

## ▀▀▀ Modèle de copie :

**Évaluation en cours de formation**

**Développeur Web et Web Mobile**

**Ceci est un modèle de copie. N’oubliez pas de renseigner vos prénom/nom, ainsi que le nom et le lien vers le projet.**

**Vous pouvez bien sûr agrandir les cadres pour répondre aux questions sur la description du projet si nécessaire.**

**Prénom : Sandrine**

**Nom : Orieux**

**ATTENTION ! PENSEZ À RENSEIGNER VOS NOM ET PRÉNOM DANS LE TITRE DE VOS FICHIERS/PROJETS !**

Nom du projet : Coupart-Nutrition

Lien Github du projet : <https://github.com/SandrineOrieux/Coupart-Nutrition.git>

Les documents demandés sont dans le dossier documentations à la racine du projet

Les originaux des diagrammes peuvent être visualisés sur vsc avec l’extension Draw.io intégration

**Attention ! Merci de bien classer vos documents dans votre Github ou votre drive.**

URL du site (si vous avez mis votre projet en ligne) : <https://coupart-nutrition.sandrine-orieux.fr/>

**Description du projet**

1. Liste des compétences du référentiel qui sont couvertes par le projet

Activité – Type 1 : Développer la partie front-end d’une application web ou web

Mobile en intégrant les recommandations de sécurité

● 1 Maquetter une application

● 2 Réaliser une interface utilisateur web statique et adaptable

● 3 Développer une interface utilisateur web dynamique

● 4 Réaliser une interface utilisateur avec une solution de gestion de contenu ou e-commerce

Activité – Type 2 : Développer la partie back-end d’une application web ou web mobile en intégrant les recommandations de sécurité

● 5 Créer une base de données

● 6 Développer les composants d’accès aux données

● 7 Développer la partie back-end d’une application web ou web mobile

● 8 Élaborer et mettre en œuvre des composants dans une application de gestion de contenu ou e-commerce.

1. Résumé du projet en français d’une longueur d’environ 20 lignes soit 200 à 250 mots, ou environ 1200 caractères espaces non compris

Sandrine Coupart est une diététicienne-nutritionniste dont le cabinet est situé à Caen. En tant que professionnelle de santé, elle prend en charge des patients dans le cadre de consultations diététiques.

Madame Coupart anime également des ateliers de prévention et d’information sur la nutrition. Elle veut que ses services soient mis en valeur.

Son fonctionnement habituel était de transmettre par email à ses patients des recettes santé. N’ayant pas de site web, elle voulait profiter de l’occasion pour partager, à l’avenir, quelques-unes de ses recettes à un plus grand nombre de visiteurs. Il y aurait donc des recettes publiques (accessibles aux visiteurs non connectés) et des recettes réservées aux patients.

De plus, madame Coupart désirerait qu’il y ait un système d’authentification sur son site, afin de proposer pour chaque patient les recettes supplémentaires (non visible pour les visiteurs) adaptées à son régime et à ses allergies alimentaires. En effet, un patient connecté ne devra pas voir les recettes qui ne correspondent pas à ses régimes alimentaires, ainsi que des recettes qui comportent des ingrédients dont il est allergique.

Elle veut que le site visiteur se conforme à la charte graphique et veut disposer d’un dashboard simple et fonctionnel pour gérer les patients, les recettes et les commentaires.

Elle veut une page qui informe comment contacter le cabinet : téléphone, adresse postale, adresse mail et horaires d’ouverture du cabinet.

Il faudra aussi une page politique de confidentialité et mentions légale pour se conformer à la législation en vigueur.

1. Cahier des charges, expression des besoins, ou spécifications fonctionnelles du projet

US1. Se connecter

Utilisateurs concernés : Administrateur, Patients

US2. Créer un patient

Utilisateurs concernés : Administrateur

- Il est important de préciser la liste des allergènes pouvant provoquer une réaction au patient.

- Plusieurs types de régime peuvent lui être associés.

US3. Ajouter une recette

Utilisateurs concernés : Administrateur

Une case doit pouvoir être cochée afin que cette recette soit seulement accessible aux patients.

US4. Voir les recettes

Utilisateurs concernés : Administrateur, Visiteurs, Patients

- Les visiteurs voient les recettes de base.

- Les patients accèdent aux recettes supplémentaires. De plus, un filtrage est fait automatiquement pour ne montrer que celles qui sont adaptées à son régime et allergènes à éviter.

- Les patients peuvent écrire un avis et donner une note (de 1 à 5). Pour plus de dynamisme, la soumission de l’avis ne devra pas recharger la page.

US5. Découvrir les services du cabinet

Plusieurs pages statiques devront être mises en place, et optimisées pour s’afficher correctement sur tout support :

- Une page d’accueil avec au moins une section "À propos" et “Mes services”,

- Une page de contact,

- Une page de mentions légales (Obligatoire depuis 2004),

- Une page de politique de confidentialité (Obligatoire depuis le RGPD en 2018).

1. Spécifications techniques du projet, élaborées par le candidat, y compris pour la sécurité et le web mobile

Les différentes technologies que j’ai utilisés pour ce projet :

- Environnement de travail :

Le site trello : pour l’organisation des taches à réaliser. J’ai fonctionné sur un modèle agile. Avec des users story créées à partir d’un story mapping (présent dans le dossier documentation du dépôt git).

<https://trello.com/invite/b/5pKYS4Gf/ATTI3e78f41fddda2487271feb2d9751edb9EE4470A2/4-coupartnutrition>

Le site figma : pour la maquette du site. Figma permet de faire des maquettes rapidement avec son système de composant.

L’outil de versionning Git et le dépôt distant gitHub. J’ai créé une branche dev que j’ai mergé après chaque fonctionnalité du cahier des charges. Cet outil a été très utile car j’ai deux ordinateurs, un fixe et un portable, cela m’a permis de passer facilement de l’un à l’autre et ainsi simulé un travail d’équipe.

Concernant l’IDE j’utilise VSC car il propose des extensions qui facilite le développement php comme intelliphense mais aussi l’utilisation de git et gitHub est facilitée.

-Partie Back :

J’ai utilisé Symfony qui permet la création d’une application web avec des lignes de commande. Ce framework propose de nombreux bundle pour gérer l’administration, l’authentification et la sécurité du site. Le système MVC clair permet de bien différencier : le rendu avec twig, les entités correspondent facilement à la base de données avec l’ORM doctrine, les controllers gèrent la logique via l’abstractController, les formulaires sont mis en place grâce à l’abstractType et enfin les repositories manipulent les requêtes à la base de données.

Le gestionnaire de package est composer, c’est un outil simple et fiable pour intégrer les packages nécessaires au fonctionnement de Symfony.

-Partie Front :

Le gestionnaire de package npm m’a permit d’installer les packages utiles pour le front.

J’ai installé le bundle web-pack encore, il gère la structure du front et est compatible avec le bundle easyAdmin qui gère l’administration du site.

Consternant le style, j’ai choisi bootstrap car sa grille est très puissante, l’apport du style grâce à des classes prédéfinies rend la gestion du style efficace tout en permettant la personnalisation avec un fichier .scss et les breakspoints optimisent le responsive.

Le langage twig hautement compatible avec Symfony permet de récupérer facilement les données des Controller et d’insérer de la logique pour conditionner l’affichage.

1. Description de la veille, effectuée par le candidat durant le projet, sur les vulnérabilités de sécurité

J’utilise Doctrine, il protège les requêtes “classiques” sur des objets.  
Pour les requêtes personnalisées, j’utilise la méthode setParameter. Ainsi, les requêtes personnalisées deviennent des “requêtes paramétrées”, et évitent les injections SQL. La validation des imputs avec des types variés permet la protection contre le Cross Site Scripting, twig permet aussi d’échapper les données en sortie. Par ailleurs l’authentification n’est possible qu’en HTTPS. Tous les inputs du site ne sont accessibles qu’aux patients ou à l’administrateur. Un patient ne peut être créé que par l’administrateur. La réinitialisation du mot de passe est aussi gérée par l’administrateur.

1. Description d’une situation de travail ayant nécessité une recherche, effectuée par le candidat durant le projet, à partir de site anglophone

Pendant le déploiement de l’application, j'ai fait les étapes suivante :  
1- git clone url de mon dépot git : qui a créer un dossier Coupart-Nutrition à la racine du server  
2- installer compser à la racine du server (j'en aurai besoin sur d'autre site)  
3-upload les dépendances depuis le dossier Coupart-Nutrition  
4-modifier mon fichier .env : pour mettre prod et database  
et quand j’ai voulu gérer voulu voir le rendu il manquait le dossier build dans le dossier public. Je me suis donc rendu dans la documentation symfony dans la rubrique How do i deploy My Encore assets ? J’ai donc appris qu’il n’y avait pas besoin d’installer les packages node.js en production qu’il fallait juste importer le dossier build manuellement.

1. Extrait du site anglophone, utilisé dans le cadre de la recherche décrite précédemment, accompagné de la traduction en français effectuée par le candidat sans traducteur automatique (environ 750 signes).

“There are two important things to remember when deploying your assets.

1) Compile Assets for Production

Optimize your assets for production by running:

./node\_modules/.bin/encore production

That will minify your assets and make other performance optimizations. Yay!

But, what server should you run this command on? That depends on how you deploy. For example, you could execute this locally (or on a build server), and use rsync or something else to transfer the generated files to your production server. Or, you could put your files on your production server first (e.g. via git pull) and then run this command on production (ideally, before traffic hits your code). In this case, you'll need to install Node.js on your production server.

2) Only Deploy the Built Assets

The only files that need to be deployed to your production servers are the final, built assets (e.g. the public/build directory). You do not need to install Node.js, deploy webpack.config.js, the node\_modules directory or even your source asset files, unless you plan on running encore production on your production machine. Once your assets are built, these are the only thing that need to live on the production server.”

Ma traduction :

Il faut absolument se rappeler de deux choses lors du déploiement de vos Assets :

1 Compiler les Assets pour la production :

Optimiser les assets pour le mode production en exécutant la commande suivante :

«  ./node\_modules/.bin/encore production »

Cela va compresser vos assets et améliorer les performances. Youpi !

Cependant, sur quel serveur devez-vous exécuter cette commande ? Cela dépend de votre méthode de déploiement. Par exemple, vous pourriez exécuter ceci localement (ou sur un serveur de build) et utiliser rsync ou un autre pour télécharger les fichiers générés vers votre serveur de production. Ou bien, vous pourriez d’abord placer vos fichiers sur votre serveur de production (par exemple via git pull) et ensuite exécuter cette commande en production (idéalement, avant que le trafic n'atteigne votre code). Dans ce cas, vous devrez installer Node.js sur votre serveur de production.

2 Déployer uniquement les ressources compilées :

Les seuls fichiers qui doivent être déployés sur vos serveurs de production sont les ressources finales et compilées (par exemple, le répertoire public/build). Vous n'avez pas besoin d'installer Node.js, de déployer webpack.config.js, le répertoire node\_modules ou même vos fichiers sources, sauf si vous prévoyez d'exécuter encore production sur votre machine de production. Une fois que vos ressources sont compilées, ce sont les seules choses qui doivent être présentes sur le serveur de production.

1. Autres ressources

- Composer :

https://getcomposer.org/download/

- Symfony CLI :

https://symfony.com/download

- Symfony :

https://symfony.com/

- Symfony webpack encore :

https://symfony.com/doc/current/frontend/encore/installation.html

- Symfony easyadmin :

https://symfony.com/bundles/EasyAdminBundle/

- UserCrudController pour le hash du mot de passe et confirmation :

https://dev.to/nabbisen/symfony-6-and-easyadmin-4-hashing-password-3eec

- Symfony vich uploader :

https://github.com/dustin10/VichUploaderBundle/blob/master/docs/index.md

- Icones du dashbord :

https://fontawesome.com

- Requête ajax avec axios :

https://axios-http.com/fr/docs/intro

- Rédaction de la page politique de confidentialité et mentions légales :

https://www.legalplace.fr/guides/politique-de-confidentialite-rgpd/

https://www.legalplace.fr/guides/mentions-legales/

1. Informations complémentaires

C’est mon premier projet Symfony en solo, je suis consciente que beaucoup de choses peuvent être améliorées, mais il faut bien rendre quelque chose à un moment ! Je me suis concentrer sur les fonctionnalités demandées dans le sujet mais d’autre fonctionnalités seraient intéressantes à développer comme un système de gestion du mot de passe oublié, améliorer la sécurité notamment ajouter des Assert sur les entités qui sont liées à des inputs et créer un formulaire de contact.