



# DOSSIER PROFESSIONNEL (DP)

*Nom de naissance* ▶ SENG  
*Nom d'usage* ▶ SENG  
*Prénom* ▶ LAURE  
*Adresse* ▶ 13 RUE DU PERTHUS  
31880 LA SALVETAT SAINT GILLES

## 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>**

# DOSSIER PROFESSIONNEL (DP)

## Sommaire

### Exemples de pratique professionnelle

#### **Activité-type 1 : Développer la partie front-end d'une application web ou web mobile sécurisée.** **p**

- ▶ CP 1 Installer et configurer son environnement de travail en fonction du projet web ou web mobile **p**
- ▶ CP 2 Maquetter des interfaces utilisateur web ou web mobile **p**
- ▶ CP 3 Réaliser des interfaces utilisateur statiques web ou web mobile **p**
- ▶ CP 4 Développer la partie dynamique des interfaces utilisateur web ou web mobile **p**

#### **Intitulé de l'activité-type n° 2 : Développer la partie back-end d'une application web ou web mobile sécurisée.** **p**

- ▶ CP 5 Mettre en place une base de données relationnelle **p**
- ▶ CP 6 Développer des composants d'accès aux données SQL et NoSQL **p**
- ▶ CP 7 Développer des composants métier coté serveur **p**
- ▶ CP 8 Documenter le déploiement d'une application dynamique web ou web mobile **p**

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

#### **Déclaration sur l'honneur** **p**

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

#### **Annexes** **p**

# **EXEMPLES DE PRATIQUE PROFESSIONNELLE**

# DOSSIER PROFESSIONNEL (DP)

## Activité-type 1

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

**CP 1** ▶ *Installer et configurer son environnement de travail en fonction du projet web ou web mobile*

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

Pour installer un environnement de travail en vue de réaliser une application web statique en HTML, CSS et Javascript, il faut passer par plusieurs étapes.

Tout d'abord, j'ai opté pour le **système d'exploitation Linux** et ai installé la distribution Mint d'après une image ISO téléchargée sur le site officiel.

Une fois l'OS installé, il faut avoir les logiciels de base nécessaires au développement. Pour écrire du code pour la partie front-end, sans framework, il suffit d'avoir un éditeur de texte mais un **IDE** (Integrated Development Environment) tel que Vscode permet d'avoir nombre de fonctionnalités qui vont faciliter le développement en évitant les erreurs et en augmentant la productivité.

En effet, **VsCode** présente des fonctionnalités par défaut très puissantes telles que **Intellisense** qui va gérer l'autocomplétion, la coloration syntaxique et l'affichage d'informations lors de la saisie du code selon les langages choisis, **Emmet** qui va permettre de saisir du HTML plus rapidement, un **Terminal** intégré et bien plus de fonctionnalités telles que les multi-curseurs, les snippets. Une multitude d'extensions peuvent être installées selon les besoins du développeur. Pour le Javascript, on peut utiliser des linters qui vont vérifier la qualité du code tels que **ESLint** ou des formateurs de code tels que **Prettier**. Des outils de prévisualisation comme le **Live Server** et le Live Preview sont très utiles aussi.

Il est indispensable d'avoir un **navigateur** internet pour afficher les pages web et aussi avoir accès aux **Dev Tools**. Il y en a en général installé par défaut sur tous les systèmes mais Chrome et Firefox restent les plus courants et offrent des Dev Tools

puissants. Des extensions mettant à disposition des outils pratiques peuvent être utilisées telles que les extensions **ColorZilla** ou **Simple Ruler**.

Une fois ces outils installés, on peut coder un projet mais pour pouvoir sauvegarder son évolution, l'idéal est d'avoir recours à un **gestionnaire de versions**, le plus répandu étant **Git**, et d'envoyer son code sur une plateforme d'hébergement de code source telle que **GitHub** ou GitLab, sur laquelle on a au préalable créé un compte. **La connexion à ce compte doit se faire via un token ou via une clé SSH**. Ces outils sont, en plus d'être des outils de sauvegarde du versioning du code, des outils puissants pour le travail en collaboration via le système des branches, des Pull Request, des Forks.

A propos de collaboration, il est nécessaire pour pouvoir travailler en pair-programming, d'avoir quelques outils installés :

- l'extension **LiveShare** de VsCode
- un **logiciel de communication** en ligne tels que Slack, Discord, pour communiquer en audio, visio ou partager son écran.
- des outils tels que **ngrok** peuvent permettre de partager son site diffusé en localhost à d'autres personnes.

## 2. Précisez les moyens utilisés :

L'installation a été réalisée sur mon ordinateur portable connecté à un accès internet.

## 3. Avec qui avez-vous travaillé ?

Cette installation a été réalisée durant la formation.

## 4. Contexte

Nom de l'entreprise, organisme ou association ▶

Ecole O'Clock

Chantier, atelier, service ▶

*Exercice réalisé en cours de formation*

Période d'exercice ▶

Du : 30/10/2023 au : 22/04/2024

## 5. Informations complémentaires (facultatif)

# DOSSIER PROFESSIONNEL (DP)

## Activité-type 1

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

### CP 2 ▶ Maquetter des interfaces utilisateur web ou web mobile

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

Nous allons parler de la **conception** du site Miamiam. Le site est un site de recettes ayant une page d'accueil avec un menu qui renvoie vers les pages des différents catégories de recettes.

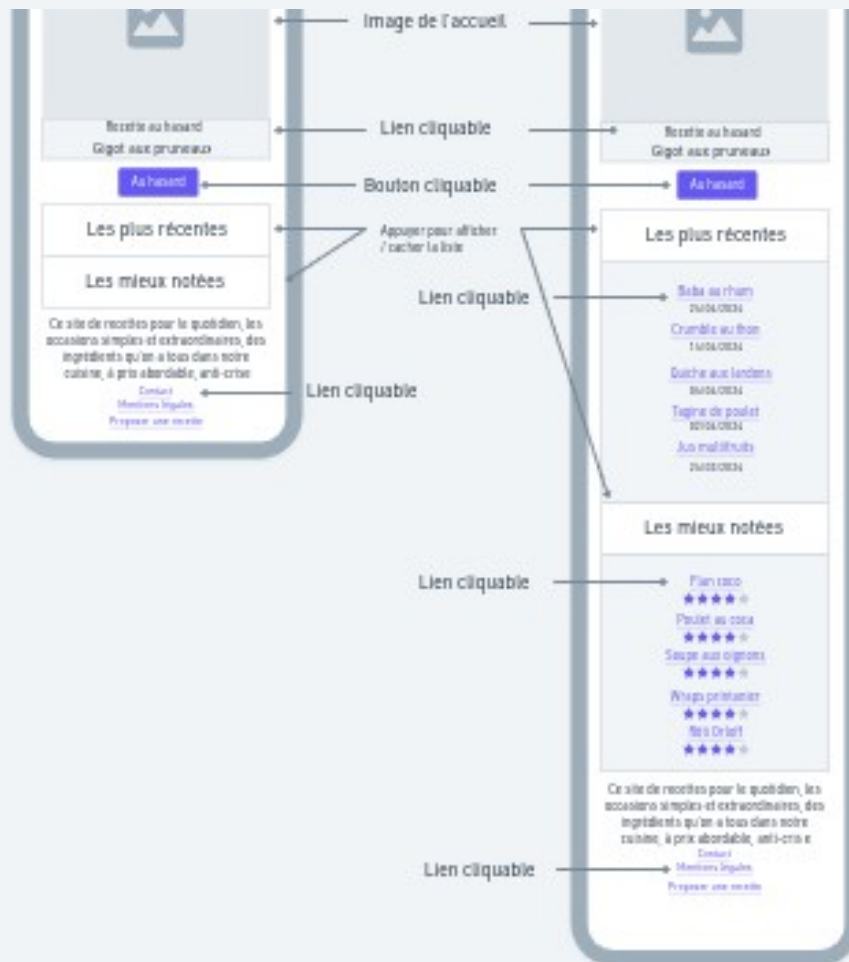
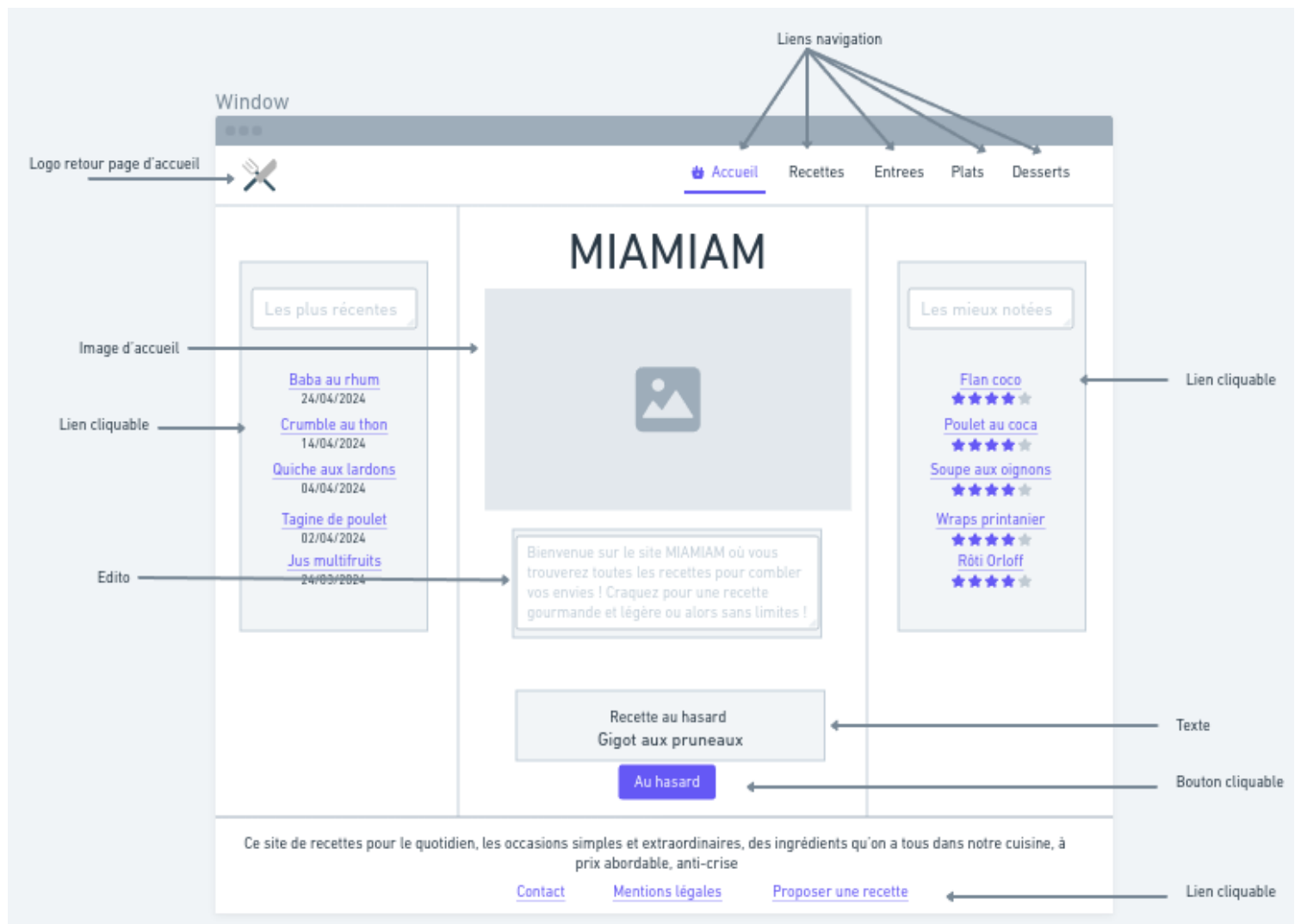
On nous a demandé de fournir les documents liés à la conception de ce site.

Voici quelques **user stories** qui ont été rédigées:

En tant que	Je veux	Afin de
Utilisateur	Afficher les recettes d'une catégorie	Choisir la recette à faire
Utilisateur	Afficher une recette au hasard	M'inspirer
Utilisateur	Afficher la liste des ingrédients d'une recette	Les rassembler dans ma cuisine
Administrateur	Modifier le texte de l'édition	Parler de thèmes différents selon la période de l'année

Nous avons ensuite réalisé les wireframes, schémas représentant les pages avec les différents éléments et leurs emplacements, ainsi que des légendes pour expliquer en détail le **zoning**.

Nous avons réalisé ces wireframes pour trois tailles d'écran : ordinateur de bureau, tablette et téléphone portable.







MINISTÈRE CHARGÉ  
DE L'EMPLOI

# DOSSIER PROFESSIONNEL (DP)

## 2. Précisez les moyens utilisés :

Le logiciel en ligne Whimsical

### 3. Avec qui avez-vous travaillé ?

### 4. Contexte

Nom de l'entreprise, organisme ou association ▶ Ecole O'clock

Chantier, atelier, service ▶ *Exercice réalisé en cours de formation*

Période d'exercice ▶ Du : 30/10/2023 au : 22/04/2024

### 5. Informations complémentaires (facultatif)

# DOSSIER PROFESSIONNEL (DP)

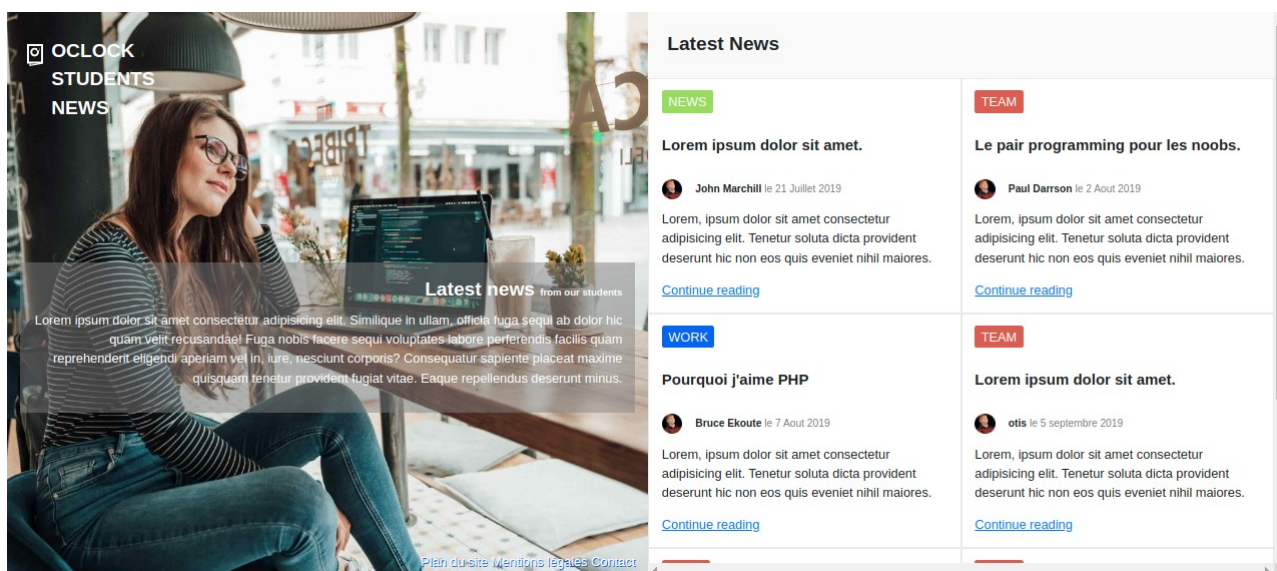
## Activité-type 1

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

**CP 3** ▶ *Réaliser des interfaces utilisateur statiques web ou web mobile*

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

Lors d'un atelier en pair programming, nous avons réalisé un site statique à partir d'une maquette fournie.



J'ai d'abord créé la partie **HTML** en remplissant le **<head>** avec les **méta-données** requises (balises **<meta>**) : encodage des caractères, configuration du viewport, description et titre de la page (utiles pour le référencement naturel).

Puis j'ai précisé les liens vers les **fichiers de style** (balises **<link>**) : un fichier **reboot.css** pour normaliser le style de la page pour qu'il soit cohérent d'un navigateur à l'autre, et un fichier **style.css** qui contient le css spécifique à la page.

```

1  <!DOCTYPE html>
2  <html lang="fr">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <meta name="description" content="Onews, blog de l'école Oclock">
7      <title>Onews</title>
8      <link rel="stylesheet" href="../css/reboot.css">
9      <link rel="stylesheet" href="../css/style.css">
10 </head>
11 <body>
12 <main>
13
14 <section id="left-side">
15     <h1>  OCLOCK STUDENTS NEWS</h1>
16     <article>
17         <h2> Latest news <span> from our students </span></h2>
18         <p>
19             Lorem ipsum dolor sit amet consectetur adipisicing elit.
20             Similique in ullam, officia fuga sequi ab dolor hic quam velit recusandae!
21             Fuga nobis facere sequi voluptates labore perferendis
22             facilis quam reprehenderit eligendi aperiam vel in, iure,
23             nesciunt corporis? Consequatur sapiente placeat maxime quisquam tenetur provident fugiat vitae. Eaque repellendus deserunt minus.
24         </p>
25     </article>
26     <nav>
27         <span> Plan du site</span>
28         <span> Mentions légales</span>
29         <span> Contact</span>
30     </nav>
31 </section>

```

Puis j'ai construit la **structure en HTML** en utilisant les balises adaptées. Généralement, j'ai utilisé des **balises sémantiques** pour favoriser le **SEO** (Search Engine Optimization : optimisation du site pour les moteurs de recherche) et, quand cela était nécessaire, quelques balises <div> qui avaient un rôle plutôt structurel, pour faciliter par la suite la mise en place du CSS. Vous trouverez un extrait de l'HTML créé à la fin de cette partie.

Le CSS a été effectué par mon collègue avec lequel je collaborais via les fonctionnalités LiveShare de VScode.

Les **sélecteurs CSS** utilisés correspondent aux balises sémantiques ou à des **identifiants prédéfinis** dans les balises html grâce aux attributs « **id** » ou « **class** ». Nous avons aussi utilisé des **pseudo-éléments** pour cibler précisément des sous-parties d'éléments html ainsi que des **pseudo-classes** pour cibler certains états ( :hover par exemple).

Les propriétés CSS choisies ont concerné la taille des éléments (width,height), leur couleur (color), leur disposition(display), la police(font-size par exemple).

# DOSSIER PROFESSIONNEL (DP)

```
1  main {
2    display: flex;
3    flex-direction: row;
4    height: 100dvh;
5  }
6
7  section {
8    width: 50dvw;
9  }
10
11 /*partie de gauche*/
12
13 #left-side{
14   padding: 2em 1em 0.5em 1em;
15   display : flex;
16   flex-direction: column;
17   justify-content: space-between;
18   background-image: url(../images/nicole.jpg);
19   background-repeat: no-repeat;
20   background-size: cover;
21   background-position: center ;
22   color : white ;
23 }
24
25 #left-side img {
26   margin: 0.4em;
27   width: 20px;
28   height: auto;
29 }
30
31 #left-side h1 {
32   display: flex;
33   text-align: left;
34   align-items: flex-start;
35   width: 25%;
36   margin: 0;
37   padding: 0;
38   align-self: start;
39 }
40
41 #left-side article {
42   background-color: rgba(131, 130, 130, 0.493);
43   align-self : center;
44   text-align: end;
45   justify-self :center ;
46   padding: 1em;
47 }
48
49 #left-side article h2 span {
50   font-size: 0.5em;
51   font-size: italic;
52 }
53
54 #left-side nav {
55   align-self : end;
56 }
57
58 #left-side nav span {
59   text-shadow: 1px 1px #0766F0;
60 }
61
62 #left-side nav span:hover{
63   background-color: #0766F0;
64 }
```

La maquette fournie représente le site tel qu'il devrait apparaître sur des écrans larges, en **version desktop**. Nous avons réalisé ce projet pour cette version là, bien que la bonne pratique serait de coder en « **mobile first** », c'est à dire pour les petits écrans, et d'ensuite rajouter des éléments sur les écrans plus larges. Pour que le site soit **responsive**, il aurait fallu créer des versions différentes suivant la taille des supports en utilisant les **Media Queries** qui conditionnent le style en fonction de la taille (**breakpoints**) et du type de support. Voici une utilisation possible des Media Queries.

```
1  @media(max-width:767px) {
2      main {
3          flex-direction: column;
4          height: auto;
5      }
6
7      section {
8          width: 100dvw;
9      }
10 }
11
```

## 2. Précisez les moyens utilisés :

Le code a été réalisé sous VScode. L'extension Emmet a été installée pour permettre une complétion du HTML et du CSS. Le LiveShare a permis de partager le code en temps réel.



# DOSSIER PROFESSIONNEL (DP)

## 3. Avec qui avez-vous travaillé ?

Cet atelier a été réalisé en binôme avec un collègue de formation.

## 4. Contexte

Nom de l'entreprise, organisme ou association ▶ Ecole O'Clock

Chantier, atelier, service ▶ Exercice réalisé en cours de formation

Période d'exercice ▶ Du : 30/10/2023 au : 22/04/2024

## 5. Informations complémentaires (facultatif)

## Activité-type

1

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

**CP 4** ▶ *Développer la partie dynamique des interfaces utilisateur web ou web mobile*

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

Lors de cet atelier, nous avons créé un projet d'un site proposant des séjours touristiques.



Plusieurs fonctionnalités étaient attendues : navigation, slider d'images, mise en favoris, choix d'une couleur personnalisée pour les boutons du slider, filtre par note.



# DOSSIER PROFESSIONNEL (DP)

Afin de mettre en place des **interactions** entre l'utilisateur et l'interface, le JS (JavaScript) a été utilisé. L'organisation du JS a suivi une **structure modulaire** avec un fichier principal App.js dans lequel nous avons importé des fichiers concernant chacune des fonctionnalités de l'UI (User Interface).

```
1 import { destination } from "../modules/destination.js"
2 import { newsletter } from "../modules/newsletter.js";
3 import { reviewsFilter } from "../modules/reviewsFilter.js";
4 import { slider } from "../modules/slider.js";
5 import { theme } from "../modules/theme.js";
6
7 const app = {
8
9   run : function(){
10     destination.init();
11     newsletter.init();
12     reviewsFilter.init();
13     slider.init();
14     theme.init();
15   }
16 }
17
18 document.addEventListener("DOMContentLoaded", app.run);
```

En effet, le Javascript est le langage qui permet d'**interagir avec le DOM** (Document Object Model), c'est à dire l'arborescence des éléments HTML du document.

On va pouvoir **cibler des éléments du DOM** grâce à des fonctions telles que `querySelector()`, `getElementById()` et y **associer des Event Listeners**, écouteurs d'évènements qui vont réagir à des actions très variées ( souris, clavier, navigateur...).

L'association des Event Listeners se fait grâce à la fonction **AddEventListener** qui utilise des **callbacks**, c'est à dire des fonctions qui se lanceront au déclenchement de l'évènement. On appelle ces callbacks, dans ce cas d'écoute d'évènements, des **handlers**.

```

1  const colors = {
2    greenButtonElement : null ,
3    redButtonElement : null ,
4    blueButtonElement : null ,
5    changeColor : function(color) {
6      theme.bodyElement.classList.toggle(color);
7    } ,
8    init : function () {
9      colors.greenButtonElement = document.querySelector('#theme-green');
10     colors.redButtonElement = document.querySelector('#theme-red');
11     colors.blueButtonElement = document.querySelector('#theme-blue');
12
13     colors.greenButtonElement.addEventListener("click", colors.changeColorGreen) ;
14     colors.redButtonElement.addEventListener("click", colors.changeColorRed) ;
15     colors.blueButtonElement.addEventListener("click", colors.changeColorBlue);
16   },
17   changeColorGreen : () =>{toggle.toggle("theme-green")},
18   changeColorRed : () =>{toggle.toggle("theme-red")},
19   changeColorBlue :() =>{toggle.toggle("theme-blue")},
20
21 }
22 document.addEventListener("DOMContentLoaded", colors.init);
23 ;

```

Ci-dessus, le module colors.js permet de changer la couleur du thème du site en cliquant sur les boutons respectifs. En effet, Javascript dispose aussi de fonctions qui permettent d'**interagir avec le CSS**, soit en modifiant directement des propriétés css via la propriété style de l'élément, soit en modifiant les classes liées aux éléments avec la propriété classlist et ses fonctions add(),remove(), toggle().

## 2. Précisez les moyens utilisés :

Ce projet a été réalisé avec VS Code.

## 3. Avec qui avez-vous travaillé ?

Il a été réalisé dans le cadre de la formation.



# DOSSIER PROFESSIONNEL (DP)

## 4. Contexte

Nom de l'entreprise, organisme ou association ▶ Ecole O'clock

Chantier, atelier, service ▶ Exercice réalisé en cours de formation

Période d'exercice ▶ Du : 30/10/2023 au : 22/04/2024

## 5. Informations complémentaires (facultatif)

## Activité-type

2

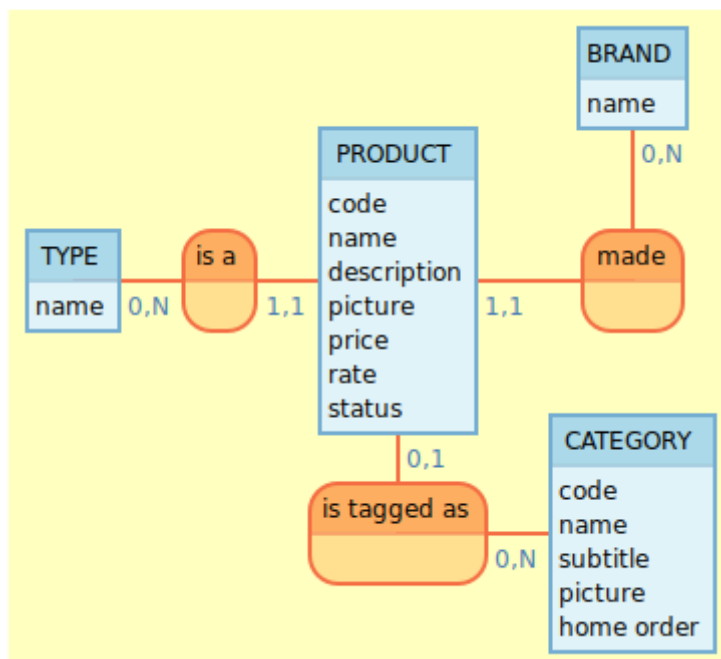
Développer la partie back-end d'une application web ou web mobile sécurisée.

### CP 5 ► Mettre en place une base de données relationnelle

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

Nous allons parler dans cette activité de la mise en place d'une base de données relationnelle dans le cadre d'un projet d'un site de vente de chaussures en ligne.

Des documents étaient fournis pour préciser la structure que devait suivre la mise en place de la base de données. Ils décrivaient les besoins du site et une structure du catalogue. Nous avons à partir de ces écrits pu définir un schéma représentant les liens entre les articles : le MCD.



Le **MCD** (Modèle Conceptuel de Données) permet de préciser les différentes **entités** qui rentrent en jeu, avec leurs **propriétés/attributs**, et les **relations** qui les lient. Il y a 4 **cardinalités** à déterminer pour chaque entité. Elles correspondent à combien de fois au minimum et au maximum une occurrence d'une entité peut être liée à une occurrence de l'autre entité.

Une fois que le MCD est conçu, on peut le modéliser sous forme de **MLD** (Modèle Logique de Données), modèle qui va être une représentation des tables qu'on voudra enregistrer en base de données. Pour chaque entité, on aura une table et des colonnes

# DOSSIER PROFESSIONNEL (DP)

qui correspondront à ses propriétés. On déterminera quelle est la propriété qui servira de discriminant pour celle-ci, en général, on va créer un identifiant unique qui servira de **clé primaire**. On aplanit les relations en créant des **clés étrangères** ou on les modélise dans une table si nécessaire (table pivot). Le MLD correspondant à nos données est donc :

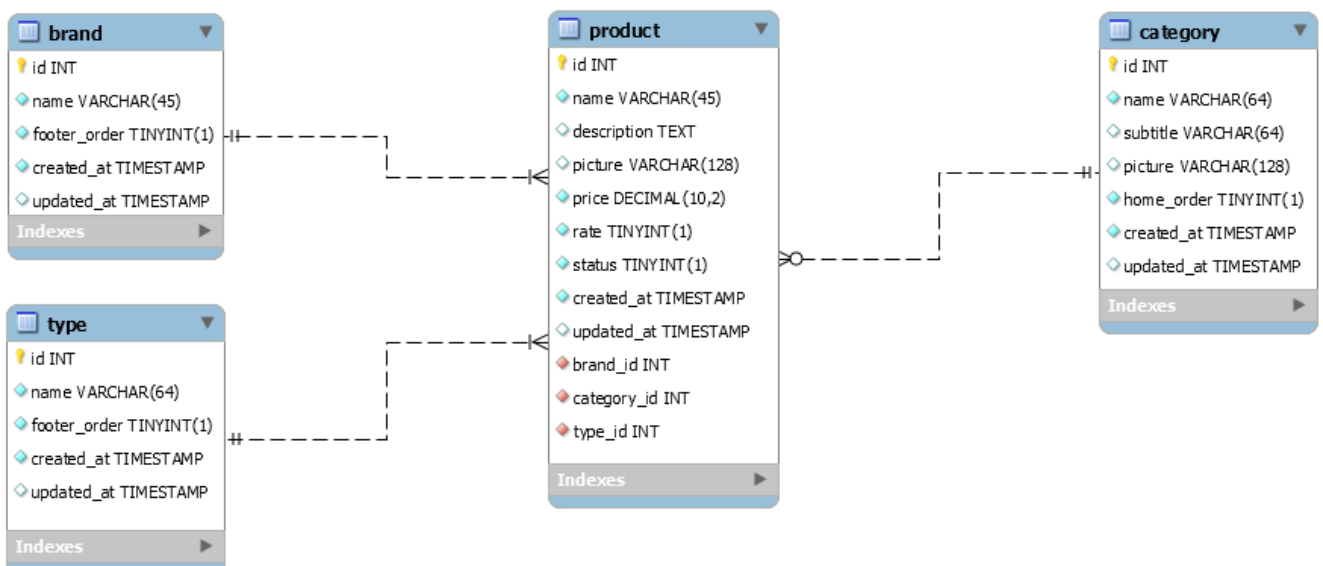
PRODUCT : product\_id, name, description, picture, price, rate, status, brand\_id, category\_id, type\_id

TYPE : type\_id, name, footer\_order

BRAND : brand\_id, name, footer\_order

CATEGORY : category\_id, name, subtitle, picture, home\_order

La mise en place du **MPD** (Modèle Physique de Données) finalise le travail de préparation pour l'implémentation dans le **SGBDR** (Système de Gestion de Base de Données Relationnelle). Pour chaque colonne de chaque table, il est important de choisir **quel type de variable** sera enregistrée et aussi **quelle taille** celle-ci aura, cela ayant des incidences sur les performances du site. Un identifiant(id) est de rigueur pour pouvoir identifier, dans une table, un enregistrement d'un autre.



Une fois ce schéma réalisé, nous avons créé une base de données utilisant le SGBDR(Système de Gestion de Bases de Données Relationnelles) MariaDB. Après avoir créé la base de données et un utilisateur, nous avons créé les tables et leurs colonnes en choisissant leur type, si elles pouvaient être nulles, si elles étaient en relation avec d'autres tables.

## 2. Précisez les moyens utilisés :

Le MCD a été réalisé avec **MOCODO**, outil de création de MCD à partir d'un langage simple.

L'outil en interface graphique **Adminer** a permis de créer la base de données et de l'administrer.

## 3. Avec qui avez-vous travaillé ?

Ce projet a été réalisé dans le cadre de la formation.

## 4. Contexte

Nom de l'entreprise, organisme ou association ▶ Ecole O'clock

Chantier, atelier, service ▶ Exercice réalisé en cours de formation

Période d'exercice ▶ Du : *30/10/2023* au : *22/04/2024*

## 5. Informations complémentaires (facultatif)

# DOSSIER PROFESSIONNEL (DP)

## Activité-type 2

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

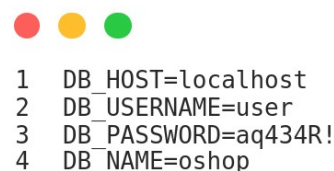
**CP 6** - *Développer des composants d'accès aux données SQL et NoSQL*

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

Nous allons nous appuyer sur le projet dont je vous ai parlé précédemment (compétence 5) pour illustrer comment nous avons interagi avec la base de données.

Une fois la base de données créée, il faut à présent pouvoir accéder aux données qu'elle contient à partir de notre site. Notre site étant codé en PHP, nous avons donc utilisé **PDO** (PHP Data Objects) qui est l'interface que fournit PHP pour interagir avec les bases de données via des **requêtes SQL**. PDO fournit de nombreuses méthodes qui vont permettre d'exécuter des requêtes(`execute()`,`query()`) , de récupérer des données (`fetch()`, `fetchAll()`), d

Avant de pouvoir mettre en place les requêtes, il faut configurer l'accès à la base de données. Nous avons utilisé un fichier **Database.php** qui permet de créer une classe qui va gérer l'accès à la base de données en utilisant PDO. Elle fait appel à des variables qui seront définies dans un fichier **config.ini** qui contient le nom de l'hôte, le nom de la base de données, l'identifiant et le mot de passe de l'utilisateur. Ces données étant sensibles, il faut veiller à ne pas les transmettre, par exemple dans un repo Github. La pratique est de mettre un fichier d'exemple en ligne que l'utilisateur remplira sur sa machine.



```
1 DB_HOST=localhost
2 DB_USERNAME=user
3 DB_PASSWORD=aq434R!
4 DB_NAME=oshop
```

A présent, pour récupérer des données de la base de données, nous n'avons plus qu'à importer les classes PDO et Database dans nos Models pour lancer des requêtes SQL. Voici un exemple de requête SQL tiré du modèle Category.php.

```

1 public function findHomeCategories()
2 {
3     // Requete SQL pour rechercher toutes les categories dans la BDD
4     $sql = 'SELECT categories.*
5           FROM categories
6           WHERE home_order > 0
7           ORDER BY home_order ASC
8           LIMIT 5;';
9
10    return $this->fetchMany($sql);
11 }
12

```

La méthode ci-dessus va faire appel à une autre méthode fetchMany() qui s'occupe de lancer la requête en utilisant PDO et la base de données :

```

1 protected function fetchMany($sqlRequest)
2 {
3     // Connexion à la BDD
4     $pdo = Database::getPDO();
5
6     // Exécuter la requete SELECT
7     $pdoStatement = $pdo->query($sqlRequest);
8
9     // static::class retourne le FQCN de la classe appelante
10    // cad une des classe fille qui utilise cette fonction...
11    $results = $pdoStatement->fetchAll(PDO::FETCH_CLASS, static::class);
12
13    // retourner les résultats
14    return $results;
15 }

```

## 2. Précisez les moyens utilisés :

Ce projet a utilisé VScode et PHP 8.1.

## 3. Avec qui avez-vous travaillé ?





# DOSSIER PROFESSIONNEL (DP)

## 4. Contexte

Nom de l'entreprise, organisme ou association ▶ Ecole O'clock

Chantier, atelier, service ▶ Exercice réalisé en cours de formation

Période d'exercice ▶ Du : 30/10/2023 au : 22/04/2024

## 5. Informations complémentaires (facultatif)

## Activité-type

2

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

### CP 7 ▶ Développer des composants métier coté serveur

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

Dans cette partie, nous allons continuer à nous appuyer sur le projet du site Oshoes pour illustrer la compétence 7.

Tout d'abord nous allons expliquer plus en détail le **design pattern** que nous avons adopté pour **l'architecture du projet** : le **MVC**. Il consiste à séparer le projet en **trois parties** ou composants dont les rôles sont distincts : le **Model**, qui va concerner la gestion des données, la **View**, qui va concerner l'affichage de celles-ci et le **Controller**, qui va faire le lien entre le Model et la View, en transmettant les informations du Model à la View et de la View au Model.

Cette architecture va nous permettre de bien organiser le code et structurer nos méthodes. Ainsi, les **Models** contiendront les **propriétés** des entités du projet et leurs **getters** et **setters**, les **méthodes spécifiques** qui serviront à récupérer les données. Les **méthodes qui serviront à afficher les données** seront dans les Controllers.

Dans notre projet, nous avons créé un CoreModel qui a servi de **classe parente** commune aux autres classes enfants. Cela a permis d'avoir des **propriétés communes** à toutes ces classes comme les user stories le demandaient et d'éviter de nous répéter. Les propriétés dont l'accès est **protected** ou **public** seront transmises aux classes enfants, contrairement à une propriété dont l'accès est **private**.

```
1 class CoreModel
2 {
3     protected $id;
4     protected $name;
5     protected $created_at;
6     protected $updated_at;
```

```
1 class Category extends CoreModel
2 {
3
4     private $subtitle;
5     private $picture;
6     private $home_order;
```

En **étendant** la classe parente CoreModel, la classe Category **héritera** des propriétés « id », «name», «created\_at» et«updated\_at», auxquelles on aura juste à rajouter les propriétés propres à sa classe : «subtitle», «picture», et«home\_order»,

# DOSSIER PROFESSIONNEL (DP)

En parlant de la propriété « home\_order », elle a été implémentée pour pouvoir permettre de dynamiser les catégories affichées sur la page d'accueil. Grâce à une requête SQL, on ne récupérera que les catégories ayant une valeur positive à cette propriété et on les classera par ordre croissant, de manière à n'avoir que les 5 premières :

```
1  /**
2  * Recherche les 5 catégories à mettre en avant sur la home page
3  * (filtrer avec le home_order)
4  *
5  * @return array liste des 5 catégories de la home page
6  */
7  public function findHomeCategories()
8  {
9      $sql = 'SELECT categories.*
10      FROM categories
11      WHERE home_order > 0
12      ORDER BY home_order ASC
13      LIMIT 5;';
14
15      return $this->fetchMany($sql);
16  }
```

C'est aussi grâce à cette **notion d'héritage** qu'on a pu **déclarer les méthodes** fetchMany() et fetchOne() qui seront héritées dans les classes enfants ce qui évitera la répétition dans les méthodes des modèles.

## 2. Précisez les moyens utilisés :

Ce projet a utilisé VScode et PHP 8.1.

## 3. Avec qui avez-vous travaillé ?

#### 4. Contexte

Nom de l'entreprise, organisme ou association ▶ *Ecole O'clock*

Chantier, atelier, service ▶ *Exercice réalisé en cours de formation*

Période d'exercice ▶ Du : 30/10/2023 au : 22/04/2024

#### 5. Informations complémentaires (facultatif)

# DOSSIER PROFESSIONNEL (DP)

## Activité-type 2

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

**CP 8** ▶ *Documenter le déploiement d'une application dynamique web ou web mobile*

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

La plateforme de conteneurs **Docker** peut être utilisée pour plusieurs usages notamment la mise en production d'un site. Dans cet exemple, nous avons utilisé Docker pour déployer un projet Laravel : Pomodoro sur une VM Linux.

Une fois le repo du projet cloné dans /var/www/html, on va installer Docker avec les conteneurs dont on a besoin.

#### 1/ Installation de Docker

Tout d'abord, grâce au gestionnaire de paquets APT, nous avons **installé Docker depuis le terminal** et exécuté les commandes suivantes, d'après la documentation du site officiel :

```
sudo apt update
sudo apt install -y ca-certificates curl
sudo install -m 0755 -d /etc/apt/keyrings
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc
sudo chmod a+r /etc/apt/keyrings/docker.asc
echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt update
sudo apt install -y docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
sudo usermod -aG docker $USER
```

Nous **vérifions la bonne installation des paquets** grâce aux commandes « docker --version » et « docker compose version ». Nous allons pouvoir ainsi utiliser Docker et les commandes « docker build » et « docker compose » pour **construire notre**

**propre image** de Docker par la suite afin d'y intégrer les différents logiciels dont nous aurons besoin, dans des conteneurs respectifs.

## 2/ Création d'une image personnalisée grâce au Dockerfile

Le **Dockerfile** est un fichier propre à Docker, qui permet de créer nos propres images Docker personnalisées à partir d'images existantes. Nous allons donc **nous baser sur l'image de PHP 8.1 avec Apache** et sur le Dockerfile suivant qui permet **d'installer les dépendances nécessaires à Laravel**.

Ce fichier Dockerfile a été placé dans le répertoire « back » de notre projet. On peut pusher ce fichier sur le site DockerHub après s'être connecté pour le réutiliser ultérieurement.

On va construire notre image en lançant la commande « **docker build** » dans le dossier contenant le Dockerfile.

## 3/ Création d'une app multi-conteneurs

Pour que l'installation soit complète, il faut **installer SQL**. Pour que ce soit maintenable plus facilement, on va diviser notre application en plusieurs conteneurs qu'on va faire fonctionner ensemble grâce à **Docker Compose**. On crée donc un **fichier de configuration compose.yaml** qui va contenir la liste des services qui correspondent aux trois conteneurs dont on a besoin : un pour le frontend, un pour le backend et un pour la base de données. **On se basera sur notre image créée précédemment**. A l'intérieur de ce fichier, on va aussi **lier nos volumes** pour le frontend (bind mounts), **configurer la base de données** et les **variables d'environnement** qui y sont liées et **charger la base de données**. On va bien sûr aussi rediriger le port 8000 de l'ordinateur hôte vers le port 80 du conteneur.

```
version: "3"
services:
  frontend:
    image: httpd
    ports:
      - "8000:80"
    volumes:
      - ./front:/usr/local/apache2/htdocs
  backend:
    image: PSEUDO-DH/pomodoro-backend-laravel
    ports:
      - "8080:80"
    environment:
      - DB_HOST=db
      - DB_DATABASE=pomodoro
      - DB_USERNAME=root
      - DB_PASSWORD=pomodoro
  db:
    image: mariadb
    environment:
      - MARIADB_ROOT_PASSWORD=pomodoro
      - MARIADB_DATABASE=pomodoro
    volumes:
      - ./DB.sql:/docker-entrypoint-initdb.d/DB.sql
```



# DOSSIER PROFESSIONNEL (DP)

Pour démarrer tous les conteneurs, on lance la commande « docker compose up -d -d » permettant de lancer le conteneur en tâche de fond, on arrêtera avec « docker compose down ».

## 2. Précisez les moyens utilisés :

L'installation de Docker s'est effectuée sur une VM Cloud Linux fournie par l'organisme de formation. L'application Docker a été installée depuis un CDN.

## 3. Avec qui avez-vous travaillé ?

## 4. Contexte

Nom de l'entreprise, organisme ou association ▶ *Ecole O'clock*

Chantier, atelier, service ▶ *Exercice réalisé en cours de formation*

Période d'exercice ▶ Du : 30/10/2023 au : 22/04/2024

## 5. Informations complémentaires (facultatif)

## Titres, diplômes, CQP, attestations de formation

*(facultatif)*

Intitulé	Autorité ou organisme	Date





# DOSSIER PROFESSIONNEL (DP)

## Déclaration sur l'honneur

Je soussigné(e) Laure SENG ,  
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 La-Salvetat-Saint-Gilles le 28 avril 2024

pour faire valoir ce que de droit.

Signature :

## Documents illustrant la pratique professionnelle

*(facultatif)*

Intitulé



MINISTÈRE CHARGÉ  
DE L'EMPLOI

# DOSSIER PROFESSIONNEL (DP)

## ANNEXES