

Dossier projet

**CONCEPTEUR DEVELOPPEUR
D'APPLICATIONS**

Torcheux Marion

Table des matières

Présentation	2
Contexte et objectifs	3
Spécificités des conceptions mobiles et web	3
Identification des utilisateurs cibles	4
Objectifs de l'application.....	4
Analyse des besoins utilisateurs	5
UX/UI : expérience utilisateur et interface utilisateur	9
Diagramme use case	9
Expérience utilisateur	10
Interface utilisateur	12
Conception de la base de données	14
Dictionnaire de données.....	15
MCD	16
Cahier des charges	17
Technologies utilisées et outils	20
Etapes du développement	25
Architecture de l'application	26
Déploiement	28
Evolutions et fonctionnalités prévues	29
Activités	30
Recherches	30
Veilles	30
Conclusion.....	31

Présentation

Je m'appelle Marion. Il y a deux ans, j'ai quitté mon emploi d'infographiste, que j'ai occupé pendant cinq ans. En occupant ce poste, j'étais souvent en relation avec les développeurs, qui m'expliquaient parfois leur travail. Cette proximité avec le développement a éveillé ma curiosité et mon intérêt pour le domaine. Fascinée par la possibilité de créer des solutions pratiques et innovantes, j'ai décidé de me former et de me reconverter dans le développement.

Mon parcours en infographie m'a permis de développer un certain sens de l'esthétique et de la convivialité, que j'applique désormais dans mes projets de développement. Je suis particulièrement passionnée par la création d'interfaces utilisateur intuitives et agréables, qui offrent une expérience utilisateur exceptionnelle.

À l'avenir, j'aspire à continuer à développer des applications innovantes qui répondent aux besoins des utilisateurs tout en intégrant des pratiques durables et écologiques. Mon objectif est de devenir une développeuse polyvalente et accomplie, capable de mener des projets de bout en bout, de la conception à la mise en production.

Mon projet actuel est une application mobile appelée RemindMe. Cette application est conçue pour améliorer la gestion des tâches et des rappels dans la vie quotidienne des utilisateurs. RemindMe permet aux utilisateurs de créer, organiser et suivre leurs tâches de manière simple et efficace, grâce à une interface utilisateur intuitive et visuellement agréable.

Ce projet est un aboutissement de mes compétences en développement mobile et en design d'interface utilisateur. Il reflète ma passion pour la création de solutions pratiques et esthétiques, tout en mettant en avant mon engagement à fournir des expériences utilisateur soignées.

À travers RemindMe, j'aspire à aider les utilisateurs à mieux gérer leur temps et leurs tâches, en leur offrant un outil puissant et agréable à utiliser. Ce projet marque une étape importante dans ma carrière de développeuse, et j'ai hâte de le faire évoluer en ajoutant de nouvelles fonctionnalités et en intégrant les retours des utilisateurs.

Contexte et objectifs

Spécificités des conceptions mobiles et web

J'avais déjà décidé de créer une application todolist, il me restait à choisir si elle serait web ou mobile.

Pour faire mon choix j'ai effectué quelques recherches sur les différences entre la conception mobile et la conception web :

- Les applications mobiles sont spécifiquement conçues pour être utilisées sur des appareils mobiles tels que les smartphones et les tablettes, tandis que les applications web sont conçues pour être utilisées dans un navigateur web sur un ordinateur ou un appareil mobile.
- L'espace d'affichage des applications mobiles est très réduit par rapport à l'application web : il faut bien choisir les informations à afficher pour ne pas encombrer l'écran et perdre l'utilisateur.
- La navigation sur un appareil mobile n'est pas la même que sur un écran d'ordinateur : il y a les gestes, les balayages et les touches.
- Les applications mobiles sont généralement plus rapides que les application web car elles sont installées sur l'appareil de l'utilisateur.
- Les applications mobiles peuvent utiliser les fonctionnalités natives de l'appareil comme la caméra, les capteurs de mouvement, les notifications push, ... pour ajouter des interactions avec l'utilisateur. Contrairement aux applications web qui n'ont pas accès à ces fonctionnalités.
- Pour corriger des bugs ou ajouter des fonctionnalités spécifiques, les applications mobiles nécessitent des mises à jour propres à chaque plateforme alors que les applications web peuvent être mises à jour instantanément pour tous les utilisateurs.
- Au niveau des coûts de développement, une application mobile peut être plus couteuse qu'une application web en raison de la nécessité de développer pour plusieurs plateformes.
- Pour ce qui est de la connectivité et du réseau, les mobiles peuvent être sujets à des qualités de réseaux variables, et pour pallier ça il existe des techniques pour accéder aux données hors connexion, avec le stockage local (local storage) par exemple, puis lorsque l'appareil est reconnecté à internet, il envoie les mises à jour.

En prenant en compte ces éléments, ainsi qu'en étudiant ma cible et ses habitudes, j'ai vite opté pour une application mobile : mon utilisateur doit avoir accès à ses listes n'importe où, n'importe quand. Une application type todolist est un bon format pour les écrans mobiles. Pour les coûts de développement, j'ai choisi d'utiliser Flutter comme Framework, qui permet de déployer sur Android et sur iOS avec un seul code.

Les applications mobiles sont devenues omniprésentes dans notre vie quotidienne, leur conception et leur modélisation jouent un rôle crucial dans leur succès. Une application mobile bien conçue peut offrir une expérience utilisateur exceptionnelle et améliorer la satisfaction de l'utilisateur tandis qu'une application mal conçue peut être frustrante et inutile.

Identification des utilisateurs cibles

Les femmes qui jonglent entre une carrière professionnelle et les responsabilités domestiques ont un emploi du temps chargé et complexe. Elles incarnent souvent le concept de la "femme moderne", capable de mener de front de multiples rôles et responsabilités avec détermination et efficacité.

Une femme engagée dans une carrière professionnelle investit beaucoup de temps et d'énergie dans son travail, qui parfois demande concentration, créativité et grand sens de l'organisation.

À cela s'ajoute la gestion des tâches domestiques : en plus de leurs responsabilités professionnelles, ces femmes assument souvent une part importante des tâches domestiques. Elles jonglent avec la préparation des repas, le ménage, la gestion des rendez-vous familiaux, et bien d'autres responsabilités liées à la vie quotidienne.

C'est là que l'application intervient : face à un emploi du temps chargé, ces femmes ont un besoin d'organisation et de productivité. Elles recherchent des outils et des solutions qui leur permettent de rester organisées, de gérer efficacement leur temps et de rester concentrées sur leurs objectifs, à la fois au travail et à la maison.

Cette application mobile va offrir aux femmes la possibilité de créer des listes de tâches ou de notes, d'être notifiées sur certaines tâches en fonction des dates et elles auront accès à leurs informations partout, à partir de leur mobile.

Il sera également possible de partager une liste de tâches avec un autre utilisateur.

Même si ma cible principale est bien définie, l'application pourra être utilisée par d'autres catégories de personnes, le design sera neutre et l'utilisation conviviale.

Objectifs de l'application

L'objectif principal est de faciliter la gestion des tâches : fournir aux utilisatrices un outil efficace et intuitif pour gérer leurs tâches quotidiennes. Cela inclut la création de listes de tâches, le suivi de l'état et des priorités des tâches, la gestion des dates d'échéance et des rappels, ainsi que la possibilité d'ajouter des fichiers attachés pour une organisation complète.

Améliorer la productivité : en offrant des fonctionnalités avancées telles que le partage des tâches en temps réel et l'analyse des performances, mon application

visé à aider les utilisatrices à optimiser leur productivité. L'objectif est de leur permettre de mieux gérer leur temps, de rester concentrées sur leurs propres objectifs et de réaliser leurs tâches de manière efficace.

Toutefois, d'autres objectifs sont visés en second plan :

Promouvoir l'organisation et la clarté mentale : en offrant une interface utilisateur intuitive et personnalisable, ainsi que des fonctionnalités avancées de filtrage et de suivi des tâches, mon application vise à aider les utilisatrices à rester organisées et à maintenir une clarté mentale. L'objectif est de réduire le stress et l'anxiété liés à la gestion des tâches et d'offrir aux utilisatrices un environnement propice à la concentration et à la productivité.

Encourager l'adoption de pratiques durables : en mettant en avant l'aspect écologique de mon application et en sensibilisant les utilisatrices à l'impact positif de leur utilisation sur l'environnement, je cherche à encourager l'adoption de pratiques durables. L'objectif est d'inspirer les utilisatrices à adopter un mode de vie plus respectueux de la planète et à contribuer à la préservation de l'environnement.

Précisions sur la dimension écologique

Même si c'est un impact mineur, j'ai réfléchi à l'aspect écologique de mon application : grâce à elle, on peut faire un petit pas vers un mode de vie plus durable, notamment par la réduction de l'utilisation de papier, ce qui préserve les ressources naturelles : cela signifie moins d'arbres abattus, moins d'eau utilisée dans le processus de fabrication du papier, et moins de déchets envoyés aux décharges.

Dans une future version de mon application, je pense mettre en avant sur un écran de chargement cet aspect écologique afin de fédérer les utilisatrices autour de la cause environnementale et de renforcer leur engagement en faveur de la durabilité. En affichant des messages inspirants ou des faits sur l'impact positif de leur utilisation de l'application sur l'environnement, je peux les encourager à adopter des pratiques plus écologiques au quotidien. Ce moment de réflexion, pendant le chargement de l'application, pourrait devenir un rappel régulier de l'importance de leur contribution individuelle à la préservation de notre planète, et ainsi les inciter à poursuivre leur engagement pour un avenir plus vert.

Analyse des besoins utilisateurs

J'ai utilisé plusieurs moyens pour réaliser mon analyse des besoins utilisateurs : j'ai demandé à mon entourage, j'ai effectué des recherches sur internet et j'ai également fait une analyse de la concurrence. Grâce à ça j'ai pu identifier les besoins communs des utilisatrices et les lacunes qui peuvent être comblées par mon application. Ces analyses approfondies m'ont permis de comprendre les attentes et les préférences des utilisatrices et de concevoir une application qui répond de manière précise à leurs besoins spécifiques.

Étude de l'entourage

Avant de me lancer plus loin dans la conception, j'ai demandé à certaines personnes de mon entourage quelles seraient leurs attentes vis-à-vis de cette application.

J'ai interrogé une dizaine de personnes, des femmes pour la plupart. Les hommes interrogés m'ont dit que ce serait une application très utile pour leur femme.

Globalement elles souhaitent pouvoir trier leurs tâches, et surtout pouvoir les cocher ou décocher pour que ce soit clair visuellement. Elles souhaitent aussi que la longueur de la tâche ne soit pas limitée en termes de texte. Le fait de pouvoir voir ce qui a été réalisé sur une période donnée est un plus, une satisfaction du travail accompli et une clarté dans les tâches qu'il reste à faire.

Les personnes interrogées m'ont rapporté qu'elles ont essayé plusieurs applications de listes, elles trouvent qu'elles sont parfois trop complexes, et pas attractives visuellement parlant.

Analyse de la concurrence

En étudiant attentivement les applications concurrentes telles que Remember The Milk, Any.do, Notion et Todoist, plusieurs fonctionnalités clés ont été identifiées. Ces applications offrent toutes des outils sophistiqués pour la gestion des tâches, avec des fonctionnalités telles que :

Les notifications : toutes les applications examinées intègrent des notifications pour rappeler aux utilisatrices les tâches à accomplir, garantissant ainsi qu'aucune échéance ne soit manquée.

Personnalisation des couleurs : La personnalisation des couleurs est également une caractéristique récurrente, permettant aux utilisatrices de personnaliser l'apparence de l'application en fonction de leurs préférences esthétiques.

Widget sur écran d'accueil : Les widgets sur l'écran d'accueil offrent un accès rapide aux tâches en cours et aux prochains événements, permettant aux utilisatrices de rester organisées sans avoir à ouvrir l'application principale.

Calendrier intégré : L'intégration d'un calendrier dans l'application permet aux utilisatrices de visualiser leurs tâches et leurs événements dans un format chronologique, facilitant ainsi la planification de leur emploi du temps.

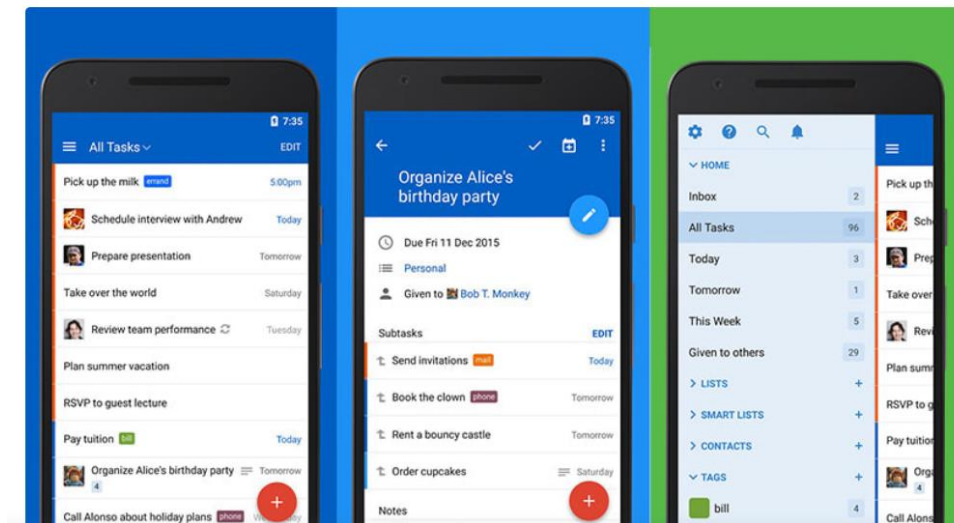
Suivi des tâches : Toutes les applications proposent un suivi des tâches, permettant aux utilisatrices de marquer les tâches comme complétées une fois qu'elles sont terminées.

Filtrage des tâches : La possibilité de filtrer les tâches en fonction de différents critères (par exemple, par date d'échéance, par étiquette ou par priorité) permet aux utilisatrices de se concentrer sur les tâches les plus pertinentes à un moment donné.

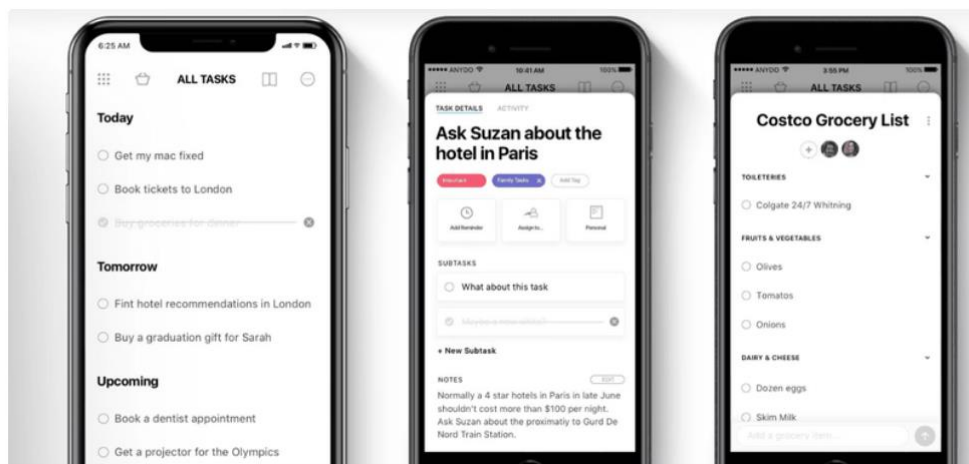
Priorisation des tâches : En attribuant des priorités aux tâches, les utilisatrices peuvent mettre en évidence les tâches les plus importantes ou urgentes, les aidant ainsi à hiérarchiser leurs activités.

J'ai pu remarquer que ces applications n'ont pas vraiment de design personnalisé, ce sont des listes avec des fonctionnalités dont l'interface devient vite compliquée visuellement.

Voici deux screens des applications retenues pour mon étude :



Screen étude de la concurrence : remember the milk



Screen étude de la concurrence : Any.do

Après l'étude de la concurrence, j'ai pu chercher des idées pour me différencier :

Il faudra bien travailler sur une interface intuitive et conviviale, simple et jolie. La fonctionnalité principale et de dresser des listes, mon utilisateur doit pouvoir le faire rapidement sans détour.

L'ajout de partage de liste de tâche avec d'autres utilisateurs est une fonctionnalité que n'ont pas toutes les autres applications.

La fonctionnalité d'historique prévue dans la base de données pourrait également me servir à faire des statistiques et de proposer un écran à mes utilisateurs sur leurs tâches : par exemple, combien de tâches ont été remplies cette semaine ?

Dans un premier temps je veux que mon utilisateur puisse changer la couleur principale de l'application, mais je souhaite aller plus loin dans la personnalisation dans une future version.

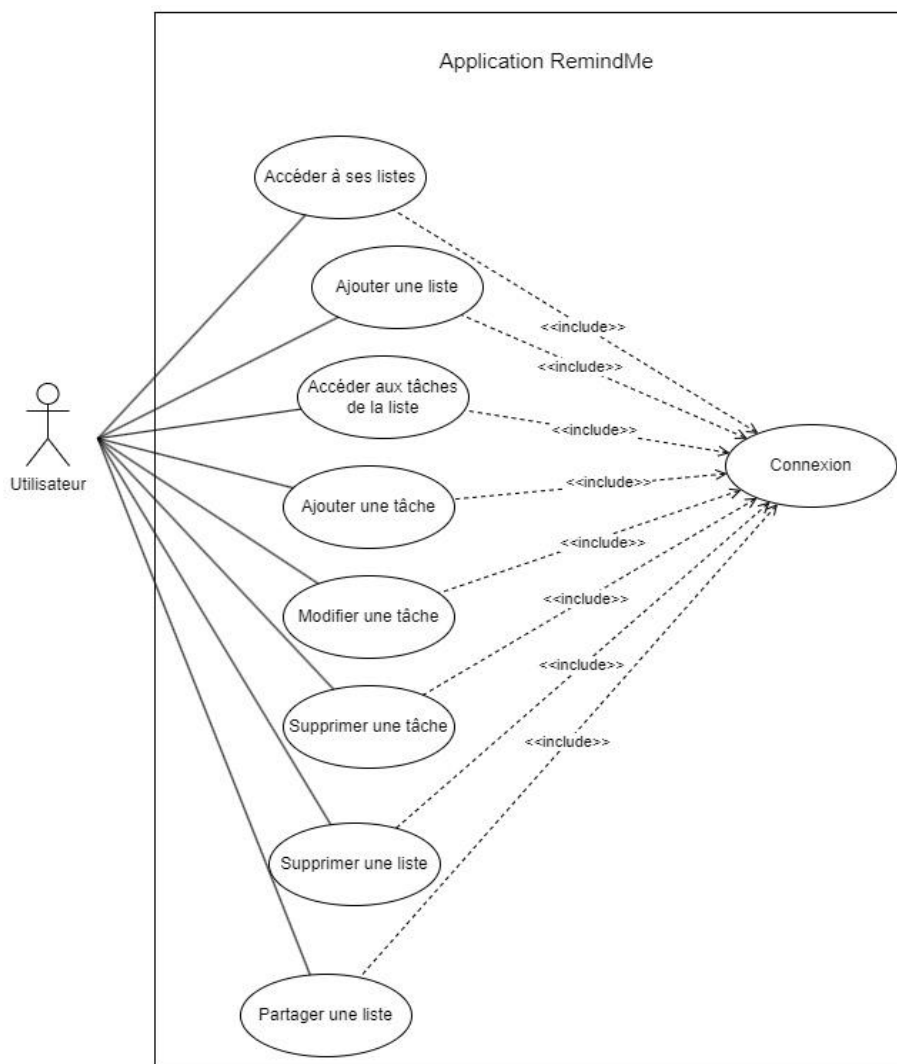
Ma démarche a pour but de démarquer mon application des autres et de répondre au mieux aux besoins des utilisateurs.

UX/UI : expérience utilisateur et interface utilisateur

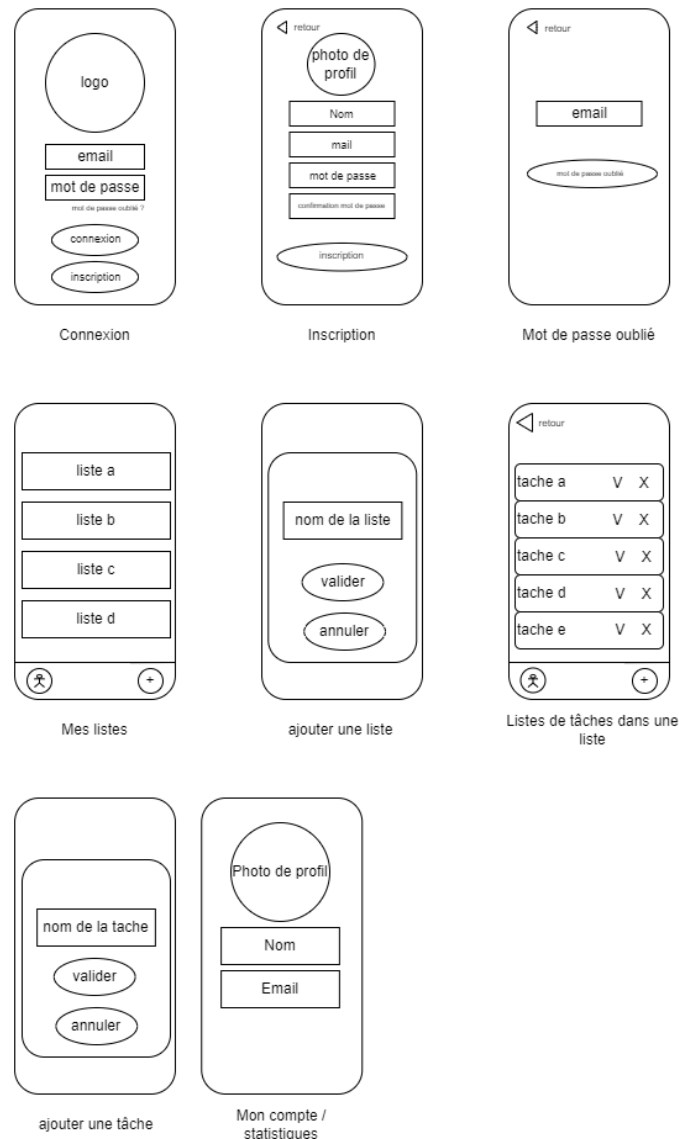
L'UI concerne la conception visuelle et interactive de l'interface utilisateur, tandis que l'UX englobe l'ensemble de l'expérience de l'utilisateur, y compris les aspects émotionnels, cognitifs et physiques de son interaction avec l'application ou le site web. Ensemble, l'UI et l'UX jouent un rôle essentiel dans la création d'une expérience utilisateur optimale et mémorable.

Diagramme use case

Ce diagramme de cas d'utilisation ou use case représente les fonctionnalités principales de mon application : l'interaction avec les listes et les tâches. Chaque cas d'utilisation est représenté par une ellipse, avec les actions spécifiques indiquées à l'intérieur.



Première ébauche : WireFrame



Expérience utilisateur

Étude des couleurs

Je ne savais pas par où commencer pour trouver une unité graphique qui pourrait correspondre à mon application. C'est donc en gardant en tête l'expérience utilisateur que j'ai effectué quelques recherches sur la psychologie des couleurs : il s'agit d'une étude sur la perception et l'impact des couleurs sur l'activité humaine.

J'ai recherché une couleur qui stimulerai la mémoire : le bleu.

Il est souvent associé à la stabilité, à la confiance et à la sérénité. C'est une couleur qui évoque un sentiment de calme et de clarté mentale, ce qui en fait un choix idéal pour une application axée sur la prise de notes et la gestion des tâches.

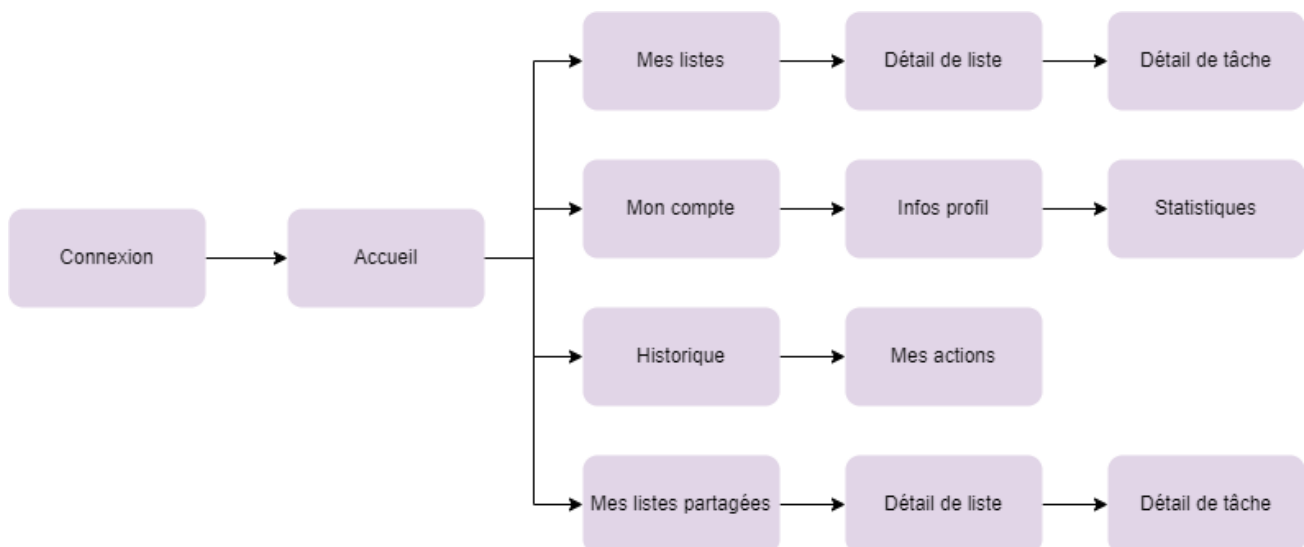
Le bleu est également connu pour favoriser la concentration et la réflexion, ce qui peut être bénéfique pour les utilisatrices qui cherchent à mémoriser des informations importantes ou à se concentrer sur leurs tâches.

En outre, des études ont montré que le bleu peut aider à réduire le stress et l'anxiété, ce qui est essentiel pour les femmes actives professionnellement qui jonglent avec de nombreuses responsabilités. En utilisant des nuances de bleu pour mon application, je peux créer un environnement visuel apaisant qui encourage la concentration, la mémorisation d'informations et la productivité.

Il est également important de noter que le choix des couleurs peut être subjectif et varier en fonction des préférences individuelles des utilisatrices. Par conséquent, j'ai décidé d'inclure des options de personnalisation pour leur permettre de choisir la palette de couleurs qui leur convient le mieux, tout en veillant à ce que le bleu soit présent comme option principale pour ses bénéfices psychologiques.

Arborescence : parcours utilisateur

L'arborescence permet d'avoir un aperçu du flux de navigation qui se doit d'être fluide et intuitif pour les utilisateurs et de créer une structure cohérente pour l'ensemble de l'application mobile.

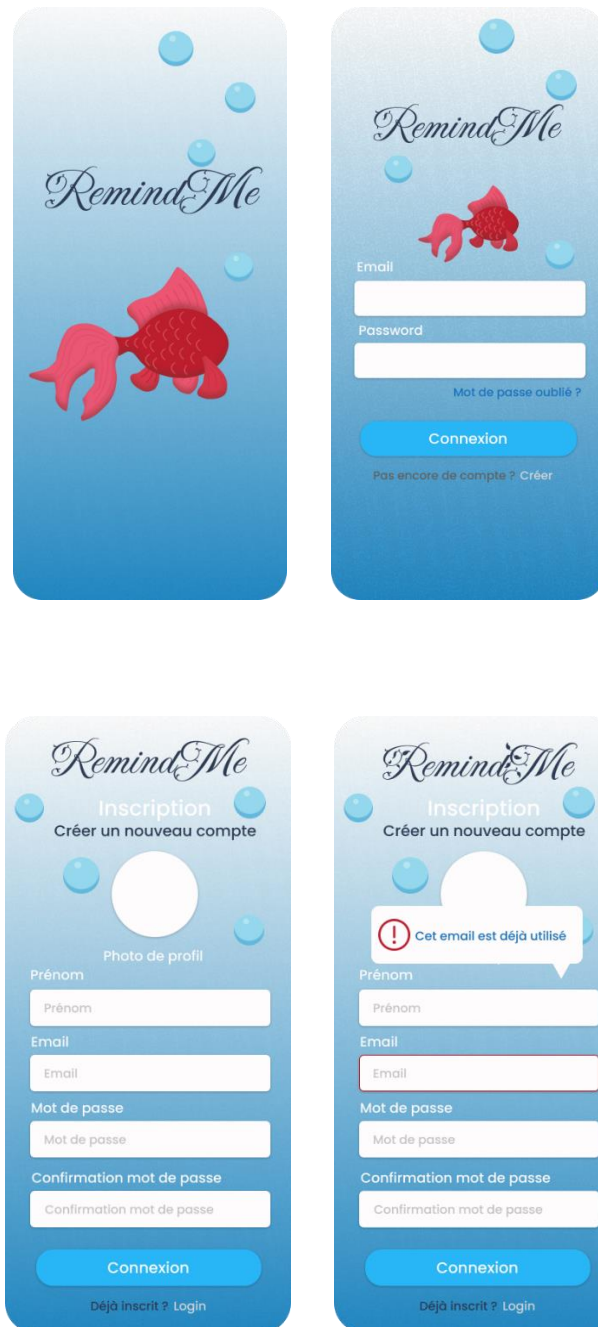


Schémas : arborescence de l'application

Interface utilisateur

L'interface utilisateur est la partie visible et interactive de l'application mobile, qui permet aux utilisateurs d'interagir avec les fonctionnalités et les informations. Elle vise à rendre l'application conviviale, intuitive, cohérente et attrayante tout en facilitant la communication et en offrant une expérience utilisateur optimale.

Voici en exemple quelques écrans que j'ai créé sur Figma :



Précisions sur l'utilité de créer les maquettes au début de la conception

Lorsqu'un projet est demandé par un client, les maquettes permettent de se projeter sur les fonctionnalités attendues, la disposition des éléments ou encore le design : on valide la conception.

Avec une maquette prototypée (sur Figma par exemple), on peut déjà tester la cohérence des concepts et réaligner le projet en cas d'erreur avant même la phase de développement.

Conception de la base de données

La méthode MERISE est utilisée dans la gestion de projet pour concevoir les systèmes d'information. A la base de cette méthode se trouve la création du dictionnaire de données, une liste exhaustive et organisée de toutes les entités.

Définitions des concepts de base de la méthode Merise :

Une entité représente un objet ou un concept du monde réel (exemple : voiture)

Les attributs décrivent les caractéristiques de ces entités (exemple : marque)

Les relations définissent les liens entre les entités (exemple : un utilisateur possède une voiture).

Les contraintes d'intégrité sont des règles qui garantissent la cohérence et la validité des données. Elles incluent notamment les clés primaires et les clés étrangères.

Les cardinalités spécifient le nombre d'instances d'une entité qui peuvent être associées à une autre (exemple : une voiture a un seul utilisateur mais un utilisateur peut avoir plusieurs voitures)

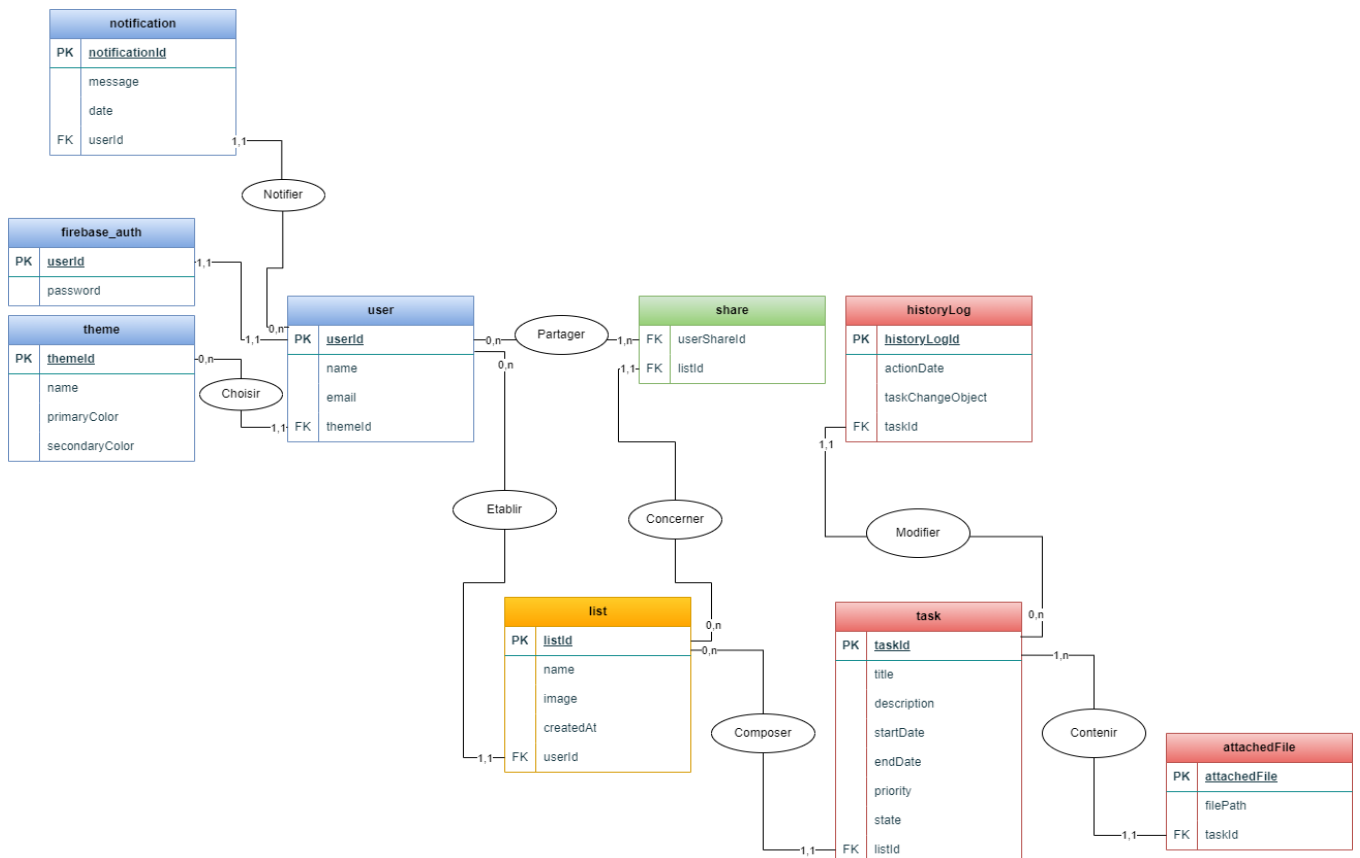
Dictionnaire de données

Afin de créer mon dictionnaire de données, j'ai pris mes maquettes et j'ai relevé chaque information à stocker.

Attribut	Type	Taille	Signification
name_user	String	255	Nom de l'utilisateur
email_user	String	255	Email de l'utilisateur
password_user	String	255	Mot de passe de l'utilisateur
name_list	String	255	Nom de la liste
imagePath_list	String	255	Chemin de l'image de la liste
createdAt_list	Date		Date de création de la liste
name_task	String	255	Nom de la tâche
description_task	String	255	Description de la tâche
startDate_task	Date		Date de départ de la tâche
endDate_task	Date		Date de fin de la tâche
priority_task	String	155	Niveau de priorité de la tâche
state_task	Boolean		Etat de la tâche
filePath_attachedFile	String	255	Chemin du fichier attaché à la tâche
actionDate_historyLog	Date		Date d'évènement
taskChangeObject_historyLog	String	255	
message_notification	String	255	Message de la notification
date_notification	Date		Date de la notification
name_theme	String	255	Nom du thème
primaryColor_theme	String	255	Couleur primaire du thème
secondaryColor_theme	String	255	Couleur secondaire du thème

MCD

Le Modèle Conceptuel de Données (MCD) de mon application de gestion de tâches est conçu pour offrir une structure organisée et efficace pour stocker et gérer les informations relatives aux tâches et aux utilisateurs. Le MCD comprend plusieurs entités clés, telles que "user", "list", "task". L'entité "user" contient des informations sur les utilisateurs de l'application, telles que leur nom, leur adresse email et leurs informations d'authentification. Chaque utilisateur peut créer plusieurs "list", qui servent à regrouper des tâches similaires ou liées. Chaque liste de tâches est associée à un utilisateur spécifique et peut contenir plusieurs "task". Les tâches sont des éléments individuels à accomplir, comprenant des informations telles que leur nom, leur description, leur état (en cours, terminé, en attente), leur priorité et leur date d'échéance. Ensemble, ces entités et leurs relations constituent une base solide pour la gestion efficace des tâches dans l'application.



Précisions sur l'entité firebase_auth

J'ai créé cette entité en plus de l'entité user pour représenter la séparation entre les responsabilités d'authentification et de la gestion des données utilisateur. L'entité firebase_auth est spécifiquement dédiée à la vérification des identités, les permissions et les accès.

L'entité user gère les informations spécifiques à l'application : prénom, mail.

Cahier des charges

Présentation du projet

Développement d'une application mobile à l'intention d'utilisateurs de smartphones, leur permettant de créer et de gérer des listes de tâches.

Compétences métier

Utilisateur

- L'utilisateur doit pouvoir créer un compte
- Pour créer une liste de tâches et des tâches l'utilisateur doit être inscrit et connecté
- L'utilisateur doit être inscrit et connecté pour consulter ses listes de tâches
- L'utilisateur doit être inscrit et connecté pour modifier ses listes de tâches et ses tâches

Listes et tâches

- Les listes et les tâches ne sont accessibles qu'à l'utilisateur inscrit et connecté et ne voit que les listes qui sont rattachées à son compte.
- Il doit être possible d'ajouter, de modifier ou de supprimer des listes de tâches ou des tâches.

Design

- L'interface doit être faite de façon à s'afficher correctement sur tous les appareils mobiles (smartphones)

Page d'accueil

- Cette page d'entrée affiche toutes les listes de tâches de l'utilisateur connecté
- Depuis cette page il peut ajouter une liste de tâches ou cliquer sur une liste existante pour la consulter.

Page Liste

- L'utilisateur connecté qui a cliqué sur une liste voit les tâches qui la compose sur ce nouvel écran. D'ici il peut ajouter une tâche à cette liste, cocher ou décocher une tâche, ou cliquer sur une tâche pour afficher son détail.

Page détail de tâche

- Lorsque l'utilisateur clique sur une tâche, il voit les détails de celle-ci sur un nouvel écran. D'ici il peut modifier les informations de cette tâche. Les informations sur cet écran sont les suivantes : titre de la liste à laquelle la tâche appartient, titre de la tâche, description, priorité, date de début, date de fin, et l'état de la tâche (si cochée ou décochée)

Fonctionnalités prévues

On pourra se connecter/s'inscrire/s'authentifier et une fois connecté on pourra créer des listes (avec une image, optionnelle) et à l'intérieur on a les tâches, les tâches auront un état (cochée ou pas) et des priorités (trois degrés de priorité : urgent, moyen, faible), les tâches peuvent aussi avoir un ou plusieurs fichiers attachés (image, doc, pdf). L'application prend en compte l'historique des actions sur les tâches. On peut partager une liste de tâches avec un autre utilisateur. Comme on peut mettre une date de fin sur une tâche on peut être notifié quand celle-ci est bientôt terminée. Et pour finir mon utilisateur pourra changer les couleurs du thème de l'application, mettre un fond d'écran si le design le permet. Voici un tableau récapitulatif :

Fonctionnalité	Description
Authentification	Permet aux utilisateurs de s'identifier dans l'application
Gestion des listes	Permet de créer, modifier, supprimer des listes
Gestion des tâches	Permet de créer, modifier, supprimer des tâches
Fichiers attachés à une tâche	Permet d'ajouter des fichiers liés à une tâche
Priorisation de tâche	Permet de prioriser la tâche selon trois degrés de priorité
Etat des tâches	Permet de gérer l'état de la tâche : fait, pas fait
Historique	Chaque action sur une liste ou une tâche est enregistrée dans la base de données et permet de rendre un historique à l'utilisateur
Partage de listes	Permet la collaboration sur une liste de tâches avec un autre utilisateur
Notification Push	Permet d'être notifié suivant certaines conditions
Changement de couleur du thème	Permet de changer la couleur de base de l'application
Accès aux données hors ligne	Permet de modifier ou lire les données hors connexion, mise à jour quand l'appareil retrouve la connexion

Livrables attendus

Pour la première version :

Back-end :

- Inscription/ connexion / mot de passe oublié
- Ajout, modification et suppression des listes de tâches ou des tâches
- Changer le statut d'une tâche

Front-end :

- Inscription / connexion / mot de passe oublié
- Affichage des listes de tâches
- Affichage des tâches qui composent la liste
- Cocher / décocher une tâche
- Ajout, affichage, modification et suppression des listes de tâches ou des tâches

Connexion

Pour pouvoir utiliser les fonctionnalités de l'application, l'utilisateur doit s'inscrire ou se connecter :

Pour l'inscription, les champs suivants sont requis :

- Email
 - Mot de passe (et une vérification sur le mot de passe)
 - Vérification du mot de passe
- Optionnel :*
- Ajouter une photo de profil

La connexion requiert les deux champs email et mot de passe.

Barre de navigation

La barre de navigation sera identique sur toutes les pages. Elle sera composée de plusieurs liens de navigation :

- Accueil (mes listes)
- Mes listes partagées
- Historique
- Profil

Technologies utilisées et outils

Flutter

Pour le développement de mon application, j'ai choisi Flutter.

C'est un framework de développement d'interfaces utilisateur créé par Google. Il permet de développer des applications mobiles à partir d'un seul code source, en utilisant le langage de programmation Dart. J'ai choisi de l'utiliser pour sa polyvalence et sa capacité à créer des interfaces utilisateur réactives et attrayantes. De nombreux Widgets sont déjà disponibles mais on peut aussi les personnaliser.

Avec Flutter, mon application vise un large public : mon code est compatible pour iOS et Android, ce qui fait que je n'ai qu'un code à maintenir (réduction des coûts et des efforts de développement).

Le hot reload de Flutter est une fonctionnalité très intéressante qui permet de voir les modifications de code de manière instantanée sur l'émulateur. Le processus de développement est donc plus rapide et plus agréable.

GetX

Même si de nombreux packages sont disponibles avec Flutter, je n'en utilise qu'un : GetX. Il sert à gérer l'état de l'application, plutôt que d'utiliser des StatefulWidget, on peut utiliser simplement des StatelessWidget qui sont moins gourmands en ressources. Il gère également la navigation, l'affichage de snackbars et le storage.

Voici votre variable:

```
var name = 'Jonatas Borges';
```



Pour la rendre observable, il vous suffit d'ajouter ".obs" à la fin:

```
var name = 'Jonatas Borges'.obs;
```



Et dans l'interface utilisateur, lorsque vous souhaitez afficher cette valeur et mettre à jour l'écran chaque fois qu'elle change, faites simplement:

```
Obx(() => Text("${controller.name}"));
```



Capture d'écran de la documentation de getX pour observer les variables

Si vous envisagez d'utiliser des routes/snackbars/dialogs/bottomsheets sans 'context', GetX est également excellent pour vous, voyez par vous-même:

Ajoutez "Get" avant votre MaterialApp, en le transformant en GetMaterialApp

```
GetMaterialApp( // Avant: MaterialApp(  
  home: MyHome(),  
)
```

Accédez à un nouvel écran:

```
Get.to(ÉcranSuivant());
```

Accédez au nouvel écran par le nom. Voir plus de détails sur les itinéraires nommés (named routes) [ici](#)

```
Get.toNamed('/details');
```

Pour fermer des snackbars, dialogs, bottomsheets, ou tout ce que vous auriez normalement fermé avec Navigator.pop(context);

```
Get.back();
```

Pour aller à l'écran suivant avec aucune option pour revenir à l'écran précédent (pour une utilisation dans SplashScreens, écrans de connexion, etc.)

```
Get.off(NextScreen());
```

Pour aller à l'écran suivant et annuler tous les itinéraires précédents (utile dans les paniers d'achat en ligne, les sondages et les tests)

```
Get.offAll(NextScreen());
```

Capture d'écran de la documentation de getX pour la navigation simplifiée

GetX simplifie grandement le développement et la clarté du code.

Je choisis de ne pas utiliser d'autres packages et dépendances pour garder le contrôle sur mon code.

Firebase

Firebase est une plateforme de développement d'applications mobiles de Google qui offre une multitude de services cloud, notamment l'authentification, la base de données en temps réel, le stockage de fichiers, les notifications push, etc. L'intégration de Firebase avec Flutter est très simple et offre une solution complète pour le backend de mon application, ce qui permet de gagner du temps et des ressources dans le développement.


Firebase va gérer ma base de données ainsi que l'inscription et l'authentification de mes utilisateurs. Il propose des fonctionnalités de sécurité robustes, telles que le stockage sécurisé des données et des règles de sécurité personnalisables.

Précisions sur les règles de sécurité personnalisables de Firebase

Firebase offre une robustesse en matière de sécurité grâce à ses règles de sécurité personnalisables, permettant de contrôler l'accès aux données stockées dans ma base de données. Ces règles de sécurité servent à protéger les données des utilisateurs et assurer leur confidentialité. Elles permettent de définir qui peut lire et écrire des données en fonction de divers critères, tels que l'authentification des utilisateurs, les propriétés des données et les rôles des utilisateurs. Firebase facilite également le test et la simulation de ces règles pour s'assurer qu'elles fonctionnent comme prévu avant de les déployer en production. En résumé, les règles de sécurité personnalisables de Firebase sont un outil puissant pour maintenir la sécurité et l'intégrité des données au sein d'une application. Voici les screens de mes recherches pour prendre en main ces règles de sécurité :

Libre accès

Lors de la configuration de Firestore, vous avez peut-être défini des règles autorisant le libre accès lors du développement. Vous pensez peut-être que vous êtes la seule personne à utiliser votre application, mais si vous l'avez déployée, elle est disponible sur Internet. Si vous n'authentifiez pas les utilisateurs et ne configurez pas les règles de sécurité, toute personne qui devine l'ID de votre projet peut voler, modifier ou supprimer les données.

 **Non recommandé:** accès en lecture et en écriture pour tous les utilisateurs.

```
// Allow read/write access to all users under any conditions
// Warning: **NEVER** use this rule set in production; it allows
// anyone to overwrite your entire database.

service cloud.firestore {
  match /databases/{database}/documents {
    match /{document=**} {
      allow read, write: if true;
    }
  }
}
```

Screen règles de sécurité de ma base de données : configuration par défaut

 **Solution:** règles limitant l'accès en lecture et en écriture.

Créez des règles adaptées à la hiérarchie de vos données. L'une des solutions courantes pour remédier à ce scénario non sécurisé consiste à appliquer une sécurité basée sur l'utilisateur à l'aide de Firebase Authentication. En savoir plus sur [l'authentification des utilisateurs à l'aide de règles](#).

```
Propriétaire de contenu uniquement    Accès public et privé mixte

service cloud.firestore {
  match /databases/{database}/documents {
    // Allow only authenticated content owners access
    match /some_collection/{document} {
      allow read, write: if request.auth != null && request.auth.uid == request.resource.data.autho
    }
  }
}
```

Screen règles de sécurité de ma base de données : résolution avec Firebase Authentication

Précisions sur Firebase Authentication

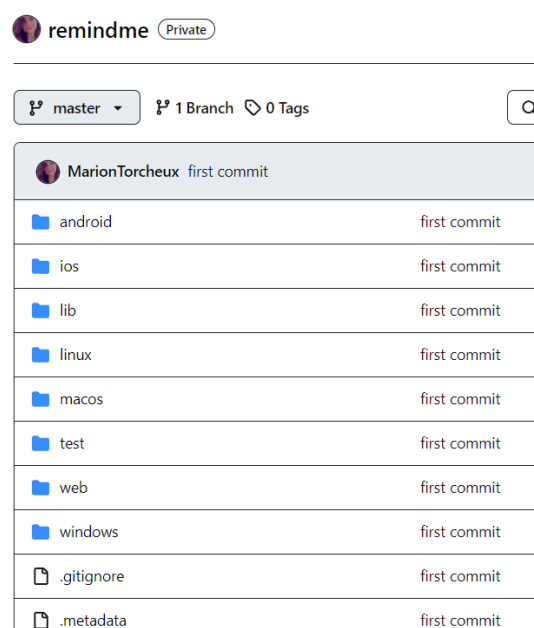
Pour ce qui est des mots de passe de mes utilisateurs, j'utilise un outil FireBase : Firebase Authentication. Lorsqu'un utilisateur crée un compte ou met à jour son mot de passe, Firebase Authentication utilise une fonction de hachage cryptographique pour transformer le mot de passe en une chaîne de caractères aléatoire de longueur fixe, puis elle est stockée dans la base de données. Les fonctions de hachage utilisées par l'outil sont conçues pour être cryptographiquement sécurisées, ce qui signifie qu'il est pratiquement impossible de retrouver le mot de passe d'origine à partir de la chaîne de caractère créée. Cela garantit que même si la base de données Firebase est compromise, les mots de passe des utilisateurs restent sécurisés.

Lorsqu'un utilisateur se connecte à son compte, Firebase Authentication utilise le même processus de hachage pour vérifier que le mot de passe saisi correspond à celui stocké dans la base de données. Si les deux hachés correspondent, l'utilisateur est authentifié avec succès.

La RGPD (règlement général sur la protection des données) stipule que même les administrateurs ne doivent pas avoir accès en clair aux mots de passe stockés dans la base de données. En utilisant FireBase Auth pour mon projet, cette loi est respectée.

GitHub

Afin de mieux gérer les versions du code, j'utilise gitHub. Le code supprimé ou remplacé n'est jamais perdu et il est toujours possible d'annuler une modification.



Screen de mon projet sur GitHub

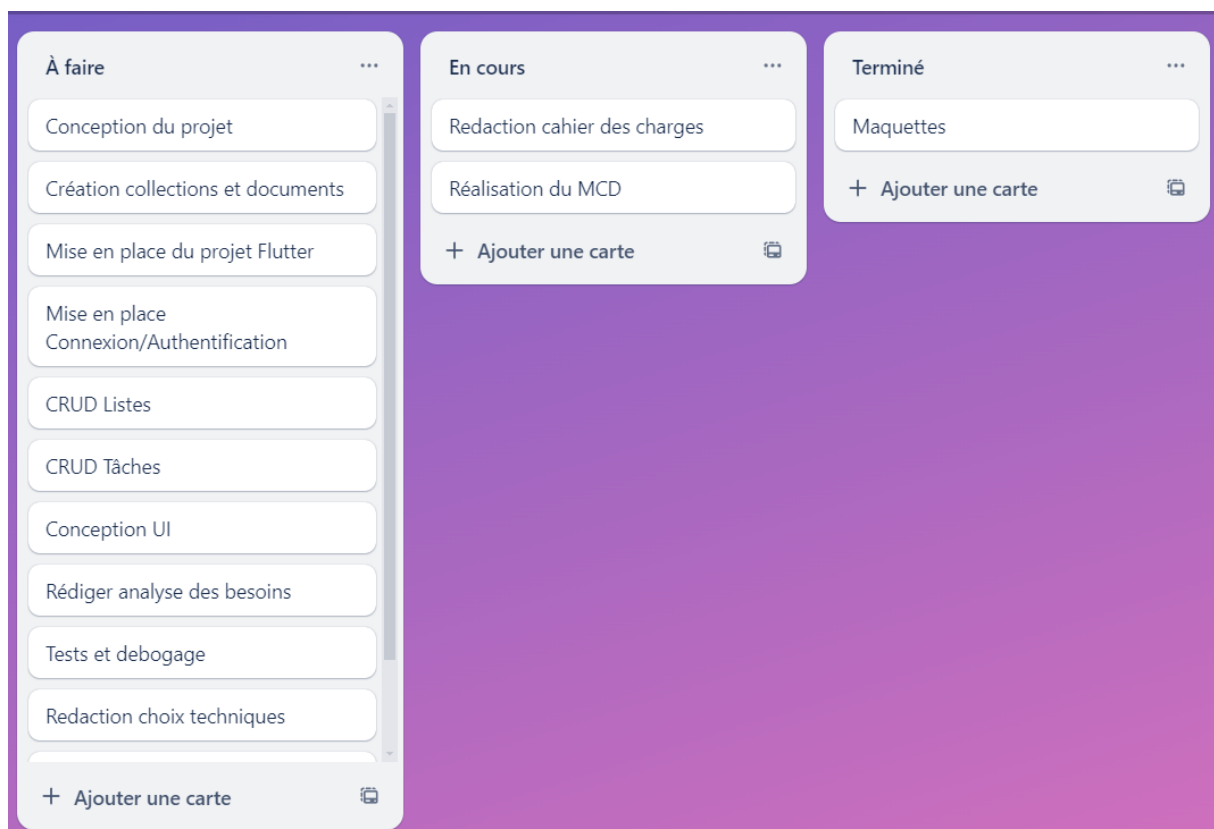
Sauvegarde supplémentaire

Pour assurer la sécurité et l'intégrité de mon travail de développement, j'ai pris la précaution de réaliser une sauvegarde complète sur un autre ordinateur non connecté à Internet. Cette mesure est cruciale pour protéger mes données contre les risques de pertes ou de corruption pouvant survenir en ligne, comme les attaques de logiciels malveillants, les ransomwares, ou les pannes de serveur. En conservant une copie de mon projet hors ligne, je m'assure d'avoir un accès fiable et sécurisé à mon travail, même en cas de problème avec mes systèmes principaux ou de perturbations de connexion internet.

Trello

Afin d'organiser le développement de mon projet, j'utilise l'outil de gestion de projet en ligne Trello. Il me permet de lister les tâches à réaliser, de voir les tâches en cours ainsi que les tâches terminées.

J'ai découpé mon projet en petites tâches, ce qui permet d'en valider régulièrement et de voir l'avancement et le bon déroulement du projet.



Screen Trello de mon projet

Etapes du développement

Configuration de l'environnement de développement : j'ai installé Flutter sur ma machine, et j'ai utilisé l'IDE IntelliJ pour développer. Dans un terminal j'ai utilisé la commande `Flutter create +nom du projet` pour créer le projet, puis une fois les dossiers en place, j'ai renseigné mes dépendances dans le fichier `pubspec.yml` (dépendances Firebase et GetX).

Très tôt lors du lancement du projet, j'ai créé un repo sur gitHub pour gérer les versions de mon application et pour sauvegarder mon travail de manière sécurisée.

Le projet lancé, j'ai commencé par implémenter l'authentification avec Firebase Auth. Mes utilisateurs peuvent s'inscrire, se connecter et récupérer leur mot de passe en cas de perte.

Ensuite, la gestion des tâches : ce sont les fonctionnalités principales de l'application. Cela comprend la création, la modification, la suppression et la consultation de listes de tâches, ainsi que l'attribution d'états et de priorités aux tâches individuelles.

Base de données : j'ai configuré mes collections et mes documents Firestore pour stocker les listes de tâches et les détails des tâches.

Développement de l'interface utilisateur : en suivant ma maquette, j'ai implémenté mon interface utilisateur avec les widgets Flutter, ainsi que la navigation entre les pages en utilisant GetX

Tests et débogage : à chaque nouvelle fonctionnalité développée, j'ai mené à bien des tests pour éprouver mon code.

Exemple de tests fonctionnels : créer un compte

Scénario	Résultat et réponse
L'utilisateur ne remplit aucun des champs et clique sur inscription	Tous les champs ont une erreur personnalisée pour chaque cas
L'utilisateur renseigne une adresse email non valide	Le champ est en erreur « veuillez entrer une adresse valide »
L'utilisateur ne rentre pas le même mot de passe dans la vérification	Erreur : « les mots de passe ne correspondent pas, conservation des champs
L'utilisateur entre un email déjà utilisé	SnackBar d'erreur email déjà utilisé, conservation des champs
L'utilisateur remplit tous les champs correctement	Inscription réussie retour à l'écran de connexion, snackbar de confirmation

Architecture de l'application

Mon application est développée en respectant les principes de l'architecture MVC, modèle, vue, contrôleur. Chacun de ces composants exerce des responsabilités distinctes :

Le modèle représente la structure des données de l'application ainsi que les règles métier qui régissent leur manipulation. Il gère la logique métier, les opérations de lecture et d'écriture des données, ainsi que les interactions avec la base de données.

La vue est responsable de l'affichage des données au sein de l'interface utilisateur de l'application. Elle se concentre sur la présentation des informations au travers d'éléments visuels. Dans ce composant il n'y a pas de traitement, juste de l'affichage.

Les informations sont fournies à la vue par le contrôleur. Celui-ci agit comme un intermédiaire entre le modèle et la vue. Il gère les requêtes de l'utilisateur, traite les entrées, met à jour le modèle en conséquence et sélectionne la vue appropriée pour afficher les résultats sur l'interface utilisateur. Il contient la logique de l'application et coordonne les actions entre le modèle et la vue.

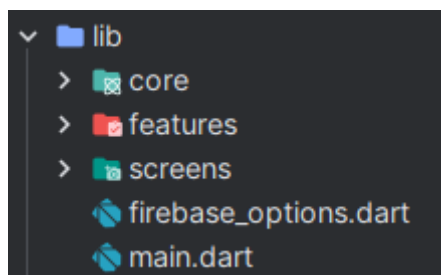
Dans mon arborescence, j'ai un dossier par vue, dans ce dossier il y a également un contrôleur, qui n'appartient qu'à la vue. Lorsque j'ai des méthodes utiles à plusieurs vues, elles sont placées dans le contrôleur Mixin à la racine, pour ne pas avoir à les écrire à plusieurs endroits.

Cette architecture permet de faciliter la maintenance, la réutilisation du code, tout en permettant une collaboration efficace si une éventuelle autre personne rejoint le projet.

Les dossiers et fichiers sont également nommés en suivant une logique, afin de mieux se repérer dans le projet : le dossier d'une vue est nommé comme la vue, le fichier de la vue est nommé comme son dossier avec `_screen` à la fin, le fichier controller de la vue est également nommé comme la vue avec `_controller` à la fin.

Un widget personnalisé sera nommé `custom_` en préfixe, et les widgets sont nommés comme la vue avec un suffixe représentant ce qu'ils sont (ex : `_card`)

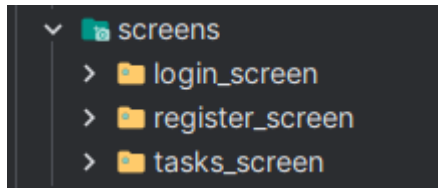
Voici des captures d'écran de mon arborescence de dossiers :



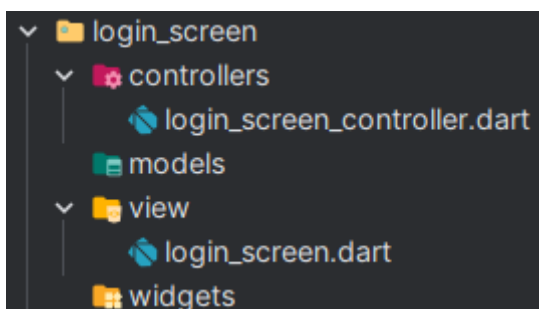
La racine :

- core contient le cœur de mon application : les classes, les models, les routes.

- features contient tous les widgets que j'ai personnalisé. Ils sont accessibles à toutes les vues.
- Le dossier screens contient tous les écrans de mon application,



Et dans chaque dossier de screen on retrouve un dossier controllers, associé à la vue, un dossier models si un model est associé uniquement à cette vue, un dossier view pour la vue, et un dossier widget si j'ai un widget qui ne sera utilisé que dans cette vue.



Déploiement

Mon application sera déployable sur App Store d'Apple et le Play Store de Google : ces deux plateformes offrent une visibilité mondiale et sont les principaux points de distribution pour les applications mobiles IOS et Android.

Comparaison des deux marketplaces cible :

	<i>App Store</i>	<i>Play Store</i>
<i>Propriétaire</i>	Apple	Google
<i>Système d'exploitation</i>	IOS	Android
<i>Disponibilité</i>	64	56
<i>Nombre d'applications</i>	Disponible dans plus de 175 pays	Disponible dans plus de 190 pays
<i>Processus de soumission</i>	Environ 2,2 millions	Environ 3,5 millions
<i>Modèle économique</i>	Principalement des applications payantes, avec des applications gratuites avec achats intégrés	Un plus grand nombre d'applications gratuites mais avec publicités et achats intégrés
<i>Sécurité</i>	Réputation de sécurité élevé	Mesures de sécurité mises en place mais plus vulnérables aux applications malveillantes
<i>Audience cible</i>	Principalement des utilisateurs d'appareil IOS (Iphone, Ipad, Mac)	Utilisateurs d'appareils Android (smartphone, tablettes)
<i>Taux de commission</i>	30% pour les ventes et les achats intégrés	30% pour les ventes d'applications et les achats intégrés (15% pour les abonnements après la 1 ^{ère} année)
<i>Règles de publication</i>	Règles strictes sur la conception, l'expérience utilisateur et la sécurité des applications	Règles moins strictes avec un accent sur la sécurité et la qualité de l'application
<i>Processus de mise à jour</i>	L'app Store examine les mises à jour de l'application proposée. Délais de quelques jours à quelques semaines	Le Play Store met à jour automatiquement après approbation des politiques, ce qui peut prendre quelques heures.

Bien que les deux plateformes offrent un vaste public potentiel, il est important de prendre en compte les différences dans le processus de soumission, les frais, les

langages de programmation pris en charge, la part de marché et la communauté de développeurs lors du choix de la plateforme de déploiement. En choisissant de distribuer notre application sur les deux plateformes, je vise à atteindre un public aussi large que possible et à répondre aux besoins des utilisateurs sur les appareils iOS et Android.

Evolutions et fonctionnalités prévues

L'objectif a toujours été de créer un outil simple, intuitif et efficace pour aider les utilisateurs à organiser leurs tâches quotidiennes. À mesure que l'application a évolué, les retours des utilisateurs ont joué un rôle crucial dans l'orientation de son développement. Grâce à ces feedbacks précieux, j'ai pu identifier les besoins réels et les attentes des utilisateurs, permettant ainsi d'envisager des améliorations pertinentes et des fonctionnalités innovantes pour les prochaines versions.

J'ai divisé ces fonctionnalités en deux catégories : celles qui ont effectivement été demandées par les utilisateurs, et celles que j'ai imaginées par curiosité technique.

En intégrant ces nouvelles fonctionnalités, mon but est de rendre l'application encore plus utile et agréable à utiliser, tout en respectant les normes de protection des données et en garantissant une expérience utilisateur harmonieuse et cohérente.

1. **Droits des utilisateurs** : Pour respecter les droits des utilisateurs en matière de protection des données, dans la section « Mon compte », on pourra modifier les informations du compte et de supprimer des données personnelles facilement.
2. **Refonte des icônes** : Je vais redessiner toutes les icônes moi-même afin de maintenir une cohérence et une unité graphique dans toute l'application, renforçant ainsi l'identité visuelle.
3. **Accès et écriture de données hors connexion** : Pour améliorer l'expérience utilisateur, l'application permettra l'accès et la modification des données même sans connexion Internet, assurant une utilisation continue et sans interruption.
4. **Accès partiel sans compte** : Certaines fonctionnalités de l'application seront accessibles même sans création de compte, facilitant ainsi l'adoption initiale et l'accessibilité de l'application pour de nouveaux utilisateurs.
5. **Pouvoir dessiner une note** : Cette fonctionnalité, ajoutée par curiosité technique, permettra aux utilisateurs de dessiner des notes, offrant ainsi une flexibilité et une créativité accrues dans la prise de notes.
6. **Suppression d'une tâche en secouant le téléphone** : également inspirée par l'aspect technique, cette fonctionnalité innovante permettra de

supprimer une tâche en secouant le téléphone, rendant l'interaction avec l'application plus intuitive et amusante.

Ces nouvelles fonctionnalités et améliorations visent à enrichir l'expérience utilisateur tout en explorant de nouvelles possibilités techniques.

Activités

Recherches

Effectuer des recherches est une compétence essentielle pour tout développeur, surtout lorsqu'il s'agit de déboguer du code. La capacité à rechercher des informations efficacement est cruciale pour résoudre les erreurs et les bugs qui surviennent inévitablement lors du développement. Utiliser les bons mots-clés pour trouver des solutions précises et pertinentes sur des forums, des documentations officielles, et des plateformes de questions-réponses comme Stack Overflow, peut faire toute la différence. En tant que développeuse junior, j'ai souvent dû effectuer de nombreuses recherches pour comprendre et corriger les erreurs dans mon code. Cette pratique m'a permis de non seulement résoudre les problèmes plus rapidement, mais aussi d'améliorer ma compréhension des technologies et des outils que j'utilise. En affinant mes compétences en recherche, j'ai appris à poser les bonnes questions et à utiliser des sources fiables, ce qui m'a aidé à surmonter des défis techniques complexes et à améliorer la qualité de mes développements. Rechercher efficacement pour déboguer est donc non seulement une compétence technique, mais aussi une démonstration de persévérance et d'ingéniosité, essentielles dans le métier de développeur.

Veilles

Faire une veille sur le développement est crucial pour rester compétitif et pertinent dans le domaine en constante évolution des technologies de l'information. La veille technologique permet de suivre les dernières tendances, innovations et meilleures pratiques, assurant ainsi que les compétences et les connaissances restent à jour. En surveillant les évolutions des langages de programmation, des frameworks, des outils et des méthodologies, on peut adopter les solutions les plus efficaces et les plus performantes pour leurs projets. La veille permet également d'anticiper les changements du marché et de s'adapter rapidement aux nouvelles exigences, réduisant ainsi les risques de retard technologique. Elle favorise également l'amélioration continue, l'innovation et la créativité en exposant les professionnels à de nouvelles idées et perspectives. Faire une veille régulière est essentiel pour maintenir une expertise de pointe, optimiser les processus de développement et offrir des produits de haute qualité répondant aux besoins et aux attentes des utilisateurs.

Pour maintenir mes compétences à jour et rester informée des dernières tendances en matière de développement, j'adopte une approche proactive de veille

technologique. Je consulte régulièrement et suis des pages spécialisées sur LinkedIn, où des experts et des professionnels partagent des articles et des nouvelles concernant les meilleures pratiques et les innovations dans le domaine du développement. En lisant les commentaires et les discussions qui suivent ces publications, je peux comprendre les perspectives et les expériences d'autres développeurs, ce qui enrichit ma propre pratique. De plus, j'utilise l'application DailyDev, une plateforme dédiée qui regroupe des articles et des nouvelles exclusivement sur le développement, me permettant d'accéder facilement à une large gamme de contenus pertinents et actualisés.

Enfin, je participe activement à des réunions avec d'autres développeurs, où nous échangeons sur des sujets techniques, discutons des défis rencontrés et partageons des solutions innovantes. Ces interactions me permettent non seulement de rester informée, mais aussi de bénéficier des expériences et des conseils de mes pairs, favorisant ainsi une amélioration continue de mes compétences et de mes pratiques professionnelles.

Conclusion

Lors de mes anciennes expériences en matière de développement, j'ai surtout participé ou mené à bien des projets web. Ce projet mobile, que j'ai conçu entièrement seule (tout en faisant valider chaque étape par plusieurs développeurs plus expérimentés que moi), m'a posé de nombreux défis. Par exemple, je me suis entêtée à commencer par la conception de ma base de données, comme pour un projet web, ce qui m'a fait perdre beaucoup de temps, car je ne voulais pas commencer par les maquettes. C'est en suivant les conseils de mes professeurs que j'ai réalisé que la conception pour un projet mobile ne suit pas les mêmes règles.

À partir de là, j'ai pu progresser et créer quelque chose de viable, même si ce n'est pas parfait. Cette première approche m'a permis d'acquérir de nouvelles compétences et une meilleure compréhension des spécificités du développement mobile. J'ai également appris l'importance de rester flexible et ouverte aux conseils, ainsi que de bien planifier les étapes d'un projet en fonction de son contexte spécifique.

Ce projet ouvre la porte à d'autres projets bien plus grands et plus complexes. Forte de cette expérience, je me sens mieux préparée à relever de nouveaux défis dans le développement d'applications mobiles. Mon objectif est de continuer à apprendre et à me perfectionner, tout en cherchant à créer des solutions innovantes et efficaces qui répondent aux besoins des utilisateurs. Cette aventure m'a également montré l'importance de la persévérance et de l'apprentissage continu, des qualités essentielles pour réussir dans le domaine du développement.