

Développement d'une application pour de la gestion d'évènements

ACCUEIL

CHANSON

CHANTEUR

CLIENT

FESTIVAL

ASSOCIATIONS

INSCRIPTION FESTIVAL

INSCRIPTION RAPIDE

Festivals

Id	Nom festival	Date debut	Lieu	Genre	Nb places	Voir	Modifier
1	Francofolies	13-07-2022 19:00	La Rochelle, France	Pop	1500	Voir	Modifier
2	TomorrowLand	22-07-2022 20:15	Belgique	Fous Furieux	1750	Voir	Modifier
3	Festival Rock in Rio	18-06-2022 16:45	Lisbonne, Portugal	Rock	1386	Voir	Modifier
4	Rock en Seine 2022	25-08-2022 15:30	Paris, France	Rock	850	Voir	Modifier



Symfony

BTS SERVICES INFORMATIQUES AUX ORGANISATIONS		SESSION 2022
ANNEXE 7-1-B : Fiche descriptive de réalisation professionnelle (recto)		
Épreuve E5 - Conception et développement d'applications (option SLAM) - Coefficient 4		

DESCRIPTION D'UNE RÉALISATION PROFESSIONNELLE		N° réalisation :
Nom, prénom : DOMAS-VASSEROT Jade		N° candidat : 02146788571
Épreuve ponctuelle <input checked="" type="checkbox"/>	Contrôle en cours de formation <input type="checkbox"/>	Date : 15 / 04 / 2022
Contexte de la réalisation professionnelle Une entreprise d'organisation d'événements souhaite une solution afin de préparer leur gestion de festivals. Nous avons donc créé une application web permettant de gérer les festivals, les chansons, les chanteurs et les client avec le Framework Symfony		
Intitulé de la réalisation professionnelle Développement d'une application pour de la gestion d'évènements		
Période de réalisation : 2eme année Lieu : Lyon Modalité : <input type="checkbox"/> Seul(e) <input checked="" type="checkbox"/> En équipe		
Compétences travaillées <input checked="" type="checkbox"/> Concevoir et développer une solution applicative <input type="checkbox"/> Assurer la maintenance corrective ou évolutive d'une solution applicative <input checked="" type="checkbox"/> Gérer les données		
Conditions de réalisation¹ (ressources fournies, résultats attendus) Ressources fournies : - Sujet de projet, Cours de symfony, accès à la documentation internet Résultats attendus : - Conception d'une base de données - Développement d'une application Symfony en MVC (CRUD)		
Description des ressources documentaires, matérielles et logicielles utilisées ² Logiciels : VS code WampServer (PhpMyAdmin, Apache) Accès internet Navigateur Chrome Installation de Composer Langages : Twig pour la vue SQL PHP avec le framework Symfony		
Modalités d'accès aux productions ³ et à leur documentation ⁴ GitHub: https://github.com/JadeDomasVasserot/festival		

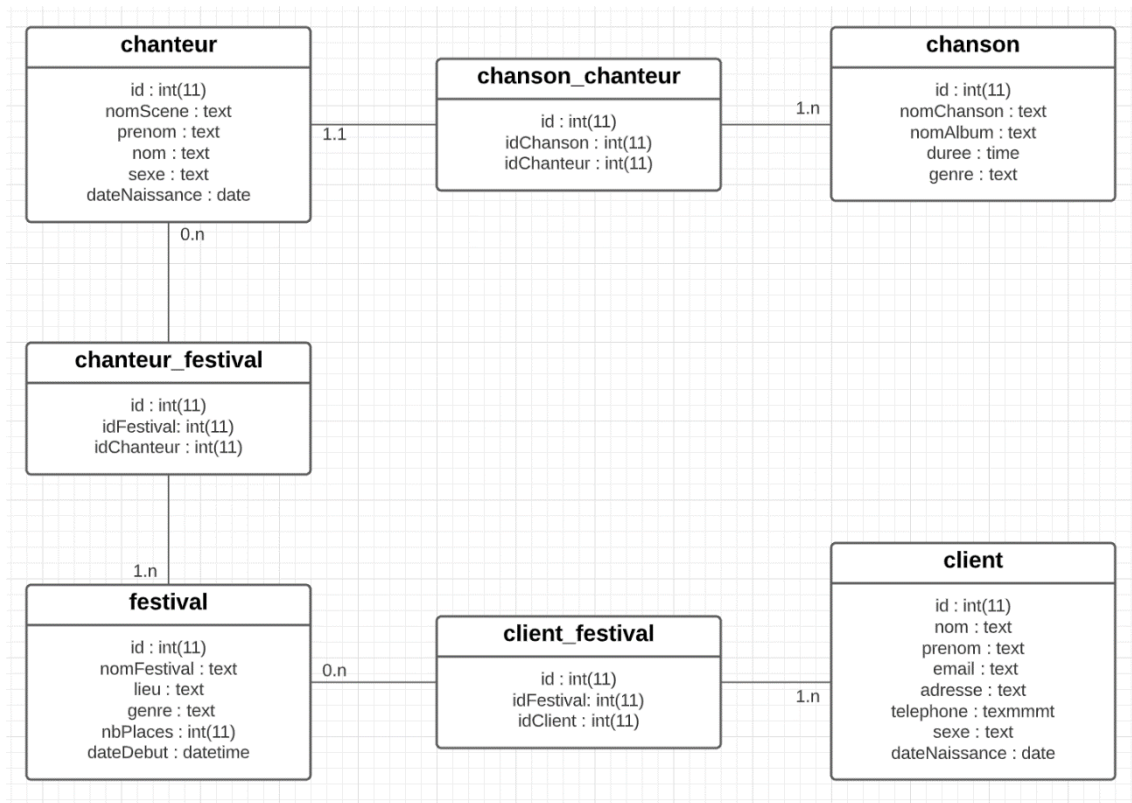
¹ En référence aux *conditions de réalisation et ressources nécessaires* du bloc « Conception et développement d'applications » prévues dans le référentiel de certification du BTS SIO.

² Les réalisations professionnelles sont élaborées dans un environnement technologique conforme à l'annexe II.E du référentiel du BTS SIO.

³ Conformément au référentiel du BTS SIO « Dans tous les cas, les candidats doivent se munir des outils et ressources techniques nécessaires au déroulement de l'épreuve. Ils sont seuls responsables de la disponibilité et de la mise en œuvre de ces outils et ressources. La circulaire nationale d'organisation précise les conditions matérielles de déroulement des interrogations et les pénalités à appliquer aux candidats qui ne se seraient pas munis des éléments nécessaires au déroulement de l'épreuve. ». Les éléments peuvent être un identifiant, un mot de passe, une adresse réticulaire (URL) d'un espace de stockage et de la présentation de l'organisation du stockage.

⁴ Lien vers la documentation complète, précisant et décrivant, si cela n'a été fait au verso de la fiche, la réalisation professionnelle, par exemple service fourni par la réalisation, interfaces utilisateurs, description des classes ou de la base de données.

Schéma de BDD :



Description de l'activité :

- Conception d'une BDD
- Développement d'une application Symfony

```
File Edit Selection View Go Run Terminal Help ChansonChanteurController.php - festival - Visual Studio Code

EXPLORER
  FESTIVAL
    > .idea
    > bin
    > config
    > migrations
    > public
    > sql
    > src
      > Controller
        ChansonChanteurController.php
        ChansonController.php
        ChanteurController.php
        ClientController.php
        ClientFestivalController.php
        FestivalController.php
        IndexController.php
      > Entity
      > Form
      > Repository
      > Kernel.php
    > templates
      > chanson
      > chanson_chanteur
      > chanteur
      > chanteur_festival
      > client
      > client_festival
      > festival
      > index
      > base.html.twig
      > index.html.twig
    > OUTLINE
    > TIMELINE

ChansonChanteurController.php
src > Controller > ChansonChanteurController.php
1 <?php
2
3 namespace App\Controller;
4
5 use App\Entity\ChansonChanteur;
6 use App\Form\ChansonChanteurType;
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  * @Route("/chanson/chanteur")
14  */
15 class ChansonChanteurController extends AbstractController
16 {
17     /**
18      * @Route("/", name="chanson_chanteur_index", methods={"GET"})
19      */
20     public function index(): Response
21     {
22         $chansonChanteurs = $this->getDoctrine()
23             ->getRepository(ChansonChanteur::class)
24             ->findAll();
25
26         return $this->render('chanson_chanteur/index.html.twig', [
27             'chanson_chanteurs' => $chansonChanteurs,
28         ]);
29     }
30
31     /**
32      * @Route("/new", name="chanson_chanteur_new", methods={"GET", "POST"})
33      */
34     public function new(Request $request): Response
35     {
36         $chansonChanteur = new ChansonChanteur();
37         $form = $this->createForm(ChansonChanteurType::class, $chansonChanteur);
38         $form->handleRequest($request);
39     }
40 }
```

Contexte :

Avec l'arrivée de l'été et la digitalisation de plus en plus commune des applications, une association d'organisations d'évènements a voulu digitaliser son processus de gestion de festivals.

Il nous demande le fait de pouvoir ajouter/modifier/supprimer :

- Un festival
 - Un chanteur
 - Un client
 - Une chanson
- Et de lier chaque information.

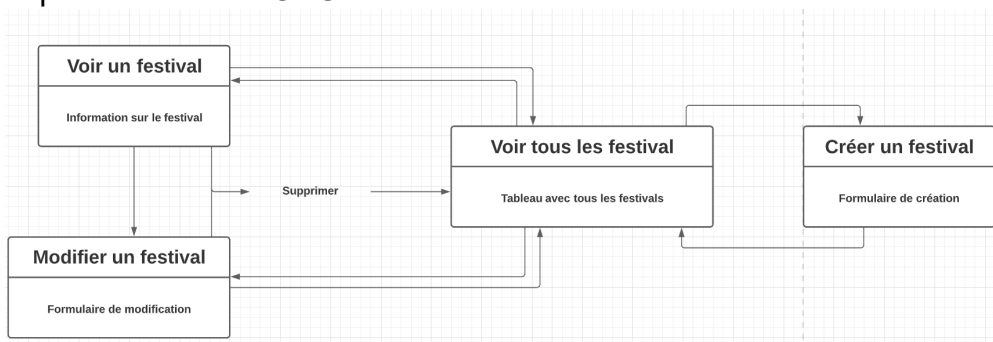
Réalisation :

Pour cela, il nous a fallu dans un premier temps concevoir le processus de l'application afin de comprendre les besoins de l'association.

L'association avait besoin de gérer les clients, chansons, festivals, chanteurs.

On a donc conçu la base de données (voir plus haut). Puis, nous avons décidé d'utiliser la méthode CRUD (Create Read Update Delete) pour gérer les données.

Exemple de process avec le CRUD :



Pour chaque entité dans notre table (voir BDD).

Nous avons par la suite créé un projet Symfony.

composer create-project symfony/website-skeleton

Il a fallu connecter notre base de données dans .env qui a été créée au préalable.

La base de donnée a été créer avant.

```
.env
# IMPORTANT: You MUST configure your server version, either here or in config/packages/doctrine.yaml
#
# DATABASE_URL="sqlite://%kernel.project_dir%/var/data.db"
27 DATABASE_URL="mysql://root:@127.0.0.1:3306/festival?serverVersion=5.7"
28
29 #DATABASE_URL="postgresql://db_user:db_password@127.0.0.1:5432/db_name?serverVersion=13&charset=utf8"
30
31 ###< doctrine/doctrine-bundle ###
32
33
```

Pour créer les entités à partir de la BDD puis les getters/setters puis les contrôleurs.

Commande utilisée :

php bin/console doctrine:mapping:import "App\Entity" annotation --path=src/Entity

php bin/console make:entity --regenerate App

php bin/console make:controller

Exemple d'entité générée :

```
1 <?php
2
3 namespace App\Entity;
4
5 use Doctrine\ORM\Mapping as ORM;
6
7 /**
8  * Festival
9  *
10  * @ORM\Table(name="festival")
11  * @ORM\Entity
12  */
13 class Festival
14 {
15     /**
16      * @var int
17      *
18      * @ORM\Column(name="id", type="integer", nullable=false)
19      * @ORM\Id
20      * @ORM\GeneratedValue(strategy="IDENTITY")
21      */
22     private $id;
23
24     /**
25      * @var string
26      *
27      * @ORM\Column(name="nomFestival", type="text", length=65535, nullable=false)
28      */
29     private $nomFestival;
30
31     /**
32      * @var \DateTime
33      *
34      * @ORM\Column(name="dateDebut", type="datetime", nullable=false)
35      */
36     private $dateDebut;
37
38     /**
39      * @var string
40      *
41      * @ORM\Column(name="lieu", type="text", length=65535, nullable=false)
42      */
43     private $lieu;
44
45     /**
46      * @var string
47      *
48      * @ORM\Column(name="genre", type="text", length=65535, nullable=false)
49      */
50     private $genre;
51
52     /**
53      * @var int
54      *
55      * @ORM\Column(name="nbPlaces", type="integer", nullable=false)
56      */
57     private $nbPlaces;
58
59     public function getId(): ?int //Retourne l'id
60     {
61         return $this->id;
62     }
63     public function getNomFestival(): ?string //Retourne le nom du festival
64     {
65         return $this->nomFestival;
66     }
67     public function setNomFestival(string $nomFestival): self //Change le nom du festival
68     {
69         $this->nomFestival = $nomFestival;
70         return $this;
71     }
72     public function getDateDebut(): ?\DateTimeInterface //Retourne la date de début
73     {
74         return $this->dateDebut;
75     }
76     public function setDateDebut(\DateTimeInterface $dateDebut): self //Change la date de début
77     {
78         $this->dateDebut = $dateDebut;
79         return $this;
80     }
81     public function getLieu(): ?string //Retourne le lieu
82     {
83         return $this->lieu;
84     }
85     public function setLieu(string $lieu): self //Change le lieu
86     {
87         $this->lieu = $lieu;
88         return $this;
89     }
90     public function getGenre(): ?string //Retourne le genre
91     {
92         return $this->genre;
93     }
94     public function setGenre(string $genre): self //Change le genre
95     {
96         $this->genre = $genre;
97         return $this;
98     }
99     public function getNbPlaces(): ?int //Retourne le nombre de places
100     {
101         return $this->nbPlaces;
102     }
103     public function setNbPlaces(int $nbPlaces): self //Change le nombre de place
104     {
105         $this->nbPlaces = $nbPlaces;
106         return $this;
107     }
108     public function __toString()
109     {
110         return $this->id.'-'.$this->nomFestival;
111     }
112 }
```

Après la création des entités nous avons fait les contrôleurs avec la méthode du CRUD.
Commande utilisée :

php bin/console make:crud

Exemple de contrôleur :

```
1 <?php
2
3 namespace App\Controller;
4 use App\Entity\Festival;
5 use App\Form\FestivalType;
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  * @Route("/festival")
12  */
13 class FestivalController extends AbstractController
14 {
15     /**
16      * @Route("/", name="festival_index", methods={"GET"})
17      */
18     public function index(): Response
19     {
20         $festivals = $this->getDoctrine()
21             ->getRepository(Festival::class)
22             ->findAll();
23
24         return $this->render('festival/index.html.twig', [
25             'festivals' => $festivals,
26         ]);
27     }
28
29     /**
30      * @Route("/new", name="festival_new", methods={"GET","POST"})
31      */
32     public function new(Request $request): Response
33     {
34         $festival = new Festival();
35         $form = $this->createForm(FestivalType::class, $festival);
36         $form->handleRequest($request);
37
38         if ($form->isSubmitted() && $form->isValid()) {
39             $entityManager = $this->getDoctrine()->getManager();
40             $entityManager->persist($festival);
41             $entityManager->flush();
42
43             return $this->redirectToRoute('festival_index', [], Response::HTTP_SEE_OTHER);
44         }
45
46         return $this->renderForm('festival/new.html.twig', [
47             'festival' => $festival,
48             'form' => $form,
49         ]);
50     }
51
52     /**
53      * @Route("/{id}/edit", name="festival_edit", methods={"GET","POST"})
54      */
55     public function edit(Request $request, Festival $festival): Response
56     {
57         $form = $this->createForm(FestivalType::class, $festival);
58         $form->handleRequest($request);
59
60         if ($form->isSubmitted() && $form->isValid()) {
61             $this->getDoctrine()->getManager()->flush();
62
63             return $this->redirectToRoute('festival_index', [], Response::HTTP_SEE_OTHER);
64         }
65
66         return $this->renderForm('festival/edit.html.twig', [
67             'festival' => $festival,
68             'form' => $form,
69         ]);
70     }
71
72     /**
73      * @Route("/{id}", name="festival_delete", methods={"POST"})
74      */
75     public function delete(Request $request, Festival $festival): Response
76     {
77         if ($this->isCsrfTokenValid('delete', $festival->getId(), $request->request->get('_token'))) {
78             $entityManager = $this->getDoctrine()->getManager();
79             $entityManager->remove($festival);
80             $entityManager->flush();
81
82             return $this->redirectToRoute('festival_index', [], Response::HTTP_SEE_OTHER);
83         }
84     }
85 }
```

Nous avons donc pu partir de cette base afin de faire les fonctionnalités voulues en modifiant les contrôleurs, routes, entités.
Comme la page d'index où l'on affiche tous les festivals.
En parallèle, nous avons fait les pages twig pour le visuel avec de l'HTML CSS.

Page d'index :

Festivals							
Id	Nom festival	Date debut	Lieu	Genre	Nb places	Voir	Modifier
1	Francofolies	13-07-2022 19:00	La Rochelle, France	Pop	1500	Voir	Modifier
2	Tomorrow, Land	22-07-2022 20:15	Belgique	Fous Furieux	1750	Voir	Modifier
3	Festival Rock in Rio	18-06-2022 16:45	Lisbonne, Portugal	Rock	1386	Voir	Modifier
4	Rock en Seine 2022	25-08-2022 15:30	Paris, France	Rock	850	Voir	Modifier

Présentation du projet :

[Téléchargement et aperçu de notre BDD](#)

Table	Action	Champs	Type	Interclassement	Taille	Partie
chanson	Parcourir Structure Rechercher Insérer Vider Supprimer	18	IntnoDB	utf8mb4_unicode_ci	16,0 Kio	-
chanson_chanteur	Parcourir Structure Rechercher Insérer Vider Supprimer	18	IntnoDB	utf8mb4_unicode_ci	48,0 Kio	-
chanteur	Parcourir Structure Rechercher Insérer Vider Supprimer	9	IntnoDB	utf8mb4_unicode_ci	16,0 Kio	-
chanteur_festival	Parcourir Structure Rechercher Insérer Vider Supprimer	2	IntnoDB	utf8mb4_unicode_ci	16,0 Kio	-
client	Parcourir Structure Rechercher Insérer Vider Supprimer	6	IntnoDB	utf8mb4_unicode_ci	16,0 Kio	-
client_festival	Parcourir Structure Rechercher Insérer Vider Supprimer	2	IntnoDB	utf8mb4_unicode_ci	16,0 Kio	-
festival	Parcourir Structure Rechercher Insérer Vider Supprimer	7	IntnoDB	utf8mb4_unicode_ci	16,0 Kio	-
7 tables	Somme	14	MySQL	utf8mb4_unicode_ci	208,0 Kio	100%

Page index d'un contrôleur :

Index de tous les festivals							
Id	Nomfestival	Datedebut	Lieu	Genre	Nbplaces	Voir	Modifier
1	Francofolies	13-07-2022 19:00	La Rochelle, France	Pop	1500	Voir	Modifier
2	Tomorrow, Land	22-07-2022 20:15	Belgique	Fous Furieux	1750	Voir	Modifier
3	Festival Rock in Rio	18-06-2022 16:45	Lisbonne, Portugal	Rock	1386	Voir	Modifier
4	Rock en Seine 2022	25-08-2022 15:30	Paris, France	Rock	850	Voir	Modifier

[Créer un festival](#)

Page de modification :

Editer votre festival

Nom Du Festival

Date Et Heure

Jul

13

2022

19

00

Lieu

Genre

Nombre De Places

[Enregistrer](#)

[Supprimer](#)

[Retour à la liste](#)

Page de création :

Créer un nouveau Festival

Nom Du Festival

Date Et Heure

Jan

1

2017

00

00

Lieu

Genre

Nombre De Places

[Sauvegarder](#)

[Retour à la liste](#)

Page pour voir une entité :

[ACCUEIL](#) [CHANSON](#) [CHANTEUR](#) [CLIENT](#) [FESTIVAL](#) [ASSOCIATIONS](#) [INSCRIPTION FESTIVAL](#) [INSCRIPTION RAPIDE](#)

Festival : Francofolies

Id	1
Nomfestival	Francofolies
Datedebut	13-07-2022 19:00
Lieu	La Rochelle, France
Genre	Pop
Nbplaces	1500

[Retour à l'index](#)[Editer](#)[Supprimer](#)