# Sujet de projet : AnimeCollect - Application de gestion de collection d'animés

## Introduction

Ce projet consiste à développer une application mobile de gestion de collection d'animés, permettant aux utilisateurs de suivre leurs séries animées, découvrir les nouveautés et organiser leur collection personnelle. L'application sera développée avec Expo et React Native, intégrera une API externe de référence d'animés, et utilisera SQLite avec Drizzle ORM comme base de données locale pour un fonctionnement hors ligne. Une attention particulière sera accordée à la sécurité et aux tests de l'application.

## Objectifs pédagogiques

- Développer une application React Native complète avec Expo
- Maîtriser la navigation entre les écrans avec Expo Router
- Intégrer une API externe (Kitsu API ou autre si vous avez mieux)
- Implémenter une base de données locale avec SQLite et Drizzle ORM
- Utiliser Tailwind CSS via la bibliothèque TWRNC pour le stylisme
- Créer des interfaces utilisateur réactives et esthétiques
- Concevoir une expérience utilisateur fluide et intuitive
- Mettre en place une stratégie de tests complète
- Implémenter des mesures de sécurité pour protéger les données utilisateur

## Contexte du projet

Les passionnés d'animés ont souvent du mal à suivre leur collection : quelles séries ils regardent déjà, quels épisodes ils ont visionnés, quelles sont les dernières sorties de leurs séries préférées. AnimeCollect vise à résoudre ces problèmes en offrant une application complète de gestion de collection d'animés.

## Fonctionnalités requises

### Écran des nouveautés

- Affichage des dernières sorties d'animés (images, titres, numéros d'épisodes)
- Vue détaillée d'un épisode au clic (image, titre, réalisateur, date de sortie, synopsis)
- Boutons d'action : "Marquer comme vu" et "Ajouter à ma liste à regarder"
- Mise à jour via l'API Kitsu (ou autre si vous avez mieux)

### Écran de ma collection

- Liste des séries dans la collection de l'utilisateur
- Affichage du titre, du nombre d'épisodes visionnés et du nombre total d'épisodes
- Barre de progression visuelle pour chaque série
- Vue détaillée d'une série au clic, montrant tous les épisodes
- Badge visuel indiquant les épisodes visionnés

- Vue détaillée d'un épisode au clic
- Persistance des données avec SQLite

### Écran de recherche

- Barre de recherche pour filtrer les animés
- Affichage des résultats avec images et informations de base
- Vue détaillée d'une série au clic, montrant tous les épisodes disponibles
- Vue détaillée d'un épisode spécifique au clic
- Option pour marquer un épisode comme vu ou l'ajouter à la liste à regarder

### Tests et qualité du code

- Tests unitaires pour les composants clés
- Tests d'intégration pour les fonctionnalités principales
- Tests de la base de données et des services
- Documentation du code (JSDoc)
- Couverture de tests minimale de 70%

### Sécurité de l'application

- Validation des entrées utilisateur
- Gestion sécurisée des communications avec l'API
- Protection contre les injections SQL
- Gestion appropriée des erreurs sans exposition d'informations sensibles

### Fonctionnalités supplémentaires

- Design responsive avec thème cohérent utilisant TWRNC (Tailwind CSS pour React Native)
- Gestion des erreurs et des états de chargement
- Fonctionnement hors ligne pour consulter sa collection

## Contraintes techniques

- Utilisation obligatoire d'Expo SDK et React Native
- Utilisation de TWRNC (tw from "twrnc") pour le stylisme
- Intégration de Kitsu API pour les données d'animés (ou autre si vous avez mieux)
- Stockage local via SQLite avec Drizzle ORM (https://orm.drizzle.team/)
- Navigation avec Expo Router
- Tests avec Jest et React Native Testing Library
- Mise en place de mesures de sécurité
- Structure de projet claire et modulaire
- Code commenté et documenté

## Structure du projet suggérée

```
app/
├── (tabs)/
│   ├── index.tsx           # Écran des nouveautés
│   ├── collection.tsx      # Écran de ma collection
│   ├── search.tsx          # Écran de recherche
│   └── _layout.tsx         # Layout des onglets
├── anime/
│   ├── [id].tsx            # Détails d'une série animée
```

```
│   └── [id]/[episode].tsx  # Détails d'un épisode spécifique
├── collection/
│   └── [id].tsx            # Vue d'une série dans la collection
├── components/
│   ├── AnimeCard.tsx       # Carte affichant un animé
│   ├── EpisodeCard.tsx     # Carte affichant un épisode
│   ├── ProgressBar.tsx     # Barre de progression
│   └── ...
├── hooks/
│   ├── useDatabase.ts      # Hook pour SQLite avec Drizzle
│   ├── useAnimeApi.ts      # Hook pour l'API Jikan
│   └── ...
├── services/
│   ├── apiService.ts       # Service d'accès à l'API
│   ├── databaseService.ts  # Service de base de données
│   └── securityService.ts  # Service de sécurité
├── utils/
│   ├── formatters.ts       # Utilitaires de formatage
│   ├── validators.ts       # Validation des entrées
│   └── ...
├── db/
│   ├── schema.ts           # Schéma Drizzle ORM
│   ├── migrations/         # Migrations de la base de données
│   └── index.ts            # Configuration de la base de données
├── tests/
│   ├── components/         # Tests des composants
│   ├── hooks/              # Tests des hooks
│   ├── services/           # Tests des services
│   └── db/                 # Tests de la base de données
└── _layout.tsx             # Layout principal
```

## Points d'attention

Attention l'installation du projet, vous devez être en version 52 de Expo, car la version 53 est encore en bêta

## Points d'attention

- L'interface devra être intuitive pour faciliter la gestion d'une grande collection d'animés.

## Points optionnels

- Les APIs d'animés disponibles ne proposent généralement pas les titres en français. Une approche possible serait d'afficher le titre original et de permettre à l'utilisateur d'éditer manuellement certaines informations pour sa collection personnelle.
- La gestion des saisons multiples pour une même série animée doit être prise en compte dans la conception de la base de données.
- Certains animés peuvent avoir des épisodes spéciaux ou des OVAs qui devront être correctement catégorisés.

## Ressources utiles

- Documentation Kitsu API : https://kitsu.docs.apiary.io/
- Documentation Jikan API (API MyAnimeList non officielle): https://jikan.moe/
- Documentation Expo: https://docs.expo.dev/
- Documentation TWRNC: https://www.npmjs.com/package/twrnc
- Documentation SQLite: https://docs.expo.dev/versions/latest/sdk/sqlite/
- Documentation Drizzle ORM: https://orm.drizzle.team/
- Documentation Expo Router: https://docs.expo.dev/router/introduction/
- Documentation Jest: https://jestjs.io/
- Documentation React Native Testing Library: https://callstack.github.io/react-native-testing-library/

## Livrables attendus

1. Code source complet de l'application
2. Documentation d'installation et d'utilisation
3. Présentation du projet
4. Démonstration de l'application fonctionnelle
5. Rapport de tests incluant la couverture de code
6. Diagramme UML des classes de l'application

Bon courage pour ce projet !