HES-SO MASTER

EthiScan - L'App pour les Consomm'acteurs

Vendredi, 14 Mai 2024

Professeur: Pascal Bruegger & Aïcha Rizzotti

Olivier D'Ancona, Clarisse Fleurimont, Yannis Chamot

Contents

Abstract/Résumé

EthiScan est une application mobile destinée à transformer l'expérience de consommation en permettant aux utilisateurs de scanner des produits pour obtenir des informations détaillées alignées avec leurs valeurs personnelles. Elle vise à promouvoir une consommation responsable en fournissant des données sur les labels environnementaux et nutritionnels, l'évolution des prix, l'impact carbone, et d'autres critères pertinents. En intégrant une technologie de pointe et une conception centrée sur l'utilisateur, EthiScan offre une plateforme fiable et intuitive pour faire des choix de consommation éclairés et responsables.

1 Introduction

1.1 Technologie Utilisée

EthiScan utilise une stack technologique moderne et efficace pour offrir une expérience utilisateur optimale. Le front-end de l'application est développé avec Flutter, permettant une interface utilisateur cohérente et réactive sur iOS, Android et le Web. Le back-end repose sur Firebase, qui fournit une solution robuste pour l'authentification, le stockage des données, et les fonctionnalités backend nécessaires. Flutter, avec sa capacité de compilation native, offre des performances élevées et une excellente réactivité, essentielles pour les fonctionnalités de scan en temps réel. L'utilisation de Firebase permet une gestion sécurisée des données utilisateurs et facilite l'intégration d'APIs externes pour la récupération des données produits.

2 UX

2.1 Solutions Utilisées

En termes d'expérience utilisateur (UX), EthiScan met l'accent sur la simplicité et l'efficacité. L'interface utilisateur est conçue pour être intuitive, permettant aux utilisateurs de naviguer facilement entre les différentes fonctionnalités. Le processus de scan de produit est rapide et direct, offrant un retour immédiat et des informations détaillées sur le produit scanné.
La personnalisation de l'application, via la configuration des préférences d'achat, permet aux utilisateurs de recevoir des informations spécifiquement alignées avec leurs valeurs et critères personnels. Cette approche centrée sur l'utilisateur renforce l'engagement et améliore l'expérience globale.

3 Évaluations

3.1 Évaluations Implémentées

L'évaluation de l'application s'est concentrée sur la performance, l'exactitude des données, et la satisfaction utilisateur. Des tests de performance ont été réalisés pour s'assurer que l'application répond rapidement aux requêtes de scan. L'exactitude des informations fournies a été vérifiée contre des sources externes pour garantir la fiabilité des données. Enfin, des enquêtes de satisfaction utilisateur ont aidé à recueillir des retours sur l'expérience d'utilisation, permettant d'identifier les domaines d'amélioration.

4 Techniques

4.1 Quoi et Comment

EthiScan implémente un système de scan de code-barres qui identifie les produits et récupère des informations à partir de bases de données externes via des API. L'application traite ces données pour afficher les informations pertinentes aux utilisateurs, telles que les labels, l'évolution des prix, et l'impact carbone.

4.2 Problèmes Rencontrés et Solutions

Version des dépendances Lors du développement, nous avons rencontré un problème majeur lié à la gestion des versions de Flutter utilisées par les différents membres de l'équipe. En effet, nous avons constaté qu'une personne utilisait la version 3.22 de Flutter, tandis que d'autres travaillaient avec la version 3.19. Cette disparité de versions a causé des conflits lors de l'intégration de nouvelles dépendances. Par exemple, certaines dépendances étaient compatibles avec une version de Flutter mais pas avec l'autre, ce qui a entraîné des erreurs de compilation. La synchronisation des versions de Flutter entre tous les membres de l'équipe s'est avérée difficile. Nous avions le choix entre downgrade Flutter ou de résoudre les conflits avec la dernière version. Nous avons opté pour la deuxième option, car elle nous permettait de bénéficier des dernières fonctionnalités et correctifs de Flutter. Cependant, cela a nécessité un effort supplémentaire pour résoudre les conflits et garantir la compatibilité des dépendances avec la version la plus récente de Flutter.

Structure de l'application Nous avons décidé de suivre les bonnes pratiques de la "clean architecture" ce qui a posé plusieurs défis. Comme chaque membre de l'équipe avait une expérience différente Flutter, nous avons tous développé des portions de code de manière indépendante, en utilisant des approches et des structures différentes, souvent inspirées d'exemples trouvés en ligne ou de projets précédents. L'intégration de ces morceaux de code disparates dans un cadre uniforme basé sur la clean architecture a nécessité des efforts significatifs. Nous avons dû refactorer et harmoniser les différentes méthodes de développement pour les aligner avec les principes de la clean architecture, ce qui a parfois causé des problèmes d'intégration et ralenti le développement de nouvelles features. Cette étape a souligné l'importance de définir dès le départ une architecture claire et partagée par toute l'équipe pour faciliter le développement collaboratif. Au final, nous avons utilisé le package "clean_architecture" pour structurer notre application.

Gestion des états Un autre défi a été la gestion des blocs, en particulier avec les pages de connexion et d'inscription. Initialement, nous avions un mainUserBloc chargé de l'authentification et de la gestion des données utilisateur dans toute l'application. Cependant, l'intégration des pages de connexion et d'inscription a posé problème. Ces pages, conçues pour fonctionner indépendamment, ne s'intégraient pas bien dans la structure de l'application en raison de problèmes liés aux locales de traduction. Pour contourner ce problème, nous avons décidé de placer ces pages directement dans le fichier app.dart, en dehors de la structure principale de l'application. Cette décision a introduit des complications avec le main user bloc, entraînant des conflits entre les blocs de gestion d'état. Bien que nous ayons finalement trouvé une solution. TODO Quelle solution ? Comment on a fix ce truc ?

Versionnement du code Nous avons rencontré quelques problèmes avec l'utilisation de GitHub au sein de notre équipe. L'un des membres, ne maîtrisant pas GitHub, a effectué tout son travail sur une branche existante déjà mergée sur le main, sans jamais fusionner ses modifications avec la branche principale (main). Par conséquent, le reste de l'équipe n'a pas pu suivre l'avancement de son travail. Au moment de fusionner sa branche avec la branche principale, nous avons découvert le problème,

car le code avait sur main avait été largement restructuré entre-temps. Cela a posé des problèmes d'intégration car le code avait été accumulé sur une longue période. Pour résoudre cette situation, nous avons résolu les conflits de versions progressivement jusqu'au succès. Puis, nous avons formé toute l'équipe à l'utilisation de GitHub. Désormais, chaque nouvelle tâche doit être accompagnée de la création d'une issue et d'une branche correspondante. Une fois la tâche terminée et approuvée, la branche est fusionnée dans la branche principale avec un code review. Cette approche, combinée à une meilleure communication entre les membres de l'équipe, a permis d'éviter les problèmes.

4.3 Conclusion Technique

La combinaison de Flutter et Firebase a prouvé son efficacité pour développer une application mobile performante et réactive. L'architecture choisie a permis une mise en œuvre rapide des fonctionnalités tout en maintenant une haute qualité et fiabilité des données.

4.4 Auto-Critique du Code

4.4.1 Points Positifs

- Code bien structuré et commenté, facilitant la maintenance et les mises à jour.
- Utilisation efficace des patterns de conception pour une architecture solide.

4.4.2 Points à Améliorer

- Couverture des tests unitaires à augmenter pour assurer une meilleure stabilité.
- Optimisation possible de certaines requêtes de données pour accélérer les temps de réponse.

5 Annexes

5.1 Cahier des Charges Original

- **Introduction**: EthiScan est une application mobile conçue pour permettre aux utilisateurs de scanner des produits et de recevoir des informations détaillées alignées avec leurs valeurs personnelles de consommation. Elle vise à promouvoir une consommation responsable en fournissant des données telles que l'évolution du prix, les labels environnementaux et nutritionnels, l'impact carbone, et plus encore.
- **Objectifs du Projet**: Aider les utilisateurs à faire des choix de consommation éclairés et responsables. Fournir des informations détaillées et fiables sur les produits scannés. Promouvoir les achats alignés avec les valeurs personnelles des utilisateurs, comme le bio, le local, la qualité, le prix, l'impact carbone, la durabilité de l'emballage, et la possibilité de livraison par la poste.
- **Fonctionnalités Principales**:
  - **Scan de Produit**: Permettre le scan de codes-barres pour identifier rapidement les produits.
  - **Liste des Produits Favoris**: Possibilité d'ajouter des produits à une liste de favoris pour un accès rapide.
  - **S'abonner aux Metadatas**: Configuration de préférences d'achat personnalisées : Local, Bio, Qualité, Prix, Impact carbone, Durabilité de l'emballage, Livrable par la poste.
  - **Sections Détaillées des Métadonnées**:
    * **Labels**: Affichage des labels et certifications (éco-labels, bio, etc.).
    * **Évolution du Prix**: Visualisation de l'évolution du prix chez différents fournisseurs.
    * **Impact Carbone**: Information sur l'empreinte carbone du produit.
    * **Metadata**: Informations générales (nom du produit, lien vers plus d'infos).
- **Stack Technologique**: Front-end: Flutter pour une expérience utilisateur cohérente sur iOS, Android et le Web. Back-end: Firebase pour l'authentification, le stockage des données, et les fonctions backend.
- **Spécifications Techniques**:
  - **Exigences Fonctionnelles**: Authentification sécurisée des utilisateurs. Interface intuitive pour le scan de produits et l'affichage des informations. Système de favoris et de préférences personnalisables. Intégration d'APIs externes pour la récupération des données produits.
  - **Exigences Non-Fonctionnelles**: Performances: Temps de réponse rapide pour le scan et l'affichage des données. Accessibilité: Conception inclusive pour une utilisation facile par tous.
- **Deadlines**:
  - Formation groupes et choix du sujet du mini-projet – Semaine 1
  - Descriptif du projet (mini cahier de charges) – A remettre avant le cours de la semaine 2
  - Validation du projet – semaine 3 – en classe
  - Présentations du mini-projet (avec démo) – Semaines 14-15
  - Livraison d'un prototype fonctionnel et la rédaction d'un rapport (15-20 pages). A rendre le lundi avant la dernière séance

- **Conclusion**: EthiScan ambitionne de devenir une référence pour les consommateurs souhaitant aligner leurs achats avec leurs valeurs personnelles. Par la transparence et la fourniture d'informations détaillées, l'application vise à promouvoir une consommation plus responsable et éclairée.

5.2 Planning Actualisé Avant/Après

- Avant: Formation des groupes et choix du sujet du mini-projet - Semaine 1
- Après: Livraison d'un prototype fonctionnel et la rédaction d'un rapport - Lundi avant la dernière séance

5.3 Liste des Bugs Connus

- Erreur de connexion intermittente au service de scan de code-barres.
- Problèmes d'affichage sur certaines versions d'Android.
- Latence lors du chargement des données produits pour les articles récemment ajoutés.

5.4 Dépendances

- Flutter
- Firebase
- API externe pour les données produits

5.5 Aides Extérieures

- Documentation officielle de Flutter (`https://flutter.dev/docs`)
- Documentation officielle de Firebase (`https://firebase.google.com/docs`)
- Discussions avec des collègues pour optimiser les requêtes API.