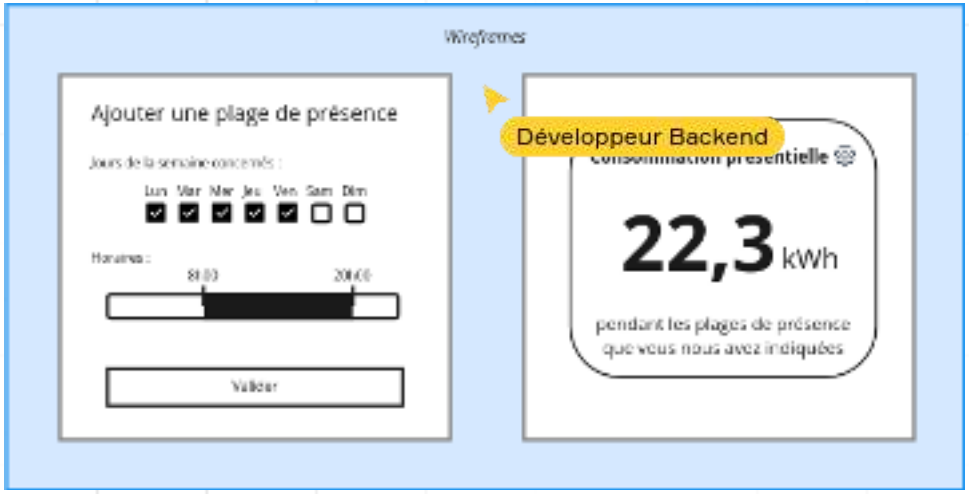
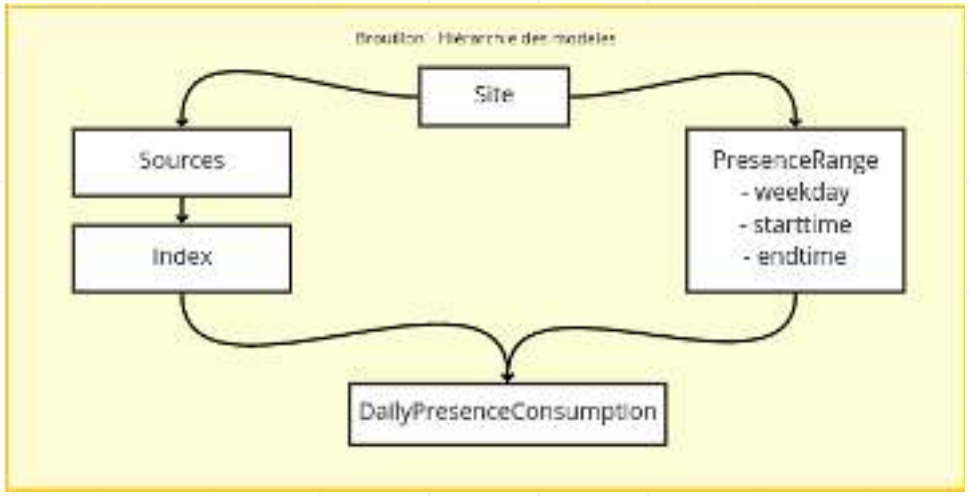
Rapprocher les cultures UX et backend

(pour travailler moins)





Laurent Lévêque
Certification UX Design 2020/21

Introduction

> “Ça, on ne peut pas le faire.”
> - un développeur anonyme, 2021

Ce coup de massue semble s’abattre inévitablement sur le travail des UX designers, avec plus ou moins de gravité. Dans certains cas, la proposition peut encore être corrigée, dans d’autres elle doit être abandonnée, parfois en plein développement. Dans le scénario catastrophe, l’équipe doit défaire une partie du travail réalisé, au prix d’un effort imprévu (et frustrant).

Dans mon expérience récente, j’ai vécu avec les développeurs “backend” de mon équipe plusieurs situations de ce type. Le travail suivant explore la relation particulière qui existe entre nous, les UX designers et les développeurs backend, et ce qui peut induire ces situations de blocage.

L’objectif final : **Rapprocher les cultures UX et backend, pour mieux travailler ensemble** (et éviter du travail inutile)

Une démarche d’UX design a été appliquée pour cadrer le problème et générer des solutions centrées utilisateurs. Ainsi :

- Après un premier cadrage du sujet, la **recherche exploratoire** livrera les définitions et des clés sur le processus de production logicielle
- Une **recherche utilisateurs** nous aidera ensuite à identifier les circonstances dans lesquelles les blocages peuvent survenir
- Puis, de nouveaux outils seront introduits pour **répondre à notre problématique**, avec les résultats de premiers tests d’utilisation
- Enfin, nous présenterons une **roadmap** pour diffuser largement ces outils auprès d’autres équipes et donnerons quelques pistes de réflexions supplémentaires sur le sujet.

Résumé

La nature de notre travail d'UX designers nous amène souvent à collaborer étroitement avec les développeurs informatiques.

Côté frontend, les bonnes pratiques pour travailler ensemble sont connues et des outils adaptés ont émergé pour faciliter la coopération.

Ce mémoire se propose d'explorer l'articulation du travail avec les développeurs backend, et en particulier les conditions qui peuvent mener à des blocages de l'équipe et de la chaîne de production.

En se limitant aux cas où les équipes travaillent de manière itérative et en mode agile, nous déterminerons le parcours des UX designers et des développeurs backend pour livrer une amélioration du produit.

Après avoir identifié des points de frustration, nous utiliserons certaines opportunités pour construire une boîte à outils de quatre activités qui pourront rapprocher nos disciplines.

Nous présenterons enfin les résultats encourageants obtenus à l'évaluation de cette proposition.

Abstract

Being a UX designer often means working hand-in-hand with software developers.

When it comes to working with frontend developers, we can usually rely on well-known best practices and specific tools have emerged to help us both in that way.

This work is about exploring how we play around with backend developers, especially when things go wrong and the team and production line are impacted.

We will focus on agile teams working iteratively and will define the user journeys for both UX designers and backend developers when they improve the product they are working on.

After studying their pain points, we will build upon identified opportunities to put together a toolbox of four engaging activities that could bring UX designers and backend developers culturally closer to each other.

We will ultimately present a usability study of this toolbox and its encouraging results.

Table des matières

Cadrage du sujet

Point de départ

Le point de départ de ce travail est une situation de frustration chez les développeurs backend d'Eco CO_2, l'entreprise de conseil en économies d'énergie qui m'emploie en tant qu'UX designer.

Ayant été développeur par le passé, et notamment dans cette équipe, une certaine sensibilité technique infuse mon travail d'UX design. Pourtant, plusieurs fois au cours de l'année passée, les développeurs backend ont exprimé des frustrations concernant des décisions de conception. Notamment, certains choix techniques sous-optimaux leur semblaient imposés par les spécifications UX dont ils disposaient au moment de développer. D'autres fois, des malentendus autour de ces spécifications sont passés inaperçus et les fonctionnalités concernées ont dû être ré-implémentées plusieurs fois avant que l'implémentation ne traduise l'intention UX.

Si ces deux frustrations sont courantes et régulièrement discutées, c'est le fait qu'elles ne viennent, dans mon équipe, que de développeurs *backend* qui m'a interpellé.

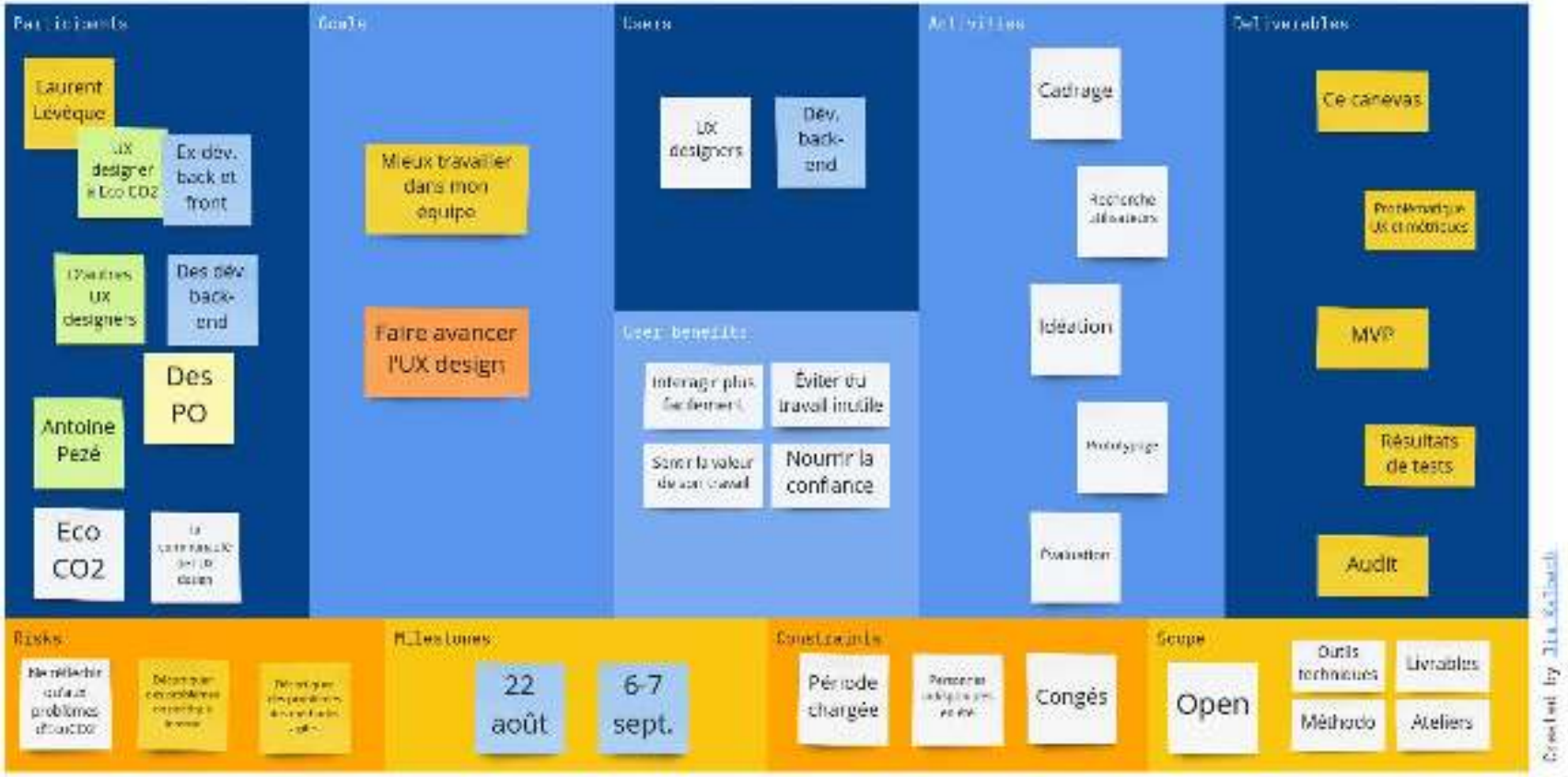
La cible

Si le problème initial se manifeste à Eco CO_2, on peut imaginer qu'il apparaisse également dans d'autres équipes qui adoptent aussi une démarche **d'UX design**, où le travail est organisé de manière **agile** et où le **développement backend est séparé** de celui du frontend.

Dans ces équipes, les premiers utilisateurs concernés seront les développeurs backend et les UX designers.

L'objectif

L'objectif principal est de **résoudre pour mon équipe** les frustrations entre UX designers et développeurs backend - mais si possible, j'aimerais que la solution envisagée puisse **profiter à d'autres UX designers** ou plus généralement à d'autres équipes en difficulté sur les relations UX/backend.



Atelier de cadrage réalisé avec un stakeholder

Recherches

Recherche exploratoire

Les frustrations étant relevées en particulier par les développeurs backend, il a semblé naturel d'orienter une partie de la recherche exploratoire sur les particularités de la coopération UX/backend, notamment par rapport à la coopération UX/frontend.

Par ailleurs, comme cette collaboration s'inscrit dans le cadre d'une équipe qui travaille de manière agile, l'impact de ces méthodes sur la coopération entre les UX designers et les développeurs backends a également été exploré.

En plus de recherches documentaires, des entretiens libres ont été réalisés auprès de six personnes de l'équipe Eco CO_2 (cinq développeurs backend et un product owner) pour consolider la compréhension générale du sujet.

Le développement backend et sa relation avec l'UX

Le backend : une définition

Un des premiers résultats notables de cette recherche est que le périmètre du poste de développeur backend semble avoir évolué avec la modernisation du Web, et par ailleurs paraît très différent d'une entreprise à l'autre. Par conséquent, les caractérisations sont nombreuses et on ne trouve pas de définition qui fasse référence pour le métier de développeur backend.

Toutefois, la "séparation des préoccupations", une idée émise par Dijkstra en 1974[1] et largement adoptée comme bonne pratique de développement a induit une différenciation entre le code backend et le code frontend à partir de laquelle on peut cerner les contours des deux métiers. Ainsi, si le frontend est la partie du produit qui présente les données et avec laquelle l'utilisateur est en interaction directe, le backend **donne vie aux concepts fondamentaux** derrière le service et gérant l'accès à ces mêmes données.

Développer le backend d'un produit, c'est implémenter l'état du système, c'est-à-dire la structure de données qui peut décrire exactement dans quel état se trouve notre produit à un instant donné (ex.: les informations de toutes les commandes, de tous les comptes clients, et la hiérarchie de ces différents concepts) : c'est la modélisation dans la base de données.

[1] Contributeurs multiples, "Separation of concerns", 2021, https://en.wikipedia.org/wiki/Separation_of_concerns

Côté backend, on développe également des moyens d'examiner l'état du système (ex.: API de lecture, et éventuellement représentations des données) et des moyens de modifier cet état (ex.: API d'écriture, et éventuellement interfaces d'administration).

Autour de ce cœur de métier, le poste de développeur backend implique des **tâches connexes** comme la documentation, l'instrumentation et la maintenance du système, la réponse à des tickets de support spécifiques,...

L'importance du backend

Les techniques de développement backend ont considérablement évolué ces dernières années. Par ailleurs, le backend revêt aussi une importance stratégique car il implémente la logique métier et manipule les données utilisateurs. Il est également soumis à des contraintes et risques assez importants, concernant l'infrastructure (puissance, capacité, trafic, pannes matérielles…), la réglementation des données (RGPD,...), la sécurité (vol de données, appropriation de ressources,...).
Certaines entreprises ont accumulé une énorme dette technique côté backend, ce qui les empêche de réagir rapidement aux nouveaux marchés du numérique. Les organisations dont le backend peut pivoter rapidement acquièrent quant à elles un avantage compétitif sur leur marché.

D'après Jakob Nielsen, quand un utilisateur commet une erreur dans l'utilisation d'un site web, c'est souvent parce qu'il a formé un modèle mental erroné[2]. En d'autres termes, la représentation mentale que se fait l'utilisateur du système n'est pas alignée avec la modélisation qui a été implémentée dans le code. Si certaines de ces confusions naissent dans les perceptions visuelles de l'interface et donc dans le frontend, d'autres trouvent leurs origines dans la modélisation backend. L'interface, par nature, cache une bonne partie de la complexité du système. Par exemple, dans le backend de Twitter, un simple tweet est modélisé par au moins 90 attributs (contenu, auteur, géolocalisation,...), la plupart étant invisible dans l'interface utilisateur[3] (ces attributs servent aux analyses marketing, comme en témoigne un article de The Economist en 2010[4]). Ainsi, il est facile d'imaginer que des implémentations complexes comme celle-là ne soient pas parfaitement alignées avec les représentations mentales que les utilisateurs se construisent face à une interface épurée.

De l'expérience utilisateur à la stratégie de l'entreprise, le backend a donc une **importance capitale**.

---

[2] Jakob Nielsen, "Mental Models", 2010, https://www.nngroup.com/articles/mental-models/
[3] https://developer.twitter.com/en/docs/twitter-api/v1/data-dictionary/object-model/example-payloads
[4] "What's in a tweet ?", The Economist, 2011, https://www.economist.com/graphic-detail/2011/09/29/whats-in-a-tweet

La visibilité du travail et ses impacts

Le caractère visuel du travail de frontend le rapproche naturellement de l'UX. En effet, le code produit est directement perçu par les utilisateurs, ce qui induit que les développeurs frontend se soucient assez naturellement des utilisateurs et de l'expérience ressentie.
De plus, la plupart des UX designers ayant souvent la double casquette UX/UI designer, le contact entre les deux spécialités est très rapproché.
Une manifestation directe de ce phénomène est que beaucoup d'outils font le pont entre l'UX design, l'UI design et le développement frontend en notamment en automatisant les spécifications CSS (Sketch, XD, Figma, proto.io,...). Cette proximité nourrit également la sensibilité UX des développeurs frontend.

À l'inverse, le travail des développeurs backend n'est perçu qu'indirectement par les utilisateurs : il n'est pas visible. Par extension, les problèmes d'utilisabilité issus du backend sont aussi perçus indirectement - et cette particularité semble les **éloigner du souci de l'UX**.
Ce caractère non-graphique complique également les échanges avec les designers, habitués aux outils visuels : manquer d'outils agréables aux deux disciplines, c'est **manquer d'un lieu pour dialoguer**.
Enfin, de manière plus générale, parce que le travail des backends est dur à démontrer, il est souvent perçu par les UX designers comme étant très technique et particulièrement inaccessible (au grand dam des développeurs frontend, dont le travail n'est pas moins technique).

Tout ceci semble **freiner l'émergence d'une sensibilité UX** chez les développeurs backends.[5]

L'impact de l'agilité

Les relations entre les UX designers et les développeurs backend s'inscrivent dans le cadre de la production logicielle, où l'on développe des produits interactifs à partir de code-source. Cette industrie bénéficie à la fois d'une large formalisation des pratiques et d'une hétérogénéité notable d'organisation des équipes, des outils et des moyens de production - par définition, ces choix d'organisations impactent les relations de travail, notamment celles entre UX designers et développeurs backend.

Les méthodes agiles

En 2001, plusieurs experts du développement logiciel ont écrit le Manifeste Agile, qui forme la base des méthodes agiles pour le développement. Ces méthodes s'appuient sur un développement cyclique, incrémental et qui s'adapte aux changements des besoins. Parmi les

[5] Viktoria Menlychuk, "How Backend Developers Impact UX: Actionable Insights", 2020, https://dzone.com/articles/how-backend-developers-impact-ux-actionable-insigh

méthodes agiles, la méthode Scrum est la plus répandue et pousse au découpage du travail en tâches au scope très limité, dans l'idée d'en maîtriser le risque associé.
D'après une étude de l'organisation Designers interactifs menée en France en 2018, **91% des équipes disent travailler en mode agile ou hybride** (agile/cascade)[6].

Les concepts de l'agilité

Les équipes agiles travaillent généralement par périodes de deux semaines, les sprints, pendant lesquelles des User Stories (US) rédigées par le Product Owner (PO) sont implémentées par les développeurs. Les User Stories contiennent toutes les informations disponibles pour réaliser la tâche, et le travail effectué est démontré systématiquement à toute l'équipe, en fin de sprint.

D'autres rituels ponctuent le sprint : les discussions quotidiennes (standups), le raffinage du backlog, les rétrospectives,...

La place de l'UX

Les méthodologies agiles **ne définissent pas précisément la place de l'UX designer** aux côtés de l'équipe développement et l'Agile Alliance, qui fait référence sur le sujet, a dû émettre des recommandations spécifiques[7].

Pourtant, le côté multidisciplinaire et l'accent mis sur la collaboration sont des points communs entre l'UX et l'Agile et les deux peuvent s'accorder. Pour Page Laubheimer[8], de Nielsen Norman Group, les facteurs clés du succès sont le soutien du management, la flexibilité du process, l'intégration de l'UX designer dans l'équipe de développement et un comportement proactif de l'UX designer

Enfin, Jeff Gothelf et Josh Seiden ont théorisé le Lean UX[9], qui combine les préceptes des méthodes agiles, du Design thinking et de la Lean Startup et fournit des outils méthodologiques UX plus adaptés au rythme soutenu et fragmenté du développement agile.

---

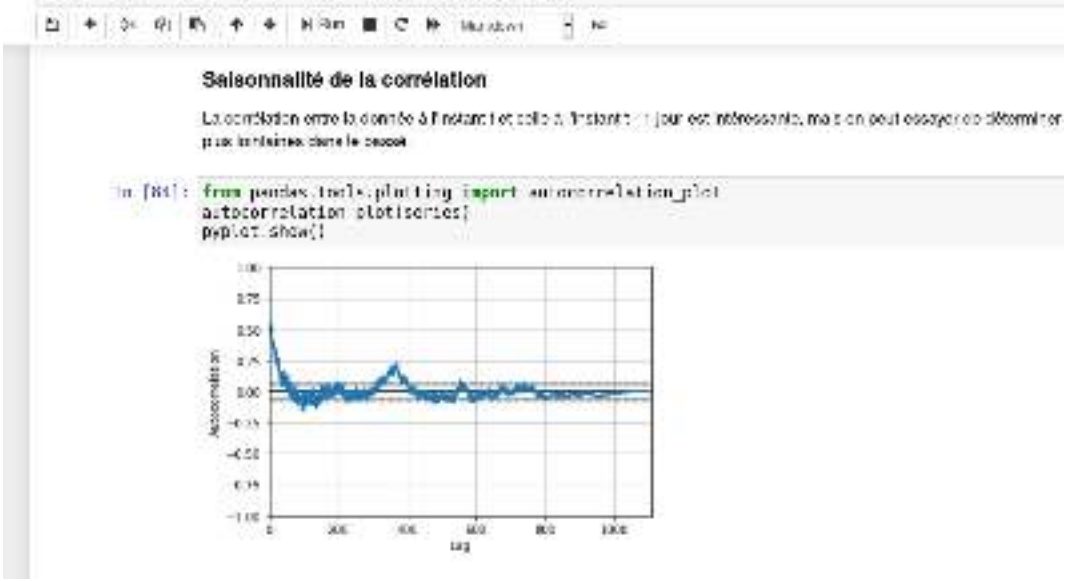
[6] "La place de l'UX dans la stratégie des entreprises", Designers Interactifs, 2018, https://www.slideshare.net/designersinteractifs/la-place-de-lux-dans-la-strategie-des-entreprises-2018
[7] "UX Within a Sprint? Designers Part of a Cross-Functional Team? Yes, Design Can be "Done"!", Agile Alliance, https://www.agilealliance.org/resources/sessions/ux-within-a-sprint-designers-part-of-a-cross-functional-team-yes-design-can-be-done-2/
[8] Page Laubheimer, "Attributes of Effective Agile UX", 2018, https://www.youtube.com/watch?v=XLvx-hCmKPk
[9] Jeff Gothelf & Josh Seiden, "Lean UX: Designing Great Products with Agile Teams", 2016

Outils pour la coopération UX/Backend

Sans laisser apparaître des outils spécifiques à la coopération UX/Backend, la recherche a toutefois permis de repérer des outils numériques pouvant être le lieu de dialogue entre les deux disciplines. Chacun de ces outils est très spécialisé et focalisé sur une technologie ou une pratique, mais l'ensemble laisse tout de même entrevoir un certain potentiel dans cette voie-là.

Jupyter notebooks : Analyse d'un problème UX

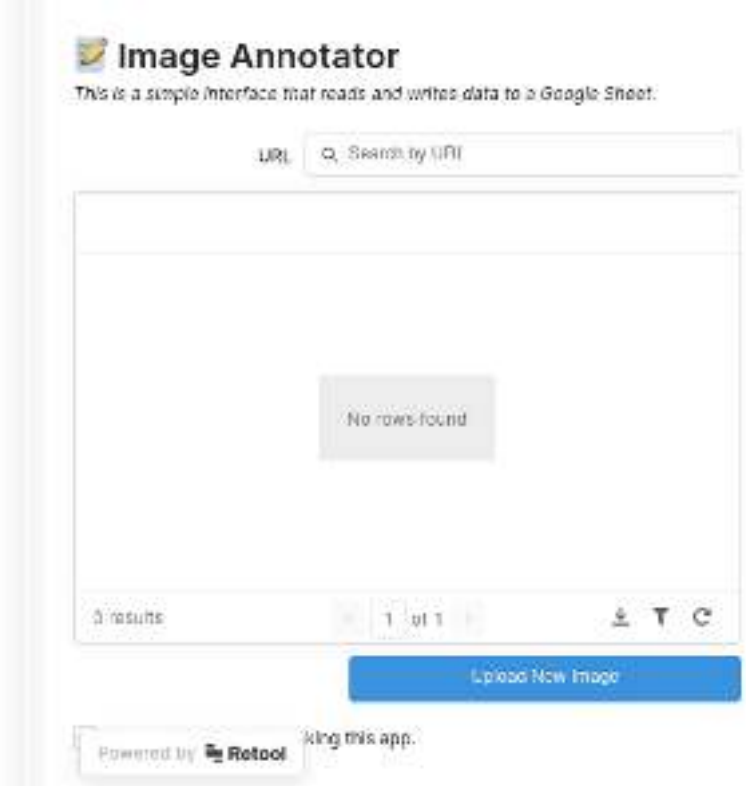
Les notebooks Jupyter sont des cahiers de laboratoires virtuels où l'on peut exécuter du code (souvent du langage Python). Cet outil permet de manipuler des données d'un backend Python et de prendre des notes structurées et mises en forme à côté. Jupyter offre ainsi l'opportunité de documenter un effet constaté sur l'UX, de vérifier la reproductibilité du problème, d'identifier les modèles concernés dans le backend et éventuellement de trouver et documenter des pistes de résolution.
Cependant, l'outil ne permet pas la collaboration à plusieurs utilisateurs en temps réel.



Capture d'écran de Jupyter Notebooks

Retool : Prototypage orienté données

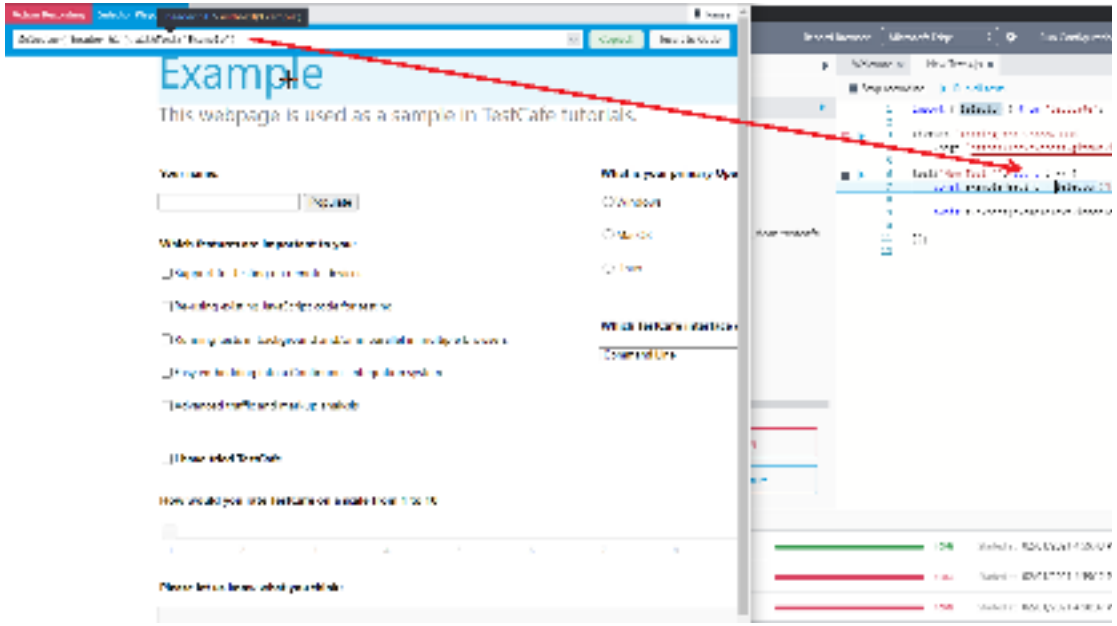
Retool est un outil en ligne qui offre la possibilité de construire des prototypes d'interfaces en quelques clics. L'intérêt réside dans le fait que l'on puisse connecter Retool au vrai backend de son produit : les interfaces sont ainsi beaucoup plus réalistes, fonctionnellement, et permettent de se confronter aux cas particuliers qui existent dans la base de données.
Aucune personnalisation graphique n'est possible, cependant, ce qui empêche de l'utiliser pour du maquettage dans le cadre de l'UX.



Capture d'écran de Retool

TestCafé Studio : Formalisation de scénarios d'usage

Le logiciel TestCafé Studio permet d'enregistrer facilement des parcours d'utilisation d'un site web et de rejouer ces scénarios. Il offre ainsi la possibilité de vérifier que les parcours utilisateurs prévus sont toujours valides sur le produit fini, et d'automatiser cette vérification, ainsi que d'introduire de nouveaux scénarios de tests quand on corrige un problème d'UX dans le produit.



Capture d'écran de TestCafé Studio

Recherches utilisateurs

Pour affiner la problématique de travail, une campagne d'entretiens utilisateurs a été organisée.

Objectifs des entretiens

Les entretiens doivent nous éclairer sur la manière dont le travail s'articule entre UX designers et développeurs backend dans le processus agile. Dans ce cadre, on cherche à comprendre **le chemin d'une proposition UX** avant qu'elle n'arrive entre les mains des développeurs pour être implémentée, mais aussi les **rôles exacts joués dans ce processus** par les personnes interrogées. Enfin, on peut déterminer le **niveau de compréhension mutuelle** qui peut exister entre les deux corps de métier.

Méthodologie

Dans cette optique, la méthode des entretiens semi-directifs a été retenue, pour recueillir des données qualitatives à partir de questions ouvertes. La technique des triptyques Faits/Hypothèses/Questions a été utilisée pour préparer le guide de ces entretiens (cf Tryptique et guides d'entretiens, en Annexe p.48).

Recrutement

La recherche utilisateurs s'est limitée à des UX designers et développeurs backends qui travaillaient sur des projets informatiques impliquant l'autre corps de métier, dans un contexte agile.

Une attention particulière a été portée sur le fait de recruter des personnes d'équipes différentes, pour réduire les biais liés aux organisations elles-mêmes et pour favoriser l'émergence de résultats que l'on pourra généraliser. Cependant, un tel recrutement a été compliqué par le travail distanciel, qui ne favorise pas le réseautage entre les entreprises, et par l'approche de la période estivale qui voit la charge de travail augmenter dans des équipes aux effectifs diminués.

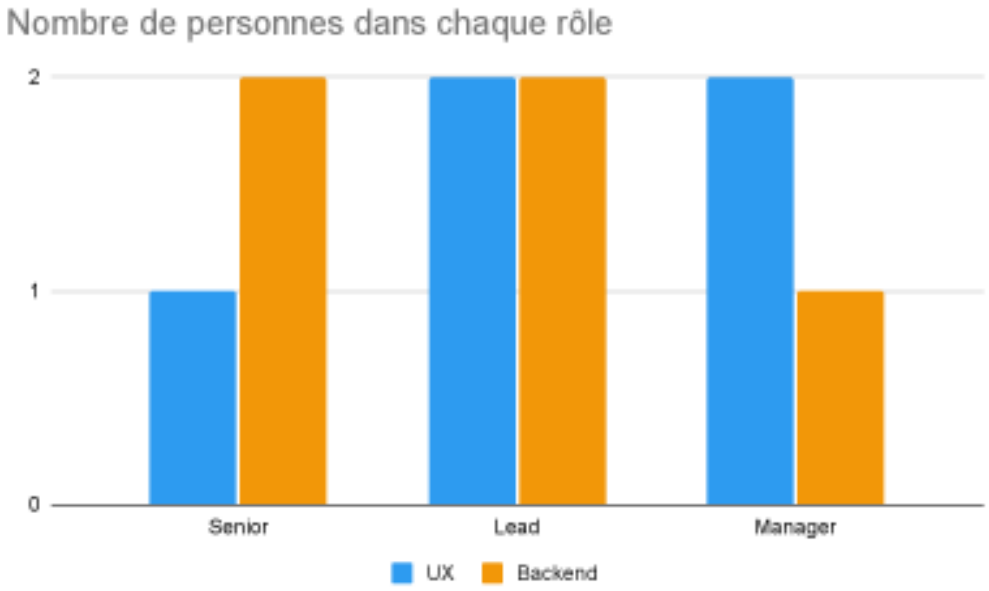
Dans ce contexte, l'objectif de recrutement a été fixé à cinq personnes par corps de métier pour rendre possible la découverte de la majorité du champ des connaissances sur le sujet[10]. Dix personnes ont donc été interrogées (cinq développeurs backend et cinq UX designers), dans huit équipes différentes de sept entreprises distinctes.

[10]Michael Seaman, "The right number of user interviews", 2015, https://medium.com/@mitchelseaman/the-right-number-of-user-interviews-de11c7815d9

Profil des personnes interrogées

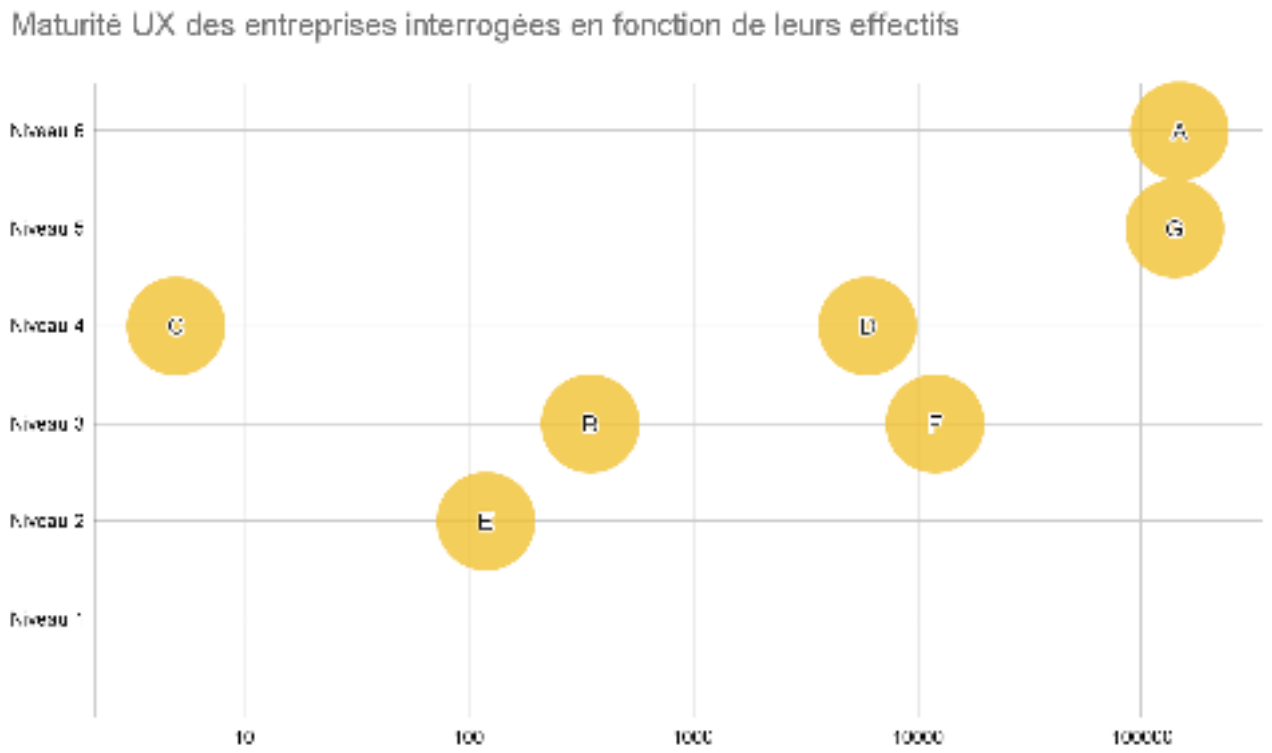
Expérience des personnes interrogées

Le recrutement a amené des personnes plutôt expérimentées dans leur métier, ayant parfois un rôle senior, lead et même manager ou CTO. Ceci constitue un biais, mais l'on peut estimer que ce niveau d'expérience a plutôt enrichi les échanges.



Profil des entreprises

Les structures dont font partie les personnes interrogées sont des entreprises de différents secteurs (numérique, énergie, conseil, robotique), à des niveaux de maturité UX différents (d'après l'UX Maturity Model de Renato Feijo[11]) et d'effectifs variants de plusieurs ordres de grandeur.



[11] Renato Feijo, "Planning your UX strategy", 2010, https://renatofeijo.com/blog-UXstrat.html

Résultats

Les personas

La recherche utilisateur nous permet avant toute chose de constituer deux personas correspondant aux deux profils qui nous intéressent.

L'UX designer : Yu Higgs

Yu Higgs

Âge
32 ans

Métier
UX designer en startup

Sens du défi technique

Empathie utilisateurs

Créativité

Assurance en groupe

Biographie

Formation aux Gobelins

A un peu travaillé en agence, mais ce n'était pas son truc

Apprécie son poste actuel, où l'équipe ne change pas trop

Aime retravailler le produit en permanence en capitalisant sur les connaissances de l'équipe

Personnalité

Un peu exubérante

Vient au contact pour discuter

Gribouille souvent

Travaille en musique

Semble éparpillée

Part du principe qu'elle a raison

Objectifs

Répondre rapidement aux nouvelles attentes des utilisateurs

Travailler de manière rationnelle, à partir de données fiables

Trouver *la* fonctionnalité qui va rendre le produit incroyable

Frustrations

Devoir répéter 150 fois les mêmes informations, pour justifier ses choix

Découvrir des problèmes tard parce quelqu'un ne les a pas exprimés avant

Être limitée par la technique

Le développeur backend : Ken D'Ba

Ken D'Ba

Âge
32 ans

Métier
Lead Backend

Sens du défi technique

Empathie utilisateurs

Créativité

Assurance en groupe

Biographie

Formation d'ingénieur

A toujours été développeur backend, mais a fait beaucoup de tâches connexes dans ses postes précédents

A pris la casquette de Lead avec plaisir, parce qu'il aime aider ses collègues quand ils sont bloqués

Personnalité

Peu communicant à l'écrit

Aime décortiquer un problème à plusieurs

Planifie avant d'agir

N'aime pas les réunions

Se sent peu créatif

Est ordonné

Objectifs	Frustrations
Maîtriser la modélisation du problème qu'on lui donne	Avoir des consignes floues ou incomplètes
Produire du code simple et stable	Écrire du code bancal à cause de choix de conception incompréhensibles
Limiter les tâches annexes (infrastructure, maintenance,...)	Ne pas être reconnu pour son expertise et son travail

Note

Dans la suite du travail, on différencie certains éléments par profil concerné :

les **UX designers sont en bleu** et les **développeurs backend en orange.**

Le parcours utilisateur

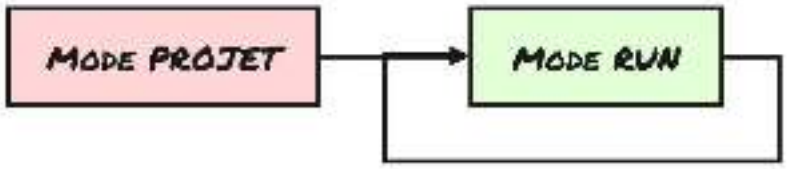
On tire aussi de cette recherche utilisateurs des apprentissages sur le parcours des UX designers et des développeurs backend dans la ligne de production logicielle.

Les modes de production

D'après les entretiens effectués, le travail de production informatique peut se dérouler dans deux modes distincts :

- le mode "Projet", où l'équipe conçoit le produit "en partant de zéro" (ex.: le lancement du site vitrine d'un nouveau modèle de vélo)
- le mode "Run", où l'on développe un produit existant, en l'améliorant en fonction de certaines métriques

Mode Projet	Mode Run
Processus linéaire	Processus itératif
Équipe fraîchement constituée	Équipe stable, avec des habitudes/rituels
Benchmark des expériences de la concurrence	Métriques UX mesurées sur le produit
Cible utilisateurs définie par le brief	Utilisateurs réels à disposition
Travail massif	Travail découpé



La plupart des personnes interrogées ont utilisé les deux modes de travail dans leurs différentes expériences professionnelles.
Cependant, peu ont vécu les deux modes sur un même projet, en assurant l'amélioration continue d'un produit qu'elles ont conçu. D'autre part, certains projets (souvent dans le cadre du travail d'agence) s'arrêtent après la première livraison et ne comportent pas de travail d'amélioration en mode Run.

Le mode Projet, parce qu'il est linéaire, laisse peu de place à l'erreur et nécessite de cadrer tôt et explicitement les responsabilités de chacun, les livrables UX pour transmettre l'information, les marges de manœuvre, etc… L'intention est souvent donnée par le côté business, comme une stratégie marketing ou des opportunités à saisir - une conception centrée utilisateur a un intérêt certain pour susciter de l'engagement ou consolider une image de marque, mais est pleine de risque parce que le mode de travail est peu propice à la révision des propositions (ou alors, avec des ambitions à la baisse).

Le mode Run est propice à une démarche UX efficace, car sa nature itérative promet une opportunité d'amélioration continue, d'adaptation du produit aux facteurs humains qui sont forcément changeants. De plus, il donne **l'opportunité d'optimiser à la volée les pratiques de l'équipe** en termes de workflow, de points de contacts, de livrables/passations, etc… Plus le travail est découpé, plus les cycles sont courts et plus l'équipe a l'occasion de corriger le process. Ce mode de travail est donc le plus favorable à l'amélioration de l'articulation UX/backend.

Le type de fonctionnalités

Les entretiens ont souligné que les équipes ne travaillent pas seulement sur la résolution de problèmes d'expérience utilisateur, mais aussi sur les corrections de bugs et sur des tâches techniques (parfois appelées "spike") demandées par les équipes de développement elles-mêmes.
Même si l'UX designer peut être impliqué sur ces deux autres types de tâches, il convient de focaliser notre travail sur les tâches qui naissent de problématiques UX appuyées par des mesures quantitatives ou qualitatives : ces données permettent de justifier les choix UX et on se prémunit ainsi de situations où la proposition UX, plus arbitraire, est plus difficile à défendre.

La variabilité de l'organisation

Enfin, les entretiens utilisateurs ont permis de constater que, même en se limitant au mode Run, la chaîne de production logicielle varie de manière notable d'une équipe à l'autre, en fonction de la structure de l'équipe, des rôles existants, des procédures en place, des rituels informels,...

Un parcours pour améliorer l'UX en mode Run

Le diagramme ci-dessous décrit un parcours qui généralise les différents cas rencontrés et récapitule les différentes étapes traversées, depuis la manifestation d'un problème d'UX sur le produit jusqu'à l'évaluation de l'impact de la solution livrée aux utilisateurs.

Le parcours se divise en trois macro-étapes :

1. Le **cadrage du problème** à résoudre
2. La **conception de la solution**
3. La **livraison** aux utilisateurs.

Première partie : Cadrage du problème

On retrouve dans cette première macro-étape le point de départ de la démarche UX et le premier des deux diamants qui la modélisent souvent.

Manifestation

Il est important de souligner que l'on se limite à un scénario où l'équipe résout un problème d'UX sur le produit. On ne s'intéresse ni à la résolution des bugs (où le produit ne se comporte pas comme spécifié) ni de tâches dites "techniques" (refactoring, amélioration de performance sans impact utilisateur,...) qui résolvent des problèmes pour les développeurs.

L'amélioration du produit commence donc à la manifestation (par la mesure) ou au signalement (par la recherche) d'un problème d'expérience utilisateur.

Contextualisation

Le problème identifié est replacé dans son contexte UX et son contexte technique.

Pour l'UX, on cherche à définir d'où vient l'utilisateur quand il rencontre le problème, ce qu'il cherche à faire, l'émotion dans laquelle il se trouve,...

Du côté technique, on cherche les contraintes existantes qui encadrent l'activité de l'utilisateur au moment où le problème se manifeste, les problèmes de performance qui pourraient exister dans le cadre de son parcours,...

Problématisation

L'équipe définit précisément la problématique utilisateur qu'elle veut traiter, pour cadrer le champ de travail. On définit également les métriques clés qui montrent le problème et les objectifs à atteindre sur ces métriques pour estimer que le problème est résolu.

Deuxième partie : Conception de la solution

Le deuxième diamant s'inscrit dans la macro-étape de conception de la solution, dont l'aboutissement est marqué par la spécification UX.

Conception

Les techniques de créativité et d'UX permettent à l'équipe de trouver ensemble des idées pour solutionner le problème, de définir le nouveau parcours désiré.

Rationalisation

La solution envisagée est prototypée pour être mise à l'épreuve.
L'UX designer mène des tests pour lever les doutes de conception et vérifier que la proposition UX est valide.
Du côté développement, on étudie en détail la faisabilité technique, au vu du système existant.

Si la solution doit changer, on reprend à l'étape précédente de conception.

Spécifications UX

Une fois la proposition stabilisée, on documente formellement la solution.

L'idée est de décrire les concepts introduits ou modifiés par la solution proposée et de lister de manière exhaustive les interactions envisagées entre l'utilisateur et le système (entrée/sortie de données, calculs,...).

Troisième partie : Livraison aux utilisateurs

La dernière macro-étape capture le cœur du développement agile, à partir de la préparation des tâches jusqu'aux procédures de contrôle.

Spécification technique

Une partie de l'équipe, menée par le Product Owner, détermine les tâches à effectuer, leur découpage exact et leur ordonnancement, pour constituer un backlog.

À cette étape, les tâches sont censées être documentées de manière à lever les incertitudes pour l'implémentation.

Implémentation

L'équipe développe tout ce qui est nécessaire à la solution : logiciel, matériel, graphisme, copie,...

On fait abstraction, dans cette étape, de la manière précise dont l'équipe organise le développement.

Contrôle

On contrôle le parcours utilisateur livré avant la mise en production. Puis, on contrôle l'impact sur l'expérience utilisateur grâce aux métriques-clés définies plus tôt.

On ne s'intéresse pas dans cette étape aux contrôles sur la qualité du code.

Ressenti du parcours par les utilisateurs

Parce que l'on s'intéresse à la collaboration entre les deux classes d'utilisateurs, UX designers et développeurs backend, on choisit de mettre en regard leurs expériences en mêlant leurs courbes d'émotions pendant ce parcours.

Émotions

On note particulièrement l'écart émotionnel important pendant l'étape de conception, mais également une tendance de fond d'un ressenti plus positif pour les UX designers que pour les développeurs backend. L'étude des pain points et des gain points nous permet d'expliquer un peu les formes de ces deux courbes.

> **Note**
>
> Dans un souci de lisibilité, l'Experience Map qui rattache les pain/gain points aux courbes d'émotion est présentée en Annexe, p.50.

Pain points

Le manque de bagage technique des UX designers

Les utilisateurs relèvent très vite le fait que les UX designers produisent des propositions infaisables, par manque de connaissances techniques. Ce pain point explique à lui seul la chute d'humeur des UX designers à l'étape de rationalisation, et qui plombe la suite du parcours.

> *"Quand toi, UX, tu arrives avec des jolis mockups pas du tout implémentables parce que le backend est structuré d'une certaine manière, il y a ce gap de compréhension qui te tombe dessus."*
>
> *"Au début, quand tu travailles sur des outils backends, c'est un vrai challenge. Les gens te parlent et vraiment, tu ne comprends rien. C'est hyper déstabilisant, tu conçois des trucs que tu ne comprends pas."*

> *"Si ceux qui proposent des solutions connaissaient les technos, de meilleurs choix seraient faits."*
>
> *"Le danger, c'est d'avoir un UX qui est hors-sol par rapport à la faisabilité."*

L'approche non-centrée utilisateur des développeurs

De la même manière, il semble très clair que les développeurs ne sont pas outillés pour adopter une démarche centrée utilisateurs, ce qui se traduit par leur non-implication dans les premières étapes du parcours, jusqu'à la frustration de ne pas être inclus dans les décisions de conception.

> *"Les développeurs backend prennent toujours la perspective technique sur un problème, et veulent le résoudre avec des systèmes, des langages parce que c'est la formation qu'ils ont reçue et que c'est ce qu'ils ont choisi de faire. Comprendre l'utilisateur ou son parcours, ils n'en ont rien à faire !"*
>
> *"Je pense que les développeurs backend n'utilisent jamais le produit."*

> *"On met en production l'incompréhension du produit par les développeurs."*

L'autorité des spécifications UX

La question de pouvoir challenger les propositions UX est aussi évoquée avec frustration dans les deux métiers. En les subissant, les développeurs ont un mauvais ressenti des étapes de spécifications techniques et d'implémentation.

> *“Quand tu sors d’école, on ne t’a pas appris à quel point tes designs vont être critiqués, qu’ils devront être remodelés.”*
>
> *“Après un gros malentendu, j'ai été obligée de reprendre mon design pour chercher un compromis, un peu pour les utilisateurs, un peu pour éviter des refactoring côté backend.”*

> *“J’ai l’impression que ce que propose l’UX designer n’est jamais remis en cause”*

Les échecs de transmission

La transmission des spécifications UX aux développeurs se fait souvent à l’occasion du rituel de raffinage du backlog, mené par le Product Owner et où l’UX designer n’intervient pas souvent. Sans échange sur les propositions UX en amont, cette transmission indirecte des connaissances se révèle risquée et on voit les deux courbes d’émotion chuter entre la spécification UX et la spécification technique.

> *“D’après moi, tout était indiqué dans les livrables fournis, mais il ne les avait pas lus, je pense.”*
>
> *“Je montre tous les documents aux développeurs parce qu’en gros personne ne leur a expliqué le travail, quand ils prennent la tâche.”*
>
> *“J’ai essayé de faire de l’UX agile pendant des années, mais ça ne marche pas.”*

> *“Souvent, quand on lit les User Stories, les subtilités nous échappent. Mais ce n’est pas la faute de l’US, c’est notre faute à nous, les développeurs.”*

Le manque d’outils

Aux écueils culturels et organisationnels s’ajoutent les difficultés liées au manque d’outils adaptés pour collaborer :

> *“Dans Figma, on ne peut pas utiliser des données réelles dans les prototypes. On aimerait pouvoir injecter notre backend en live, pour mettre le design à l’épreuve.”*

> *“Si on met de côté les tests fonctionnels, je ne connais pas d’outils pour analyser des parcours d'utilisation directement à partir du code-source.”*

Gain points

La reconnaissance mutuelle

Malgré ces frustrations, la reconnaissance mutuelle et une certaine solidarité existent entre les deux disciplines :

> *“On ne pense pas souvent le développeur backend comme étant à la base de la stratégie de l'entreprise, pourtant c’est vraiment ça.”*
>
> *“Avec les développeurs, on est un peu les seuls à avoir vraiment les mains dans le cambouis.”*

> *“L’UX designer est la seule personne qui s'occupe du bien-être des utilisateurs avec une approche scientifique/expérimentale.”*

La volonté de s’investir

Les courbes d’émotion s’alignent sur deux points de contact : la rationalisation de la proposition et la phase de contrôle. Chaque partie paraît prête à participer à d’autres échanges, notamment pour la critique constructive de la proposition UX.

> *“Quand on commence à définir les parcours, s'ils peuvent me dire "ça c'est super facile à faire" ou "il existe telle techno", ça peut m'aider à prendre de meilleures décisions en amont.”*
>
> *“Le rôle de l'UX est d'être transparent et itératif. Je montre les designs super tôt, quand ce n'est pas fini du tout du tout. On met un front et un back dans une salle et on cherche les compromis pour obtenir la faisabilité, ça marche bien.”*

> *“En ce moment on a des développeurs qui voudraient participer au choix des problématiques, en amont.”*
>
> *“Je ne veux pas critiquer les décisions prises, juste les mettre en discussion.”*

Des outils pour collaborer

Les outils de tableau blanc virtuels type Miro, Mural ou FigJam ont été largement mentionnés par les UX designers et sont utilisés (dans le cadre d’ateliers, pas forcément liés à l’UX design) par une partie des développeurs interrogés. Ils peuvent ainsi être des lieux d’échanges entre les UX designers et les développeurs backend.

Synthèse

La recherche utilisateurs nous a permis de définir deux personas (UX designer et développeur backend) et de préciser le cycle de développement agile dans lequel leurs interactions s'inscrivent. À partir des ressentis des utilisateurs interrogés sur ce parcours, nous avons pu mettre en lumière des difficultés que l'on peut regrouper en trois grands thèmes :

- un **fossé culturel** et de compétences entre les deux profils
- une **transmission compliquée** des propositions UX liées à l'organisation agile du travail
- un **manque d'outils** pour collaborer entre UX designers et développeurs backend

Problématique

Pour rappel, le cadrage nous indique que la problématique concerne des équipes de développement qui travaillent en agilité à l'amélioration d'un produit, en adoptant une démarche d'UX design et de séparation des préoccupations frontend et le backend.

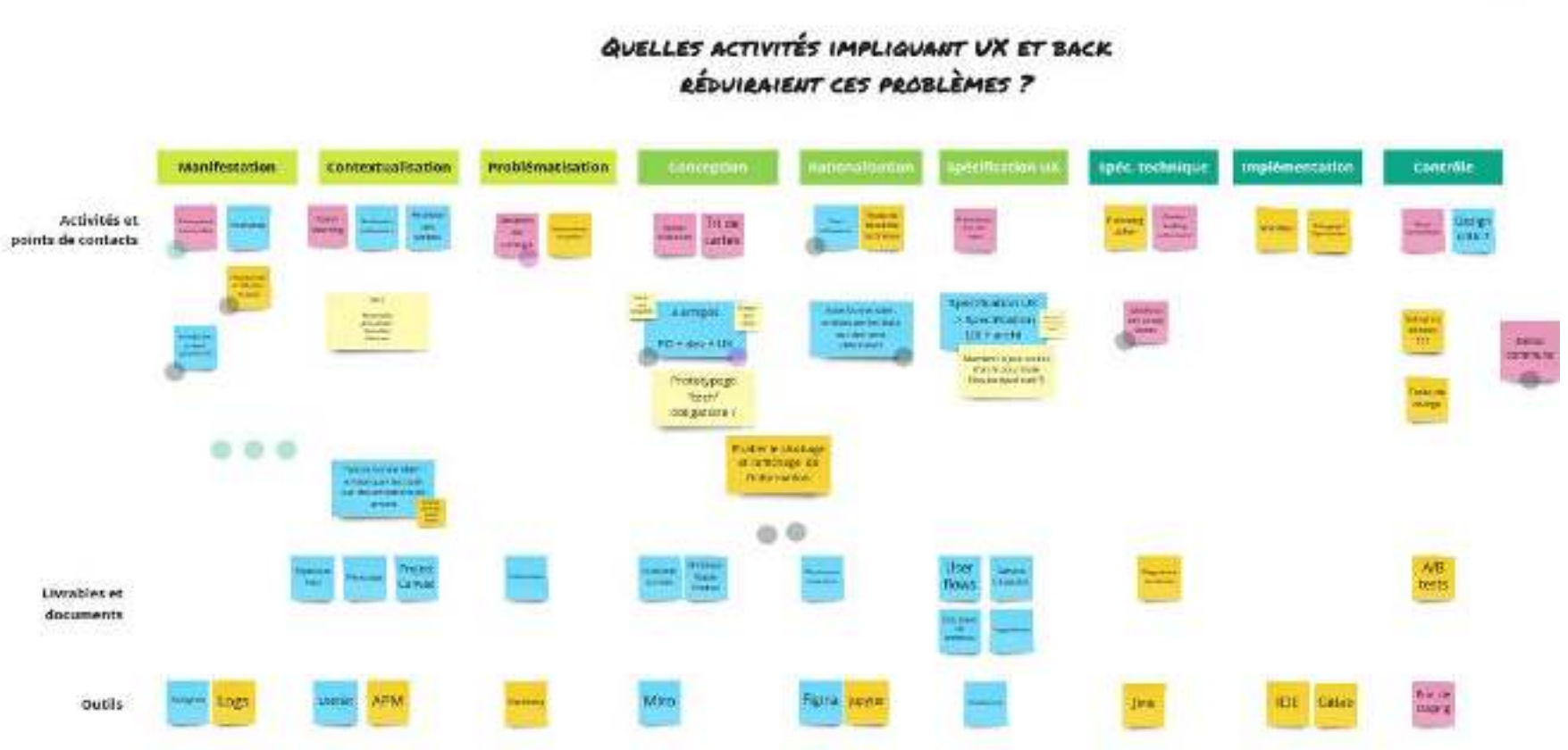
Au vu des résultats de la recherche utilisateurs, on pose ainsi la problématique suivante :

> Comment **rapprocher culturellement** les UX designers et les développeurs backend, en consolidant les propositions UX, mais **sans interférer avec le process agile** ?

Construction d'une solution

Atelier brainstorming

Pour favoriser la génération d'idées, un atelier d'idéation a été réalisé avec deux UX designers (dont un qui anime l'atelier), trois développeurs backends (dont un Lead backend) et un Product Owner.



La piste des Tres Amigos

Dans cet atelier d'idéation, des échanges ont notamment eu lieu à propos du rituel agile des "Tres Amigos"[12]. Son principe est simple : le Product Owner, un développeur et un ingénieur qualité se réunissent avant le raffinage du backlog pour lire une User Story et la traduisent en tests fonctionnels, qui vont guider la bonne implémentation de la tâche. La forme de l'atelier (durée, cheminement de pensée) n'est pas définie.

Cet outil est peu répandu et n'a pas été détecté pendant la recherche exploratoire, notamment parce qu'il est hors-cadre pour notre problématique : il n'implique pas l'UX designer et intervient après la spécification UX, donc trop tard pour dérisquer la faisabilité.

Cependant, certains aspects sont intéressants. D'une part, l'exercice est **transdisciplinaire**. Il permet en outre, entre les différents participants, d'**aligner les visions** autour d'une User Story. Enfin, les tests fonctionnels produits lors de l'atelier sont **analogues à des parcours utilisateurs** pour l'UX designer.

Un atelier d'idéation peu productif ?

Il m'a semblé que peu d'idées exploitables avaient été émises pendant l'atelier d'idéation.

J'identifie plusieurs facteurs qui pourraient l'expliquer :

- une mauvaise gestion du temps (trop de temps à présenter la recherche, pas assez à manier les idées)
- une problématique de travail mal définie
- un déséquilibre d'effectifs entre les UX et les backends

Je n'ai pas eu la possibilité d'organiser un autre atelier en corrigeant ces points et ai construit ma proposition à partir de la piste des "Tres Amigos" et en capitalisant sur les résultats de mes recherches.

[12] Judicaël Paquet, "Les 3 Amigos en agile", 2019, https://blog.myagilepartner.fr/index.php/2019/02/06/les-3-amigos-agile/

Objectifs de conception

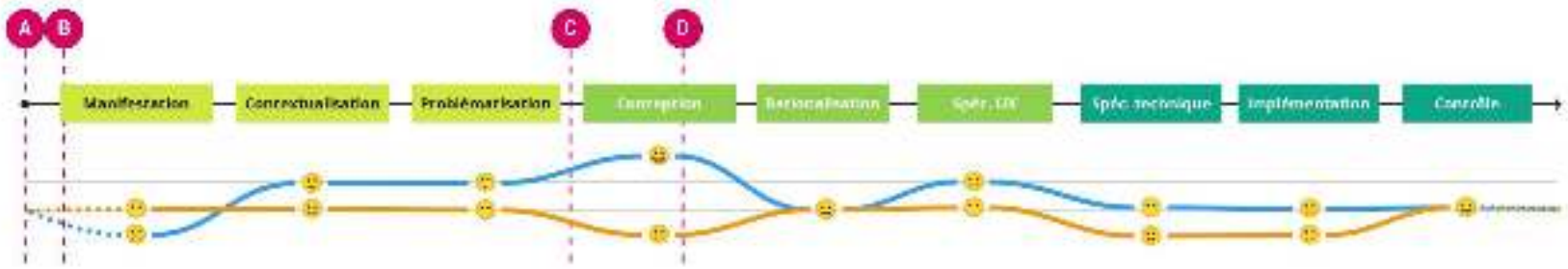
À partir de la problématique retenue, on dérive plusieurs objectifs de conception :

- **Intervenir tôt** dans le parcours identifié, avant le process de développement agile
- Faire **gagner du temps** à l'équipe, au global
- **Rester modulaire** pour que les équipes puissent adapter leur usage de la solution
- Capitaliser sur les **points communs** entre les deux disciplines

Proposition

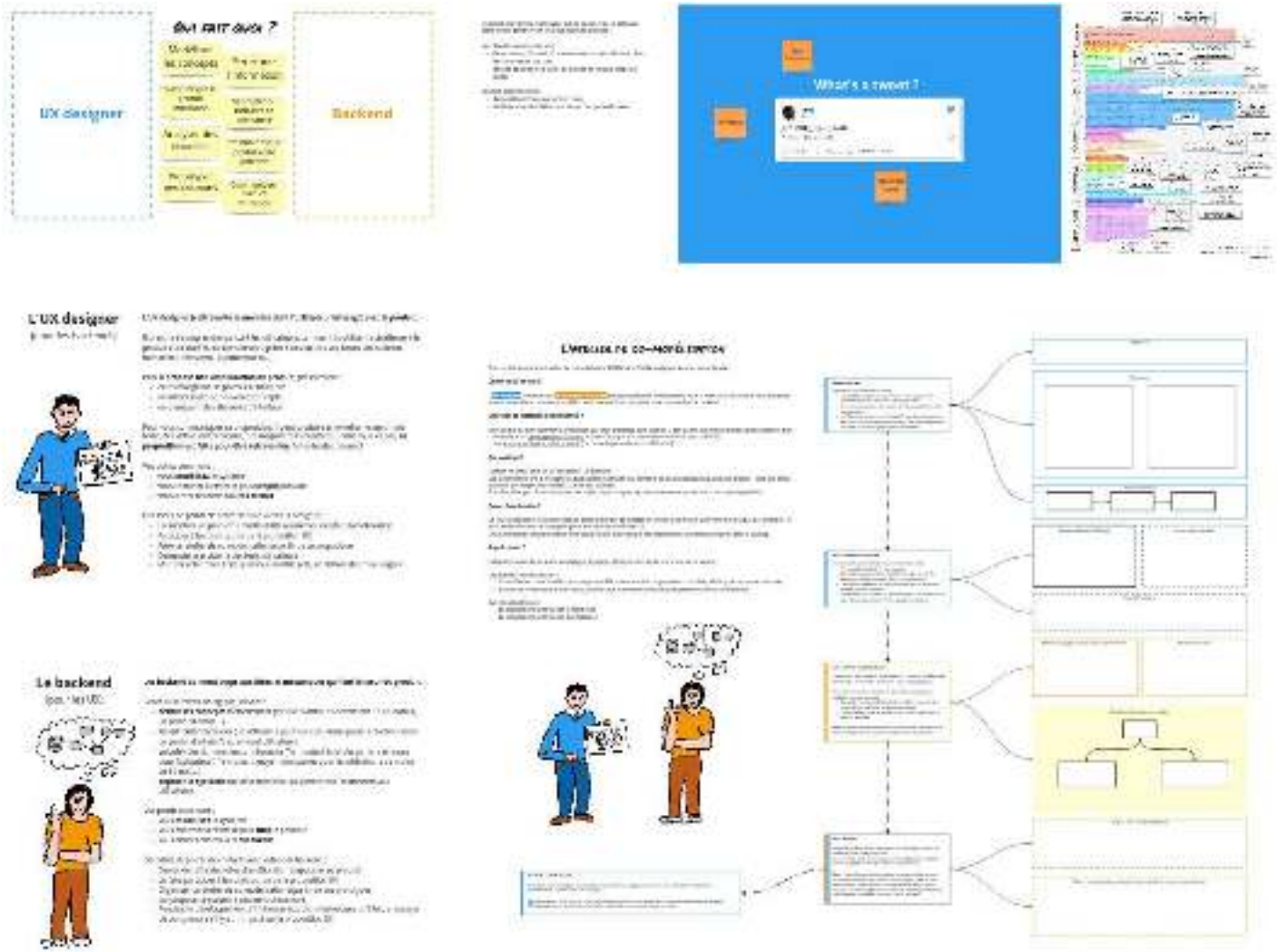
En poussant plus loin l'idée de créer un nouvel atelier, est venue celle de créer plusieurs activités indépendantes et complémentaires pour favoriser les échanges entre UX et backend. Plus accessible qu'une méthodologie de travail et moins lourde à développer qu'un outil logiciel, une telle "boîte à outils" permettrait aux équipes qui l'adoptent d'y piocher en fonction de leurs besoins.

En pratique, on propose un **ensemble d'activités**, soutenues par des templates Miro et de la documentation, pour permettre aux UX designers et aux développeurs backend de nourrir un dialogue, d'une part sur le produit et ses problématiques, et d'autre part sur leurs métiers en eux-mêmes. On vise le **début de cycle** pour construire les cultures et la **phase de conception**, où l'écart émotionnel est le plus grand.



Cette boîte à outils contient :

A. un icebreaker pour consolider la compréhension du métier de l'autre
B. un mémo sur chaque métier
C. un icebreaker pour découvrir les limites des wireframes
D. un guide d'atelier de co-modélisation



L'ensemble des templates Miro de la boîte à outils

A. Icebreaker : “Qui fait quoi ?”

Cet exercice rapide réunit les UX designers et les développeurs backend et peut être fait à tout moment du cycle de développement. D’autres personnes de l’équipe peuvent être invitées pour enrichir les discussions.
Un animateur gère la limite de temps (3 minutes pour trier les cartes) et anime les échanges qui ont lieu ensuite.

Déroulement

Des cartes où sont inscrites des tâches sont disposées sur le canevas, dans une zone neutre. L’animateur demande aux participants de déterminer ensemble qui a la responsabilité de ces tâches entre les UX designers et les développeurs backend. Pour cela, les participants les déplacent dans les zones correspondantes ou les laissent dans la zone neutre si la responsabilité est partagée.

Les tâches sont :

- Modéliser les concepts
- Structurer l’information
- S’assurer que le produit fonctionne
- Se soucier du bien-être de l’utilisateur
- Analyser des données
- S’assurer que le produit reste pérenne
- Prototyper des solutions
- Communiquer avec les frontends

Pourquoi cet exercice ?

Les cartes sont préparées pour être ambiguës et chacune peut être interprétée comme incombant à chaque corps de métier. Leur meilleure place est donc dans la zone neutre, mais l’important est de discuter les choix qui ont fait l’unanimité : par exemple, “Pourquoi prototyper des solutions n’appartiendrait qu’aux UX designers ?”.

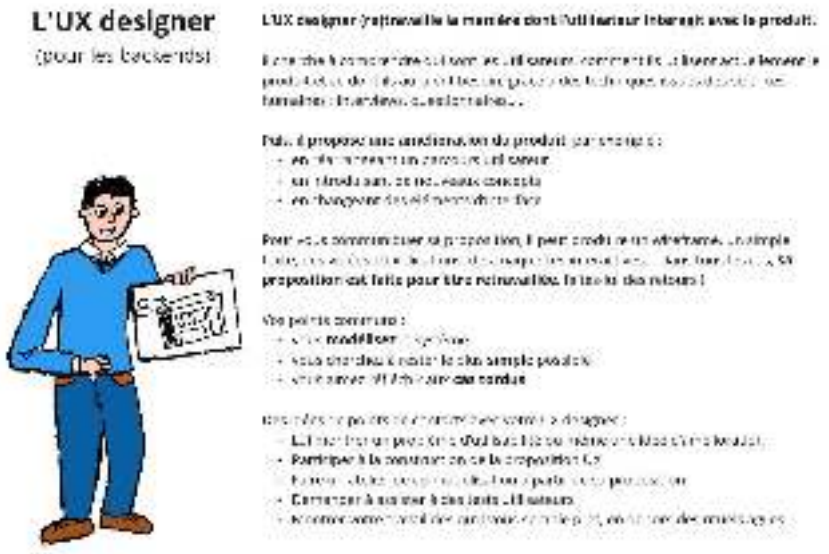
Cet exercice permet d’**initier un rapprochement** entre UX designers et développeurs backend car il montre bien certains **points communs** et analogies des deux métiers. De plus, il peut révéler dans l’équipe certains **malentendus sur les responsabilités** de chacun.

B. Mémos des métiers

Le deuxième outil est une paire de fiches “mémos” présentant le métier d’UX designer et celui de développeur backend. Elles visent à améliorer la compréhension mutuelle.

Chaque fiche est rédigée à l’attention de l’autre corps de métier, et notamment aux personnes qui ne sont pas acculturées à l’autre discipline. Ainsi, elles dressent un portrait simple mais correct, qui récapitule **les missions** de chaque poste. Elles mettent aussi en avant **les points communs** qui existent entre UX designers et développeurs backend. Enfin, des idées d’activités pour **initier un dialogue** avec l’autre y sont aussi indiquées.

Ces fiches peuvent être mises à disposition d’une équipe à tout moment ou présentées activement, par exemple après l’icebreaker “Qui fait quoi ?” présenté ci-dessus. L’objectif est qu’elles servent de points de repère dans les équipes où la relation UX/backend est naissante ou déjà bien établie.

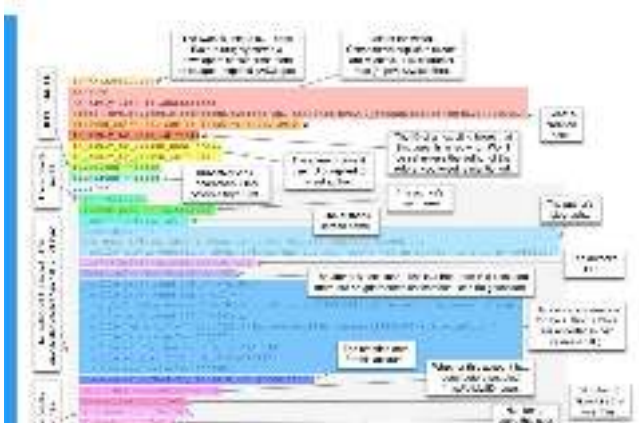


C. Icebreaker : What's a tweet ?

Cet exercice rapide (5-10 minutes) est prévu pour les UX designers et les développeurs backend, avec un animateur, et peut être réalisé avant ou pendant la phase de conception.

Déroulement

Un animateur présente la capture d'écran d'un tweet et demande aux participants de lister toutes les informations qui le composent. On part de ce qui est visible mais on va aussi loin que possible : on cherche à **retrouver le modèle sous-jacent**. D'autres personnes de l'équipe, comme le Product Owner, peuvent se joindre à l'atelier.



Le groupe n'aura pas capturé l'entièreté du modèle : d'après Twitter, il est composé de plus 90 attributs ! L'animateur dévoile la liste de ces attributs et peut montrer qu'on y retrouve les réponses données par l'équipe. Après un rapide tour de table au sujet de la surprise sûrement ressentie, l'animateur fait le lien avec les **faiblesses des maquettes UX** quand elles sont transmises sans explications complémentaires.

Pourquoi cet exercice ?

Cet exercice rapide montre qu'une représentation graphique (une capture d'écran, une maquette, un wireframe,...) ne peut pas transmettre l'ensemble des informations utiles à l'implémentation d'un système et qu'il convient donc d'organiser des échanges autour de celles-ci.

Cet icebreaker introduit également la notion de **co-modélisation**, abordée dans l'atelier suivant.

D. Atelier de co-modélisation

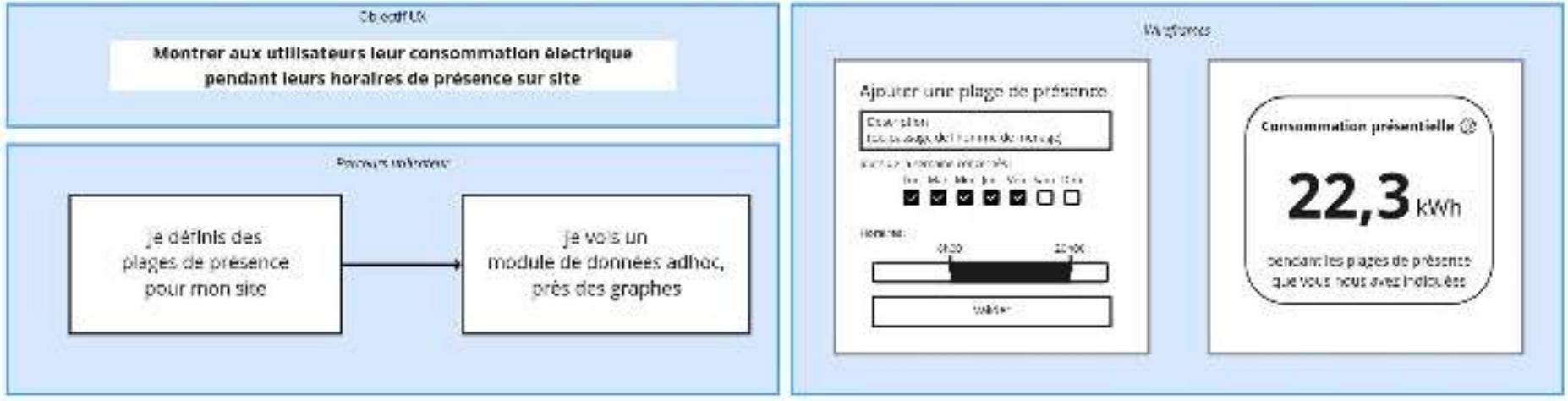
Les développeurs backends et les UX designers sont habitués à **modéliser les problèmes** qu'ils traitent, mais sous des formes différentes. Nous introduisons un atelier où cette activité de modélisation est réalisée en commun, pour dérisquer une idée de fonctionnalité. L'atelier doit être réalisé pendant l'étape de conception, dès lors que l'UX designer a un wireframe, même basique ou incomplet.

L'atelier est prévu pour deux personnes : un Lead UX (qui anime l'atelier) et un Lead backend. Le fait d'avoir des profils Lead n'est pas obligatoire, mais cela favorise une capacité à se projeter face à des informations incomplètes. Pour des échanges plus riches ou sur des sujets risqués, on peut ajouter un deuxième participant pour chaque spécialité.

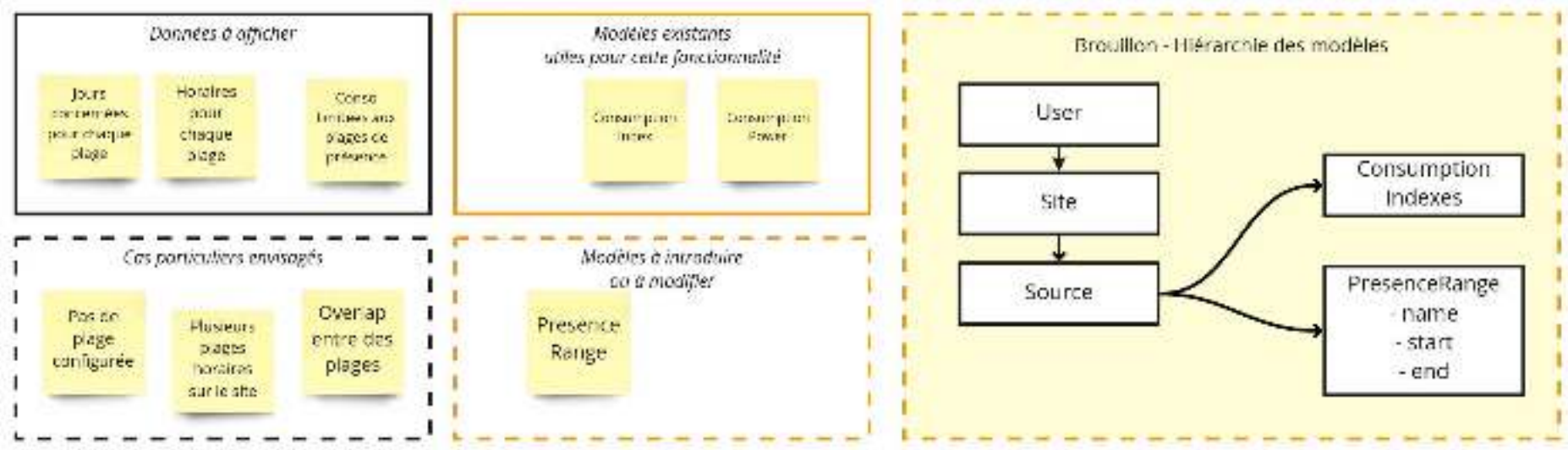
L'atelier ne doit pas durer plus d'une vingtaine de minutes, et pourra être réalisé plus rapidement avec l'habitude.

Déroulement

L'UX designer présente à un développeur backend une idée de fonctionnalité, sous la forme d'un objectif UX, d'un wireframe et d'un parcours utilisateur simplifié. Il est critique que la proposition présentée ne soit pas trop complexe et... qu'elle implique un développement backend.



Puis, le binôme remplit un canevas pour déterminer comment cette idée va se traduire dans le backend.



Ensemble, ils listent d'abord les données nécessaires à la fonctionnalité (en partant de celles visibles dans le wireframe) et les cas particuliers qu'ils peuvent déjà envisager. L'UX designer, toujours animateur, demande au backend de définir les modèles qui seront utiles dans l'implémentation existante et ceux qu'il faudra introduire - il l'incite également à faire un brouillon de leur hiérarchie.

C'est là que se fait la **co-modélisation**, où les échanges naissent autour de la manière dont l'idée va être implémentée dans le système.

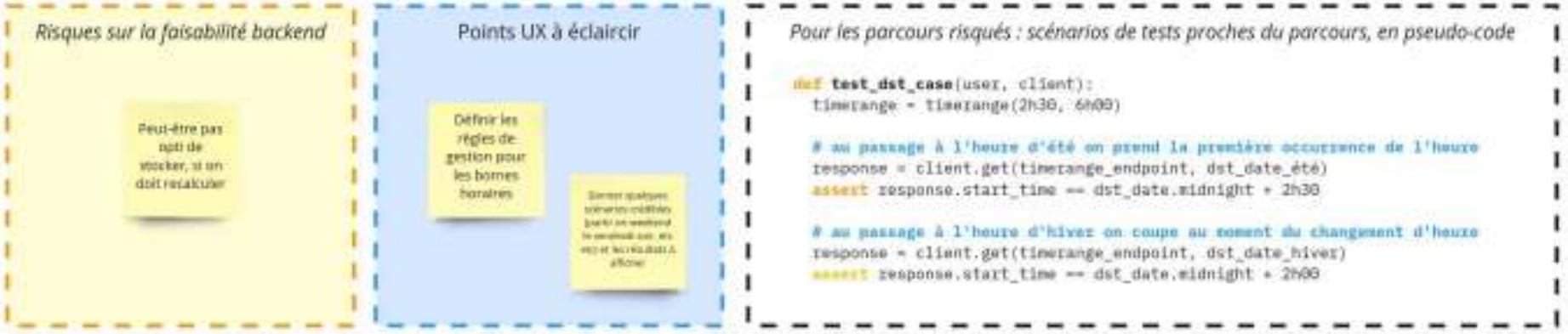
Attention, pas de spécification !

L'atelier est mené pendant la phase de conception, c'est-à-dire que tout peut être remis en question dans la proposition UX, et tout peut changer dans la stratégie d'implémentation. Rien de ce qui est produit ne peut être considéré comme une spécification.

Cet atelier se veut être avant tout un espace de dialogue entre les deux métiers.

Pour finir l'atelier, le binôme détermine les risques de faisabilité qui pèsent sur l'implémentation backend et les points à éclaircir côté UX, peut-être sous la forme de nouveaux wireframes ou parcours.

Si les discussions font émerger des scénarios qui sont propices aux malentendus, un espace est prévu pour les décrire avec le comportement attendu du système (l'idée est d'écrire des pseudo-tests, sous une forme lisible par les UX et les backends).



Pourquoi cet atelier ?

Cet atelier permet avant tout d'établir un dialogue entre les UX designers et les développeurs backend, autour d'une activité concrète. Mais en alignant leur vision autour de la fonctionnalité, les participants découvrent des zones d'ombre et réduisent ainsi le risque associé à son développement.

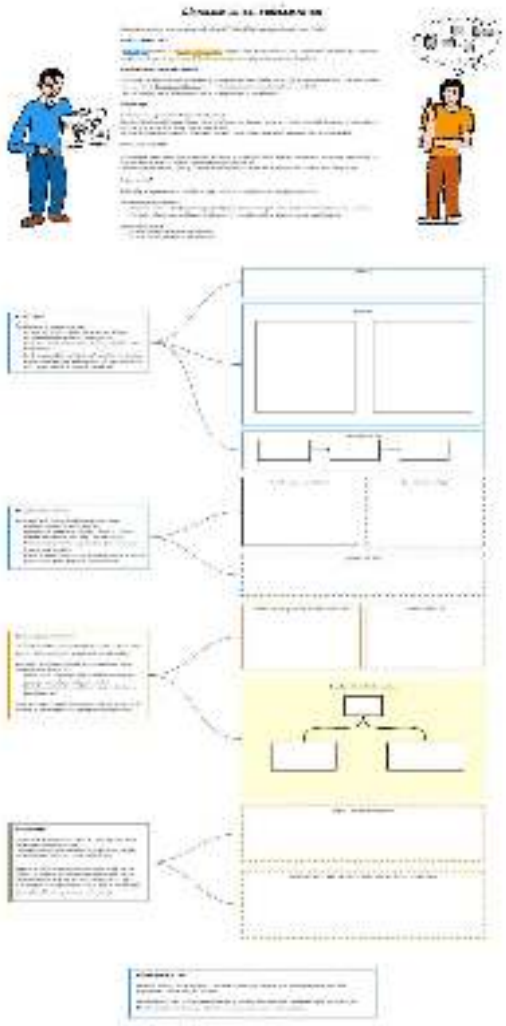
L'atelier peut faire gagner du temps à l'UX designer en lui indiquant ce qui n'est pas clair et qu'il faut prototyper - et ce qui au contraire, n'appelle pas de travail supplémentaire. Côté backend, la co-modélisation permet de donner une piste de travail à la personne qui implémentera la tâche.

Il est à noter que l'on peut réaliser cet atelier régulièrement, sur des tâches différentes (tant qu'elles respectent les deux critères évoqués), mais qu'il n'est pas prévu pour être mené sur toutes les tâches, de manière systématique.

Évaluation de la solution

Prototype évalué

Le prototype que l'on évalue est un template Miro contenant le **canevas support** de l'atelier de co-modélisation et le **guide d'explications** associé, pour l'animateur.



Cible

Dans ce test, on cible des UX designers et des développeurs backend qui travaillent en mode Run, sur des projets impliquant de l'UX et du backend, en suivant les méthodes agiles. Le recrutement nous amène six testeurs (trois UX designers et trois développeurs backend, dont deux juniors et quatre seniors).

Scénario

Les utilisateurs jouent leur propre rôle, avec l'idée qu'on leur a demandé de mener ou de participer à un atelier de co-modélisation suite à des frictions récentes entre UX et backend.
Cet atelier se déroule en visio à deux, avec un accès au canevas fourni sur Miro.

Cas des UX designers

Les UX designers testent l'atelier en l'animant : je joue le rôle de développeur backend.
Ils choisissent une proposition UX qui leur tient à cœur dans un de leurs projets, sous réserve qu'elle ne soit pas trop complexe, mais qu'elle implique un développement backend.

En suivant les consignes du guide explicatif, ils commencent par documenter leur proposition UX dans les premières cases du canevas, pour la présenter au développeur backend.
Puis, ils complètent le reste du canevas avec le développeur backend, qui challenge leur proposition.

Cas des développeurs Backend

Pour les développeurs backend, j'anime l'atelier en tant qu'UX designer et ils y participent.

Je leur présente une proposition de fonctionnalité qui est fictive mais qui concerne le produit sur lequel ils travaillent.
Puis, je leur demande de m'aider à remplir les cases du canevas, en répondant au fur et à mesure aux questions qu'ils me posent sur la proposition UX.

Hypothèses

1. Les échanges provoqués par l'exercice apportent de l'information par rapport au wireframe
2. Toutes les informations nécessaires pour implémenter la tâche sont transmises
3. L'atelier plaît autant aux UX designers qu'aux développeurs backend

Mesures

Pour vérifier ces hypothèses, on met en place deux items d'évaluation à échelle, et le Net Promoter Score (NPS) :

- Sur une échelle de 0 à 10, à quel point les échanges ont-ils clarifié le wireframe présenté ?
- Sur une échelle de 0 à 10, à quel point vous sentez-vous confiant pour que la tâche soit implémentée sans échange supplémentaire ?
- Sur une échelle de 0 à 10, à quel point recommanderiez-vous cet atelier à vos collègues ?

Résultats

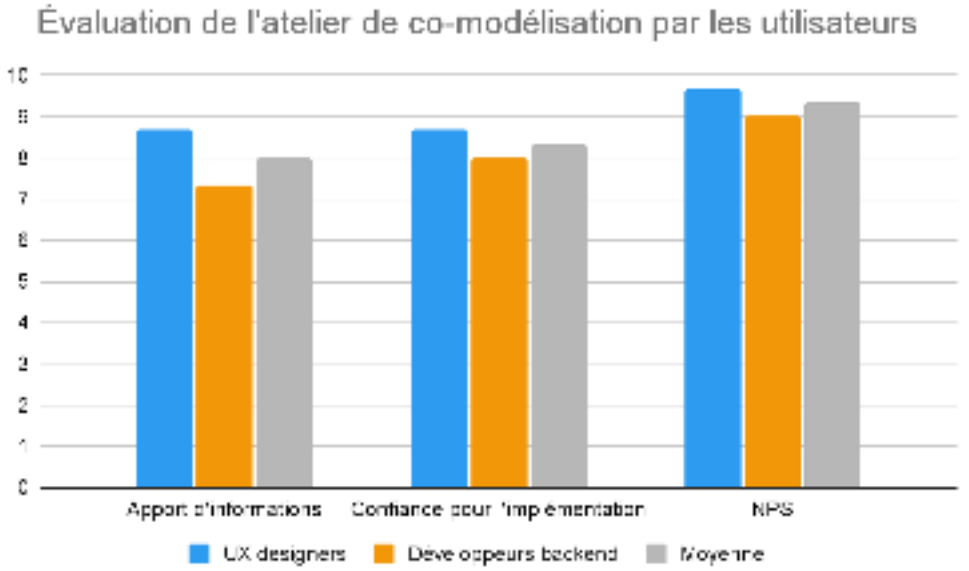
Comme premier signal d'évaluation de l'UX, le **Net Promoter Score atteint 100%** : c'est très encourageant !

Vérification des hypothèses

Les participants se posent tous en promoteurs de l'atelier, avec une **note moyenne de 9,3** - il n'y a aucun détracteur.

Une note moyenne de 8,0 a été donnée pour la question de la clarification par les échanges verbaux et de 8,3 pour la question de la confiance en l'implémentation de la fonctionnalité sans échange supplémentaire. Les **deux premières hypothèses sont donc vérifiées** sur ce prototype.

La troisième, malgré le très bon NPS, peut être questionnée. En effet, on constate sur chaque métrique que **les développeurs backend ont accordé de moins bonnes notes que les UX designers**. Cet écart est particulièrement net sur la question de la clarté apportée par les échanges (7,3 pour les backends contre 8,6 pour les UX).



Une partie de ce résultat pourrait s'expliquer par le fait que les scénarios soient finalement assez différents (les backends étaient exposés à une fonctionnalité fictive, alors que les designers présentaient une fonctionnalité qui leur tenait à cœur), mais **il conviendrait de mener de nouveaux tests pour le vérifier**.

Remarques complémentaires

La collaboration UX/backend fonctionne

Le format de l'atelier semble propice à la collaboration entre les deux corps de métier :

> "C'est très motivant de croiser nos compétences autour d'un même objectif.""Le format est propice à l'échange et à la clarté de cet échange."

La proposition UX est consolidée

Les participants comprennent bien la proposition UX et la font évoluer ensemble :

> "Je trouve ça bien, ça permet de déceler directement s'il y a des soucis dans la proposition."

> "C'est très intéressant de noter les cas particuliers, parce que tout le monde n'y pense pas, dans l'équipe."

L'équipe gagne du temps

Les utilisateurs ont le sentiment de gagner du temps en participant à cet atelier :

> "Je pense qu'on a gagné du temps, parce qu'on échange très vite à deux."
>
> "Tu découvres les mécanismes de pensée de l'autre et tu développes des réflexes qui vont te faire gagner du temps hors atelier"

Points de vigilance

Les utilisateurs ont toutefois relevé plusieurs points de vigilance concernant l'atelier :

- modéliser si tôt va influencer la modélisation finale, ce qui n'est pas forcément une bonne chose
- le côté méthodique de l'atelier pourrait se révéler trop lourd
- l'atelier n'a pas de livrable associé, ce qui peut le rendre difficile à justifier

Conclusion et Roadmap

Avoir été développeur avant d'avoir été UX designer est une chance qui m'aide à éviter certains écueils techniques dans mon travail et à entretenir un dialogue régulier avec l'équipe de développement.
Ce mémoire m'a permis de mieux comprendre ce qui rend particulière la relation UX/backend : l'écart culturel entre les deux disciplines, l'impact de l'agile sur la transmission des connaissances et le manque d'outils comme lieux d'échange.

Avoir structuré une pensée autour de ces problèmes nous a permis de faire émerger une solution, sous la forme d'un ensemble d'activités qui rapprocheront de manière constructive les UX designers et les développeurs backend. Cette boîte à outils a atteint ses objectifs auprès des premiers testeurs et l'atelier de co-modélisation va maintenant être utilisé régulièrement par mon équipe à Eco CO_2. Nous en verrons sûrement les effets d'ici quelques mois : travaillerons-nous enfin en toute compréhension et sans problèmes de faisabilité ?

La solution, en tout cas, ne demande maintenant qu'à être utilisée plus largement, et enrichie. Pour cela, elle sera traduite en anglais et publiée dans le catalogue de templates MiroVerse pour être disponible à tout utilisateur de Miro, qu'il soit UX designer, développeur backend, Product Owner,... Pour accompagner cette diffusion, une communication sera prévue sur des canaux appropriés (blog Eco CO_2, forums spécialisés, Meetups UX à Paris,...).

Pour faire grandir la proposition, plusieurs pistes que nous avons volontairement laissées de côté pourraient être explorées. Par exemple, comment adapter ces outils au mode Projet, où la transmission des propositions UX se fait souvent "Over the fence", sans réel échange ? Et même en mode Run, peut-on imaginer inverser le scénario et utiliser ces exercices pour dérisquer la *faisabilité UX* d'un changement dans le backend ?

Épilogue

La promesse de cette boîte à outils, au fond, est celle d'un monde où la responsabilité de l'UX est partagée par chacun. J'ai eu la chance d'interroger un développeur d'une structure très mature sur ce point, qui m'a calmement annoncé... notre fin ?

> *"Parce qu'on attend de nous, développeurs, d'être complètement autonomes, on est tous responsables de l'UX - on est tous UX designer. Du coup, on n'a plus d'UX designer."*

Remerciements

Ce travail n'aurait pas été possible sans les remarques passionnantes et les conseils avisés d'Antoine Pezé, en sa qualité de tuteur pour ce mémoire.

Mais il n'aurait absolument rien contenu sans les nombreuses et nombreux UX designers et développeurs backend des quatre coins d'Europe qui se sont pliés à mes (longs) entretiens, mes (interminables) tests et mes appels à l'aide (pendant les weekends et jours fériés) : Sandrine, Kévin, Robin, Axel, Emmanuel, Houssam, Nicolas, Jean-Christophe, Alexandre, Fanny, Diane, Samuel, Fabien, Geoffrey, Benjamin, Henri, Teddy.

Une mention particulière à Sylvia, qui pour moi a déjà eu sa certification UX depuis longtemps et qui a plusieurs fois fait rebondir mon étude.

Vous n'auriez pas lu un traître mot sur ce sujet sans le patient soutien d'Ariane, l'appui indéfectible de mes parents et la compréhension de Kim, au milieu de vacances en Auvergne.

Bibliographie

1. Contributeurs multiples, “Separation of concerns”, 2021, https://en.wikipedia.org/wiki/Separation_of_concerns
2. Jakob Nielsen, “Mental Models”, 2010, https://www.nngroup.com/articles/mental-models/
3. “Example payloads”, Twitter, 2021, https://developer.twitter.com/en/docs/twitter-api/v1/data-dictionary/object-model/example-payloads
4. “What’s in a tweet ?”, The Economist, 2011, https://www.economist.com/graphic-detail/2011/09/29/whats-in-a-tweet
5. Viktoria Menlychuk, "How Backend Developers Impact UX: Actionable Insights", 2020, https://dzone.com/articles/how-backend-developers-impact-ux-actionable-insigh
6. "La place de l’UX dans la stratégie des entreprises", Designers Interactifs, 2018, https://www.slideshare.net/designersinteractifs/la-place-de-lux-dans-la-strategie-des-entreprises-2018
7. "UX Within a Sprint? Designers Part of a Cross-Functional Team? Yes, Design Can be “Done”!", Agile Alliance, 2021, https://www.agilealliance.org/resources/sessions/ux-within-a-sprint-designers-part-of-a-cross-functional-team-yes-design-can-be-done-2/
8. Page Laubheimer, “Attributes of Effective Agile UX”, 2018, https://www.youtube.com/watch?v=XLvx-hCmKPk
9. Jeff Gothelf & Josh Seiden, “Lean UX: Designing Great Products with Agile Teams”, 2016
10. Michael Seaman, “The right number of user interviews”, 2015, https://medium.com/@mitchelseaman/the-right-number-of-user-interviews-de11c7815d9
11. Renato Feijo, “Planning your UX strategy”, 2010, https://renatofeijo.com/blog-UXstrat.html
12. Judicaël Paquet, “Les 3 Amigos en agile”, 2019, https://blog.myagilepartner.fr/index.php/2019/02/06/les-3-amigos-agile/

Outils utilisés

Pour effectuer mon travail, j'ai utilisé :

- ☑ UserBit (https://userbitapp.com/) pour consigner ma recherche utilisateurs
- ☑ Miro (https://miro.com/) pour brainstormer, organiser les ateliers, maquetter ma solution
- ☑ Google Docs (https://docs.google.com/) et Google Sheets (https://sheets.google.com/) pour rédiger ce document
- ☑ Firefox (https://getfirefox.com/) pour accéder à tout cela
- ☑ Ubuntu Linux (https://ubuntu.com/download) pour accéder à Firefox

Annexes

Tryptique et guides d'entretiens

Développeurs backend

Faits	Hypothèses	Questions
Pas de définition officielle du métier de backend	Le périmètre de travail de chaque backend est variable d'une structure à l'autre	Dans le scope suivant, qu'est-ce qui vous incombe à votre poste actuel ?
Le choix des outils fait partie du métier	Les outils sont variables d'une structure/équipe/personne à l'autre	Quels outils utilisez-vous pour réaliser votre travail ?
Le dév travaille en méthode agile	Il prend part à la définition de ses tâches	Racontez-moi la vie d'une tâche qui vous arrive en développement ? D'où vient-elle, avec quelles spécifications vous travaillez, quelle marge de manœuvre et quelle est la DoD ?
Le dév travaille avec un UX designer	Il comprend le rôle de l'UX designer	Comment définiriez-vous le rôle d'UX designer ?
	Il interagit avec l'UX designer en phase de conception, ou plus tard	À quels moments interagissez-vous avec l'UX designer ? De quelle manière ?
	Il a vécu des situations frustrantes avec l'UX designer	Quels souvenirs avez-vous d'une interaction qui s'est bien passée ou mal passée avec l'UX designer ?
	Il n'y a pas de lien "solidaire" entre le travail d'UX et le travail de backend	Selon vous, comment l'UX designer pourrait vous aider dans votre travail ? Et inversement ?

UX designers

Faits	Hypothèses	Questions
Les UX designers choisissent les briques méthodo adaptées à leur projet	Le périmètre de travail de chaque UX est variable d'un projet à l'autre	Sur le projet avec le backend, quelles briques d'UX mettez-vous en oeuvre ?
Le choix des outils fait partie du métier	Les outils sont variables d'une structure/équipe/personne à l'autre	Quels outils utilisez-vous pour réaliser votre travail ?
Les dév travaillent en méthode agile	Le travail d'UX suit aussi une organisation agile et s'intègre dans le workflow des développeurs	Racontez-moi la vie d'une tâche qui part en développement ? D'où vient-elle, dans quelle mesure la spécifiez-vous, quelle marge de manœuvre pensez-vous laisser aux développeurs et comment déterminez-vous si la tâche est réalisée ou non ?
L'UX designer travaille avec des développeurs backend	Il comprend le rôle des backends	Comment définiriez-vous le métier de backend ?
	Il intègre les backends dans des ateliers de conception, et interagit avec eux plus tard	À quels moments interagissez-vous avec les backends ? De quelle manière ?
	Il a vécu des situations frustrantes avec les développeurs backend	Quels souvenirs avez-vous d'une interaction qui s'est bien passée ou mal passée avec les backends ?
	Il n'y a pas de lien "solidaire" entre le travail d'UX et le travail de backend	Selon vous, comment les backends pourraient vous aider dans votre travail ? Et inversement ?

Experience Map