Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE

MINISTÈRE CHARGÉ
DE L'EMPLOI

DOSSIER PROFESSIONNEL (DP)

Nom de naissance ▸ RENAU
Nom d'usage ▸ RENAU
Prénom ▸ Laury
Adresse ▸ 17 rue de la mésange – 67700 SAVERNE

Titre professionnel visé

Développeur web et web mobile niveau 5

MODALITE D'ACCES :

☒ Parcours de formation
☐ Validation des Acquis de l'Expérience (VAE)

Présentation du dossier

Le dossier professionnel (DP) constitue un élément du système de validation du titre professionnel. **Ce titre est délivré par le Ministère chargé de l'emploi.**

Le DP appartient au candidat. Il le conserve, l'actualise durant son parcours et le présente **obligatoirement à chaque session d'examen**.

Pour rédiger le DP, le candidat peut être aidé par un formateur ou par un accompagnateur VAE.

Il est consulté par le jury au moment de la session d'examen.

Pour prendre sa décision, le jury dispose :

1. des résultats de la mise en situation professionnelle complétés, éventuellement, du questionnaire professionnel ou de l'entretien professionnel ou de l'entretien technique ou du questionnement à partir de productions.
2. du **Dossier Professionnel** (DP) dans lequel le candidat a consigné les preuves de sa pratique professionnelle
3. des résultats des évaluations passées en cours de formation lorsque le candidat évalué est issu d'un parcours de formation
4. de l'entretien final (dans le cadre de la session titre).

[Arrêté du 22 décembre 2015, relatif aux conditions de délivrance des titres professionnels du ministère chargé de l'Emploi]

Ce dossier comporte :

- pour chaque activité-type du titre visé, un à trois exemples de pratique professionnelle ;
- un tableau à renseigner si le candidat souhaite porter à la connaissance du jury la détention d'un titre, d'un diplôme, d'un certificat de qualification professionnelle (CQP) ou des attestations de formation ;
- une déclaration sur l'honneur à compléter et à signer ;
- des documents illustrant la pratique professionnelle du candidat (facultatif)
- des annexes, si nécessaire.

Pour compléter ce dossier, le candidat dispose d'un site web en accès libre sur le site.

http://travail-emploi.gouv.fr/titres-professionnels

Sommaire

Exemples de pratique professionnelle

Développer la partie front-end d'une application web ou web mobile sécurisée **p.**

- Exercice de formation « **Exercice Cinema »** ..... p. 5
- Intitulé de l'exemple n° 2 **« Site internet pour un streamer »** ..... p. 8
- Intitulé de l'exemple n° 3 « **Exercice Session** »..... p. 10

Développer la partie back-end d'une application web ou web mobile sécurisée **p.**

- Exercice de formation **« Exercice Forum »** ..... p. 13
- Intitulé de l'exemple n° 2 **« Exercice Cinema »** ..... p. 16
- Intitulé de l'exemple n° 3 « **Déploiement d'un site** »..... p. 18

Titres, diplômes, CQP, attestations de formation *(facultatif)* p.

Déclaration sur l'honneur p.

Documents illustrant la pratique professionnelle *(facultatif)* p.

Annexes *(Si le RC le prévoit)* p.

EXEMPLES DE PRATIQUE PROFESSIONNELLE

Activité-type 1

Développer la partie front-end d'une application web ou web mobile sécurisée

Exemple n°1 ▸ **Exercice Cinema** (Annexe 1)

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

L'exercice cinema est un site présentant des fiches concernant des : films, acteurs, réalisateurs, genres et rôles. La mise en place de formulaire a permis d'ajouter des éléments de manière dynamique.

1. **Temporisation de sortie** :
   - **Controller** : Récupère la requête http, traite la requête SQL en se connectant à la base de données via pdo et inclut la vue.

2. **Templating** :
   - **Vue** :
     > **ob_start()** : Démarre la capture du contenu HTML/PHP généré dans une vue.
     > **ob_get_clean()** : Récupère le contenu et le stocke dans une variable.
     > **Affichage** : La vue est alors affichée avec le contenu dynamique dans un template global.

3. **Responsive** :
   - Conception des **interfaces utilisateurs adaptées** à différents types d'écrans (desktop, tablette, mobile).
   - Utilisation du CSS avec des **media queries** pour ajuster la mise en page, la taille des éléments, et la disposition en fonction de la largeur de l'écran.
   - Intégration de **composants flexibles** (grilles, flexbox) pour assurer une bonne adaptabilité du contenu.

2. Précisez les moyens utilisés :

Langage de balisage : **HTML5**

Langage de programmation : **PHP 8.3**

Langage de présentation : **CSS3**

Outil de gestion de la Base De Données : **HeidiSQL v12.8**

Système de gestion de BDD : **MySQL 8.0**

Langage de requêtage : **SQL**

3. Avec qui avez-vous travaillé ?

Les formateurs d'Elan Formation.

4. Contexte

Nom de l'entreprise, organisme ou association	***Elan Formation***
Chantier, atelier, service	**Equipe développeur web et web mobile**
Période d'exercice	**Du 21/10/2024 au 28/05/2025**

5. Informations complémentaires *(facultatif)*

Cet exercice a été effectué dans une **architecture MVC simplifiée**. Ainsi le traitement de la requête s'effectue dans le **Controller** et non dans un **Repository**.

Activité-type 1

Développer la partie front-end d'une application web ou web mobile sécurisée

Exemple n°2 ▸ **Site internet pour un streamer** (Annexe 2)

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Il m'a été demandé un portfolio permettant de présenter le dernier Twitch, la dernière vidéo Youtube, un envoi direct vers le serveur discord du streamer ainsi qu'une page « about » pour le présenter.

1. Mise en place de **wireframes**, **mockups** des différents **composants** ainsi que des propositions de différentes interactions (couleurs, typographie, icônes).

2**. Création de l'architecture** de l'application avec React, en structurant les composants de manière modulaire et réutilisable.

3. **Développement des composants principaux** : Header, Footer, About, YouTubeSection, TwitchSection, DiscordSection.

4. **Utilisation des APIs YouTube et Twitch** :
- Génération des **clés API** via la console développeur.
- **Authentification sécurisée** des requêtes avec token pour Twitch.
- **Affichage dynamique** des dernières vidéos YouTube et des streams en cours sur Twitch.
- Protection des clés d'API côté client en mettant en place des **variables d'environnement** dans le fichier .env.

5. **Mise en place de la navigation** entre les différentes pages du site à l'aide de **React Router**, avec une expérience fluide de type SPA (Single Page Application).

2. Précisez les moyens utilisés :

Outil de graphisme : **Figma**

Langage de programmation : **JS (ES6+)**

Framework : **React 18.3.1**

Gestion des routes : **React Router 6.11**

Requêtes HTTP vers les APIs : **Axios**

API : **Twitch et YouTube**

Sécurisation et gestion des clés d'API : **Environnement .env**

Gestionnaire de paquets : **yarn**

3. Avec qui avez-vous travaillé ?

J'ai réalisé ce projet en **totale autonomie**, de la conception à l'intégration.

Des **échanges réguliers avec le streamer** ont été menés en amont pour **analyser le besoin**. À cette occasion, j'ai présenté un **wireframe**, un **mockup**, ainsi que plusieurs **propositions de mise en page** et d'**effets d'interaction** (animations, hover, icônes).

4. Contexte

Nom de l'entreprise, organisme ou association ▸	**M Vilela William**
Chantier, atelier, service ▸	
Période d'exercice ▸	**Du 03/06/2024 au 28/06/2024**

5. Informations complémentaires *(facultatif)*

Texte

Activité-type 1

Développer la partie front-end d'une application web ou web mobile sécurisée

Exemple n°3 ▸ **Exercice Session** (Annexe 3)

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Session est une application interne à l'entreprise permettant de programmer des sessions de formations et ainsi planifier des formateurs, des apprenants et des modules.

1. **Maquettage** :
   - Réalisation d'un **mockup** fidèle à la version finale (desktop et mobile)
   - Création des différentes pages clés (fiches sessions, formulaires, modules, stagiaires, formateurs et catégories)
   - Intégration des contraintes de **responsive design**

2. **Expérience utilisateur :**
   - **Accès rapide** aux actions principales (ajout, modification consultation)
   - **Optimisation des formulaires** pour limiter les erreurs et faciliter la saisie (champs pré-remplis, boutons ajouter/supprimer, champs numériques contrôlés)

3. **Interface utilisateur :**
   - Mise en place sur **une seule page** des différentes parties de sessions de formations, permettant de planifier les modules et leur nombre de jour et des stagiaires pour une formation.
   - Application d'une **charte graphique uniforme** : couleurs, typographie, espacements harmonisés sur tout le projet.

4. **Twig** :
   - Création d'un **template global** contenant les block communs (header, menu et body)
   - **Affichage dynamique** des contenus : listes de sessions, modules, stagiaires... injectés depuis les contrôleurs Symfony via les variables Twig ({{ variable }}).

2. Précisez les moyens utilisés :

Outil de graphisme : **Figma**

Framework : **Symfony 7.2.1**

Moteur de template : **Twig 3.15**

Langage de balisage : **HTML5**

Langage de présentation : **CSS3**

Langage de programmation **: PHP 8.4**

Outil de gestion de la Base De Données : **HeidiSQL v12.8**

Système de gestion de BDD : **MySQL 8.0**

Langage de requêtage : **SQL**

3. Avec qui avez-vous travaillé ?

Les formateurs d'Elan Formation.

4. Contexte

Nom de l'entreprise, organisme ou association	**Elan Formation**
Chantier, atelier, service	**Equipe développeur web et web mobile**
Période d'exercice	**Du 21/10/2024 au 28/05/2025**

5. Informations complémentaires *(facultatif)*

Texte

Activité-type 2

Développer la partie back-end d'une application web ou web mobile sécurisée

Exemple n°1 ▸ **Exercice Forum** (Annexe 4)

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Forum est une plateforme permettant aux utilisateurs d'échanger entre eux. Afin d'assurer un environnement sain et respectueux, des modérateurs ont été mis en place.

1. **POO (Programmation Orientée Objet) :**
- Création de classes représentant les entités du projet : User, Forum, Post, Topic, Security, Home.
- Utilisation de méthodes pour encapsuler les comportements (listPostsByTopic, addPost, lockTopic, editProfile, etc)
- Appel des objets et de leurs méthodes dans les contrôleurs pour manipuler les données de manière structurée.

2. **Model Vue Controller (MVC**) :
- **Controller** : fichiers prenant en charge les requêtes HTTP, exécutant la logique métier, et appelant les vues nécessaires.
- **Model** : fichiers PHP gérant la connexion à la base de données et les requêtes (CRUD pour les utilisateurs, messages, sujets...).
- **Vue** : templates HTML/PHP affichant les données dynamiques (liste des messages, formulaires, espace profil, etc)

3. **Sécurité** :
- Mise en place d'un système d'authentification :
  > Lors de l'enregistrement : mot de passe **hashé** grâce à password_hash()
  > Lors de la connexion : comparaison de **l'empreinte numérique** enregistrée avec password_verify())
  > Vérification de **l'unicité du pseudo et de l'email** : findOneByNickname(), findOneByEmail()

- Contrôle des accès via la **gestion de rôles** (utilisateur/administrateur) avec **restrictions d'actions** selon le niveau (getRole(), updateRoleForUser()).
- **Validation** et **filtrage des données** utilisateurs pour éviter les failles XSS et injections SQL.
  > Vérification des champs : filter_input() (ex : FILTER_VALIDATE_REGEXP)
  > Rendre **sain** les champs : ex : FILTER_SANITIZE_FULL_SPECIAL_CHARS
  > **Echappement** en sortie : htmlspecialchars()

2. Précisez les moyens utilisés :

Framework : **Elan**

Langage de balisage : **HTML5**

Langage de programmation : **PHP 8.4**

3. Avec qui avez-vous travaillé ?

Les formateurs d'Elan.

4. Contexte

Nom de l'entreprise, organisme ou association ▸ ***Elan Formation***

Chantier, atelier, service ▸ **Equipe développeur web et web mobile**

Période d'exercice ▸ **Du 21/10/2024 au 28/05/2025**

5. Informations complémentaires *(facultatif)*

Texte

Activité-type 2

Développer la partie back-end d'une application web ou web mobile sécurisée

Exemple n°2 ▸ **Exercice Cinema** (Annexe 5)

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

L'exercice cinema est un site présentant des fiches concernant des : films, acteurs, réalisateurs, genres et rôles. La mise en place de formulaire a permis d'ajouter des éléments de manière dynamique.

1. **Connexion à une base de données**
   - Utilisation de PHP Data Object (PDO) pour gérer la connexion à la base de données
   - Création de la **class Connect** pour établir le lien entre le projet et la base de données
   - Appel à cette classe dans les Controllers

2. **Mise en place de requêtes SQL préparées**
   - Utilisation de la **méthode prepare()** : création des requêtes paramétrées (ex : SELECT, INSERT INTO, DELETE)
   - Passage des **valeurs dynamiques** (:name_acteur, :firstname_acteur) pour sécuriser les données
   - Exécution des requêtes avec la **méthode execute()** avec le tableau associatif des paramètres à injecter

3. **Appel aux requêtes**
   - Les requêtes sont appelées dans les Controllers.
   - Pour SELECT : Les résultats sont récupérées avec **fetch()** ou **fetchAll()**
     Les données récupérées sont ensuite transmises à une vue pour affichage.
   - Pour INSERT INTO, DELETE etc : Les résultats sont directement enregistrés, modifiées ou supprimées en base de données.

2. Précisez les moyens utilisés :

Langage de balisage : **HTML5**

Langage de programmation : **PHP3**

Outil de gestion de la Base De Données : **HeidiSQL v12.8**

Système de gestion de BDD : **MySQL 8.0**

3. Avec qui avez-vous travaillé ?

Les formateurs d'Elan Formation

4. Contexte

Nom de l'entreprise, organisme ou association ▸	***Elan Formation***
Chantier, atelier, service ▸	**Equipe développeur web et web mobile**
Période d'exercice ▸	**Du 21/10/2024 au 28/05/2025**

5. Informations complémentaires *(facultatif)*

Cet exercice a été effectué dans une **architecture MVC simplifiée**. Ainsi le traitement de la requête s'effectue dans le **Controller** et non dans un **Repository**.

Activité-type 2

Développer la partie back-end d'une application web ou web mobile sécurisée

Exemple n°3 ▸ **Déploiement d'un site Internet** (Annexe 6)

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Dans le cadre de la création de mon portfolio, j'ai souhaité le mettre en ligne afin que les employeurs puissent le consulter librement.

1. **Hébergement web**

- Achat d'un **nom de domaine**
- Souscription à une offre **d'hébergement web**
- **Configuration des Domain Name System (DNS)** pour associer le nom de domaine à l'hébergement
- Gestion des paramètres techniques sur le dashboard

2. **Client File Transfer Protocol (FTP) / Secure File Transfer Protocol (SFTP)**

- Installation du **logiciel FileFilla** permettant le transfert des fichiers
- Connexion au serveur d'hébergement avec les identifiants fournis par l'hébergeur
- Transfert des fichiers du site (dossier complet ou fichiers modifiés)

3. **Vérification de la mise en ligne**

- Ouvrir un navigateur
- Saisir l'url correspondante au nom de domaine
- Vérifier que le site s'affiche correctement et que les éventuelles modifications sont bien prises en compte

2. Précisez les moyens utilisés :

Hébergeur web : **OVH**

Logiciel de transfert : **Client FTP/SFTP FileZilla v3.65.0**

Navigateur web **: Mozilla Firefox v118.0.1**

3. Avec qui avez-vous travaillé ?

Néant

4. Contexte

Nom de l'entreprise, organisme ou association	**Me Renau Laury**
Chantier, atelier, service	**Portfolio personnel**
Période d'exercice	**Du 01/09/2023 au 30/11/2023**

5. Informations complémentaires *(facultatif)*

Texte

Titres, diplômes, CQP, attestations de formation

(facultatif)

Intitulé	Autorité ou organisme	Date
Cliquez ici.	Cliquez ici pour taper du texte.	Cliquez ici pour sélectionner une date.
Cliquez ici.	Cliquez ici pour taper du texte.	Cliquez ici pour sélectionner une date.
Cliquez ici.	Cliquez ici pour taper du texte.	Cliquez ici pour sélectionner une date.
Cliquez ici.	Cliquez ici pour taper du texte.	Cliquez ici pour sélectionner une date.
Cliquez ici.	Cliquez ici pour taper du texte.	Cliquez ici pour sélectionner une date.
Cliquez ici.	Cliquez ici pour taper du texte.	Cliquez ici pour sélectionner une date.
Cliquez ici.	Cliquez ici pour taper du texte.	Cliquez ici pour sélectionner une date.
Cliquez ici.	Cliquez ici pour taper du texte.	Cliquez ici pour sélectionner une date.
Cliquez ici.	Cliquez ici pour taper du texte.	Cliquez ici pour sélectionner une date.
Cliquez ici.	Cliquez ici pour taper du texte.	Cliquez ici pour sélectionner une date.

Déclaration sur l'honneur

Je soussigné ..... *Me RENAU Laury* .....,

déclare sur l'honneur que les renseignements fournis dans ce dossier sont exacts et que je suis l'auteur(e) des réalisations jointes.

Fait à ..... *Saverne* ..... le ..... 16/05/2025 .....

pour faire valoir ce que de droit.

Signature :

Documents illustrant la pratique professionnelle

(facultatif)

Intitulé
Cliquez ici pour taper du texte.

ANNEXES

FRONT END

Annexe1 : Exercice Cinema
Annexe 2 : Site internet pour un streamer
Annexe 3 : Exercice Session

BACK END

Annexe 4 : Forum
Annexe 5 : Exercice Cinema
Annexe 6 : Déploiement d'un site

Annexe 1 : Exercice Cinema

```php
public function listFilms()
{
    // Connexion à la base de données via la classe Connect
    $pdo = Connect::seConnecter();
    $requete = $pdo->query("
      SELECT
        film.id_film,
        film.titre,
        GROUP_CONCAT(genre.libelle_genre) AS tous_genre,
        affiche,
        CONCAT(DAY(film.date_sortie_fr), '-', MONTH(film.date_sortie_fr), '-', YEAR(film.date_sortie_fr)) AS release_date
    FROM film
    INNER JOIN genre_film ON film.id_film = genre_film.id_film
    INNER JOIN genre ON genre_film.id_genre = genre.id_genre
    GROUP BY film.id_film, film.titre
    ORDER BY titre ASC
    ");

    require "view/listFilms.php"; //necessaire pour récuperer la vue qui nous intéresse
}
```

```php
ob_start();
$film = $requete->fetch();
?>

<div class="intro">
  <p>Il y a <?= $requete->rowCount() ?> films.</p>
  <div class="group_btn">
    <button onclick="window.location.href='index.php?action=addFilm';">Ajouter un film</button>
    <button onclick="window.location.href='index.php?action=addCasting';">Ajouter un casting</button>
  </div>
</div>

<section class="princ">
  <?php
   foreach ($requete->fetchAll() as $film) { ?>
    <div class="group">
      <div class="text">
        <p><a href="index.php?action=detFilm&id=<?= $film['id_film'] ?>"><?= $film["titre"] ?></a></p>
        <p>Sorti le : <?= $film["release_date"] ?></p>
      </div>
      <p class="imgContain"><img class="img_aff" src="<?= $film["affiche"] ?>" alt="Affiche du film . $film['titre']" /></p>
    </div>

  <?php } ?>
</section>

<?php

$titre = "<h1 class='titreH1'>Liste des films</h1>";
$contenu = ob_get_clean(); //Fin de la vue
```

Responsive :

```css
.princ {
    display: grid;
    grid-template-columns: repeat(4, 1fr);
    grid-template-rows: 1fr;
    gap: 30px 10px;
}
```

```css
@media (max-width: 720px) {
    .princ, .gpGenre {
        grid-template-columns: repeat(2,1fr);
        grid-template-rows: auto;
    }
```

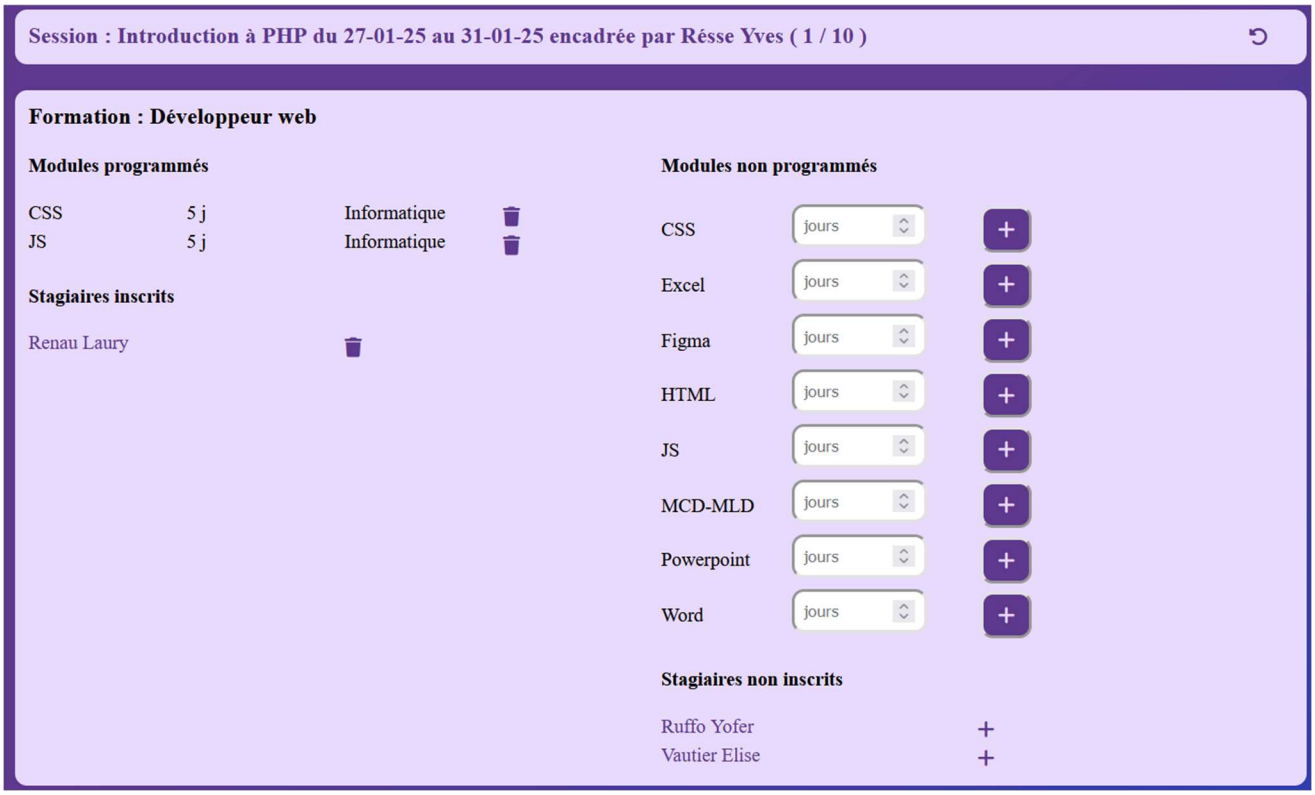
```css
@media (max-width: 1241px) {
.princ, .gpGenre {
    grid-template-columns: repeat(3,1fr);
    grid-template-rows: auto;
}
}
```

```css
@media (max-width: 520px) {
    .princ, .gpGenre {
        grid-template-columns: 1fr;
        grid-template-rows: auto;
    }
```

Annexe 2 : Site internet pour un streamer

```
// useEffect : déclenche une fonction au montage du composant
useEffect(() => {
  async function fetchLastVideo() {
    try {
      // Étape 1 : Récupération d'un token d'accès via le client ID et le secret
      const tokenResponse = await axios.post('https://id.twitch.tv/oauth2/token', null, {
        params: {
          client_id: CLIENT_ID,
          client_secret: CLIENT_SECRET,
          grant_type: 'client_credentials'
        }
      });
      const accessToken = tokenResponse.data.access_token;

      // Étape 2 : Récupération des données utilisateur pour obtenir l'ID Twitch
      const userResponse = await axios.get('https://api.twitch.tv/helix/users', {
        headers: {
          'Client-ID': CLIENT_ID,
          'Authorization': `Bearer ${accessToken}`
        },
        params: {
          login: USER_LOGIN
        }
      });
      const userId = userResponse.data.data[0].id;

      // Étape 3 : Récupération de la dernière vidéo de l'utilisateur via son ID
      const videoResponse = await axios.get('https://api.twitch.tv/helix/videos', {
        headers: {
          'Client-ID': CLIENT_ID,
          'Authorization': `Bearer ${accessToken}`
        },
        params: {
          user_id: userId,
          first: 1 // On ne récupère que la dernière vidéo
        }
      });

      // Stockage de la vidéo dans l'état
      setLatestVideo(videoResponse.data.data[0]);
    } catch (error) {
      // Gestion des erreurs d'appel à l'API
      console.error('Erreur lors de la récupération des données de l\'API Twitch:', error);
      setError('Erreur lors de la récupération des données de l\'API Twitch');
    }
  }

  // Lancement de la fonction d'appel API
  fetchLastVideo();
}, []); // Le tableau vide indique que cet effet ne s'exécute qu'une fois (au montage)
```

Annexe 3 : Exercice Session




```
    {% block javascripts %}
        {% block importmap %}
            {{ importmap('app') }}
        {% endblock %}
    {% endblock %}
</head>


    <nav id="navbar">
        <div id="welcome">
            <p>Bienvenue <a href="{{ path('app_profile') }}">toi !</a></p>
            <p><a href="{{ path('app_login') }}">Se déconnecter</a></p>
        </div>

        <ol>
            <li><a href="{{ path('app_home') }}">Accueil</a></li>
            <li><a href="{{ path('app_session') }}">Sessions</a></li>
            <li><a href="{{ path('app_training') }}">Formations</a></li>
            <li><a href="{{ path('app_module') }}">Modules</a></li>
            <li><a href="{{ path('app_category') }}">Catégories</a></li>
            <li><a href="{{ path('app_trainer') }}">Formateurs</a></li>
            <li><a href="{{ path('app_intern') }}">Stagiaires</a></li>
        </ol>

        <div class="burger_button">
            <div class="line"></div>
            <div class="line"></div>
            <div class="line"></div>
        </div>
    </nav>



    {% block body %}{% endblock %}
```

Annexe 4 : Exercice Forum

```php
//on filtre
$nickname = filter_input(INPUT_POST, "nickname", FILTER_SANITIZE_FULL_SPECIAL_CHARS);
$email = filter_input(INPUT_POST, "email", FILTER_VALIDATE_EMAIL);
$password1 = filter_input(INPUT_POST, "password1", FILTER_VALIDATE_REGEXP, array("options" => array("regexp" => "/^(?=.*[A-Z])(?
=.*\d)(?=.*[\W_]).{12,}$/")));
// ^: debut de chaine, $ : fin de chaine, (?=.*[A-Z]) : lettres, (?=.*\d) : digit, (?=.*[\W_]) : caracteres speciaux, {12,} :
12 caracteres
$password2 = filter_input(INPUT_POST, "password2", FILTER_VALIDATE_REGEXP, array("options" => array("regexp" => "/^(?=.*[A-Z])(?
=.*\d)(?=.*[\W_]).{12,}$/")));
$terms = isset($_POST['terms']) ? true : false;  // Vérifie si la case est cochée

if (!$terms) {
    $errorMessage = "Vous devez accepter les termes et conditions pour vous inscrire.";
}

//on les vérifie
if (!$errorMessage && $nickname && $email && $password1 && $password2) {

    $verifyNickname = $user->findOneByNickname($nickname);
    $verifyEmail = $user->findOneByEmail($email);

    if (!$verifyNickname && !$verifyEmail) { //Vérification si l'un ou l'autre n'existe pas en bdd
        // add in bdd
        if ($password1 == $password2 && strlen($password1) >= 12) {
            $user->add([
                "nickname" => $nickname,
                "email" => $email,
                "password" => password_hash($password1, PASSWORD_DEFAULT)
            ]);

            Session::addFlash("success", "Compte créé avec succès. Veuillez vous connecter.");
            $this->redirectTo("security", "login");
            exit();
        } else {
```

```php
<?php if (isset($errorMessage)) { ?>
    <div class="error-message">
        <p><?php echo htmlspecialchars($errorMessage); ?></p>
    </div>
<?php } ?>

<div class="blockForm title ">
    <p class="mailLogin"><label for="email">Email</label></p>
    <input type="email" name="email" id="email">
</div>

<div class="blockForm title">
    <p class="mdpLogin"><label for="password">Mot de passe</label></p>
    <input type="password" name="password" id="password">
</div>
```

Annexe 5 : Exercice Cinema

```php
abstract class Connect
{
    const HOST = 'localhost';
    const DB = 'cinemalily';
    const USER = 'root';
    const PASS = '';
```

```php
public static function seConnecter()
{
    try {
        return new \PDO(   //le \ indique au framework que PDO est une classe native
            'mysql:host=' . self::HOST . ';dbname=' . self::DB . ';charset=utf8',
            self::USER,
            self::PASS
        );
    } catch (\PDOException $ex) {
        return $ex->getMessage();
    }
}
```

```php
public function detActeur($id)
{
    $pdo = Connect::seConnecter();

    $requete2 = $pdo->prepare("
      SELECT DISTINCT genre.id_genre, film.id_film,
            role.id_role,
            film.titre,
            film.affiche,
            role.personnage,
            role.photo,
            GROUP_CONCAT(genre.libelle_genre) AS tous_genre
        FROM film
        INNER JOIN casting ON film.id_film = casting.id_film
        INNER JOIN acteur ON casting.id_acteur = acteur.id_acteur
        INNER JOIN role ON casting.id_role = role.id_role
        INNER JOIN genre_film ON film.id_film = genre_film.id_film
        INNER JOIN genre ON genre_film.id_genre = genre.id_genre
        WHERE casting.id_acteur = :id
        GROUP BY film.id_film, film.titre, film.affiche, role.id_role, role.personnage, genre.id_genre
    ");

    $requete2->execute(["id" => $id]);

    require "view
```

```php
public function addActeur()
    {
        $pdo = Connect::seConnecter();

        if (isset($_POST['submit'])) {
            $name_acteur = filter_input(INPUT_POST, 'name_acteur', FILTER_SANITIZE_FULL_SPECIAL_CHARS);
            $firstname_acteur = filter_input(INPUT_POST, 'firstname_acteur', FILTER_SANITIZE_FULL_SPECIAL_CHARS);
            $sexe = filter_input(INPUT_POST, 'sexe', FILTER_SANITIZE_FULL_SPECIAL_CHARS);
            $ddn_acteur = filter_input(INPUT_POST, 'ddn_acteur', FILTER_SANITIZE_FULL_SPECIAL_CHARS);
            $url_acteur = filter_input(INPUT_POST, 'url_acteur', FILTER_VALIDATE_URL);


            if ($name_acteur && $firstname_acteur && $sexe && $ddn_acteur && $url_acteur) {
                $requete = $pdo->prepare("
                INSERT INTO personne (nom, prenom, sexe, date_naissance, photo)
                VALUES (:name_acteur, :firstname_acteur, :sexe, :ddn_acteur, :url_acteur)
                ");

                $requete->execute([
                    "name_acteur" => $name_acteur,
                    "firstname_acteur" => $firstname_acteur,
                    "sexe" => $sexe,
                    "ddn_acteur" => $ddn_acteur,
                    "url_acteur" => $url_acteur
                ]);

                $idPersonne = $pdo->lastInsertId();

                header("Location: index.php?action=listActeurs");
                exit();
            }
        }

        require "view/addActeur.php";
```

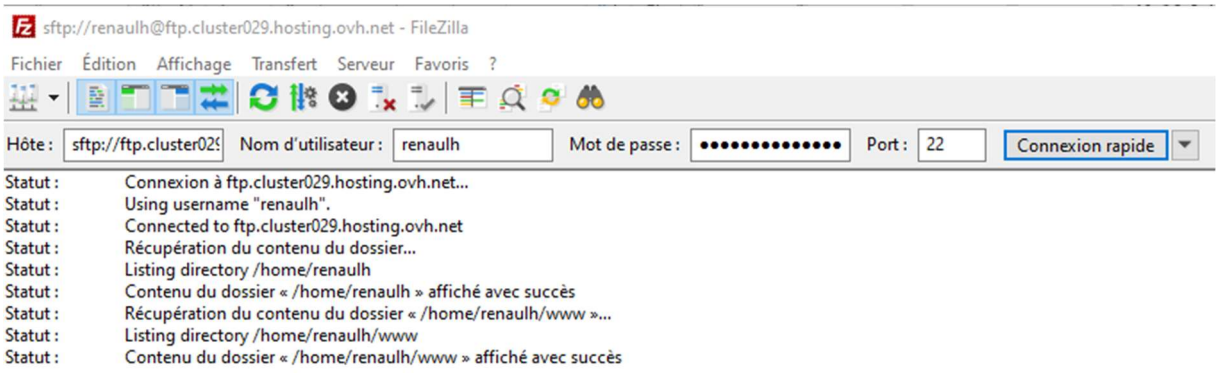
Annexe 6 : Déploiement d'un site internet

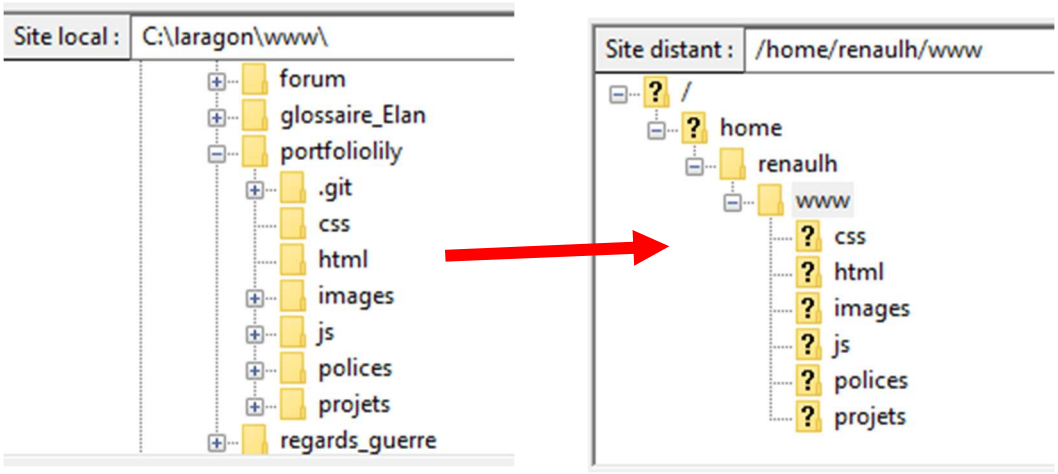
Serveur FTP et SFTP :
ftp.cluster029.hosting.ovh.net

Port FTP : 21
Port SFTP : 22

Lien FTP vers le cluster :
ftp://renaulh@ftp.cluster029.hosting.ovh.net:21/

Chemin du répertoire home :
/home/renaulh





https://www.renaulaury.fr