



420-3A4-SO

Développement d'une application mobile 1

Titre du travail : TP #2 - NutriLog

Durée : 3 semaines

Vous êtes chargé(e) de concevoir et de développer une application mobile en utilisant SwiftUI permettant aux utilisateurs de suivre leur consommation quotidienne de calories et de nutriments. L'application devra offrir une interface simple et intuitive pour ajouter, visualiser et analyser les repas consommés au fil du temps. Une section graphique permettra d'afficher l'évolution des apports journaliers et de comparer les résultats avec les objectifs fixés.

Objectifs principaux du travail :

- Structurer un projet avec les bonnes pratiques SwiftUI et MVVM :** Concevoir une application claire et bien organisée en séparant la logique, les données et l'interface.
- Afficher et persister les données avec SwiftData :** Enregistrer les repas et les calories consommées localement et afficher les informations dans une interface réactive.
- Visualiser les statistiques avec Charts :** Présenter l'évolution des apports caloriques à l'aide de graphiques simples et lisibles.
- Intégrer Face ID pour sécuriser l'accès à l'application :** Permettre à l'utilisateur de déverrouiller ou protéger ses données grâce à l'authentification biométrique.

Spécifications de l'application

L'application devra contenir les éléments suivants :

1. Une vue « **Connexion** » qui agira comme la page d'accueil de l'application pour permettre à l'utilisateur de se connecter à l'aide de Face ID.
2. Une vue « **Journée** » qui affichera les informations des calories, des macros et des repas pour la journée.
3. Une vue « **Ajouter un repas** » qui permettra d'ajouter un repas pour la journée.
4. Une vue « **Détail d'un aliment** » qui affichera un détail pour le repas sélectionné.
5. Une vue « **Graphiques** » qui affichera les données disponibles sous forme de graphiques.
6. Une barre d'onglets dans le bas de l'application permettant la navigation entre les différentes vues.

À noter que pour simplifier le travail, vous pouvez assumer que tous les repas entrés sont pour la journée actuelle.

Les visuels sont fournis pour vous donner une idée des différentes interfaces. Toutefois, les visuels ne sont pas parfaits et peuvent avoir des petits défauts (couleur primaire, couleur secondaire, couleur d'accent, espacement, texte multi-lignes, etc.) qu'il faudra prendre le temps de corriger.

Spécifications techniques

1. Utiliser iOS 18.5 comme version minimale de déploiement;
2. Implémenter la localisation de l'interface (français et anglais);
3. Utiliser les données fictives pour concevoir les interfaces;
4. Utiliser SwiftData pour la persistance des données;
5. Utiliser une couleur d'accent autre que celle par défaut;
6. Appliquer une bonne ergonomie au niveau du UI/UX;
7. Respecter les principes du design mobile et de navigation de Apple;
8. Séparer et réutiliser le code autant que possible;
9. Aucune librairie n'est permise.

Si vous effectuez des changements majeurs à l'application au niveau des interfaces et du fonctionnement demandé, vous devez faire approuver les changements par l'enseignant.

La vue « Connexion »

Cette vue représentera la page de connexion de l'application. Elle permettra à l'utilisateur de se connecter à l'aide de Face ID.

La vue doit contenir :

1. Le logo de l'application;
2. Le fond de l'écran devra avoir la couleur #FBF9F3;
3. La couleur du bouton est #F59E0B;
4. Un bouton permettant de se connecter en utilisant Face ID.

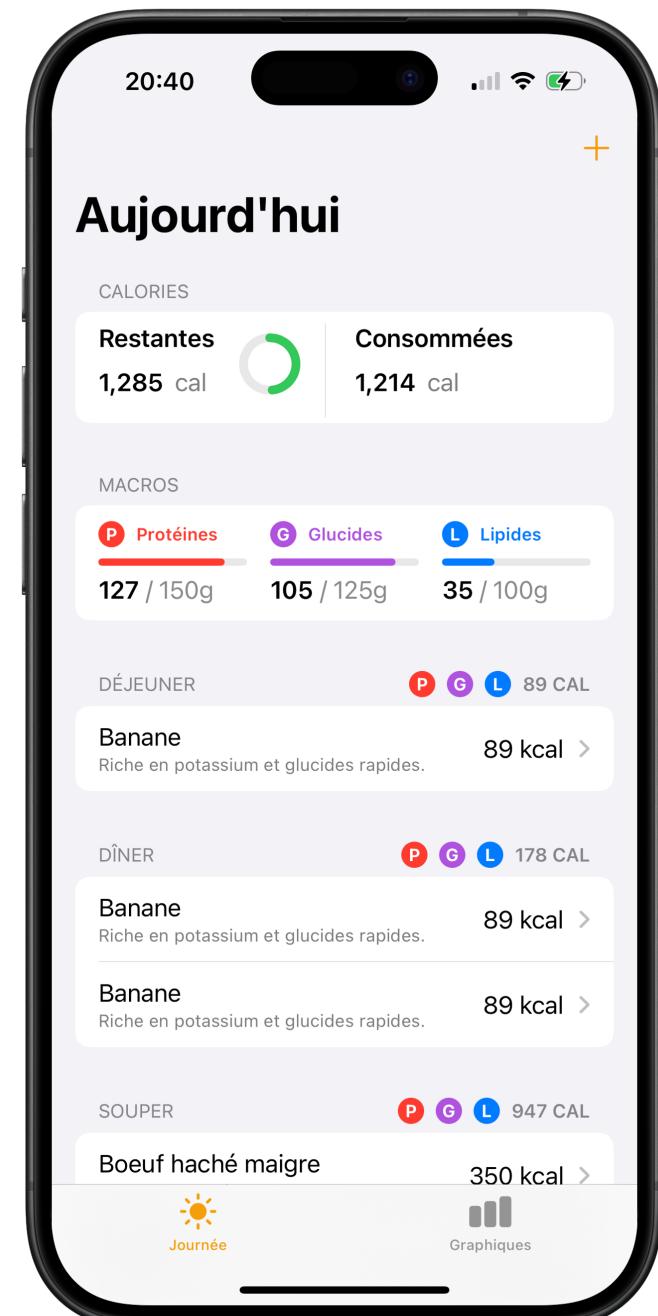


La vue « Journée »

Cette vue est la vue qui sera affichée après la connexion et présentera les calories, les macros et les aliments par repas de la journée.

La vue doit contenir :

1. Le titre de la vue;
2. Une section pour le nombre de calories consommé et le nombre de calories restant à consommer;
3. Une section pour les macros (protéines, glucides, lipides) selon les repas entrés par l'utilisateur. Pour les valeurs quotidiennes, vous utiliserez 2500 pour les calories, 150 pour les protéines, 125 pour les glucides et 100 pour les lipides;
4. Les différents aliments par repas (Déjeuner, Dîner, Souper);
5. Chaque section de repas devra afficher son type de repas, des icônes pour indiquer si les repas avaient les différents macros et le nombre total de calories pour ce repas.
6. Chaque aliment affiché devra montrer son nom, sa description et son nombre de calories.
7. Chaque aliment devra amener sur la page détail de l'aliment.
8. Une barre d'outil avec un bouton permettant d'ouvrir la vue pour ajouter un repas à l'aide un modal.

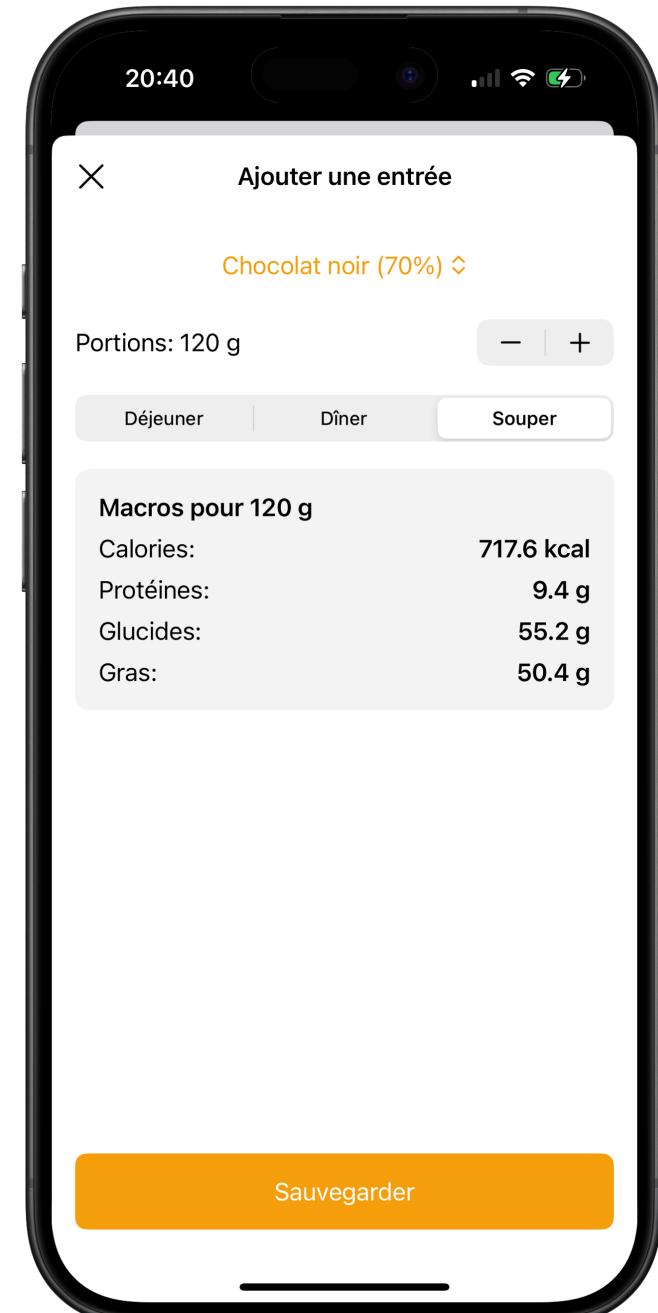


La vue « Ajouter un repas »

Cette vue affichera un formulaire permettant d'ajouter un aliment en choisissant sa portion et son repas associé.

La vue doit contenir :

1. Le titre de la vue;
2. Une option permettant de choisir l'aliment parmi toutes les données fictives des aliments;
3. Une option permettant de choisir les portions de l'aliment (en grammes);
4. Un affichage des macros de l'aliment choisi en fonction des portions (l'affichage est dynamique lorsque nous changeons les portions, les valeurs se mettent automatiquement à jour);
5. Un bouton « Sauvegarder » permettant de sauvegarder les données avec SwiftData et qui fermera automatiquement le modal. Idéalement, le bouton devrait être désactivé lorsqu'aucun aliment n'a été encore choisi;
6. Un bouton permettant de fermer le modal.



La vue « Détail d'un aliment »

Cette vue affichera le détail d'un aliment (calories et macro) en plus de montrer une liste de tous les repas avec la date où l'aliment a été consommé.

La vue doit contenir :

1. Le titre de la vue;
2. Un bouton permettant de revenir à la vue précédente;
3. Les calories et les macros pour l'aliment;
4. La liste montrant l'historique de consommation (date et type de repas où l'aliment a été consommé).



La vue « Graphiques »

Cette vue affichera les différentes données sous forme de graphiques. Le choix des graphiques et du design de cette vue est laissé à votre discrétion.

Vous devez donc :

1. Afficher trois (3) graphiques que vous jugez pertinents à montrer à l'utilisateur;
2. C'est à vous de déterminer les données que vous voulez afficher et le bon type de graphique à utiliser;
3. Le design est de votre responsabilité et vous devez respecter les bonnes pratiques d'ergonomie Apple.

L'interface fournit comme exemple est incorrect!



Évaluation

Ce travail compte pour 15% de la note finale et sera évalué sur les critères suivants :

1. **Conception de l'application** : Qualité du design UI/UX, respect des bonnes pratiques de navigation mobile.
2. **Fonctionnalité** : Implémentation correcte des fonctionnalités requises, performance et fluidité de l'application.
3. **Qualité du code** : Lisibilité, modularité, et utilisation des bonnes pratiques de programmation.

Si l'un des critères relatifs à la **Conception de l'application** ou à la **Fonctionnalité** est considéré comme insuffisant, cela pourrait entraîner une pénalité significative, voire une note de zéro pour la section relative à la **Qualité du code**.

Remise

La **date de remise du travail sera affichée sur Léa**. La remise se fera par un fichier compressé ZIP de l'ensemble de votre code source et par Github Classroom.

Il suffit simplement de pousser tous vos changements dans le dépôt Git qui sera créé lors de l'utilisation du lien Github Classroom pour ce travail (*tout commit effectué après la date de remise du travail sera considéré comme un retard*).