|  |  |  |
| --- | --- | --- |
|  |  |  |
| *Nom de naissance* |  | Shilkov |
| *Nom d’usage* |  | Shilkov |
| *Prénom* |  | Konstantin |
| *Adresse* |  | 5 boulevard Pablo Picasso 94000 Créteil |
|  |  |

|  |  |
| --- | --- |
| **Titre professionnel visé** | |
|  | |
| Développeur Web et Web mobile | |
|  | |
| **Modalité d’accès :** | |
|  | |
|  | 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**](http://travail-emploi.gouv.fr/titres-professionnels) |

|  |  |  |  |
| --- | --- | --- | --- |
| **Sommaire** | | | |
| **Exemples de pratique professionnelle** | | |
| **Activité n° 1 : Développer la partie front-end d’une application web ou web mobile en intégrant les recommandations de sécurité.** | | **p.** | **6** |
|  |  Compétence n° 1 : Maquetter une application  p. | p. | 6 |
|  |  Compétence n° 2 : Réaliser une interface utilisateur web statique et adaptable  p. | p. | 11 |
|  |  Compétence n° 3 : Développer une interface utilisateur web dynamique  p | p. | 16 |
|  |  |  |  |
| **Activité n° 2 : Développer la partie back-end d’une application web ou web mobile en intégrant les recommandations de sécurité.** | | **p.** | **25** |
|  |  Compétence n° 5 : Créer une base de données  p. | p. | 23 |
|  |  Compétence n° 6 : Développer les composants d’accès aux données  p. | p. | 34 |
|  |  Compétence n° 7 : Développer la partie back-end d’une application web ou web mobile p | p. | 41 |
|  |  |  |  |
|  |  |  |  |
| **Titres, diplômes, CQP, attestations de formation** *(facultatif)* | | **p.** | 57 |
| **Déclaration sur l’honneur** | | **p.** | 58 |
| **Documents illustrant la pratique professionnelle** *(facultatif)* | | **p.** | 59 |
| **Annexes** *(Si le RC le prévoit)* | | **p.** | 60 |

**Exemples de pratique**

**professionnelle**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Activité-type** | **1** | **Développer la partie front-end d’une application web ou web mobile en intégrant les recommandations de sécurité** | | | |
| ***Compétence n°1***  | | ***Maquetter une application*** | | | |
|  | | | | | |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | | | | |
| Pour l'introduction j'ai présenté l'objectif du projet qui est de créer un site web commercial qui facilite la possibilité d'acheter des plantes décoratives d'intérieur ou des jardins, en leurs proposant toutes sortes d'informations sur les types des plantes, leur caractéristique des soins et leur prix.  J'ai débuté par créer une planche de tendance (**Moodboard**), avec le logiciel **lnkscape**, où l'on retrouve le logo, les couleurs et quelques illustrations qui seront intégrées au projet.  Un **moodboard** est un collage visuel ou une collection d'images, de couleurs et de textures utilisées pour transmettre l'apparence et l'ambiance d'un concept ou d'une idée de conception. Il est généralement créé pour explorer et communiquer différentes directions de conception et pour aider à aligner les parties prenantes autour d'une vision partagée. Un moodboard peut inclure des photographies, des découpes de magazines, des échantillons de tissus, des échantillons de typographie et d'autres éléments visuels qui aident à capturer l'esthétique et l'ambiance souhaitées d'un projet de conception.    *Figure 1 : le MoodBoard*  Ensuite j'ai créé une **charte graphique** avec le logiciel **InkScape** ce qui consiste à détailler tous les points à respecter qui regroupe l'ensemble des codes définissant l'identité visuelle du projet.    *Figure 2 : La charte graphique* | | | | | |
| Puis j’ai créé des **personas** avec le logiciel **Figma** correspondant aux profils d’utilisateurs qui pourraient se rendre sur le site.  Une **persona** est un personnage fictif qui représente un utilisateur typique d'un produit ou d'un service. Les personas sont créées en menant des recherches et en collectant des informations sur les besoins, les objectifs, les comportements et les caractéristiques du public cible. Elles sont utilisées pour aider les designers, les développeurs et les autres parties prenantes à comprendre les besoins et les perspectives des différents groupes d'utilisateurs et à créer des produits et des services qui répondent à leurs besoins.      *Figure 3 : Les personas*  Les **personas** incluent généralement des informations démographiques telles que l'âge, le sexe, l'éducation et le revenu, ainsi que des informations psychographiques telles que les valeurs, les intérêts et les motivations. Elles peuvent également inclure des informations sur les objectifs, les tâches, les frustrations et les points douloureux de l'utilisateur, ainsi que leurs canaux de communication préférés et leur utilisation de la technologie.  En créant des **personas**, les designers et les développeurs peuvent mieux se mettre à la place de leurs utilisateurs et concevoir des produits et des services qui répondent à leurs besoins et à leurs préférences. Les personas peuvent également aider les parties prenantes à prioriser les fonctionnalités et les fonctionnalités, à identifier les opportunités et les contraintes de conception, et à tester et valider les décisions de conception.  Aprés ça j’ai créé la maquette de la page d’accueil du projet en me servant de l’application **Figma** pour la mise en page.    *Figure 4 : Maquette de la page d’accueil de mon projet Floreil*  Pour finir avec cette compétence j’ai commencé à coder mon projet en **HTML 5** (*HyperText Markup Language*) et en **CSS 3** (*Cascading Style Sheets*) avec l’éditeur de code **Visual Studio Code** et pour sauvegarder l’avancée de mon travail en créant des dépôts j’ai utilisé **Github Desktop**.    *Figure 5 : Code HTML / Code CSS / Dépôt de Visual Studio Code vers GitHub Desktop* | | | | | |
|  | | | | | |
|  | | | | |
|  | | | | | |
| **2. Précisez les moyens utilisés :** | | | | | |
| Voici les moyens que j’ai utilisé pour cette première compétence: | | | | | |
| Logiciel: *InkScape, Visual studio code.*  Site web: *GitHub desktop.*  Application: *Figma.*  Langage: *HTML5, CSS3.* | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | | |
| Pour les compétences de 1 à 8 j’ai travaillé seul en me documentant sur internet | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | |
|  | | | | | |
| **4. Contexte** | | | | | |
|  | | | |  | |
| Nom de l’entreprise, organisme ou association | | | | *Formation AFPA* | |
| Chantier, atelier, service | | | *Cliquez ici pour taper du texte.* | | |
| Période d’exerciceDu : *05/09/2022*au :*28/04/2023* | | | | | |
|  | | | | |
|  | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Activité-type** | **1** | **Développer la partie front-end d’une application web ou web mobile en intégrant les recommandations de sécurité** | | | |
| ***Compétence n°2***  | | ***Réalisation d’une interface utilisateur web statique et adaptable*** | | | |
|  | | | | | |
|  | | | | | |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | | | | |
| En développement web, une **interface utilisateur statique** fait référence à un site web ou à une application web qui possède une mise en page, un design et un contenu fixe qui ne change pas à moins que le développeur web ne le mette à jour manuellement. Une interface statique est généralement codée en HTML et CSS et ne comprend pas d'éléments dynamiques tels que des animations, du contenu généré par l'utilisateur ou des mises à jour en temps réel.  Une **interface utilisateur adaptable** est conçue pour fonctionner efficacement sur différents appareils et tailles d'écran. Avec l'utilisation croissante des appareils mobiles pour accéder au web, il est important pour les développeurs web de s'assurer que leurs interfaces peuvent s'adapter aux écrans plus petits sans perdre de fonctionnalités ou d'utilisabilité. Les interfaces adaptables utilisent généralement des techniques de conception réactive telles que des mises en page fluides, des requêtes multimédias et des images et du texte flexibles pour ajuster la mise en page et le contenu en fonction de la taille et de l'orientation de l'écran de l'appareil.  Pour cette deuxième compétence j’ai commencé par **maquetter** mon projet en format **mobile**, **tablette** et **desktop** toujours avec le logiciel **Figma**. | | | | | |
| *Figure 6 : La maquette au format Mobile, Tablette et Desktop*  Et par la suite avec l’éditeur de code Visual Studio Code, j’ai continué le codage de la page d’accueil de mon projet, en y ajoutant les média queries pour le rendre adaptable sur tous les types de supports : Mobile, Tablette et Desktop.  Mais après j’ai décidé refaire mon projet en utilisant **Bootstrap** (un framework front-end puissant) car il a une fonctionnalité de conception adaptive que media queries. En plus ce framework a beaucoup de fonctionnalité qui me donnent possibilité d’améliorer apparence et caractéristiques de mon projet.    *Figure 7 : Le système de grille de Bootstrap*  J’ai intégré du contenu multimédia, exemple une Vidéo provenant de “**youtube**“ pour accentuer la visibilité de mon site par les utilisateurs.    *Figure 8 : Une vidéo youtube*  J’ai aussi intégré un **menu hamburger** à mon bandeau de navigation (*nav-bar*), pour faciliter l’accès aux informations disponibles sur le site.    *Figure 9 : Menu hamburger*  J’ai également intégré des icônes cliquables dans mon bandeau de pied de page (*footer*), qui redirigent vers les **réseaux sociaux** du type : Twitter, Instagram et Facebook pour offrir une plus grande visibilité à ce projet sur le web.    *Figure 10 : Icône réseaux sociaux*  J’ai poursuivi le codage du site en intégrant du code **SCSS** de **Bootstrap**. Avec **SASS** (*Syntactically awesome stylesheets*) on peut imiter la structure du html il faut commencer par intégrer un flux de travail de compilateur Sass à votre projet. Ce qui permet d’obtenir des goodies exemple : conception modulaire, fichiers inclus, mixins, héritage, et de pouvoir mieux se repérer dans le code en ayant moins de lignes en introduisant des variables.    *Figure 11 : Code Scss*  En conclusion, je voudrais souligner que d’avoir **une interface utilisateur adaptable** est important pour offrir une bonne expérience utilisateur sur différents appareils et tailles d'écran. Une interface mal conçue qui n'est pas adaptable peut être frustrante pour les utilisateurs et peut entraîner une perte d'affaires ou d'engagement. En concevant une interface adaptable, les développeurs web peuvent s'assurer que leurs sites web et applications web sont accessibles et conviviaux, quel que soit l'appareil ou la plateforme utilisée. | | | | | |
|  | | | | | |
|  | | | | |
|  | | | | | |
| **2. Précisez les moyens utilisés :** | | | | | |
| Voici les moyens que j’ai utilisé pour cette deuxième compétence: | | | | | |
| Logiciel: *Visual studio code.*  Site web: *GitHub, Xtensio.com.*  Application: *Figma.*  Langage: *HTML5, CSS3, SCSS (Bootstrap)* | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | | |
| Pour les Compétences de 1 à 8 j’ai travaillé seul en me documentant sur internet | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | |
|  | | | | | |
| **4. Contexte** | | | | | | |
|  | | | |  | | |
| Nom de l’entreprise, organisme ou association | | | | *Formation AFPA* | | |
| Chantier, atelier, service | | | *Cliquez ici pour taper du texte.* | | | |
| Période d’exerciceDu : *05/09/2022*au :*28/04/2023* | | | | | | |
|  | | | | |
|  | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Activité-type** | **1** | **Développer la partie front-end d’une application web ou web mobile en intégrant les recommandations de sécurité** | | | |
| ***Compétence n°3***  | | ***Développer une interface utilisateur web dynamique*** | | | |
|  | | | | | |
|  | | | | | |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | | | | |
| Le développement **d'une interface utilisateur web dynamique** est un aspect crucial de la conception de sites web modernes et conviviaux. Les utilisateurs s'attendent à des sites web réactifs et interactifs qui leur permettent de naviguer facilement et efficacement. Pour répondre à ces attentes, il est important de développer des interfaces utilisateur qui sont visuellement attrayantes, intuitives et engageantes.  Pour développer une interface utilisateur web dynamique, vous devez avoir une bonne compréhension de **HTML**, **CSS** et **JavaScript**. Ces langages sont essentiels pour créer des pages web dynamiques et interactives. Vous pouvez utiliser une variété d'outils et de frameworks pour simplifier le processus de développement, tels que **React**, **Vue.js** ou **Angular**.  Pour la première partie de cette compétence(**3A**), toujours à partir de l’éditeur de code **Visual Studio Code** j’ai utilisé **Bootstrap** qui est un framework qui permet de simplifier l’action du développeur web en lui donnant accès à une multitude de codes déjà prêts à être intégrés à son projet pour la navigation et les éléments interactifs de celui-ci. | | | | | |
| *Figure 12 : Page d’accueil de la documentation Bootstrap, et un exemple de code pour une nav-bar avec menu-hamburger*  Cela m’a permis de rendre mon projet dynamique en y intégrant un formulaire de “sign up” intéractif à partir de **Bootstrap.**    *Figure 13 : Un formulaire de sign up intéractif*  Ensuite pour la seconde partie de cette compétence(**3B**), toujours à partir de l’éditeur de code **Visual Studio Code** j’ai repris mon projet précédent et je l’ai recréé en projet **Create React App** dans un environnement **React**.    *Figure 14 : Page d'accueil React*  **React** est une bibliothèque JavaScript populaire pour la construction d'interfaces utilisateur. Elle a été développée par Facebook et est utilisée par de nombreuses entreprises pour créer des applications web dynamiques et interactives. React permet aux développeurs de créer des composants réutilisables qui peuvent être facilement rendus et mis à jour en fonction des changements dans les données de l'application. Elle utilise également un **DOM** virtuel (**Document Object Model**) pour optimiser les performances et réduire le nombre de mises à jour nécessaires sur le DOM réel.      *Figure 15 : Exemle de code React*  Voici le rendu de ma page d’accueil et page des plantes avec mon projet **React**.    *Figure 16 : Visuel de mon projet React*  Pour tester la fonctionnalité de mon projet j’utilise **JSON Server** (bibliothèque Node.js).    *Figure 16 : Page de JSON Server et le code de mon exemple de JSON Server*  **JSON Server** vous permet de configurer rapidement une API (interface de programmation d'application) RESTful avec une interface **CRUD** (Créer, Lire, Mettre à jour, Supprimer) entièrement fonctionnelle. Il vous permet de créer un serveur API fictif ou simulé en utilisant un fichier **JSON** comme source de données. Cela est utile pour les tests, les prototypes et la création d'applications qui nécessitent un serveur sans avoir à créer une infrastructure backend complète. **JSON Server** est facile à configurer et à paramétrer, et il fournit une méthode simple pour définir des routes et gérer les **requêtes HTTP**. Il prend également en charge des fonctionnalités avancées telles que la pagination, le filtrage et le tri des données.  Enfin J'ai refait toutes les éléments répétitifs de mon projet en utilisant fonctionnalité de React.  En plus, j’ai refait mon projet en accordance de conception **AMP** (**application monopage**) en utilisant **React Router**.    *Figure 17 : Page de React Router*  **React Router** est une bibliothèque de routage pour les applications React. Elle permet de définir des routes et des URL qui correspondent à des composants ou des vues spécifiques de votre application. Cela signifie que lorsqu'un utilisateur clique sur un lien ou saisit une URL dans le navigateur, le composant approprié est rendu sans avoir besoin de recharger l'ensemble de la page. React Router fournit une API simple pour définir et naviguer entre les routes, y compris la prise en charge de routes imbriquées et de routage dynamique basé sur les paramètres d'URL. Elle permet de créer facilement des applications à page unique complexes et interactives avec plusieurs vues et états. | | | | | |
| *Figure 18 : Extrait de code de mon projet utilisant React Router* | | | | | |
|  | | | | |
|  | | | | | |
| **2. Précisez les moyens utilisés :** | | | | | |
| Voici les moyens que j’ai utilisé pour cette troisième compétence: | | | | | |
| Documentation: *Site de Bootstrap, site de Vuejs et site de Vite.*  Logiciel: *Visual studio code.*  Site web: *Github desktop.*  Application: *Figma.*  Framework: *Bootstrap, Vite Vue.js.*  Langage: *HTML5, CSS3, SCSS, JavaScript, JSON.* | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | | |
| Pour les Compétences de 1 à 8 j’ai travaillé seul en me documentant sur internet | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | |
|  | | | | | |
| **4. Contexte** | | | | | |
|  | | | |  | |
| Nom de l’entreprise, organisme ou association | | | | *Formation AFPA* | |
| Chantier, atelier, service | | | *Cliquez ici pour taper du texte.* | | |
| Période d’exerciceDu : *05/09/2022*au :*28/04/2023* | | | | | |
|  | | | | |
|  | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Activité-type** | **2** | **Développer la partie back-end d’une application web ou web mobile en intégrant les recommandations de sécurité.** | | | |
| ***Compétence n°5***  | | ***Créer une base de données*** | | | |
|  | | | | | |
|  | | | | | |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | | | | |
| Pour cette première compétence de la partie **Back-end**, j’ai commencé par comprendre le fonctionnement d’un site web et savoir faire la différence entre la partie clients et la partie serveurs, j’ai également appris à différencier un site **Statique** d’un site **Dynamique**. | | | | | |
| Dynamic Website Versus Static Website - iBrandox  *Figure 19 : La différence entre un site statique et un site dynamique*  Les sites **web statiques** sont constitués de pages web préconçues qui affichent le même contenu pour tous les utilisateurs. Ils ne nécessitent pas de traitement côté serveur ni d'interaction avec une base de données, et le contenu est généré à l'aide de technologies côté client telles que **HTML**, **CSS** et **JavaScript**. Les sites web statiques peuvent être hébergés sur n'importe quel serveur web sans aucune exigence spéciale.  Les sites **web dynamiques**, en revanche, génèrent du contenu à la volée et peuvent afficher un contenu différent pour différents utilisateurs en fonction de leurs préférences, de leurs actions ou d'autres variables. Ils nécessitent un traitement côté serveur et une interaction avec une base de données ou d'autres sources de données pour générer dynamiquement du contenu en réponse aux demandes des utilisateurs. Les sites web dynamiques sont souvent construits à l'aide de langages de programmation côté serveur tels que **PHP** et et au Systéme de gestion de bases données relationnel (**SGBDr**) **MySql**.  En résumé, les sites web statiques sont auto-contenus et ne nécessitent pas de traitement côté serveur ou d'interaction avec une base de données, tandis que les sites web dynamiques nécessitent un traitement côté serveur et une interaction avec une base de données ou d'autres sources de données pour générer dynamiquement du contenu en réponse aux demandes des utilisateurs.  J’ai commencé création de ma base de données de création de Model Concepteur de Données (MCD) en utilisant méthode Merise. Pour réaliser ça, on peut utiliser des applications spécialisées - **JMerise** ou **AnalyseSI**. J’ai choisi AnalyseSI.  **AnalyseSI** est un outil logiciel gratuit et open-source pour la conception et la modélisation de systèmes d'information. C'est un outil de modélisation visuelle qui prend en charge une gamme de techniques de modélisation, y compris les diagrammes **Entité-Relation** (ER), les **diagrammes UML**.  **AnalyseSI** permet aux utilisateurs de créer des modèles de systèmes d'information, y compris des bases de données, des processus métier et des applications logicielles. Il fournit une interface graphique pour créer et modifier des modèles, ainsi que des fonctionnalités pour générer de la documentation et du code à partir des modèles. Le logiciel inclut également des outils pour effectuer des vérifications de cohérence, générer des rapports et exporter des modèles dans différents formats.    *Figure 20 : L'interface de AnalyseSI*  Ensuite, quand le modèle est prêt, on peut générer le code SQL lequel on peut importer dans **PhpMyAdmin** pour créer la base des données.    *Figure 21 : Code SQL généré par AnalyseSI*    Ensuite j’ai installé **Wamp** pour accéder à **PhpMyAdmin** ce qui m’a permis de créer une base de données pour faire fonctionner mon code Php.  **WAMP** signifie "Windows, Apache, MySQL et PHP". C'est un ensemble de logiciels open-source populaire utilisé pour le développement web sur les systèmes d'exploitation Windows.    *Figure 22 : Wamp*  **phpMyAdmin** est un outil libre et open-source basé sur le web utilisé pour la gestion et l'administration de bases de données **MySQL** ou **MariaDB**. Il fournit une interface utilisateur graphique pour gérer les bases de données, les tables, les champs, les relations, les index et autres objets de base de données.  Avec **phpMyAdmin**, les utilisateurs peuvent effectuer diverses tâches telles que la création, la modification et la suppression de bases de données, de tables et de champs. Il permet également aux utilisateurs d'importer et d'exporter des données vers et depuis des bases de données, d'exécuter des requêtes SQL, de gérer les utilisateurs et les autorisations, et plus encore.  **phpMyAdmin** est un outil très populaire parmi les développeurs web et les administrateurs de bases de données en raison de sa facilité d'utilisation, de son accessibilité et de ses fonctionnalités complètes. Il est écrit en PHP et peut être accessible via n'importe quel navigateur web avec une connexion Internet.    *Figure 23 : Page serveur local PhpMyAdmin*    *Figure 24 : Création de base de données*  **PHP** (Hypertext Preprocessor) est un langage de script côté serveur open-source populaire utilisé principalement pour le développement web. Il est largement utilisé pour créer des pages web dynamiques, gérer des bases de données et développer des applications web.  Les scripts **PHP** sont exécutés sur le serveur, générant du HTML et d'autres sorties qui sont envoyées au navigateur web du client. PHP prend en charge un large éventail de bases de données, notamment **MySQL**, **PostgreSQL**.  L'un des principaux avantages de **PHP** est sa facilité d'utilisation et son accessibilité pour les débutants. Il est également très polyvalent et peut être utilisé pour une variété de purposes, y compris la création de sites de commerce électronique, des systèmes de gestion de contenu et des forums en ligne. De plus, PHP est soutenu par une communauté de développeurs large et active qui contribuent à son développement continu et fournissent un support aux utilisateurs.    *Figure 25 : Code PHP*    *Figure 26 : La transmission de données en utilisant formulaire* | | | | | |
| J’ai appris à transmettre des données avec les formulaires et les méthodes **GET** : (*les données transiteront par l'URL*) et **POST** : (*les données ne transiteront pas par l'URL*) ainsi que les variables **superglobales** et également les **Sessions**.    *Figure 27: Example du code avec la méthode POST et SESSION*  Et j’ai poursuivi mon apprentissage en apprenant à stocker des informations dans une base de données avec le langage **SQL** en créant des **Tables** et en modifiant leurs contenus.    *Figure 28 : Création d’une table*    *Figure 29 : La modification d’une table*  J’ai aussi appris à lire des données en autorisant l'accès de PHP aux bases de données MySQL via le **PDO** : (*PHP Data Objects*). Ainsi pour les récupérer grâce aux **requêtes**, et les écrire avec la requête **INSERT INTO** qui permet d'ajouter une entrée.    *Figure 30: Connexion à MySQL via PDO et les requêtes*    *Figure 31 : La requête INSERT INTO*  Et puis j’ai continué avec les **Fonctions Scalaires SQL**. Cela créé un "**champ virtuel**" qui n'existe que le temps de la requête, et j’ai poursuivi sur les différents types de dates que peut stocker MySQL et j’ai également travaillé sur les **jointures** entre les tables ce qui permet de pouvoir créer des **relations** entre elles et de les liées.    *Figure 32 : Les fonctions scalaires SQL*    *Figure 33 : Les différents types de dates que peut stocker MySQL*  Pour terminer cette compétence j’ai créé deux bases de données avec PHP MySQL, la thème du projet était : (*la Bibliothèque*). Pour lesquelles je me suis occupé de la création des **diagrammes** avec le **MCD** (*Modèle Conceptuel de Données*) en y ajoutant les **contraintes d'intégrité́** souhaitées dans les exercices tout cela créé via **MySQL Workbench**. Le **MCD** permet d'écrire de façon formelle les données qui seront utilisées par le système d'information. Il s'agit donc d'une représentation des données, facilement compréhensible, permettant de décrire le système d'information à l'aide d'entités.    *Figure 34 : La base de données*    *Figure 35 : Le modèle MCD*    *Figure 36 : Le modéle MLD*  Pour l’exercices demandés dans cette compétence j’ai commencé par créer une base de données qui correspond à la gestion d’une bibliothèque ensuite je me suis servi du **MLD** (*modèle logique de données*) qui est la représentation des données d'un système d'information. Les données sont représentées en prenant en compte le modèle technologique qui sera utilisée pour leur gestion. | | | | | |
|  | | | | |
|  | | | | | |
| **2. Précisez les moyens utilisés :** | | | | | |
| Voici les moyens que j’ai utilisé pour cette cinquième compétence : | | | | | |
| Logiciel: *Visual studio code, Wamp, MySQL, MySQL Workbench.*  Site web: *GitHub desktop.*  Application: *PhpMyAdmin, AnalyseSI.*  Langage: *Html5, Css3, SQL, PHP.* | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | | |
| Pour les Compétences de 1 à 8 j’ai travaillé seul en me documentant sur internet | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | |
|  | | | | | |
| **4. Contexte** | | | | | |
|  | | | |  | |
| Nom de l’entreprise, organisme ou association | | | | *Formation AFPA* | |
| Chantier, atelier, service | | | *Cliquez ici pour taper du texte.* | | |
| Période d’exerciceDu : *05/09/2022*au :*28/04/2023* | | | | | |
|  | | | | |
|  | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Activité-type** | **2** | **Développer la partie back-end d’une application web ou web mobile en intégrant les recommandations de sécurité.** | | | |
| ***Compétence n°6***  | | ***Développer les composants d’accès aux données*** | | | |
|  | | | | | |
|  | | | | | |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | | | | |
| Pour développer des composants d'accès aux données, il faut créer des classes qui interagissent avec la base de données pour récupérer, ajouter, mettre à jour et supprimer des données.  Il faut déterminer le modèle de données approprié pour votre application et concevoir le schéma de la base de données en conséquence. Ensuite, vous pouvez utiliser un système de gestion de base de données (**SGBD**) tel que **MySQL** pour créer la base de données et les tables.  Ensuite il faut écrire du code pour vous connecter à la base de données et exécuter des requêtes SQL pour effectuer des opérations **CRUD** (Create, Read, Update, Delete) sur les données. Vous pouvez utiliser une couche d'abstraction de base de données telle que **PDO**.  **PDO** (PHP Data Objects) est une extension PHP qui fournit une interface unifiée pour accéder à plusieurs types de bases de données. Elle permet aux développeurs de se connecter à diverses bases de données comme MySQL, PostgreSQL, SQLite, Oracle et plus encore. PDO permet également l'utilisation de requêtes préparées pour éviter les injections SQL, ainsi que la récupération de données sous différentes formes telles que les tableaux associatifs et les objets. Elle offre une couche d'abstraction qui permet de travailler avec les différentes bases de données de manière uniforme et cohérente.  Enfin, il faut intégrer vos composants d'accès aux données dans l'architecture de votre application, permettant à d'autres composants tels que l'interface utilisateur et la logique métier d'interagir avec les données stockées dans la base de données.  Pour cette sixième compétence J’ai commencé par réaliser la page d’accueil de mon projet en **PHP**.    *Figure 37 : Le code PHP de la page d’accueil* | | | | | |
| *Figure 38 : La page d’accueil de mon projet en PHP*  J’ai continué sur la réalisation d’un projet avec Le design pattern **MVC** ( *Modèle Views Controller*) administré **POO** ( *Programmation Orientée Objet*).  undefined  *Figure 39 : Le modèle MVC*  Dans le Modèle **MVC** on peut y intégrer le **CRUD** : (*create, read, update, delete*) qui nous permet de pouvoir : Créer – Lire – Mettre à jour et supprimer. Ce sont les quatre opérations de base pour la persistance des données, en particulier le stockage d'informations en base de données.    *Figure 40 : Le CRUD*  Voilà ce qui se passe dans la base de données au moment de l’ajout, la modification, la mise à jour ou la suppression de données grâce au **CRUD** intégré.    *Figure 41 : La base de données au moment de sa modification*  Je termine cette compétence en présentant mon Projet **MVC** personnalisé.  Voici le Dashboard (*Tableau de bord*), C’est un outil d'évaluation constitué de plusieurs indicateurs de performance à des moments donnés ou sur des périodes données.    *Figure 42 : Le Dashboard (tableau de bord)*  En conclusion, je souhaite décrire brièvement ce qui a été fait dans mon projet en dehors de ce qui a été écrit ci-dessus.  J'ai pratiquement reconstruit fonctionnellement mon projet de front-end basé sur React en utilisant PHP. Il y a une page d'accueil qui présente deux catégories qui contiennent tous mes produits, des pages pour ces catégories qui présentent une liste de produits avec leur brève description, des pages pour chaque produit avec une description complète et la possibilité d'ajouter ces produits au panier, une page de panier sur laquelle vous pouvez valider votre choix de produit ou le modifier. J’ai aussi réalisé éspace utilisateur et système de registration, login et logout.    *Figures 43 : Mon projet MVC: Page des plantes divisées par catégorie et page de la plante*    *Figures 44 : Mon projet MVC: page d’* *éspace utilisateur*    *Figures 45 : Mon projet MVC: page de panier*    *Figures 44 : Mon projet MVC: page d’* *éspace utilisateur, page de login et page de registration* | | | | | |
|  | | | | | |
|  | | | | |
|  | | | | | |
| **2. Précisez les moyens utilisés :** | | | | | |
| Voici les moyens que j’ai utilisé pour cette sixième compétence : | | | | | |
| Logiciel: *Visual studio code, Wamp, MySQL.*  Site web: *GitHub desktop.*  Application: *PhpMyAdmin.*  Langage: *Html5, Css3, PHP.* | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | | |
| Pour les Compétences de 1 à 8 j’ai travaillé seul en me documentant sur internet | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | |
|  | | | | | |
| **4. Contexte** | | | | | |
|  | | | |  | |
| Nom de l’entreprise, organisme ou association | | | | *Formation AFPA* | |
| Chantier, atelier, service | | | *Cliquez ici pour taper du texte.* | | |
| Période d’exerciceDu : *05/09/2022*au :*28/04/2023* | | | | | |
|  | | | | |
|  | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Activité-type** | **2** | **Développer la partie back-end d’une application web ou web mobile en intégrant les recommandations de sécurité.** | | | |
| ***Compétence n°7***  | | ***Développer la partie back-end d’une application web ou web mobile*** | | | |
|  | | | | | |
|  | | | | | |
| **1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :** | | | | | |
| Dans le cadre de la compétence "Développer la partie back-end d'une application web ou web mobile" de mon dossier professionnel, j'ai choisi d'utiliser **Symfony 6** pour développer le back-end de mon application web. Cette compétence consiste à créer la logique métier, la gestion des données et l'interaction avec le front-end de l'application. Symfony est un framework PHP robuste et performant qui permet de développer des applications web professionnelles de haute qualité. Pour réussir cette compétence, j'ai dû suivre plusieurs étapes, notamment la conception de l'architecture de l'application, la création des entités et des relations de base de données, la mise en place de la sécurité de l'application, la création de la logique métier et l'implémentation de l'API pour communiquer avec le front-end. | | | | | |
| *Figure 45 : Exigence et installation de Symfony 6*  Pour utiliser Symfony 6, tout d'abord, il faut installer **Composer**, et il est préférable d'installer également **Symfony CLI**.  **Composer** est un gestionnaire de dépendances pour **PHP** qui permet aux développeurs d'installer et de gérer facilement des bibliothèques et des packages tiers dans leurs projets. Il simplifie le processus de gestion des dépendances et garantit que tous les composants requis sont installés et à jour.    *Figure 46 : Installation de composer*  **Symfony CLI** est un outil d'interface de ligne de commande qui simplifie le développement et le déploiement d'applications Symfony. Il fournit aux développeurs divers outils et commandes pour créer, configurer et gérer des projets Symfony, y compris l'exécution de serveurs de développement, la gestion des dépendances et l'exécution de tests. Symfony CLI aide à rationaliser le processus de développement et à améliorer la productivité des développeurs Symfony.    *Figure 47 : Installation de Symfony CLI*  Après avoir vérifié les versions de **Symfony**, **Composer** et **Php**, J’ai poursuivi sur la création du projet **Symfony** en tapant dans le terminal la commande : (*symfony floreil --version="6.2.\*" --webapp*).  Ce qui nous permet d’avoir cette structure illustrée ci-dessous.  Ensuite je lance mon projet **Symfony** en tapant en ligne de commande : **symfony server :start**. | | | | | |
| *Figure 48 : La structure de Symfony obtenue dans Visual studio code*    *Figure 49 : Le premier démarrage du projet Symfony*  Je continue sur la création d’une application web en créant une première page, pour ça, il faut créer un **controller** avec ces **routes**. Ensuite je tape en ligne de commande : (*symfony console make:controller*) pour la création du controller et je relance le serveur.  Dans **Symfony 6**, un **controller** est une classe PHP qui gère les demandes HTTP entrantes et renvoie une réponse HTTP. Il agit en tant que médiateur entre la demande du client et la logique métier de l'application. Le controller reçoit des entrées du client via l'objet de demande et renvoie une sortie via l'objet de réponse. La logique métier est généralement encapsulée dans une classe de service distincte, appelée par le controller. Le controller est responsable de la sélection de la vue appropriée à afficher, en fonction des données de la demande et de la réponse. Il peut également effectuer des tâches telles que l'authentification et l'autorisation, la gestion des erreurs et le caching.  Les **routes** dans Symfony 6 permettent de définir les URL de l'application web et d'associer ces URL à des actions qui seront gérées par les **controllers**. Les controllers contiennent des méthodes qui gèrent les actions correspondantes aux URL définies dans les routes.  Ainsi, lorsqu'un utilisateur accède à une URL définie dans une route, Symfony 6 va chercher la méthode correspondante dans le controller associé à cette route, puis exécuter cette méthode. Le controller peut alors effectuer différentes opérations, telles que l'accès aux données, la génération de vues, etc.  En somme, les **routes** et les **controllers** travaillent ensemble pour fournir des fonctionnalités spécifiques aux utilisateurs de l'application web. Les routes permettent d'identifier les actions demandées par les utilisateurs, tandis que les controllers fournissent l'implémentation de ces actions.    *Figure 50: Le controller et ces routes*    *Figure 51 : Le controller main créé*  Dans la figure 50, on peut voir le namespace, la liste des composants utilisés par ce controller, le nom de la classe de controller - MainController - ainsi que la méthode « index » qui sera exécutée pour la route « / » nommée « app\_main ».    *Figure 52 : La visualisation de controller "main"*  Le résultat de l'exécution de cette méthode est le rendu d'un modèle HTML, qui est ensuite affiché dans le navigateur. Pour la visualisation dans Symfony, on utilise **Twig** comme moteur de template.  **Twig** est un moteur de template open-source écrit en PHP. Il permet de séparer la logique de présentation et de traitement des données dans une application web. Twig est utilisé dans le cadre de projets **Symfony**, Drupal et autres pour générer des vues HTML, XML, JSON ou d'autres formats de sortie. Il est facile à apprendre et à utiliser, grâce à une syntaxe claire et concise, et offre de nombreuses fonctionnalités, telles que l'héritage de templates, les blocs, les filtres, les fonctions et les extensions. En résumé, Twig facilite la création de pages web dynamiques, évolutives et réutilisables.    *Figure 53: fragment du code de la template "header" inclus dans la template "base"*    *Figure 54 : Code de la template "base" de mon application web*  La template de « base » de Twig est un modèle qui sert de structure de base pour toutes les autres templates d'un projet. Elle contient généralement les éléments communs à toutes les pages, tels que la balise HTML, les balises head et body, les feuilles de style et les scripts JavaScript. Elle est souvent utilisée pour inclure d'autres templates, comme le header ou le footer, et peut être étendue ou personnalisée selon les besoins du projet. La template de base permet de maintenir une structure cohérente et de faciliter la maintenance du code en regroupant les éléments communs dans un seul fichier.      *Figure 55 : Utilisation des variables et des toutes dans twig temlate*  Grâce à **Twig**, il est possible d'activer les méthodes des controllers différents en utilisant les noms des routes en tant que liens ou actions des formulaires.  La fonction Path est utilisée pour cela, avec 2 paramètres : 1 obligatoire ( le nom de la route ou la route elle-même ) et 1 optionnel. On utilise cette fonction pour créer un lien de cette manière : {{ path() }} (**fig. 55**).  On peut également transmettre des valeurs de variables entre les controllers et les afficher ou les modifier dans les templates.  L’interpolation **{{ var }}** on utilise les **doubles-accolades** pour afficher le contenu d’une variable.      *Figure 56 : Utilisation des operateurs "if", "else", "for" et de composant*  De plus, grâce à **Twig**, les pages web de l'application deviennent plus dynamiques en utilisant des opérateurs conditionnels et des boucles pour rendre le code plus flexible et plus facilement modifiable (**fig. 56**) en utilisant une structure de contrôle avec la commande : **{% cmd %}** et pour se faire on utilise l’**accolade pourcentage** pour appeler.  On peut utiliser des composants pour les parties répétitives du code afin de rendre le projet plus structuré, plus facile à modifier et plus compréhensible (**fig. 56**).  **Symfony** utilise également **Doctrine**, c’est un **ORM**.  L'**ORM** (Object-Relational Mapping) de Symfony 6 est Doctrine, qui permet de faire le lien entre les objets de l'application et la base de données relationnelle. Doctrine permet de créer des entités qui représentent les tables de la base de données, et d'établir des relations entre ces entités. Les opérations de CRUD (Create, Read, Update, Delete) peuvent être effectuées sur les entités en utilisant les méthodes fournies par Doctrine, sans avoir à écrire des requêtes SQL. Doctrine prend également en charge la gestion des migrations de base de données, ce qui facilite les mises à jour de schéma.  **Doctrine** ajoute des fonctionnalités à **PDO**. (*PHP Data Objects*).    *Figure 57 : L’ORM de Symfony*  **Entity** est une partie importante du système ORM de Symfony.  C’est une classe PHP qui représente une table dans une base de données relationnelle. Elle contient des propriétés qui correspondent aux colonnes de la table, ainsi que des méthodes qui permettent de récupérer et de modifier les données de la table. Les entités sont utilisées avec l'ORM de Symfony pour effectuer des opérations CRUD (Create, Read, Update, Delete) sur la base de données.    *Figure 58 : Fragment du code d'Entity "User"*  On peut créé l’Entity en effectuant la commande console (« symfony console make:entity ») ou manuellement. Chaque Entity doit être accompagnée par un **Repository**.  Le **Repository** est une classe qui permet d'interagir avec la base de données pour récupérer des entités (objets) de manière optimisée. Il fournit des méthodes pour effectuer des requêtes complexes sur la base de données, ainsi que des fonctions de pagination et de tri. Le Repository peut également être utilisé pour écrire des requêtes personnalisées pour récupérer des données spécifiques. Chaque Entity doit être accompagnée d'un Repository correspondant, qui est créé automatiquement lors de la génération de l'Entity avec la console Symfony.    *Figure 58 : Fragment du code de Repository "UserRepository"*  Je poursuis avec les **migrations** dans **Symfony** qui permettent de faire passer une base de données d’un état A à un état B.  Les **migrations** dans Symfony 6 sont une fonctionnalité de Doctrine ORM qui permettent de versionner la structure de la base de données d'une application web. Les migrations sont créées automatiquement (par les commandes de console « **symfony console make:migration** » et après « **symfony console doctrine:migrations:migrate** ») lorsqu'un développeur définit une nouvelle entité ou modifie une entité existante dans l'application, et elles sont utilisées pour mettre à jour la structure de la base de données lorsqu'une nouvelle version de l'application est déployée.  Les migrations sont enregistrées dans des fichiers PHP qui décrivent les changements apportés à la structure de la base de données, tels que la création ou la suppression de tables, la modification de colonnes ou la création de nouvelles relations. Les fichiers de migration peuvent être exécutés en utilisant la console Symfony pour mettre à jour la base de données avec les derniers changements.  Les migrations permettent aux développeurs de travailler de manière collaborative sur l'application sans avoir à se soucier de la gestion de la base de données. Les changements apportés à la structure de la base de données peuvent être facilement suivis et gérés à l'aide de fichiers de migration, et les erreurs de migration peuvent être détectées et corrigées rapidement.    *Figure 59 : Les migrations*  Pour relier ma base de donnée à mon projet **Symfony** avec **l’ORM Doctrine**, j’ouvre le **fichier .env** comme dans l’exemple ci-dessous et je modifie la ligne 31 en y ajoutant le nom de ma base de donnée et en vérifiant le port du serveur MySQL qui est le **3306**.  Il y a plusieurs options pour définir où l'on peut travailler avec une base de données. Par exemple, sur la **fig. 60**, je peux choisir une instance de ma base de données sur Google Cloud Platform ou une instance de ma base de données locale. On peut également utiliser d'autres types de bases de données tels que PostgreSQL ou MariaDB.    *Figure 60 : Modification du fichier .env*  On peut travailler avec une base de données existante ou la créer préalablement à l'aide de la commande console (« symfony console doctrine:database:create »).  Après avoir défini ou créé la base de données, créé les entités et les repositories, géré et affiché les données de la base de données avec des contrôleurs et des templates Twig, on peut tester mon projet avec des fausses données en utilisant des **Fixtures**.  Les **fixtures** dans Symfony 6 sont des données de test qui peuvent être préchargées dans la base de données pour les besoins de développement et de démonstration. Elles peuvent être utilisées pour remplir la base de données avec des données de test cohérentes et reproductibles, ce qui est particulièrement utile pour le développement et le débogage.    *Figure 61 : Les fixtures*  Les fixtures sont souvent utilisées en conjonction avec le bundle **DoctrineFixturesBundle**, qui fournit des outils pour charger facilement des données de test dans la base de données. Les fixtures peuvent être écrites en utilisant une variété de formats de données, tels que YAML, XML ou PHP, et sont généralement organisées en ensembles logiques appelés groupes de fixtures.  Les fixtures sont un élément clé du processus de développement et sont souvent utilisées pour tester le comportement des applications dans des scénarios spécifiques, tels que des tests d'intégration ou des tests de performance. En utilisant des fixtures pour générer des données de test cohérentes et reproductibles, les développeurs peuvent s'assurer que leur application fonctionne correctement et est capable de gérer des charges de travail élevées dans des environnements de production réels.  Pour mon projet j'utilise une classe **AppFixtures** pour générer toutes les données fausses nécessaires, mais on peut créer et utiliser plusieurs classes de Fixtures pour rendre le code plus facilement modifiable si on a un grand projet.    *Figure 62 : Fragment du code de AppFixtures*  Dans mon projet Symfony 6, j'ai utilisé toutes ces fonctionnalités pour créer une application web fonctionnelle.  J'ai défini où se trouverait la base de données de mon projet. J'ai décidé de placer ma base de données localement, mais j'ai également créé une instance de base de données dans le Google Cloud Platform au cas où je devrais transférer mon projet dans le cloud.    *Figure 63 : Instance de base de données sur Google Cloud Platform*  J’ai défini la structure de cette base de données. On a six tableaux : User, Cart, Product, Category, Order and OrderDetail.    *Figure 64 : EER modèle créé par MySQL Workbench*  Ensuite, en fonction du modèle, j'ai créé les entités pour chacune des tables de ma base de données.    *Figure 65 : Fragments de code des Entités*  Ensuite, en m'appuyant sur la conception de mon application, élaborée au début de mon apprentissage, j'ai défini la fonctionnalité des différentes pages de mon application et l'ai concrétisée à l'aide de controllers et de twig templates. J'ai utilisé des bundles supplémentaires pour la mise en œuvre de certaines fonctions de mon application (telles que l'inscription avec vérification de l'e-mail ou l'intégration du système de paiement Stripe).    *Figure 66 : Fragment de code de CheckoutController qui utilise stripe/stripe-php bundle*  Enfin, j'ai utilisé des fixtures pour tester et corriger les erreurs de la logique de l'application.      *Figure 67 : Fragments de code des Controllers* | | | | | |
|  | | | | |
|  | | | | | |
| **2. Précisez les moyens utilisés :** | | | | | |
| Voici les moyens que j’ai utilisé pour cette septième compétence : | | | | | |
| Documentation: *Site de Symfony.*  Logiciel: *Visual studio code, Wamp, MySQL, Composer.*  Site web: *Github desktop.*  Framwork: *Symfony6.*  Application: *PhpMyAdmin.*  Langage: *HTML5, CSS3, PHP, SQL, Twig.* | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |
| **3. Avec qui avez-vous travaillé ?** | | | | | |
| Pour les Compétences de 1 à 8 j’ai travaillé seul en me documentant sur internet | | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | |
|  | | | | | |
| **4. Contexte** | | | | | |
|  | | | |  | |
| Nom de l’entreprise, organisme ou association | | | | *Formation AFPA* | |
| Chantier, atelier, service | | | *Cliquez ici pour taper du texte.* | | |
| Période d’exerciceDu : *05/09/2022*au :*28/04/2023* | | | | | |
|  | | | | |
|  | | | | |
| **5. Informations complémentaires** *(facultatif)* | | | | |
|  | | | | | |
|  | | | | | |
|  | | | | | |

|  |  |  |
| --- | --- | --- |
| **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. |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |
| --- |
| **Déclaration sur l’honneur** |
|  |
|  |

Konstantin Shilkov

Je soussigné(e) [prénom et nom] ,

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

Créteil

28/02/2023

Fait à le

Pour faire valoir ce que de droit.

Signature :

|  |
| --- |
| **Documents illustrant la pratique professionnelle** |
|  |
| *(facultatif)* |
| **Intitulé** |
| Cliquez ici pour taper du texte. |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

|  |
| --- |
| **Annexes** |
|  |
| *(Si le RC le prévoit)* |