 50),
20             'published_at' => now(),
21         ];
22     }
23 }
```

SerieTest

Aquest test comprova que una sèrie pot tenir vídeos associats, seguint la metodologia TDD i la separació AAA per estructurar el procés de prova.

```
① SerieTest.php ×
1  <?php
2
3  namespace Tests\Unit;
4
5  > use ...
6
7
8  class SerieTest extends TestCase
9  {
10
11     use RefreshDatabase;
12
13
14     public function test_serie_have_videos()
15     {
16         // Arrange: Create a series instance.
17         $serie = Serie::factory()->create();
18
19         // Create a user to associate with videos if required by the Video factory/model
20         $user = User::factory()->create();
21
22
23         // Act: Create 3 videos associated with the series.
24         Video::factory()->count( count: 3)->create([
25             'series_id' => $serie->id,
26             'user_id'   => $user->id, // Providing a valid user_id if required by the model.
27         ]);
28
29         // Refresh the series model to make sure the videos relationship is updated.
30         $serie->refresh();
31
32         // Retrieve the videos count using the relationship.
33         $videosCount = $serie->videos()->count();
34
35         // Assert: Verify the series has exactly 3 videos.
36         $this->assertEquals( expected: 3, $videosCount);
37     }
38 }
```

```
aumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ php artisan test tests/Unit/SerieTest.php

  PASS  Tests\Unit\SerieTest
    ✓ serie have videos

  Tests:    1 passed (1 assertions)
  Duration: 0.10s

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

SeriesManageControllerTest

Aquest test funcional comprova la gestió de permisos i accés per a la creació, edició i eliminació de sèries, assegurant que només els usuaris autoritzats puguin realitzar aquestes accions.

```
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ php artisan test tests/Feature/SeriesManageControllerTest.php

  PASS  Tests\Feature\Series\SeriesManageControllerTest
    ✓ user with permissions can see add series
    ✓ user without series manage create cannot see add series
    ✓ user with permissions can store series
    ✓ user without permissions cannot store series
    ✓ user with permissions can destroy series
    ✓ user without permissions cannot destroy series
    ✓ user with permissions can see edit series
    ✓ user without permissions cannot see edit series
    ✓ user with permissions can update series
    ✓ user without permissions cannot update series
    ✓ user with permissions can manage series
    ✓ regular users cannot manage series
    ✓ guest users cannot manage series
    ✓ videomanagers can manage series
    ✓ superadmins can manage series

  Tests:   15 passed (28 assertions)
  Duration: 0.34s

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

Rutes de les sèries

[DatabaseSeeder.php](#)

Aquest seeder configura la gestió de permisos i crea les sèries per defecte mitjançant SeriesHelper::create_series(), assegurant que les sèries necessàries estiguin disponibles a la base de dades.

```
© DatabaseSeeder.php ×

10 class DatabaseSeeder extends Seeder
11 {
12     public function run(): void
13     {
14         // Crear permisos per a la gestió de vídeos
15         PermissionHelper::create_permissions();
16         // Crear permisos per a la gestió d'usuaris
17         PermissionHelper::create_user_management_permissions();
18         // Crear permisos per a la gestió de sèries
19         PermissionHelper::create_series_management_permissions();
20     }
}
```

[Web.php](#)

Rutes públiques: Permeten als usuaris veure totes les sèries disponibles (series.index) i consultar els detalls d'una sèrie específica (series.show).

```
php web.php ×

16 // -----
17 // RUTES PÚBLIQUES (accesibles sense autenticació)
18 // -----
19
20 // Vídeos
21 Route::get('videos', [VideosController::class, 'index'])->name('videos.index');
22 Route::get('video/{id}', [VideosController::class, 'show'])->name('video.show');
23
24 // Sèries
25 Route::get('series', [SeriesController::class, 'index'])->name('series.index');
26 Route::get('series/{id}', [SeriesController::class, 'show'])->name('series.show');
27
28 // Usuaris
29 Route::get('users', [UsersController::class, 'index'])->name('users.index');
30 Route::get('users/{id}', [UsersController::class, 'show'])->name('users.show');
```

Rutes protegides: Només accessibles per usuaris autènticats amb els permisos adequats. Permeten gestionar les sèries mitjançant accions de creació, edició, actualització i eliminació (series.manage.*), garantint que només administradors o usuaris amb permisos puguin modificar el contingut.

```
// -----
// SÈRIES MANAGEMENT (CRUD)
// -----
Route::prefix('prefix: 'seriesmanage')
->name('value: 'series.manage.')
->middleware('middleware: can:manage-series')
->group(function () {
    Route::get('uri: '/'', [SeriesManageController::class, 'index'])->name('name: 'index')
    Route::get('uri: '/create', [SeriesManageController::class, 'create'])
        ->middleware('middleware: can:create-series')
        ->name('name: 'create')
    Route::post('uri: '/'', [SeriesManageController::class, 'store'])
        ->middleware('middleware: can:create-series')
        ->name('name: 'store')
    Route::get('uri: '/{id}/edit', [SeriesManageController::class, 'edit'])
        ->middleware('middleware: can:edit-series')
        ->name('name: 'edit')
    Route::put('uri: '/{id}', [SeriesManageController::class, 'update'])
        ->middleware('middleware: can:edit-series')
        ->name('name: 'update')
    Route::get('uri: '/{id}/delete', [SeriesManageController::class, 'delete'])
        ->middleware('middleware: can:delete-series')
        ->name('name: 'delete')
    Route::delete('uri: '/{id}', [SeriesManageController::class, 'destroy'])
        ->middleware('middleware: can:delete-series')
        ->name('name: 'destroy');
});
```

VideosAppLayout.blade.php

Aquest fragment de codi gestiona la navegació i la visualització de les sèries dins l'aplicació. Mostra un enllaç a la pàgina de visualització de sèries (series.index) i permet la gestió de sèries (series.manage.index) per als usuaris amb permisos. La seva implementació està integrada tant en la barra de navegació per a escriptori com en el menú per a dispositius mòbils, garantint una experiència d'usuari fluida.

```

1 <html lang="{{ str_replace('_', '-', app()->getLocale()) }}>
2   <body class="flex flex-col min-h-screen bg-[#100c0c]">
3     <nav class="bg-gray-800 p-6 shadow-lg x-data="{ menuOpen: false }">
4       <div class="container mx-auto flex justify-between items-center">
5         <!-- Enllaços de navegació per a desktop -->
6         <div class="hidden lg:flex justify-between w-full max-w-5xl mx-auto">
7           <!-- Videos -->
8           <a href="{{ route('videos.index') }}" class="text-white text-lg font-semibold flex-grow text-center">Videos</a>
9
10          <!-- Manage Videos (només amb permís) -->
11          @can('manage-videos')
12            <a href="{{ route('videos.manage.index') }}" class="text-white text-lg font-semibold flex-grow text-center">Manage Videos</a>
13          @endcan
14
15          <!-- Sèries -->
16          <a href="{{ route('series.index') }}" class="text-white text-lg font-semibold flex-grow text-center">Sèries</a>
17
18          <!-- Manage Sèries (només amb permís) -->
19          @can('manage-series')
20            <a href="{{ route('series.manage.index') }}" class="text-white text-lg font-semibold flex-grow text-center">Manage Sèries</a>
21          @endcan
22
23          <!-- Users -->
24          <a href="{{ route('users.index') }}" class="text-white text-lg font-semibold flex-grow text-center">Users</a>
25
26          <!-- Manage Users (només amb permís) -->
27          @can('manage-users')
28            <a href="{{ route('users.manage.index') }}" class="text-white text-lg font-semibold flex-grow text-center">Manage Users</a>
29          @endcan
30
31        </div>
32      </div>
33    </nav>
34  </body>
35</html>

```

```

// Gestió d'usuaris (CRUD) - només per usuaris amb permisos de gestió
Route::prefix('usersmanager')
    ->name('users.manage.')
    ->middleware('can:manage-users')
    ->group(function () {
        Route::get('/', [UsersManageController::class, 'index'])->name('index');
        Route::get('/create', [UsersManageController::class, 'create'])->middleware('can:create-users')->name('create');
        Route::post('/', [UsersManageController::class, 'store'])->middleware('can:create-users')->name('store');
        Route::get('/{id}/edit', [UsersManageController::class, 'edit'])->middleware('can:edit-users')->name('edit');
        Route::put('/{id}', [UsersManageController::class, 'update'])->middleware('can:edit-users')->name('update');
        Route::get('/{id}/delete', [UsersManageController::class, 'delete'])->middleware('can:delete-users')->name('delete');
        Route::delete('/{id}', [UsersManageController::class, 'destroy'])->middleware('can:delete-users')->name('destroy');
    });

```

resources/markdown/terms

M→ terms.md ×

```
93
94 # Sprint 6 - Gestió de Sèries i Assignació de Vídeos a Sèries
95
96 **Funcionalitats implementades:**
97 - Creació de la migració per a la taula `series`.
98 - Creació del model `Serie` amb les relacions i accessors.
99 - Actualització del model `Video` per afegir la relació amb `Serie`.
100 - Desenvolupament dels controladors `SeriesController` (pública) i `SeriesManageController`
101 - Creació d'un helper (`SeriesHelper`) per generar 3 sèries per defecte.
102 - Creació de vistes per al CRUD de sèries i la visualització pública.
103 - Definició de rutes per a la gestió de sèries.
104 - Implementació de tests utilitzant TDD/AAA per comprovar la funcionalitat.
105 - Assignació i creació de permisos per a la gestió de sèries.
106 |
```

Sprint 6 - Gestió de Sèries i Assignació de Vídeos a Sèries

Funcionalitats implementades:

- Creació de la migració per a la taula `series`.
- Creació del model `Serie` amb les relacions i accessors.
- Actualització del model `Video` per afegir la relació amb `Serie`.
- Desenvolupament dels controladors `SeriesController` (pública) i `SeriesManageController` (CRUD).
- Creació d'un helper (`SeriesHelper`) per generar 3 sèries per defecte.
- Creació de vistes per al CRUD de sèries i la visualització pública.
- Definició de rutes per a la gestió de sèries.
- Implementació de tests utilitzant TDD/AAA per comprovar la funcionalitat.
- Assignació i creació de permisos per a la gestió de sèries.

Larastan errors.

```
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$ vendor/bin/phpstan analyse --memory-limit=1G
Note: Using configuration file /home/alumnat/Code/M7-8-9/VideoAppHarvey/phpstan.neon.
79/79 [██████████] 100%  
  
[OK] No errors  
  
alumnat@harvey:~/Code/M7-8-9/VideoAppHarvey$
```