Compte-rendu: TP Flutter - Netflim - Recherche

Nathan CAVA-LONCHAMP, SIO2

1 - Gestion de la recherche dynamique

1°) Commencer par supprimer l'onglet "Recherche". Note: veiller à la continuité du bon fonctionnement des autres onglets.

➜ *Fournir le code modifié.*

```
_tabController = TabController(length: 3, vsync: this);
```

On modifie le length de TabController pour qu'il ne contienne que 3 onglets.

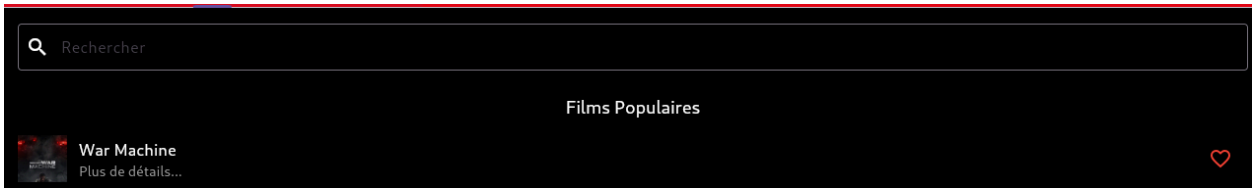
```
tabs: const [
  Tab(icon: Icon(Icons.home), text: 'Accueil'),
  Tab(icon: Icon(Icons.favorite), text: 'Favoris'),
  Tab(icon: Icon(Icons.list), text: 'A regarder'),
]
```

On retire l'icône Recherche

```
Container(
  color: UserPreferences().backgroundColor,
  child: const Center(
    child: Text(
      'Contenu de la recherche',
      style: TextStyle(color: Colors.white),
    ),
  ),
),
```

On retire également le contenu de l'onglet Recherche

2°) Si ce n'est déjà fait, mettre la méthode showSearchDialog() en commentaire. Elle n'est plus exploitée puisqu'elle était appelée par l'AppBar de MovieListScreen qui a été désactivée voire supprimée lors du dernier TP. Dans votre projet, ajouter la classe movie_search_bar.dart dans le répertoire widgets. Modifier ensuite la classe MovieListScreenState pour y intégrer le widget MovieSearchBar. ➜ Fournir le code modifié.




```dart
body: Column(
  children: [
    MovieSearchBar(onQueryChanged: (query) {
      // + tard
    }), // MovieSearchBar
```

On ajoute la MovieSearchBar avant le reste de la page.

```dart
onQueryChanged(String search) async {
  if (search.isEmpty) {
    title = "Films Populaires";
    movies = await ApiService().getPopularMovies(1);
  } else {
    title = "Votre Recherche";
    movies = await ApiService().searchForMovies(1, search);
  }
  setState(() {});
}
```

Code de la recherche

```
movies: widget.popularMovies!, title: "Films Populaires", displaySearchBar: true)),
```

On ajoute displaySearchBar aux instances de MovieListScreen

Dans le répertoire services, créer une classe NetflimTheme et y transférer la méthode theme(). Modifier le reste du projet en conséquence. ➜ Fournir le code de la classe NetflimTheme.

```
return MaterialApp(
  home: screen,
  debugShowCheckedModeBanner: false,
  theme: NetflimTheme.theme()
);
```

```
class NetflimTheme {
  /// Thème spécifique pour la TabBar
  static ThemeData theme() {
    return ThemeData(
      primaryColor: UserPreferences().backgroundColor,
      tabBarTheme: TabBarThemeData(
        labelColor: UserPreferences().mainTextColor,
        unselectedLabelColor: UserPreferences().mainTextColor.withOpacity(0.7),
        indicator: UnderlineTabIndicator(
          borderSide: BorderSide(
            color: UserPreferences().mainTextColor,
            width: 3.0,
          ), // BorderSide
        ), // UnderlineTabIndicator
      ), // TabBarThemeData
      // Ajout pour la searchbar
      inputDecorationTheme: InputDecorationTheme(
        labelStyle: TextStyle(color: UserPreferences().mainTextColor),
        focusedBorder: UnderlineInputBorder(
          borderSide: BorderSide(
              style: BorderStyle.solid,
              color: UserPreferences().mainTextColor), // BorderSide
        ) // UnderlineInputBorder
      ), // InputDecorationTheme
    ); // ThemeData
  }
}
```

Bonus

Finaliser l'application (ex: gestion correcte des cœurs dans le favoris, etc). A vous de jouer en vous mettant à la place de l'utilisateur final.