Développer une telle application mobile avec React Native et Expo est tout à fait possible et serait une bonne approche pour un tel projet. Voici un exemple de structure générale pour votre projet et les outils dont vous pourriez avoir besoin :

**\*\*1. Configuration de l'environnement de développement :\*\***

Vous devrez installer Node.js, npm (ou Yarn), Expo CLI sur votre machine de développement. Vous pouvez installer Expo CLI avec la commande suivante :

```bash

npm install -g expo-cli

```

**\*\*2. Création de l'application :\*\***

Vous pouvez créer une nouvelle application à l'aide de la commande suivante :

```bash

expo init nom\_de\_votre\_application

```

**\*\*3. Structure de l'application :\*\***

Voici un exemple de structure d'application que vous pourriez suivre :

- `/assets`: pour stocker les images, les polices, et autres fichiers statiques.

- `/components`: pour tous vos composants réutilisables.

- `/screens`: pour toutes les pages de votre application (par exemple, HomePage, ProductPage, etc.)

- `/api`: pour toutes les requêtes liées à la base de données ou toute autre API.

- `/navigation`: pour gérer la navigation entre vos écrans.

- `/state`: pour gérer l'état de l'application à l'aide de Context API ou Redux.

- `/utils`: pour toute fonction utilitaire.

**\*\*4. Développement de l'application :\*\***

- \*\*Prise de photo et stockage d'images\*\* : vous pouvez utiliser `expo-camera` pour prendre des photos et `expo-file-system` pour sauvegarder ces images dans le système de fichiers de l'appareil. Pour stocker les images dans une base de données, vous pourriez avoir à les convertir en base64 ou les stocker dans un cloud et conserver le lien dans votre base de données.

- \*\*Base de données\*\* : Vous pouvez utiliser Firebase pour créer et gérer votre base de données. Firebase offre une base de données en temps réel et c'est assez simple à utiliser avec React Native.

- \*\*Recherche de produits\*\* : Vous pouvez mettre en place une fonction de recherche en filtrant les données de votre base de données selon les critères de recherche de l'utilisateur.

- \*\*Alertes\*\* : Vous pouvez utiliser `expo-notifications` pour envoyer des notifications locales à l'utilisateur.

Ceci n'est qu'une ébauche rapide et il existe de nombreux détails que vous devrez traiter en cours de route. Si vous avez des questions spécifiques sur l'un de ces sujets, n'hésitez pas à demander.

V1.

Excellent choix ! SQLite est une bonne option pour le stockage local des données sur l'appareil, et Expo fournit un module pour interagir avec une base de données SQLite.

\*\*Installation :\*\*

SQLite est inclus dans le SDK Expo, vous n'avez donc pas besoin d'installer quoi que ce soit de plus pour l'utiliser.

\*\*Utilisation :\*\*

Voici un exemple basique d'utilisation de SQLite avec Expo :

```javascript

import \* as SQLite from 'expo-sqlite';

const db = SQLite.openDatabase('db.db'); // Remplacer 'db.db' par le nom de votre base de données

// Exécution d'une requête SQL

db.transaction(tx => {

tx.executeSql(

'create table if not exists items (id integer primary key not null, name text, quantity integer, expiration\_date text);'

);

});

```

Cette instruction ouvrira une base de données (et la créera si elle n'existe pas) puis exécutera une requête SQL pour créer une table `items` si elle n'existe pas déjà.

Pour \*\*insérer\*\* des données, vous pouvez utiliser une requête SQL INSERT comme ceci :

```javascript

const name = 'Produit';

const quantity = 10;

const expiration\_date = '2022-12-31';

db.transaction(tx => {

tx.executeSql('insert into items (name, quantity, expiration\_date) values (?, ?, ?)', [name, quantity, expiration\_date]);

});

```

Pour \*\*récupérer\*\* des données, vous pouvez utiliser une requête SQL SELECT :

```javascript

db.transaction(tx => {

tx.executeSql('select \* from items', [], (\_, { rows }) =>

console.log(JSON.stringify(rows))

);

});

```

Et pour \*\*rechercher\*\* des produits, vous pouvez ajuster votre requête SELECT :

```javascript

const search = 'Produit'; // le produit à rechercher

db.transaction(tx => {

tx.executeSql('select \* from items where name = ?', [search], (\_, { rows }) =>

console.log(JSON.stringify(rows))

);

});

```

J'espère que cela vous donne une bonne idée de la façon d'utiliser SQLite avec Expo dans React Native. Si vous avez des questions supplémentaires, n'hésitez pas à les poser !