

Développement hybride : Mini-projet React Native

Sujet : "EcoAction" – Plateforme de Bénévolat Environnemental

1. Contexte du projet

L'objectif est de développer une application mobile permettant aux citoyens de découvrir, de s'inscrire et de gérer des missions de bénévolat locales (nettoyage de plages, plantation d'arbres, ateliers zéro déchet). L'application doit offrir une expérience fluide, même en cas de réseau instable, grâce à une gestion intelligente du cache.

2. Fonctionnalités attendues (MVP)

- **Authentification (Simulée ou réelle)** : Connexion/Inscription d'un utilisateur.
- **Exploration des missions** :
 - Liste des missions disponibles avec filtres par catégorie.
 - Recherche textuelle.
 - Détails d'une mission (description, date, lieu, nombre de places restantes).
- **Gestion des participations** :
 - S'inscrire à une mission (Mutation).
 - Annuler sa participation.
 - Vue "Mes Missions" pour consulter son agenda personnel.
- **Profil Utilisateur** : Affichage des statistiques (nombre d'actions réalisées).

3. Contraintes Techniques (Impératives)

Frontend : React Native & Expo

- Utilisation d'**Expo SDK** (dernière version stable).
- Structure de navigation via **Expo Router** (recommandé) ou **React Navigation**.
- Interface soignée avec une bibliothèque comme **NativeWind** (Tailwind CSS) ou **Tamagui**.

Langage : TypeScript

- Typage strict (pas de any).
- Définition d'interfaces pour les objets API (Missions, Users, API Errors).

Gestion de l'état & Réseau : TanStack Query (React Query)

- Utilisation de useQuery pour le fetching des données.
- Utilisation de useMutation pour les inscriptions/annulations.
- Mise en place de l'**Optimistic UI** (mise à jour de l'interface avant confirmation du serveur) pour l'inscription aux missions.
- Gestion des états de chargement (isLoading) et d'erreur (isError).

- Configuration du cache (staleTime, cacheTime).

Backend / API

- Utilisation d'une API REST. (Options suggérées : **JSON-Server** pour un mock rapide, ou **Supabase** pour une solution Backend-as-a-Service réelle).

4. Livrables attendus

1. **Le code source** : Un dépôt GitHub proprement organisé (README clair, commits explicites).
2. **Une démonstration** : Une courte vidéo (GIF ou MP4) montrant les flux principaux.
3. **Une note technique (2 pages max)** : Justification de l'architecture, gestion des types complexes et stratégie de mise en cache choisie avec TanStack.

N.B : Date limite de soumission des travaux fixée au vendredi 20/02 à 9h. Une validation en classe à l'ISET est également prévue à cette date en plus des livrables demandés.