

Petit Marie-Camille

Compte rendu d'activité

MediatekFormation
2^{ème} année BTS SIO

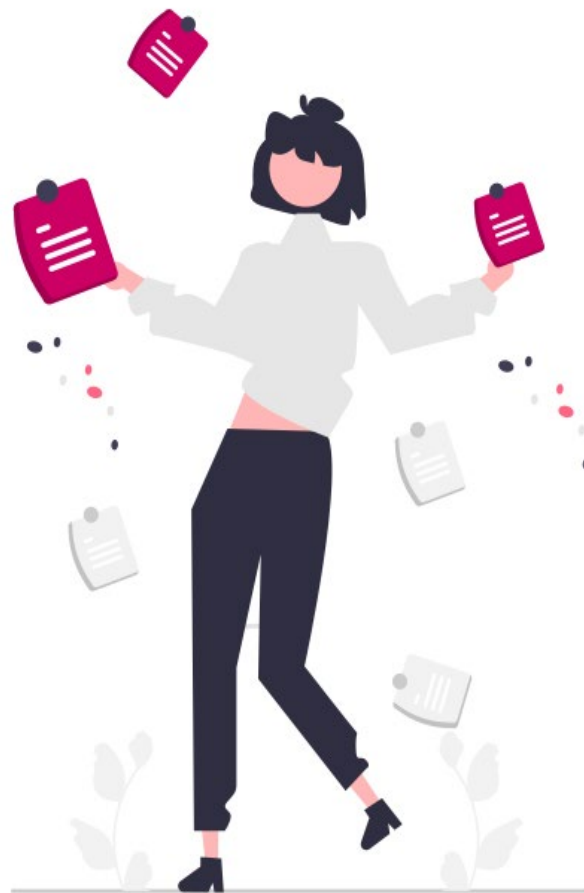
Contents

Contexte	3
Présentation de l'application existante	4
Missions	8
Mission 0 : préparer l'environnement de travail	9
Mission 1 : ajouter les niveaux.....	10
Mission 2 : coder la partie back-office	17
Mission 3	39
Mission bilan :	40
Bilan	44

Contexte

Afin de donner plus d'attractivités aux médiathèques, MediaTek86 le réseau des médiathèques de la Vienne, souhaite se développer en proposant des formations aux outils numériques et des autoformations en ligne. C'est dans ce but qu'elle a contacté la société InfoTech Services 86, une entreprise de Services Numériques (ESN) spécialisée dans le développement informatique, l'hébergement de site web, l'infogérance, la gestion de parc informatique et l'ingénierie système et réseau.

Vous travaillez en tant que technicien développeur junior pour l'ESN InfoTech Services 86 qui vient de remporter le marché pour différentes interventions au sein du réseau MediaTek86, dont celle d'ajouter certaines fonctionnalités dans la plateforme de formations en ligne.



Présentation de l'application existante

L'application existante était composée d'une page d'accueil, d'une page listant toutes les formations ainsi qu'une page donnant les informations d'une formation.

Page d'accueil :



MediaTek86

Des formations sur des outils
numériques pour tous

Accueil Formations

Bienvenue sur le site de MediaTek86 consacré aux formations

Vous allez pouvoir vous former à différents outils numériques gratuitement et directement en ligne.

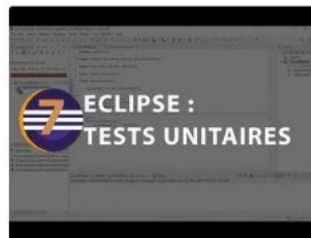
Dans la partie [Formations](#), vous trouverez la liste des formations proposées. Vous pouvez faire une recherche à partir d'un mot, trier les formations sur le titre ou la date de parution et, en cliquant sur la miniature, vous accéderez à la présentation plus détaillée de la formation ainsi que la vidéo correspondante.

Voici les deux dernières formations ajoutées au catalogue :



28/12/2020

Eclipse n°8 :
Déploiement



28/12/2020

Eclipse n°7 : Tests
unitaires

La page d'accueil présente le fonctionnement du site et les deux dernières formations ajoutées au catalogue. Elle contient également un menu pouvant aux 2 pages principales (sur l'accueil et sur la page listant les formations). En cliquant sur les miniatures des dernières formations, vous pouvez accéder aux détails d'une formation.



MediaTek86

Des formations sur des outils numériques pour tous

[Accueil](#) [Formations](#)

titre < >

< >

[filtrer](#)

Eclipse n°8 : Déploiement

28/12/2020



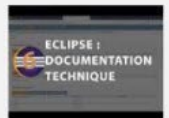
Eclipse n°7 : Tests unitaires

28/12/2020



Eclipse n°6 : Documentation technique

28/12/2020



La page des formations liste toutes les formations proposées en ligne, nous retrouvons encore le menu pour accéder aux pages principales.

Dans la partie centrale contient un tableau de trois colonnes :

- Première colonne → contient les titres des formations ;
- Deuxième colonne → contient les dates de parution ;
- Troisième colonne → contient les miniatures des vidéos.

En cliquant sur la miniature vous allez tomber sur les détails d'une formation.

Il est possible de trier les formations en fonction de l'ordre alphabétique, de les trier en fonction d'une zone de recherche avec saisie de texte et d'un tri en fonction de la date.



MediaTek86

Des formations sur des outils numériques pour tous

Accueil Formations



28/12/2020

Eclipse n°8 : Déploiement

description :

Exécution de l'application en dehors de l'IDE, en invite de commande.
Création d'un fichier jar pour le déploiement de l'application.
00:20 : exécuter l'application à partir d'un invite de commandes
04:41 : créer un fichier jar auto exécutable
06:42 : exécuter un fichier jar directement
07:09 : exécuter un fichier jar dans l'invite de commande pour avoir les retours console

Sur cette page de détails, vous allez retrouver les différentes informations d'une formation, elle est accessible en cliquant sur une miniature présente sur la page d'accueil et sur la page de formation.

Vous avez également la possibilité de regarder la vidéo sur cette page, ou d'y accéder sur

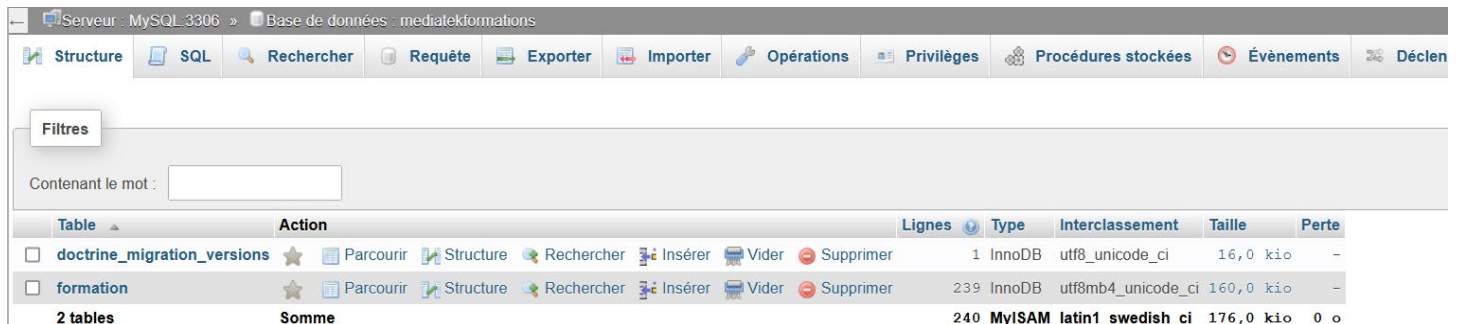
YouTube.

Base de données :

Cette base de données est au format MySQL, la structure de la table « formation » est la suivante :

```
CREATE TABLE IF NOT EXISTS `formation` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `published_at` datetime DEFAULT NULL COMMENT '(DC2Type:datetime_immutable)',  
  `title` varchar(91) COLLATE utf8mb4_unicode_ci DEFAULT NULL,  
  `description` longtext COLLATE utf8mb4_unicode_ci,  
  `miniature` varchar(46) COLLATE utf8mb4_unicode_ci DEFAULT NULL,  
  `picture` varchar(48) COLLATE utf8mb4_unicode_ci DEFAULT NULL,  
  `video_id` varchar(11) COLLATE utf8mb4_unicode_ci DEFAULT NULL,  
  PRIMARY KEY (`id`)  
)
```

Après avoir importé le script de la BDD (SGBD MySQL), nous obtenons la base de données suivante :



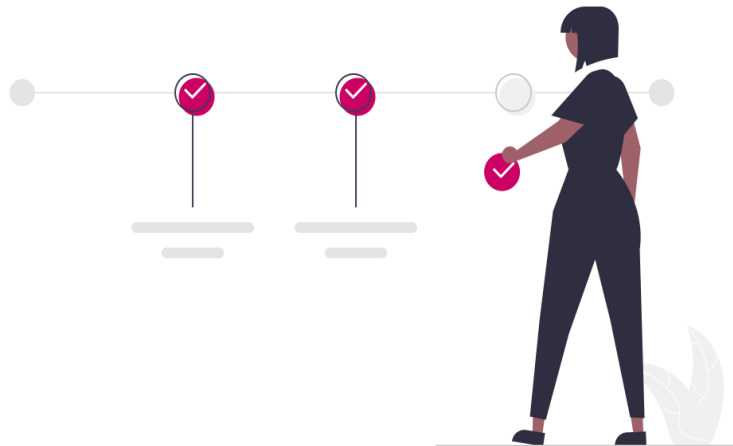
The screenshot shows the MySQL Workbench interface. The top toolbar includes buttons for Structure, SQL, Rechercher, Requête, Exporter, Importer, Opérations, Privileges, Procédures stockées, Événements, and Déclen. Below the toolbar is a 'Filtres' section with a search box. The main area displays a table list for the 'mediatekformations' database. The table list has columns: Table, Action, Lignes, Type, Interclassement, Taille, and Perte. Two tables are listed: 'doctrine_migration_versions' and 'formation'. The 'formation' table has 239 rows, is of type InnoDB, and has a size of 160,0 kio. The bottom summary row shows '2 tables' and a 'Somme' of 240 MyISAM latin1_swedish_ci 176,0 kio 0 o.

Table	Action	Lignes	Type	Interclassement	Taille	Perte
<input type="checkbox"/> doctrine_migration_versions	Parcourir Structure Rechercher Insérer Vider Supprimer	1	InnoDB	utf8_unicode_ci	16,0 kio	-
<input type="checkbox"/> formation	Parcourir Structure Rechercher Insérer Vider Supprimer	239	InnoDB	utf8mb4_unicode_ci	160,0 kio	-
2 tables	Somme	240	MyISAM	latin1_swedish_ci	176,0 kio	0 o

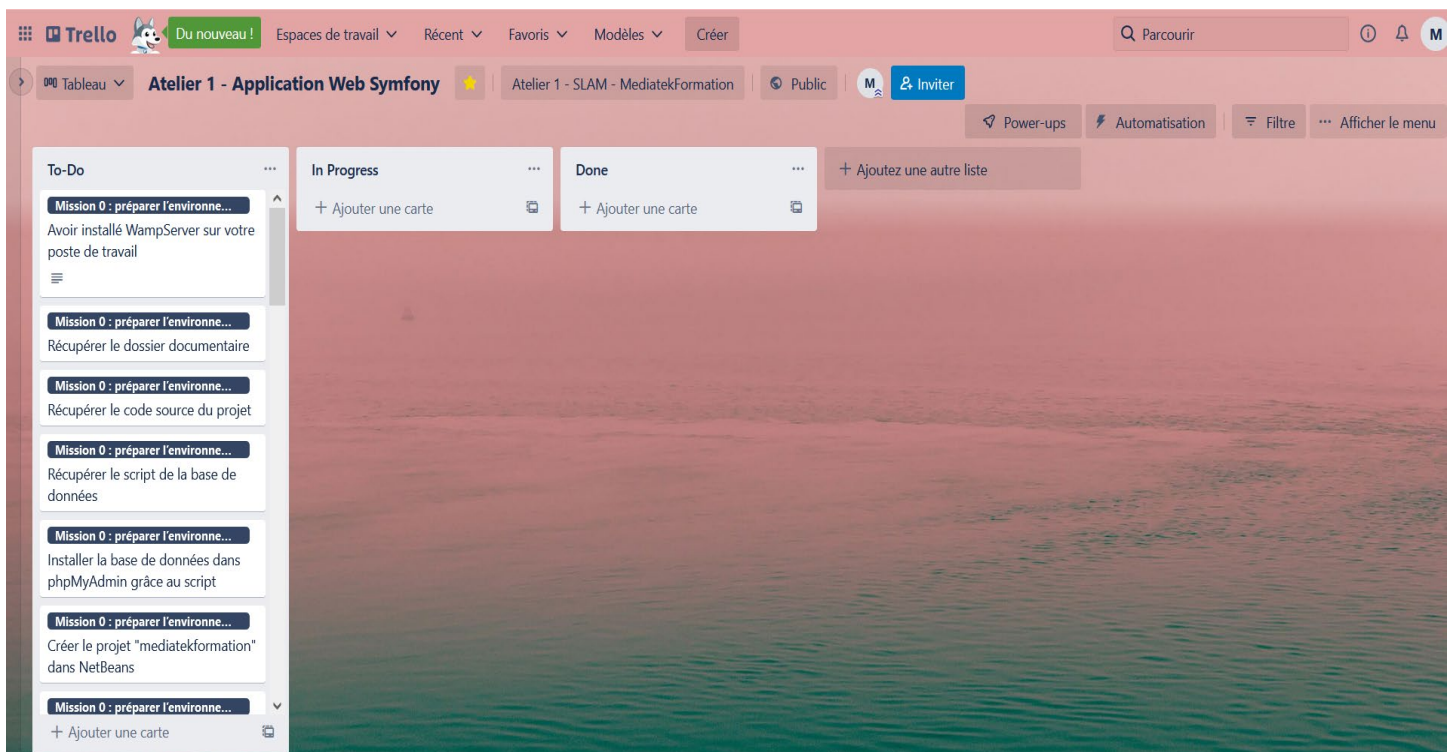
Pour continuer le projet, j'ai dû créer le projet « mediatekformation » à partir du code récupéré sur GitHub et j'ai testé l'application pour voir si tout fonctionnait bien. J'ai dû comparer les fonctionnalités avec celle décrite dans le dossier documentaire. Je n'ai eu aucun problème lors des tests. Pour finir cette première mission, j'ai créé le nouveau dépôt 'mediatekformation' sur mon compte GitHub, et j'ai lié ce dépôt à mon projet sous NetBeans.

Missions

Pour réaliser les demandes du client, je dois réaliser 5 grandes missions, qui contiennent toutes les directives à suivre afin de réaliser le projet. Dans ce compte rendu, vous allez pouvoir suivre les avancements du projet étape par étape.



Un suivi détaillé des missions du projet a été créé sur Trello : [LIEN TRELLO](#)



Mission 0 : préparer l'environnement de travail

La mission 0 consiste à la préparation de l'environnement de travail, celle-ci a pour objectif de :

- Récupérer l'application existante et créer un projet sous NetBeans à partir de cette application ;
- Récupérer le script de la base de données et la créer en local ;
- Créer un dépôt sur GitHub et relier le projet au dépôt.

Le dossier documentaire présente l'application actuelle et les demandes qui seront à traiter dans les missions suivantes. Le travail sera fait sur l'IDE NetBeans, un IDE professionnel et hébergé en local grâce à WampServer (plateforme de développement web de type WAMP, pour faire fonctionner localement des scripts PHP) pendant la partie de développement.

Pour commencer la réalisation du projet, un code source (récupérable sur [GitHub](#)) ainsi que le script de la base de données (à importer sur PhpMyAdmin) nous ont été transmis.

Le dossier documentaire et le script de la base de données. Ces documents sont récupérables sur mon portfolio.



Mission 1 : ajouter les niveaux

L'objectif de cette mission est d'ajouter un niveau pour chaque formation.

Avant de commencer, j'ai créé une branche « Mission 1 » afin de faire des commit de chaque création. Je ferai de même pour chaque mission.

La première étape de cette mission, était la création d'une table « Niveau » qui

```
c:\wamp64\www\mediatekformation>git checkout -b Mission1  
Switched to a new branch 'Mission1'
```

mémorisera les niveaux, pour faire cette étape il était demandé d'utiliser composer. Composer est un logiciel gestionnaire de dépendances libre écrit en PHP. Il permet à ses utilisateurs de déclarer et d'installer les bibliothèques dont le projet principal a besoin.

```
c:\wamp64\www\mediatekformation>php bin/console make:entity  
  
Class name of the entity to create or update (e.g. GentlePopsicle):  
> Niveau  
  
created: src/Entity/Niveau.php  
created: src/Repository/NiveauRepository.php  
  
Entity generated! Now let's add some fields!  
You can always add more fields later manually or by re-running this command.  
  
New property name (press <return> to stop adding fields):  
> level  
  
Field type (enter ? to see all types) [string]:  
> string  
  
Field length [255]:  
> 15  
  
Can this field be null in the database (nullable) (yes/no) [no]:  
>  
  
updated: src/Entity/Niveau.php  
  
Add another property? Enter the property name (or press <return> to stop adding fields):  
>  
  
Success!  
  
Next: When you're ready, create a migration with php bin/console make:migration  
  
c:\wamp64\www\mediatekformation>php bin/console make:migration  
  
Success!  
  
Next: Review the new migration "migrations/Version20220225121615.php"  
Then: Run the migration with php bin/console doctrine:migrations:migrate  
See https://symfony.com/doc/current/bundles/DoctrineMigrationsBundle/index.html  
  
c:\wamp64\www\mediatekformation>php bin/console doctrine:migrations:migrate  
  
WARNING! You are about to execute a migration in database "mediatekformations" that could result in schema changes and  
data loss. Are you sure you wish to continue? (yes/no) [yes]:  
>  
  
[notice] Migrating up to DoctrineMigrations\Version20220225121615  
[notice] finished in 1353.7ms, used 20M memory, 1 migrations executed, 2 sql queries
```

Création table « Niveau »

Chaque formation doit avoir un niveau différent (« Débutant », « Confirmé », « Expert »).

```
INSERT INTO `niveau` (`id`, `level`) VALUES (NULL,
'débutant'), (NULL, 'confirmé'), (NULL, 'expert')
```

Requête SQL pour remplir table « Niveau » sur PhpMyAdmin.

+ Options

					id	level
<input type="checkbox"/>		Éditer		Copier		Supprimer
1	Débutant					
<input type="checkbox"/>		Éditer		Copier		Supprimer
2	Confirmé					
<input type="checkbox"/>		Éditer		Copier		Supprimer
3	Expert					

Table « Niveau » après la requête

Pour qu'une formation possède un niveau (un paramètre qui sera obligatoire dans la prochaine mission), il faut d'abord créer l'entity « niveau_id_id » dans la table « Formation » avec le terminal.

```
c:\wamp64\www\mediatekformation>php bin/console make:entity formation

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

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

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

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

updated: src/Entity/Formation.php

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

Success!

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

c:\wamp64\www\mediatekformation>php bin/console make:migration

Success!

Next: Review the new migration "migrations/Version20220211143416.php"
Then: Run the migration with php bin/console doctrine:migrations:migrate
See https://symfony.com/doc/current/bundles/DoctrineMigrationsBundle/index.html

c:\wamp64\www\mediatekformation>php bin/console doctrine:migrations:migrate

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

[notice] Migrating up to DoctrineMigrations\Version20220211143416
[notice] finished in 3612.1ms, used 20M memory, 1 migrations executed, 1 sql queries

c:\wamp64\www\mediatekformation>
```

Création entity « niveau_id_id »

Afin de faire un lien entre les deux tables, j'ai créé une clé étrangère. Ensuite, j'ai dû générer des entiers entre 1 et 3 pour chaque formation pour qu'elles possèdent toutes un niveau.

Exécuter une ou des requêtes SQL sur la table « mediatekformations.formation »: ?

```
1 UPDATE formation SET niveau_id_id=FLOOR(1+rand()*3);
```

Requête pour générer entiers entre 1 et 3

Parcourir

Structure

SQL

Rechercher

Insérer

Exporter

Importer

Privileges

Operacions

Déclencheurs

Affichage des lignes 0 - 24 (total de 240, traitement en 0,0008 seconde(s).) [published_at: 2022-03-02 00:00:00... - 2020-04-04 00:00:00...]

SELECT * FROM `formation` ORDER BY `published_at` DESC

Profilage

[Éditer en ligne]

[Éditer]

[Expliquer SQL]

[Créer le code source PHP]

[Actualiser]

1

>

>>

Tout afficher

Nombre de lignes : 25

Filtrer les lignes: Chercher dans cette tat

Trier par clé : Aucun(e)

+ Options

←T→

id

published_at

title

description

miniature

picture

video_id

niveau_id_id

Éditer

Copier

Supprimer

1

2022-03-02 00:00:00

Eclipse n°8 : Déploiement

Exécution de l'application en dehors de l'IDE, en ...

https://i.ytimg.com/vi/Z4yTTXka958/default.jpg

https://i.ytimg.com/vi/Z4yTTXka958/sddefault.jpg

Z4yTTXka958

3

Éditer

Copier

Supprimer

2

2020-12-28 21:55:06

Eclipse n°7 : Tests unitaires

Intégration de JUnit dans l'application et créatio...

https://i.ytimg.com/vi/-nw42Xq6cYE/default.jpg

https://i.ytimg.com/vi/-nw42Xq6cYE/sddefault.jpg

-nw42Xq6cYE

1

Éditer

Copier

Supprimer

3

2020-12-28 21:49:27

Eclipse n°6 : Documentation technique

Intégration des commentaires normalisés et générat...

https://i.ytimg.com/vi/PrK_P3TKc00/default.jpg

https://i.ytimg.com/vi/PrK_P3TKc00/sddefault.jpg

PrK_P3TKc00

2

Éditer

Copier

Supprimer

4

2020-12-28 21:37:57

Eclipse n°5 : Refactoring

Utilisation des outils de refactoring et de généra...

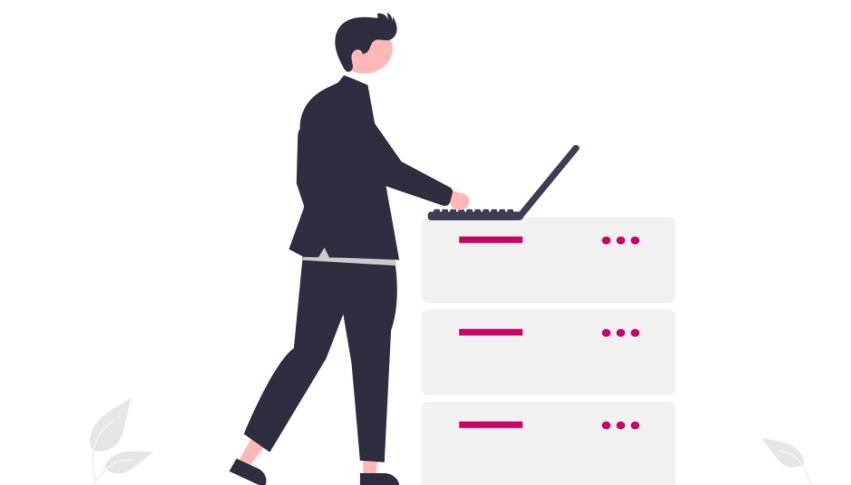
https://i.ytimg.com/vi/1p_mKDDSMnQ/default.jpg

https://i.ytimg.com/vi/1p_mKDDSMnQ/sddefault.jpg

1p_mKDDSMnQ

1

Table « Formation » avec l'entity « niveau_id_id » complété



Pour la suite de cette mission, il fallait modifier la page des formations pour afficher une colonne niveau, ainsi que pour filtrer les formations en fonction d'un niveau.



MediaTek86

Des formations sur des outils numériques pour tous

[Accueil](#)
[Formations](#)
[Connexion !](#)

Titre	Niveau	Date	
<input type="text"/> <input type="button" value="filtrer"/>	<input type="button" value="Débutant"/> <input type="button" value="Confirmé"/> <input type="button" value="Expert"/> <input type="button" value="filtrer"/>	<input type="button" value="Débutant"/> <input type="button" value="Confirmé"/> <input type="button" value="Expert"/> <input type="button" value="filtrer"/>	
Eclipse n°8 : Déploiement		02/03/2022	
Eclipse n°7 : Tests unitaires	Débutant	28/12/2020	
Eclipse n°6 : Documentation technique	Confirmé	28/12/2020	
Eclipse n°5 : Refactoring	Débutant	28/12/2020	

Page de formation avec les niveaux

Pour qu'on puisse afficher le niveau d'une formation, il a fallu créer une fonction pour récupérer le niveau dans « Formation.php ».

```

public function getLevel(): ?string
{
    return $this->getNiveauId()->getLevel();
}

```

Fonction pour récupérer le niveau

La classe « NiveauRepository » récupère toutes les informations présentes dans la table « Niveau ». Pour permettre l’affichage il faut donc créer une variable qui va stocker toutes ces valeurs.

```
/**
 * Retourne les formations
 * @Route("/formations", name="formations")
 * @return Response
 */
public function index(): Response{
    $formations = $this->repository->findAll();
    $niveauRepository = $this->getDoctrine()->getManager()->getRepository(Niveau::class);
    $niveaux = $niveauRepository->findAll();
    return $this->render(self::PAGEFORMATIONS, [
        'formations' => $formations,
        'niveaux' => $niveaux
    ]);
}
```

Fonction pour récupérer les formations

Le lien est donc créé entre les formations et les niveaux, il faut maintenant créer la fonctionnalité qui permet de trier en fonction des niveaux.

```

/**
 * Enregistrements dont un champ contient une valeur
 * ou tous les enregistrements si la valeur est vide
 * @param type $champ
 * @param type $valeur
 * @return Formation[]
 */
public function findByContainValue($champ, $valeur): array{
    if($valeur==""){
        return $this->createQueryBuilder('f')
            ->setParameter('valeur', $valeur)
            ->orderBy('f.'.$champ, 'ASC')
            ->getQuery()
            ->getResult();
    }else if($champ=="niveau"){
        return $this->createQueryBuilder('f')
            ->innerJoin('f.niveau_id', 'n')
            ->where('n.level LIKE :valeur')
            ->setParameter('valeur', $valeur)
            ->orderBy('f.publishedAt', 'DESC')
            ->setParameter('valeur', '%'.$valeur.'%')
            ->getQuery()
            ->getResult();
    }else{
        return $this->createQueryBuilder('f')
            ->where('f.'.$champ.' LIKE :valeur')
            ->setParameter('valeur', $valeur)
            ->orderBy('f.publishedAt', 'DESC')
            ->setParameter('valeur', '%'.$valeur.'%')
            ->getQuery()
            ->getResult();
    }
}

```

Fonction qui permet de trier les formations en fonction de leur niveau

Dans la vue « formation.html.twig » il a fallu ajouter l'appelle de la fonction de triage :

```
35 </div>
36 <input type="hidden" name="_token" value="{{ csrf_token('filtre_niveau') }}">
37 <button type="submit" class="btn btn-info mb-2 btn-sm">filtrer</button>
```

Appel de la fonction de trie dans la vue

Afin de permettre l’affichage des niveaux sur les pages, il suffit de modifier la vue :

- Page d’accueil :

```
33 | ..... <p><strong>Niveau : </strong>{{ formation.level }}</p>
```

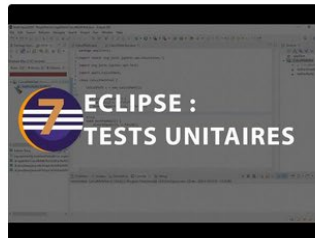
Voici les deux dernières formations ajoutées au catalogue :



02/03/2022

Eclipse n°8 :
Déploiement

Niveau : Expert



28/12/2020

Eclipse n°7 : Tests
unitaires

Niveau : Débutant

Modification + page d’accueil

- Page détail d’une formation :

```
19 | ..... <p><strong>Niveau : </strong></p>  
20 | ..... <p>{{ formation.level }}</p>
```



02/03/2022

Eclipse n°8 : Déploiement

Niveau :

Expert

Description :

description :

Exécution de l'application en dehors de l'IDE, en invite de commande.
Création d'un fichier jar pour le déploiement de l'application.
00:20 : exécuter l'application à partir d'un invite de commandes
04:41 : créer un fichier jar auto exécutable
06:42 : exécuter un fichier jar directement
07:09 : exécuter un fichier jar dans l'invite de commande pour avoir les retours console

Modification + page de détail d’une formation

Mission 2 : coder la partie back-office

Le back office doit permettre de gérer le contenu de la base de données, elle doit contenir le même visuel que le front office mais en ajoutant des fonctionnalités supplémentaires.

Les fonctionnalités à ajouter sont les suivantes :

- Créer, supprimer et modifier une formation ;
- Gestion des niveaux (ajout ou suppression) ;
- Créer la partie authentification.

Dans cette mission il faudra également contrôler la sécurité en vérifiant les requêtes paramétrées, Token et contrôles de saisie. Quand cette partie sera terminée, il faudra réaliser les tests unitaires demandés, générer la documentation technique et tester la compatibilité des navigateurs.

1. Créer, supprimer et modifier une formation

La page pour gérer les formations sera accessible après connexion sur un compte d'administrateur.

Vous êtes connecté en tant que admin ! [Se déconnecter..](#)



MediaTek86

Des formations sur des outils numériques pour tous

Gestions des formations Gestions des niveaux

Titre	Niveau	Date	Actions
<input type="text" value=""/> <input type="button" value="Filtrer"/>	<input type="button" value="Débutant"/> <input type="button" value="Filtrer"/>	<input type="button" value="<"/> <input type="button" value=">"/>	<input type="button" value="Ajouter une formation"/>
Eclipse n°8 : Déploiement	Expert	02/03/2022	 <input type="button" value="Editer"/> <input type="button" value="Supprimer"/>
Eclipse n°7 : Tests unitaires	Débutant	28/12/2020	 <input type="button" value="Editer"/> <input type="button" value="Supprimer"/>
Eclipse n°6 : Documentation technique	Confirmé	28/12/2020	 <input type="button" value="Editer"/> <input type="button" value="Supprimer"/>

Résultat final attendu

Symfony propose un outil pour générer un formulaire à partir d'une Entity, grâce à la commande suivante le formulaire vas être construit avec les propriétés de l'entity « Formation » et donc récupérer les éléments qui se trouve dans celle-ci.

```
c:\wamp64\www\mediatekformation>php bin/console make:form

The name of the form class (e.g. BravePuppyType):
> FormationType

The name of Entity or fully qualified model class name that the new form will be bound to (empty for none):
> formation

[ERROR] Entity "formation" doesn't exist; please enter an existing one or create a new one.

The name of Entity or fully qualified model class name that the new form will be bound to (empty for none):
> Formation

created: src/Form/FormationType.php

Success!
```

Création d'un formulaire

Dans NetBeans, un nouveau fichier a été créé : "FormationType.php", contenant la méthode suivante :

```
class FormationType extends AbstractType
{
    public function buildForm(FormBuilderInterface $builder, array $options): void
    {
        $builder
            ->add('publishedAt', DateType::class, [
                'label' => 'Date de publication : ',
                'data' => new DateTime(),
                'required' => true,
            ])
            ->add('title', TextType::class, [
                'label' => 'Titre : ',
                'attr' => ['maxlength' => 100],
                'required' => true,
            ])
            ->add('description', TextareaType::class, [
                'label' => 'Description : ',
                'attr' => ['rows' => 6],
                'required' => false,
            ])
            ->add('miniature', UrlType::class, [
                'label' => 'Miniature URL : ',
                'attr' => ['maxlength' => 46],
                'required' => false,
            ])
            ->add('picture', UrlType::class, [
                'label' => 'Image URL : ',
                'attr' => ['maxlength' => 48],
                'required' => false,
            ])
            ->add('videoId', TextType::class, [
                'label' => 'Video ID : ',
                'attr' => ['maxlength' => 11],
                'required' => false,
            ])
            ->add('niveau_id', EntityType::class, [
                'label' => 'Niveau : ',
                'class' => Niveau::class,
                'choice_label' => 'level',
                'required' => true,
            ])
            ->add('submit', SubmitType::class, [
                'label' => 'Enregistrer',
            ])
        ;
    }
}
```

Contenu de FormationType.php

La méthode « buildForm » va construire le formulaire, nous pouvons ajouter ou supprimer des champs en fonction de nos envies.

Il faut maintenant utiliser la classe « Submit » pour construire le formulaire dans le contrôleur et l'envoyer à la vue. Dans "AdminFormationsController", la méthode "edit" de cette façon ressemble à ça :

```
/**
 * Fonction qui permet d'éditer une formation
 * @Route("/admin/edit/{id}", name="admin.formation.edit")
 * @param Formation $formation
 * @param Request $request
 * @return Response
 */
public function edit(Formation $formation, Request $request): Response
{
    $formFormation = $this->createForm(FormationType::class, $formation);
    $formFormation->handleRequest($request);
    if($formFormation->isSubmitted() && $formFormation->isValid()){
        $this->om->flush();
        $this->addFlash('Bravo!', 'Vous avez édité la formation!');
        return $this->redirectToRoute('admin.formations');
    }
    return $this->render("admin/admin.formation.edit.html.twig", [
        'formation' => $formation,
        'formformation' => $formFormation->createView()
    ]);
}
```

Méthode pour modifier une formation

Il faut maintenant récupérer le formulaire dans la vue pour l'afficher, dans "admin.formation.edit.html.twig".

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

{% block body %}
    <h2>Modification de la formation </h2>
    <h3>{{ formation.title }}</h3>
    {{ include ('_admin.formations.form.html.twig') }}
{% endblock %}
```

Appel de la méthode edit dans "admin.formation.edit.html.twig"

Le fichier « _admin.formations.form.html.twig » ressemble à ça, on crée une page twig pour pouvoir réutiliser celle-ci pour la partie suivante (d'ajout une formation) :

```
{{ form_start(formformation) }}
<div class="row mt-3">
    <div class="col">
        <div class="row">
            <div class="col">
                {{ form_row(formformation.title) }}
                {{ form_row(formformation.publishedAt) }}
                {{ form_row(formformation.description) }}
                {{ form_row(formformation.submit) }}
            </div>
            <div class="col">
                <div class="row">
                    <div class="col-8">
                        {{ form_row(formformation.niveau_id) }}
                        {{ form_row(formformation.picture) }}
                        {{ form_row(formformation.miniature) }}
                        {{ form_row(formformation.videoId) }}
                    </div>
                </div>
            </div>
        </div>
    </div>
</div>
{{ form_end(formformation) }}
```



Des formations sur des outils numériques pour tous

Gestions des formations Gestions des niveaux

Modification de la formation

Eclipse n°8 : Déploiement

Titre :

Eclipse n°8 : Déploiement

Niveau :

Expert

Date de publication :

Mar

9

2022

Image URL :

<https://i.ytimg.com/vi/Z4yTTXka958/sddefault>

Description :

Exécution de l'application en dehors de l'IDE, en invite de commande.
Création d'un fichier jar pour le déploiement de l'application.
00:20 : exécuter l'application à partir d'un invite de commandes
04:41 : créer un fichier jar auto exécutable
06:42 : exécuter un fichier jar directement
07:09 : exécuter un fichier jar dans l'invite de commande pour avoir les retours console

Miniature URL :

<https://i.ytimg.com/vi/Z4yTTXka958/default.jp>

Video ID :

Z4yTTXka958

Enregistrer

La page de modification

Dans un second temps, j'ai dû mettre en place la page d'ajout, la logique vas être très proche de celle pour la modification. Dans un premier temps il faut créer un fichier twig « admin.formations.ajout.html » :

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

{% block body %}
    <h2>Ajout d'une formation</h2>
    {{ include ('_admin.formations.form.html.twig') }}
{% endblock %}
```

admin.formations.ajout.html.twig

Il faut ensuite appeler ce fichier dans « admin.formations.html » :

```
<th class="text-center align-top" scope="col">
    Actions
    <p>
        <a href={{ path('admin.formations.ajout') }} class="btn btn-info mb-2">Ajouter une formation</a>
    </p>
</th>
```

Appel de la méthode ajout dans la vue

Le code de la méthode « ajout » ressemble beaucoup à la méthode « edit » :

```
/**
 * Fonction qui permet l'ajout d'une formation grâce à la form FormationType
 * @Route("/admin/ajout", name="admin. formations. ajout")
 * @param Request $request
 * @return Response
 */
public function ajout(Request $request): Response{
    $formation = new Formation();
    $formFormation = $this->createForm(FormationType::class, $formation);

    $formFormation->handleRequest($request);
    if($formFormation->isSubmitted() && $formFormation->isValid()){
        $this->om->persist($formation);
        $this->om->flush();
        $this->addFlash('Bravo!', 'Vous avez ajouté la formation!');
        return $this->redirectToRoute('admin. formations');
    }

    return $this->render("admin/admin. formations. ajout. html. twig", [
        'formation' => $formation,
        'formformation' => $formFormation->createView()
    ]);
}
```

Méthode ajout



MediaTek86

Des formations sur des outils numériques pour tous

Gestions des formations Gestions des niveaux

Titre	Niveau	Date	Actions
<div>< ></div> <div><input type="text"/></div> <div>Filtrer</div>	<div>Débutant ▾</div> <div>Filtrer</div>	<div>< ></div>	<div>Ajouter une formation</div>

Page de formations



MediaTek86

Des formations sur des outils numériques pour tous

Gestions des formations Gestions des niveaux

Ajout d'une formation

Titre :

Niveau :

Débutant



Date de publication :

Mar



10



2022



Image URL :

Description :

Miniature URL :

Video ID :

Enregistrer

Page d'ajout de formation


```

/**
 * Fonction qui permet de supprimer une formation
 * @Route("/admin/suppr/{id}", name="admin.formation.suppr")
 * @param Formation $formation
 * @return Response
 */
public function suppr(Formation $formation): Response{
    $this->om->remove($formation);
    $this->om->flush();
    $this->addFlash('Bravo!', 'Vous avez supprimé la formation!');
    return $this->redirectToRoute('admin.formations');
}

```

Méthode pour supprimer une formation

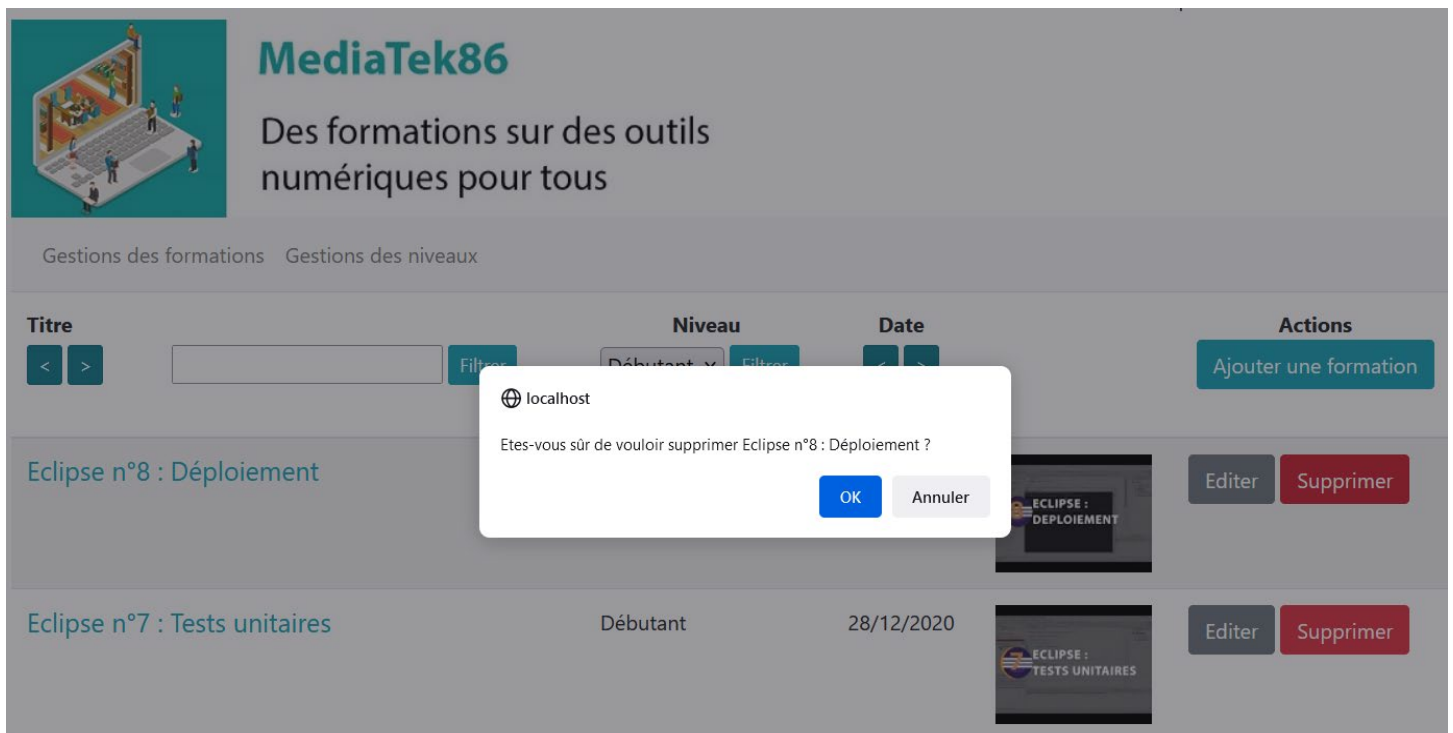
```

<a href="{{ path('admin.formation.suppr', {id:formation.id}) }}" class="btn btn-danger" onclick="return confirm('Etes-vous sûr de vouloir supprimer {{ formation.title }} ?')">Supprimer</a>

```

Appel de la méthode dans la vue

Lors du clic sur le bouton « Supprimer » une boîte de dialogue apparaît.



Message de suppression

2. Gestion des niveaux (ajout ou suppression)

Pour pouvoir gérer les niveaux, il faut créer un fichier twig qui contiendra le contenu de la page des niveaux.

```
admin.niveaux.html.twig
1 {% extends "basefrontadmin.html.twig" %}
2 {% block body %}
3     <table class="table table-striped">
4         <thead>
5             <tr>
6                 <th class="text-left align-top" style="width: 200px;" scope="col">
7                     Niveaux
8                 </th>
9                 <th class="text-center align-top" scope="col">
10                     Actions
11                 </th>
12             </tr>
13         </thead>
14         <tbody>
15             <form class="form-inline mt-2 mb-2" method="POST" action="{{ path('admin.niveau.ajout') }}">
16                 <tr>
17                     <td>
18                         <div class="form-group">
19                             <input type="text" name="Niveau" placeholder="Saisir le niveau à ajouter">
20                         </div>
21                     </td>
22                     <td class="text-center">
23                         <input type="hidden" name="csrf_token" value="{{ csrf_token('Ajout token') }}">
24                         <button type="submit" class="btn btn-primary" style="width: 100px;">Ajouter</button>
25                     </td>
26                 </tr>
27             </form>
28             {% for niveau in niveaux %}
29                 <tr>
30                     <td>
31                         <div class="text-info">
32                             {{ niveau.level|capitalize }}
33                         </div>
34                         <div class="text-center">
35                             <a href="{{ path('admin.niveaux.suppr', (id:niveau.id)) }}" class="btn btn-danger" style="width: 100px;" onclick="return confirm('Etes-vous sûr de vouloir supprimer le niveau : {{ niveau.level }} ?')">Supprimer</a>
36                         </div>
37                     </td>
38                 </tr>
39             {% endfor %}
40         </tbody>
41     </table>
42 {% endblock %}
```

Contenue du fichier twig de la page de gestion de niveaux

Sur la vue, il faut envoyer les fonctions d'ajout et de suppression (avec l'id du niveau pour pouvoir le supprimer).



MediaTek86

Des formations sur des outils numériques pour tous

Gestions des formations Gestions des niveaux

Niveaux	Actions
<input type="text" value="Saisir le niveau à ajouter"/>	<button>Ajouter</button>
Débutant	<button>Supprimer</button>
Confirmé	<button>Supprimer</button>
Expert	<button>Supprimer</button>

Page gestion de niveau

Il faut créer un nouveau contrôleur pour créer les différentes fonctions (« AdminNiveauxController.php »), et ensuite la création des fonctions.

```
/**
 * Fonction qui permet de supprimer un niveau
 * @Route("/admin/niveaux/suppr/{id}", name="admin.niveaux.suppr")
 * @param Niveau $niveau
 * @return Response
 */
public function suppr(Niveau $niveau): Response{
    try{
        $this->om->remove($niveau);
        $this->om->flush();
        $this->addFlash('Bravo!', 'Vous avez supprimé le niveau!');
    }
    catch (ForeignKeyConstraintViolationException $e){
        $this->addFlash('impossible', 'Vous ne pouvez pas supprimé ce niveau.. Des formations utilise celui-ci');
    }
    return $this->redirectToRoute('admin.niveaux');
}

/**
 * Fonction qui permet d'ajouter un niveau
 * @Route("/admin/niveaux/ajout", name="admin.niveau.ajout")
 * @param Request $request
 * @return Response
 */
public function ajout(Request $request): Response {
    if($this->isCsrfTokenValid('Ajout_token', $request->get('_token'))){
        $addNiveau = $request->get("Niveau");
        $niveau = new Niveau();
        $niveau->setLevel($addNiveau);
        $this->om->persist($niveau);
        $this->om->flush();
        $this->addFlash('Bravo!', 'Vous avez ajouté le niveau!');
    }
    return $this->redirectToRoute('admin.niveaux');
}

/**
 * Retourne les niveaux
 * @Route("/admin/niveaux", name="admin.niveaux")
 * @return Response
 */
public function index(): Response{
    $niveaux = $this->repository->findAll();
    return $this->render("admin/admin.niveaux.html.twig", [
        'niveaux' => $niveaux,
    ]);
}
```

Contenu de « AdminNiveauxController.php »

Pour la fonction qui permet de supprimer un niveau, il faut tester si le niveau est utilisé par une formation, si c'est le cas un message d'erreur apparait, j'ai utilisé un try/catch pour cette fonction.

Vous ne pouvez pas supprimé ce niveau.. Des formations utilise celui-ci

Message d'erreur pour suppression d'un niveau utilisé

3. Sécurisation

Il y a différente sécurisation à mettre en place des requêtes paramétrées, les Token et les contrôles de saisie.

a) Les requêtes paramétrées

Pour apporter une bonne protection, nous devons utiliser Doctrine, mais il faut bien l'utiliser. Nous devons absolument utiliser « setParameter » lors de la construction d'une requête. Cette requête permet de récupérer pour lesquelles le \$champ est égal à la valeur contenue dans \$valeur.

```
/**
 * Enregistrements dont un champ contient une valeur
 * ou tous les enregistrements si la valeur est vide
 * @param type $champ
 * @param type $valeur
 * @return Formation[]
 */
public function findByContainValue($champ, $valeur): array{
    if($valeur==""){
        return $this->createQueryBuilder('f')
            ->setParameter('valeur', $valeur)
            ->orderBy('f.'.$champ, 'ASC')
            ->getQuery()
            ->getResult();
    }else if($champ=="niveau"){
        return $this->createQueryBuilder('f')
            ->innerJoin('f.niveau_id', 'n')
            ->where('n.level LIKE :valeur')
            ->setParameter('valeur', $valeur)
            ->orderBy('f.publishedAt', 'DESC')
            ->setParameter('valeur', '%'.$valeur.'%')
            ->getQuery()
            ->getResult();
    }else{
        return $this->createQueryBuilder('f')
            ->where('f.'.$champ.' LIKE :valeur')
            ->setParameter('valeur', $valeur)
            ->orderBy('f.publishedAt', 'DESC')
            ->setParameter('valeur', '%'.$valeur.'%')
            ->getQuery()
            ->getResult();
    }
}
```

Requête paramétrées

b) Les Token

Pour se protéger des failles CSRF (Cross-site Request Forgery), il faut utiliser des Token (ou jetons unique). Les formulaires générés par Symfony intègrent automatiquement un token pour le contrôle du formulaire. Le token est un jeton qui contient un code spécifique est qui est vérifié lorsque le formulaire est soumis (clic sur submit). Pour les formulaires classiques, il faut ajouter ce token.

```
<input type="hidden" name="_token" value="{{ csrf_token('Ajout_token') }}">
```

```
/**
 * Fonction qui permet d'ajouter un niveau
 * @Route("/admin/niveaux/ajout", name="admin.niveau.ajout")
 * @param Request $request
 * @return Response
 */
public function ajout(Request $request): Response {
    if($this->isCsrfTokenValid('Ajout_token', $request->get('_token'))){
        $addNiveau = $request->get("Niveau");
        $niveau = new Niveau();
        $niveau->setLevel($addNiveau);
        $this->om->persist($niveau);
        $this->om->flush();
        $this->addFlash('Bravo!', 'Vous avez ajouté le niveau!');
    }
    return $this->redirectToRoute('admin.niveaux');
}
```

Utilisation de Token

c) Les contrôles de saisie

Le contrôle de saisie permet de savoir si nous pouvons insérer des valeurs incorrectes dans une zone de saisie. Pour sécuriser les saisies, il faut trouver les différentes possibilités de contraintes dans la documentation de Symfony : "documentation > Guides > Validation > Constraints". J'ai adapté mes champs en fonction des contraintes de saisie.

```
/**
 * @Assert\NotBlank
 * @Assert\Length(max = 90)
 * @ORM\Column(type="string", length=90, nullable=true)
 */
private $title;
```

Contrôle de saisie

4. Test unitaire

Le test permet de contrôler le fonctionnement de la méthode qui retourne la date de parution au format string.

Lors de la première utilisation de cette commande, phpunit installe les composants nécessaires pour réaliser les tests. Lors des prochaines utilisations de cette commande, les tests seront lancés.

```
c:\wamp64\www\mediatekformation>php bin/phpunit
```

Dans NetBeans, nous devons créer une nouvelle classe PHP, qui contiendra les tests. Cette classe contiendra la méthode qui va tester :

```
<?php

namespace App\Tests;

/**
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

use App\Entity\Formation;
use PHPUnit\Framework\TestCase;
use Symfony\Component\Validator\Constraints\DateTime;
use DateTimeInterface;

/**
 * Description of FormationTest
 *
 * @author petit
 */
class FormationTest extends TestCase{

    public function testGetDateparutionString(){
        $formation = new Formation();
        $formation->setPublishedAt(new \DateTime("2021-06-26"));
        $this->assertEquals("26/06/2021", $formation->getPublishedAtString());
    }
}
```

Code du test unitaire

Pour lancer le test, allez dans la fenêtre de commandes et tapez à nouveau la commande qui exécute phpunit :

```
c:\wamp64\www\mediatekformation>php bin/phpunit
PHPUnit 9.5.10 by Sebastian Bergmann and contributors.

Testing
.
1 / 1 (100%)

Time: 00:10.872, Memory: 8.00 MB

OK (1 test, 1 assertion)
```

Exécution du test

Le test a réussi.

5. Accès avec authentification

Pour commencer la création de l'authentification, il faut créer l'entity User :

```
c:\wamp64\www\mediatekformation>php bin/console make:user
```

```
The name of the security user class (e.g. User) [User]:
>

Do you want to store user data in the database (via Doctrine)? (yes/no) [yes]:
>

Enter a property name that will be the unique "display" name for the user (e.g.
email, username, uuid) [email]:
> username

Will this app need to hash/check user passwords? Choose No if passwords are not
needed or will be checked/hashed by some other system (e.g. a single sign-on se
rver).

Does this app need to hash/check user passwords? (yes/no) [yes]:
>

created: src/Entity/User.php
created: src/Repository/UserRepository.php
updated: src/Entity/User.php
updated: config/packages/security.yaml

Success!
```

Création de l'entity User

La création de l'entity User a réussi et maintenant il faut remplir la base de donnée, Symfony nous offre la possibilité de le faire grâce aux « fixtures ». La commande à utiliser est :

```
c:\wamp64\www\mediatekformation>php bin/console make:fixture

The class name of the fixtures to create (e.g. AppFixtures):
> UserFixture

created: src/DataFixtures/UserFixture.php

Success!

Next: Open your new fixtures class and start customizing it.
Load your fixtures by running: php bin/console doctrine:fixtures:load
Docs: https://symfony.com/doc/current/bundles/DoctrineFixturesBundle/index.html
```

Création de UserFixture

J'ai donné le nom "UserFixture" à la classe qui va être créée et qui va contenir le code pour générer les enregistrements. Dans la méthode "load", il faut écrire le code qui permet de générer l'utilisateur que l'on veut créer. Pour cela, on a besoin d'un objet qui va permettre de hacher le mot de passe. Il suffira ensuite d'utiliser une fonction d'encodage sur cet objet.

```
<?php

namespace App\DataFixtures;

use App\Entity\User;
use Doctrine\Bundle\FixturesBundle\Fixture;
use Doctrine\Persistence\ObjectManager;
use Symfony\Component\Security\Core\Encoder\UserPasswordEncoderInterface;
/**
 * Utilisation de UserFixture pour la création d'un User
 */
class UserFixture extends Fixture
{
    private $passwordEncoder;
    public function __construct(UserPasswordEncoderInterface $passwordEncoder)
    {
        $this->passwordEncoder = $passwordEncoder;
    }

    public function load(ObjectManager $manager): void
    {
        $user = new User();
        $user->setUsername(" ");
        $user->setPassword($this->passwordEncoder->encodePassword($user, ' '));
        $user->setRoles(['ROLE_ADMIN']);
        $manager->persist($user);
        $manager->flush();
    }
}
```

Code de UserFixture

Les cadres rouges, sont les endroits où il doit y avoir les identifiants de l'admin.

La propriété « \$passwordEncoder » permet de récupérer, dans le constructeur, l'encodeur passé en paramètre. Cette propriété peut alors être utilisée dans la méthode "load" pour encoder le mot de passe. Pour exécuter « load » il faut utiliser la commande suivante dans le terminal :

```
c:\wamp64\www\mediatekformation>php bin/console doctrine:fixtures:load --append --group=UserFixture
> loading App\DataFixtures\UserFixture
```

Exécution de la fonction load

Dans la BDD, le compte a bien été créé et encodé :

	id	username	roles	password
<input type="checkbox"/> Éditer <input type="checkbox"/> Copier <input type="checkbox"/> Supprimer	1	admin	["ROLE_ADMIN"]	\$argon2id\$v=19\$m=65536,t=4,p=1\$U2xjYzdKL2Y0dkl1bVh...

BDD User

Il faut préciser que, pour accéder au chemin "/admin", il faut avoir le rôle "ROLE_ADMIN".

```
# Easy way to control access for large sections of your site
# Note: Only the *first* access control that matches will be used
access_control:
    - { path: ^/admin, roles: ROLE_ADMIN }
    # - { path: ^/profile, roles: ROLE_USER }
```

Pour permettre à la personne de s'identifier, j'ai utilisé un formulaire :

```
c:\wamp64\www\mediatekformation>php bin/console make:auth

What style of authentication do you want? [Empty authenticator]:
[0] Empty authenticator
[1] Login form authenticator
> 1

The class name of the authenticator to create (e.g. AppCustomAuthenticator):
> LoginFormAuthenticator

Choose a name for the controller class (e.g. SecurityController) [SecurityController]:
>

Do you want to generate a '/logout' URL? (yes/no) [yes]:
> ^C

c:\wamp64\www\mediatekformation>php bin/console make:auth

What style of authentication do you want? [Empty authenticator]:
[0] Empty authenticator
[1] Login form authenticator
> 1

The class name of the authenticator to create (e.g. AppCustomAuthenticator):
> LoginFormAuthenticator

Choose a name for the controller class (e.g. SecurityController) [SecurityController]:
>

Do you want to generate a '/logout' URL? (yes/no) [yes]:
>

created: src/Security/LoginFormAuthenticator.php
updated: config/packages/security.yaml
created: src/Controller/SecurityController.php
created: templates/security/login.html.twig

Success!

Next:
- Customize your new authenticator.
- Finish the redirect "TODO" in the App\Security\LoginFormAuthenticator::onAuthenticationSuccess() method.
- Review & adapt the login template: templates/security/login.html.twig.
```

Utilisation d'un formulaire pour gérer la connexion

Les fichiers suivant on était créés et contrôle les authentifications :

```
<?php

namespace App\Controller;

use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Annotation\Route;
use Symfony\Component\Security\Http\Authentication\AuthenticationUtils;

class SecurityController extends AbstractController
{
    /**
     * Page de connexion
     * @Route("/login", name="app_login")
     */
    public function login(AuthenticationUtils $authenticationUtils): Response
    {
        // if ($this->getUser()) {
        //     return $this->redirectToRoute('target_path');
        // }

        // get the login error if there is one
        $error = $authenticationUtils->getLastAuthenticationError();
        // last username entered by the user
        $lastUsername = $authenticationUtils->getLastUsername();

        return $this->render('security/login.html.twig', ['last_username' => $lastUsername, 'error' => $error]);
    }

    /**
     * Fonction pour permettre la déconnexion
     * @Route("/logout", name="app_logout")
     */
    public function logout(): void
    {
        throw new \LogicException('This method can be blank - it will be intercepted by the logout key on your firewall');
    }
}
```

"SecurityController.php"

Ce fichier comporte les deux méthodes :

- "login" pour accéder au formulaire d'authentification,
- "logout" pour la déconnexion.

```

{% extends 'base.html.twig' %}

{% block title %}Connexion!{% endblock %}

{% block body %}
<div class="Container">
  <div class="row">
    <div class="col-md-10 ml-md-auto">
      <div class="">
        <div class="card bg-light mb-3 mt-5" style="width: 800px;">
          <div class="card-body">
            <form class="form-horizontal" method="post">
              {% if error %}
                <div class="alert alert-danger">{{ error.messageKey|trans(error.messageData, 'security') }}</div>
              {% endif %}
              <div>
                <div class="card-header mb-3">Page de connexion :</div>
                <div class="form-group">
                  <label for="inputEmail" class="col-md-4 control-label">Utilisateur</label>
                  <div class="col-md-12">
                    <input type="text" value="{{ last_username }}" name="username" id="inputEmail" class="form-control" autocomplete="username" required autofocus>
                  </div>
                </div>
                <div class="form-group">
                  <label for="inputPassword">Mot de passe</label>
                  <div class="col-md-12">
                    <input type="password" name="password" id="inputPassword" class="form-control" autocomplete="current-password" required>
                  </div>
                </div>
                <input type="hidden" name="_csrf_token"
                  value="{{ csrf_token('authenticate') }}"
                >

                {#
                Uncomment this section and add a remember_me option below your firewall to activate remember me functionality.
                See https://symfony.com/doc/current/security/remember_me.html

                <div class="checkbox mb-3">
                  <label>
                    <input type="checkbox" name="_remember_me"> Remember me
                  </label>
                </div>
                #}
                <div class="form-group">
                  <div class="col-md-12">
                    <button class="btn btn-primary" type="submit">
                      Se connecter
                    </button>
                  </div>
                </div>
              </form>
            </div>
          </div>
        </div>
      </div>
    </div>
  </div>
</div>
{% endblock %}

```

Le code du formulaire d'authentification

```

class LoginFormAuthenticator extends AbstractFormLoginAuthenticator implements PasswordAuthenticatedInterface
{
    use TargetPathTrait;

    public const LOGIN_ROUTE = 'app_login';

    private EntityManager;
    private UrlGenerator;
    private CsrfTokenManager;
    private PasswordEncoder;

    public function __construct(EntityManagerInterface $entityManager, UrlGeneratorInterface $urlGenerator, CsrfTokenManagerInterface $csrfTokenManager, UserPasswordEncoderInterface $passwordEncoder)
    {
        $this->entityManager = $entityManager;
        $this->urlGenerator = $urlGenerator;
        $this->csrfTokenManager = $csrfTokenManager;
        $this->passwordEncoder = $passwordEncoder;
    }

    public function supports(Request $request)
    {
        return self::LOGIN_ROUTE === $request->attributes->get('_route')
            && $request->isMethod('POST');
    }

    public function getCredentials(Request $request)
    {
        $credentials = [
            'username' => $request->request->get('username'),
            'password' => $request->request->get('password'),
            'csrf_token' => $request->request->get('_csrf_token'),
        ];
        $request->getSession()->set(
            Security::LAST_USERNAME,
            $credentials['username']
        );

        return $credentials;
    }

    public function getUser($credentials, UserProviderInterface $userProvider)
    {
        $token = new CsrfToken('authenticate', $credentials['csrf_token']);
        if (!$this->csrfTokenManager->isTokenValid($token)) {
            throw new InvalidCsrfTokenException();
        }

        $user = $this->entityManager->getRepository(User::class)->findOneBy(['username' => $credentials['username']]);

        if (!$user) {
            throw new UsernameNotFoundException('Utilisateur introuvable.');
        }

        return $user;
    }

    public function checkCredentials($credentials, UserInterface $user)
    {
        return $this->passwordEncoder->isPasswordValid($user, $credentials['password']);
    }

    /**
     * Used to upgrade (rehash) the user's password automatically over time.
     */
}

```

La classe LoginFormAuthenticator est sollicitée dès que le formulaire d'authentification est soumis (clic sur submit).

6. Documentation technique

Pour générer la documentation technique, j'ai utilisé les commandes présentées sur le site de phpdocumentor.

Vous pouvez récupérer la documentation sur le portfolio.

7. Scénario et test de compatibilité

Selenium est un framework de tests qui permet d'enregistrer un scénario (un ensemble de manipulations réalisées sur un site) et de le rejouer.

The screenshot shows the Selenium IDE interface within a Mozilla Firefox browser window. The title bar indicates 'Extension : (Selenium IDE) - Selenium IDE - TestMediatekFormation - Mozilla Firefox'. The main window displays a test scenario for 'TestMediatekFormation' at the URL 'http://192.168.1.94/mediatekformation/public/index.php'.

Test 1	Command	Target	Value	
Untitled	1	open	http://192.168.1.94/mediatekformation/public/index.php/	
	2	set window size	1406x979	
	3	click	css=.mt-3 > a	
	4	click	linkText=Accueil	
	5	click	css=td:nth-child(1) .card-img-top	
	6	click	linkText=Formations	
	7	click	linkText=>	
	8	click	linkText=<	
	9	click	name=recherche	
	10	type	name=recherche	blabla
	11	click	css=.col .btn	
	12	click	name=recherche	
	13	type	name=recherche	UML
	14	send keys	name=recherche	{KEY_ENTER}

Below the table, there are input fields for 'Command' (set to 'open'), 'Target' (set to 'http://192.168.1.94/mediatekformation/public/index.p'), 'Value' (empty), and 'Description' (empty). The bottom of the interface shows tabs for 'Log' and 'Reference'.

Scénario créer pour MediatekFormation

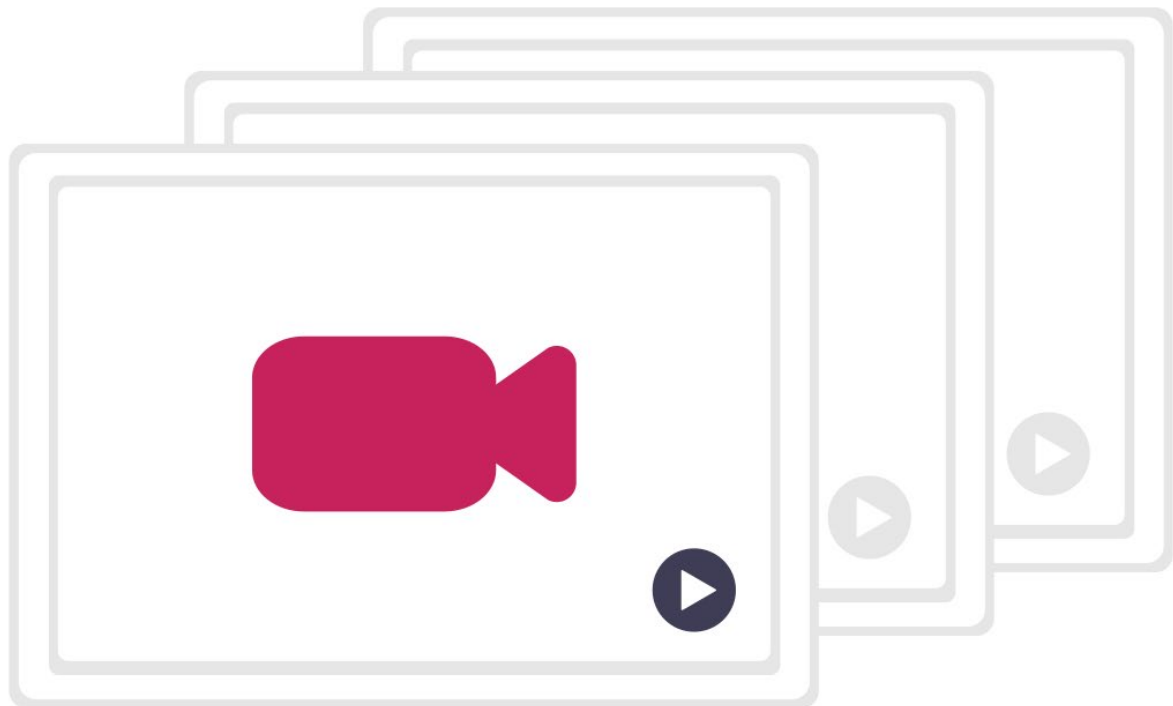
Pour la suite j'ai dû mettre en place un serveur Selenium (Grid) pour gérer l'exécution automatique de tests, en envoyant plusieurs fois le scénario à partir de clients différents. J'ai donc installé différents outils pour mener tester en locale et sur une machine virtuelle (Selenium Grid, les navigateurs, Selenium Side Runner).

Mission 3

L'objectif de la mission était de créer une documentation utilisateur (démonstration des fonctionnalités des parties front et back-office) sous forme de vidéo. Pour réaliser cette mission, j'ai enregistré mon écran avec l'enregistreur « Gamer » (offert par Windows) et ajouté le son avec BeeCut.

Vous pouvez retrouver la vidéo sur le lien suivant :

<https://youtu.be/4as8UMvbAj0>



Mission bilan :

Gérer le déploiement, rédiger le compte rendu, créer la page du portfolio dédiée à cet atelier

Les objectifs de cette mission :

- Gérer le déploiement de l'application.
- Rédiger un compte rendu d'activité.
- Créer une page dans le portfolio pour présenter l'activité et donner tous les liens nécessaires.

1. Déployer l'application en ligne

Pour déployer mon site en ligne, j'ai choisi de prendre un VPS. Les VPS répondaient le plus à mes demandes, car je peux installer les logiciels et services que je souhaite (LAMP). J'ai pris un VPS sur le site [DigitalOcean](#) à 5€ et également un nom de domaine sur [NameCheap](#).

Pour installer mon application en ligne, j'ai suivi les différents tutoriels proposés par DigitalOcean :

[ICI](#)

Les principales étapes de déploiement sont :

- Configuration initiale de serveur avec Ubuntu
- Comment installer LAMP ?
- Comment installer et utiliser Composer ?
- Comment installer Git ?
- Création de la base de données
- Cloner le projet (provenant de GitHub)
- Configurer l'application et la base de données
- Configurer le serveur web



MediaTek86

Des formations sur des outils numériques pour tous

[Accueil](#) [Formations](#)

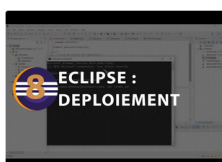
[Connexion !](#)

Bienvenue sur le site de MediaTek86 consacré aux formations

Vous allez pouvoir vous former à différents outils numériques gratuitement et directement en ligne.

Dans la partie [Formations](#), vous trouverez la liste des formations proposées. Vous pouvez faire une recherche à partir d'un mot, trier les formations sur le titre ou la date de parution et, en cliquant sur la miniature, vous accéderez à la présentation plus détaillée de la formation ainsi que la vidéo correspondante.

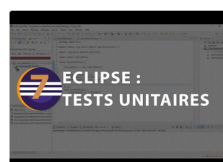
Voici les deux dernières formations ajoutées au catalogue :



02/03/2022

**Eclipse n°8 :
Déploiement**

Niveau : Expert



28/12/2020

**Eclipse n°7 : Tests
unitaires**

Niveau : Débutant

2. Faire le compte rendu d'activité

Le compte rendu d'activité contient :

- Un sommaire automatisé avec les liens vers les différentes parties ;
- Le rappel en résumé du contexte ;
- Le rappel des missions à effectuer ;
- La présentation de chaque mission avec l'explication du travail réalisé et un bilan sur les objectifs atteints (insérer des captures d'écrans, extraits de code et capture de l'évolution du suivi du projet) ;
- Un bilan final ;
- La liste des compétences officielles couvertes (B1, B2, B3).

Vous pouvez retrouver mon compte rendu sur le portfolio, ainsi que tous les autres documents nécessaires à la compréhension du projet.


3. Créer une page dans le portfolio

La page dans votre portfolio doit contenir :




- Une présentation rapide (quelques lignes) du contexte et de l'activité ;
- Le lien vers le PDF qui présente le contexte ;
- Le lien vers le PDF qui présente l'existant et l'expression des besoins ;
- Le lien vers le dépôt distant qui doit contenir les différentes étapes de sauvegarde et le script SQL complet de la BDD ;
- Le lien vers la documentation technique mise en ligne ou en téléchargement ;
- Le lien vers le PDF du compte rendu d'activité ;
- L'adresse du site en ligne pour pouvoir le tester ;
- L'intégration de la vidéo de démonstration de l'utilisation du site.

MediatekFormation

Vous êtes connecté en tant que admin ! [Se déconnecter...](#)

**MediaTek86**
Des formations sur des outils numériques pour tous

Gestions des formations Gestions des niveaux

Titre	Niveau	Date	Actions	
< >	Filtrer	Débutant Filtrer	< >	Ajouter une formation
Eclipse n°8 : Déploiement	Expert	02/03/2022		Editer Supprimer
Eclipse n°7 : Tests unitaires	Débutant	28/12/2020		Editer Supprimer
Eclipse n°6 : Documentation technique	Confirmé	28/12/2020		Editer Supprimer

Information

Langage: PHP - Symfony

Année: 2ème année BTS SIO (CNED)

[Site](#)

[GitHub du projet](#)

[Contexte](#)

[PDF qui présente l'existant et l'expression des besoins](#)

[PDF compte rendu](#)

[Documentation Technique + BDD](#)

[Vidéo](#)

MediatekFormation met à disposition une médiathèque de vidéo d'auto-formations, ces vidéos sont aussi disponible sur Youtube.

Designed by [BootstrapMade](#)

4. Bilan Final

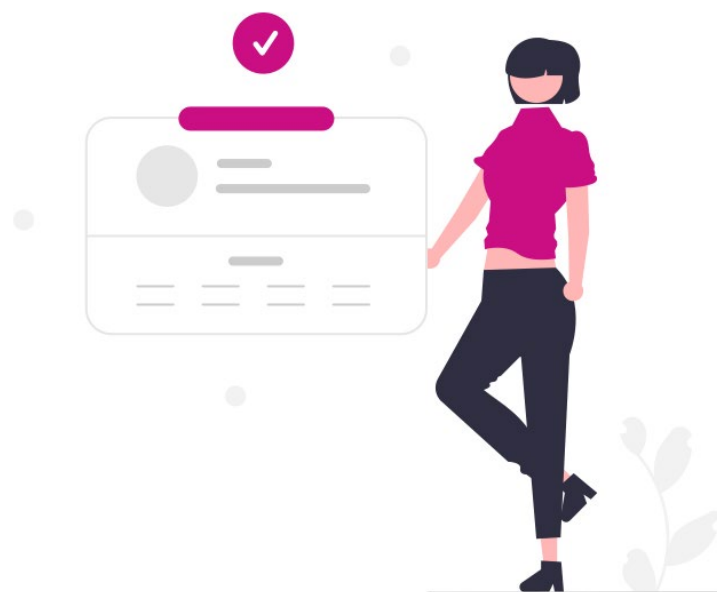
Vous pouvez accéder à l'application avec les liens suivant :

- mediatekformations.xyz
- 46.101.53.66

Vous retrouverez les documents essentiels à la compréhension du projet sur mon [portofio](#).

Pour tester la partie admin :

Utilisateur		Mot de passe	
admin		adminmediatek9910	
Les compétences officielles couvertes			
B1.4	Travailler en mode projet		
B1.5	Mettre à disposition des utilisateurs un service informatique		
B2.1	Conception et développer une solution applicative		
B2.2	Assurer la maintenance corrective ou évolutive d’une solution applicative		
B2.3	Gérer les données		
B3.5	Assurer la cyber sécurité d’une solution applicative et de son développement		



Bilan

Pour conclure, c'était un projet très intéressant à faire. La mise en place de cette application web, m'a permis de découvrir un peu plus de choses dans le domaine du développement web ainsi que les différentes possibilités que nous offre PHP et Symfony.

Ce projet m'a conforté dans l'idée de ma future orientation professionnelle en tant que développeuse web et il m'a donné envie d'en découvrir encore plus.

