



DOSSIER PROFESSIONNEL (DP)



<i>Nom de naissance</i>	- Du Breul de Saconay
<i>Nom d'usage</i>	- Du Breul de Saconay
<i>Prénom</i>	- Benoit
<i>Adresse</i>	- 14 rue du Tageret 14540 Soliers

Titre professionnel visé

Développeur Web et Web Mobile

MODALITÉ D'ACCÈS :

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

DOSSIER PROFESSIONNEL (DP)

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 ;

DOSSIER PROFESSIONNEL^(DP)

- 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.	5
» Intitulé de l'exemple n° 1 Maquetter une interface utilisateur web	p.	6
» Intitulé de l'exemple n° 2 Réaliser une interface utilisateur web statique	p.	9
» Intitulé de l'exemple n° 3 Réaliser une interface utilisateur web dynamique	p.	16

Développer la partie back-end d'une application web ou web mobile sécurisée	p.	
» Intitulé de l'exemple n° 1 Mettre en place une base de données relationnelle	p.	21
» Intitulé de l'exemple n° 2 Développer des composants d'accès aux données SQL et NoSQL	p.	25
» Intitulé de l'exemple n° 3 Développer des composants métier côté serveur	p.	33

Titres, diplômes, CQP, attestations de formation (facultatif)	p.	37
--	-----------	-----------

DOSSIER PROFESSIONNEL^(DP)

Déclaration sur l'honneur

p. 38

Documents illustrant la pratique professionnelle (*facultatif*)

p. 39

Annexes (*Si le RC le prévoit*)

p. 40

DOSSIER PROFESSIONNEL^(DP)

EXEMPLES DE PRATIQUE PROFESSIONNELLE

DOSSIER PROFESSIONNEL^(DP)

Activité-type 1 Développer la partie front-end d'une application web ou web mobile sécurisée

Exemple n°1 - Maquettage de site web personnel

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

Lors de la création de la maquette pour mon site WordPress:

« <https://benoitdubreul.wordpress.com/> »

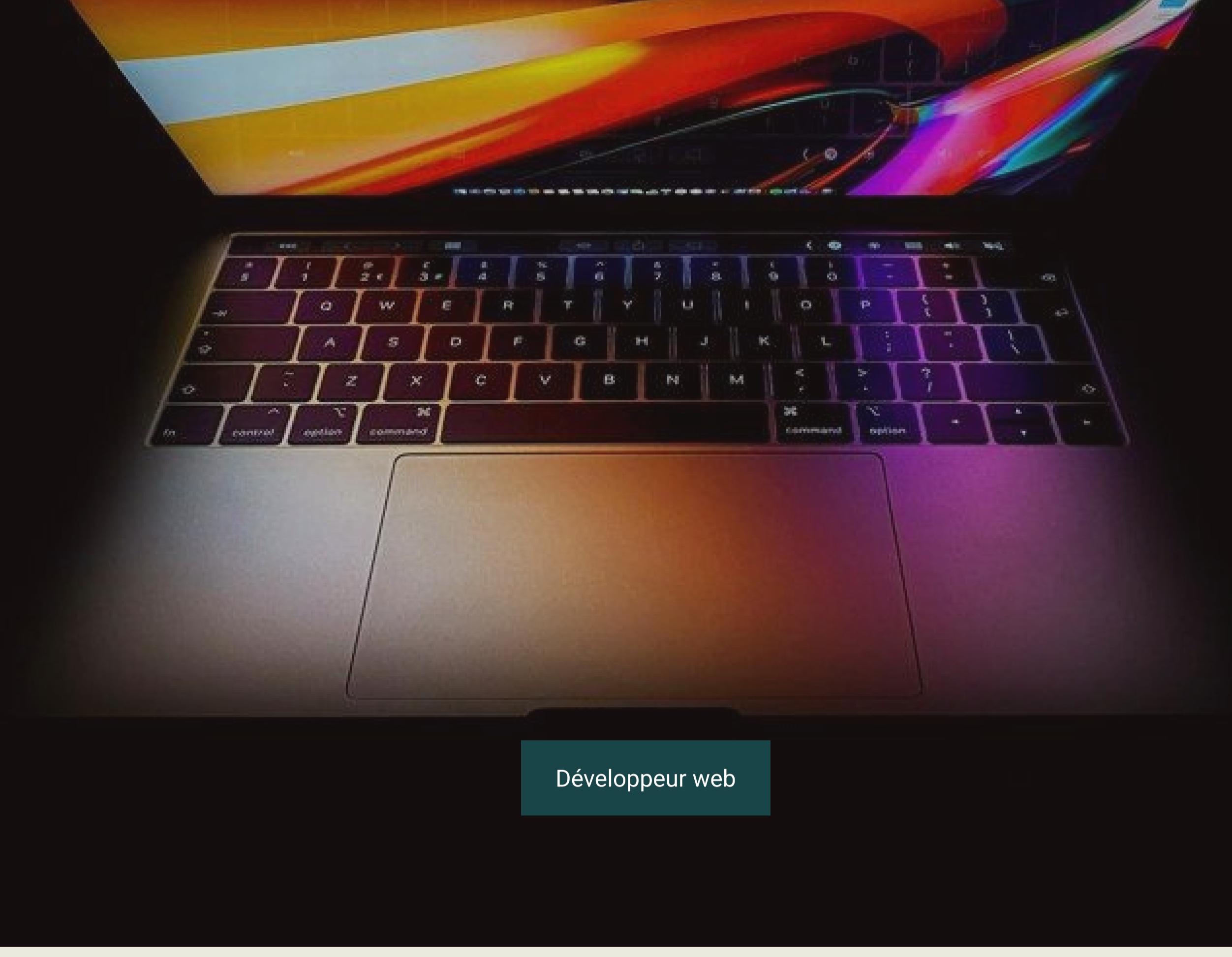
J'ai commencé par identifier les éléments essentiels à intégrer : une présentation claire de mes compétences en développement web, un portfolio de projets réalisés, ainsi que des sections dédiées aux services offerts et aux contacts.

J'ai conçu un design moderne et épuré, en cohérence avec l'identité professionnelle que je souhaitais véhiculer. J'ai veillé à ce que la navigation soit intuitive, optimisée pour tous les types d'écrans. Cette maquette a servi de guide pour la construction du site sur WordPress.

Ce design moderne et épuré reflète mon identité professionnelle en tant que développeur web. Vous y verrez comment j'ai structuré les sections essentielles, telles que la présentation de mes compétences, mon portfolio, et les services proposés.

Du Breul de Saconay Benoit

Développeur web ici!



Développeur web

HTM
L

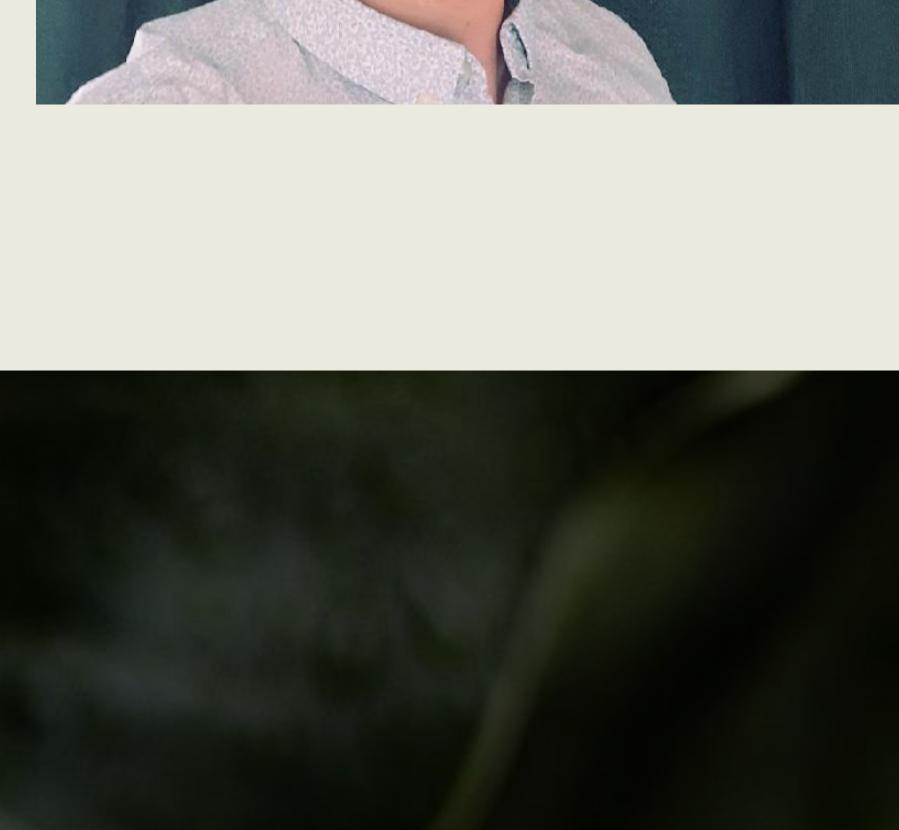
CSS

PH
P

JAVASCRIPT

Je dédie mon temps personnel à la maîtrise dans le développement web. En utilisant mes connaissances afin de créer des expériences numériques exceptionnelles. Ma passion pour l'innovation et mon engagement pour la précision me poussent à fournir un travail de qualité.

L'une de mes créations



Mes services proposés

WordPress

Je peux améliorer votre site web grâce à des conceptions web captivantes

Figma

Pour transformer vos idées en designs réactifs et interactifs!

Trello

Permet de gérer n'importe quel type de projet ou de flux de travail, ou encore de suivre les tâches!

Lucid

Pour des diagrammes toujours plus efficaces!

Prenez contact avec moi pour des solutions web réalisées par un développeur web passionné

Contactez moi

Mon mail de contact

Contactez moi dès aujourd'hui

Commençons à construire des sites web ensemble !

DOSSIER PROFESSIONNEL^(DP)

2. Précisez les moyens utilisés :

Pour réaliser la maquette du site WordPress, j'ai utilisé Figma, un outil de conception d'interface collaborative, permettant de créer des prototypes interactifs et des designs responsive. J'ai intégré des éléments graphiques et des icônes vectorielles pour améliorer l'aspect visuel. L'optimisation du site a été réalisée avec WordPress, en utilisant des thèmes personnalisables et des plugins pour étendre les fonctionnalités.

3. Avec qui avez-vous travaillé ?

J'ai travaillé seul sur ce projet, de la maquette à la création du site web.

4. Contexte

Nom de l'entreprise, organisme ou association ➤ Seul à domicile via l'école Studi

Chantier, atelier, service ➤

Période d'exercice ➤ Du : 01/07/2024 au : 07/07/2024

5. Informations complémentaires (*facultatif*)

DOSSIER PROFESSIONNEL^(DP)

Activité-type 1 Développer la partie front-end d'une application web ou web mobile sécurisée

Exemple n°2 - Le super compteur

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

Lors de la création de l'application web « Le Super Compteur », j'ai conçu une interface utilisateur statique et minimaliste en utilisant HTML, CSS et JavaScript. Le projet a été réalisé dans des conditions où la performance et la simplicité étaient prioritaires. J'ai développé une interface responsive, adaptée aux écrans mobiles et de bureau, avec un compteur interactif permettant à l'utilisateur de modifier la valeur en temps réel. Le design a été épuré pour se concentrer sur l'efficacité et la clarté des fonctionnalités.

Le projet « Le Super Compteur » est une application web simple qui permet aux utilisateurs d'incrémenter et de décrémenter un compteur via des boutons interactifs. L'interface utilisateur est minimaliste, avec un compteur numérique affiché en grand au centre de la page et des boutons "+" et "-" situés en dessous pour ajuster la valeur du compteur .

DOSSIER PROFESSIONNEL^(DP)

```
const counter = 0;
const totalResult = document.getElementById("total");
const btnreset = document.getElementById("reset");
const message = document.getElementById("message");

function incremente() {
    counter++;
    totalResult.textContent = counter;
}

const plusbtn = document.getElementById("plusbtn");

plusbtn.addEventListener("click", (event) => {
    incremente();
    checkCounter();
});

function decremente() {
    counter--;
    totalResult.textContent = counter;
}

const moinsbtn = document.getElementById("moins-btn");
moinsbtn.addEventListener("click", (event) => {
    decremente();
    checkCounter();
});
```

DOSSIER PROFESSIONNEL (DP)

```
function resetbtn() {
    counter = 0;
    totalResult.textContent = counter;
    message.style.color = ""; // Réinitialise la couleur du texte de message
}
btnreset.addEventListener("click", (event) => {
    resetbtn();
});

function checkCounter() {
    if (counter < 3 && counter > 0) {
        message.style.color = "red";
    } else if (counter > 3 && counter < 10) {
        message.style.color = "green";
    } else if (counter > 10) {
        message.style.color = "blue";
    } else if (counter < 0) {
        message.style.color = "yellow";
    }
}
```

Fonctionnalités du Code

1. **Initialisation** : Définit la valeur initiale du compteur à 0 et récupère les éléments HTML nécessaires pour l'affichage et l'interaction (le total du compteur, le bouton de réinitialisation, et le message).
2. **Incrémantation** :
 - **Fonction incrementer()** : Augmente la valeur du compteur de 1 et met à jour l'affichage du compteur.
3. **Décrémentation** :
 - **Fonction decremente()** : Diminue la valeur du compteur de 1 et met à jour l'affichage du compteur.

DOSSIER PROFESSIONNEL^(DP)

4. Réinitialisation :

- **Fonction resetbtn()** : Remet le compteur à 0 et réinitialise la couleur du message.

5. Écouteurs d'Événements :

- **Bouton +** : Incrémente le compteur lorsque cliqué et vérifie la couleur du message.
- **Bouton -** : Décrémente le compteur lorsque cliqué et vérifie la couleur du message.
- **Bouton reset** : Réinitialise le compteur lorsque cliqué.

6. Changement de Couleur :

- **Fonction checkCounter()** : Change la couleur du texte du message en fonction de la valeur du compteur :
 - Rouge pour les valeurs entre 1 et 2.
 - Vert pour les valeurs entre 4 et 9.
 - Bleu pour les valeurs au-dessus de 10.
 - Jaune pour les valeurs négatives.
 -

En résumé, ce code permet à l'utilisateur d'incrémenter, de décrémenter et de réinitialiser un compteur, tout en changeant la couleur d'un message en fonction de la valeur actuelle du compteur.

DOSSIER PROFESSIONNEL (DP)

Le superCompteur !

Calcul

0

-

+

reset

Ce super compteur vous permet d'incrémenter ou bien de décrementer de 1.

L'interface du super compteur présente un titre centré, un affichage clair de la valeur du compteur, des boutons pour incrémenter, décrémenter et réinitialiser, ainsi qu'un message dynamique dont la couleur change en fonction de la valeur du compteur.

DOSSIER PROFESSIONNEL^(DP)

2. Précisez les moyens utilisés :

Pour réaliser l'application « Le Super Compteur », j'ai utilisé Visual Studio Code (VSCode) comme environnement de développement principal, permettant une gestion efficace du code avec ses nombreuses extensions. Le projet a été codé en HTML, CSS et JavaScript pour créer les interactions dynamiques. J'ai également utilisé Netlify pour déployer et héberger l'application. Ces outils ont permis de garantir un développement fluide, optimisé pour le web et les mobiles.

Netlify offre plusieurs avantages en matière de sécurité pour les sites web déployés sur sa plateforme :

1. **HTTPS automatique** : Netlify génère automatiquement des certificats SSL pour sécuriser les connexions avec HTTPS, sans configuration complexe.
2. **Protection contre les attaques DDoS** : La plateforme bénéficie d'une infrastructure résiliente pour atténuer les attaques par déni de service distribué (DDoS).
3. **Mises à jour automatiques** : Les déploiements sont automatisés, garantissant que le site est toujours à jour avec les dernières modifications de code, réduisant ainsi les risques de failles.

Ces fonctionnalités permettent de sécuriser efficacement les projets web hébergés sur Netlify.

3. Avec qui avez-vous travaillé ?

Seul

DOSSIER PROFESSIONNEL^(DP)

4. Contexte

Nom de l'entreprise, organisme ou association ➤ Studi

Chantier, atelier, service ➤

Période d'exercice ➤ Du :12/06/2024 au : **15/06/2024**

5. Informations complémentaires (*facultatif*)

DOSSIER PROFESSIONNEL (DP)

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

Exemple n° 3 - To Do List

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

Pour la création de l'application « To-Do List », j'ai conçu et développé une interface utilisateur statique et réactive en utilisant HTML, CSS, et JavaScript. Le projet a été réalisé sous des conditions où la simplicité d'utilisation et l'efficacité étaient prioritaires. J'ai mis en place une structure de gestion des tâches permettant à l'utilisateur d'ajouter, de cocher ou de supprimer des éléments de sa liste de tâches. Le design est minimaliste pour garantir une expérience utilisateur claire et directe.

Vous trouverez ci-joint le code JavaScript permettant le dynamisme de cette ToDoList.

```
document.getElementById('add-task-button').addEventListener('click', function() {
    let taskText = document.getElementById('task-input').value;
    if (taskText === '') return;

    let taskItem = document.createElement('li');
    taskItem.className = 'task';
    taskItem.innerHTML =
        <span>${taskText}</span>
        <button class="delete-task">x</button>
    ;

    taskItem.querySelector('span').addEventListener('click', function() {
        this.classList.toggle('completed');
    });

    taskItem.querySelector('.delete-task').addEventListener('click', function() {
        taskItem.remove();
    });

    document.getElementById('task-list').appendChild(taskItem);
    document.getElementById('task-input').value = '';
});
```

DOSSIER PROFESSIONNEL (DP)

2. Précisez les moyens utilisés :

J'ai utilisé **Visual Studio Code (VSCode)** comme IDE principal pour l'écriture et l'organisation du code. L'application a été hébergée sur **Netlify**, ce qui a permis de bénéficier d'avantages en termes de sécurité, tels que le déploiement automatique avec **HTTPS**, la protection contre les attaques DDoS, et une infrastructure fiable pour assurer la disponibilité continue du site.

Ce code JavaScript gère l'ajout, la complétion et la suppression de tâches dans une liste de tâches. Voici une explication de chaque partie du code :

Fonctionnalités du Code

1. Écouteur d'Événement pour le Bouton "Ajouter Tâche" :

- **Action** : Lorsque le bouton avec l'ID `add-task-button` est cliqué, la fonction associée est exécutée.

2. Création d'une Nouvelle Tâche :

- **Texte de la Tâche** : Récupère le texte saisi dans l'élément avec l'ID `task-input`. Si le champ est vide, la fonction se termine sans ajouter de tâche.
- **Élément de Tâche** : Crée un nouvel élément `` avec la classe `task`, contenant le texte de la tâche et un bouton de suppression.

3. Ajout d'Écouteurs d'Événements pour la Tâche :

- **Complétion de la Tâche** : Ajoute un événement de clic sur le texte de la tâche pour basculer la classe `completed` (pour marquer la tâche comme terminée).
- **Suppression de la Tâche** : Ajoute un événement de clic sur le bouton de suppression pour retirer l'élément de la liste.

4. Ajout de la Tâche à la Liste :

- **Insertion** : Ajoute le nouvel élément de tâche à la liste des tâches (`task-list`).
- **Réinitialisation du Champ de Saisie** : Vide le champ de saisie de la tâche après ajout.

Résumé

Ce code permet aux utilisateurs de :

- Ajouter des tâches à une liste.
- Marquer les tâches comme complètes en cliquant sur leur texte.
- Supprimer les tâches en cliquant sur un bouton dédié à la suppression.

DOSSIER PROFESSIONNEL (DP)



DOSSIER PROFESSIONNEL^(DP)

3. Avec qui avez-vous travaillé ?

Seul

4. Contexte

Nom de l'entreprise, organisme ou association ➤

Chantier, atelier, service ➤

Période d'exercice ➤ Du : 20/06/2024 au : **25/06/2024**

5. Informations complémentaires (*facultatif*)

Le body est stylisé pour utiliser la police Arial, avec un affichage en flexbox centré horizontalement et verticalement (justify-content: center; align-items: center;). Le fond est d'un vert foncé (#11381a), avec une image de fond centrée, non répétée, et redimensionnée à 50%.

- .todo-container est un conteneur de 300px de large avec un fond vert clair (#afdf23c), des coins arrondis, et une ombre légère pour l'effet de profondeur.
- #task-input est un champ de texte large avec centrage et marges adaptées.
- Les boutons sont stylisés avec un fond vert (#28a745), du texte blanc et sans bordure pour un aspect propre.

DOSSIER PROFESSIONNEL (DP)

```
body {  
    font-family: Arial, sans-serif;  
    display: flex;  
    justify-content: center;  
    align-items: center;  
    height: 100vh;  
    margin: 0;  
    background-color: #11381a;  
    background-size: 50%; /* Agrandit l'image pour qu'elle paraisse plus éloignée */  
    background-position: center; /* Centre l'image horizontalement et verticalement */  
    background-repeat: no-repeat; /* Empêche la répétition de l'image */  
}  
  
.todo-container {  
    background-color: #afdb3c;  
    padding: 20px;  
    border-radius: 8px;  
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);  
    width: 300px;  
    text-align: center;  
}  
  
#task-input {  
    width: calc(100% - 20px);  
    padding: 10px;  
    margin-bottom: 10px;  
    text-align: center;  

```

DOSSIER PROFESSIONNEL (DP)

Activité-type 2 Développer la partie back-end d'une application web ou web mobile sécurisée

Exemple n° 1 - Mettre en place une base de données relationnelles

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

Dans le cadre de mon projet, la gestion de la base de données joue un rôle central pour stocker et gérer de manière persistante les informations relatives au compteur. J'ai utilisé un système de gestion de bases de données relationnelles, pour structurer, sauvegarder, et récupérer les données associées au compteur et à son historique.

J'ai utilisé phpMyAdmin pour générer un dump SQL de la base de données super_compteur_db. Ce dump contient à la fois la structure et les données des tables utilisées dans mon application de compteur. Voici les tâches que j'ai réalisées :

Structure de la table compteur :

- J'ai créé une table nommée compteur, destinée à stocker la valeur actuelle du compteur, avec deux colonnes :
 - id (clé primaire auto-incrémente) pour identifier chaque enregistrement.
 - counter_value pour stocker la valeur actuelle du compteur.

Données initiales dans la table compteur :

- J'ai inséré une ligne initiale dans cette table avec un id de 1 et une valeur de compteur à 0.
- **Structure de la table compteur_history :**
- J'ai aussi créé une table nommée compteur_history pour stocker l'historique des valeurs du

DOSSIER PROFESSIONNEL (DP)

compteur lors des réinitialisations. Cette table comporte trois colonnes :

- id (clé primaire auto-incrémentée).
- counter_value pour enregistrer la valeur du compteur lors de chaque réinitialisation.
- reset_time pour capturer la date et l'heure exactes de la réinitialisation, avec un timestamp par défaut sur l'heure actuelle.

- **Données dans la table compteur_history :**

- J'ai inséré des données dans cette table, représentant les réinitialisations du compteur avec des valeurs et des heures précises :
 - Par exemple, la première réinitialisation a enregistré une valeur de 3 à la date du 2024-09-05 à 19:55:16.

- **Index et clés primaires :**

- J'ai ajouté des clés primaires sur les colonnes id des deux tables (compteur et compteur_history) afin d'assurer une identification unique pour chaque enregistrement.
- J'ai ensuite modifié la colonne id dans chaque table pour être auto-incrémentée, facilitant ainsi l'ajout automatique de nouvelles lignes avec un identifiant unique.

```
CREATE TABLE `compteur` (
    `id` int(11) NOT NULL,
    `counter_value` int(11) NOT NULL
)
```

```
INSERT INTO `compteur`(`id`, `counter_value`) VALUES
(1, 0);
```

```
CREATE TABLE `compteur_history` (
    `id` int(11) NOT NULL,
    `counter_value` int(11) NOT NULL,
    `reset_time` timestamp NOT NULL DEFAULT current_timestamp()
)
```

DOSSIER PROFESSIONNEL (DP)

2. Précisez les moyens utilisés :

Pour mener à bien ce projet, j'ai utilisé plusieurs outils et technologies afin de mettre en place la base de données et de gérer les composants métiers :

1. **phpMyAdmin** :

- J'ai utilisé **phpMyAdmin** version 5.2.1 pour gérer et administrer ma base de données MySQL. Cet outil graphique m'a permis d'effectuer des opérations comme la création de tables, l'insertion de données.
- **phpMyAdmin** m'a également facilité la visualisation des structures des tables, ainsi que des données enregistrées.

2. **MySQL** :

- La base de données que j'ai utilisée est MySQL, qui offre des fonctionnalités robustes pour la gestion des données relationnelles.
- J'ai utilisé des **requêtes SQL** pour créer les tables, insérer les données et définir les index et les clés primaires nécessaires au bon fonctionnement de l'application.

3. Avec qui avez-vous travaillé ?

seul

4. Contexte

Nom de l'entreprise, organisme ou association ➤ studi

Chantier, atelier, service ➤

Période d'exercice ➤ Du : 10/08/2024 au : **15/08/2024**

DOSSIER PROFESSIONNEL^(DP)

5. Informations complémentaires (*facultatif*)

DOSSIER PROFESSIONNEL^(DP)

Activité-type 2 Développer la partie back-end d'une application web ou web mobile sécurisée

Exemple n° 2 - Développer des Composants d'Accès aux Données SQL et NoSQL

DOSSIER PROFESSIONNEL (DP)

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

Configuration et Connexion à la Base de Données

- **Tâches :**
 - **Configuration des Paramètres de Connexion :** J'ai défini les paramètres nécessaires pour accéder à la base de données (serveur, utilisateur, mot de passe, nom de la base).
 - **Création de la Connexion :** J'ai utilisé l'extension `mysql` pour établir la connexion avec la base de données MySQL.
 - **Vérification de la Connexion :** J'ai mis en place une vérification pour m'assurer que la connexion est réussie. En cas d'échec, le script se termine avec un message d'erreur.
 - **Définition de l'Encodage des Caractères :** J'ai configuré l'encodage des caractères à `utf8` pour garantir une gestion correcte des données textuelles.
- **Conditions :**
 - La connexion fonctionne correctement si les informations de la base de données sont exactes et accessibles.
 - L'encodage est défini pour éviter les problèmes d'affichage ou de stockage des caractères spéciaux.

2. Récupération ou Création d'un Compteur

- **Tâches :**
 - **Vérification de l'Existence d'un Compteur :** J'ai exécuté une requête SQL pour vérifier si un enregistrement du compteur existe déjà dans la table `compteur`.
 - **Récupération des Données :** Si un enregistrement est trouvé, j'ai récupéré la valeur du compteur et l'ai stockée dans une session.
 - **Insertion d'un Nouveau Compteur :** Si aucun enregistrement n'existe, j'ai inséré un nouveau compteur avec une valeur initiale de 0 et stocké l'identifiant nouvellement créé dans la session.
- **Conditions :**
 - Les données doivent être accessibles dans la table `compteur`. Sinon, un nouvel enregistrement est créé pour initialiser le compteur.
 - L'utilisation des sessions permet de conserver la valeur du compteur entre les requêtes de l'utilisateur.

DOSSIER PROFESSIONNEL (DP)

3. Traitement des Actions de l'Utilisateur via AJAX

- **Tâches :**
 - **Traitement des Requêtes POST :** J'ai géré les requêtes AJAX envoyées via POST pour effectuer des actions telles que increment, decrement, et reset sur le compteur.
 - **Utilisation de Requêtes Préparées :** Pour sécuriser les opérations de mise à jour et d'insertion, j'ai utilisé des requêtes préparées avec des paramètres liés.
- **Conditions :**
 - Les requêtes POST doivent être correctement formatées et contenir des actions valides (increment, decrement, reset).
 - Les requêtes préparées préviennent les injections SQL et garantissent que les données sont correctement insérées ou mises à jour dans la base de données.

4. Mise à Jour des Données dans la Base de Données

- **Tâches :**
 - **Insertion dans compteur_history :** Lors d'une réinitialisation du compteur, j'ai inséré la valeur actuelle du compteur dans la table `compteur_history` pour conserver un historique.
 - **Mise à Jour dans compteur :** J'ai mis à jour la valeur du compteur dans la table `compteur` en fonction des actions effectuées par l'utilisateur.
- **Conditions :**
 - Les données doivent être mises à jour de manière atomique pour éviter les incohérences.
 - Les modifications doivent être correctement reflétées dans les tables `compteur` et `compteur_history`.

Résumé

J'ai développé des composants d'accès aux données SQL en configurant et établissant une connexion à la base de données, en récupérant ou créant un compteur selon les besoins, en traitant les actions de l'utilisateur avec des requêtes préparées sécurisées, et en mettant à jour les données dans les tables appropriées. Les opérations ont été réalisées dans un environnement où les paramètres de connexion sont corrects et les données sont correctement formatées et sécurisées.

DOSSIER PROFESSIONNEL (DP)

2. Précisez les moyens utilisés :

```
<?php
// Configuration de la base de données
$servername = "localhost";
$username = "root";
$password = "root";
$dbname = "super_compteur_db";

// Créer une connexion
$conn = new mysqli($servername, $username, $password, $dbname);

// Vérifier la connexion
if ($conn->connect_error) {
    die("La connexion a échoué: " . $conn->connect_error);
}

// Définir l'encodage de caractères
$conn->set_charset("utf8");
```

Cette section de code établit la connexion à la base de données MySQL. La connexion est vérifiée pour détecter toute erreur, et l'encodage des caractères est défini pour assurer une manipulation correcte des données textuelles.

```
// Initialiser le compteur s'il n'est pas encore défini
if (!isset($_SESSION['counter_id'])) {
    // Récupérer ou créer un enregistrement du compteur
    $result = $conn->query("SELECT * FROM compteur LIMIT 1");
    if ($result->num_rows > 0) {
        $row = $result->fetch_assoc();
        $_SESSION['counter_id'] = $row['id'];
        $_SESSION['counter_value'] = $row['counter_value'];
    } else {
        // Insérer un enregistrement initial si aucun n'existe
        $conn->query("INSERT INTO compteur (counter_value) VALUES (0)");
        $_SESSION['counter_id'] = $conn->insert_id;
        $_SESSION['counter_value'] = 0;
    }
}
```

DOSSIER PROFESSIONNEL (DP)

Ce code récupère l'état du compteur de la base de données. Si aucun compteur n'existe encore, il en crée un avec une valeur initiale de 0.

```
// Gérer les requêtes AJAX
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $action = $_POST['action'] ?? '';

    if ($action === 'increment') {
        $_SESSION['counter_value']++;
    } elseif ($action === 'decrement') {
        $_SESSION['counter_value']--;
    } elseif ($action === 'reset') {
        // Sauvegarder la valeur du compteur dans la table historique
        $counter_value = $_SESSION['counter_value'];
        $stmt = $conn->prepare("INSERT INTO compteur_history (counter_value) VALUES (?)");
        $stmt->bind_param("i", $counter_value);
        $stmt->execute();
        $stmt->close();

        // Réinitialiser le compteur
        $_SESSION['counter_value'] = 0;
    }

    // Mettre à jour la base de données
    $counter_value = $_SESSION['counter_value'];
    $counter_id = $_SESSION['counter_id'];
    $stmt = $conn->prepare("UPDATE compteur SET counter_value = ? WHERE id = ?");
    $stmt->bind_param("ii", $counter_value, $counter_id);
    $stmt->execute();
    $stmt->close();

    echo json_encode(['counter_value' => $counter_value]);
    exit;
}
```

Les requêtes préparées sont utilisées pour insérer les valeurs du compteur dans la table `compteur_history` et pour mettre à jour la valeur actuelle dans la table `compteur`. Cela assure la sécurité des opérations en prévenant les injections SQL.

DOSSIER PROFESSIONNEL (DP)

Développement de Composants d'Accès aux Données NoSQL

Pour intégrer Firebase Firestore dans mon application, voici ce que j'ai réalisé :

1. **Configuration de Firebase** : J'ai initialisé l'application Firebase avec les paramètres de configuration fournis. Cela a permis de connecter mon application à Firestore.

2. **Gestion des Données avec Firestore** :

- **Obtention de la Valeur Actuelle** : J'ai utilisé la fonction `getCounterValue` pour lire la valeur du compteur depuis Firestore.
- **Mise à Jour du Compteur** : Avec la fonction `updateCounter`, j'ai modifié la valeur du compteur selon l'action demandée (incrémenter, décrémenter, réinitialiser) et mis à jour Firestore en conséquence.

3. **Fonction `updateCounter`** :

- **Lecture** : J'ai d'abord récupéré la valeur actuelle du compteur.
- **Calcul** : En fonction de l'action reçue (`increment`, `decrement`, `reset`), j'ai calculé la nouvelle valeur du compteur.
- **Écriture** : J'ai ensuite mis à jour cette valeur dans Firestore pour la rendre persistante.
- **Retour** : La nouvelle valeur du compteur a été renvoyée pour mettre à jour l'affichage sur la page.

Cette approche garantit que les modifications apportées au compteur sont stockées de manière fiable et cohérente dans la base de données NoSQL, tout en assurant une interaction fluide avec l'utilisateur via l'interface web.

DOSSIER PROFESSIONNEL (DP)

```
// Importer les bibliothèques Firebase nécessaires
import { initializeApp } from "https://www.gstatic.com/firebasejs/9.9.4/firebase-app.js";
import { getFirestore, doc, getDoc, setDoc } from "https://www.gstatic.com/firebasejs/9.9.4/firebase-firebase.js";

// Configuration de Firebase
const firebaseConfig = {
    apiKey: "AIzaSyAXfQJUqM2C1nQnw-wwTqTlA9cUTbrxzso",
    authDomain: "supercompteur-14.firebaseio.com",
    projectId: "supercompteur-14",
    storageBucket: "supercompteur-14.appspot.com",
    messagingSenderId: "3903353421",
    appId: "1:3903353421:web:32256567a7f3163a99c204"
};

// Initialiser Firebase avec la configuration fournie
const app = initializeApp(firebaseConfig);
const db = getFirestore(app);

// Fonction pour obtenir la valeur actuelle du compteur depuis Firestore
async function getCounterValue() {
    // Référence au document contenant la valeur du compteur
    const docRef = doc(db, 'counter', 'value');
    // Lire le document depuis Firestore
    const docSnap = await getDoc(docRef);
    // Vérifier si le document existe
    if (docSnap.exists()) {
        // Retourner la valeur du compteur si le document existe
        return docSnap.data().value;
    } else {
        // Retourner une valeur par défaut (0) si le document n'existe pas
        return 0;
    }
}

// Fonction pour mettre à jour la valeur du compteur dans Firestore
async function updateCounter(action) {
    const docRef = doc(db, 'counter', 'value');
    let newValue;

    // Obtenir la valeur actuelle du compteur
    const currentCounter = await getCounterValue();

    // Déterminer la nouvelle valeur du compteur en fonction de l'action
    if (action === 'increment') {
        newValue = currentCounter + 1;
    } else if (action === 'decrement') {
        newValue = currentCounter - 1;
    } else if (action === 'reset') {
        newValue = 0;
    }

    // Mettre à jour le document dans Firestore avec la nouvelle valeur du compteur
    await setDoc(docRef, { value: newValue });
    // Retourner la nouvelle valeur du compteur
    return newValue;
}
```

DOSSIER PROFESSIONNEL^(DP)

3. Avec qui avez-vous travaillé ?

seul

4. Contexte

Nom de l'entreprise, organisme ou association ➤ studi

Chantier, atelier, service ➤

Période d'exercice ➤ Du : 15/08/2024 au : **20/08/2024**

5. Informations complémentaires (*facultatif*)

DOSSIER PROFESSIONNEL^(DP)

Activité-type 2 Développer la partie back-end d'une application web ou web mobile sécurisée

Exemple n° 3 - Développer des Composants Métier Côté Serveur

DOSSIER PROFESSIONNEL (DP)

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

Dans le cadre de ce projet, j'ai développé les composants métier côté serveur pour gérer l'état d'un compteur et ses interactions avec l'utilisateur. Voici un résumé des tâches effectuées :

1. **Initialisation de la Session** : J'ai démarré une session avec `session_start()` pour suivre et maintenir la valeur du compteur entre les différentes requêtes de l'utilisateur. Cette initialisation est cruciale pour garantir que les données du compteur restent cohérentes tout au long de la navigation.
2. **Gestion du Compteur** : Lors de la première visite ou si la session n'a pas encore de compteur, j'ai vérifié la présence d'un enregistrement dans la base de données. Si un compteur existait, j'ai récupéré sa valeur et l'ai stockée dans la session. Sinon, j'ai créé un nouveau compteur avec une valeur initiale de 0 et ai enregistré son identifiant dans la session. Cette approche assure que chaque utilisateur a un compteur actif dès le début.
3. **Traitement des Actions Utilisateur via AJAX** : J'ai mis en place la logique pour gérer les requêtes AJAX POST permettant à l'utilisateur d'incrémenter, décrémenter ou réinitialiser le compteur.
 - Pour l'action `increment`, j'ai augmenté la valeur du compteur de 1.
 - Pour l'action `decrement`, j'ai diminué la valeur du compteur de 1.
 - Pour l'action `reset`, j'ai sauvegardé la valeur actuelle du compteur dans l'historique avant de le réinitialiser à 0.
4. **Sauvegarde et Réinitialisation** : Lorsqu'un utilisateur demande la réinitialisation du compteur, j'ai d'abord enregistré la valeur actuelle dans la table `compteur_history`. Ensuite, j'ai réinitialisé le compteur à 0 pour préparer le compteur pour une nouvelle série de modifications.
5. **Mise à Jour de la Base de Données** : Après chaque action utilisateur, j'ai mis à jour la valeur du compteur dans la base de données pour refléter les changements effectués. Cette mise à jour est essentielle pour garantir que l'état du compteur est correctement synchronisé avec les modifications apportées par l'utilisateur.

En résumé, j'ai conçu des composants métier côté serveur qui gèrent l'état du compteur, traitent les actions des

DOSSIER PROFESSIONNEL (DP)

utilisateurs, maintiennent un historique des valeurs, et assurent que les modifications sont correctement enregistrées dans la base de données. Ces tâches sont réalisées en maintenant une cohérence et une intégrité des données tout au long du processus.

2. Précisez les moyens utilisés :

```
// Initialiser le compteur s'il n'est pas encore défini
if (!isset($_SESSION['counter_id'])) {
```

Ce segment initialise le compteur en vérifiant s'il existe déjà dans la session. S'il n'existe pas, il récupère le compteur de la base de données ou en crée un nouveau. Cette logique métier gère l'état du compteur dans la session utilisateur.

```
// Gérer les requêtes AJAX
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $action = $_POST['action'] ?? '';

    if ($action === 'increment') {
        $_SESSION['counter_value']++;
    } elseif ($action === 'decrement') {
        $_SESSION['counter_value']--;
    } elseif ($action === 'reset') {
        // Sauvegarder la valeur du compteur dans la table historique
        $counter_value = $_SESSION['counter_value'];
        $stmt = $conn->prepare("INSERT INTO compteur_history (counter_value) VALUES (?)");
        $stmt->bind_param("i", $counter_value);
        $stmt->execute();
        $stmt->close();

        // Réinitialiser le compteur
        $_SESSION['counter_value'] = 0;
    }

    // Mettre à jour la base de données
    $counter_value = $_SESSION['counter_value'];
    $counter_id = $_SESSION['counter_id'];
```

DOSSIER PROFESSIONNEL^(DP)

Ce code gère les actions de l'utilisateur via des requêtes POST, comme incrémenter, décrémenter ou réinitialiser le compteur. En fonction de l'action, il met à jour la valeur du compteur dans la session et dans la base de données.

3. Avec qui avez-vous travaillé ?

seul

4. Contexte

Nom de l'entreprise, organisme ou association ➔ studi

Chantier, atelier, service ➔

Période d'exercice ➔ Du : 15/08/2024 au : **20/08/2024**

5. Informations complémentaires (*facultatif*)

DOSSIER PROFESSIONNEL (DP)

Titres, diplômes, CQP, attestations de formation

(facultatif)

DOSSIER PROFESSIONNEL^(DP)

Déclaration sur l'honneur

Je soussigné(e) Benoit Du Breul de Saconay ,

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 Soliers le 09/09/2024

pour faire valoir ce que de droit.

Signature :

DOSSIER PROFESSIONNEL (DP)

Documents illustrant la pratique professionnelle

(facultatif)

DOSSIER PROFESSIONNEL (DP)

ANNEXES

(Si le RC le prévoit)