



DOSSIER-PROJET

Nom de naissance ▶ FÉLIX

Nom d'usage ▶

Prénom ▶ Jérémy

Adresse ▶ 70 chemin de la Foyère, 73800, CRUET, FRANCE.

Titre professionnel visé

Concepteur développeur d'application

Sommaire

1 - Remerciements	p. 3
2 - Introduction	p. 4
3 - Liste des compétences du référentiel	p. 5
4 - Conception du projet	p. 6
4.1 – Première idée et évolutions du projet	
4.2 - Personas	
4.3 – Charte graphique	
4.4 – Wireframe	
4.5 – User journeys	
4.6 – Prototype	
5 – Gestion de projet	p. 21
5.1 – Tableau KANBAN	
5.2 – Rétroplanning	
6 - Spécifications fonctionnelles	p. 25
6.1 – Fonctionnement général de l'application	
6.2 – Plan de site	
6.3 – Use cases	
6.4 – Modèle conceptuel de données	
6.5 – Modèle relationnel	
7 - Spécifications techniques	p. 34
7.1 – Technologies choisies	
7.2 – Architecture de l'application	
8 - Réalisation du projet	p. 39
8.1 – Développement	
8.2 – Difficultés rencontrées et axes d'améliorations dans le travail en équipe	
9 - Conclusion	p. 47

1 - REMERCIEMENTS

Je tiens à remercier toute l'équipe pédagogique de SIMPLON pour leur aide et leur accompagnement bienveillant tout au long de cette formation.

Et j'adresse un merci tout particulier à Thomas LAFORGES et à Anthony YOUSSEF, nos deux formateurs dévoués.

2 - INTRODUCTION

Dans le cadre du cursus “Concepteur / Développeur d’Applications DevOps” proposé par l’organisme de formation SIMPLON, j’ai pris en charge toutes les phases de conception et de réalisation de mon projet chef-d’œuvre intitulée “Les As de l’UX”.

Il s'agit d'une application qui permet principalement de mettre en relation des chargés de création ou de refontes de projets web (sites ou d’applications) avec les autres utilisateurs de l’application (amateurs et professionnels de l’UX design confondus) afin d'automatiser les tests utilisateurs et donc de recueillir plusieurs avis extérieurs et constructifs sous la forme d'une analyse globale, téléchargeable par le client sous la forme d'un PDF. Tout cela permet au client de garder la maîtrise de son projet en obtenant simplement des conseils constructifs de la part de plusieurs utilisateurs / testeurs de leur projet web.

A la fin du processus, certains utilisateurs sont récompensés pour leur commentaires. Cette méthode de détection de bug rémunéré repose sur les concepts du “Bug Bounty” (conçu à la base pour détecter des bugs dans le code), de l’amélioration continue et de l’intelligence collective.

J’ai choisi la thématique de l’UX Design car cela s’inscrit dans l’un de mes centres d’intérêt et c’est une notion à laquelle j’ai été sensibilisée lors mes précédents cursus de formation.

L’UX design (« User eXperience » en anglais) est une démarche visant à créer une expérience utilisateur optimale durant toute son interaction avec un produit, un service, une entreprise, etc.

L’UX design et l’UI design sont des disciplines différentes. Souvent mises en comparaison, elles n’ont fondamentalement pas le même objet d’étude :

- UX design (User eXperience design) = conception de l’expérience utilisateur.
- UI design (User Interface design) = conception d’interface utilisateur.

Pour expliquer les raisons pour lesquelles j’ai réalisé ce projet en autonomie, il faut savoir que j’ai d’abord commencer à travailler en binôme avec Arnaud LAFORGUE mais, comme mon concept de base n’était pas encore très bien défini, les formateurs lui ont conseillé de se lancer sur son projet et moi de me mettre avec lui ; Or, comme je ne me projetais pas vraiment dans son projet de générateur de CV à partir des réponses données à des QCM sur des sujets scientifiques, j’ai préféré développer mon idée d’application car je restais convaincu que le secteur de l’UX Design avait un grand potentiel pratique dans le monde du web. En résumé, travailler seul m’a demandé une plus grande organisation et une plus grande charge de travail que si j’avais été intégré à un groupe mais j’ai depuis longtemps l’habitude de mener des projets de A à Z.

Pour réaliser ce projet chef-d’œuvre, notre promotion a disposé de 6 semaines en tout. Et, puisque nous avons commencé le projet en cours de l’année scolaire, j’ai eu l’occasion de mettre en application progressivement les compétences acquises lors des TP et des briefs effectués (pour me permettre de savoir où chercher les informations plus rapidement, j’ai aussi énuméré des projets de référence qui m’ont ensuite servi de ressources pour reproduire la même démarche dans le code de mon projet personnel).

3 - LISTE DES COMPÉTENCES DU RÉFÉRENTIEL

1. Concevoir et développer des composants d'interface utilisateur en intégrant les recommandations de sécurité

- Maquetter une application.
- Développer une interface utilisateur de type desktop.
- Développer des composants d'accès aux données.
- Développer la partie front-end d'une interface utilisateur web.
- Développer la partie back-end d'une interface utilisateur web.

2. Concevoir et développer la persistance des données en intégrant les recommandations de sécurité

- Concevoir une base de données.
- Mettre en place une base de données.
- Développer des composants dans le langage d'une base de données.

3. Concevoir et développer une application multicouche répartie en intégrant les recommandations de sécurité

- Collaborer à la gestion d'un projet informatique et à l'organisation de l'environnement de développement.
- Concevoir une application.
- Développer des composants métier.
- Construire une application organisée en couches.
- Développer une application mobile.
- Préparer et exécuter les plans de tests d'une application.
- Préparer et exécuter le déploiement d'une application.

4 - CONCEPTION DU PROJET

4.1 - Première idée et évolutions du projet

A la base, je voulais faire un jeu de carte permettant de découvrir les lois de l'UX Design et, lorsque j'ai appris qu'il fallait concevoir et développer un projet chef-d'œuvre, j'ai voulu transformer ce concept initial pour en faire une application.

Voici mon cheminement de réflexion et les raisons pour lesquelles je n'ai pas retenu les options suivantes :

- Mise en relation contractuelle entre particuliers et professionnels ; j'ai écarté cette idée car il aurait fallu gérer le remplissage et la signature numérique d'un devis en ligne. Il y a aussi le risque que les acteurs contournent mon application pour ne pas payer la commission pour le service rendu. De plus, il y a déjà ce même service proposé par des géants du secteur comme Malt. De plus, cela me semblait difficile de mettre en place un "business plan" efficace avec ce mode de fonctionnement.

- Mise en relation entre élèves et professeurs en UX Design en proposant des formations et du coaching en ligne ; j'ai écarté cette idée car cela aurait nécessité de fournir des certifications diplômantes aux élèves et je n'aimais pas l'idée de rester purement dans un enseignement théorique, sans projet concret sur lequel travailler. De plus, il y a déjà ce même service proposé par des géants du secteur comme OpenClassroom.

- Génération aléatoire d'images de wireframes pour trouver de l'inspiration ; j'ai écarté cette idée car je me suis rendu compte que cela risquait d'être assez abstrait puisque l'on ne connaît pas les détails du projet ayant motivé le wireframe.

Par la suite, j'ai finalement simplifié mon application en ne gardant que le côté "Analyse UX" de projets web existants par des avis extérieurs, eux-même utilisateurs de la plateforme.

De plus, l'application "Les As de l'UX" se démarque des autres produits sur le marché car je n'ai pas encore trouvé d'outil permettant de produire des tests utilisateur automatisés spécifiquement autour des notions-clés de l'UX Design.


4.2 - Personnas

Une fiche persona est un document qui fait partie de la stratégie marketing et qui sert à cibler plusieurs clients-types (personnages fictifs) qui sont autant d'utilisateurs potentiels du produit en cours de conception (par exemple, ici, une application) afin de coïncider leurs exigences (explicites ou implicites) et ainsi d'orienter la prise de décision concernant les solutions qui lui seront proposées.


Les principaux intérêts d'élaborer des fiches persona sont les suivants :

- Améliorer le taux de conversion.
- Optimiser l'expérience utilisateur.
- Cibler le contenu du site et des publicités, etc.

Les 3 fiches personas suivantes m'ont permises de mieux cerner les besoins concrets des futurs utilisateurs de mon application afin de me mettre à leur place lors des différentes phases de conception du produit en gardant à l'esprit leur objectifs, leur problématiques et leurs traits de personnalité (je n'ai gardé que les critères les plus pertinents en lien avec l'utilisation du produit) :

	Biographie <p>Juliette se forme actuellement à créer des sites web et des applications pour en faire son métier. Elle est très sensible aux questions écologiques et aime passer du temps avec ses animaux de compagnie. D'ailleurs, durant son temps libre, elle est bénévole dans la SPA de son quartier et doit concevoir leur application (ce qui lui permet de s'exercer).</p>	Personnalité <ul style="list-style-type: none"> • Curieuse, • Créative, • Manque de confiance en elle, • Timide.
Situation <ul style="list-style-type: none"> • Prénom : Juliette • Age : 21 ans • Localisation : Saint Martin d'Hères • Job : Etudiante en école de développement web • Salaire : 700€ 	Objectifs et besoins <ul style="list-style-type: none"> • Obtenir des conseils pratiques pour améliorer la qualité de l'application de son association, • Appliquer concrètement pour de vrais projets les connaissances théoriques qu'elle apprend à son école et ainsi augmenter son esprit critique sur les bonnes pratiques de l'UX Design et sur le multimédia en général, • Mieux comprendre comment fonctionne la psychologie humaine. 	Problèmes <ul style="list-style-type: none"> • A du mal à demander de l'aide aux autres directement, • N'a pas l'impression de faire beaucoup de progrès dans sa scolarité même si elle aime beaucoup apprendre, • A des ressources financières assez limitées.

Fiche persona n°1

	Biographie <p>Alexandre est à la tête d'une grande équipe (composée de salariés et de bénévoles) dont le but est d'organiser des activités au sein de la banlieue lyonnaise. Il a souvent beaucoup d'idées intéressantes n'hésite pas à demander de l'aide aux autres association de la région. Durant son temps libre, il aime voyager et rencontrer de nouvelles personnes.</p>	Personnalité <ul style="list-style-type: none"> • Serviable, • Franc, • Mauvaise mémoire, • Impatient.
Situation <ul style="list-style-type: none"> • Prénom : Alexandre • Age : 42 ans • Localisation : Lyon • Job : Responsable d'une association subventionnée par la région • Salaire : 1800€ - 2200€ 	Objectifs et besoins <ul style="list-style-type: none"> • Encadrer la refonte du site web de son association pour la rendre plus qualitative. • Garder la mission de refonte en interne de son équipe pour ne pas risquer de dénaturer l'image de marque de son association. • Avoir le feedback de spécialistes UX afin d'orienter le webdesigner (qui n'est pas designer UX) de son équipe dans ses choix de conception via le CMS WordPress. 	Problèmes <ul style="list-style-type: none"> • Manque d'expérience dans le domaine de l'informatique, • Est débordé en raison de ses nombreuses responsabilités et n'a pas le temps de faire de la veille sur l'UX Design, • Il ne sait pas à qui s'adresser pour la refonte de son site web très volumineux.

Fiche persona n°2

	Biographie <p>Kévin a travaillé 15 ans dans plusieurs agences de communication. A présent, il souhaite mettre à profit son expérience pour donner des cours dans le secteur du multimédia et de l'IHM. Il est exaspéré par les sites et les applications où l'expérience utilisateur est négligée. Il aime se fixer des challenges pour progresser techniquement.</p>	Personnalité <ul style="list-style-type: none"> • Autonome, • Positif, • Altruiste, • Stressé.
Situation <ul style="list-style-type: none"> • Prénom : Kévin • Age : 38 ans • Localisation : Grenoble • Job : Designer UX et formateur en freelance • Salaire : 2200€ - 2800€ 	Objectifs et besoins <ul style="list-style-type: none"> • Partager ses connaissances en UX Design pour aiguiller la création et la refonte de projets numériques concrets et motivants, • Mieux argumenter ses choix techniques face à ses clients ou bien vis-à-vis de ses élèves, • Confronter son point de vue à celui des autres, • Trouver des sources d'inspirations pour rédiger un livre qui liste les bonnes pratiques sur l'UX Design et aussi des TP proches de la réalité. 	Problèmes <ul style="list-style-type: none"> • Manque de clientèle pour sa profession de Designer UX, • N'a plus de mentorat depuis environ 2 mois, • Manque de visibilité pour l'avenir de son activité de freelance car il y a beaucoup de concurrents dans ce secteur d'activité.

Fiche persona n°3

4.3 - Charte graphique

Une charte graphique est un document qui définit et décrit l'ensemble des éléments visuels utilisés pour représenter une entreprise, une marque ou un projet. Elle établit les règles et les normes graphiques à suivre afin d'assurer une cohérence visuelle sur tous les supports de communication, tels que les sites web, les réseaux sociaux, les brochures, les affiches, etc.

La charte graphique inclut généralement des éléments tels que les couleurs principales et secondaires, les typographies, les logos, les icônes, les illustrations, les schémas, ainsi que les règles de mise en page et d'utilisation de ces éléments. Elle vise à garantir une identité visuelle forte et reconnaissable, facilitant ainsi la mémorisation de la marque par le public cible.

Logotype

Tout d'abord, voici le logotype de mon application chef-d'œuvre :



*Logotype de l'application
"Les As de l'UX"*

La forme générale (s'insérant dans un carré) est un blason qui permet de délimiter joliment le logo et rappelle aussi l'harmonie et l'équilibre que l'UX Design rend possible s'il est mis en pratique dans les projets web.

Au centre, le diamant représente les critères de qualité permettant d'améliorer l'expérience des utilisateurs.

Palette de couleurs

Voici ensuite la palette de couleurs que j'ai intégré à mon prototype ainsi qu'à la partie frontend de mon projet :

<div> PALETTE ET CONTRASTES COLORIMÉTRIQUES </div> <div> > 4.50 : contraste suffisant pour toutes les tailles de textes. 3.00 - 4.50 : contraste suffisant pour les textes en gras de plus de 14 points et pour les textes de plus de 18 points. </div>							
GRIS ANTHRACITE <div> <div>#303030</div> <div>#303030</div> <div>#303030</div> <div>#303030</div> <div>#303030</div> <div>#303030</div> </div> <div> <div>#F5F5F5</div> <div>#C0C0C0</div> <div>#F0C300</div> <div>#FF7F50</div> <div>#01D758</div> <div>#00CED1</div> </div> <div> <div>12.11</div> <div>7.25</div> <div>7.86</div> <div>5.28</div> <div>6.83</div> <div>6.76</div> </div>						GRIS MAT <div>#696969</div> <div>#F5F5F5</div> <div>5.04</div>	GRIS ARGENT <div>#C0C0C0</div> <div>#303030</div> <div>7.25</div>
BLANC FUMÉE <div> <div>#F5F5F5</div> <div>#F5F5F5</div> <div>#F5F5F5</div> <div>#F5F5F5</div> <div>#F5F5F5</div> <div>#F5F5F5</div> </div> <div> <div>#303030</div> <div>#696969</div> <div>#DE691E</div> <div>#D61E33</div> <div>#096A09</div> <div>#067790</div> </div> <div> <div>12.11</div> <div>5.04</div> <div>3.13</div> <div>4.71</div> <div>6.26</div> <div>4.76</div> </div>						MARRON CHOCOLAT <div>#DE691E</div> <div>#F5F5F5</div> <div>3.13</div>	JAUNE AMBRE <div>#F0C300</div> <div>#303030</div> <div>7.86</div>
ROUGE VIF <div>#D61E33</div> <div>#F5F5F5</div> <div>4.71</div>	ROSE CORAIL <div>#FF7F50</div> <div>#303030</div> <div>5.28</div>	VERT BOUTEILLE <div>#096A09</div> <div>#F5F5F5</div> <div>6.26</div>	VERT ÉMERAUDE <div>#01D758</div> <div>#303030</div> <div>6.83</div>	BLEU PAON <div>#067790</div> <div>#F5F5F5</div> <div>4.76</div>	TURQUOISE FONCÉ <div>#00CED1</div> <div>#303030</div> <div>6.76</div>		

Palette et contrastes colorimétriques

Pour une question d'accessibilité vis-à-vis des personnes malvoyantes, je n'ai associé que les couleurs présentant un contraste colorimétrique suffisant d'après les règles pour l'accessibilité des contenus web définies par le « Web Content Accessibility Guideline » (WCAG).

Pour limiter le nombre de choix de combinaison possible, j'ai repris les codes couleur d'un courant artistique, à savoir : l'art déco (c'était un mouvement très à la mode dans les années 20-30).

Ma couleur principale est la teinte "bleu paon" qui se rapporte au monde professionnel et au calme. Et la teinte secondaire est le "marron chocolat" qui permet de mettre en avant les titres principaux de l'application.

J'ai rajouté une déclinaison claire pour chaque couleur, c'est afin de pouvoir gérer le mode sombre dans second temps (tout en gardant un contraste suffisant avec le noir).

Pour reposer les yeux des utilisateurs, je n'ai pas utilisé de blanc et de noir à 100% mais des teintes plus douces, à savoir le "blanc fumée" et le "gris anthracite".

Chacune des autres couleurs a aussi un rôle indicatif et fonctionnel (que j'utilise notamment pour les boutons et les toasts) :

- Information : "Bleu paon" / "Turquoise foncé".
- Succès : "Vert bouteille" / "Vert émeraude".
- Avertissement : "Marron chocolat" / "Jaune ambre".
- Danger : "Rouge vif" / "Rose corail".
- Invalide : "Gris mat" / "Gris argent".

Polices d'écriture

Enfin, voici les deux polices d'écritures complémentaires que j'ai intégré à mon prototype ainsi qu'à la partie frontend de mon projet :



Polices d'écriture choisies

La police de caractères "Metropolis" est utilisée pour les différents niveaux de titres et pour le texte des boutons tandis que la police "IBM Plex Serif" est utilisée pour le texte courant (car les caractères avec empâtement rendent généralement les paragraphes plus lisibles).

Iconographie

J'ai utilisé un set d'icônes prêt à l'emploi depuis le site « iconify.design » pour illustrer les différentes informations présentes dans mon application. Voici les icônes dont je me suis servies (ou que je prévois de me servir) dans le prototype et dans l'application :

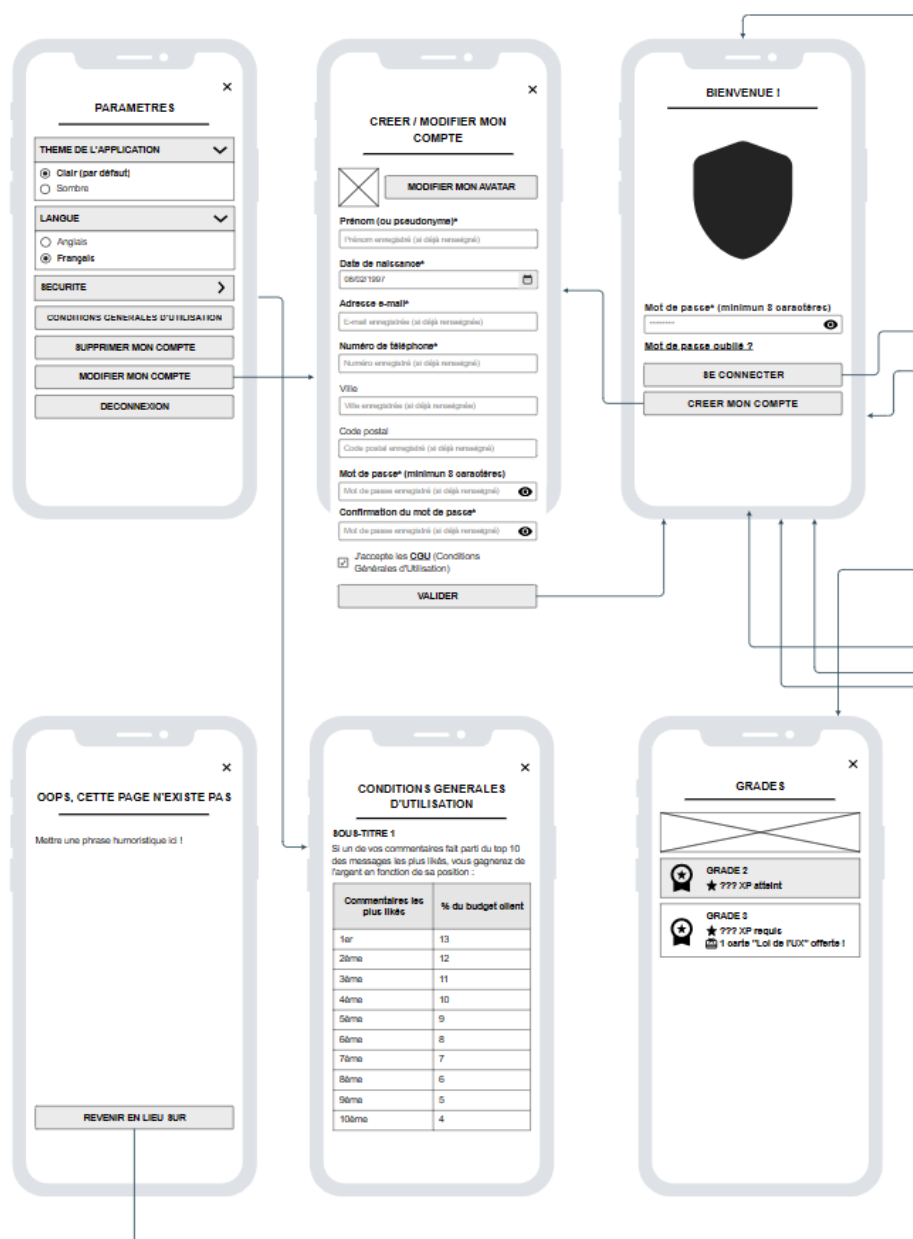


Extrait du set d'icônes "Carbon"

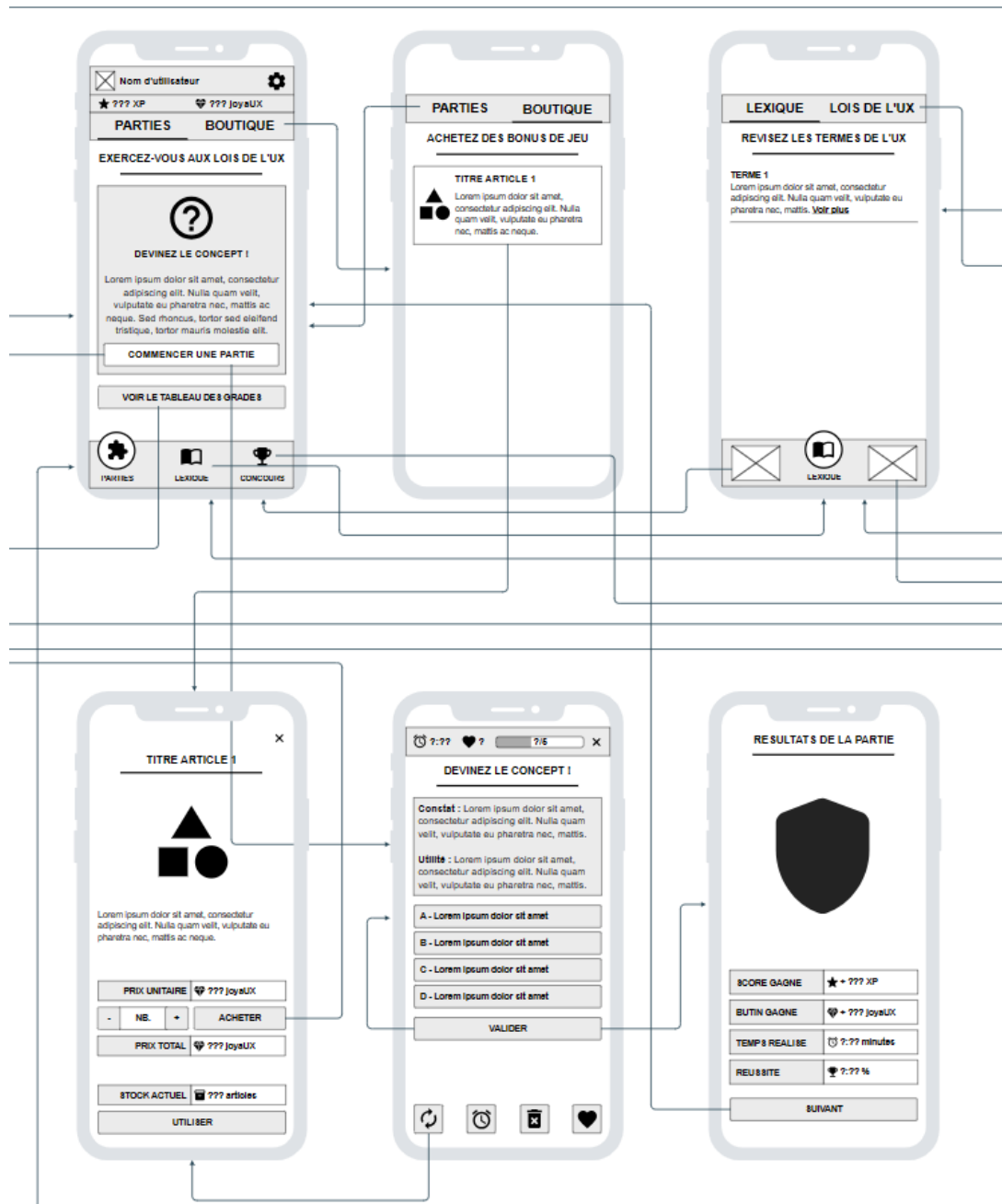
4.4 - Wireframe

Un wireframe est une représentation simplifiée de l'interface d'un produit numérique (sans application de la charte graphique) dans le but de valider le contenu des pages, la navigation entre celles-ci ainsi que de l'expérience utilisateur.

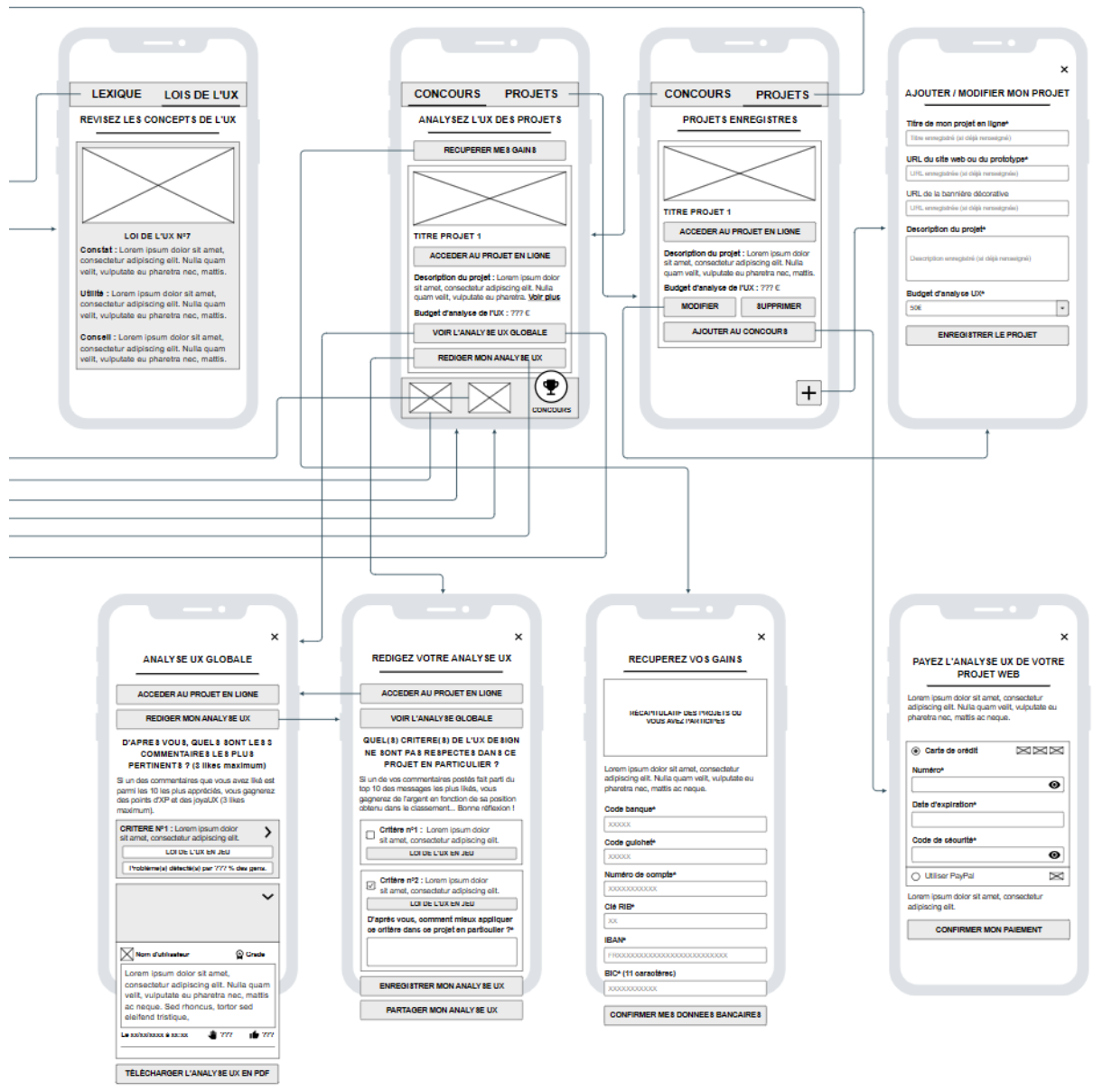
Voici le wireframe de la première version de mon application (réalisé sur « Moqups ») :



Pages du wireframe de la première version de l'application (partie 1)



Pages du wireframe de la première version de l'application (partie 2)



Pages du wireframe de la première version de l'application (partie 3)

L'inconvénient d'avoir utilisé Moqups en version gratuite est qu'il y a une limite de 400 éléments et seulement 2 projets maximum. Ce qui m'a fait perdre du temps pour réfléchir à quels éléments étaient les plus importants à conserver ou non (puisque je ne pouvais pas tous les intégrer).

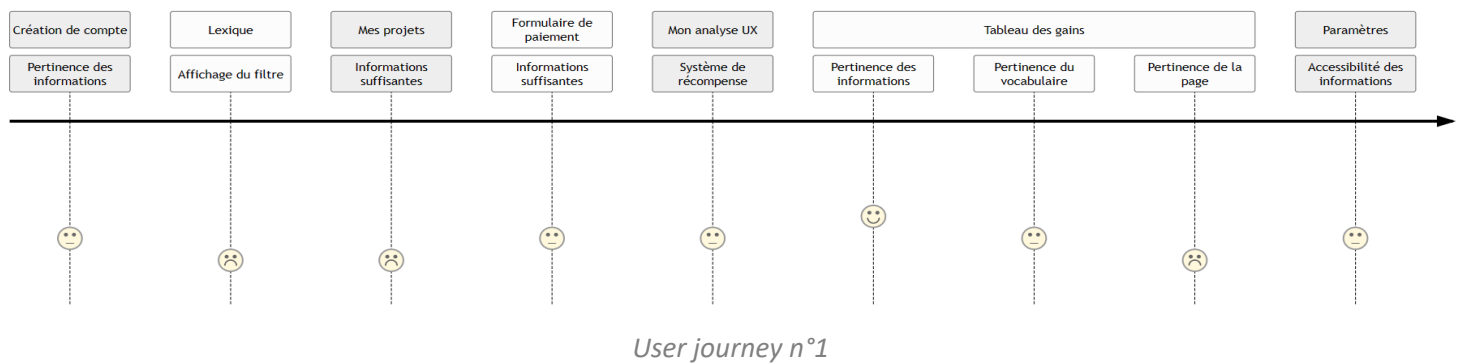
4.5 - User journeys

Un user journey est un schéma qui permet de visualiser les remarques et suggestion d'amélioration des utilisateurs suite à leur "navigation" sur chaque page du wireframe (et ainsi envisager des changements pertinents le tût possible dans la phase de conception afin d'économiser du temps et de l'argent).

Voici les 2 user journeys que j'ai réalisé avec la collaboration d'Anthony YOUSSEF et d'Élodie BOTTON :

User journey n°1

User journey n° 1 (avec Anthony YOUSSEF)



Voici les retours reçus d'Anthony YOUSSEF lors de son observation du wireframe page par page :

Création de compte :

- "Tout le monde n'a pas envie d'un avatar personnalisé, ni de répondre à la façon dont on a découvert l'application, ni de donner son numéro de téléphone (l'adresse mail est suffisante être contacté)."

Lexique :

- "Le filtre n'est pas nécessaire car il n'y aura probablement pas beaucoup de termes d'UX à parcourir et on peut simplement les ranger par ordre alphabétique sur une seule page."

Mes projets :

- "Il manque une bannière descriptive du projet (comme pour les briefs sur simplonline)."
 - "Il manque un bouton 'ajouter un projet' ou un bouton '+' (flottant en bas à droite) pour rediriger l'utilisateur sur une page dédiée afin de séparer le formulaire de nouveau projet avec les projets déjà référencés."

Formulaire de paiement :

- "Il manque une 3ème possibilité : on peut aussi payer par virement bancaire (où il faudra renseigner le RIB)."

Mon analyse UX :

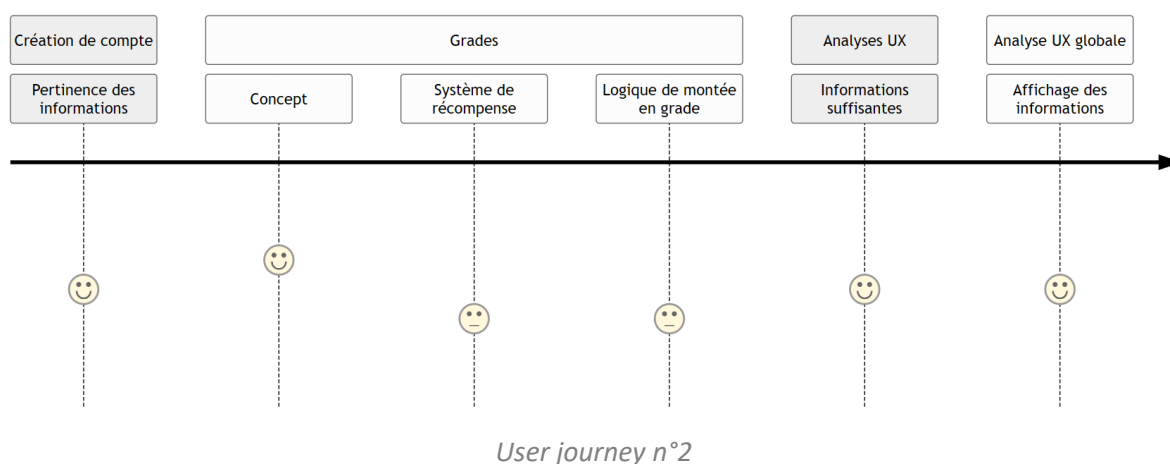
- Il faut motiver l'utilisateur à partager leur analyse UX d'un projet en lui donnant des points immédiatement après l'envoi (et lui indiquer dès le début ce qu'il a à gagner en le faisant).

Tableau des gains :

- "Le mot "gain" fait trop penser au vocabulaire d'un casino."
- "Supprimer la troisième colonne 'calcul en €' (pas besoin d'avoir le montant exact à remporter pour chaque place dans le classement ; le budget de l'analyse UX collective se suffit à lui-même)."
- "Intégrer les informations de cette rubrique dans les CGU."

Paramètres :

- "Au lieu de rediriger l'utilisateur sur une autre page à chaque fois qu'il clique sur un bouton, ce serait plus direct d'avoir un menu déroulant pour accéder aux détails de chaque paramètre."

User journey n°2**User journey n°2 (avec Elodie BOTTON)**

Voici les retours reçus d'Élodie BOTTON lors de son observation du wireframe page par page :

Création de compte :

- "Il serait pertinent de demander la date de naissance des utilisateurs pour connaître les tranches d'âge majoritaires."

Grades :

- "Offrir quelques cartes d'UX en fonction du grade obtenu pour récompenser l'utilisateur de son investissement et lui donner envie de s'investir davantage (en publiant une analyse UX, en déposant un projet ou en faisant de la pub de bouche à oreille)."
- "Conditionner la montée en grade par un nombre d'analyse UX ou de commentaires laissés (peut-être en vérifiant si ces derniers sont qualitatifs ou pas)."

Analyses UX :

- "Ajouter un bouton pour séparer les 2 actions possibles : "Rédiger une analyse UX" et "Voir l'analyse UX globale".
- "Il faut que l'utilisateur comprenne que rédiger des commentaires constructifs (sur des vrais projets de création ou de refontes de sites web) va le faire progresser en augmentant son esprit critique sur l'UX Design."

Analyse UX globale :

- "Rendre public le grade actuel de l'utilisateur dans les commentaires qu'il a laissés (afin que le client fasse plus attention aux utilisateurs de longue date de l'application lorsqu'il doit sélectionner les 10 meilleures analyses UX)."

4.6 - Prototype

Le prototype est une maquette dynamique qui permet de tester le fonctionnement d'une application et son utilisabilité du point de vue des utilisateurs de façon plus réaliste qu'un wireframe.

Voici quelques captures d'écran du prototype de mon application (réalisé sur Figma) :



Pages du prototype (partie 1)

CRÉATION / MODIFICATION DE MON COMPTE

Pseudonyme *

Jéjé3873!

Date de naissance

08/02/1997

Adresse e-mail *

jeremyfelix41@gmail.com

Numéro de téléphone *

0630770590

Ville

Grenoble

Code postal

38000

Mot de passe * (minimum 8 caractères dont au moins 1 majuscule, 1 chiffre et 1 caractère spécial)

Confirmation du mot de passe *

RIB (pour recevoir des gains automatiquement si un de vos commentaires est sélectionné par le dépositaire d'un projet)

☒ J'accepte les **CGU** (Conditions Générales d'Utilisation)

VALIDER LE FORMULAIRE

CRÉATION / MODIFICATION DE MON PROJET

Titre *

Site web de l'association Entropie

URL du projet en ligne *

http://projet.com

AJOUTER UNE IMAGE DE BANNIÈRE

Description du projet *

Description (...)

Rajouter un scénario utilisateur à analyser *

Aller chercher une notice

Scénario 1

Scénario 2

Scénario 3

Rajouter une page à analyser*

Accueil

VALIDER LA PAGE

Page 1

Page 2

Page 3

Page 4

BUDGET*

50€

100€

150€

200€

250€

Nombre de participants maximum : ???

FAVORIS

Oui

Non

VALIDER LE FORMULAIRE

PROJET 42

ACCÉDER AU PROJET EN LIGNE

Description :

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla quam velit, vulputate.

Scénario utilisateur à vérifier :

☒ Scénario 1

☐ Scénario 2

☐ Scénario 3

Pages du projet à vérifier :

☒ Page 1

☐ Page 2

☐ Page 4

Budget d'analyse UX collective : ??? €

Commentaires restants : ??? / ???

VOIR L'ANALYSE UX

PARTAGER LE PROJET (+ 5 POINTS)

AJOUTER LE PROJET AUX FAVORIS

OPTIONS DE PROJET PERSONNEL

Cette partie est réservée au créateur d'un projet non publié.

PROCÉDER AU PAIEMENT EN LIGNE

MODIFIER MON PROJET

SUPPRIMER MON PROJET

ANALYSE UX

PROJET 42

VOIR LA DESCRIPTION DU PROJET

ACCÉDER AU PROJET EN LIGNE

CATÉGORIES DE BUGS

PROBLÈMES DE NAVIGATION

PROBLÈMES D'INTERFACE

PROBLÈMES D'ACCESSIBILITÉ

PROBLÈMES DE PERFORMANCE

Choisissez une catégorie puis :

1. Lisez les commentaires laissés par la communauté pour ne pas répéter la même chose qu'eux

Choisissez un banni et pour être gagnant de l'argent si un de ses commentaires est dans le top 5 des préférences du client !

ASTUCES

Pour vous permettre de détecter plus facilement d'éventuels problèmes, vous pouvez vous aider :

- Des cartes des lois de l'UX que vous avez déjà débloquées avec vos points de jeu
- Des questions de réflexion liées à chaque sous-rubrique de problèmes à vérifier (une réponse positive à l'une d'entre elles signifie que la plateforme a un bug qu'il serait bien de signaler au client)
- Des analyses UX globales rapportant les remarques laissées sur d'autres projets

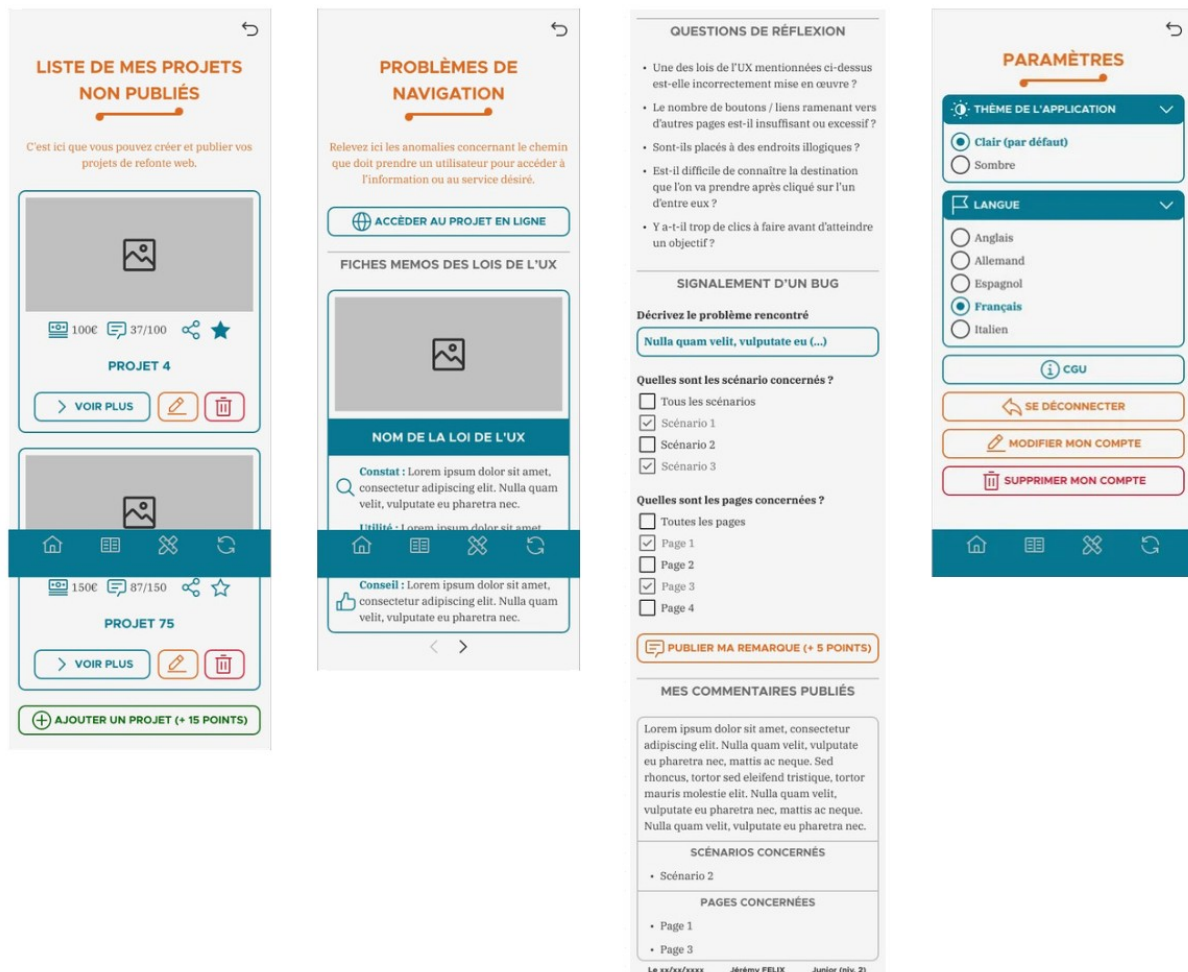
PARTIE ADMINISTRATEUR

Cette partie est réservée au client qui a déposé ce projet.

Pour avoir la possibilité de télécharger le PDF de l'analyse UX globale, **parcourez les commentaires des utilisateurs dans les différentes catégories de problèmes** pour désigner les 5 commentaires qui vous semblent les plus pertinents.

TÉLÉCHARGER LE PDF DE L'ANALYSE

Pages du prototype (partie 2)



Pages du prototype (partie 3)

Les principales différences entre le wireframe et le prototype sont que :

- La charte graphique a été appliquée sur tous les éléments d'interface.
- Il y a une page d'accueil qui explique les intérêts d'utiliser l'application pour les différents publics concernés.
- Il y a une page de description plus détaillée pour chaque projet publié.
- Les pages et les fonctionnalités liées aux QCM sur les lois de l'UX ainsi que la boutique ont disparus pour privilégier les éléments liés à l'analyse UX pour de vrais projets.
- La page d'analyse UX est désormais divisée en 4 catégories (navigation, interface, accessibilité et performance) au lieu d'une checklist très guidée autour des lois de l'UX (cela permet de moins brider les tests utilisateur pour augmenter le nombre d'anomalies remontées).
- Ce ne sont plus les utilisateurs qui votent pour leurs commentaires préférés (avec un système de likes) mais uniquement le client (qui procède à un classement à la fin d'une analyse globale) ; cela permet d'éviter le risque de fraude de personnes qui se créeraient automatiquement plusieurs compte en attribuant des likes à leurs propres commentaires (ce programme s'appelle le "botting").
- La partie sur le paiement a été simplifiée (il n'y a que le RIB qui a été conservé).

5 - GESTION DE PROJET

5.1 - Tableau KANBAN

Pour organiser mon travail et la gestion des priorités (en se fixant des délais à tenir), j'ai utilisé la fonctionnalité du tableau KANBAN proposée par l'outil de gestion de projet « Trello » :

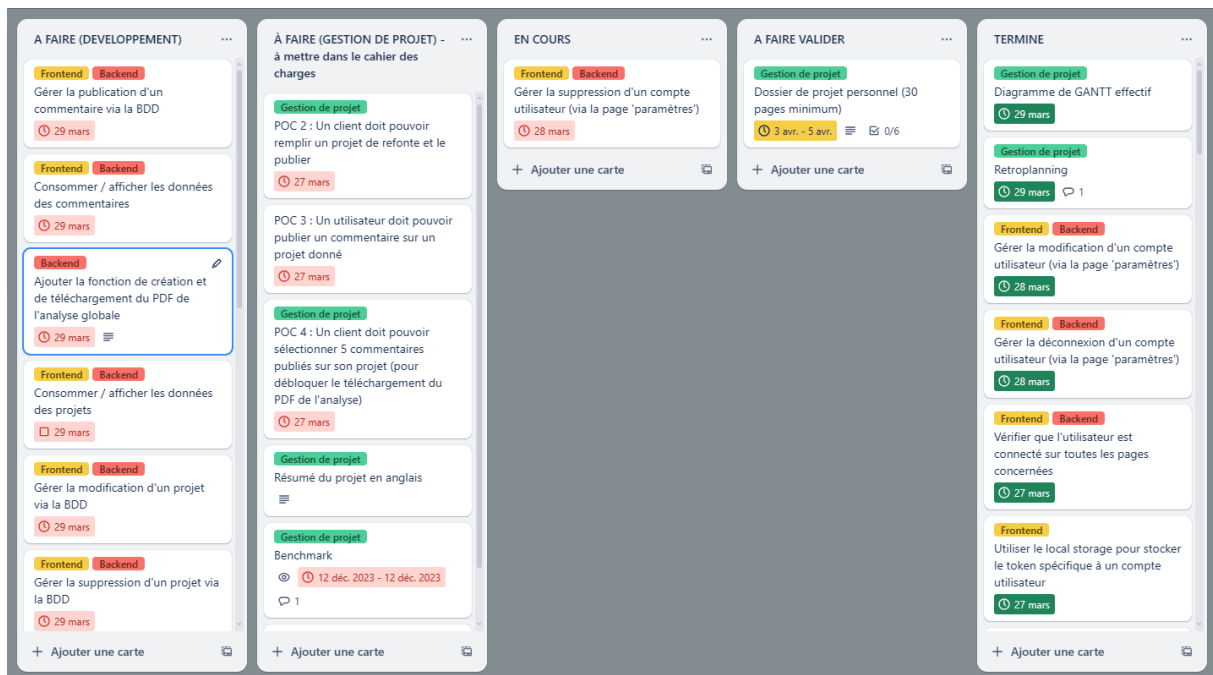


Tableau KANBAN du 05-04-2024

J'ai réparti les tickets (c'est-à-dire les tâches à réaliser) en les rangeant dans plusieurs colonnes :

- A FAIRE (développement).
- A FAIRE (gestion de projet).
- EN COURS.
- A FAIRE VALIDER.
- TERMINE.

J'ai également classé mes tickets en plusieurs catégories (appelées "étiquettes") :

- Gestion de projet
- Frontend.
- Backend.
- DevOps / Tests.
- Base de données.
- Hébergement / déploiement.

5.2 - Rétroplanning

Tout au long du déroulement du projet, à la fin de chaque sprint (c'est-à-dire la charge de travail répartie sur une période de temps définie), j'ai pris l'habitude de rédiger un rétroplanning pour faire le bilan sur l'accomplissement des tâches planifiées ainsi que sur les axes d'amélioration de la gestion de projet à prévoir lors des prochains sprints.

Sprint du 11-03-2024 au 15-03-2024

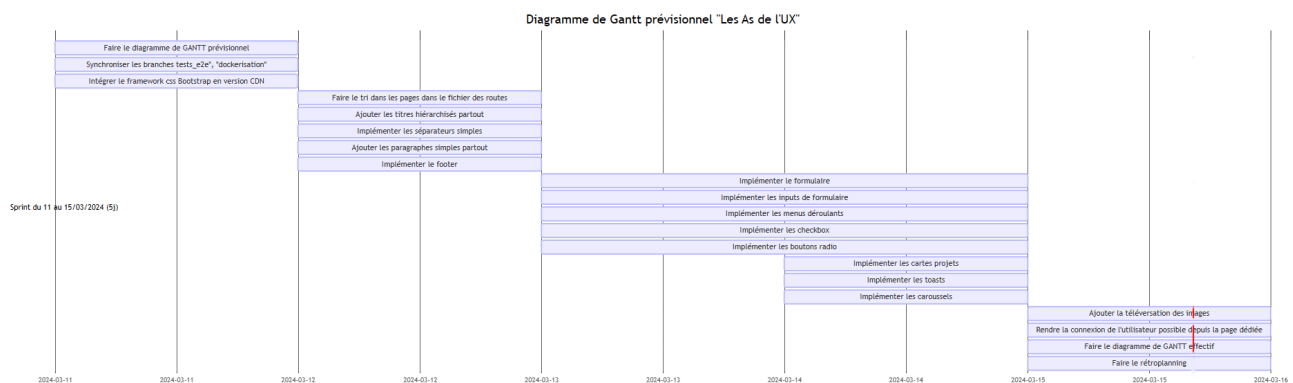


Diagramme de GANTT prévisionnel du sprint du 11-03-2024 au 15-03-2024

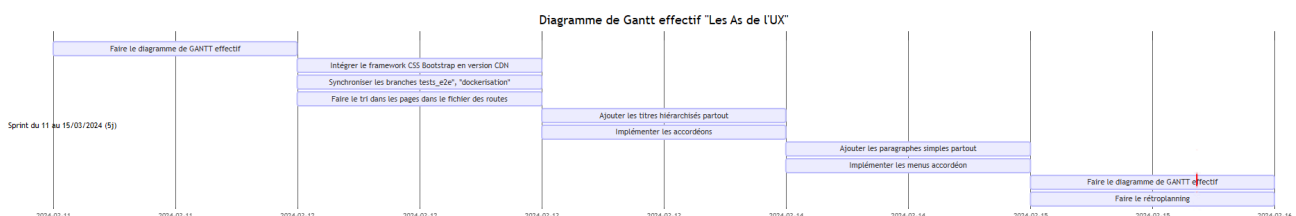


Diagramme de GANTT effectif du sprint du 11-03-2024 au 15-03-2024

Problèmes réglés depuis le dernier sprint

- J'ai fait des commits beaucoup plus régulièrement, ce qui m'a permis de moins devoir réfléchir à leur nommage et aussi d'avoir un historique plus détaillé des différentes versions de mon projet.
- J'ai pris l'habitude de créer une branche par ticket, ce qui m'a permis de me fixer des petits objectifs clairement identifiés.
- Je me suis concentré sur la partie frontend de mon projet et cela m'a beaucoup plu car je pouvais vérifier visuellement le résultat dans le navigateur.

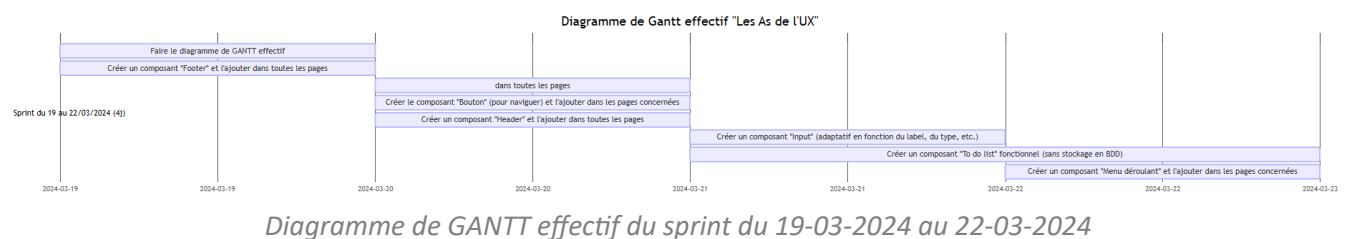
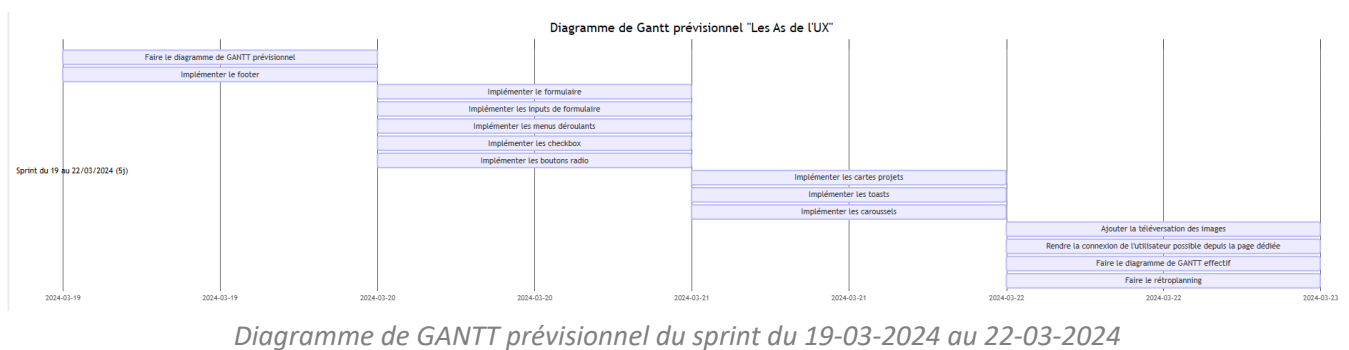
Problèmes rencontrés

- Mon diagramme de GANTT prévisionnel était trop ambitieux.
- J'étais très fatigué le vendredi et j'avais donc du mal à me concentrer sur mon travail.
- J'ai fait du CSS en même temps que l'implémentation des composants de Shadcn (même si j'avais prévu de le faire dans un second temps).
- J'ai eu du mal à prendre du recul sur mon travail lorsque je rencontrais une difficulté (et mon stress se faisait ressentir autour de moi).

Déductions pour la suite du projet

- Programmer moins de choses pour ne pas être frustré.
- Créer des tickets plus petits si besoin (pour avoir l'impression de progresser chaque jour).
- Essayer de se limiter aux tickets définis pour le sprint en cours (et de ne pas trop en rajouter en cours de route).
- Consacrer le vendredi après-midi à la rédaction du rétroplanning et au diagramme de GANTT de la semaine suivante.

Sprint du 11-03-2024 au 15-03-2024



Problèmes réglés depuis le dernier sprint

- J'ai presque réussi à faire tout ce que j'avais indiqué dans mon diagramme de GANTT prévisionnel (planifié en début de sprint).
- J'ai bien avancé sur la partie frontend de mon application (fidèlement à mon prototype).
- J'ai fait des commits plus régulièrement (à chaque fois que je produis du code fonctionnel).

Problèmes rencontrés

- J'étais absent le lundi 18/03/2024 (en raison notamment d'un entretien d'embauche le matin), ce qui m'a fait prendre du retard par rapport à l'avancement du projet.
- Les tickets définis sur mon tableau KANBAN n'étaient pas assez précis.
- J'ai trouvé que les composants générés par la bibliothèque de « Shadcn » étaient souvent compliqués à comprendre et donc à adapter (surtout pour changer leur style).
- J'ai passé beaucoup de temps à régler des problèmes de style (alors que c'est un chantier secondaire au vue des exigences attendues dans le cadre de ce projet).

Déductions pour la suite du projet

- Indiquer une condition d'arrêt précise pour chacun des tickets définis sur mon tableau KANBAN
- Faire des composants manuellement pour les éléments simples à concevoir et n'utiliser une bibliothèque React uniquement pour créer les composants plus complexes (menus déroulants, carrousels, etc.).
- Se concentrer davantage sur les scénarios utilisateurs essentiels au fonctionnement de l'application (réserver l'ajout du style et du vrai texte ultérieurement).

6 - SPÉCIFICATIONS FONCTIONNELLES

6.1 - Fonctionnement général de l'application

Voici un schéma avec les principales étapes permettant à l'application de fonctionner correctement :

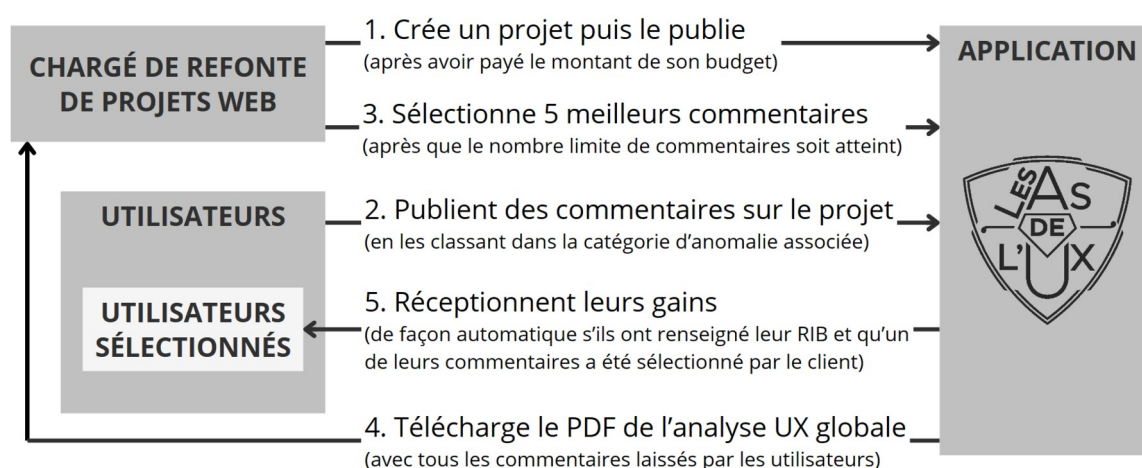


Schéma du fonctionnement de l'application "Les As de l'UX"

Voici les différentes étapes qui permettent à l'application de fonctionner normalement :

1. D'abord, le dépositaire d'un projet (client) renseigne quelques indications sur le projet web à refondre ou le prototype (pour donner envie aux autres de s'impliquer et faciliter les tests utilisateurs qui suivront). Puis, le client paye une somme d'argent pour rendre son projet accessible aux utilisateurs de l'application (ce montant déterminera le nombre maximal d'analyses UX que les utilisateurs pourront publier et qui, une fois atteint, cela mettra fin à l'analyse UX collective) ; l'argent est versé sur un compte dédié "Les As de l'UX" pour pouvoir toucher une commission dessus et de reverser l'argent à certains utilisateurs (comme pour un site de mise en relation classique).

2. Ensuite, les autres utilisateurs peuvent choisir un projet et effectuer une analyse UX en classant les anomalies et les axes d'améliorations repérés en 4 catégories (navigation, accessibilité, interface graphique et performance) afin de gagner des points (avec lesquels on peut monter en grade et débloquer des fiches mémos pédagogiques sur les lois de l'UX).

3. Une fois le nombre d'analyses UX maximales atteint, le client est informé par un email puis, sur l'application, il parcourt la liste des critiques de son site et sélectionne les 5 qui lui semblent les plus pertinentes selon lui pour l'aider dans sa refonte de son site ou de son application.

4. En effectuant ce classement, le client peut télécharger le PDF de l'analyse UX globale pour s'en servir concrètement pour son projet.

5. Les 5 commentateurs correspondants reçoivent une partie du budget initialement versé par le client et réparti comme suit :

- 1er : 27%.
- 2ème : 22%.
- 3ème : 17%.
- 4ème : 12%.
- 5ème : 7%.
- Marge dégagée pour "Les As de l'UX" : 15%.

Moyens de gagner des points

Il est possible de collecter des points tout au long de l'utilisation de l'application en effectuant une des actions suivantes :

- Créer un nouveau projet de refonte.
- Partager un projet de refonte (publié).
- Publier une analyse UX.
- Arriver dans le top 5 des commentaires les plus appréciés par le client.

Chaque semaine, tous les utilisateurs perdent un certain nombre de points d'XP (pour rétrograder au bout d'un certain temps s'ils ne s'entraînent pas régulièrement).

A quoi servent les points ?

Avec les points d'UX, on peut :

- Monter en grade (qui est affiché lorsque l'on publie un commentaire).
- Débloquer des cartes "Lois de l'UX" (de façon automatique lorsque l'on atteint le niveau supérieur).

Description des pages principales

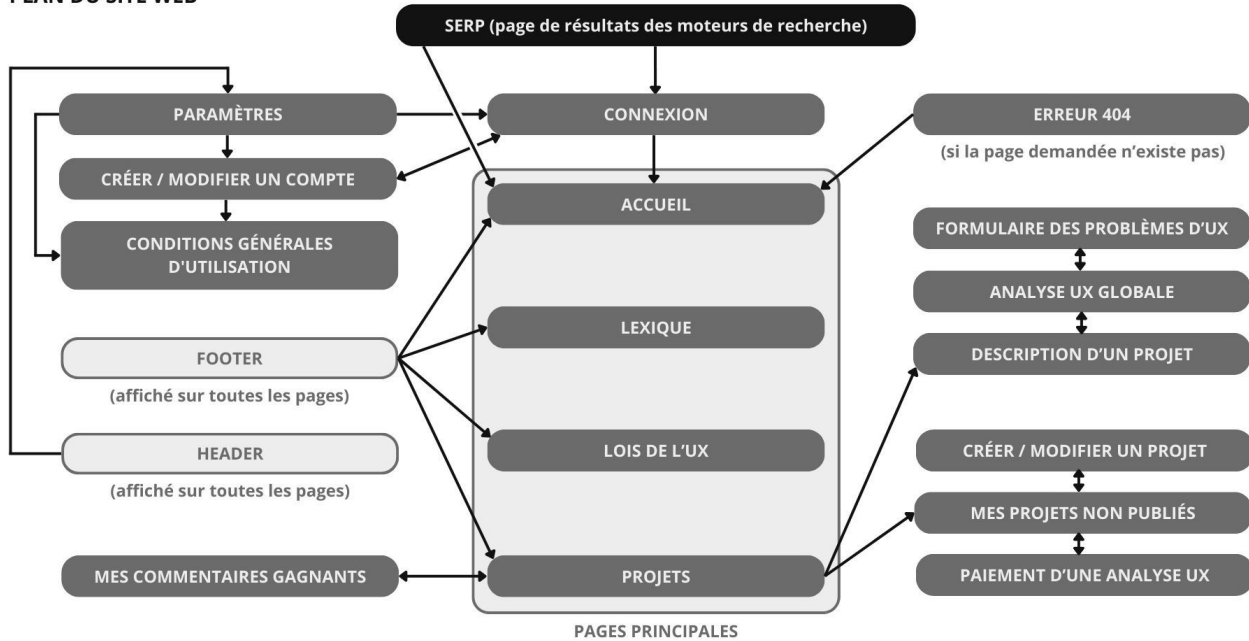
Voici l'intérêt des pages principales de mon application :

- La page d'accueil sert à présenter les avantages et les services rendus par l'application pour les chargés de refonte de site et les autres utilisateurs.
- Les pages "Lexique" et "Lois de l'UX" servent à guider la rédaction des commentaires en fournissant respectivement un glossaire de l'UX Design (pour parler avec un vocabulaire professionnel) et une liste de principes psychologiques sur l'UX Design (permettant d'améliorer l'expérience utilisateur s'ils sont appliqués).
- La page "projets" sert à lister les projets de création ou de refonte de sites publiés et donc analysables par tous les utilisateurs de l'application.

6.2 - Plan de site

Voici le plan de site qui permet de visualiser la navigation au sein des pages de mon application :

PLAN DU SITE WEB

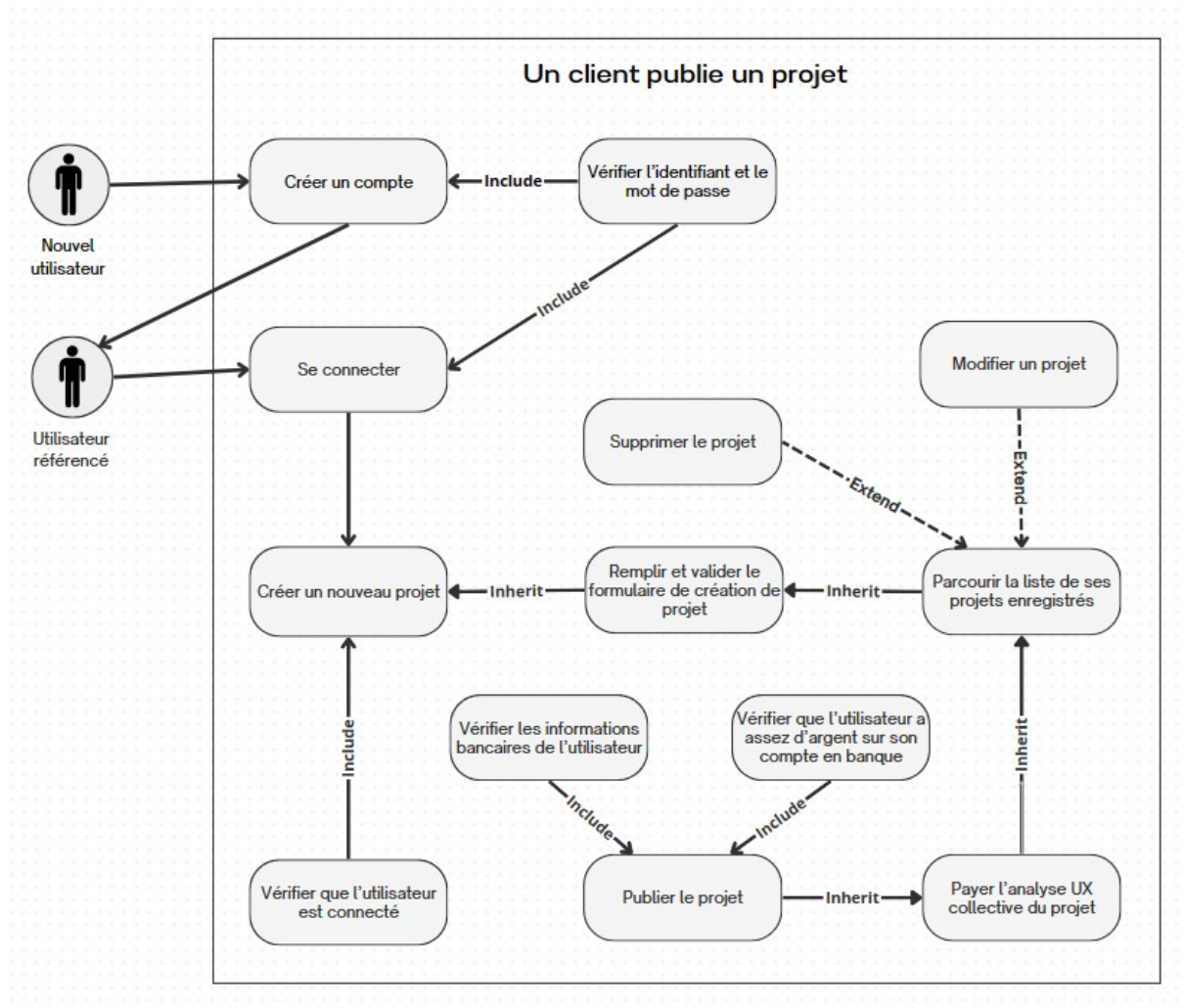


Plan de site de l'application « Les As de l'UX »

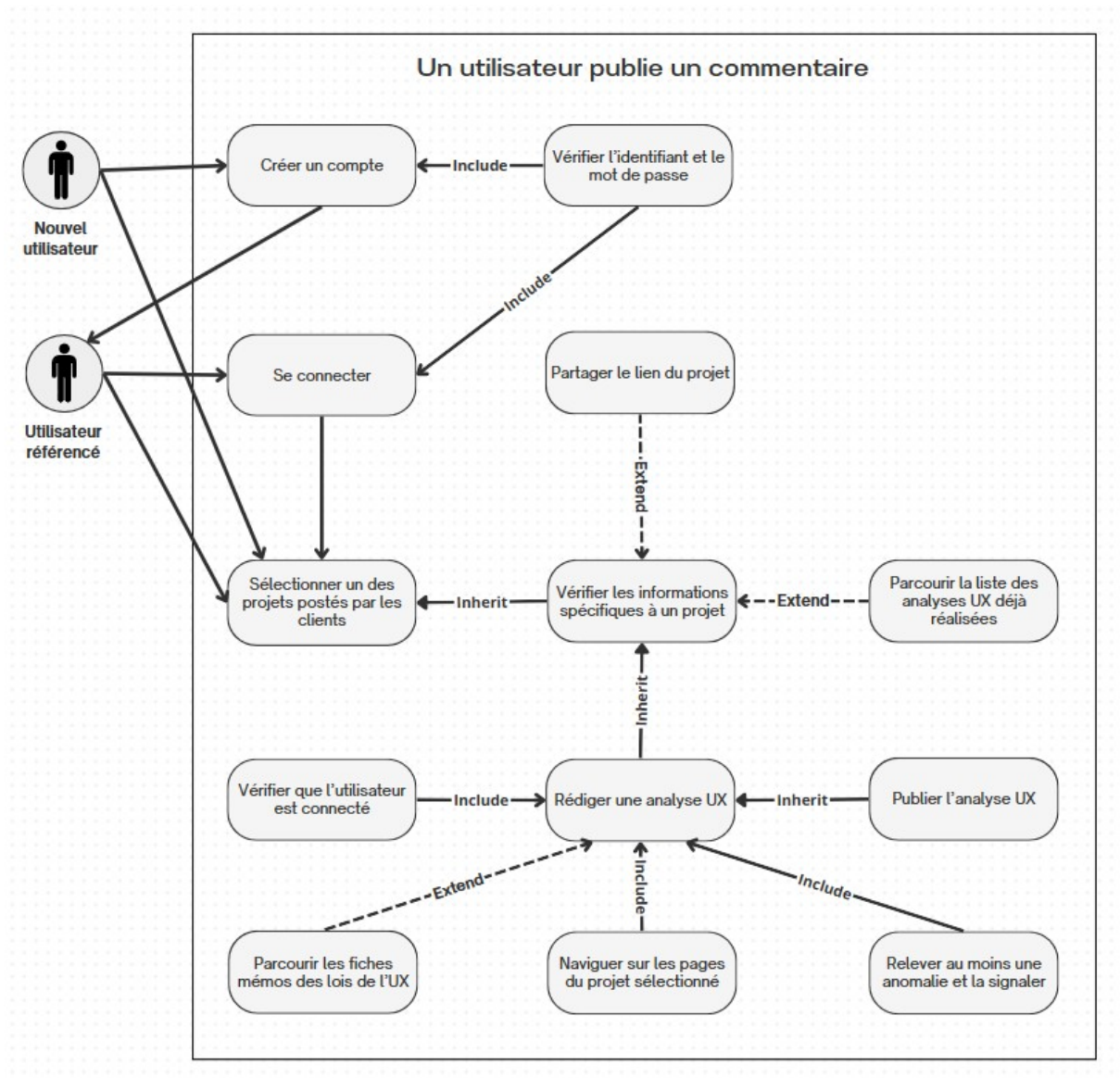
6.3 - Use cases

Un use case est un schéma qui représente un ensemble d'actions du produit apportant de la valeur par leurs interactions avec les utilisateurs.

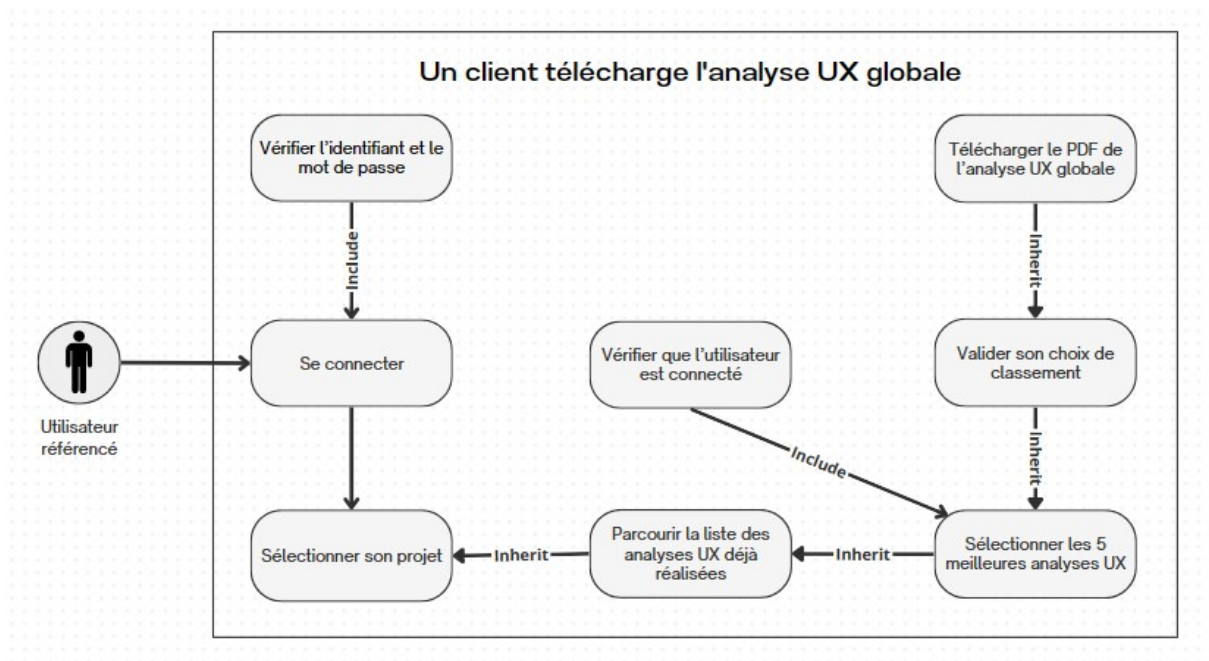
Voici les différents uses cases de mon projet qui correspondent aux scénarios principaux se produisant au sein de l'application :



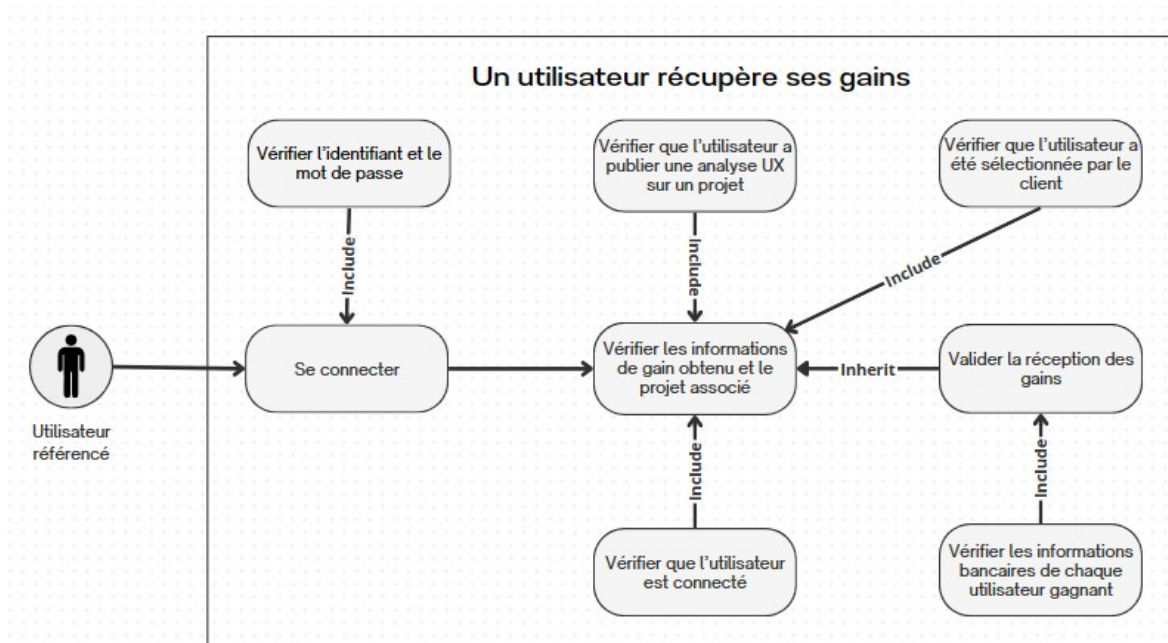
Use case « Un client publie un projet »



Use case « Un utilisateur publie un commentaire »



Use case « Un client télécharge l'analyse UX globale »



Use case « Un utilisateur récupère ses gains »

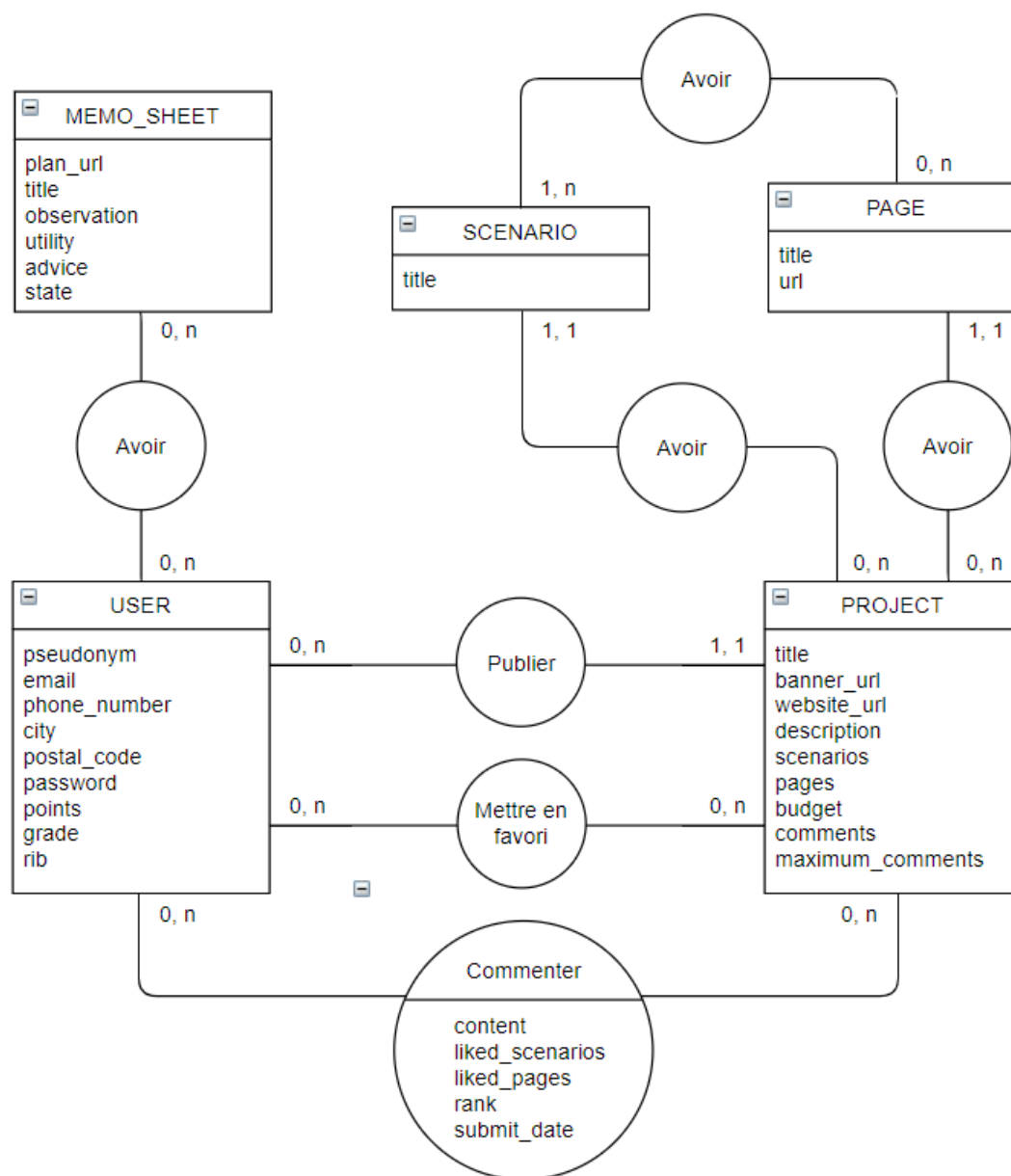
Le mot "Inh rit" signifie que l' tape est obligatoire ; cette  tape indique aussi le parcours suivi par l'utilisateur du d but   la fin du sc nario (cette  tape peut  tre compl t e par plusieurs autres  tapes obligatoires ou non).

Le mot "Include" signifie que l' tape est obligatoire pour assurer la validit  du sc nario tandis que le mot "Extend" signifie que c'est une  tape facultative (mais fortement recommand e).

6.4 - Modèle conceptuel de données

Un Modèle Conceptuel de Données (MCD) est un schéma qui permet de comprendre simplement comment les différentes tables sont liées entre elles.

Voici le MCD de mon projet (que j'ai réalisé sur le logiciel GitMind) :



Modèle Conceptuel des Données

J'ai intégré des verbes d'action entre les tables qui seront amenés à interagir ensemble dans la base de données de mon application (qui n'existait pas encore à cette étape de conception).

Voici les différentes liaisons fonctionnelles entre les tables de ma base de données qui m'ont permis de réaliser le Modèle Conceptuel de Données (avec le nombre d'occurrences possibles, c'est-à-dire les cardinalités, entre les 2 tables associées et dans les 2 sens) :

- Un utilisateur peut publier 0 ou plusieurs projet(s).
- Un projet peut être publié par 1 et 1 seul utilisateur.

- Un projet peut avoir 0 ou plusieurs scénario(s).
- Un scénario peut appartenir à 1 et 1 seul projet.

- Un projet peut avoir 0 ou plusieurs page(s).
- Une page peut appartenir à 1 et 1 seul projet.

- Un commentaire peut avoir 0 ou plusieurs scénario(s).
- Un scénario peut appartenir à 1 et 1 seul commentaire.

- Un commentaire peut avoir 0 ou plusieurs page(s).
- Une page peut appartenir à 1 et 1 seul commentaire.

- Un scénario peut avoir 1 ou plusieurs page(s).
- Une page peut appartenir à 0 ou à plusieurs scénario(s).

- Un utilisateur peut mettre en favori 0 ou plusieurs projet(s).
- Un projet peut être mis en favori par 0 ou plusieurs utilisateur(s).

- Un utilisateur peut commenter 0 ou plusieurs projet(s).
- Un projet peut être commenté par 0 ou plusieurs utilisateur(s).

- Un utilisateur peut avoir 0 ou plusieurs fiche(s) mémo.
- Une fiche mémo peut appartenir à 0 ou à plusieurs utilisateur(s).

6.5 - Modèle relationnel

Un modèle relationnel est plus détaillé que le MCD ; c'est un schéma qui permet de non seulement de voir les liaisons qui existent entre les tables mais aussi entre les champs des tables différentes entre elles (le type des champs est également visible) Son intérêt principal est de pouvoir s'en servir pour construire la base de donnée.

J'ai réalisé le modèle relationnel suivant sur la plateforme Dbdiagram.io (qui s'occupe de convertir automatiquement le code produit dessus au langage SQL afin d'être compréhensible par une base de données) :



Modèle relationnel des tables de la base de données

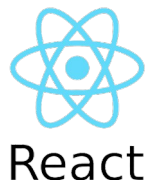
Lorsque il y a une relation "many to many" entre 2 tables (c'est-à-dire, quand un enregistrement d'une table peut être liées à plusieurs enregistrements de l'autre table et réciproquement), il faut rajouter une table de liaison dont les champs sont les clés étrangères pour les 2 tables (c'est-à-dire leur clé primaire respective permettant d'identifier un seul enregistrement spécifique).

7 - SPÉCIFICATIONS TECHNIQUES

7.1 - Technologies choisies

Je vais à présent vous présenter la justification de mes choix de technologies appliquées dans le cadre de ce projet (selon des critères les plus rationnels possibles).

Technologies du Frontend



J'ai choisi **Vite** pour construire rapidement mon application côté frontend car :

- Cela offre un environnement de développement prêt à l'emploi et un temps de démarrage ultra-rapide.
- Les modifications apportées au code sont reflétées instantanément dans le navigateur, sans avoir besoin de recharger la page.

J'ai choisi framework JavaScript **React** pour construire des interfaces utilisateur dynamiques et interactives car :

- Il est relativement simple et à prendre en main grâce à son architecture à base de composants qui fonctionnent tous de la même façon (avec des états, des comportements et un affichage).
- Il utilise l'extension de syntaxe JSX qui permet d'écrire du code HTML directement dans du code JavaScript ; cela rend la création d'interfaces utilisateur plus intuitive et facilite le travail avec les composants.
- Grâce à son Virtual DOM, il offre des performances élevées, même pour les applications complexes.
- De plus, il est soutenu par une vaste communauté de développeurs et dispose d'un écosystème riche en outils, bibliothèques et extensions.

J'ai choisi la plateforme de cloud computing **Vercel** pour mettre en ligne la partie frontend de mon application car :

- Elle offre un déploiement rapide et sans effort pour les applications web.
- Elle prend en charge une large gamme de technologies web (par ex. : React, Vue.js, Angular, etc.).
- Elle permet de déployer automatiquement ses projets à partir du référentiel Git dès que des changements sont effectués.
- Elle prend en charge le déploiement basé sur les branches.

Technologies du Backend

Express JS

Sequelize

SQLite

Docker

node JS

Render

J'ai choisi le framework **Express** pour créer rapidement une API robuste et évolutive car :

- C'est un framework minimaliste qui offre une grande flexibilité (il permet aux développeurs de choisir les bibliothèques et les outils qui correspondent le mieux à leurs besoins spécifiques)
- Il peut gérer efficacement de nombreuses connexions simultanées avec une latence minimale.

J'ai choisi l'ORM **Sequelize** pour servir d'intermédiaire entre l'API et la base de données car :

- Il offre une abstraction de la base de données en permettant d'utiliser des objets JavaScript et des requêtes plutôt que des requêtes SQL brutes.
- Il facilite la définition et la gestion des relations entre les tables de la base de données.

J'ai choisi la base de donnée **SQLite** car :

- Il est simple à utiliser et ne nécessite pas de configuration de serveur complexe (il est directement intégré dans la partie backend sous la forme d'un fichier).
- Il permet notamment la validation des contraintes liées aux clés étrangères.
- SQLite offre des fonctionnalités de sécurité intégrées telles que le cryptage des bases de données, qui peuvent être activées pour protéger les données stockées (même si, sur le long terme et lorsqu'il y aura de vraies données des utilisateurs, il vaudrait mieux migrer sur une base de données de type "client-serveur" pour renforcer la sécurité).

J'ai choisi **Docker** pour déployer et gérer l'application car :

- Il utilise des conteneurs à la fois légers (mais qui comporte toutes les dépendances logicielles nécessaires pour exécuter l'application) et portables (c'est-à-dire qui peuvent être exécutés sur n'importe quel environnement prenant en charge Docker).
- Il utilise une technologie de virtualisation légère qui nécessite moins de ressources système par rapport aux machines virtuelles traditionnelles.
- Il facilite l'évolutivité des applications en permettant le déploiement et la gestion de multiples conteneurs de manière automatisée.

J'ai choisi **NodeJS** pour exécuter du code JavaScript côté serveur car :

- Il peut facilement s'adapter à la croissance de l'application.
- Il peut gérer efficacement de nombreuses connexions simultanées sans consommer beaucoup de ressources système.
- Son modèle asynchrone et son écosystème de modules en font une plateforme polyvalente et puissante pour le développement d'applications côté serveur.

J'ai choisi la plateforme de cloud computing **Render** pour mettre en ligne la partie backend de mon application car :

- Elle permet de déployer rapidement une application sans avoir à se soucier de la gestion complexe de l'infrastructure et des serveurs.
- Elle offre des performances optimales même en cas de pics de trafic.
- Elle propose des fonctionnalités de sécurité avancées, telles que la gestion des certificats SSL, les pare-feu et les contrôles d'accès, pour protéger les applications et les données sensibles.

Technologies communes au frontend et au backend



J'ai choisi l'IDE **Visual Studio Code** car :

- Il prend en charge une grande variété de langages de programmation, de frameworks et d'outils.
- Il est léger et démarre rapidement (contrairement à certains autres IDE plus lourds).
- Il dispose d'un large choix d'extensions très riche qui permet d'étendre les fonctionnalités de l'IDE en fonction de ses besoins spécifiques.
- Il propose des fonctionnalités de débogage intégrées pour de nombreux langages de programmation.
- Il est intégré nativement avec l'outil de versionning Git (ce qui facilite le suivi des modifications, la gestion des branches et des conflits, et la collaboration sur des projets basés sur Git directement depuis l'IDE).

7.2 - Architecture de l'application

L'architecture d'une application web permet de définir la manière dont les différents composants sont structurés entre eux et aussi la manière dont ils interagissent les uns avec les autres.

Voici le diagramme de l'architecture de mon application :

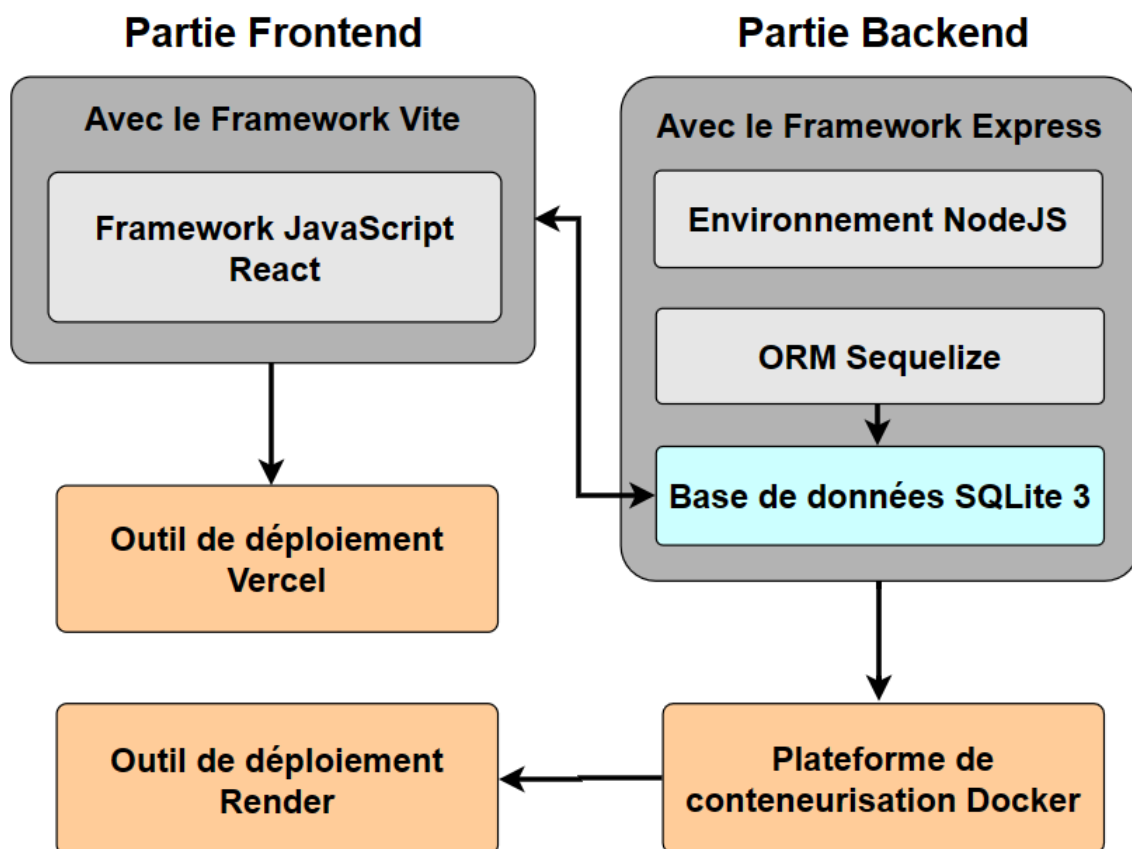


Schéma de l'architecture de l'application "Les As de l'UX"

8 - RÉALISATION DU PROJET

8.1 - Développement

Voici plusieurs captures d'écran significatives de mon code ainsi que l'explication pour décrire le fonctionnement du programme :

Extraits de code de la partie backend

Exemple n°1

Ces lignes de code montrent la configuration de la route gérant l'authentification des utilisateurs :

```
src > TS index.ts > ...
```

```
162 // Import des routes
163 import { authRouter } from "../router/authRouter";
164 import { userRouter } from "../router/userRouter";
165
166
167 const apiRouter = express.Router();
168
169 // Route pour l'authentification des utilisateurs
170 apiRouter.use('/auth', authRouter);
```

Configuration de la route d'authentification des utilisateurs (partie backend)

Le programme crée une nouvelle instance du routeur Express appelée « apiRouter » puis utilise sa méthode « use » pour reconstituer la route complète de la requête des utilisateurs vers le router « authRouter » (où la connexion et la déconnexion d'un utilisateur sont gérées).

Exemple n°2

Ces lignes de code montrent la gestion de la connexion d'un utilisateur via une requête POST sur le chemin '/local' du routeur d'authentification « authRouter » :

```
src > router > TS authRouter.ts > authRouter.post("/logout") callback
11 // Connexion d'un utilisateur
12 authRouter.post("/local", async (req, res) => {
13   const userEmail = req.body.identifiant;
14   const userPassword = req.body.password;
15
16   const sameUserEmail = await User.findOne( {where: {email: userEmail} })
17
18   if(sameUserEmail) {
19     const isPasswordCorrect = await bcrypt.compare(userPassword, sameUserEmail.dataValues.password);
20
21     if (isPasswordCorrect) {
22       delete sameUserEmail.dataValues.password;
23
24       const token = jwt.sign(sameUserEmail.dataValues, process.env.JWT_SECRET!); // JWT SECRET est la clé
25       // pour coder et encoder les informations communiquées entre le temps entre le client et le serveur.
26
27       res.status(200).json({
28         message: "L'utilisateur a bien été connecté.",
29         jwt: token,
30       });
31     }
32     else {
33       res.status(400).send("Email or Password is incorrect");
34     }
35   } else {
36     res.status(400).json(
37       {
38         message: "Il n'existe aucun compte ayant ce couple email / mot de passe.",
39       }
40     )
41   }
42 })
```

Le programme effectue les actions suivantes :

1. Il récupère l'email et le mot de passe fournis dans le corps de la requête « req.body.identifiant » et « req.body.password ».
2. Il recherche dans la base de données un utilisateur ayant l'email fourni.
3. S'il trouve un utilisateur avec cet email, il vérifie si le mot de passe fourni correspond au mot de passe stocké dans la base de données en utilisant bcrypt pour comparer les hashes des mots de passe.
4. S'il y a correspondance, il génère un token JWT signé avec les informations de l'utilisateur (à l'exception du mot de passe) et renvoie ce token dans la réponse.
5. Si l'email ou le mot de passe est incorrect, il renvoie une réponse d'erreur avec le statut 400.
6. Si aucun utilisateur n'est trouvé avec l'email fourni, il renvoie également une réponse d'erreur avec le statut 400.

Exemple n°3

J'ai utilisé des modèles de table pour simplifier le développement de mon application en offrant une abstraction cohérente pour interagir avec la base de données, tout en renforçant la sécurité et en améliorant la maintenabilité du code.

Ce code définit un modèle de données pour un projet en utilisant l'ORM Sequelize :

```
src > models > TS ProjectModel.ts > ...
1  import { DataTypes, Sequelize } from "sequelize";
2
3  export const ProjectModel = (sequelize: Sequelize) => {
4    return sequelize.define('project', {
5      title: {
6        type: DataTypes.STRING,
7        allowNull: false, // champs obligatoire
8        unique: true // enregistrement unique
9      },
10     bannerUrl: {
11       type: DataTypes.STRING,
12       allowNull: true, // champs non obligatoire
13       unique: false // enregistrement non unique
14     },
15     websiteUrl: {
16       type: DataTypes.STRING,
17       allowNull: false, // champs obligatoire
18       unique: true // enregistrement unique
19     },
20     description: {
21       type: DataTypes.TEXT,
22       allowNull: false, // champs obligatoire
23       unique: true // enregistrement unique
24     },
25     budget: {
26       type: DataTypes.NUMBER,
27       allowNull: false, // champs obligatoire
28       unique: false // enregistrement non unique
29     },
30     comment: DataTypes.NUMBER,
31     maximumComments: DataTypes.NUMBER
32   });
33 }
```

Modèle de la table "Project" (partie backend)

Pour chaque champ de la table « Project » (sauf les 2 derniers), le programme spécifie le type de donnée (texte, nombre, etc.), s'il peut être vide ou pas (donc si l'utilisateur devra obligatoirement le remplir ou pas) ainsi que le caractère unique ou pas de la donnée saisie (l'autorisation ou le refus des doublons).

Exemple n°4

Ces lignes de code montrent la configuration initiale d'une base de données SQLite avec Sequelize, ainsi que la définition des modèles de données correspondants :

```
src > TS index.ts > ...
44 // Création de la BDD
45 export const sequelize = new Sequelize({
46   dialect: 'sqlite',
47   storage: 'db/database.sqlite'
48 });
49
50 // Création des tables principales par Sequelize
51 export const User = UserModel(sequelize);
52 export const Project = ProjectModel(sequelize);
53 export const MemoSheet = MemoSheetModel(sequelize);
54
55 // Création des tables de jonction par Sequelize (qui comportent des champs propres à elles-mêmes)
56 export const Comment = CommentModel(sequelize);
57 export const Scenario = ScenarioModel(sequelize);
58 export const Page = PageModel(sequelize);
59 export const TokenBlackList = TokenBlackListModel(sequelize);
```

Création des tables (partie backend)

De la ligne 45 à 48, le programme crée une instance de Sequelize avec le dialecte SQLite et spécifie le chemin de stockage de la base de données.

De la 51 à 53, le programme crée et exporte des modèles de données pour les entités User, Project et MemoSheet en utilisant les modèles définis par Sequelize et la connexion à la base de données.

De la ligne 56 à 59, le programme crée et exporte des modèles de données pour les tables de jonction « Comment », « Scenario », « Page » et « TokenBlackList » en utilisant les modèles définis par Sequelize et la connexion à la base de données.

Une table de jonction (ou table de liaison) est une table utilisée dans les bases de données relationnelles pour représenter une relation "many-to-many" (plusieurs-à-plusieurs) entre deux autres tables, c'est-à-dire quand un enregistrement d'une table peut être relié à plusieurs enregistrements d'une autre table et réciproquement aussi.

Extraits de code de la partie frontend

Exemple n°1

Voici le contenu du composant « FormField » qui me permet de gérer l’affichage des champs de formateurs de façon centralisée dans la partie frontend :

```
src > components > elements > FormField.tsx > ...
4  export default function FormField(props: { label: string, type: string, placeholder: any, value: any,
    requiredField: boolean, todo: boolean, onTransmitValue: (newValue: any) => void }) {
5      // ETATS
6      const [inputValue, setInputValue] = useState(props.value);
7      // const [allToDo, setAllToDo] = useState([]);
8
9      // COMPORTEMENTS
10     const handleChange = useCallback(async (e: React.ChangeEvent<HTMLInputElement>) => {
11         props.onTransmitValue(e.target.value); // Envoi de la nouvelle valeur saisie par l'utilisateur du composant
            enfant ("FormField") au parent (par exemple : une page de formulaire).
12         setInputValue(e.target.value) // Mise à jour de l'état "inputValue" "e.target.value" avec la nouvelle
            valeur saisie par l'utilisateur grâce au setter "setInputValue".
13     }, [inputValue, props.onTransmitValue])
14
15     // AFFICHAGE
16     return (
17         <>
18         <div className="label-input">
19             <label>{props.label}{props.requiredField === true ? <span className="required-asterix">&nbsp;*</span> :
                ""}</label>
20             <input onChange={handleChange} type={props.type} placeholder={props.placeholder} value={inputValue}
                required={props.requiredField === true ? true : false} ></input>
21         </div>
```

Contenu du composant "FormField" (partie frontend)

Le programme initialise l'état « inputValue » avec la propriété « value » afin qu'il reflète la valeur initiale transmise par le parent.

La fonction « handleChange » permet de récupérer la nouvelle valeur (saisie dans le champ par l'utilisateur) à chaque fois qu'elle change (c'est-à-dire, à chaque fois que la valeur de « e.target.value » change). Puis elle l'envoie cette nouvelle valeur au composant parent via l'événement « onTransmitValue » défini dans les propriétés du composant.

Exemple n°2

Dans le composant « AccountForm », voici l'utilisation du composant « FormField » (stocké dans une constante afin de pouvoir être utilisé plusieurs fois dans le code) :

```
src > components > pages > AccountForm.tsx > ...
25  export default function AccountForm(){
60
61      const pseudonymFormField =
62      <FormField
63          label={"Pseudonyme"}
64          type={InputTypes.Text}
65          placeholder={"Exemple : Toto"}
66          value={ pseudonymValue !== "" ?
67              pseudonymValue
68              :
69              user["pseudonym"]
70          }
71          requiredField={true}
72          todo={false}
73          onTransmitValue={(newValue: string) => setPseudonymValue(newValue)}
74      />
```

Utilisation du composant "FormField" (partie frontend)

Le programme transmet au composant enfant « FormField » les valeurs des différentes propriétés indiquées et reçoit la nouvelle valeur saisie par l'utilisateur via l'événement « onTransmitValue ».

Exemple n°3

Voici l’affichage conditionnel du composant « FormField » ou, si l’utilisateur a déjà un compte, de l’affichage des données récupérées dans la base de données (avec la possibilité d’éditer chaque champ) :

src > components > pages > AccountForm.tsx > AccountForm

```

25  export default function AccountForm(){
464      <form>
465
466      { userIsConnected === false ?
467          pseudonymFormField
468      : <div></div>
469      }
470      { userIsConnected === true && updatingPseudonymField === true ?
471          <div className='completed-field'>
472              {pseudonymFormField}
473              <div onClick={() => handleEditField("pseudonym")} className="icon validate
474                  small-icon carbon--checkmark-outline"></div>
475          </div>
476      : <div></div>
477      }
478      { userIsConnected === true && updatingPseudonymField === false ?
479      <div className='completed-field'>
480          <p><span className='bold'>Pseudonyme : </span>
481          { pseudonymValue !== "" ?
482              pseudonymValue
483          :
484              user["pseudonym"] }
485          </p>
486          <div onClick={() => handleEditField("pseudonym")} className="icon edit
487              small-icon carbon--edit"></div>
488      </div>
489      : <div></div>
490      }

```

Affichage conditionnel du composant "FormField" (partie frontend)

Le programme utilise 3 ternaires pour afficher des éléments en fonction de la circonstance actuelle :

- Soit, l’utilisateur n’est pas connecté ; c’est donc le champ « FormField » qui s’affiche (sans valeur pré-remplie car l’application n’a pas accès à la base de données liée à un compte spécifique)
- Soit l’utilisateur est connecté et il est en mode « édition » de ce champ du formulaire ; c’est donc le champ « FormField » qui s’affiche (avec la valeur pré-remplie récupérée depuis la base de données liée à ce compte)
- Soit l’utilisateur est connecté et il n’est pas en mode « édition » de ce champ du formulaire ; l’affichage comprend donc le nom du champ (label), la valeur récupérée depuis la base de données et l’icône « éditer » (pour modifier la valeur du champs)

8.2 - Difficultés rencontrées et axes d'améliorations dans le travail en équipe

Difficultés rencontrées

- Le fait que j'ai dû travailler seul ne m'a pas permis d'avoir beaucoup d'avis extérieurs et de prendre le recul nécessaire pour résoudre les différents problèmes techniques.
- Au début de la phase de développement du projet, j'ai pris la mauvaise habitude de pousser mes branches de fonctionnalité sur GitHub sans les fusionner avec la branche « develop », ce qui a occasionné un manque de synchronisation entre la branche develop locale (à jour) et celle sur le cloud.

Axes d'améliorations

- Lorsque je bloque sur un problème, je devrais mettre par écrit les difficultés rencontrées puis, après avoir cherché la solution pendant un temps raisonnable, d'aller demander de l'aide à une personne référente (formateurs, collègue, etc.).
- Même dans le cas de figure où je suis amené à développer seul, il faut toujours mettre en pratique les méthodologies de travail en équipe (tableau KANBAN, diagramme de GANTT, utilisation des branches, pull requests, code review, etc.).

CONCLUSION

La partie du projet que j'ai préféré faire, et aussi celle dont je suis le plus fier, est sans doute le frontend de l'application que j'ai codé en reprenant le même contenu et la même présentation des pages du prototype.

Dans le cadre de ce projet, j'aurais voulu améliorer davantage :

- L'espace administrateur avec les fonctionnalités qui lui sont spécifiques
- La présence du concept de gamification lors des différentes étapes du parcours utilisateur
- Automatiser le paiement de la publication d'un projet par un client et le virement bancaire des utilisateurs qui ont publié des commentaires sélectionnés par le client
- Le système de filtrage des projets de refonte publiés pour trouver plus facilement des projets qui correspondent aux affinités des utilisateurs
- La refactorisation de certaines parties redondantes de mon code (pour en améliorer la maintenabilité)
- Etc.

A l'heure actuelle, j'estime ne pas avoir terminé les ambitions que je m'étais fixé pour "Les As de l'UX"... Or, à terme, j'aspire à ce que l'application "Les As de l'UX" devienne une application iconique qui puisse automatiser des tests utilisateurs de qualité pour améliorer l'expérience des utilisateurs sur de nombreux projets web (dont les exigences à leur égard ne cessent de croître à l'époque du "tout numérique").

Idéalement, je prévois également que l'application puisse être utilisée dans un environnement professionnel (pour y publier des projets web) mais également dans un environnement extra-professionnel (pour y publier des commentaires sur des projets web).

Pour faire le bilan de mon année de formation à Simplon, au début de l'année, j'avais déjà des bases en développement informatique mais la découverte du framework JavaScript React a été pour moi très bénéfique car cela m'a permis d'agencer mon code de façon plus optimisée, sous la forme de composants imbriqués et aux fonctionnalités limitées.

En guise de conclusion, je dirais que l'élaboration de ce projet chef-d'œuvre m'a encore plus conforté à exercer le métier de développeur frontend pour la suite de ma carrière professionnelle.