



DOSSIER PROFESSIONNEL (DP)

MINISTÈRE CHARGÉ
DE L'EMPLOI

Nom de naissance ► IMPINNA
Nom d'usage ►
Prénom ► David
Adresse ► 91 rue de Gravelle 94 700

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 ;
- ▶ 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 n° 1 : Développer la partie FRONT-END d'une application web ou web mobile en intégrant les recommandations de sécurité p. 5

- ▶ exemple n° 1 Maquetter une application avec Figma.....p. 31
- ▶ exemple n° 2 Réaliser une interface web utilisateur statique et adaptable : Site vitrine « *Tindog* ». p. 32
- ▶ exemple n° 3.1 Réaliser une interface utilisateur web dynamique : Personnage jeux vidéos.....p. 35

- ▶ exemple n° 3.2 Réaliser une interface utilisateur web dynamique : Quizz.....p. 38
- ▶ exemple n° 3.3 Réaliser une interface utilisateur web dynamique : IMC.....p. 40
- ▶ exemple n° 4.1 Réaliser une interface utilisateur avec une gestion de contenu ou e-commerce : Boutique Wordpress.....p. 42

- ▶ exemple n° 4.2 Réaliser une interface utilisateur avec une gestion de contenu ou e-commerce : Boutique Symfonyp. 44

DOSSIER PROFESSIONNEL (DP)



Activité-type n° 2 : Développer la partie BACK-END d'une application web ou web mobile en intégrant les recommandations de sécurité

p. 5

- ▶ exemple n° 5.1 Création d'une base de données : BD locale site Wordpress.....p. p. 47
- ▶ exemple n° 5.2 Création d'une base de données : MongoDBCompass.....p. p. 49
- ▶ exemple n° 6 Développer les composants d'accès aux données : site Javascript.....p. p. 50

p.

- ▶ exemple n° 7 Développer la partie back-end d'une application web ou web mobile.....p. p. 52
- ▶ exemple n° 8 Élaborer et mettre en œuvre des composants dans une application de gestion de contenus.....p. p. 57

Titres, diplômes, CQP, attestations de formation (facultatif)

p.

Déclaration sur l'honneur

p.

Documents illustrant la pratique professionnelle (facultatif)

p.

Annexes (Si le RC le prévoit)

p.

EXEMPLES DE PRATIQUE PROFESSIONNELLE

DOSSIER PROFESSIONNEL (DP)

Activité-type 1 Développer la partie FRONT-END d'une application web ou web mobile en intégrant les recommandations de sécurité

Exemple n°1 ► Maquetter une application

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

Nous nous sommes entraînés à utiliser le logiciel de maquettage Figma, un logiciel couramment utilisé dans le design. Il est aussi possible de maquetter en dessinant.

2. Précisez les moyens utilisés :

Figma permet aux utilisateurs de concevoir et de collaborer en ligne. Il offre une variété d'outils pour la conception graphique, tels que la création de formes, la mise en page, la sélection de couleurs et de polices, ainsi que la personnalisation d'icônes et d'images.

Il permet également de faire des tests utilisateur en créant des prototypes interactifs à partir de la maquette, ce qui permet aux utilisateurs de tester l'application avant même son développement.

On utilise des composants pour faciliter la conception d'interface utilisateur. Ce sont des éléments graphiques réutilisables, comme des boutons, des formulaires, des icônes, etc., qui peuvent être créés une seule fois et ensuite utilisés à plusieurs endroits dans la maquette. Cela permet de gagner du temps et d'assurer la cohérence de l'interface utilisateur, car si vous devez effectuer des modifications sur un composant, elles seront automatiquement reflétées partout où il est utilisé dans la maquette.

En somme, Figma est un outil puissant pour la conception d'interface utilisateur, qui offre une variété d'outils pour la conception graphique, une collaboration en temps réel et des fonctionnalités de prototypage interactif.

3. Avec qui avez-vous travaillé ?

Nous avons travaillé avec notre formateur puis seuls.

4. Contexte

Nom de l'entreprise, organisme ou association

Cliquez ici pour taper du texte.

DOSSIER PROFESSIONNEL (DP)

Activité-type 1 Développer la partie FRONT-END d'une application web ou web mobile en intégrant les recommandations de sécurité

Exemple n°2 ► Réaliser une interface web utilisateur statique et adaptable : Vitrine « Tindog »

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

A coté du cours que nous suivions sur le css et le framework Bootstrap, j'ai réalisé avec l'aide d'un cours en ligne une application vitrine statique. J'ai désiré me perfectionner dans cette méthode car sa simplicité et son gain de productivité m'ont beaucoup séduit.

2. Précisez les moyens utilisés :

Je me suis servi de notre cours, du tutoriel et du framework css créé par la compagnie Twitter Bootstrap, ainsi que de sa documentation. Comme le dit la page Wikipédia à son sujet,
« Bootstrap est une collection d'outils utiles à la création du design (graphisme, animation et interactions avec la page dans le navigateur, etc.) de sites et d'applications web. C'est un ensemble qui contient des codes HTML et CSS, des formulaires, boutons, outils de navigation et autres éléments interactifs, ainsi que des extensions JavaScript en option. »

Bootstrap est responsive, ce qui veut dire qu'il permet de créer des sites adaptables automatiquement à toutes tailles d'écrans. Il est aussi facilement personnalisable et possède une grande communauté de développeurs qui fournissent des conseils et des solutions aux problèmes rencontrés lors de l'utilisation de la bibliothèque.

3. Avec qui avez-vous travaillé ?

J'ai travaillé seul avec de la documentation et des tutoriels.

4. Contexte

Nom de l'entreprise, organisme ou association

Cliquez ici pour taper du texte.

DOSSIER PROFESSIONNEL (DP)

Activité-type 1 Développer la partie FRONT-END d'une application web ou web mobile en intégrant les recommandations de sécurité

DOSSIER PROFESSIONNEL (DP)

Exemple n°3.1 ► Réaliser une interface utilisateur web dynamique : Personnage jeu vidéo

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

J'ai crée une petite interface graphique permettant créer un personnage de jeux vidéo avec des caractéristiques qui lui sont propres.

2. Précisez les moyens utilisés :

Pour créer cette interface uniquement graphique il à fallu la construire avec tous les éléments dont elle avait besoin, selon les règles de la bibliothèque React. Ceci permettra de réutiliser l'interface très simplement. Nous avons donc créer les composants réutilisables, à savoir le titre, le personnage (son apparence et ses points de caractéristiques), les armes équipables et les boutons de création et de réinitialisation. Il faut aussi définir ce qu'on appelle les « states », ces objets JavaScript qui contiennent des données dynamiques à l'intérieur de leur composant et qui peuvent être modifiés avec des clics de souris par exemple. Ici les states principaux sont l'apparence du personnage ainsi que ses caractéristiques et leur quantité respectives, et la liste des armes.

J'ai utilisé la REACT, c'est une bibliothèque open-source de développement d'interfaces utilisateur pour le web, créée et maintenue par Facebook. Elle permet de construire des interfaces utilisateur interactives et réactives en utilisant des composants réutilisables.

Au lieu de manipuler directement le DOM (Document Object Model) pour mettre à jour l'interface utilisateur en réponse à des événements, React utilise une structure d'arbre de composants pour décrire la hiérarchie des éléments de l'interface. Lorsque le state d'un composant change, React met à jour le DOM de manière efficace et optimisée.

React est devenu très populaire dans la communauté du développement web en raison de sa simplicité, de sa flexibilité, et de ses performances. Il est utilisé par de nombreuses entreprises pour développer des applications web de toutes tailles et complexités.

React est donc facile d'utilisation car il permet de diviser l'interface utilisateur en petites parties, ce qui facilite la maintenance du code et l'ajout de nouvelles fonctionnalités.

Il est pour cela très populaire et dispose d'une grande communauté de développeurs actifs qui contribuent à son développement et fournissent de nombreux outils et bibliothèques tierces.

3. Avec qui avez-vous travaillé ?

J'ai travaillé seul avec de la documentation et des tutoriels.

4. Contexte

DOSSIER PROFESSIONNEL (DP)

Nom de l'entreprise, organisme ou association

Cliquez ici pour taper du texte.

DOSSIER PROFESSIONNEL (DP)

Activité-type 1 Développer la partie FRONT-END d'une application web ou web mobile en intégrant les recommandations de sécurité

Exemple n°3.2 ▶ Réaliser une interface utilisateur web dynamique : Quizz

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

Il s'agissait ici de réaliser un petit quizz fonctionnant avec du javascript pur sans framework. J'ai en premier lieu codé la structure html en y ajoutant les éléments nécessaires comme des blocs de question à différents choix, un titre, un bouton de validation pour traiter les réponses et l'affichage du score accompagné d'un commentaire.

J'ai ensuite stylisé le tout grâce à du css, puis écrit le code javascript.

Lorsque l'utilisateur soumet le formulaire, les réponses sélectionnées sont récupérées, puis comparées avec les réponses correctes stockées dans un tableau. Ensuite, en fonction du nombre de réponses incorrectes, un message de feedback est affiché pour l'utilisateur, et le score final est calculé et affiché. En outre, ce code utilise des emojis pour fournir un retour visuel sur chaque question en fonction de la réponse de l'utilisateur. Les questions correctement répondues ont un fond vert, tandis que les questions incorrectes ont un fond rouge. Enfin, le code utilise également un gestionnaire d'événements pour réinitialiser la couleur de fond de chaque question si l'utilisateur modifie sa réponse.

2. Précisez les moyens utilisés :

Ce code utilise plusieurs moyens pour réaliser les fonctionnalités du quiz :

La méthode querySelector pour récupérer les éléments HTML correspondants aux différents blocs du quiz.

La méthode addEventListener pour ajouter un événement de soumission de formulaire au bloc de formulaire du quiz.

L'utilisation de tableaux pour stocker les réponses correctes et les résultats de l'utilisateur.

Des boucles forEach pour parcourir les éléments HTML et effectuer des opérations sur chacun d'eux.

L'utilisation d'une instruction switch pour déterminer le message de feedback et le score final à afficher.

Et enfin l'utilisation de la propriété.textContent pour modifier le texte affiché dans les éléments HTML correspondants.

3. Avec qui avez-vous travaillé ?

J'ai travaillé seul avec de la documentation et des tutoriels.

4. Contexte

DOSSIER PROFESSIONNEL (DP)

Nom de l'entreprise, organisme ou association

Cliquez ici pour taper du texte.

Activité-type 1

Développer la partie FRONT-END d'une application web ou web mobile en intégrant les recommandations de sécurité

Exemple n°3.3 ▶ Réaliser une interface utilisateur web dynamique : IMC

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

Ici c'est une petite interface qui permet de calculer l'IMC d'une personne, aussi en javascript pur. J'ai d'abord codé l'html et le css : deux inputs pour pouvoir y insérer la taille et le poids de la personne, un bouton pour traiter les données et calculer l'IMC, et une zone d'affichage du résultat.

Viens ensuite le travail sur le javascript :

Tout d'abord, un tableau `BMIData` est défini avec les différentes catégories de poids et les plages d'IMC correspondantes. Chaque objet dans le tableau contient le nom de la catégorie, la couleur associée et la plage d'IMC.

Ensuite, le formulaire est sélectionné à l'aide de `querySelector` et un événement `submit` est ajouté avec `addEventListener` pour déclencher la fonction `handleForm` lorsque l'utilisateur soumet le formulaire. Lorsque l'utilisateur soumet le formulaire, la fonction `handleForm` est appelée pour empêcher la soumission par défaut, puis la fonction `calculateBMI` est appelée.

La fonction `calculateBMI` récupère la taille et le poids à partir des deux éléments `input` dans le formulaire, et calcule l'IMC en utilisant la formule standard. Il effectue également une vérification pour s'assurer que la taille et le poids sont corrects et qu'ils ne sont pas nuls ou négatifs.

Si les données sont correctes, la fonction `showResult` est appelée pour trouver la catégorie de poids correspondante à l'IMC calculé en utilisant la méthode `find()` et en vérifiant si l'IMC est compris dans les plages définies pour chaque catégorie.

Enfin, la fonction `showResult` met à jour l'affichage HTML avec l'IMC calculé, la couleur et le nom de la catégorie de poids correspondante. Si les données ne sont pas correctes, la fonction `handleError` est appelée pour afficher un message d'erreur.

2. Précisez les moyens utilisés :

Pour créer cette interface de calcul d'IMC en JavaScript, plusieurs moyens ont été utilisés :

1. HTML : Une page web a été créée avec des éléments HTML pour afficher l'interface utilisateur, notam-

DOSSIER PROFESSIONNEL (DP)

ment un formulaire avec des champs de saisie pour la taille et le poids, ainsi qu'un bouton de soumission.

2. CSS : Des styles CSS ont été appliqués à la page pour créer une interface utilisateur attrayante et conviviale.
3. JavaScript : Le langage JavaScript a été utilisé pour ajouter des fonctionnalités à la page. Les principaux éléments JavaScript utilisés dans ce code sont :
 - `document.querySelector()` et `document.querySelectorAll()`, qui sont utilisés pour sélectionner des éléments HTML dans la page et leur ajouter des écouteurs d'événements.
 - `Math.pow()` qui est utilisé pour calculer la puissance de la taille en mètre carré.
 - Les méthodes `toFixed()` et `find()` sont utilisées pour calculer l'IMC et trouver la catégorie de poids correspondante.
 - Les méthodes `textContent` et `style` sont utilisées pour mettre à jour le contenu HTML et les styles CSS de certains éléments dans la page.

En utilisant ces moyens, cette interface de calcul d'IMC peut fonctionner correctement et fournir une réponse rapide à l'utilisateur.

3. Avec qui avez-vous travaillé ?

J'ai travaillé seul avec de la documentation et des tutoriels.

4. Contexte

Nom de l'entreprise, organisme ou association

Cliquez ici pour taper du texte.

DOSSIER PROFESSIONNEL (DP)

Activité-type 1 Développer la partie FRONT-END d'une application web ou web mobile en intégrant les recommandations de sécurité

Exemple n°4.1 ▶ Réaliser une interface utilisateur avec une gestion de contenu ou e-commerce : Boutique Wordpress

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

Choisir un thème WordPress approprié pour la boutique en ligne. Les thèmes WordPress sont des modèles graphiques qui déterminent l'apparence et la mise en page de la boutique en ligne. Il existe de nombreux thèmes WordPress disponibles, certains gratuits et d'autres payants, qui peuvent être adaptés aux besoins spécifiques de la boutique en ligne.

Personnaliser le thème WordPress pour s'adapter aux besoins de la boutique en ligne. Une fois le thème WordPress choisi, il peut être nécessaire de personnaliser certains éléments pour mieux correspondre à l'identité visuelle de la boutique en ligne. Cela peut inclure la modification des couleurs, des polices, des images, des entêtes, des pieds de page, etc.

Ajouter des images de produits attrayantes et des descriptions détaillées. Les images de produits sont un élément clé de la boutique en ligne, car elles aident les clients à visualiser les produits qu'ils achètent. Les descriptions de produits doivent être claires et détaillées, afin que les clients sachent exactement ce qu'ils achètent.

Créer des pages de catégorie et de produit claires et organisées. Les pages de catégorie et de produit sont essentielles pour aider les clients à naviguer dans la boutique en ligne et à trouver rapidement ce qu'ils recherchent. Les pages de catégorie doivent être organisées de manière logique et facile à comprendre, tandis que les pages de produits doivent fournir des informations détaillées sur chaque produit.

Assurer une navigation claire et facile à utiliser. La navigation est un élément clé de toute boutique en ligne, car elle permet aux clients de trouver rapidement ce qu'ils recherchent. La navigation doit être organisée de manière logique et facile à comprendre, avec des menus clairs et des liens évidents vers les pages importantes.

Tester le site pour s'assurer que toutes les pages et tous les éléments fonctionnent correctement. Une fois que la boutique en ligne a été conçue et personnalisée, il est important de la tester pour s'assurer que toutes les pages et tous les éléments fonctionnent correctement et que l'expérience utilisateur est fluide.

En général, la création d'une boutique en ligne attrayante et efficace sur WordPress nécessite une bonne planification, une attention aux détails et des compétences en conception graphique. Il est également important de s'assurer que le site est facile à naviguer et qu'il offre une expérience utilisateur agréable et transparente pour les clients.

2. Précisez les moyens utilisés :

Je me suis servi d'un hébergement web pour stocker les fichiers de votre site et les rendre accessibles sur

DOSSIER PROFESSIONNEL (DP)

internet.

J'ai utilisé le très célèbre système de gestion de contenu open-source Wordpress car il est facile à utiliser, même pour les personnes sans compétences techniques en programmation ou en conception de sites web. Il dispose d'une interface intuitive qui permet de gérer facilement le contenu d'un site.

Il est aussi très flexible et peut être utilisé pour créer différents types de sites web. Il offre une grande variété de thèmes et de plugins qui permettent de personnaliser facilement sa boutique en ligne et d'ajouter des fonctionnalités spécifiques à nos besoins.

Wordpress est aussi soutenu par une grande communauté de développeurs et d'utilisateurs qui contribuent régulièrement à son développement et à son amélioration. Cela signifie que l'on peut trouver de l'aide et des ressources en ligne pour résoudre des problèmes.

3. Avec qui avez-vous travaillé ?

Nous avons travaillé avec notre formateur et seul avec de la documentation et des tutoriels.

4. Contexte

Nom de l'entreprise, organisme ou association

Cliquez ici pour taper du texte.

Activité-type 1 Développer la partie FRONT-END d'une application web ou

DOSSIER PROFESSIONNEL (DP)

web mobile en intégrant les recommandations de sécurité

Exemple n°4.2 ▶ Réaliser une interface utilisateur avec une gestion de contenu ou e-commerce : Boutique Symfony

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

Ici je n'ai réalisé que le squelette de la page d'accueil d'une boutique sous le framework Symfony.

Je n'ai pas eu besoin de faire une maquette car j'ai pu trouver un modèle.

Symfony fonctionne avec le système de vues Twig.

Pour mettre en place les vues Twig dans ma boutique e-commerce Symfony, j'ai dû effectuer plusieurs étapes. Tout d'abord, j'ai créé un contrôleur Symfony pour gérer la logique de mon application et récupérer les données à afficher à partir de mes différentes entités de base de données.

Ensuite, j'ai créé des fichiers Twig dans le dossier "templates" de mon application Symfony. Ces fichiers contiennent le code HTML et les instructions Twig nécessaires pour afficher mes données. J'ai utilisé des variables Twig pour afficher les données dynamiques dans mes fichiers Twig, qui sont transmises à partir du contrôleur Symfony via l'objet "Response" renvoyé par le contrôleur.

Pour charger des fichiers CSS ou JS dans mes fichiers Twig, j'ai utilisé des extensions Twig telles que "asset", et pour générer des URL de manière dynamique, j'ai utilisé "path".

En somme, les vues Twig dans Symfony m'ont permis de séparer clairement la logique de mon application de la présentation de mes données, en utilisant un système de templates flexible et puissant.

2. Précisez les moyens utilisés :

Pour créer le squelette de cette boutique j'ai utilisé Symfony et le système de vues Twig.

Symfony est un framework PHP open-source qui permet de créer des applications web de haute qualité et de grande envergure. Il fournit des outils et des fonctionnalités pour faciliter la création d'applications web, telles que la gestion de la configuration, la manipulation de bases de données, l'authentification et la sécurité, et bien plus encore.

Twig, quant à lui, est un moteur de templates pour PHP qui permet de séparer le code PHP de la logique de présentation HTML. Il permet de créer des templates plus facilement, de manière plus lisible et plus maintenable, en utilisant une syntaxe simplifiée et intuitive.

Pour mettre en place les vues Twig dans ma boutique e-commerce Symfony, j'ai utilisé les outils fournis par le framework Symfony lui-même. J'ai utilisé l'interface en ligne de commande Symfony (CLI) pour générer les fichiers de base nécessaires, tels que le contrôleur et les fichiers Twig de base. J'ai également utilisé Composer, le gestionnaire de dépendances de PHP, pour installer les bibliothèques Twig et Symfony Twig Bundle.

J'ai également utilisé un navigateur web pour tester mes vues Twig en temps réel, en utilisant le serveur de développement intégré de Symfony.

Enfin, j'ai utilisé une base de données MySQL pour stocker mes données d'application, que j'ai manipulées à

DOSSIER PROFESSIONNEL (DP)

travers des requêtes SQL et des requêtes Doctrine dans mon contrôleur Symfony.

3. Avec qui avez-vous travaillé ?

J'ai travaillé seul avec de la documentation et des tutoriels.

4. Contexte

Nom de l'entreprise, organisme ou association

Cliquez ici pour taper du texte.

Activité-type 2 Développer la partie BACK-END d'une application web ou web mobile en intégrant les recommandations de sécurité

Exemple n°5.1 ▶ Création d'une base de données : BD locale un site Wordpress

DOSSIER PROFESSIONNEL (DP)

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

J'ai créé une base de données locale pour un site WordPress en utilisant un logiciel appelé XAMPP. XAMPP est un ensemble d'outils qui permettent de créer un environnement de développement PHP/MySQL local sur votre ordinateur.

Une fois que j'ai installé XAMPP sur mon ordinateur, j'ai lancé le logiciel et j'ai démarré les modules Apache et MySQL. Ensuite, j'ai ouvert phpMyAdmin, qui est un outil de gestion de bases de données web, et j'ai créé une nouvelle base de données pour mon site WordPress.

J'ai ensuite téléchargé et installé WordPress sur mon ordinateur en utilisant XAMPP. Pendant le processus d'installation de WordPress, j'ai renseigné les informations de connexion à la base de données locale que j'avais créée précédemment. Cela a permis à WordPress de se connecter à la base de données et de stocker les données de mon site web.

Une fois que j'ai terminé la configuration de la base de données, j'ai pu commencer à développer et personnaliser mon site WordPress localement sur mon ordinateur.

2. Précisez les moyens utilisés :

Pour créer une base de données locale pour mon site WordPress, j'ai utilisé le logiciel XAMPP. Ce logiciel m'a permis de créer un environnement de développement local en incluant une pile de serveur web (Apache), une base de données (MySQL) et un langage de script côté serveur (PHP). J'ai téléchargé XAMPP depuis leur site web, puis j'ai installé le logiciel sur mon ordinateur. J'ai ensuite configuré les paramètres de la base de données MySQL pour créer ma base de données locale pour mon site WordPress. Ce processus a été facile à suivre et m'a permis de travailler sur mon site WordPress en local avant de le publier en ligne.

3. Avec qui avez-vous travaillé ?

Nous avons travaillé avec notre formateur puis seuls.

4. Contexte

Nom de l'entreprise, organisme ou association

Cliquez ici pour taper du texte.

DOSSIER PROFESSIONNEL (DP)

Activité-type 2 Développer la partie BACK-END d'une application web ou web mobile en intégrant les recommandations de sécurité

Exemple n°5.2 ▶ Création d'une base de données : MongoDBCompass

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

Pour créer une base de données avec MongoDB Compass, j'ai effectué plusieurs tâches ou opérations dans les conditions suivantes :

1. J'ai téléchargé et installé MongoDB Compass sur mon ordinateur.
2. Ensuite, j'ai créé une connexion à un serveur MongoDB en spécifiant l'adresse IP du serveur, le port et les informations d'identification.
3. J'ai créé une nouvelle base de données en cliquant sur le bouton "Create Database" dans MongoDB Compass. J'ai donné un nom à la base de données et j'ai spécifié les options de configuration, telles que les options de stockage et les autorisations d'accès.
4. J'ai créé des collections pour stocker les données en cliquant sur le bouton "Create Collection" dans MongoDB Compass. J'ai spécifié le nom de la collection, les options de configuration et les schémas de données.
5. J'ai ajouté des données à ma base de données en utilisant l'interface de MongoDB Compass. J'ai cliqué sur le bouton "Insert Document" pour ajouter des données à mes collections.

Dans l'ensemble, j'ai trouvé que la création d'une base de données avec MongoDB Compass était facile et intuitive. L'interface utilisateur était conviviale et m'a permis de créer et de gérer mes bases de données MongoDB sans avoir à utiliser une ligne de commande.

2. Précisez les moyens utilisés :

Pour créer une base de données avec MongoDB Compass, j'ai utilisé le logiciel MongoDB Compass qui est spécialement conçu pour interagir avec les bases de données MongoDB. Avant de commencer, j'ai téléchargé et installé MongoDB Compass sur mon ordinateur.

Ensuite, j'ai créé une connexion à un serveur MongoDB en spécifiant l'adresse IP du serveur, le port et les informations d'identification. Cela m'a permis d'accéder à MongoDB et de créer ma base de données en utilisant l'interface graphique de MongoDB Compass.

DOSSIER PROFESSIONNEL (DP)

MongoDB est une base de données NoSQL open source, qui utilise un modèle de données basé sur des documents plutôt que des tables pour stocker les données. Les documents sont stockés sous forme de paires clé-valeur, ce qui offre une grande flexibilité et une capacité de stockage évolutive.

En résumé, pour créer une base de données avec MongoDB Compass, j'ai utilisé un logiciel spécialement conçu pour interagir avec les bases de données MongoDB, qui est MongoDB Compass. Et j'ai utilisé MongoDB comme système de gestion de base de données, qui utilise un modèle de données basé sur des documents pour stocker les données.

3. Avec qui avez-vous travaillé ?

J'ai travaillé seul avec de la documentation et des tutoriels.

4. Contexte

Nom de l'entreprise, organisme ou association

Cliquez ici pour taper du texte.

DOSSIER PROFESSIONNEL (DP)

Activité-type 2 Développer la partie BACK-END d'une application web ou web mobile en intégrant les recommandations de sécurité

Exemple n°6 ► Développer les composants d'accès aux données : site Javascript

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

Pour développer les composants d'accès aux données de mon application CRUD en Node.js, j'ai effectué plusieurs tâches ou opérations, telles que :

- Concevoir le modèle de données : j'ai défini les entités que mon application doit stocker (dans ce cas, les livres), ainsi que les attributs de chaque entité (par exemple, titre, auteur, date de publication, etc.). J'ai utilisé un schéma de base de données pour modéliser mon modèle de données.
- Configurer la base de données : j'ai choisi une base de données pour stocker les données de mon application. J'ai configuré mon application pour qu'elle puisse se connecter à cette base de données en utilisant une bibliothèque de connexion de base de données telle que Mongoose. J'ai installé les modules Mongoose et Morgan.
- Écrire les requêtes de base de données : j'ai écrit des requêtes pour effectuer les opérations CRUD sur ma base de données (par exemple, insérer un nouveau livre, récupérer tous les livres, mettre à jour un livre existant, supprimer un livre). J'ai utilisé un ORM ou une bibliothèque de requêtes de base de données pour simplifier ce processus.
- Créer les endpoints de l'API : j'ai créé des endpoints HTTP pour permettre à l'application d'interagir avec la base de données. J'ai utilisé un framework tel que Express.js pour gérer ces endpoints.
- Tester et déboguer : j'ai testé chaque opération CRUD pour m'assurer qu'elle fonctionne correctement. J'ai débogué mon code pour corriger les erreurs et les bugs qui ont été découverts.

2. Précisez les moyens utilisés :

DOSSIER PROFESSIONNEL (DP)

Pour développer les composants d'accès aux données de mon application CRUD en Node.js, j'ai utilisé plusieurs moyens :

- Pour concevoir le modèle de données, j'ai utilisé la bibliothèque Mongoose qui permet de définir des schémas pour les données stockées dans MongoDB, la base de données que j'ai choisie pour mon application.
- Pour configurer la base de données, j'ai utilisé le module 'mongoose' de Node.js pour établir une connexion à la base de données MongoDBCompass en utilisant l'URI fourni.
- Pour écrire les requêtes de base de données, j'ai utilisé les méthodes fournies par Mongoose pour effectuer les opérations CRUD (create, read, update, delete) sur les entités de mon application (les livres).
- Pour créer les endpoints de l'API, j'ai utilisé le framework Express.js pour définir les routes HTTP qui permettent à l'application d'interagir avec la base de données via les méthodes Mongoose CRUD que j'ai implémentées.
- Débogage manuel du code

3. Avec qui avez-vous travaillé ?

J'ai travaillé seul avec de la documentation et des tutoriels.

4. Contexte

Nom de l'entreprise, organisme ou association

Cliquez ici pour taper du texte.

Activité-type 2 Développer la partie BACK-END d'une application web ou web mobile en intégrant les recommandations de sécurité

Exemple 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 :

J'ai importé les modules nécessaires pour le bon fonctionnement de mon application, tels que express, mongoose, body-parser et session. Ensuite, j'ai configuré la session en utilisant la méthode server.use() avec les options secret, resave, saveUninitialized et cookie, ce qui m'a permis de stocker les données utilisateur et de garder leur état lorsqu'ils naviguent sur l'application.

J'ai également configuré la base de données MongoDB en utilisant la méthode mongoose.connect() pour me connecter à la base de données en utilisant l'URL de connexion. Ensuite, j'ai configuré le serveur Express en utilisant la méthode server.use() pour spécifier les middleware, tels que express.static(), morgan(), bodyParser.urlencoded(), et en utilisant la méthode server.set() pour définir l'option trust proxy.

Pour gérer les messages utilisateur, j'ai créé une fonction middleware qui stocke les messages utilisateur dans la session et les supprime une fois qu'ils ont été affichés à l'utilisateur. Enfin, j'ai défini les routes dans le fichier routeur.js en utilisant la méthode server.use(). Les routes sont les points d'entrée de l'application où les utilisateurs peuvent interagir avec l'application.

En somme, j'ai configuré le serveur et la base de données pour qu'ils puissent fonctionner ensemble et j'ai défini les points d'entrée de l'application pour permettre aux utilisateurs d'interagir avec mon application.

2. Précisez les moyens utilisés :

- Pour l'importation des modules nécessaires, j'ai utilisé la syntaxe "require('nom-du-module')" pour chaque module dont j'avais besoin.
- Pour la configuration de la session, j'ai utilisé la méthode "server.use()" avec les options "secret", "resave", "saveUninitialized" et "cookie" pour stocker les données utilisateur et garder leur état lorsqu'ils naviguent sur l'application. J'ai également installé le module "express-session" à l'aide de la commande "npm install express-session".
- Pour la configuration de la base de données MongoDB, j'ai utilisé la méthode "mongoose.connect()" pour me connecter à la base de données en utilisant l'URL de connexion. J'ai également installé le module "mongoose" à l'aide de la commande "npm install mongoose".
- Pour la configuration du serveur Express, j'ai utilisé la méthode "server.use()" pour spécifier les middleware tels que "express.static()", "morgan()", "bodyParser.urlencoded()" et j'ai utilisé la méthode "server.set()" pour définir l'option "trust proxy". J'ai également installé les modules correspondants à l'aide des commandes "npm install express", "npm install morgan" et "npm install body-parser".

DOSSIER PROFESSIONNEL (DP)

- Pour la gestion des messages utilisateur, j'ai créé une fonction middleware qui stocke les messages utilisateur dans la session et les supprime une fois qu'ils ont été affichés à l'utilisateur.

- Pour la définition des routes, j'ai utilisé la méthode "server.use()" dans le fichier "routeur.js" pour définir les différentes routes que l'application doit gérer.

Dans l'ensemble, j'ai utilisé différents modules et packages disponibles sur le gestionnaire de paquets npm pour faciliter le développement de la partie back-end de l'application.

3. Avec qui avez-vous travaillé ?

J'ai travaillé seul avec de la documentation et des tutoriels.

4. Contexte

Nom de l'entreprise, organisme ou association

Cliquez ici pour taper du texte.

DOSSIER PROFESSIONNEL (DP)

Activité-type 2 Développer la partie BACK-END d'une application web ou web mobile en intégrant les recommandations de sécurité

Exemple n°8 ▶ *Élaborer et mettre en œuvre des composants dans une application de gestion de contenu*

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

Dans le cadre de l'élaboration et de la mise en oeuvre des composants d'une application de gestion de contenu, j'ai effectué plusieurs tâches et opérations. Tout d'abord, j'ai analysé les besoins fonctionnels et techniques de l'application afin de déterminer les composants nécessaires et les fonctionnalités à implémenter.

Ensuite, j'ai choisi les technologies les plus adaptées à ces besoins, en prenant en compte les contraintes du projet, telles que le budget, les délais, les compétences de l'équipe et les exigences de performance et de sécurité.

J'ai ensuite procédé à la conception des différents composants de l'application, en utilisant des méthodes de modélisation et de conception appropriées, telles que les diagrammes de classes, les diagrammes de séquence et les diagrammes de déploiement.

Une fois la conception terminée, j'ai commencé à implémenter les différents composants de l'application, en utilisant les technologies choisies et en respectant les bonnes pratiques de développement. J'ai également effectué des tests unitaires et des tests d'intégration pour m'assurer que les composants fonctionnaient correctement et qu'ils répondaient aux exigences spécifiées.

Enfin, j'ai déployé l'application sur un serveur ou sur un service de cloud, en prenant en compte les exigences de disponibilité et de performance de l'application. J'ai également mis en place des mécanismes de surveillance et de gestion de l'application, afin de détecter les erreurs et les dysfonctionnements et de les corriger rapidement.

2. Précisez les moyens utilisés :

DOSSIER PROFESSIONNEL (DP)

Pour élaborer et mettre en œuvre les différents composants de l'application de gestion de contenu, j'ai utilisé plusieurs moyens.

Tout d'abord, j'ai utilisé des langages de programmation tels que JavaScript, HTML et CSS pour développer les différentes fonctionnalités de l'application.

J'ai également utilisé plusieurs frameworks et bibliothèques pour faciliter le développement, tels que Node.js, Express, React et Redux. Node.js m'a permis de développer la partie back-end de l'application en utilisant JavaScript, tandis qu'Express m'a aidé à gérer les routes et les middlewares. React et Redux ont été utilisés pour développer la partie front-end de l'application en permettant de gérer efficacement l'état global de l'application et de construire des interfaces utilisateur dynamiques et réactives.

J'ai également utilisé une base de données MongoDB pour stocker les données de l'application, en utilisant le module Mongoose pour faciliter la manipulation des données.

3. Avec qui avez-vous travaillé ?

J'ai travaillé seul avec de la documentation et des tutoriels.

4. Contexte

Nom de l'entreprise, organisme ou association

Cliquez ici pour taper du texte.

DOSSIER PROFESSIONNEL (DP)

Titres, diplômes, CQP, attestations de formation

(facultatif)

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

DOSSIER PROFESSIONNEL (DP)

Déclaration sur l'honneur

Je soussigné(e) [prénom et nom] **David Impinna**,

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 **Maisons Alfort**

le **11/04/2023**

pour faire valoir ce que de droit.

Signature :

DOSSIER PROFESSIONNEL (DP)

ANNEXE

Intitulé

Cliquez ici pour taper du texte.

DOSSIER PROFESSIONNEL (DP)

DOSSIER PROFESSIONNEL (DP)

ANNEXE 1 : Maquetter une application avec Figma.....

The collage consists of four screenshots from the Figma interface:

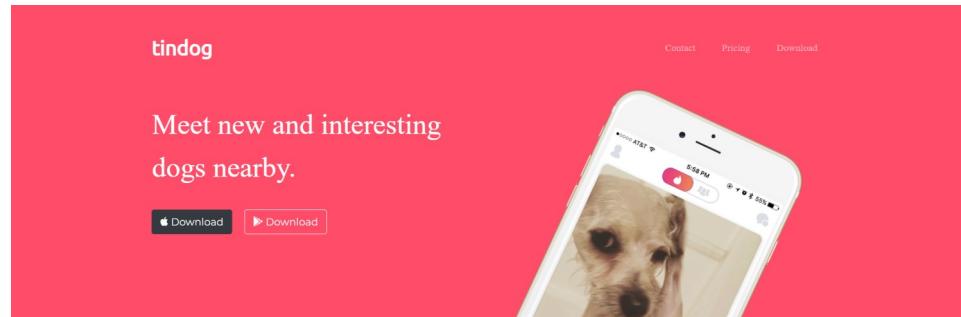
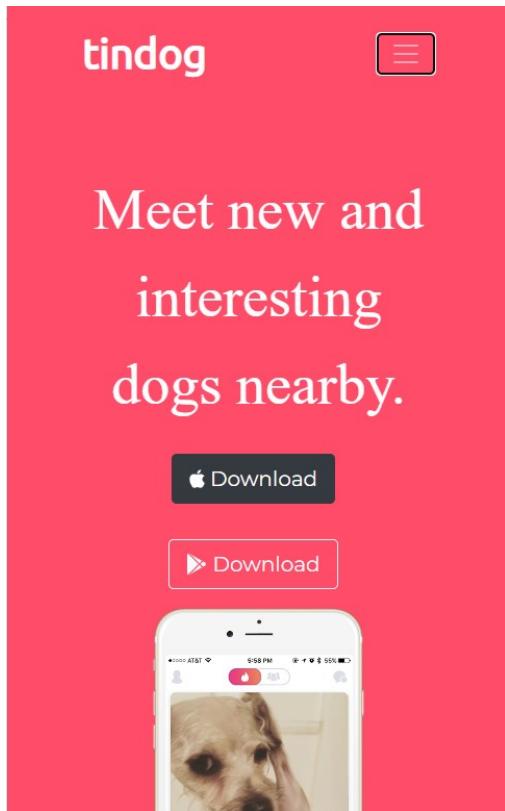
- Screenshot 1:** Shows two wireframes side-by-side: a "Welcome" screen with a "Connexion" button and an "Inscription" screen with fields for "Adresse mail:", "Mot de passe:", and "Confirmation:", followed by a "S'inscrire" button.
- Screenshot 2:** Shows a wireframe for an "Accès/inscription" page with sections for "Inscription", "Mot de passe:", "Confirmation:", and "S'inscrire". The background is light blue.
- Screenshot 3:** A hand-drawn sketch on lined paper showing three products labeled "Product 1", "Product 2", and "Product 3", each with a wavy line underneath. Below them are three boxes labeled "Details".
- Screenshot 4:** Shows a wireframe for a "Social Media Templates" page featuring various Instagram template components like "Instagram Square Image", "Instagram Stories", and "Instagram Stories Components".
- Screenshot 5:** A wireframe for a "Maquette" (mockup) page. It includes a navigation sidebar with categories like "Pages", "Wireframes", "Maquette", and "Identité-graphique". The main area displays three wireframes for "Accueil", "Menu", and "Contact". The "Accueil" wireframe shows a grid of cards. The right sidebar contains styling tools for "Code", "Text Styles", and "Color Styles".

DOSSIER PROFESSIONNEL (DP)

ANNEXE 2 : Réaliser une interface web utilisateur statique et adaptable : Site vitrine « *Tindog* »

```

8  <!-- Google Fonts -->
9   <link href="https://fonts.googleapis.com/css?family=Montserrat|Ubuntu" rel="stylesheet">
10 
11 <!-- CSS Stylesheets -->
12 <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css" integrity="sha384-GvHaXgBTBfXKXHn1OxWnDZP9yjwEGLGmCJ1SAwIC" crossorigin="anonymous">
13 <link rel="stylesheet" href="css/styles.css">
14 
15 <!-- Font Awesome -->
16 <script defer src="https://use.fontawesome.com/releases/v5.0.7/js/all.js"></script>
17 
18 <!-- Bootstrap Scripts -->
19 <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js" integrity="sha384-KJ3o2DKtIkYIK3UENmM7KCkR/rE9/Opg6aAZGJwFDM/NA/GpGFF93hXpG5KkN" crossorigin="anonymous">
20 <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js" integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUiBXR9j7fakFPskvXusvfa0b4" crossorigin="anonymous">
21 <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js" integrity="sha384-JZR6Spejh4U0d8d8j0t6vLEHfe/JQGiRRSQQxSFFWpi1MquVdAyjUar5+76PVCmYl" crossorigin="anonymous">
22 </head>
```



```

26 <section class="colored-section" id="title">
27   <div class="container-fluid">
28     <!-- Nav Bar -->
29 
30   <nav class="navbar navbar-expand-lg navbar-dark">
31     <a class="navbar-brand" href="#">tindog
32     <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarTogglerDemo02">
33       <span class="navbar-toggler-icon"></span>
34     </button>
35 
36     <div class="collapse navbar-collapse" id="navbarTogglerDemo02">
37       <ul class="navbar-nav ml-auto">
38         <li class="nav-item">
39           <a class="nav-link" href="#footer">Contact</a>
40         </li>
41         <li class="nav-item">
42           <a class="nav-link" href="#pricing">Pricing</a>
43         </li>
44         <li class="nav-item">
45           <a class="nav-link" href="#cta">Download</a>
46         </li>
47       </ul>
48     </div>
49   </nav>
```

```

45   /* Navigation Bar */
46 
47   .navbar {
48     padding: 0 0 4.5rem;
49   }
50 
51   .navbar-brand {
52     font-family: "Ubuntu";
53     font-size: 2.5rem;
54     font-weight: bold;
55   }
56 
57   .nav-item {
58     padding: 0 18px;
59   }
60 
61   .nav-link {
62     font-size: 1.2rem;
63     font-family: "Montserrat-Light";
64   }
65 
66   /* Buttons */
67 
68   .download-button {
69     margin: 5% 3% 5% 0;
70   }
```

```

80   #title .container-fluid {
81     padding: 3% 15% 7%;
82   }
83 
84   /* Title Image */
85 
86   .title-image {
87     width: 60%;
88     transform: rotate(25deg);
89     position: absolute;
90     right: 30%;
91   }
92 
93   /* Features Section */
94 
95   #features {
96     position: relative;
97   }
98 
99   .feature-title {
100    font-size: 1.5rem;
101  }
102 
103   .feature-box {
104    padding: 4.5%;
105  }
106 
107   .icon {
108     color: #ef8172;
109     margin-bottom: 1rem;
110   }
111 
112   .icon:hover {
113     color: #ff4c68;
114   }
```

DOSSIER PROFESSIONNEL (DP)

I no longer have to sniff other dogs for love. I've found the hottest Corgi on TinDog. Woof.



Pebbles, New York

TechCrunch

TNW
THE NEXT WEB

BUSINESS INSIDER

Mashable

```
110 <!-- Testimonials -->
111
112 <section class="colored-section" id="testimonials">
113
114   <div id="testimonial-carousel" class="carousel slide" data-ride="false">
115     <div class="carousel-inner">
116       <div class="carousel-item active container-fluid">
117         <h2 class="testimonial-text">I no longer have to sniff other dogs for love. I've found the hottest Corgi on TinDog. Woof.</h2>
118         
119         <em>Pebbles, New York</em>
120       </div>
121       <div class="carousel-item container-fluid">
122         <h2 class="testimonial-text">My dog used to be so lonely, but with TinDog's help, they've found the love of their life. I think.</h2>
123         
124         <em>Beverly, Illinois</em>
125       </div>
126     </div>
127     <a class="carousel-control-prev" href="#testimonial-carousel" role="button" data-slide="prev">
128       <span class="carousel-control-prev-icon"></span>
129     </a>
130     <a class="carousel-control-next" href="#testimonial-carousel" role="button" data-slide="next">
131       <span class="carousel-control-next-icon"></span>
132     </a>
133   </div>
134 </section>
```

A Plan for Every Dog's Needs

Simple and affordable price plans for you and your dog.

Chihuahua	Labrador
Free	\$49 / mo
5 Matches Per Day	Unlimited Matches
10 Messages Per Day	Unlimited Messages
Unlimited App Usage	Unlimited App Usage
Sign Up	Sign Up

```
116 /* Testimonial Section */
117
118 #testimonials {
119   background-color: #ef8172;
120 }
121
122 .testimonial-text {
123   font-size: 3rem;
124   line-height: 1.5;
125 }
126
127 .testimonial-image {
128   width: 10%;
129   border-radius: 100%;
130   margin: 20px;
131 }
132
133 #press {
134   background-color: #ef8172;
135   padding-bottom: 3%;
136 }
137
138 .press-logo {
139   width: 15%;
140   margin: 20px 20px 50px;
141 }
```

Mastiff
\$99 / mo
Priority Listing
Unlimited Matches
Unlimited Messages
Unlimited App Usage
Sign Up

A Plan for Every Dog's Needs

Simple and affordable price plans for you and your dog.

Chihuahua	Labrador	Mastiff
Free	\$49 / mo	\$99 / mo
5 Matches Per Day	Unlimited Matches	Priority Listing
10 Messages Per Day	Unlimited Messages	Unlimited Matches
Unlimited App Usage	Unlimited App Usage	Unlimited Messages
Sign Up	Sign Up	Sign Up

DOSSIER PROFES

Page

DOSSIER PROFESSIONNEL (DP)

A Plan for Every Dog's Needs

Simple and affordable price plans for your and your dog.

Chihuahua

Free

5 Matches Per Day
10 Messages Per Day
Unlimited App Usage

[Sign Up](#)

Labrador

\$49 / mo

Unlimited Matches
Unlimited Messages
Unlimited App Usage

[Sign Up](#)

Mastiff

\$99 / mo

Priority Listing
Unlimited Matches
Unlimited Messages
Unlimited App Usage

[Sign Up](#)

```
149 <!-- Pricing -->
150
151 <section class="white-section" id="pricing">
152
153 <h2 class="section-heading">A Plan for Every Dog's Needs</h2>
154 <p>Simple and affordable price plans for your and your dog.</p>
155
156 <div class="row">
157
158 <div class="pricing-column col-lg-4 col-md-6">
159   <div class="card">
160     <div class="card-header">
161       <h3>Chihuahua</h3>
162     </div>
163     <div class="card-body">
164       <h2 class="price-text">Free</h2>
165       <p>5 Matches Per Day</p>
166       <p>10 Messages Per Day</p>
167       <p>Unlimited App Usage</p>
168       <button class="btn btn-lg btn-block btn-outline-dark" type="button">Sign Up</button>
169     </div>
170   </div>
171 </div>
172
173 <div class="pricing-column col-lg-4 col-md-6">
174   <div class="card">
175     <div class="card-header">
176       <h3>Labrador</h3>
177     </div>
178     <div class="card-body">
179       <h2 class="price-text">$49 / mo</h2>
180       <p>Unlimited Matches</p>
181       <p>Unlimited Messages</p>
182       <p>Unlimited App Usage</p>
183       <button class="btn btn-lg btn-block btn-dark" type="button">Sign Up</button>
184     </div>
185   </div>
186 </div>
```

Find the True
Love of Your
Dog's Life
Today.

 Download

 Download



© Copyright 2018 TinDog

```
212 <!-- Call to Action -->
213
214 <section class="colored-section" id="cta">
215
216   <div class="container-fluid">
217
218     <h3 class="big-heading">Find the True Love of Your Dog's Life Today.</h3>
219     <button class="download-button btn btn-lg btn-dark" type="button"><i class="fab fa-apple"></i> Download</button>
220     <button class="download-button btn btn-lg brn-light" type="button"><i class="fab fa-google-play"></i> Download</button>
221   </div>
222 </section>
223
224
225 <!-- Footer -->
226
227 <footer class="white-section" id="footer">
228   <div class="container-fluid">
229     <i class="social-icon fab fa-facebook-f"></i>
230     <i class="social-icon fab fa-twitter"></i>
231     <i class="social-icon fab fa-instagram"></i>
232     <i class="social-icon fas fa-envelope"></i>
233     <p>© Copyright 2018 TinDog</p>
234   </div>
235 </footer>
236
237 </body>
```

```
162 /* Footer Section */
163
164
165 .social-icon {
166   margin: 20px 10px;
167 }
168
169 @media (max-width: 1028px) {
170
171   #title {
172     text-align: center;
173   }
174
175   .title-image {
176     position: static;
177     transform: rotate(0);
178   }
179 }
```

DOSSIER PROFESSIONNEL

DOSSIER PROFESSIONNEL (DP)

ANNEXE 3.1 : Réaliser une interface utilisateur web dynamique : Personnage jeux vidéos

Créateur de personnage

Nom :

Points restants : 7
Force +
Agilité +
Intelligence +

Réinitialiser Crée

Créateur de personnage

Nom : Bidule

Points restants : 1
Force ➤ +
Agilité +
Intelligence ➤ +

epee fleau arc hache

Réinitialiser Crée

Créateur de personnage

Points restants : 7
Force +
Agilité +
Intelligence +

epee fleau arc hache

Réinitialiser Crée

- Voici les composants principaux que nous allons créer dans l'application :
 - Titre
 - Personnage
 - Armes
 - Bouton

- Les states principaux seront :
 - Personnage :
 - Image, force, agilité, intelligence et arme
 - nombre de points de caractéristique
 - Liste des armes

DOSSIER PROFESSIONNEL (DP)

```
✓ Projet2
  > node_modules
  > public
  ✓ src
    ✓ assets\images
      > armes
      > persos
    ✓ components
      > Bouton
      > Titres
      > containers
    JS App.js
    JS index.js
    ♦ .gitignore
  {} package-lock.json
  {} package.json
```

JS index.js

```
Projets React > Projet2 > src > JS index.js
1 import React from 'react';
2 import ReactDOM from 'react-dom';
3 import App from './App';
4
5 ReactDOM.render(
6   <React.StrictMode>
7     | <App />
8   </React.StrictMode>,
9   document.getElementById('root')
10 );
```

```
✓ containers
  ✓ CreateurPerson...
    ✓ Armes
      ✓ Arme
        JS Arme.js
        JS Armes.js
    ✓ Personnage
      ✓ CaracPerso
        ✓ Carac
          JS Carac.js
          # Carac.mod...
          JS CaracPerso.js
      ✓ ImagePerso
        JS ImagePerso.js
        # ImagePerso...
        JS Personnage.js
    JS CreateurPerso...
  ✓ ListePersonnage
    ✓ Personnage
      JS Personnage.js
      JS ListePersonna...
JS App.js
JS index.js
```

JS App.js

```
Projets React > Projet2 > src > JS App.js > ...
1 import React, { Component } from 'react';
2 import CreateurPersonnage from "./containers/CreateurPersonnage/CreateurPersonnage";
3 import ListePersonnage from "./containers/ListePersonnage/ListePersonnage";
4
5 class App extends Component {
6   state = {
7     refresh:false
8   }
9
10  handleRefresh = () => {
11    this.setState({oldState} => {
12      return {
13        refresh: !oldState.refresh
14      }
15    });
16  }
17
18  render () {
19    return (
20      <>
21        <CreateurPersonnage refresh = {this.handleRefresh}/>
22        <ListePersonnage refresh={this.state.refresh}/>
23      </>
24    );
25  }
26}
27
28 export default App;
```

JS Armes.js

```
Projets React > Projet2 > src > containers > CreateurPersonnage > Armes > JS Armes.js > default
1 import React from "react";
2 import Arme from "./Arme/Arme";
3 import imgArc from "../../../../../assets/images/armes/arc.png";
4 import imgEpee from "../../../../../assets/images/armes/epee.png";
5 import imgFleau from "../../../../../assets/images/armes/fleau.png";
6 import imgHache from "../../../../../assets/images/armes/hache.png";
7
8 const armes = (props) => (
9   <div className="row no-gutters text-center">
10     {props.listeArmes.map(arame => {
11       let imgArme;
12       switch(arame){
13         case "arc" : imgArme=imgArc;
14         break;
15         case "epee" : imgArme=imgEpee;
16         break;
17         case "fleau" : imgArme=imgFleau;
18         break;
19         case "hache" : imgArme=imgHache;
20         break;
21       }
22       return (
23         <div className="col-3" key={arame}>
24           <Arme
25             imageArme={imgArme}
26             isCurrentArme={props.currentArme === arame}
27             clic={() => props.changeArme(arame)}
28             >{arame}</Arme>
29         </div>
30       );
31     })}
32   </div>
33 );
34
35 export default armes;
```

DOSSIER PROFESSIONNEL (DP)

```
JS CaracPerso.js ×
Projects React > Projet2 > src > containers > CreateurPersonnage > Personnage > CaracPerso > JS
1 import React from "react";
2 import Carac from "./Carac/Carac";
3
4 const caracPerso = (props) => (
5   <>
6     <div>
7       Points restants :
8         <span className="badge badge-success">
9           {props.nbPointsDisponibles}
10        </span>
11      </div>
12      <div>
13        <Carac
14          nbPoints={props.force}
15          moins={() => props.enleverPoint('force')}
16          plus={() => props.ajouterPoint('force')}
17        >Force</Carac>
18        <Carac nbPoints={props.agilite}
19          moins={() => props.enleverPoint('agilite')}
20          plus={() => props.ajouterPoint('agilite')}
21        >Agilite</Carac>
22        <Carac nbPoints={props.intelligence}
23          moins={() => props.enleverPoint('intelligence')}
24          plus={() => props.ajouterPoint('intelligence')}
25        >Intelligence</Carac>
26      </div>
27    </>
28  );
29
30 export default caracPerso;
```

```
JS ImagePerso.js ×
Projects React > Projet2 > src > containers > CreateurPersonnage > Personnage > ImagePerso > JS ImagePerso.js > default
1 import React from "react";
2 import ImagePerso1 from "../../../../../assets/images/persos/player1.png";
3 import ImagePerso2 from "../../../../../assets/images/persos/player2.png";
4 import ImagePerso3 from "../../../../../assets/images/persos/player3.png";
5
6 import classes from "./ImagePerso.module.css";
7
8 const imagePerso = (props) => {
9   let imageToPrint="";
10   if(props.numImage === 1) imageToPrint = ImagePerso1;
11   else if(props.numImage === 2) imageToPrint = ImagePerso2;
12   else if(props.numImage === 3) imageToPrint = ImagePerso3;
13
14   return (
15     <div className="row no-gutters text-center align-items-center">
16       <div className=[["col-1",classes.fleche,classes.gauche].join(' ')] onClick={props.flecheGauche}></div>
17       <div className="col">
18         <img src={imageToPrint} alt='perso'/>
19       </div>
20       <div className=[["col-1",classes.fleche,classes.droite].join(' ')] onClick={props.flecheDroite}></div>
21     </div>
22   );
23 };
24
25 export default imagePerso;
```

DOSSIER PROFESSIONNEL (DP)

ANNEXE 3.2 : Développer une interface utilisateur web dynamique : Quizz

Quizz Culture générale.

Qui est sacré empereur de France le 2 décembre 1804 ?

- Clovis
- Abraham Lincoln
- Napoléon Bonaparte

Quand la déclaration d'indépendance des Etats-Unis a-t-elle été votée ?

- 4 juillet 1776
- 18 avril 1856
- 30 juin 1925

Quand a eu lieu la chute de l'empire romain d'occident ?

- 15 ap. J.-C.
- 476 ap. J.-C.
- 740 av. J.-C.

Quelle est la capitale de la Slovénie ?

- Ljubljana
- Belgrade
- Bratislava

Combien d'habitants compte l'Irlande en 2020 ?

- 1,365 Millions
- 21 Millions
- 4,9 Millions

VALIDER

Il reste quelques erreurs.
Score : **2 / 5**
Retentez une autre réponse dans les cases rouges, puis re validez !

Quizz Culture générale.

Qui est sacré empereur de France le 2 décembre 1804 ?

- Clovis
- Abraham Lincoln
- Napoléon Bonaparte

Quand la déclaration d'indépendance des Etats-Unis a-t-elle été votée ?

- 4 juillet 1776
- 18 avril 1856
- 30 juin 1925

Quand a eu lieu la chute de l'empire romain d'occident ?

- 15 ap. J.-C.
- 476 ap. J.-C.
- 740 av. J.-C.

Quelle est la capitale de la Slovénie ?

- Ljubljana
- Belgrade
- Bratislava

Combien d'habitants compte l'Irlande en 2020 ?

- 1,365 Million
- 21 Millions
- 4,9 Millions

VALIDER

Bravo, c'est un sans faute !
Score : **5 / 5**
Quelle culture ...

```

14 <h1>
15   <span>Quizz</span>
16   Culture générale. 
17 </h1>
18
19 <div class="global-container">
20
21   <form class="quiz-form">
22     <div class="question-block">
23       <h4>Qui est sacré empereur de France le 2 décembre 1804 ?</h4>
24
25       <div>
26         <input type="radio" id="Clovis" name="q1" value="a" checked />
27         <label for="Clovis">Clovis</label>
28       </div>
29
30       <div>
31         <input type="radio" id="Lincoln" name="q1" value="b" />
32         <label for="Lincoln">Abraham Lincoln</label>
33       </div>
34
35       <div>
36         <input type="radio" id="Bonaparte" name="q1" value="c" />
37         <label for="Bonaparte">Napoléon Bonaparte</label>
38       </div>
39
40     </div>
41     <div class="question-block">
42       <h4>Quand la déclaration d'indépendance des Etats-Unis a-t-elle été votée ?</h4>
43
44       <div>
45         <input type="radio" id="date1" name="q2" value="a" checked />
46         <label for="date1">4 juillet 1776</label>
47       </div>
48
49       <div>
50         <input type="radio" id="date2" name="q2" value="b" />
51         <label for="date2">18 avril 1856</label>
52       </div>
53
54       <div>
55         <input type="radio" id="date3" name="q2" value="c" />
56         <label for="date3">30 juin 1925</label>
57       </div>
58
59     </div>
60     <div class="question-block">
61       <h4>Quand a eu lieu la chute de l'empire romain d'occident ?</h4>
62
63       <div>
64         <input type="radio" id="date4" name="q3" value="a" checked />
65         <label for="date4">15 ap. J.-C.</label>
66       </div>
67
68       <div>
69         <input type="radio" id="date5" name="q3" value="b" />
70         <label for="date5">476 ap. J.-C.</label>
71       </div>
72
73       <div>
74         <input type="radio" id="date6" name="q3" value="c" />
75         <label for="date6">-740 av. J.-C.</label>
76       </div>
77
78     </div>
79
80   </form>
81 </div>

```

DOSSIER PROFESSIONNEL (DP)

```
# style.css
Quizz > # style.css > question-block > div
1   *
2   :before,
3   :after {
4     box-sizing: border-box;
5     margin: 0;
6     padding: 0;
7   }
8
9 body {
10   min-height: 100vh;
11   font-family: Open sans, sans-serif;
12   background: linear-gradient(to right, #e0e0fc, #cfdef3);
13   padding: clamp(20px, 5vw, 50px) 20px 10px;
14 }
15
16 h1 {
17   font-family: Roboto, sans-serif;
18   font-size: 40px;
19   margin-bottom: 40px;
20   font-weight: 400;
21   letter-spacing: 2px;
22   text-align: center;
23   color: #0e0e0e;
24 }
25 h1 span {
26   font-weight: 800;
27 }
28 .global-container {
29   max-width: 800px;
30   margin: 0 auto;
31 }
32 .question-block {
33   padding: 25px;
34   margin: 20px 0;
35   border-radius: 5px;
36   background: #f1f1f1;
37   box-shadow: 0 5px 10px rgba(104,104,104,0.5);
38 }
39 .question-block h4 {
40   font-size: 24px;
41   font-weight: 500;
42   margin-bottom: 20px;
43 }
44 .question-block > div {
45   margin: 10px 0;
46   display: flex;
47   align-items: center;
48 }
49
50 .question-block label, .question-block input[type="radio"] {
51   cursor: pointer;
52 }
53 .question-block label {
54   font-size: 20px;
55 }
56 .question-block input[type="radio"] {
57   width: 20px;
58   height: 20px;
59 }
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
```

```
1 const responses = ["c", "a", "b", "a", "c"];
2 const emojis = ["✓", "★", "❀", "⌚", "⚠"];
3
4 const form = document.querySelector(".quiz-form");
5 form.addEventListener("submit", handleSubmit);
6
7 function handleSubmit(e) {
8   e.preventDefault();
9
10  const results = [];
11
12  const radioButtons = document.querySelectorAll("input[type='radio']:checked");
13
14  radioButtons.forEach((radioButton, index) => {
15    if (radioButton.value === responses[index]) {
16      results.push(true);
17    } else {
18      results.push(false);
19    }
20  });
21
22
23  showResults(results);
24  addColors(results);
25 }
26
27
28 const titleResult = document.querySelector(".results h2");
29 const markResult = document.querySelector(".mark");
30 const helpResult = document.querySelector(".help");
31
32 function showResults(results) {
33  const errorsNumber = results.filter(el => el === false).length;
34
35  console.log(errorsNumber);
36  switch (errorsNumber) {
37    case 0:
38      titleResult.textContent = `✓ Bravo, c'est un sans faute ! ✓`;
39      helpResult.textContent = "Quelle culture ...";
40      helpResult.style.display = "block";
41      markResult.innerHTML = "Score : <span>5 / 5</span>";
42      markResult.style.display = "block";
43      break;
44    case 1:
45      titleResult.textContent = `★ Vous y êtes presque ! ★`;
46      helpResult.textContent =
47        "Retenez une autre réponse dans la case rouge, puis re-validez !";
48      helpResult.style.display = "block";
49      markResult.innerHTML = "Score : <span>4 / 5</span>";
50      markResult.style.display = "block";
51      break;
52    case 2:
53      titleResult.textContent = `⌚ Encore un effort ... ❀`;
54      helpResult.textContent =
55        "Retenez une autre réponse dans les cases rouges, puis re-validez !";
56      helpResult.style.display = "block";
57      markResult.innerHTML = "Score : <span>3 / 5</span>";
58      markResult.style.display = "block";
59      break;
60  }
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
```

```
case 5:
  titleResult.textContent = `⚠ Peut mieux faire ! ⚠`;
  helpResult.style.display = "block";
  helpResult.textContent =
    "Retenez une autre réponse dans les cases rouges, puis re-validez !";
  markResult.style.display = "block";
  markResult.innerHTML = "Score : <span>0 / 5</span>";
  break;

default:
  titleResult.textContent = "Wops, cas inattendu.";
}

const questions = document.querySelectorAll(".question-block");

function addColors(results) {
  results.forEach((response, index) => {
    if (results[index]) {
      questions[index].style.backgroundImage = "linear-gradient(to right, #a8ff78, #78ffd6)";
    } else {
      questions[index].style.backgroundImage = "linear-gradient(to right, #f5567b, #fd674c)";
    }
  })
}

const radioInputs = document.querySelectorAll("input[type='radio']")

radioInputs.forEach(radioInput => radioInput.addEventListener('input', resetColor))

function resetColor(e) {
  const index = e.target.getAttribute("name").slice(1) - 1;
  const parentQuestionBlock = questions[index];

  parentQuestionBlock.style.backgroundColor = "#f1f1f1";
  parentQuestionBlock.style.backgroundImage = "none";
}
```

DOSSIER PROFESSIONNEL (DP)

ANNEXE 3.3 : Développer une interface utilisateur web dynamique : IMC

The figure consists of three side-by-side screenshots of a web application titled "Calcul d'IMC".

- Screenshot 1:** Shows two empty input fields for height ("Votre taille en centimètres") and weight ("Votre poids en kg"). A yellow button labeled "Calculer un IMC" is at the bottom.
- Screenshot 2:** Shows height set to 178 and weight set to 69. The yellow button is still present.
- Screenshot 3:** Shows height set to 186 and weight set to 115. The yellow button is still present. Below the inputs, the calculated BMI value "33.2" is displayed in large orange digits, followed by the result message "Résultat : Obésité modérée".

```
IMC > index.html > ...
6   <meta name="viewport" content="width=device-width, initial-scale=1.0" />
7   <title>Calcul IMC</title>
8   <link rel="stylesheet" href="./style.css" />
9   <link rel="preconnect" href="https://fonts.googleapis.com" />
10  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin />
11  <link
12    href="https://fonts.googleapis.com/css2?family=Lato:wght@100;300;400;700
13    rel="stylesheet"
14  />
15  </head>
16  <body>
17    <div class="container">
18      <h1>Calcul <span>d'IMC </span></h1>
19
20      <form>
21        <div class="inputs-container">
22          <div class="input-group">
23            <label for="height">Votre taille en centimètres</label>
24            <input
25              type="number"
26              placeholder="Votre taille en centimètres"
27              id="height"
28              class="height-input"
29            />
30        </div>
31
32        <div class="input-group">
33          <label for="weight">Votre poids en kg</label>
34          <input
35            type="number"
36            placeholder="Votre poids en kg"
37            id="weight"
38            class="weight-input"
39          />
40        </div>
41      </div>
42
43      <button>Calculer un IMC</button>
44    </form>
45
46    <div class="info">
47      <p class="bmi-value">0</p>
48      <p class="result">En attente du résultat...</p>
49    </div>
50
51  </div>
52
53  <script src="app.js"></script>
54 </body>
</html>
```

DOSSIER PROFESSIONNEL (DP)

The image shows a code editor with two tabs: 'app.js' and 'style.css'. The 'style.css' tab is active, displaying CSS code for a BMI calculator application. The 'app.js' tab is also visible, showing JavaScript code for the same application.

```
JS app.js      # style.css  x
Projets terminés > IMC > # style.css > ↗
38   justify-content: center;
39 }
40 .inputs-container label {
41   display: block;
42   margin-bottom: 10px;
43   margin-left: 5px;
44 }
45 .inputs-container input {
46   min-width: 350px;
47   font-size: 18px;
48   padding: 15px;
49   border: none;
50   border-bottom: 1px solid #333333;
51   box-shadow: 0 5px 5px rgba(0,0,0,.3);
52   border-radius: 5px;
53 }
54 .inputs-container input:focus {
55   outline: 1px solid #222222be;
56 }
57 .input-group:nth-child(2){
58   margin-left: 20px;
59 }
60
61 form button {
62   font-size: 20px;
63   min-width: 200px;
64   margin: 40px auto;
65   display: block;
66   padding: 10px;
67   border: none;
68   font-weight: 700;
69   border-radius: 5px;
70   color: #333;
71   cursor: pointer;
72   background: #fabe33;
73   box-shadow: 0 2px 3px rgba(0,0,0,.5)
74 }
75 form button:hover {
76   background: #f2b527;
77 }
78 .info {
79   text-align: center;
80 }
81 .bmi-value {
82   font-size: 45px;
83   margin-bottom: 5px;
84   font-weight: bolder;
85 }
86 .result {
87   font-family: Raleway, sans-serif;
88   font-size: 20px;
89 }
90
91 @media (max-width: 800px) {
92   .container {
93     max-width: 600px;
94   }
95   .inputs-container {
96     flex-direction: column;
97     align-items: center;
98   }
99   .input-group:nth-child(2) {
100     margin-top: 30px;
101     margin-left: 0;
102   }
103   .input-group {
104     width: 100%;
105   }
106   .input-group input {
107     width: 100%;
108     min-width: auto;
109   }
110 }
```

```
JS app.js      x
Projets terminés > IMC > JS app.js > ...
1  const BMIData = [
2    { name: "Maigreux", color: "midnightblue", range: [0, 18.5] },
3    { name: "Bonne santé", color: "green", range: [18.5, 25] },
4    { name: "Surpoids", color: "lightcoral", range: [25, 30] },
5    { name: "Obésité modérée", color: "orange", range: [30, 35] },
6    { name: "Obésité sévère", color: "crimson", range: [35, 40] },
7    { name: "Obésité morbide", color: "purple", range: 40 },
8  ];
9
10 // IMC = poids en kg / taille2 en m
11
12 const form = document.querySelector("form");
13
14 form.addEventListener("submit", handleForm);
15
16 function handleForm(e) {
17   e.preventDefault();
18
19   calculateBMI();
20 }
21
22 const inputs = document.querySelectorAll("input");
23
24 function calculateBMI() {
25   const height = inputs[0].value;
26   const weight = inputs[1].value;
27
28   if (!height || !weight || height <= 0 || weight <= 0) {
29     handleError();
30     return;
31   }
32
33   const BMI = (weight / Math.pow(height / 100, 2)).toFixed(1);
34   console.log(BMI);
35
36   showResult(BMI);
37 }
38
39 const displayBMI = document.querySelector(".bmi-value");
40 const result = document.querySelector(".result");
41
42 function handleError() {
43   displayBMI.textContent = "Wops";
44   displayBMI.style.color = "inherit";
45   result.textContent = "Remplissez correctement les inputs.";
46 }
47
48 function showResult(BMI) {
49   const rank = BMIData.find(data => {
50     if (BMI >= data.range[0] && BMI < data.range[1]) return data;
51     else if (typeof data.range === "number" && BMI >= data.range) return data;
52   });
53
54   displayBMI.textContent = BMI;
55   displayBMI.style.color = `${rank.color}`;
56   result.textContent = `Résultat : ${rank.name}`;
57 }
```

DOSSIER PROFESSIONNEL (DP)

ANNEXE 4.1 : Réaliser une interface utilisateur avec une gestion de contenu ou e-commerce : Boutique Wordpress

Home

Acheter par catégorie

Produits populaires

Meilleures ventes

Acheter par catégorie



Desktop (6)



Cart

Produit	Prix	Quantité	Sous-total
HP OMEN Tower PC Desktop	3500,00 €	1	3500,00 €
Sunglasses	90,00 €	1	90,00 €
Samsung Galaxy A51 (6GB, 128GB)	450,00 €	1	450,00 €

Samsung Galaxy A51 (6GB, 128GB) Dual Sim With Official Warranty
450,00 €

Ajouter au panier



Dell Inspiron 3567 Core i3-7130U 8GB 1TB HDD 15,6 HD LED Win10 Black
With International Warranty
1000,00 €

Ajouter au panier

Meilleures ventes



Archives

Categories

Total panier

Sous-total	4050,00 €
Expédition	Forfait
Les options de livraison seront mises à jour lors de la commande.	
Calculer les frais d'expédition	
Total	4050,00 €

[Valider la commande →](#)

DOSSIER PROFESSIONNEL (DP)

Cart

Produit: HP OMEN Tower PC Desktop Core i7 7th Gen 8GB RAM 1TB HDD 8GB NVIDIA GeForce GTX1070 Graphic Card

Prix: 3500,00 €

Quantité: 1

Sous-total: 3500,00 €

Produit: Sunglasses

Prix: 90,00 €

Quantité: 1

Sous-total: 90,00 €

Produit: Samsung Galaxy A51 (6GB, 128GB) Dual Sim With Official Warranty

Prix: 450,00 €

Prix: Quantité: Sous-total:

Prix: Quantité: Sous-total:

Prix: Quantité: Sous-total:

Code promo

Appliquer le code promo

Mettre à jour le panier

Total panier

Sous-total: 4058,00 €

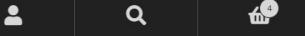
Expédition: Forfait

Les options de livraison seront mises à jour lors de la commande.

Calculer les frais d'expédition

Total: 4058,00 €

Valider la commande →



e-commerce Menu

Accueil > Produits > Résultats de recherche pour "iphone"

Résultats de recherche : « iphone »

Pertinence ▾

4 résultats affichés

iPhone 7 (Refurbished)
650,00 €

HP OMEN Tower PC
Desktop Core i7 7th Gen
8GB RAM 1TB HDD 8GB NVIDIA
GeForce GTX1070 Graphic Card
1 x 3500,00 €

Sunglasses
1 x 90,00 €

Samsung Galaxy
A51 (6GB, 128GB) Dual
Sim With Official Warranty
1 x 450,00 €

Vneck Tshirt
1 x 18,00 €

Sous-total : 4058,00 €

Voir le panier →

Commander →

DOSSIER PROFESSIONNEL (DP)

ANNEXE 4.2 : Réaliser une interface utilisateur avec une gestion de contenu ou e-commerce : Boutique Symfony

The screenshot shows a web application interface for "La Boutique Française". At the top, there is a navigation bar with links for "Home", "Search", and "Cart". Below the navigation, a header section features a large dark grey circle with the word "Heading" below it. To the left of the circle, there is a heading "Example headline." followed by a short text snippet and a "Sign up today" button.

The main content area displays a grid of three products, each represented by a dark grey circle with the word "Heading" below it. Each product has a small description snippet and a "View details" button. The first product's snippet includes placeholder text: "Donec sed odio dui. Etiam porta sem malesuada magna mollis euismod. Nullam id dolor id nibh ultricies vehicula ut id elit. Morbi leo risus, porta ac consectetur ac, vestibulum at eros. Praesent commodo cursus magna, vel scelerisque nisl consectetur fusce dapibus, tellus ac cursus commodo." The second and third products have similar snippets.

On the right side of the screen, there is a sidebar with the same "Heading" element and a "Sign up today" button. At the bottom, there is a footer section with a large dark grey circle labeled "Heading" and a "View details" button. The footer also contains a snippet of placeholder text: "Donec sed odio dui. Etiam porta sem malesuada magna mollis euismod. Nullam id dolor id nibh ultricies vehicula ut id elit. Morbi leo risus, porta ac consectetur ac, vestibulum at eros. Praesent commodo cursus magna, vel scelerisque nisl consectetur fusce dapibus, tellus ac cursus commodo." A browser toolbar is visible at the very bottom.

DOSSIER PROFESSIONNEL (DP)

File tree:

```

templates
└── account
    └── index.html.twig
home
└── index.html.twig
home_controller
└── index.html.twig
register
└── index.html.twig
security
└── login.html.twig
base.html.twig

```

index.html.twig (Content)

```

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

{% block title %}Mon Compte - La Boutique Francaise{% endblock %}

{% block content %}
<h1>Mon Compte</h1>
Bienvenue {{ app.user.firstname }} dans votre compte.
{% endblock %}

```

base.html.twig (Content)

```

<!DOCTYPE html>
<html lang="fr">
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
<meta name="description" content="">
<meta name="author" content="Mark Otto, Jacob Thornton, and Bootstrap contributors">
<meta name="generator" content="Jekyll v4.1.1">
<title>{{ block('title') }} La Boutique Francaise - 100% Made in France {{ endblock }}</title>

<!-- Bootstrap core CSS -->
<link href="{{ asset('assets/css/bootstrap.min.css') }}" rel="stylesheet">

<style>
.bd-placeholder-img {
font-size: 1.125rem;
text-anchor: middle;
-webkit-user-select: none;
-moz-user-select: none;
-ms-user-select: none;
user-select: none;
}

@media (min-width: 768px) {
.bd-placeholder-img-lg {
font-size: 3.5rem;
}
}
</style>
<!-- Custom styles for this template -->
<link href="{{ asset('assets/css/carousel.css') }}" rel="stylesheet">
</head>
<body>
<header>
<nav class="navbar navbar-expand-md navbar-dark fixed-top bg-dark">
<a class="navbar-brand" href="#">La Boutique Francaise</a>
<button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarCollapse" aria-controls="navbarCollapse" aria-expanded="false" aria-label="Toggle navigation">

</button>
<div class="collapse navbar-collapse" id="navbarCollapse">


- Home <span class="sr-only">\(current\)</span>


<form class="form-inline mt-2 mt-md-0">

<button class="btn btn-outline-success my-2 my-sm-0" type="submit">Search</button>



base.html.twig (Content)



```

<main role="main">
{# if block('carousel') is defined #}

<div id="myCarousel" class="carousel slide" data-ride="carousel">
<ol class="carousel-indicators">
-
-
-

<div class="carousel-inner">
<div class="carousel-item active">

<div class="carousel-caption text-left">
<h1>Example headline.</h1>
<p>Cras justo odio, dapibus ac facilisis in, egestas eget quam. Donec id elit non mi porta ac nibus faucibus.
<div class="carousel-caption">
<h1>Another example headline.</h1>
<p>Cras justo odio, dapibus ac facilisis in, egestas eget quam. Donec id elit non mi porta ac nibus faucibus.
<div class="carousel-caption text-right">
<h1>One more for good measure.</h1>
<p>Cras justo odio, dapibus ac facilisis in, egestas eget quam. Donec id elit non mi porta ac nibus faucibus.
Previous

Next


```


```

DOSSIER PROFESSIONNEL (DP)

```
/ index.html.twig A X
templates > home > / index.html.twig
1  (% extends 'base.html.twig' %)
2
3  (% block carousel %){% endblock %}
4
5  (% block content %)
6      <div class="container marketing">
7
8          <!-- Three columns of text below the carousel -->
9          <div class="row">
10             <div class="col-lg-4">
11                 <img class="bd-placeholder-img rounded-circle" width="140" height="140" xmlns="http://www.w3.org/2000/svg" preserveAspectRatio="xMidYMid slice 100% 100%" alt="Placeholder image for the first column."/>
12                 <h2>Heading</h2>
13                 <p>Donec sed odio dui. Etiam porta sem malesuada magna mollis euismod. Nullam id dolor id nibh ultricies vehicula ut id elit. Morbi leo risus, tristique nec, pretium ut, fermentum at, nunc.</p>
14                 <p><a href="#" role="button">View details &raquo;</a></p>
15             </div><!-- /.col-lg-4 -->
16             <div class="col-lg-4">
17                 <img class="bd-placeholder-img rounded-circle" width="140" height="140" xmlns="http://www.w3.org/2000/svg" preserveAspectRatio="xMidYMid slice 100% 100%" alt="Placeholder image for the second column."/>
18                 <h2>Heading</h2>
19                 <p>Duis mollis, est non commodo luctus, nisi erat porttitor ligula, eget lacinia odio sem nec elit. Cras mattis consectetur purus sit amet facilisis magna etiam tempor quam semper viverra.</p>
20                 <p><a href="#" role="button">View details &raquo;</a></p>
21             </div><!-- /.col-lg-4 -->
22             <div class="col-lg-4">
23                 <img class="bd-placeholder-img rounded-circle" width="140" height="140" xmlns="http://www.w3.org/2000/svg" preserveAspectRatio="xMidYMid slice 100% 100%" alt="Placeholder image for the third column."/>
24                 <h2>Heading</h2>
25                 <p>Donec sed odio dui. Cras justo odio, dapibus ac facilisis in, egestas eget quam. Vestibulum id ligula porta felis euismod semper. Praesent commodo massa, pulvinar at, mollis non, vestibulum in, risus.</p>
26                 <p><a href="#" role="button">View details &raquo;</a></p>
27             </div><!-- /.col-lg-4 -->
28         </div><!-- /.row -->
29
30
31         <!-- START THE FEATURETTES -->
32
33         <hr class="featurette-divider">
34
35         <div class="row featurette">
36             <div class="col-md-7">
37                 <h2>Featurette heading</h2>
38                 <p>Donec ullamcorper nulla non metus auctor fringilla. Vestibulum id ligula porta felis euismod semper. Praesent commodo massa, pulvinar at, mollis non, vestibulum in, risus.</p>
39             </div>
40             <div class="col-md-5">
41                 
42             </div>
43         </div>
44
45         <hr class="featurette-divider">
46
47         <div class="row featurette">
48             <div class="col-md-7 order-md-2">
49                 <h2>Featurette heading</h2>
50                 <p>Oh yeah, it's that good. <span>See for yourself.</span></p>
```

```
/ login.html.twig A X
templates > security > / login.html.twig
1  (% extends 'base.html.twig' %)
2
3  (% block title %)Se connecter - La Boutique Francaise(% endblock %)
4
5  (% block content %)
6      <form method="post">
7          (% if error %)
8              <div class="alert alert-danger">{{ error.messageKey|trans(error.messageData, 'security') }}</div>
9          (% endif %)
10
11          (% if app.user %)
12              <div class="mb-3">
13                  You are logged in as {{ app.user.userIdentifier }}, <a href="{{ path('app_logout') }}>Logout</a>
14              </div>
15          (% endif %)
16
17          <h3>Merci de vous connecter</h3>
18          <label for="inputEmail">votre email</label>
19          <input type="email" value="{{ last_username }}" name="email" id="inputEmail" class="form-control" placeholder="votre email" autocomplete="email" required autofocus>
20          <label for="inputPassword">votre mot de passe</label>
21          <input type="password" name="password" id="inputPassword" class="form-control" placeholder="votre mot de passe" autocomplete="current-password" required>
22
23          <input type="hidden" name="_csrf_token"
24              value="{{ csrf_token('authenticate') }}>
25
26
27          # Uncomment this section and add a remember me option below your firewall to activate remember me functionality.
28          See https://symfony.com/doc/current/security/remember_me.html
29
30          <div class="checkbox mb-3">
31              <label>
32                  <input type="checkbox" name="_remember_me"> Remember me
33              </label>
34          </div>
35
36      (% if errors %)
37          <hr>
38          <button class="btn btn-lg btn-primary btn-sm" type="submit">
39              Se Connecter
40          </button>
41      (% endif %)
42  (% endblock %)
43
```

DOSSIER PROFESSIONNEL (DP)

ANNEXE 5.1 Création d'une base de données : BD locale site Wordpress

MySQL **Apache** **Nginx**

MAMP supports MySQL 5.7 The leading HTTP server is integrated Also the Apache alternative is included

PHP **Caches** **MAMP Cloud***

The leading scripting language in web development For your choice: APC, eAccelerator, XCache und OPCache Sync host data through Dropbox

XAMPP Control Panel v3.3.0 [Compiled: Apr 6th 2021]

Modules	Service	Module	PID(s)	Port(s)	Actions
		Apache	11792 13284	443, 8080	<input type="button" value="Stop"/> <input type="button" value="Admin"/> <input type="button" value="Config"/> <input type="button" value="Logs"/>
		MySQL	1448	3306	<input type="button" value="Stop"/> <input type="button" value="Admin"/> <input type="button" value="Config"/> <input type="button" value="Logs"/>
		FileZilla			<input type="button" value="Start"/> <input type="button" value="Admin"/> <input type="button" value="Config"/> <input type="button" value="Logs"/>
		Mercury			<input type="button" value="Start"/> <input type="button" value="Admin"/> <input type="button" value="Config"/> <input type="button" value="Logs"/>
		Tomcat			<input type="button" value="Start"/> <input type="button" value="Admin"/> <input type="button" value="Config"/> <input type="button" value="Logs"/>

```

01:01:44 [main] Checking for prerequisites
01:01:45 [main] All prerequisites found
01:01:45 [main] Initializing Modules
01:01:45 [main] Starting Check-Timer
01:01:45 [main] Control Panel Ready
01:01:47 [Apache] Attempting to start Apache app...
01:01:47 [Apache] Status change detected: running
01:01:48 [mysql] Attempting to start MySQL app...
01:01:49 [mysql] Status change detected: running

```

Disque local (C) > xampp > htdocs

Nom	Modifié le	Type	Taille
.well-known	13/02/2023 23:56	Dossier de fichiers	
wp-admin	13/02/2023 23:11	Dossier de fichiers	
wp-content	08/04/2023 01:05	Dossier de fichiers	
wp-includes	08/04/2023 01:05	Dossier de fichiers	
.htaccess	13/02/2023 23:59	Fichier HTACCESS	1 Ko
index.php	06/02/2020 07:33	Fichier source PHP	1 Ko
license.txt	08/04/2023 01:05	Document texte	20 Ko
readme.html	08/04/2023 01:05	Chrome HTML Do...	10 Ko
wp-activate.php	17/09/2022 01:13	Fichier source PHP	8 Ko
wp-blog-header.php	06/02/2020 07:33	Fichier source PHP	1 Ko
wp-comments-post.php	10/11/2021 00:07	Fichier source PHP	3 Ko
wp-config.php	13/02/2023 23:56	Fichier source PHP	4 Ko
wp-config-sample.php	08/04/2023 01:05	Fichier source PHP	4 Ko
wp-cron.php	08/04/2023 01:05	Fichier source PHP	6 Ko
wp-links-opml.php	08/04/2023 01:05	Fichier source PHP	3 Ko
wp-load.php	08/04/2023 01:05	Fichier source PHP	4 Ko
wp-login.php	08/04/2023 01:05	Fichier source PHP	49 Ko
wp-mail.php	08/04/2023 01:05	Fichier source PHP	9 Ko
wp-settings.php	08/04/2023 01:05	Fichier source PHP	25 Ko
wp-signup.php	17/09/2022 02:35	Fichier source PHP	34 Ko
wp-trackback.php	08/04/2023 01:05	Fichier source PHP	5 Ko
xmlrpc.php	08/04/2023 01:05	Fichier source PHP	4 Ko

Identifiant ou adresse e-mail: dave

Mot de passe:

Se souvenir de moi

Mot de passe oublié ?

← Aller sur e-commerce

WordPress > Setup Configuration [localhost:8080/wp-admin/setup-config.php?step=1]

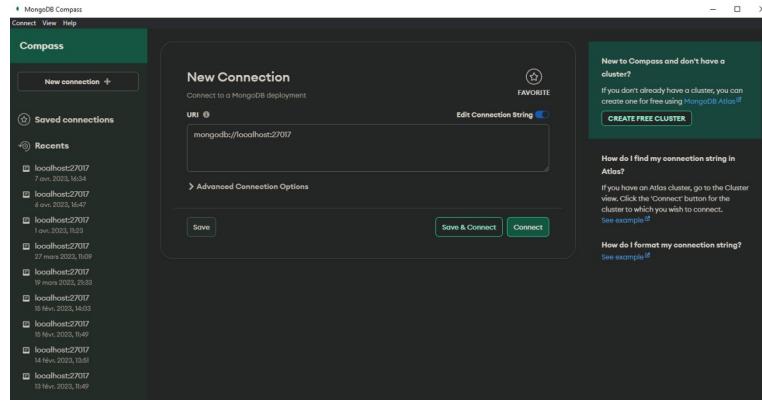
Below you should enter your database connection details. If you are not sure about these, contact your host.

Database Name	wordpress	The name of the database you want to use with WordPress.
Username	username	Your database username.
Password	password	Your database password.
Database Host	localhost	You should be able to get this info from your web host, if localhost does not work.
Table Prefix	wp_	If you want to run multiple WordPress installations in a single database, change this.

DOSSIER PROFESSIONNEL (DP)

DOSSIER PROFESSIONNEL (DP)

ANNEXE 5.2 Création d'une base de données : MongoDBCompass



The screenshot shows the MongoDB Compass interface connected to the 'biblio2.livres' collection. The collection has 5 documents and 1 index. The documents are listed as follows:

- ```
_id: ObjectId('63a22af34bd4c485cd638d89')
nom: "L'algorithme selon Hérodote"
auteur: "Mathieu Gaston"
pages: 200
```
- ```
_id: ObjectId('6427f6590c1e6bb87caef')
nom: "4ème Retour de l'Amour"
auteur: "Hubert Bonisseur de la Bath"
pages: 1
description: ""
..._v: 0
```
- ```
_id: ObjectId('6427ff1ea21fc892b9c8c44d')
nom: ""
auteur: ""
pages: null
description: ""
..._v: 0
```
- ```
_id: ObjectId('642881c434ae2979ee1beb1')
nom: ""
```

DOSSIER PROFESSIONNEL (DP)

ANNEXE 6 : Développer les composants d'accès aux données : site Javascript

```
{} package.json x
JS-16 > siteServeur > {} package.json > ...
1
2   "name": "siteServeur",
3   "version": "1.0.0",
4   "description": "",
5   "main": "index.js",
6   "scripts": {
7     "test": "echo \\"$Error: no test specified\\" && exit 1"
8   },
9   "keywords": [],
10  "author": "",
11  "license": "ISC"
12 }
```

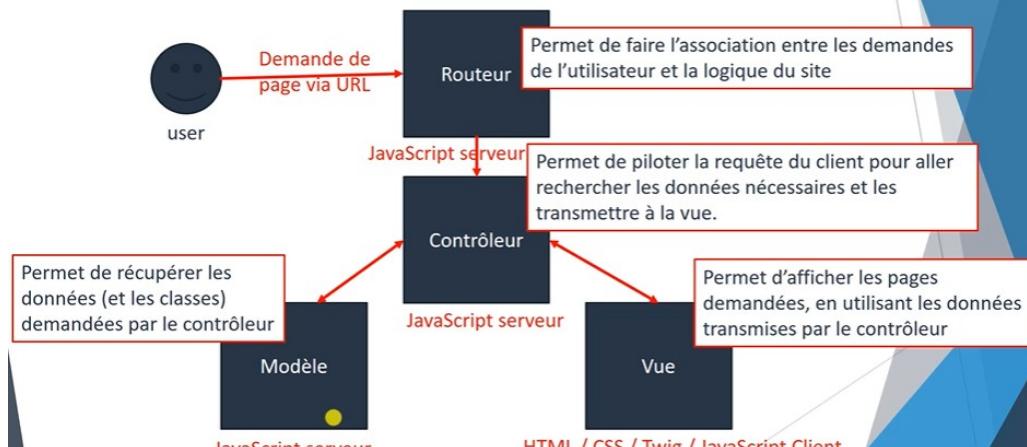
```
JS serveur.js x
C: > Users > impin > Downloads > E3 (2) > siteServeur > JS serveur.js > ...
1 var express = require("express");
2 var server = express();
3
4 server.listen(3000);
5
6 server.get("/", (requete, reponse) =>{
7   console.log("demande recue avec la méthode GET sur l'url /")
8   reponse.end("Demande GET reçue")
9 })
10
11 server.get("/test", (requete, reponse) =>{
12   console.log("demande recue avec la méthode GET sur l'url /test")
13   reponse.end("Demande GET reçue")
14 })
15
16 server.post("/test", (requete, reponse) =>{
17   console.log("demande recue avec la méthode POST sur l'url /test")
18   reponse.end("Demande POST reçue")
19 })
```

Module nodemon

- Pour chaque modification réalisée, nous sommes obligé de relancer le serveur, ce qui peut devenir fastidieux.
- Le module **nodemon** permet de relancer automatiquement le serveur à chaque modification et sauvegarde de ce dernier
- Etape 1 : installation du module c:\Users\Matthieu\Desktop\mongo\biblio>npm install --save-dev nodemon
- Etape 2 : indiquer au serveur qu'il doit se lancer en utilisant ce module

```
{} package.json x JS serveur.js
mongo > biblio > {} package.json > ...
1
2   "name": "biblio",
3   "version": "1.0.0",
4   "description": "",
5   "main": "index.js",
6   "scripts": {
7     "test": "echo \\"$Error: no test specified\\" && exit 1"
8   },
9   "keywords": []
10 }
```

L'architecture MVC :



DOSSIER PROFESSIONNEL (DP)

Module morgan

- Le module « morgan » va nous permettre d'ajouter des informations dans le terminal en « logant » les demandes clientes reçues

- Installation du module :

```
c:\Users\Matthieu\Desktop\mongo\biblio>npm install --save morgan
```

```
JS serveur.js X
mongo > biblio > JS serveur.js > ...
1 var express = require("express");
2 var server = express();
3 var morgan = require("morgan");
4
5 server.use(morgan("dev"));
6
7 server.listen(3000);

{} package.json JS serveur.js X

JS serveur.js X
mongo > biblio > JS serveur.js > ...
1 var express = require("express");
2 var server = express();
3 var morgan = require("morgan");
4
5 server.use(morgan("dev"));
6
7 server.listen(3000);

J-16 > siteServeur > JS serveur.js > server.use() callback
 1 reponse.end(`Demande POST reçue`)
2 }
3
4 server.use((requete,reponse,suite) => {
5   const error = new Error("Page non trouvée");
6   error.status= 404;
7   suite(error);
8 }
9
10 server.use((error,requete,reponse) => {
11   reponse.status(error.status || 500);
12   reponse.end(error.message);
13 })
```

```
PROBLÈMES 1 SORTIE CONSOLE DE DÉBOGAGE TERMINAL 3: node + ⌂ ⌂

at c:\Users\Matthieu\Desktop\JS-16\siteServeur\node_modules\express\lib\router\index.js:28
[nodemon] restarting due to changes...
[nodemon] starting `node serveur.js` ⚡
GET /pages 404 3.058 ms - 1374
Error: Page non trouvée
  at server.use (c:\Users\Matthieu\Desktop\JS-16\siteServeur\serveur.js:25:19)
  at Layer.handle [as handle_request] (c:\Users\Matthieu\Desktop\JS-16\siteSe
xpress\lib\router\layer.js:95:5)
  at trim_prefix (c:\Users\Matthieu\Desktop\JS-16\siteServeur\node modules\ex
```

Fichier de routage

- Pour alléger le fichier serveur.js nous pouvons décider de faire un fichier de routage (ou plusieurs).
- On peut donc créer le fichier « router.js », et y inscrire toutes les routes
- Pour récupérer le fichier routeurs dans le serveur, il suffit de l'exporter « module.exports » depuis le fichier « routeur.js » et de le récupérer dans le serveur

```
Serveur.js
mongo > biblio > JS serveur.js > ...
1 var express = require("express");
2 var server = express();
3 var morgan = require("morgan");
4 var router = require("./routeur");
5
6 server.use(morgan("dev"));
7 server.use("/",router);
8
9 server.listen(3000);
```

```
Router.js
mongo > biblio > JS routeur.js > router.post("/test") callback
1 var express = require("express");
2 var router = express.Router();
3
4 router.get("/", (requete, reponse) => {
5   reponse.render("accueil.html.twig");
6 });
7
8 router.post("test", (requete, reponse) => {
9   console.log("Demande reçue via un formulaire");
10  reponse.end("page test méthode POST");
11 });
12
13 //Gère l'erreur 404
14 router.use((requete, reponse, suite)>{
15   const error = new Error("Page non trouvée");
16   error.status = 404;
17   console.log("ici");
18   suite(error); //envoie à la route ci-dessous avec "error" générée
19 });
20
21 //Gère toutes les erreurs
22 router.use((error,requete,reponse,suite)>{
23   reponse.status(error.status || 500);
24   reponse.end(error.message);
25 });
26
27 module.exports = router
```

Envoyer une page HTML

- Pour envoyer une page HTML on va utiliser un module supplémentaire : un moteur de template. Il en existe plusieurs : EJS, Twig, Jade ...
- Ces modules permettent de faciliter la gestion des pages HTML et le passage d'informations entre client et serveur
- Installation de Twig :

```
c:\Users\Matthieu\Desktop\mongo\biblio>npm install --save twig
```
- Création de la page d'accueil dans le dossier « views », et la route « / » dans le routeur :

```
Router.js
mongo > biblio > JS routeur.js > router.post("/test") callback
1 const express = require("express");
2 const router = express.Router();
3 const twig = require("twig");
4
5 router.get("/", (requete, reponse) => {
6   reponse.render("accueil.html.twig");
7 });
8
9 router.post("test", (requete, reponse) => {
10   console.log("Demande reçue via un formulaire");
11   reponse.end("page test méthode POST");
12 });
13
14 module.exports = router
```

```
Accueil.html.twig
mongo > biblio > views > Accueil.html.twig
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <meta http-equiv="X-UA-Compatible" content="ie=edge">
7     <title>Document</title>
8   </head>
9   <body>
10     <h1>Ma page d'accueil !</h1>
11   </body>
12 </html>
```

- Par défaut, « express » va chercher les vues dans le dossier « views »

Page

DOSSIER PROFESSIONNEL (DP)

ANNEXE 7 : Développer la partie back-end d'une application web ou web mobile

```
JS routeur.js ×
JS routeur.js > ...
1 var express = require("express");
2 var routeur = express.Router();
3 const twig = require("twig");
4
5 routeur.get("/", (requete, reponse) =>{
6   reponse.render("accueil.html.twig")
7 })
8
9 routeur.get("/livres", (requete, reponse) =>{
10   reponse.render("livres/liste.html.twig")
11 })
12
13 routeur.get("/livres/:nom", (requete, reponse) => {
14   console.log(requete.params.nom)
15   reponse.render("livres/livre.html.twig", {nom:requete.params.nom})
16 })
17
18 routeur.use((requete, reponse, suite) => {
19   const error = new Error("Page non trouvée");
20   error.status = 404;
21   suite(error);
22 })
23
24 routeur.use((error, requete, reponse) => {
25   reponse.status(error.status || 500);
26   reponse.end(error.message);
27 })
28
29 module.exports = routeur;
```



```
JS serveur.js ×
JS serveur.js > ...
1 const express = require("express");
2 const server = express();
3 const morgan = require("morgan");
4 const router = require("./routeur");
5 const mongoose = require("mongoose");
6
7 mongoose.connect("mongodb://localhost/biblio2", {useNewUrlParser:true,useUnifiedTopology:true});
8
9 const livreSchema = mongoose.Schema({
10   _id: mongoose.Schema.Types.ObjectId,
11   nom: String,
12   auteur: String,
13   pages: Number
14 })
15 const livreModel = mongoose.model("Livre", livreSchema);
16
17 livreModel.find()
18   .exec()
19   .then(livres => {
20     console.log(livres)
21   })
22   .catch();
23
24 server.use(express.static("public"))
25 server.use(morgan("dev"));
26 server.use("/", router);
27
28 server.listen(3000);
```

```
JS livres.modele.js ×
siteServeur > models > JS livres.modele.js > ...
1 const mongoose = require("mongoose");
2
3 const livreSchema = mongoose.Schema({
4   _id: mongoose.Schema.Types.ObjectId,
5   nom: String,
6   auteur: String,
7   pages: Number,
8   description: String,
9 })
10 module.exports = mongoose.model("Livre", livreSchema);
```

```
JS user.modele.js ×
siteServeur > models > JS user.modele.js > userSchema > username
1 const mongoose = require('mongoose');
2
3 const userSchema = mongoose.Schema({
4   username: {type: String, required: true },
5   email: {type: String, required: true },
6   password: {type: String, required: true },
7   admin: {type: Boolean }
8 });
9
10 module.exports = mongoose.model('User', userSchema);
```

Affichage d'un livre

- Maintenant que nous disposons d'une base de données, nous pouvons véhiculer les informations de page en page et l'utiliser comme source. Ainsi, à partir de l'identifiant du livre choisi dans la liste, nous pouvons afficher ses informations.

```
<td><a href="/livres/{{livre._id}}">{{livre.nom}}</a></td>

router.get("/livres/:id", (requete, reponse) => {
  var livre = livreSchema.findById(requete.params.id)
  .exec()
  .then(livre => {
    reponse.render("livres/livre.html.twig", {livre: livre});
  })
  .catch(error => {
    console.log(error);
  });
});
```



```
mongo > biblio > views > livres > / livre.html.twig
1  {% extends "base.html.twig" %}
2  {% block titre %}{{livre.nom}} {% endblock %}
4
5  {% block contenu %}
6    <div class="row m-2">
7      <div class="col-6">
8          <h2>Titre : {{livre.nom}}</h2>
9          <div>Auteur : {{livre.auteur}}</div>
10         <div>Nombre de pages : {{livre.pages}}</div>
11         
12     </div>
13     <div class="col-6">
14         <h2>Résumé :</h2>
15         <div>{{livre.description}}</div>
16     </div>
17   </div>
18
20
21  {% endblock %}
```

DOSSIER PROFESSIONNEL (DP)



Récupération des livres

- Pour respecter le modèle MVC et faire en sorte d'avoir un code structuré, nous allons créer un répertoire dédié à la partie « modèle »

```

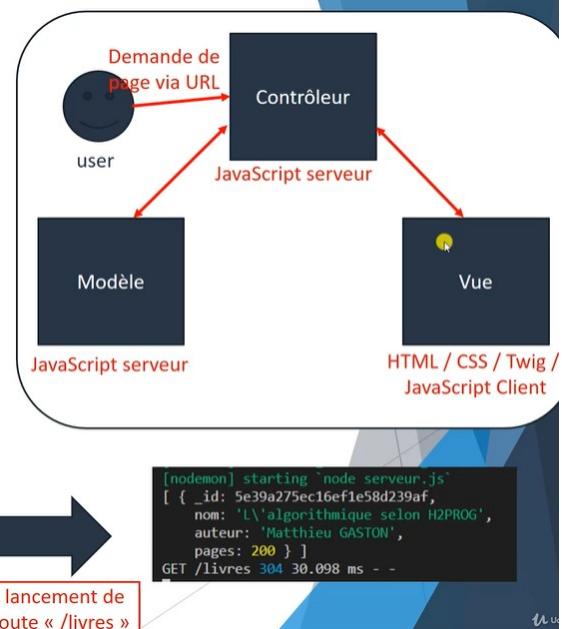
mongo > biblio > models > ls
ls livres.model.js livres.schema.js pages.js

1 const mongoose = require('mongoose');
2
3 const livreSchema = mongoose.Schema({
4   _id: mongoose.Schema.Types.ObjectId,
5   nom: String,
6   auteur: String,
7   pages: Number
8 });
9
10 module.exports = mongoose.model('Livre', livreSchema);

mongo > biblio > # routes.js > router.get('/livres') callback > livres > then callback
1  const express = require('express');
2  const router = express.Router();
3  const twig = require('twig');
4  const livreSchema = require('../models/livres.model');
5
6  router.get('/', (requete, reponse) => {
7    reponse.render('accueil.html.twig');
8  },
9
10  router.get('/livres', (requete, reponse) => {
11    var livres = livreSchema.find()
12    .exec()
13    .then(livres => {
14      console.log(livres);
15      reponse.render('livres/liste.html.twig');
16    })
17    .catch();
18  });
19 });

H2PROG

```



Au lancement de la route « /livres »

```
[nodemon] starting `node serveur.js`
[nodemon] environment: production
[nodemon] watching directory .
[nodemon] watching extensions js,json
[nodemon] starting node serveur.js
[nodemon] waiting for changes...
[nodemon] 0 info [nodemon] starting `node serveur.js`
```

Récupération des livres

- On peut modifier la réponse envoyée à la vue pour qu'elle puisse traiter les données et faire l'affichage des livres.

```

router.get("/livres", (requete, reponse) => {
  var livres = livreSchema.find()
  .exec()
  .then(livres => {
    reponse.render("livres/liste.html.twig", {livres : livres});
  })
  .catch(error => {
    console.log(error);
  });
});
```



```


| Titre                     | Auteur          | Nombre de page | Actions                                              |
|---------------------------|-----------------|----------------|------------------------------------------------------|
| L'algorithme selon H2PROG | Matthieu GASTON | 200            | <button>MODIFIER</button> <button>SUPPRIMER</button> |


```

DOSSIER PROFESSIONNEL (DP)



Ajout d'un livre

- ▶ Pour réaliser l'ajout d'un livre nous avons plusieurs possibilités :
 - ▶ Créer une page dédiée (et donc créer une nouvelle route)
 - ▶ Mettre en place un formulaire caché, qui apparaît au clic du bouton « ajouter » comme nous l'avons fait dans la partie statique.
- ▶ A des fins pédagogiques nous allons choisir cette deuxième solution pour l'ajout, et nous ferons une page dédiée pour la modification.
- ▶ Il va donc falloir envoyer en plus de la page HTML (et du CSS), un fichier JavaScript Client pour que le navigateur puisse afficher ou cacher le formulaire.
- ▶ Etape 1 : ajout du formulaire dans la page « liste.html.twig »
- ▶ Etape 2 : création du fichier JS Client « ajoutLivre.js »

```
✓ public
  > css
  ✓ images
  ✓ javascript
    JS ajoutLivre.js
```

```
mongo > biblio > public > javascript > JS ajoutLivre.js > afficherFormulaire
1   function afficherFormulaire(){
2     document.querySelector("#ajoutForm").removeAttribute("class");
3 }
```

H2PROG - All Rights Reserved – Created by : Matthieu GASTON

The screenshot shows the 'liste.html.twig' template. At the top right, there is a button labeled 'Ajouter'. Below it, a legend says 'Création d'un livre'. There are three form groups: one for the title ('Titre du livre') with an input field, one for the author ('Auteur') with an input field, and one for the number of pages ('Nombre de pages') with a number input field. At the bottom of the form is a 'Valider' button. A script tag at the bottom points to 'ajoutLivre.js'.

Ajout d'un livre

- ▶ A la différence de ce que nous avons fait lors de la programmation de la version statique, il est nécessaire d'envoyer les informations au serveur, afin qu'il puisse mettre à jour la base de données.
- ▶ Il faut donc ajouter la méthode d'envoi des données du formulaire : « POST » et la route demandée.
Dans notre cas la route est vide : « », car on va router la demande vers cette même page (mais pas avec la même méthode)
- ▶ On doit réaliser un formulaire standard HTML et ajouter l'attribut « name » aux différents « input »
- ▶ On crée la route permettant de traiter la demande « postée »

The screenshot shows the 'liste.html.twig' template. The 'POST' method and the 'submit' type for the 'Valider' button are highlighted. The rest of the code is identical to the previous screenshot.

The screenshot shows a code editor with 'Router.js' open. It contains a single line of code defining a POST route for '/livres': `router.post("/livres", (requete, reponse) => [console.log("ici");]);`

DOSSIER PROFESSIONNEL (DP)

Ajout d'un livre – envoie en BD

- ▶ Pour traiter les éléments envoyés par le navigateur client (le formulaire), il faut utiliser les éléments présents dans la requête, et pour nous simplifier la tâche nous devons installer le module « body-parser ».
- ▶ Body-parser va rendre les éléments du « body » utilisables

```
mongo > biblio > JS serveur.js > ...
1  const express = require("express");
2  const server = express();
3  const morgan = require("morgan");
4  const router = require("./routeur");
5  const mongoose = require("mongoose");
6  const bodyParser = require("body-parser");
7
8  mongoose.connect("mongodb://localhost/biblio", { useNewUrlParser: true });
9
10 server.use(express.static("public"));
11 server.use(morgan("dev"));
12 server.use(bodyParser.urlencoded({extended:false}));
13
14 server.use("/",router);
15 server.listen(3000);
```

```
c:\Users\Matthieu\Desktop\mongo\biblio>npm install --save body-parser
router.post("/livres", (requete, reponse) => {
  console.log(requete.body);
});
```

```
node server.js
GET /livres 304 25.922 ms - -
[Object: null prototype] {
  titre: 'test',
  auteur: 'matthieu',
  pages: '300',
  description: 'un super livre' }
```

Ajout d'un livre – envoie en BD

- ▶ Il reste plus qu'à générer un livre en utilisant le Schéma, et à l'envoyer en BD :

```
router.post("/livres", (requete, reponse) => {
  const livre = new livreSchema({
    _id: new mongoose.Types.ObjectId(),
    nom: requete.body.titre,
    auteur: requete.body.auteur,
    pages: requete.body.pages,
    description : requete.body.description,
  })
  livre.save()
    .then(resultat => {
      console.log(resultat);
      reponse.redirect("/livres");
    })
    .catch(error => {
      console.log(error);
    })
});
```

LISTE DES LIVRES DE LA BIBLIOTHÈQUE			
Titre	Auteur	Nombre de page	Actions
L'algorithme selon H2PROG	Matthieu GASTON	200	<button>MODIFIER</button> <button>SUPPRIMER</button>
test	Matthieu	333	<button>MODIFIER</button> <button>SUPPRIMER</button>

```
_id: ObjectId("5e39db52bf7cf04764e444ba")
nom: "L'algorithme selon H2PROG"
auteur: "Matthieu GASTON"
pages: 200
description:"Un super livre sur l'algorithme"

_id: ObjectId("5e39db52bf7cf04764e444ba")
nom: "test"
auteur: "matthieu"
pages: 333
description:"Un super livre"
```

DOSSIER PROFESSIONNEL (DP)

ANNEXE 8 : Élaborer et mettre en œuvre des composants dans une application de gestion de contenu

Message flash

- ▶ Installation du module permettant de gérer les sessions

```
c:\Users\Matthieu\Desktop\mongo\biblio>npm install --save express-session
```

- ▶ Ajout d'un middleware pour gérer les messages de session :

```
//bascule le message dans la réponse  
//Permet de supprimer le message FLASH  
server.use(function(requete, reponse, next){  
    reponse.locals.message = requete.session.message;  
    delete requete.session.message;  
    next();  
});
```

Server.js

- ▶ Génération d'un message en session lors de la demande de suppression

```
//Sessions  
server.set('trust proxy', 1) // trust first proxy  
server.use(session({  
    secret: 'keyboard cat',  
    resave: true,  
    saveUninitialized: true,  
    cookie: { maxAge: 60000 },  
}))
```

```
router.post("/livres/delete/:id", (requete, reponse) => {  
    livreSchema.remove({ _id:requete.params.id })  
    .exec()  
    .then(resultat => {  
        requete.session.message = [  
            type: 'success',  
            contenu: 'Suppression effectuée'  
        ]  
        reponse.redirect('/livres');  
    })  
    .catch(error => {  
        console.log(error);  
    })  
});
```

http://localhost:3001/livres/delete/1

Affichage du message flash

- ▶ Envoi du message sur la route « /livres »

```
router.get("/livres", (requete, reponse) => {  
    var livres = livreSchema.find()  
    .exec()  
    .then(livres => {  
        reponse.render("livres/liste.html.twig", {livres : livres, message: reponse.locals.message});  
    })  
    .catch(error => {  
        console.log(error)  
    });  
});
```

```
{% if message|length > 0 %}  
    <div class="alert alert-{{message.type}}">  
        {{message.contenu}}  
    </div>  
{% endif %}
```

- ▶ Affichage par la vue :



DOSSIER PROFESSIONNEL (DP)

Modification d'un livre

- ▶ Ajout d'une route pour afficher le Template de modification :

```
router.get("/livres/modification/:id", (requete, reponse) => {
  var livre = livreSchema.findById(requete.params.id)
  .exec()
  .then(livre => {
    reponse.render("livres/livre.html.twig", {livre : livre, isModification:true});
  })
  .catch(error => {
    console.log(error);
  });
});
```

```
router.get("/livres/:id", (requete, reponse) => {
  var livre = livreSchema.findById(requete.params.id)
  .exec()
  .then(livre => {
    reponse.render("livres/livre.html.twig", {livre : livre, isModification:false});
  })
  .catch(error => {
    console.log(error);
  });
});
```

- ▶ Le Template utilisé est le même que pour l'affichage d'un livre.
- ▶ On envoie en plus un booléen pour que la vue soit capable d'identifier le type de demande : modification ou affichage d'un livre.

```
82 //Modification d'un Livre (formulaire)
83 router.get("/livres/modification/:id", (requete, reponse)=> {
84   livreSchema.findById(requete.params.id)
85   .exec()
86   .then(livre => {
87     reponse.render("livres/livre.html.twig",{livre : livre, isModification:true})
88   })
89   .catch(error => {
90     console.log(error);
91   })
92 })
```

Modification d'un livre

- ▶ Ajout d'une route pour traiter le formulaire : « /modificationServer »

```
router.post("/modificationServer", (requete, reponse) => {
  const livreUpdate = {
    nom: requete.body.titre,
    auteur: requete.body.auteur,
    pages: requete.body.pages,
    description : requete.body.description,
  };
  livreSchema.update({_id:requete.body.i}, livreUpdate)
  .exec()
  .then(resultat => {
    console.log(resultat)
    requete.session.message = [
      type: 'success',
      contenu: 'Modification effectuée'
    ]
    reponse.redirect("/livres");
  })
  .catch(error => {
    console.log(error);
  });
});
```

```
<form method="POST" action="/modificationServer">
  <div class="form-group">
```

```
  <input type="hidden" name="identifiant" value="{{livre._id}}">
```

LISTE DES LIVRES DE LA BIBLIOTHEQUE			
Modification effectuée			
Titre	Auteur	Nombre de page	Actions
Livre	Matthieu	333	<button>MODIFIER</button> <button>SUPPRIMER</button>
Le virus Asiatique	totoa	321	<button>MODIFIER</button> <button>SUPPRIMER</button>
AJOUTER			

DOSSIER PROFESSIONNEL (DP)

```
94  router.post("/livres/modificationServer", (requete, reponse) => {
95    const livreUpdate = {
96      nom : requete.body.titre,
97      auteur: requete.body.auteur,
98      pages : requete.body.pages,
99      description : requete.body.description
100   }
101   livreSchema.update({_id:requete.body.identifiant}, livreUpdate)
102   .exec()
103   .then(resultat => {
104     if(resultat.Modified < 1) throw new Error("Requête de modification échouée");
105     requete.session.message = {
106       type : 'success',
107       contenu : 'modification effectuée'
108     }
109     reponse.redirect("/livres");
110   })
111   .catch(error => {
112     console.log(error);
113     requete.session.message = {
114       type : 'danger',
115       contenu : error.message
116     }
117     reponse.redirect("/livres");
118   })
119 })
```

Lever des erreurs

- En cas de non ajout / modification / ou suppression dans la BD, nous pouvons lever une erreur pour passer dans le « catch »

- Exemple dans la modification :

The image shows two screenshots. The top screenshot is a code editor displaying a portion of a Node.js file. It contains a POST route for '/livres/modificationServer'. Inside the route handler, there's a try-catch block. The catch block logs the error and sets a session message indicating a danger level with the message 'modification effectuée'. The bottom screenshot is a screenshot of a web application titled 'LISTE DES LIVRES DE LA BIBLIOTHÈQUE'. It shows a table with two rows: 'Livre' by 'Matthieu' (333 pages) and 'Le virus Atlantique' by 'totoa' (321 pages). Each row has 'MODIFIER' and 'SUPPRIMER' buttons. A red box highlights the 'MODIFIER' button for the first row. A yellow box highlights the 'nModified' value in the response object, which is shown as 0. A callout box points to the 'MODIFIER' button with the text 'Champ inexistant, donc la requête ne modifera rien' (Empty field, so the request will not modify anything).

```
router.post("/modificationServer", (requete, reponse) => {
  const livreUpdate = {
    nom : requete.body.titre,
    auteur: requete.body.auteur,
    pages : requete.body.pages,
    description : requete.body.description,
  };
  livreSchema.update({_id:requete.body._id}, livreUpdate)
  .exec()
  .then(resultat => {
    console.log(resultat)
    if(resultat.nModified < 1) throw new Error("Requête de modification échouée");
    requete.session.message = {
      type : 'success',
      contenu : 'Modification effectuée'
    }
    reponse.redirect("/livres");
  })
  .catch(error => {
    console.log(error);
    requete.session.message = {
      type : 'danger',
      contenu : error.message
    }
    reponse.redirect("/livres");
  })
});
```

H2PROG - All Rights Reserved - Created by : Matthieu GASTON

```
94  router.post("/livres/modificationServer", (requete, reponse) => {
95    const livreUpdate = {
96      nom : requete.body.titre,
97      auteur: requete.body.auteur,
98      pages : requete.body.pages,
99      description : requete.body.description
100   }
101   livreSchema.update({_id:requete.body.identifiant}, livreUpdate)
102   .exec()
103   .then(resultat => {
104     if(resultat.Modified < 1) throw new Error("Requête de modification échouée");
105     requete.session.message = {
106       type : 'success',
107       contenu : 'modification effectuée'
108     }
109     reponse.redirect("/livres");
110   })
111   .catch(error => {
112     console.log(error);
113     requete.session.message = {
114       type : 'danger',
115       contenu : error.message
116     }
117     reponse.redirect("/livres");
118   })
119 })
```

DOSSIER PROFESSIONNEL (DP)

Suppression d'un livre

- ▶ Ajout d'un formulaire permettant de gérer la suppression (avec demande de confirmation) :

```
<td>
  <form method="POST" action="/livres/delete/{livre._id}" onSubmit = "return confirm('voulez-vous vraiment Supprimer ?');">
    <button class="btn btn-danger" type="submit">Supprimer</button>
  </form>
</td>
```

- ▶ Création de la route permettant de traiter la demande du client :

```
router.post("/livres/delete/:id", (requete, reponse) => {
  livreSchema.remove({_id:requete.params.id})
  .exec()
  .then(resultat => {
    reponse.redirect('/livres');
  })
  .catch(error => {
    console.log(error);
  })
});
```



```
144  router.post("/livres/delete/:id", (requete, reponse) => {
145    var livre = livreSchema.findById(requete.params.id)
146    .select("image")
147    .exec()
148    .then(livre => {
149      fs.unlink("./public/images/" + livre.image, error => {
150        console.log(error);
151      })
152      livreSchema.remove({_id:requete.params.id})
153        .exec()
154        .then(resultat => {
155          requete.session.message = {
156            type : 'success',
157            contenu : 'Suppression effectuée'
158          }
159          reponse.redirect("/livres");
160        })
161        .catch(error => {
162          console.log(error);
163        })
164      })
165      .catch(error => {
166        console.log(error);
167      })
168    });
  
```

DOSSIER PROFESSIONNEL (DP)

```
170 //LOGIN
171
172 routeur.post('/api/signup', function(req, res){
173     const Data = new User({
174         username : req.body.username,
175         email : req.body.email,
176         password : bcrypt.hashSync(req.body.password, 10)
177     });
178     Data.save().then(() => {
179         console.log("User saved");
180         console.log(Data);
181         res.redirect("/livres");
182     }).catch(err=> console.log(err));
183 });
184
185 //////////CONNEXION
186
187 routeur.post('/api/signin', function(req, res){
188     User.findOne({
189         email: req.body.email
190     }).then(user => {
191         if(!user){
192             res.status(404).send('email invalid')
193         }
194         if(!bcrypt.compareSync(req.body.password, user.password)){
195             res.status(404).send('password invalid')
196         }
197         res.redirect('/livres');
198     }).catch(err => console.log(err));
199 });
200
201
202     routeur.get('/login', function(req, res){
203         res.render('Signin');
204     });
205
```