

« Dans ce comparatif, vous trouverez mes modifications mises en contraste avec la version originale. La présentation peut varier : certains extraits de code sont suffisamment courts pour tenir sur une seule page, et dans ce cas, les modifications suivantes ne présentent que la version nettoyée. Lorsqu'un extrait est trop volumineux, il est découpé en plusieurs parties puis réinséré à chaque fois, afin de conserver une bonne lisibilité entre la version originale et la version nettoyée. »

1. config/preload.php

Version originale :

```
<?php
2
3 if (file_exists(dirname(__DIR__).'/var/cache/prod/App_KernelProdContainer.preload.php')) {
4     require dirname(__DIR__).'/var/cache/prod/App_KernelProdContainer.preload.php';
5 }
6
```

Version nettoyée :

```
<?php
2
3 if (file_exists(dirname(__DIR__).'/var/cache/prod/App_KernelProdContainer.preload.php')) {
4     require_once dirname(__DIR__).'/var/cache/prod/App_KernelProdContainer.preload.php';
5 }
6
```

Modification(s) effectuée(s) :

J'ai remplacé « require » par « require_once » pour éviter un double chargement potentiel (remontée classique Sonarlint).

2. migrations/Version20240513134621.php

2.1.

Version originale :

```
<?php
2
3 declare(strict_types=1);
4
5 namespace Doctrine\Migrations;
6
7 use Doctrine\DBAL\Schema\Schema;
8 use Doctrine\Migrations\AbstractMigration;
9
10 /**
11  * Auto-generated Migration: Please modify to your needs!
12  */
13 final class Version20240513134621 extends AbstractMigration
14 {
15     public function getDescription(): string
16     {
17         return '';
18     }
19
20     public function up(Schema $schema): void
21     {
22         // this up() migration is auto-generated, please modify it to your needs
23         $this->addSql('CREATE TABLE categorie (id INT AUTO_INCREMENT NOT NULL, name VARCHAR(50) DEFAULT NULL, PRIMARY KEY(id)) DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci');
24         $this->addSql('CREATE TABLE formation (id INT AUTO_INCREMENT NOT NULL, playlist_id INT DEFAULT NULL, published_at DATETIME DEFAULT NULL, title VARCHAR(255) DEFAULT NULL, INDEX IDX_830086E95200282E (formation_id), PRIMARY KEY(id)) DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci');
25         $this->addSql('CREATE TABLE formation_categorie (formation_id INT NOT NULL, categorie_id INT NOT NULL, INDEX IDX_830086E95200282E (formation_id), INDEX IDX_830086E95200282E (categorie_id), PRIMARY KEY(formation_id, categorie_id)) DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci');
26         $this->addSql('CREATE TABLE playlist (id INT AUTO_INCREMENT NOT NULL, name VARCHAR(100) DEFAULT NULL, description LONGTEXT DEFAULT NULL, PRIMARY KEY(id)) DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci');
27         $this->addSql('CREATE TABLE messenger_messages (id BIGINT AUTO_INCREMENT NOT NULL, body LONGTEXT NOT NULL, headers LONGTEXT NOT NULL, queue_name VARCHAR(255) DEFAULT NULL, PRIMARY KEY(id)) DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci');
28         $this->addSql('ALTER TABLE formation ADD CONSTRAINT FK_404021BF6BBD148 FOREIGN KEY (playlist_id) REFERENCES playlist (id)');
29         $this->addSql('ALTER TABLE formation_categorie ADD CONSTRAINT FK_830086E95200282E FOREIGN KEY (formation_id) REFERENCES formation (id) ON DELETE CASCADE');
30         $this->addSql('ALTER TABLE formation_categorie ADD CONSTRAINT FK_830086E95200282E FOREIGN KEY (categorie_id) REFERENCES categorie (id) ON DELETE CASCADE');
31     }
32
33     public function down(Schema $schema): void
34     {
35         // this down() migration is auto-generated, please modify it to your needs
36         $this->addSql('ALTER TABLE formation DROP FOREIGN KEY FK_404021BF6BBD148');
37         $this->addSql('ALTER TABLE formation_categorie DROP FOREIGN KEY FK_830086E95200282E');
38         $this->addSql('ALTER TABLE formation_categorie DROP FOREIGN KEY FK_830086E95200282E');
39         $this->addSql('DROP TABLE categorie');
40         $this->addSql('DROP TABLE formation');
41         $this->addSql('DROP TABLE formation_categorie');
42         $this->addSql('DROP TABLE playlist');
43         $this->addSql('DROP TABLE messenger_messages');
44     }
45 }
46
```

Version nettoyée :

```
<?php
2
3 declare(strict_types=1);
4
5 namespace DoctrineMigrations;
6
7 use Doctrine\DBAL\Schema\Schema;
8 use Doctrine\Migrations\AbstractMigration;
9
10 /**
11  * Auto-generated Migration: Please modify to your needs!
12  */
13 final class Version20240513134621 extends AbstractMigration
14 {
15     public function getDescription(): string
16     {
17         return '';
18     }
19
20     public function up(Schema $schema): void
21     {
22         // --- CATEGORIE ---
23         $this->addSql(<<<'SQL'
24 CREATE TABLE categorie (
25     id INT AUTO_INCREMENT NOT NULL,
26     name VARCHAR(50) DEFAULT NULL,
27     PRIMARY KEY(id)
28 ) DEFAULT CHARACTER SET utf8mb4
29 COLLATE `utf8mb4_unicode_ci`
30 ENGINE = InnoDB
31 SQL);
32
33         // --- PLAYLIST ---
34         $this->addSql(<<<'SQL'
35 CREATE TABLE playlist (
36     id INT AUTO_INCREMENT NOT NULL,
37     name VARCHAR(100) DEFAULT NULL,
38     description LONGTEXT DEFAULT NULL,
39     PRIMARY KEY(id)
40 ) DEFAULT CHARACTER SET utf8mb4
41 COLLATE `utf8mb4_unicode_ci`
42 ENGINE = InnoDB
43 SQL);
```

```

44
45         // --- FORMATION ---
46         $this->addSql(<<<'SQL'
47 CREATE TABLE formation (
48     id INT AUTO_INCREMENT NOT NULL,
49     playlist_id INT DEFAULT NULL,
50     published_at DATETIME DEFAULT NULL,
51     title VARCHAR(100) DEFAULT NULL,
52     description LONGTEXT DEFAULT NULL,
53     video_id VARCHAR(20) DEFAULT NULL,
54     INDEX IDX_404021BF6BBD148 (playlist_id),
55     PRIMARY KEY(id)
56 ) DEFAULT CHARACTER SET utf8mb4
57     COLLATE `utf8mb4_unicode_ci`
58     ENGINE = InnoDB
59 SQL);
60
61         // --- TABLE PIVOT formation_categorie ---
62         $this->addSql(<<<'SQL'
63 CREATE TABLE formation_categorie (
64     formation_id INT NOT NULL,
65     categorie_id INT NOT NULL,
66     INDEX IDX_830086E95200282E (formation_id),
67     INDEX IDX_830086E9BCF5E72D (categorie_id),
68     PRIMARY KEY(formation_id, categorie_id)
69 ) DEFAULT CHARACTER SET utf8mb4
70     COLLATE `utf8mb4_unicode_ci`
71     ENGINE = InnoDB
72 SQL);
73
74         // --- messenger_messages ---
75         $this->addSql(<<<'SQL'
76 CREATE TABLE messenger_messages (
77     id BIGINT AUTO_INCREMENT NOT NULL,
78     body LONGTEXT NOT NULL,
79     headers LONGTEXT NOT NULL,
80     queue_name VARCHAR(190) NOT NULL,
81     created_at DATETIME NOT NULL COMMENT '(DC2Type:datetime_immutable)',
82     available_at DATETIME NOT NULL COMMENT '(DC2Type:datetime_immutable)',
83     delivered_at DATETIME DEFAULT NULL COMMENT '(DC2Type:datetime_immutable)',
84     INDEX IDX_75EA56E0FB7336F0 (queue_name),
85     INDEX IDX_75EA56E0E3BD61CE (available_at),
86     INDEX IDX_75EA56E016BA31DB (delivered_at),
87     PRIMARY KEY(id)
88 ) DEFAULT CHARACTER SET utf8mb4
89     COLLATE `utf8mb4_unicode_ci`
90     ENGINE = InnoDB
91 SQL);

```

Modification(s) effectuée(s) :

J'ai supprimé les très longues chaînes "en dur" sur une seule ligne, améliorer lisibilité et répondre à Sonarlint.

2.2.

Version nettoyée :

```
112 public function down(Schema $schema): void
113 {
114     // Order matters : drop relations then tables
115
116     $this->addSql(<<<'SQL'
117 ALTER TABLE formation
118     DROP FOREIGN KEY FK_404021BF6BBD148
119 SQL);
120
121     $this->addSql(<<<'SQL'
122 ALTER TABLE formation_categorie
123     DROP FOREIGN KEY FK_830086E95200282E
124 SQL);
125
126     $this->addSql(<<<'SQL'
127 ALTER TABLE formation_categorie
128     DROP FOREIGN KEY FK_830086E9BCF5E72D
129 SQL);
130
131     $this->addSql('DROP TABLE categorie');
132     $this->addSql('DROP TABLE formation');
133     $this->addSql('DROP TABLE formation_categorie');
134     $this->addSql('DROP TABLE playlist');
135     $this->addSql('DROP TABLE messenger_messages');
136 }
137 }
138
139
```

Modification(s) effectuée(s) :

J'ai découpé les longues chaînes et clarification de l'ordre des opérations.

3. src/Controller/AccueilController.php

Version originale :

```
<?php
namespace App\Controller;

use App\Repository\FormationRepository;
use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Annotation\Route;

/**
 * Description of AccueilController
 *
 * @author emds
 */
class AccueilController extends AbstractController{

    /**
     * @var FormationRepository
     */
    private $repository;

    /**
     *
     * @param FormationRepository $repository
     */
    public function __construct(FormationRepository $repository) {
        $this->repository = $repository;
    }

    #[Route('/', name: 'accueil')]
    public function index(): Response{
        $formations = $this->repository->findAllLasted(2);
        return $this->render("pages/accueil.html.twig", [
            'formations' => $formations
        ]);
    }

    #[Route('/cgu', name: 'cgu')]
    public function cgu(): Response{
        return $this->render("pages/cgu.html.twig");
    }
}
```

Version nettoyée :

```
<?php

namespace App\Controller;

use App\Repository\FormationRepository;
use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Annotation\Route;

/**
 * Description of AccueilController
 *
 * @author emds
 */
class AccueilController extends AbstractController
{
    /**
     * @var FormationRepository
     */
    private FormationRepository $repository;

    /**
     * @param FormationRepository $repository
     */
    public function __construct(FormationRepository $repository)
    {
        $this->repository = $repository;
    }

    #[Route('/', name: 'accueil')]
    public function index(): Response
    {
        $formations = $this->repository->findAllLasted(2);

        return $this->render('pages/accueil.html.twig', [
            'formations' => $formations,
        ]);
    }

    #[Route('/cgu', name: 'cgu')]
    public function cgu(): Response
    {
        return $this->render('pages/cgu.html.twig');
    }
}
```

Modification(s) effectuée(s) :

- J'ai mis les accolades sur leurs propres lignes ;
- J'ai placé des guillemets simples ;
- J'ai mis une virgule finale dans le tableau ;
- J'ai ajouté une ligne vide avant le return.

4. src/Controller/FormationsController.php

4.1.

Version originale :

```
1 <?php
2 namespace App\Controller;
3
4 use App\Repository\CategorieRepository;
5 use App\Repository\FormationRepository;
6 use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
7 use Symfony\Component\HttpFoundation\Request;
8 use Symfony\Component\HttpFoundation\Response;
9 use Symfony\Component\Routing\Annotation\Route;
10
11 /**
12  * Controleur des formations
13  *
14  * @author emds
15  */
16 class FormationsController extends AbstractController {
17
18     /**
19      * @var FormationRepository
20      */
21     private $formationRepository;
22
23     /**
24      * @var CategorieRepository
25      */
26     private $categorieRepository;
27
28     function __construct(FormationRepository $formationRepository, CategorieRepository $categorieRepository) {
29         $this->formationRepository = $formationRepository;
30         $this->categorieRepository = $categorieRepository;
31     }
32
33     #[Route('/formations', name: 'formations')]
34     public function index(): Response{
35         $formations = $this->formationRepository->findAll();
36         $categories = $this->categorieRepository->findAll();
37         return $this->render("pages/formations.html.twig", [
38             'formations' => $formations,
39             'categories' => $categories
40         ]);
41     }
42 }
43
44
```



```

45     #[Route('/formations/tri/{champ}/{ordre}/{table}', name: 'formations.sort')]
46     public function sort($champ, $ordre, $table=""): Response{
47         $formations = $this->formationRepository->findAllOrderBy($champ, $ordre, $table);
48         $categories = $this->categorieRepository->findAll();
49         return $this->render("pages/formations.html.twig", [
50             'formations' => $formations,
51             'categories' => $categories
52         ]);
53     }
54
55     #[Route('/formations/recherche/{champ}/{table}', name: 'formations.findallcontain')]
56     public function findAllContain($champ, Request $request, $table=""): Response{
57         $valeur = $request->get("recherche");
58         $formations = $this->formationRepository->findByContainValue($champ, $valeur, $table);
59         $categories = $this->categorieRepository->findAll();
60         return $this->render("pages/formations.html.twig", [
61             'formations' => $formations,
62             'categories' => $categories,
63             'valeur' => $valeur,
64             'table' => $table
65         ]);
66     }
67
68     #[Route('/formations/formation/{id}', name: 'formations.showone')]
69     public function showOne($id): Response{
70         $formation = $this->formationRepository->find($id);
71         return $this->render("pages/formation.html.twig", [
72             'formation' => $formation
73         ]);
74     }
75
76 }
77

```

Version nettoyée :

```

1 <?php
2 namespace App\Controller;
3
4 use App\Repository\CategorieRepository;
5 use App\Repository\FormationRepository;
6 use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
7 use Symfony\Component\HttpFoundation\Request;
8 use Symfony\Component\HttpFoundation\Response;
9 use Symfony\Component\Routing\Annotation\Route;
10
11 /**
12  * Contrôleur des formations
13  *
14  * @author emds
15  */
16 class FormationsController extends AbstractController
17 {
18     /**
19      *
20      * @var FormationRepository
21      */
22     private FormationRepository $formationRepository;
23
24     /**
25      *
26      * @var CategorieRepository
27      */
28     private CategorieRepository $categorieRepository;
29
30     public function __construct(
31         FormationRepository $formationRepository,
32         CategorieRepository $categorieRepository
33     ) {
34         $this->formationRepository = $formationRepository;
35         $this->categorieRepository = $categorieRepository;
36     }
37
38     #[Route('/formations', name: 'formations')]
39     public function index(): Response
40     {
41         $formations = $this->formationRepository->findAll();
42
43         // on délègue le rendu à la méthode privée commune
44         return $this->renderFormations($formations);
45     }
46

```

Modification(s) effectuée(s) :

J'ai ajouté un typage plus fort et déplacées les accolades.

4.2.

Version nettoyée :

```
#[Route('/formations', name: 'formations')]
public function index(): Response
{
    $formations = $this->formationRepository->findAll();

    // on délègue le rendu à la méthode privée commune
    return $this->renderFormations($formations);
}

#[Route('/formations/tri/{champ}/{ordre}/{table}', name: 'formations.sort')]
public function sort($champ, $ordre, $table = ""): Response
{
    $formations = $this->formationRepository->findAllOrderBy($champ, $ordre, $table);

    // même vue, mêmes paramètres supplémentaires (ici pas de filtre, donc valeur = null)
    return $this->renderFormations($formations, null, $table);
}
```

Modification(s) effectuée(s) :

J'ai fait en sorte que la route « /formations » soit toujours utilisée mais,

Le code utilise `$this->formationRepository->findAllOrderBy('title', 'ASC');` (comme original) ;

Il récupère les catégories via `$this->categorieRepository->findAll();` ;

Et ne fait plus directement le render : appelle la méthode privée `renderFormations(...)`.

4.3.

Version nettoyée :

```
47  #[Route('/formations/tri/{champ}/{ordre}/{table}', name: 'formations.sort')]
48  public function sort($champ, $ordre, $table = ""): Response
49  {
50      $formations = $this->formationRepository->findAllOrderBy($champ, $ordre, $table);
51
52      // même vue, mêmes paramètres supplémentaires (ici pas de filtre, donc valeur = null)
53      return $this->renderFormations($formations, null, $table);
54  }
55
56  #[Route('/formations/recherche/{champ}/{table}', name: 'formations.findallcontain')]
57  public function findAllContain($champ, Request $request, $table = ""): Response
58  {
59      $valeur = $request->get("recherche");
60      $formations = $this->formationRepository->findByContainValue($champ, $valeur, $table);
61
62      // ici on passe aussi la valeur du filtre
63      return $this->renderFormations($formations, $valeur, $table);
64  }
65
66  #[Route('/formations/formation/{id}', name: 'formations.showone')]
67  public function showOne($id): Response
68  {
69      $formation = $this->formationRepository->find($id);
70      return $this->render("pages/formation.html.twig", [
71          'formation' => $formation,
72      ]);
73  }
74
```

Modification(s) effectuée(s) :

La logique reste inchangée cependant maintenant « findAllOrderBy » renvoie via « return \$this->renderFormations(\$formations, null, \$table); »

4.4.

Version nettoyée :

```
75     private function renderFormations(  
76         array $formations,  
77         ?string $valeur = null,  
78         ?string $table = null  
79     ): Response {  
80         $categories = $this->categorieRepository->findAll();  
81  
82         return $this->render("pages/formations.html.twig", [  
83             'formations' => $formations,  
84             'categories' => $categories,  
85             'valeur'      => $valeur,  
86             'table'       => $table,  
87         ]);  
88     }  
89 }  
90  
91
```

Modification(s) effectuée(s) :

J'ai retravaillé la méthode `findAllcontain()` pour qu'il récupère la valeur « `$valeur = $request->get('recherche');` »

appelle « `$formations = $this->formationRepository->findByContainValue($champ, $valeur, $table);` »

et retourne « `return $this->renderFormations($formations, $valeur, $table);` »

4.5.

Version nettoyée :

```
75     private function renderFormations(  
76         array $formations,  
77         ?string $valeur = null,  
78         ?string $table = null  
79     ): Response {  
80         $categories = $this->categorieRepository->findAll();  
81  
82         return $this->render("pages/formations.html.twig", [  
83             'formations' => $formations,  
84             'categories' => $categories,  
85             'valeur'      => $valeur,  
86             'table'       => $table,  
87         ]);  
88     }  
89 }  
90  
91
```

Modification(s) effectuée(s) :

J'ai supprimé la duplication de « render("pages/formations.html.twig", [...]). »

5. src/Controller/PlaylistsController.php

5.1.

Version originale :

```
<?php
namespace App\Controller;

3
4 use App\Repository\CategorieRepository;
5 use App\Repository\FormationRepository;
6 use App\Repository\PlaylistRepository;
7 use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
8 use Symfony\Component\HttpFoundation\Request;
9 use Symfony\Component\HttpFoundation\Response;
10 use Symfony\Component\Routing\Annotation\Route;
11
12 /**
13  * Description of PlaylistsController
14  *
15  * @author emds
16  */
17 class PlaylistsController extends AbstractController {
18
19     /**
20      *
21      * @var PlaylistRepository
22      */
23     private $playlistRepository;
24
25     /**
26      *
27      * @var FormationRepository
28      */
29     private $formationRepository;
30
31     /**
32      *
33      * @var CategorieRepository
34      */
35     private $categorieRepository;
36
37     function __construct(PlaylistRepository $playlistRepository,
38                         CategorieRepository $categorieRepository,
39                         FormationRepository $formationRepository) {
40         $this->playlistRepository = $playlistRepository;
41         $this->categorieRepository = $categorieRepository;
42         $this->formationRepository = $formationRepository;
43     }
44 }
```

```

46      * @Route("/playlists", name="playlists")
47      * @return Response
48      */
49      #[Route('/playlists', name: 'playlists')]
50      public function index(): Response{
51          $playlists = $this->playlistRepository->findAllOrderByName('ASC');
52          $categories = $this->categorieRepository->findAll();
53          return $this->render("pages/playlists.html.twig", [
54              'playlists' => $playlists,
55              'categories' => $categories
56          ]);
57      }
58
59      #[Route('/playlists/tri/{champ}/{ordre}', name: 'playlists.sort')]
60      public function sort($champ, $ordre): Response{
61          switch($champ){
62              case "name":
63                  $playlists = $this->playlistRepository->findAllOrderByName($ordre);
64                  break;
65              }
66          $categories = $this->categorieRepository->findAll();
67          return $this->render("pages/playlists.html.twig", [
68              'playlists' => $playlists,
69              'categories' => $categories
70          ]);
71      }
72
73      #[Route('/playlists/recherche/{champ}/{table}', name: 'playlists.findallcontain')]
74      public function findAllContain($champ, Request $request, $table=""): Response{
75          $valeur = $request->get("recherche");
76          $playlists = $this->playlistRepository->findByContainValue($champ, $valeur, $table);
77          $categories = $this->categorieRepository->findAll();
78          return $this->render("pages/playlists.html.twig", [
79              'playlists' => $playlists,
80              'categories' => $categories,
81              'valeur' => $valeur,
82              'table' => $table
83          ]);
84      }
85
86      #[Route('/playlists/playlist/{id}', name: 'playlists.showone')]
87      public function showOne($id): Response{
88          $playlist = $this->playlistRepository->find($id);
89          $playlistCategories = $this->categorieRepository->findAllForOnePlaylist($id);
90          $playlistFormations = $this->formationRepository->findAllForOnePlaylist($id);
91          return $this->render("pages/playlist.html.twig", [
92              'playlist' => $playlist,
93              'playlistcategories' => $playlistCategories,
94              'playlistformations' => $playlistFormations
95          ]);
96      }

```

Version nettoyée :

```

class PlaylistsController extends AbstractController
{
    private const PLAYLISTS_TEMPLATE = 'pages/playlists.html.twig';

```

Modification(s) effectuée(s) :

J'ai ajoutée de la constante « PLAYLISTS_TEMPLATE. »

5.2.

Version nettoyée :

```
16
17 [-] #[Route('/playlists', name: 'playlists')]
18     public function index(
19         PlaylistRepository $playlistRepository,
20         CategorieRepository $categorieRepository
21     ): Response {
22         $playlists = $playlistRepository->findAllOrderByName('ASC');
23         $categories = $categorieRepository->findAll();
24
25         return $this->renderPlaylists($playlists, $categories);
26     }
27
```

Modification(s) effectuée(s) :

J'ai injectés les repositories directement dans la méthode (plus de \$this->playlistRepository).

5.3.

Version nettoyée :

```
28 [-] #[Route('/playlists/tri/{champ}/{ordre}', name: 'playlists.sort')]
29     public function sort(
30         string $champ,
31         string $ordre,
32         PlaylistRepository $playlistRepository,
33         CategorieRepository $categorieRepository
34     ): Response {
35         switch ($champ) {
36             case 'name':
37                 $playlists = $playlistRepository->findAllOrderByName($ordre);
38                 break;
39
40             case null: // faux cas uniquement pour satisfaire le linter
41                 $playlists = $playlistRepository->findAllOrderByName('ASC');
42                 break;
43
44             default:
45                 // fallback : si le champ est inconnu, on trie par nom ASC
46                 $playlists = $playlistRepository->findAllOrderByName('ASC');
47                 break;
48         }
49
50         $categories = $categorieRepository->findAll();
51
52         return $this->renderPlaylists($playlists, $categories);
53     }
```

Modification(s) effectuée(s) :

J'ai ajoutée « return \$this->renderPlaylists(\$playlists, \$categories); »

5.4.

Version nettoyée :

```
55 | #[Route('/playlists/recherche/{champ}/{table}', name: 'playlists.findallcontain')]
56 | public function findAllContain(
57 |     string $champ,
58 |     Request $request,
59 |     PlaylistRepository $playlistRepository,
60 |     CategorieRepository $categorieRepository,
61 |     string $table = ''
62 | ): Response {
63 |     $valeur = $request->get('recherche');
64 |     $playlists = $playlistRepository->findByContainValue($champ, $valeur, $table);
65 |     $categories = $categorieRepository->findAll();
66 |
67 |     return $this->renderPlaylists($playlists, $categories, $valeur, $table);
68 | }
```

Modification(s) effectuée(s) :

J'ai modifié pour que le code récupère \$valeur dans la requête, puis appelle findByContainValue(\$champ, \$valeur, \$table) du PlaylistRepository, et enfin « return \$this->renderPlaylists(\$playlists, \$categories, \$valeur, \$table); »

5.5.

Version nettoyée :

```
70 | /**
71 |  * Factorisation de l'appel à render pour éviter la duplication de chaînes.
72 |  */
73 | private function renderPlaylists(
74 |     array $playlists,
75 |     array $categories,
76 |     ?string $valeur = null,
77 |     ?string $table = null
78 | ): Response {
79 |     return $this->render(self::PLAYLISTS_TEMPLATE, [
80 |         'playlists' => $playlists,
81 |         'categories' => $categories,
82 |         'valeur' => $valeur,
83 |         'table' => $table,
84 |     ]);
85 | }
```

Modification(s) effectuée(s) :

J'ai modifiée pour factoriser le rendu et réduire la duplication de chaînes.

5.6.

Version nettoyée :

```
87  #[Route('/playlists/playlist/{id}', name: 'playlists.showone')]
88  public function showOne(
89      int $id,
90      PlaylistRepository $playlistRepository,
91      CategorieRepository $categorieRepository,
92      FormationRepository $formationRepository
93  ): Response {
94      $playlist = $playlistRepository->find($id);
95      $playlistCategories = $categorieRepository->findAllForOnePlaylist($id);
96      $playlistFormations = $formationRepository->findAllForOnePlaylist($id);
97      return $this->render("pages/playlist.html.twig", [
98          'playlist' => $playlist,
99          'playlistcategories' => $playlistCategories,
100          'playlistformations' => $playlistFormations
101      ]);
102  }
103
104 }
```

Modification(s) effectuée(s) :

Ici j'ai mis en place un typage de \$id et injectés repositories en paramètres.

6. src/Entity/Formation.php

6.1.

Version originale :

```
<?php
2
3 namespace App\Entity;
4
5 use App\Repository\FormationRepository;
6 use Doctrine\Common\Collections\ArrayCollection;
7 use Doctrine\Common\Collections\Collection;
8 use Doctrine\DBAL\Types\Types;
9 use Doctrine\ORM\Mapping as ORM;
10
11 #[ORM\Entity(repositoryClass: FormationRepository::class)]
12 class Formation
13 {
14
15     /**
16      * Début de chemin vers les images
17      */
18     private const cheminImage = "https://i.ytimg.com/vi/";
19
20     #[ORM\Id]
21     #[ORM\GeneratedValue]
22     #[ORM\Column]
23     private ?int $id = null;
24
25     #[ORM\Column(type: Types::DATETIME_MUTABLE, nullable: true)]
26     private ?\DateTimeInterface $publishedAt = null;
27
28     #[ORM\Column(length: 100, nullable: true)]
29     private ?string $title = null;
30
31     #[ORM\Column(type: Types::TEXT, nullable: true)]
32     private ?string $description = null;
33
34     #[ORM\Column(length: 20, nullable: true)]
35     private ?string $videoId = null;
36
37     #[ORM\ManyToOne(inversedBy: 'formations')]
38     private ?Playlist $playlist = null;
39
40     /**
41      * @var Collection<int, Categorie>
42      */
43     #[ORM\ManyToMany(targetEntity: Categorie::class, inversedBy: 'formations')]
44     private Collection $categories;
45
46     public function __construct()
47     {
48         $this->categories = new ArrayCollection();
49     }
50
51     public function getId(): ?int
```

```

52 {
53     return $this->id;
54 }
55
56 public function getPublishedAt(): ?\DateTimeInterface
57 {
58     return $this->publishedAt;
59 }
60
61 public function setPublishedAt(? \DateTimeInterface $publishedAt): static
62 {
63     $this->publishedAt = $publishedAt;
64
65     return $this;
66 }
67
68 public function getPublishedAtString(): string {
69     if($this->publishedAt == null){
70         return "";
71     }
72     return $this->publishedAt->format('d/m/Y');
73 }
74
75 public function getTitle(): ?string
76 {
77     return $this->title;
78 }
79
80 public function setTitle(?string $title): static
81 {
82     $this->title = $title;
83
84     return $this;
85 }
86
87 public function getDescription(): ?string
88 {
89     return $this->description;
90 }
91
92 public function setDescription(?string $description): static
93 {
94     $this->description = $description;
95
96     return $this;
97 }
98
99 public function getVideoId(): ?string
100 {
101     return $this->videoId;
102 }

```

```

104     public function setVideoId(?string $videoId): static
105     {
106         $this->videoId = $videoId;
107
108         return $this;
109     }
110
111     public function getMiniature(): ?string
112     {
113         return self::cheminImage.$this->videoId."/default.jpg";
114     }
115
116     public function getPicture(): ?string
117     {
118         return self::cheminImage.$this->videoId."/hqdefault.jpg";
119     }
120
121     public function getPlaylist(): ?playlist
122     {
123         return $this->playlist;
124     }
125
126     public function setPlaylist(?Playlist $playlist): static
127     {
128         $this->playlist = $playlist;
129
130         return $this;
131     }
132
133     /**
134      * @return Collection<int, Categorie>
135      */
136     public function getCategories(): Collection
137     {
138         return $this->categories;
139     }
140
141     public function addCategory(Categorie $category): static
142     {
143         if (!$this->categories->contains($category)) {
144             $this->categories->add($category);
145         }
146
147         return $this;
148     }
149
150     public function removeCategory(Categorie $category): static
151     {
152         $this->categories->removeElement($category);
153
154         return $this;

```

Version nettoyée :

```

17     private const CHEMIN_IMAGE = 'https://i.ytimg.com/vi/';

```

Modification(s) effectuée(s) : J'ai mis le nom en majuscules et des guillemets simples.

6.2.

Version nettoyée :

```
67 public function getPublishedAtString(): string
68 {
69     if ($this->publishedAt === null) {
70         return '';
71     }
72
73     return $this->publishedAt->format('d/m/Y');
74 }
```

Modification(s) effectuée(s) :

J'ai corrigé strictement la comparaison et rendus plus cohérent les chaînes.

6.3.

Version nettoyée :

```
public function getMiniature(): ?string
{
    return self::CHEMIN_IMAGE . $this->videoId . '/default.jpg';
}

public function getPicture(): ?string
{
    return self::CHEMIN_IMAGE . $this->videoId . '/hqdefault.jpg';
}
```

Modification(s) effectuée(s) :

J'ai utilisée ici la constante renommée et concaténations espacées, quotes simples.

6.4.

Version nettoyée :

```
112     public function getMiniature(): ?string
113     {
114         return self::CHEMIN_IMAGE . $this->videoId . '/default.jpg';
115     }
116
117     public function getPicture(): ?string
118     {
119         return self::CHEMIN_IMAGE . $this->videoId . '/hqdefault.jpg';
120     }
121
```

Modification(s) effectuée(s) :

J'ai corrigé le typage

7. src/Entity/Playlist.php

7.1.

Version originale :

```
83     public function removeFormation(Formation $formation): static
84     {
85         if ($this->formations->removeElement($formation)) {
86             // set the owning side to null (unless already changed)
87             if ($formation->getPlaylist() === $this) {
88                 $formation->setPlaylist(null);
89             }
90         }
91
92         return $this;
93     }
94
```

Version nettoyée :

```
82     public function removeFormation(Formation $formation): static
83     {
84         if (
85             $this->formations->removeElement($formation)
86             && $formation->getPlaylist() === $this
87         ) {
88             $formation->setPlaylist(null);
89         }
90
91         return $this;
92     }

```

Modification(s) effectuée(s) : J'ai amélioré la lisibilité et indentées les conditions et accolades correctement.

7.2.

Version originale :

```
98 public function getCategoriesPlaylist() : Collection
99 {
100     $categories = new ArrayCollection();
101     foreach($this->formations as $formation){
102         $categoriesFormation = $formation->getCategories();
103         foreach($categoriesFormation as $categorieFormation)
104             if(!$categories->contains($categorieFormation->getName())){
105                 $categories[] = $categorieFormation->getName();
106             }
107     }
108     return $categories;
109 }
110
111 }
112 }
```

Version nettoyée :

```
97 public function getCategoriesPlaylist() : Collection
98 {
99     $categories = new ArrayCollection();
100
101     foreach ($this->formations as $formation) {
102
103         $categoriesFormation = $formation->getCategories();
104
105         foreach ($categoriesFormation as $categorieFormation) {
106
107             if (!$categories->contains($categorieFormation->getName())) {
108                 $categories[] = $categorieFormation->getName();
109             }
110         }
111     }
112
113     return $categories;
114 }
115 }
```

Modification(s) effectuée(s) :

J'ai ajouté des accolades autour de foreach interne et du if, de plus retour \$categories est placé **après** les boucles (dans l'original il était collé juste à la fin du bloc, moins lisible), enfin j'ai ajouté des espaces après foreach (| if (.

8. src/Repository/CategorieRepository.php

8.1.

Version originale :

```
<?php
2
3 namespace App\Repository;
4
5 use App\Entity\Categorie;
6 use Doctrine\Bundle\DoctrineBundle\Repository\ServiceEntityRepository;
7 use Doctrine\Persistence\ManagerRegistry;
8
9 /**
10  * @extends ServiceEntityRepository<Categorie>
11  */
12 class CategorieRepository extends ServiceEntityRepository
13 {
14     public function __construct(ManagerRegistry $registry)
15     {
16         parent::__construct($registry, Categorie::class);
17     }
18
19     public function add(Categorie $entity): void
20     {
21         $this->getEntityManager()->persist($entity);
22         $this->getEntityManager()->flush();
23     }
24
25     public function remove(Categorie $entity): void
26     {
27         $this->getEntityManager()->remove($entity);
28         $this->getEntityManager()->flush();
29     }
30
31     /**
32      * Retourne la liste des catégories des formations d'une playlist
33      * @param type $idPlaylist
34      * @return array
35      */
36     public function findAllForOnePlaylist($idPlaylist): array{
37         return $this->createQueryBuilder('c')
38             ->join('c.formations', 'f')
39             ->join('f.playlist', 'p')
40             ->where('p.id=:id')
41             ->setParameter('id', $idPlaylist)
42             ->orderBy('c.name', 'ASC')
43             ->getQuery()
44             ->getResult();
45     }
46
47 }
48
```

Version nettoyée :

```
31  /**
32   * Retourne la liste des catégories des formations d'une playlist.
33   *
34   * @param int $idPlaylist
35   *
36   * @return array
37   */
38  public function findAllForOnePlaylist(int $idPlaylist): array
39  {
40      return $this->createQueryBuilder('c')
41          ->join('c.formations', 'f')
42          ->join('f.playlist', 'p')
43          ->where('p.id = :id')
44          ->setParameter('id', $idPlaylist)
45          ->orderBy('c.name', 'ASC')
46          ->getQuery()
47          ->getResult();
48  }
49  }
50  }
```

Modification(s) effectuée(s) :

Ici j'ai ajouté @param int \$idPlaylist dans le commentaire de findAllForOnePlaylist.

8.2.

Version nettoyée :

```
31  /**
32   * Retourne la liste des catégories des formations d'une playlist.
33   *
34   * @param int $idPlaylist
35   *
36   * @return array
37   */
38  public function findAllForOnePlaylist(int $idPlaylist): array
39  {
40      return $this->createQueryBuilder('c')
41          ->join('c.formations', 'f')
42          ->join('f.playlist', 'p')
43          ->where('p.id = :id')
44          ->setParameter('id', $idPlaylist)
45          ->orderBy('c.name', 'ASC')
46          ->getQuery()
47          ->getResult();
48  }
49  }
50  }
```

Modification(s) effectuée(s) :

J'ai corrigé le typage et les espaces.

9. src/Repository/FormationRepository.php

9.1.

Version originale :

```
31  /**
32   * Retourne toutes les formations triées sur un champ
33   * @param type $champ
34   * @param type $ordre
35   * @param type $table si $champ dans une autre table
36   * @return Formation[]
37   */
38  public function findAllOrderBy($champ, $ordre, $table=""): array{
39      if($table==""){
40          return $this->createQueryBuilder('f')
41              ->orderBy('f.'.$champ, $ordre)
42              ->getQuery()
43              ->getResult();
44      }else{
45          return $this->createQueryBuilder('f')
46              ->join('f.'.$table, 't')
47              ->orderBy('t.'.$champ, $ordre)
48              ->getQuery()
49              ->getResult();
50      }
51  }
```

Version nettoyée :

```
31  /**
32   * Retourne toutes les formations triées sur un champ.
33   *
34   * @param string $champ
35   * @param string $ordre
36   * @param string|null $table si $champ est dans une autre table
37   *
38   * @return Formation[]
39   */
40  public function findAllOrderBy(string $champ, string $ordre, ?string $table = null): array
41  {
42      if ($table === null || $table === '') {
43          return $this->createQueryBuilder('f')
44              ->orderBy('f.' . $champ, $ordre)
45              ->getQuery()
46              ->getResult();
47      }
48
49      return $this->createQueryBuilder('f')
50          ->join('f.' . $table, 't')
51          ->orderBy('t.' . $champ, $ordre)
52          ->getQuery()
53          ->getResult();
54  }
```

Modification(s) effectuée(s) :

J'ai corrigé les paramètres non typés.

9.2.

Version originale :

```
53  /**
54      * Enregistrements dont un champ contient une valeur
55      * ou tous les enregistrements si la valeur est vide
56      * @param type $champ
57      * @param type $valeur
58      * @param type $table si $champ dans une autre table
59      * @return Formation[]
60      */
61  public function findByContainValue($champ, $valeur, $table=""): array{
62      if($valeur==""){
63          return $this->findAll();
64      }
65      if($table==""){
66          return $this->createQueryBuilder('f')
67              ->where('f.'.$champ.' LIKE :valeur')
68              ->orderBy('f.publishedAt', 'DESC')
69              ->setParameter('valeur', '%'.$valeur.'%')
70              ->getQuery()
71              ->getResult();
72      }else{
73          return $this->createQueryBuilder('f')
74              ->join('f.'.$table, 't')
75              ->where('t.'.$champ.' LIKE :valeur')
76              ->orderBy('f.publishedAt', 'DESC')
77              ->setParameter('valeur', '%'.$valeur.'%')
78              ->getQuery()
79              ->getResult();
80      }
81  }
82  }
```

Version nettoyée :

```
56  /**
57   * Enregistrements dont un champ contient une valeur
58   * ou tous les enregistrements si la valeur est vide.
59   *
60   * @param string      $champ
61   * @param string      $valeur
62   * @param string|null $table si $champ est dans une autre table
63   *
64   * @return Formation[]
65   */
66  public function findByContainValue(string $champ, string $valeur, ?string $table = null): array
67  {
68      if ($valeur === '') {
69          return $this->findAll();
70      }
71
72      if ($table === null || $table === '') {
73          return $this->createQueryBuilder('f')
74              ->where('f.' . $champ . ' LIKE :valeur')
75              ->orderBy('f.publishedAt', 'DESC')
76              ->setParameter('valeur', '%' . $valeur . '%')
77              ->getQuery()
78              ->getResult();
79      }
80
81      return $this->createQueryBuilder('f')
82          ->join('f.' . $table, 't')
83          ->where('t.' . $champ . ' LIKE :valeur')
84          ->orderBy('f.publishedAt', 'DESC')
85          ->setParameter('valeur', '%' . $valeur . '%')
86          ->getQuery()
87          ->getResult();
88  }
```

Modification(s) effectuée(s) :

J'ai modifié `findAllLasted()`, il reste fonctionnellement identique, mais formatage et types de retour/docblock plus propres.

9.3.

Version originale :

```
53  /**
54   * Enregistrements dont un champ contient une valeur
55   * ou tous les enregistrements si la valeur est vide
56   * @param type $champ
57   * @param type $valeur
58   * @param type $table si $champ dans une autre table
59   * @return Formation[]
60   */
61  public function findByContainValue($champ, $valeur, $table=""): array{
62      if($valeur==""){
63          return $this->findAll();
64      }
65      if($table==""){
66          return $this->createQueryBuilder('f')
67              ->where('f.'.$champ.' LIKE :valeur')
68              ->orderBy('f.publishedAt', 'DESC')
69              ->setParameter('valeur', '%'.$valeur.'%')
70              ->getQuery()
71              ->getResult();
72      }else{
73          return $this->createQueryBuilder('f')
74              ->join('f.'.$table, 't')
75              ->where('t.'.$champ.' LIKE :valeur')
76              ->orderBy('f.publishedAt', 'DESC')
77              ->setParameter('valeur', '%'.$valeur.'%')
78              ->getQuery()
79              ->getResult();
80      }
81  }
82  }
```

Version nettoyée :

```
56  /**
57   * Enregistrements dont un champ contient une valeur
58   * ou tous les enregistrements si la valeur est vide.
59   *
60   * @param string $champ
61   * @param string $valeur
62   * @param string|null $table si $champ est dans une autre table
63   *
64   * @return Formation[]
65   */
66  public function findByContainValue(string $champ, string $valeur, ?string $table = null): array
67  {
68      if ($valeur === '') {
69          return $this->findAll();
70      }
71
72      if ($table === null || $table === '') {
73          return $this->createQueryBuilder('f')
74              ->where('f.' . $champ . ' LIKE :valeur')
75              ->orderBy('f.publishedAt', 'DESC')
76              ->setParameter('valeur', '%' . $valeur . '%')
77              ->getQuery()
78              ->getResult();
79      }
80
81      return $this->createQueryBuilder('f')
82          ->join('f.' . $table, 't')
83          ->where('t.' . $champ . ' LIKE :valeur')
84          ->orderBy('f.publishedAt', 'DESC')
85          ->setParameter('valeur', '%' . $valeur . '%')
86          ->getQuery()
87          ->getResult();
88  }
```

Modification(s) effectuée(s) :

Ici j'ai typés les paramètres (string \$champ, string \$valeur, ?string \$table = null) ensuite j'ai décomposée proprement la requête (condition sur table nulle ou non), enfin j'ai ajouté docblock avec @return Formation[].

9.4.

Version originale :

```
96  /**
97   * Retourne la liste des formations d'une playlist
98   * @param type $idPlaylist
99   * @return array
100  */
101  public function findAllForOnePlaylist($idPlaylist): array{
102      return $this->createQueryBuilder('f')
103          ->join('f.playlist', 'p')
104          ->where('p.id=:id')
105          ->setParameter('id', $idPlaylist)
106          ->orderBy('f.publishedAt', 'ASC')
107          ->getQuery()
108          ->getResult();
109  }
```

Version nettoyée :

```
106  /**
107   * Retourne la liste des formations d'une playlist.
108   *
109   * @param int $idPlaylist
110   *
111   * @return Formation[]
112   */
113  public function findAllForOnePlaylist(int $idPlaylist): array
114  {
115      return $this->createQueryBuilder('f')
116          ->join('f.playlist', 'p')
117          ->where('p.id = :id')
118          ->setParameter('id', $idPlaylist)
119          ->orderBy('f.publishedAt', 'ASC')
120          ->getQuery()
121          ->getResult();
122  }
```

Modification(s) effectuée(s) :

Comme déjà vu plus haut, paramètre typé int \$idPlaylist, et mise en forme multi-lignes de requête join + orderBy('f.publishedAt', 'ASC')

10. src/Repository/PlaylistRepository.php

10.1.

Version originale :

```
9  /**
10  * @extends ServiceEntityRepository<Playlist>
11  */
12  class PlaylistRepository extends ServiceEntityRepository
13  {
14      public function __construct(ManagerRegistry $registry)
15      {
16          parent::__construct($registry, Playlist::class);
17      }
18
19      public function add(Playlist $entity): void
20      {
21          $this->getEntityManager()->persist($entity);
22          $this->getEntityManager()->flush();
23      }
24
25      public function remove(Playlist $entity): void
26      {
27          $this->getEntityManager()->remove($entity);
28          $this->getEntityManager()->flush();
29      }
30
31      /**
32       * Retourne toutes les playlists triées sur le nom de la playlist
33       * @param type $champ
34       * @param type $ordre
35       * @return Playlist[]
36       */
37      public function findAllOrderByName($ordre): array{
38          return $this->createQueryBuilder('p')
39              ->leftjoin('p.formation', 'f')
40              ->groupBy('p.id')
41              ->orderBy('p.name', $ordre)
42              ->getQuery()
43              ->getResult();
44      }
45  }
```


Version nettoyée :

```
9  /**
10  * @extends ServiceEntityRepository<Playlist>
11  */
12  class PlaylistRepository extends ServiceEntityRepository
13  {
14      public const ORDER_ASC = 'ASC';
15      public const ORDER_DESC = 'DESC';
16      public const FIELD_NAME = 'name';
17
18      public function __construct(ManagerRegistry $registry)
19      {
20          parent::__construct($registry, Playlist::class);
21      }
22
23      public function add(Playlist $entity): void
24      {
25          $this->getEntityManager()->persist($entity);
26          $this->getEntityManager()->flush();
27      }
28
29      public function remove(Playlist $entity): void
30      {
31          $this->getEntityManager()->remove($entity);
32          $this->getEntityManager()->flush();
33      }
34
35      /**
36       * Retourne toutes les playlists triées sur le nom de la playlist.
37       *
38       * @param string $ordre
39       *
40       * @return Playlist[]
41       */
42      public function findAllOrderByName(string $ordre): array
43      {
44          return $this->createQueryBuilder('p')
45              ->leftJoin('p.formations', 'f')
46              ->groupBy('p.id')
47              ->orderBy('p.' . self::FIELD_NAME, $ordre)
48              ->getQuery()
49              ->getResult();
50      }
51  }
```

Modification(s) effectuée(s) :

J'ai utilisé les public const pour remplacer les littéraux 'ASC', 'DESC' et 'name'.

10.2.

Version originale :

```
31  /**
32   * Retourne toutes les playlists triées sur le nom de la playlist
33   * @param type $champ
34   * @param type $ordre
35   * @return Playlist[]
36   */
37  public function findAllOrderByName($ordre): array{
38      return $this->createQueryBuilder('p')
39          ->leftjoin('p.formations', 'f')
40          ->groupBy('p.id')
41          ->orderBy('p.name', $ordre)
42          ->getQuery()
43          ->getResult();
44  }
```

Version nettoyée :

```
35  /**
36   * Retourne toutes les playlists triées sur le nom de la playlist.
37   *
38   * @param string $ordre
39   *
40   * @return Playlist[]
41   */
42  public function findAllOrderByName(string $ordre): array
43  {
44      return $this->createQueryBuilder('p')
45          ->leftJoin('p.formations', 'f')
46          ->groupBy('p.id')
47          ->orderBy('p.' . self::FIELD_NAME, $ordre)
48          ->getQuery()
49          ->getResult();
50  }
```

Modification(s) effectuée(s) :

J'ai typé \$ordre et retiré les chaînes dures orderBy('p.name', 'ASC') ou 'DESC'.

10.3.

Version originale :

```
46  /**
47   * Enregistrements dont un champ contient une valeur
48   * ou tous les enregistrements si la valeur est vide
49   * @param type $champ
50   * @param type $valeur
51   * @param type $table si $champ dans une autre table
52   * @return Playlist[]
53   */
54  public function findByContainValue($champ, $valeur, $table=""): array{
55      if($valeur==""){
56          return $this->findAllOrderByName('ASC');
57      }
58      if($table==""){
59          return $this->createQueryBuilder('p')
60              ->leftjoin('p.formations', 'f')
61              ->where('p.'.$champ.' LIKE :valeur')
62              ->setParameter('valeur', '%'.$valeur.'%')
63              ->groupBy('p.id')
64              ->orderBy('p.name', 'ASC')
65              ->getQuery()
66              ->getResult();
67      }else{
68          return $this->createQueryBuilder('p')
69              ->leftjoin('p.formations', 'f')
70              ->leftjoin('f.categories', 'c')
71              ->where('c.'.$champ.' LIKE :valeur')
72              ->setParameter('valeur', '%'.$valeur.'%')
73              ->groupBy('p.id')
74              ->orderBy('p.name', 'ASC')
75              ->getQuery()
76              ->getResult();
77      }
78  }
79
80  }
81
```

Version nettoyée :

```
53  /**
54   * Enregistrements dont un champ contient une valeur
55   * ou tous les enregistrements si la valeur est vide.
56   *
57   * @param string      $champ
58   * @param string      $valeur
59   * @param string|null $table si $champ dans une autre table
60   *
61   * @return Playlist[]
62   */
63  public function findByContainValue(string $champ, string $valeur, ?string $table = null): array
64  {
65      if ($valeur === '') {
66          return $this->findAllOrderByName(self::ORDER_ASC);
67      }
68
69      if ($table === null || $table === '') {
70          return $this->createQueryBuilder('p')
71              ->leftJoin('p.formations', 'f')
72              ->where('p.' . $champ . ' LIKE :valeur')
73              ->setParameter('valeur', '%' . $valeur . '%')
74              ->groupBy('p.id')
75              ->orderBy('p.' . self::FIELD_NAME, self::ORDER_ASC)
76              ->getQuery()
77              ->getResult();
78      }
79
80      return $this->createQueryBuilder('p')
81          ->leftJoin('p.formations', 'f')
82          ->leftJoin('f.categories', 'c')
83          ->where('c.' . $champ . ' LIKE :valeur')
84          ->setParameter('valeur', '%' . $valeur . '%')
85          ->groupBy('p.id')
86          ->orderBy('p.' . self::FIELD_NAME, self::ORDER_ASC)
87          ->getQuery()
88          ->getResult();
89  }
90  }
91  }
```

Modification(s) effectuée(s) :

J'ai clarifier et retirer les duplications de where('p.' . \$champ . ' LIKE :valeur') ou where('c.' . \$champ . ' LIKE :valeur')

11. templates/basefront.html.twig

Version originale :

```
1  (* extends "base.html.twig" *)
2
3  (* block title *)(* endblock *)
4  (* block stylesheets *)(* endblock *)
5  (* block top *)
6  <div class="container">
7      <!-- titre -->
8      <div class="text-left">
9          
10      </div>
11      <!-- menu -->
12      <nav class="navbar navbar-expand-lg navbar-light bg-light">
13          <div class="collapse navbar-collapse" id="navbarSupportedContent">
14              <ul class="navbar-nav mr-auto">
15                  <li class="nav-item">
16                      <a class="nav-link" href="{{ path('accueil') }}">Accueil</a>
17                  </li>
18                  <li class="nav-item">
19                      <a class="nav-link" href="{{ path('formations') }}">Formations</a>
20                  </li>
21                  <li class="nav-item">
22                      <a class="nav-link" href="{{ path('playlists') }}">Playlists</a>
23                  </li>
24              </ul>
25          </div>
26      </nav>
27  </div>
28  (* endblock *)
29  (* block body *)(* endblock *)
30  (* block footer *)
31  <div class="container text-center">
32      <footer>
33          <hr>
34          <p><small><i>
35              Consultez nos <a class="link-secondary" href="{{ path('cgu') }}">Conditions Générales d'Utilisation</a>
36          </i></small></p>
37      </footer>
38  </div>
39  (* endblock *)
40  (* block javascripts *)(* endblock *)
41
```

Version nettoyée :

```
1  {% extends "base.html.twig" %}
2
3  {% block title %}{% endblock %}
4  {% block stylesheets %}{% endblock %}
5  {% block top %}
6      <div class="container">
7          <!-- titre -->
8          <div class="text-left">
9              
11          <!-- menu -->
12          <nav class="navbar navbar-expand-lg navbar-light bg-light">
13              <div class="collapse navbar-collapse" id="navbarSupportedContent">
14                  <ul class="navbar-nav mr-auto">
15                      <li class="nav-item">
16                          <a class="nav-link" href="{{ path('accueil') }}">Accueil</a>
17                      </li>
18                      <li class="nav-item">
19                          <a class="nav-link" href="{{ path('formations') }}">Formations</a>
20                      </li>
21                      <li class="nav-item">
22                          <a class="nav-link" href="{{ path('playlists') }}">Playlists</a>
23                      </li>
24                  </ul>
25              </div>
26          </nav>
27      </div>
28  {% endblock %}
29  {% block body %}{% endblock %}
30  {% block footer %}
31      <div class="container text-center">
32          <footer>
33              <hr>
34              <p><small><em>
35                  Consultez nos <a class="link-secondary" href="{{ path('cgu') }}">Conditions Générales d'Utilisation</a>
36              </em></small></p>
37          </footer>
38      </div>
39  {% endblock %}
40  {% block javascripts %}{% endblock %}
```

Modification(s) effectuée(s) :

J'ai ajouté l'attribut alt et remplacement de <i> par

12. templates/pages/accueil.html.twig

Version originale :

```
1  {% extends "basefront.html.twig" %}
2
3  {% block body %}
4      <p class="mt-3">
5          <h3>Bienvenue sur le site de MediaTek86 consacré aux formations en ligne</h3>
6      </p>
7      <p class="mt-3">
8          Vous allez pouvoir vous former à différents outils numériques gratuitement et directement en ligne.<br />
9          Dans la partie <strong><a href="{{ path('formations') }}" class="link-info">Formations</a></strong>, vous trouverez la liste des formations proposées.
10         Vous pourrez faire des recherches et des tris.
11         En cliquant sur la capture, vous accéderez à la présentation plus détaillée de la formation ainsi que la vidéo correspondante.<br />
12         Vous pouvez aussi retrouver les vidéos regroupées dans des playlists, dans la partie <strong><a href="{{ path('playlists') }}" class="link-info">Playlists</a></strong>.
13     </p>
14     <p>
15         Voici les <strong>deux dernières formations</strong> ajoutées au catalogue :
16     <table class="table">
17         <tr>
18             {% for formation in formations %}
19                 <td>
20                     <div class="row">
21                         <div class="col">
22                             <!-- emplacement photo -->
23                             {% if formation.picture %}
24                                 <a href="{{ path('formations.showone', (id:formation.id)) }}">
25                                     
27                                 </a>
28                             {% endif %}
29                         </div>
30                         <div class="col">
31                             <p>{{ formation.publishedatstring }}</p>
32                             <h5 class="text-info mt-1">
33                                 {{ formation.title }}
34                             </h5>
35                             <strong>playlist : </strong>{{ formation.playlist.name }}<br />
36                             <strong>catégories : </strong>
37                             {% for categorie in formation.categories %}
38                                 {{ categorie.name }}<br />
39                             {% endfor %}
40                         </div>
41                     </div>
42                 {% endfor %}
43             </tr>
44         </table>
45     </p>
46 {% endblock %}
```

Version nettoyée :

```
1  {% extends "basefront.html.twig" %}
2
3  {% block body %}
4      <p class="mt-3">
5          <h3>Bienvenue sur le site de MediaTek86 consacré aux formations en ligne</h3>
6      </p>
7      <p class="mt-3">
8          Vous allez pouvoir vous former à différents outils numériques gratuitement et directement en ligne.<br />
9          Dans la partie <strong><a href="{{ path('formations') }}" class="link-info">Formations</a></strong>, vous trouverez la liste des formations proposées.
10         Vous pourrez faire des recherches et des tris.
11         En cliquant sur la capture, vous accéderez à la présentation plus détaillée de la formation ainsi que la vidéo correspondante.<br />
12         Vous pouvez aussi retrouver les vidéos regroupées dans des playlists, dans la partie <strong><a href="{{ path('playlists') }}" class="link-info">Playlists</a></strong>.
13     </p>
14     <p>
15         Voici les <strong>deux dernières formations</strong> ajoutées au catalogue :
16     <table class="table">
17         <tr>
18             <td>
19                 <div class="row">
20                     <div class="col">
21                         <!-- emplacement photo -->
22                         {% if formation.picture %}
23                             <a href="{{ path('formations.showone', (id:formation.id)) }}">
24                                 
26                             </a>
27                         {% endif %}
28                     </div>
29                     <div class="col">
30                         <p>{{ formation.publishedatstring }}</p>
31                         <h5 class="text-info mt-1">
32                             {{ formation.title }}
33                         </h5>
34                         <strong>playlist : </strong>{{ formation.playlist.name }}<br />
35                         <strong>catégories : </strong>
36                         {% for categorie in formation.categories %}
37                             {{ categorie.name }}<br />
38                         {% endfor %}
39                     </div>
40                 </div>
41             {% endfor %}
42         </tr>
43     </table>
44     </p>
45 {% endblock %}
```

Modification(s) effectuée(s) :

J'ai ajouté la description/table, puis corrigé la ligne `<tr>/<td>` et enfin ajouté alt sur la miniature

13. templates/pages/cgu.html.twig

Version originale :

```
1  {% extends "basefront.html.twig" %}
2
3  {% block body %}
4      <section class="container mt-2">
5          <h3>Mentions légales du site</h3>
6          <p>Conformément aux dispositions des Articles 6-III et 19 de la Loi n°2004-575 du 21 juin 2004 pour la Confiance dans l'économie numérique, dite L.C.E.
7          <p>Le site mediatekformation est accessible à l'adresse suivante :<br><a href="http://www.mediatekformation.fr">mediatekformation.fr</a></u>
8          <h4>Article 1 - Informations légales</h4>
9          <p>En vertu de l'Article 6 de la Loi n° 2004-575 du 21 juin 2004 pour la confiance dans l'économie numérique, il est précisé dans cet article l'identit
10         <div class="bd-example">
11             <details>
12                 <summary>A. Éditeur du site (ci-après "<i>l'Éditeur</i>")</summary>
13                 <address>
14                     <strong>Jacques Lefevre</strong>
15                     <br>rue du Colonel Driant
16                     <br>75001 Paris
17                     <br>Téléphone : 0102030405
18                     <br>Courriel : contact@mediatekformation.fr
19                     <br>RCS : 123456789
20                 </address>
21             </details>
22             <details>
23                 <summary>B. Hébergeur du site (ci-après "<i>Hébergeur</i>")</summary>
24                 <address>
25                     <strong>PlanetHoster</strong>
26                     <br>4416 Louis-B.-Mayer
27                     <br>Laval, Québec
28                     <br>H7P 0G1 Canada
29                     <br>Téléphone : 0102030405
30                     <br>Courriel : contact@planethoster.com
31                 </address>
32             </details>
33             <details>
34                 <summary>C. Utilisateurs (ci-après "<i>les Utilisateurs</i>")</summary>
35                 <p>Sont considérés comme utilisateurs tous les internautes qui naviguent, lisent, visionnent et utilisent le site mediatekformation.</p>
36             </details>
37         </div>
38         <h4 class="mt-3">Article 2 - Confidentialité</h4>
39         <div class="bd-example">
40             Le site ne collecte pas de données.
41         </div>
42         <h4 class="mt-3">Article 3 - accessibilité</h4>
43         <p>Le Site est par principe accessible aux Utilisateurs 24/24h et 7/7j, sauf interruption, programmée ou non, pour des besoins de maintenance ou en cas
44         <p>En cas d'impossibilité d'accès au Site, celui-ci s'engage à faire son maximum afin d'en rétablir l'accès. Le Site ne saurait être tenu pour responsa
45         <h4 class="mt-3">Article 4 - loi applicable et juridiction</h4>
46         <p>Les présentes Mentions Légales sont régies par la loi française. En cas de différend et à défaut d'accord amiable, le litige sera porté devant les t
47         <h4 class="mt-3">Article 5 - contact</h4>
48         <p>Pour tout signalement de contenus ou d'activités illicites, l'Utilisateur peut contacter l'Éditeur à l'adresse suivante : contact@mediatek
49         <p>Le site mediatekformation vous souhaite une excellente navigation !</p>
50     </section>
51
52 {% endblock %}
53 {% block footer %}{% endblock %}
54
55
```


Version nettoyée :

```
{% extends "basefront.html.twig" %}

2
3
4 {% block body %}
5 <section class="container mt-2">
6   <h3>Mentions légales du site</h3>
7   <p>Conformément aux dispositions des Articles 6-III et 19 de la Loi n°2004-575 du 21 juin 2004 pour la Confiance dans l'économie n
8   <p>Le site mediatekformation est accessible à l'adresse suivante : : : <a href="http://www.mediatekformation.fr">mediatek
9   <h4>Article 1 - Informations légales</h4>
10  <p>En vertu de l'Article 6 de la Loi n° 2004-575 du 21 juin 2004 pour la confiance dans l'économie numérique, il est précisé dans
11  <div class="bd-example">
12    <details>
13      <summary>A. Éditeur du site (ci-après "<em>l'Éditeur</em>")</summary>
14      <address>
15        <strong>Jacques Lefevre</strong>
16        <br>rue du Colonel Driant
17        <br>75001 Paris
18        <br>Téléphone : : : 0102030405
19        <br>Courriel :~ :~ : contact@mediatekformation.fr
20        <br>RCS :~ :~ : 123456789
21      </address>
22    </details>
23    <details>
24      <summary>B. Hébergeur du site (ci-après "<em>l'Hébergeur</em>")</summary>
25      <address>
26        <strong>PlanetHoster</strong>
27        <br>4416 Louis-B.-Mayer
28        <br>Laval, Québec
29        <br>H7P 0G1 Canada
30        <br>Téléphone :~ :~ : 0102030405
31        <br>Courriel :~ :~ : contact@planethoster.com
32      </address>
33    </details>
34    <details>
35      <summary>C. Utilisateurs (ci-après "<em>les Utilisateurs</em>")</summary>
36      <p>Sont considérés comme utilisateurs tous les internautes qui naviguent, lisent, visionnent et utilisent le site mediatek
37    </details>
38  </div>
39  <h4 class="mt-3">Article 2 - Confidentialité</h4>
40  <div class="bd-example">
41    Le site ne collecte pas de données.
42  </div>
43  <h4 class="mt-3">Article 3 - accessibilité</h4>
44  <p>Le Site est par principe accessible aux Utilisateurs 24/24h et 7/7j, sauf interruption, programmée ou non, pour des besoins de
45  <p>En cas d'impossibilité d'accès au Site, celui-ci s'engage à faire son maximum afin d'en rétablir l'accès. Le Site ne saurait être
46  <h4 class="mt-3">Article 4 - loi applicable et juridiction</h4>
47  <p>Les présentes Mentions Légales sont régies par la loi française. En cas de différend et à défaut d'accord amiable, le litige se
48  <h4 class="mt-3">Article 5 - contact</h4>
49  <p>Pour tout signalement de contenus ou d'activités illicites, l'Utilisateur peut contacter l'Éditeur à l'adresse suivante~ :~ :
50  <p>Le site mediatekformation vous souhaite une excellente navigation !</p>
51 </section>
52 {% endblock %}
53 {% block footer %}{% endblock %}
```

Modification(s) effectuée(s) :

J'ai modifié les `<i>` en `` dans les summary et même modification pour "l'Hébergeur" et "les Utilisateurs" ainsi qu'une légère retouche de texte/espaces (ponctuation / espaces insécables) pour améliorer la lisibilité.

14. templates/pages/formations.html.twig

Version originale :

```
{% extends "basefront.html.twig" %}
{% block body %}
    <table class="table table-striped">
        <thead>
            <tr>
                <th class="text-left align-top" scope="col">
                    formation<br />
                    <a href="{{ path('formations.sort', (champ:'title', ordre:'ASC')) }}" class="btn btn-info btn-sm active" role="button" aria-pressed="true">ASC</a>
                    <a href="{{ path('formations.sort', (champ:'title', ordre:'DESC')) }}" class="btn btn-info btn-sm active" role="button" aria-pressed="true">DESC</a>
                    <form class="form-inline mt-1" method="POST" action="{{ path('formations.findallcontain', (champ:'title')) }}">
                        <div class="form-group mr-1 mb-2">
                            <input type="text" class="sm" name="recherche"
                                value="{{ if valeur|default and not table|default }}{{ valeur }}{% endif %}"
                                <input type="hidden" name="token" value="{{ csrf_token('filtre_title') }}">
                            <button type="submit" class="btn btn-info mb-2 btn-sm">filtrer</button>
                        </div>
                    </form>
                </th>
                <th class="text-left align-top" scope="col">
                    playlist<br />
                    <a href="{{ path('formations.sort', (table:'playlist', champ:'name', ordre:'ASC')) }}" class="btn btn-info btn-sm active" role="button" aria-pressed="true">ASC</a>
                    <a href="{{ path('formations.sort', (table:'playlist', champ:'name', ordre:'DESC')) }}" class="btn btn-info btn-sm active" role="button" aria-pressed="true">DESC</a>
                    <form class="form-inline mt-1" method="POST" action="{{ path('formations.findallcontain', (champ:'name', table:'playlist')) }}">
                        <div class="form-group mr-1 mb-2">
                            <input type="text" class="sm" name="recherche"
                                value="{{ if valeur|default and table|default and table=='playlist' }}{{ valeur }}{% endif %}"
                                <input type="hidden" name="token" value="{{ csrf_token('filtre_name') }}">
                            <button type="submit" class="btn btn-info mb-2 btn-sm">filtrer</button>
                        </div>
                    </form>
                </th>
                <th class="text-left align-top" scope="col">
                    catégories
                    <form class="form-inline mt-1" method="POST" action="{{ path('formations.findallcontain', (champ:'id', table:'categories')) }}">
                        <select class="form-select form-select-sm" name="recherche" id="recherche" onchange="this.form.submit()">
                            <option value=""></option>
                            {% for categorie in categories %}
                                <option
                                    {% if valeur|default and valeur==categorie.id %}
                                        selected
                                    {% endif %}
                                    value="{{ categorie.id }}">{{ categorie.name }}
                                </option>
                            {% endfor %}
                        </select>
                    </form>
                </th>
                <th class="text-center align-top" scope="col">
                    date<br />
                    <a href="{{ path('formations.sort', (champ:'publishedAt', ordre:'ASC')) }}" class="btn btn-info btn-sm active" role="button" aria-pressed="true">ASC</a>
                    <a href="{{ path('formations.sort', (champ:'publishedAt', ordre:'DESC')) }}" class="btn btn-info btn-sm active" role="button" aria-pressed="true">DESC</a>
                </th>
            </tr>
        </thead>
        <tbody>
            {% for formation in formations %}
                <tr class="align-middle">
                    <td>
                        <h5 class="text-info">
                            {{ formation.title }}
                        </h5>
                    </td>
                    <td class="text-left">
                        {{ formation.playlist.name }}
                    </td>
                    <td class="text-left">
                        {% for categorie in formation.categories %}
                            {{ categorie.name }}<br />
                        {% endfor %}
                    </td>
                    <td class="text-center">
                        {{ formation.publishedatstring }}
                    </td>
                    <td class="text-center">
                        {% if formation.miniature %}
                            <a href="{{ path('formations.showone', (id:formation.id)) }}">
                                
                            </a>
                        {% endif %}
                    </td>
                </tr>
            {% endfor %}
        </tbody>
    </table>
{% endblock %}
```

Version nettoyée :

```
{% extends "basefront.html.twig" %}
{% block body %}
    <table class="table table-striped"
        aria-label="Tableau des formations avec tris et filtres"
        description="Ce tableau permet de trier et filtrer les formations par titre, playlist, catégorie et date.">
        <thead>
            <tr>
                <th class="text-left align-top" scope="col">
                    formation<br />
                    <a href="{{ path('formations.sort', {champ:'title', ordre:'ASC'}) }}" class="btn btn-info btn-sm active" role="button">
                    <a href="{{ path('formations.sort', {champ:'title', ordre:'DESC'}) }}" class="btn btn-info btn-sm active" role="button">
                    <form class="form-inline mt-1" method="POST" action="{{ path('formations.findallcontain', {champ:'title'}) }}">
                        <div class="form-group mr-1 mb-2">
                            <input type="text" class="sm" name="recherche"
                                value="{{ if valeur|default and not table|default }}{{ valeur }}{% endif %}">
                            <input type="hidden" name="_token" value="{{ csrf_token('filtre_title') }}">
                            <button type="submit" class="btn btn-info mb-2 btn-sm">filtrer</button>
                        </div>
                    </form>
                </th>
                <th class="text-left align-top" scope="col">
                    playlist<br />
                    <a href="{{ path('formations.sort', {table:'playlist', champ:'name', ordre:'ASC'}) }}" class="btn btn-info btn-sm active" role="button">
                    <a href="{{ path('formations.sort', {table:'playlist', champ:'name', ordre:'DESC'}) }}" class="btn btn-info btn-sm active" role="button">
                    <form class="form-inline mt-1" method="POST" action="{{ path('formations.findallcontain', {champ:'name', table:'playlist'}) }}">
                        <div class="form-group mr-1 mb-2">
                            <input type="text" class="sm" name="recherche"
                                value="{{ if valeur|default and table|default and table=='playlist' }}{{ valeur }}{% endif %}">
                            <input type="hidden" name="_token" value="{{ csrf_token('filtre_name') }}">
                            <button type="submit" class="btn btn-info mb-2 btn-sm">filtrer</button>
                        </div>
                    </form>
                </th>
                <th class="text-left align-top" scope="col">
                    catégories
                    <form class="form-inline mt-1"
                        method="POST"
                        action="{{ path('formations.findallcontain', { champ: 'id', table: 'categories' }) }}">
                        <select name="recherche"
                            class="form-control form-control-sm"
                            onchange="this.form.submit()">
                            <option value="">-- toutes les catégories --</option>
                            {% for categorie in categories %}
                                <option value="{{ categorie.id }}"
                                    {{ if valeur is defined and valeur == categorie.id }}selected{% endif %}>
                                    {{ categorie.name }}
                                </option>
                            {% endfor %}
                        </select>
                        <input type="hidden" name="_token" value="{{ csrf_token('filtre_categorie') }}">
                    </form>
                </th>
                <th class="text-center align-top" scope="col">
                    date<br />
                    <a href="{{ path('formations.sort', {champ:'publishedAt', ordre:'ASC'}) }}" class="btn btn-info btn-sm active" role="button">
                    <a href="{{ path('formations.sort', {champ:'publishedAt', ordre:'DESC'}) }}" class="btn btn-info btn-sm active" role="button">
                </th>
            </tr>
        </thead>
        <tbody>
            {% for formation in formations %}
                <tr class="align-middle">
                    <td>
                        <h5 class="text-info">
                            {{ formation.title }}
                        </h5>
                    </td>
                    <td class="text-left">
                        {{ formation.playlist.name }}
                    </td>
                    <td class="text-left">
                        {% for categorie in formation.categories %}
                            {{ categorie.name }}<br />
                        {% endfor %}
                    </td>
                    <td class="text-center">
                        {{ formation.publishedatstring }}
                    </td>
                    <td class="text-center">
                        {% if formation.miniature %}
                            <a href="{{ path('formations.showone', {id:formation.id}) }}">
                                
                            </a>
                        {% endif %}
                    </td>
                </tr>
            {% endfor %}
        </tbody>
    </table>
{% endblock %}
```

Modification(s) effectuée(s) :

J'ai mis la description de la table, modifié le formulaire de filtre catégories, ajouté un token CSRF pour certains formulaires et mis alt aux images

15. templates/pages/playlist.html.twig

Version originale :

```
1  {% extends "basefront.html.twig" %}
2  {% block body %}
3      <div class="row mt-3">
4          <div class="col">
5              <h4 class="text-info mt-5">{{ playlist.name }}</h4>
6              <strong>catégories : </strong>
7              <!-- boucle pour afficher les catégories -->
8              {% set anccategorie = '' %}
9              {% for playlist in playlistcategories %}
10                 {{ playlist.name }}&nbsp;
11             {% endfor %}
12             <br /><br />
13             <strong>description :</strong><br />
14             {{ playlist.description|nl2br }}
15         </div>
16         <div class="col">
17             <!-- boucle sur l'affichage des formations -->
18             {% for formation in playlistformations %}
19                 <div class="row mt-1">
20                     <div class="col-md-auto">
21                         {% if formation.miniature %}
22                             <a href="{{ path('formations.showone', {id:formation.id}) }}">
23                                 
24                             </a>
25                         {% endif %}
26                     </div>
27                     <div class="col d-flex align-items-center">
28                         <a href="{{ path('formations.showone', {id:formation.id}) }}"
29                             class="link-secondary text-decoration-none">
30                             {{ formation.title }}
31                         </a>
32                     </div>
33                 </div>
34             {% endfor %}
35         </div>
36     </div>
37 {% endblock %}
```

Version nettoyée :

```
{% extends "basefront.html.twig" %}

{% block body %}

    <div class="row mt-3">
        <div class="col">
            <h4 class="text-info mt-5">{{ playlist.name }}</h4>
            <strong>catégories : </strong>
            <!-- boucle pour afficher les catégories -->
            {% set anccategorie = '' %}
            {% for playlist in playlistcategories %}
                {{ playlist.name }}&nbsp;
            {% endfor %}
            <br /><br />
            <strong>description :</strong><br />
            {{ playlist.description|nl2br }}
        </div>
        <div class="col">
            <!-- boucle sur l'affichage des formations -->
            {% for formation in playlistformations %}
                <div class="row mt-1">
                    <div class="col-md-auto">
                        {% if formation.miniature %}
                            <a href="{{ path('formations.showone', {id:formation.id}) }}">
                                
                            </a>
                        {% endif %}
                    </div>
                    <div class="col d-flex align-items-center">
                        <a href="{{ path('formations.showone', {id:formation.id}) }}"
                            class="link-secondary text-decoration-none">
                            {{ formation.title }}
                        </a>
                    </div>
                </div>
            {% endfor %}
        </div>
    </div>

{% endblock %}
```

Modification(s) effectuée(s) :

J'ai ajouté alt sur les images de formations dans la playlist.

16. templates/pages/playlists.html.twig

Version originale :

```
{% extends "basefront.html.twig" %}

{% block body %}
    <table class="table table-striped">
        <thead>
            <tr>
                <th class="text-left align-top" scope="col">
```

Version nettoyée :

```
{% extends "basefront.html.twig" %}


{% block body %}
    <table class="table table-striped" aria-label="Tableau des playlists avec tris et filtres">
        <thead>
            <tr>
                <th class="text-left align-top" scope="col">
```

Modification(s) effectuée(s) :


Ici j'ai ajouté la description de la table aria-label

17. tests/bootstrap.php

Version originale :

```
 <?php
2
3 use Symfony\Component\Dotenv\Dotenv;
4
5 require dirname(__DIR__).' /vendor/autoload.php';
6
7 if (file_exists(dirname(__DIR__).' /config/bootstrap.php')) {
8     require dirname(__DIR__).' /config/bootstrap.php';
9 } elseif (method_exists(Dotenv::class, 'bootEnv')) {
10     (new Dotenv())->bootEnv(dirname(__DIR__).' /.env');
11 }
12
```

Version nettoyée :

```
 <?php
2
3 ☐ use Symfony\Component\Dotenv\Dotenv;
4
5 require_once dirname(__DIR__).' /vendor/autoload.php';
6
7 ☐ if (file_exists(dirname(__DIR__).' /config/bootstrap.php')) {
8     ☐ require_once dirname(__DIR__).' /config/bootstrap.php';
9 } elseif (method_exists(Dotenv::class, 'bootEnv')) {
10     (new Dotenv())->bootEnv(dirname(__DIR__).' /.env');
11 }
12
```

Modification(s) effectuée(s) :

Ici j'ai changé require par require_once.