ENTREGA 2

Modificaciones y/o correcciones:

-Agregar el metodo toString a todas las Entity:

º Motivo: Para poder convertir un Objeto en una cadena y asi poder mostrarlo en las plantillas.

```
    return $this;
}
public function __toString(): string
{
    return $this->nombre ?? 'Sin nombre';
}
```

-Correcciones en la relación Playlist-UsuarioPlaylist:

º Modificaremos el código para que una **Playlist** pueda o no haber sido reproducida por un **Usuario.**

-Antes:

```
#[ORM\ManyToOne(inversedBy: 'reproduccionesDeUsuario')]
private ?Playlist $playlist = null;
```

Desde Entity\UsuarioPlaylist

-Ahora:

```
#[ORM\ManyToOne(targetEntity: Playlist::class, inversedBy: 'reproduccionesDeUsuario')]
#[ORM\JoinColumn(nullable:false, onDelete: 'CASCADE')]
private Playlist $playlist ;
```

Desde Entity\UsuarioPlaylist

º Cambios:

1º Añadimos **`targetEntity`** en la relación para poder mapear la entidad en la **BBDD**.

2º Añadimos **#[ORM\JoinColumn(nullable: false)]** para que si un **Usuario** reproduce una **Playlist** haya una **Playlist** asociada a su reproducción.

3º **onDelete** garantiza que si se elimina una **Playlist** la **UsuarioPlaylist** asociada se eliminará también.

4º Modificamos **private Playlist $playlist ;** para que en el momento del registro tenga que tener una **Playlist** asociada.

-Corrección de la relación Usuario-Perfil:

º Modificamos el código para que un **Usuario** pueda existir sin un **Perfil**, pero un **Perfil** no pueda hacerlo sin un **Usuario** asociado.

-Antes:

```php
#[ORM\OneToOne(inversedBy: 'usuario', cascade: ['persist', 'remove'])]
private ?Perfil $perfil = null;
```

Desde Entity\Usuario

```php
#[ORM\OneToOne(mappedBy: 'perfil', cascade: ['persist', 'remove'])]
private ?Usuario $usuario = null;
```

Desde Entity\Perfil

-Ahora:

```php
#[ORM\OneToOne(targetEntity: Perfil::class, mappedBy: 'usuario', cascade: ['persist', 'remove'])]
private ?Perfil $perfil = null;
```

Desde Entity\Usuario

```php
#[ORM\OneToOne(targetEntity: Usuario::class, inversedBy: 'perfil')]
#[ORM\JoinColumn(nullable: false, onDelete: 'CASCADE')]
private Usuario $usuario;
```

Desde Entity\Perfil

º Cambios en **Usuario**:

1º Definimos el **targetEntity** .

2º **mappedBy: 'usuario'** indica que la relacion inversa esta definida en la propiedad **usuario** de la entidad **Perfil**

ºCambios en **Perfil:**

1º Definimos el **targetEntity.**

2º **inversedBy: 'perfil'** indica que la relación inversa está definida en la propiedad '**perfil'** de la entidad U**suario.**

3º El **ORM\JoinColum** asegura que un **Perfil** no pueda existir sin un **Usuario** asociado, **onDelete** garantiza que si se elimina un **Usuario** su **Perfil** asociado se eliminará también.

4º Eliminar ? Y null , para obligar a la asociacion de un **Usuario** al **Perfil.**

-Correcciones en la relación Usuario-UsuarioPlaylist:

º Modificaremos el código para que un **Usuario** pueda o no haber reproducido una **Playlist.**

-Antes:

```php
#[ORM\ManyToOne(inversedBy: 'playlistsReproducidas')]
private ?Usuario $usuario = null;
```

Desde Entity\UsuarioPlaylist

-Ahora:

```php
#[ORM\ManyToOne(targetEntity: Usuario::class, inversedBy: 'playlistsReproducidas')]
#[ORM\JoinColumn(nullable:false, onDelete: 'CASCADE')]
private Usuario $usuario ;
```

Desde Entity\UsuarioPlaylist

º Cambios:

1º Añadimos **`targetEntity`** en la relación para poder mapear la entidad en la **BBDD**.

2º Añadimos **#[ORM\JoinColumn(nullable: false)]** para que si se reproduce una **Playlist** haya un **Usuario** asociado a su reproducción.

3º **onDelete** garantiza que si se elimina un **Usuario** el **UsuarioPlaylist** asociado se eliminará también.

4º Modificamos **private Usuario $usuario;** para que en el momento del registro tenga que tener un **Usuario** asociado.

-Corrección en la relación entre Canción y Estilo:

º Modificaremos el código para que si un **estilo** es borrado, el valor asociado a la propiedad **género** de **canción** tome un valor NULL.

-Antes:

Desde Entity\Cancion

-Ahora:

```php
#[ORM\ManyToOne(targetEntity: Estilo::class, inversedBy: 'canciones')]
#[ORM\JoinColumn(nullable: true, onDelete: 'SET NULL')]
private ?Estilo $genero = null;
```

Desde Entity\Cancion

º Cambios:

1º Declaración del **targetEntity**

2º Agregación del **JoinColumn** para que si se elimina el **Estilo** asociado, el valor de **género** sea Null

Correción en la relación entre PlaylistCancion con Playlist y Cancion:

º Modificaciones: modificaremos el código para que si se borra una **canción** o una **Playlist** que esté asociada a un registro en la tabla intermedia **PlaylistCancion** , se borre éste registro, impidiendo datos huerfanos.

-Antes:

```php
#[ORM\ManyToOne(inversedBy: 'canciones')]
private ?Playlist $playlist = null;

#[ORM\ManyToOne(inversedBy: 'playlists')]
private ?Cancion $cancion = null;
```

Desde Entity\PlaylitCancion

-Ahora:

```php
#[ORM\ManyToOne(targetEntity:Playlist::class, inversedBy: 'canciones')]
#[ORM\JoinColumn(nullable:false, onDelete: 'CASCADE')]
private Playlist $playlist ;

#[ORM\ManyToOne(targetEntity:Cancion::class,inversedBy: 'playlists')]
#[ORM\JoinColumn(nullable:false, onDelete: 'CASCADE')]
private Cancion $cancion ;
```

Desde Entity\PlaylitCancion

º Cambios:

1º Declaración del **targetEntity.**

2º Agregación del **JoinColumn** para no permitir valores null y generar un borrado en cascada desde las entidades asociadas.

-Correción en la relación entre Usuario y Playlist (como creador)

º Modificaciones: Estableceremos que cuando el **Usuario** creador de lista es eliminado, la **Playlist** que creo se borre con él.

-Antes:

```
#[ORM\ManyToOne(inversedBy: 'Playlists')]
private ?Usuario $propietario = null;
```

Desde Entity\Playlist

-Ahora:

```
#[ORM\ManyToOne(targetEntity:Usuario::class, inversedBy: 'Playlists')]
#[ORM\JoinColumn(nullable: false, onDelete: 'CASCADE')]
private Usuario $propietario ;
```

Desde Entity\Playlist

º Cambios:

1º Definicion del **targetEntity.**

2º Declaración de la linea **JoinColumn** para para no permitir valores null y generar un borrado en cascada desde las entidades asociadas.

-Creación de la relación NM entre Perfil y Estilo mediante tabla intermedia:

1º Creacion de la tabla: PerfilEstilo:

```
root@9a36b1fcca44:/var/www/symfony/NoelProyectoSymfony# php bin/console make:entity

 Class name of the entity to create or update (e.g. FierceElephant):
 > PerfilEstilo

 Add the ability to broadcast entity updates using Symfony UX Turbo? (yes/no) [no]:
 > no

 created: src/Entity/PerfilEstilo.php
 created: src/Repository/PerfilEstiloRepository.php

 Entity generated! Now let's add some fields!
 You can always add more fields later manually or by re-running this command.

 New property name (press <return> to stop adding fields):
 > id

 Field type (enter ? to see all types) [string]:
 > integer

 Can this field be null in the database (nullable) (yes/no) [no]:
 > no

 Not generating PerfilEstilo::getId(): method already exists
 updated: src/Entity/PerfilEstilo.php

 Add another property? Enter the property name (or press <return> to stop adding fie
ds):
 >


  Success!


 Next: When you're ready, create a migration with php bin/console make:migration
```

2º Creación de la relación con Perfil:

```
root@9a36b1fcca44:/var/www/symfony/NoelProyectoSymfony# php bin/console make:entity

 Class name of the entity to create or update (e.g. DeliciousKangaroo):
 > PerfilEstilo

 Your entity already exists! So let's add some new fields!

 New property name (press <return> to stop adding fields):
 > perfiles

 Field type (enter ? to see all types) [string]:
 > relation

 What class should this entity be related to?:
 > Perfil
```

```
 Relation type? [ManyToOne, OneToMany, ManyToMany, OneToOne]:
 > ManyToOne

 Is the PerfilEstilo.perfiles property allowed to be null (nullable)? (yes/no) [yes]:
 > no

 Do you want to add a new property to Perfil so that you can access/update PerfilEsti
lo objects from it - e.g. $perfil->getPerfilEstilos()? (yes/no) [yes]:
 > yes

 A new property will also be added to the Perfil class so that you can access the rel
ated PerfilEstilo objects from it.

 New field name inside Perfil [perfilEstilos]:
 > estilosPreferidos

 Do you want to activate orphanRemoval on your relationship?
 A PerfilEstilo is "orphaned" when it is removed from its related Perfil.
 e.g. $perfil->removePerfilEstilo($perfilEstilo)

 NOTE: If a PerfilEstilo may *change* from one Perfil to another, answer "no".

 Do you want to automatically delete orphaned App\Entity\PerfilEstilo objects (orphan
Removal)? (yes/no) [no]:
 >

 updated: src/Entity/PerfilEstilo.php
 updated: src/Entity/Perfil.php

 Add another property? Enter the property name (or press <return> to stop adding fiel
ds):
 >


 Success!


 Next: When you're ready, create a migration with php bin/console make:migration
```

3º Creación de la relación con Estilo:

```
root@9a36b1fcca44:/var/www/symfony/NoelProyectoSymfony# php bin/console make:entity

 Class name of the entity to create or update (e.g. OrangeChef):
 > PerfilEstilo

 Your entity already exists! So let's add some new fields!

 New property name (press <return> to stop adding fields):
 > estilos

 Field type (enter ? to see all types) [string]:
 > relation

 What class should this entity be related to?:
 > Estilo
```

```
 Relation type? [ManyToOne, OneToMany, ManyToMany, OneToOne]:
 > ManyToOne

 Is the PerfilEstilo.estilos property allowed to be null (nullable)? (yes/no) [yes]:
 > no

 Do you want to add a new property to Estilo so that you can access/update PerfilEsti
lo objects from it - e.g. $estilo->getPerfilEstilos()? (yes/no) [yes]:
 > yes

 A new property will also be added to the Estilo class so that you can access the rel
ated PerfilEstilo objects from it.

 New field name inside Estilo [perfilEstilos]:
 > perfilesSeguidores

 Do you want to activate orphanRemoval on your relationship?
 A PerfilEstilo is "orphaned" when it is removed from its related Estilo.
 e.g. $estilo->removePerfilEstilo($perfilEstilo)

 NOTE: If a PerfilEstilo may *change* from one Estilo to another, answer "no".

 Do you want to automatically delete orphaned App\Entity\PerfilEstilo objects (orphar
Removal)? (yes/no) [no]:
 > no

 updated: src/Entity/PerfilEstilo.php
 updated: src/Entity/Estilo.php

 Add another property? Enter the property name (or press <return> to stop adding fiel
ds):
 >


 Success!


 Next: When you're ready, create a migration with php bin/console make:migration
```

4º Reflejar en la BBDD con migración.

```
root@9a36b1fcca44:/var/www/symfony/NoelProyectoSymfony# php bin/console make:migration

 [WARNING] You have 1 available migrations to execute.


 Are you sure you wish to continue? (yes/no) [yes]:
 > yes

 created: migrations/Version20250202231348.php


 Success!


 Review the new migration then run it with php bin/console doctrine:migrations:migrate
 See https://symfony.com/doc/current/bundles/DoctrineMigrationsBundle/index.html
root@9a36b1fcca44:/var/www/symfony/NoelProyectoSymfony# php bin/console doctrine:migrations:migrate


 WARNING! You are about to execute a migration in database "spotifyDB" that could result in schema
changes and data loss. Are you sure you wish to continue? (yes/no) [yes]:
 > yes

[notice] Migrating up to DoctrineMigrations\Version20250202231348
[notice] finished in 8624.4ms, used 22M memory, 1 migrations executed, 10 sql queries

 [OK] Successfully migrated to version: DoctrineMigrations\Version20250202231348
```

-Correcciones en la relación entre PerfilEstilo con Perfil y Estilo:

º Modificaremos el código para que si se elimina una de las 2 entidades relacionadas mediante ésta tabla, se borre el registro de datos de su relación.

-Antes:

```php
#[ORM\ManyToOne(inversedBy: 'estilosPreferidos')]
#[ORM\JoinColumn(nullable: false)]
private ?Perfil $perfiles = null;

#[ORM\ManyToOne(inversedBy: 'perfilesSeguidores')]
#[ORM\JoinColumn(nullable: false)]
private ?Estilo $estilos = null;
```

Desde Entity\PerfilEstilo

-Ahora:

```php
#[ORM\ManyToOne(targetEntity: Perfil::class, inversedBy: 'estilosPreferidos')]
#[ORM\JoinColumn(nullable: false, onDelete: 'CASCADE')]
private Perfil $perfil;

#[ORM\ManyToOne(targetEntity: Estilo::class, inversedBy: 'perfilesSeguidores')]
#[ORM\JoinColumn(nullable: false, onDelete: 'CASCADE')]
private Estilo $estilo;
```

Desde Entity\PerfilEstilo

º Cambios:

1º Definición del **targetEntity**

2º **onDelete: 'CASCADE'** para eliminar datos huérfanos.

3º Eliminación de posible valor null para **Perfil** y **Estilo**

-Corrección del controlador de perfil

º Modificaciones: Modificaciòn del codigo para validar datos en la relación entre **perfil** y **estilo** a traves de la tabla intermedia **perfilEstilo:**

-Antes:

```php
#[Route('perfil/new_perfil', name: 'app_perfil_new')]
public function newPerfil(EntityManagerInterface $e): JsonResponse
{
    $perfilRepository = $e->getRepository(Perfil::class);

    $perfil = new Perfil();
    $perfil->setDescripcion('Usuario Premium creador de listas variadas');



        $e->persist($perfil);
        $e->flush();

        return $this->json([
            'message' => 'Perfil '.$perfil->getId().' creado',
            'path' => 'src/Controller/PerfilController'
        ]);

}
```

Desde Entity\Perfil

-Ahora:

```php
#[Route('perfil/new_perfil/{nombreUsuario}/{estiloFavorito}', name: 'app_perfil_new')]
public function newPerfil(EntityManagerInterface $e, string $nombreUsuario, string $estiloFavorito): JsonResponse
{
    $usuarioRepository = $e->getRepository(Usuario::class);
    $estiloRepository = $e->getRepository(Estilo::class);

    $estilo = $estiloRepository->findOneByNombre($estiloFavorito);
    $usuario = $usuarioRepository->findOneByNombre($nombreUsuario);



    if (!$usuario) {
        return $this->json([
            'message' => 'Usuario no encontrado, error al crear perfil',
            'path' => 'src/Controller/PerfilController'
        ], 404);
    }

    if ($usuario->getPerfil() !== null) {
        return $this->json([
            'message' => 'Error, el usuario ' . $nombreUsuario . ' ya tiene perfil creado',
            'path' => 'src/Controller/PerfilController'
        ], 400);
    }

    if (!$estilo) {
        return $this->json([
            'message' => 'Error, estilo musical no encontrado o inexistente.
            Si quieres registrar el nuevo estilo entra a ésta URL:
            http://localhost:8000/estilo/new_estilo/{nombreEstilo}/{descripcion}
            sustituyendo los campos entre {} por su valor'
        ], 404);
    } else {
```

```
    ], 404);
} else {
    // creacion del perfil
    $perfil = new Perfil();
    $perfil->setDescripcion('Usuario Premium creador de listas variadas');
    $perfil->setFoto('../imagenes/img002.jpg');
    $perfil->setUsuario($usuario);

    // creacion de la relacion en la tabla intermedia
    $perfilEstilo = new PerfilEstilo();
    $perfilEstilo->setPerfil($perfil);
    $perfilEstilo->setEstilo($estilo);

    $perfil->addEstilosPreferidos($perfilEstilo);

    $e->persist($perfil);
    $e->persist($perfilEstilo);
    $e->flush();

    return $this->json([
        'message' => 'Perfil para el usuario ' . $usuario->getNombre() . ' creado. Estilos favorito añadido: '
        'path' => 'src/Controller/PerfilController'
    ], 200);
}
```

º Cambios:

1º Añadidas las validaciones de datos con los repository

2º Asociación del Usuario al Perfil a traves de **setUsuario($usuario);**

3º Creación de los datos para el registro de la tabla intermedia que manejará la relación entre perfil y estilo guardando ambas entidad bajo una id única.

4º Guardar el registro creado ‘perfilEstilo’ en los estilosPreferidos del Perfil

-Creaciòn del método addEstilo para perfiles ya existentes:

º Motivos: Poder añadir mas estilos a la colección **EstilosPreferidos** después de haber creado el **perfil.**

```php
#[Route('perfil/añadir_estilo/{nombreUsuario}/{estiloFavorito}', name: 'app_perfil_add_estilo')]
public function addEstilo(EntityManagerInterface $e, string $nombreUsuario, string $estiloFavorito): JsonResponse
{
    $usuarioRepository = $e->getRepository(Usuario::class);
    $estiloRepository = $e->getRepository(Estilo::class);

    $estilo = $estiloRepository->findOneByNombre($estiloFavorito);
    $usuario = $usuarioRepository->findOneByNombre($nombreUsuario);

    // validaciones
    if (!$usuario) {
        return $this->json([
            'message' => 'Usuario no encontrado, error al añadir estilos favoritos al perfil',
            'path' => 'src/Controller/PerfilController'
        ], 404);
    }

    if (!$estilo) {
        return $this->json([
            'message' => 'Estilo no encontrado, error al añadir estilo favorito al perfil',
            'path' => 'src/Controller/PerfilController'
        ], 404);
    }

    $perfil = $usuario->getPerfil();
    if (!$perfil) {
        return $this->json([
            'message' => 'Error, el usuario ' . $nombreUsuario . ' no tiene perfil creado',
            'path' => 'src/Controller/PerfilController'
        ], 400);
    }
    // comprueba si el estilo ya esta en los favoritos del perfil
    $existeEstiloEnPerfil = false;
    $estilosDelPerfil = $perfil->getEstilosPreferidos();
    foreach ($estilosDelPerfil as $perfilEstilo) {
        if ($perfilEstilo->getEstilo() === $estilo) {
            $existeEstiloEnPerfil = true;
            break;
        }
    }

    if ($existeEstiloEnPerfil) {
```

```php
    if ($existeEstiloEnPerfil) {
        return $this->json([
            'message' => 'El estilo ' . $estiloFavorito . ' ya está en los estilos favoritos de ' . $nombreUsuario
        ], 400);
    } else {
        $perfilEstilo = new PerfilEstilo();
        $perfilEstilo->setPerfil($perfil);
        $perfilEstilo->setEstilo($estilo);

        $perfil->addEstilosPreferidos($perfilEstilo);

        $e->persist($perfilEstilo);
        $e->flush();

        return $this->json([
            'message' => 'Estilo ' . $estiloFavorito . ' añadido a los estilos preferidos del usuario ' . $nombreUsuario
        ], 200);
    }
}
```

-Creacion de dashboard Admin:

```
root@9a36b1fcca44:/var/www/symfony/NoelProyectoSymfony# php bin/console make:admin:dashboard

 Which class name do you prefer for your Dashboard controller? [DashboardController]:
 >

 In which directory of your project do you want to generate "DashboardController"? [src/Controlle
Admin/]:
 >


 [OK] Your dashboard class has been successfully generated.

 Next steps:
 * Configure your Dashboard at "src/Controller/Admin/DashboardController.php"
 * Run "make:admin:crud" to generate CRUD controllers and link them from the Dashboard.
```

-Configuracion del Controller/Admin/DashboardController

```php
public function configureDashboard(): Dashboard
{
    return Dashboard::new()
        ->setTitle('Gestion de Entidades Spotify');
}

public function configureMenuItems(): iterable
{
    yield MenuItem::linkToDashboard('Dashboard', 'fa fa-home');
    yield MenuItem::linkToCrud('Canciones', 'fa fa-music', Cancion::class);
    yield MenuItem::linkToCrud('Usuarios', 'fa fa-music', Usuario::class);
    yield MenuItem::linkToCrud('Playlists', 'fa fa-music', Playlist::class);
    yield MenuItem::linkToCrud('Perfiles', 'fa fa-music', Perfil::class);
    yield MenuItem::linkToCrud('Estilos', 'fa fa-music', Estilo::class);

}
```

-Creación del archivo admin/dashboard.html.twig.

```
root@9a36b1fcca44:/var/www/symfony/NoelProyectoSymfony# cd templates
root@9a36b1fcca44:/var/www/symfony/NoelProyectoSymfony/templates# mkdir admin
root@9a36b1fcca44:/var/www/symfony/NoelProyectoSymfony/templates# chmod 777 -R admin
root@9a36b1fcca44:/var/www/symfony/NoelProyectoSymfony/templates# cd admin
root@9a36b1fcca44:/var/www/symfony/NoelProyectoSymfony/templates/admin# nano dashboard.html.twig
```

```
  GNU nano 7.2                                dashboard.html.twig *
{% extends '@EasyAdmin/page/content.html.twig' %}

{% block content %}
    <h1>Bienvenido al Panel de Administraci  n</h1>
    <p>Desde aqu   puedes gestionar usuarios, playlists y canciones.</p>
{% endblock %}
```

-Creación de CRUD para Usuario Playlist Perfil Cancion Estilo (solo mostraremos Usuario):

```
root@9a36b1fcca44:/var/www/symfony/NoelProyectoSymfony# php bin/console make:admin:crud

 Which Doctrine entity are you going to manage with this CRUD controller?:
  [0] App\Entity\Cancion
  [1] App\Entity\Estilo
  [2] App\Entity\Perfil
  [3] App\Entity\PerfilEstilo
  [4] App\Entity\Playlist
  [5] App\Entity\PlaylistCancion
  [6] App\Entity\Usuario
  [7] App\Entity\UsuarioPlaylist
 > 6

 Which directory do you want to generate the CRUD controller in? [src/Controller/Admin/]:
 >

 Namespace of the generated CRUD controller [App\Controller\Admin]:
 >

 [OK] Your CRUD controller class has been successfully generated.

 Next steps:
 * Configure your controller at "src/Controller/Admin/UsuarioCrudController.php"
 * Read EasyAdmin docs: https://symfony.com/doc/master/bundles/EasyAdminBundle/index.html
```

```php
<?php

namespace App\Controller\Admin;

use App\Entity\Cancion;
use EasyCorp\Bundle\EasyAdminBundle\Controller\AbstractCrudController;
use EasyCorp\Bundle\EasyAdminBundle\Field\IdField;
use EasyCorp\Bundle\EasyAdminBundle\Field\TextField;
use EasyCorp\Bundle\EasyAdminBundle\Field\IntegerField;
use EasyCorp\Bundle\EasyAdminBundle\Field\AssociationField;

class CancionCrudController extends AbstractCrudController
{
    public static function getEntityFqcn(): string
    {
        return Cancion::class;
    }


    public function configureFields(string $pageName): iterable
    {
        return [
            IdField::new('id')->hideOnForm(),
            TextField::new('title'),
            TextField::new('album'),
            TextField::new('autor'),
            IntegerField::new('duracion'),
            IntegerField::new('likes'),
            AssociationField::new('playlists', 'Playlists')
                ->setFormTypeOptions([
                    'by_reference' => false,
                ]),
            AssociationField::new('genero', 'Estilo')
                ->setFormTypeOptions([
                    'by_reference' => false,
                ]),
        ];
    }
}
```