**Тема проекту:** веб-сайт для пошуку фільмів з можливістю зберігати фільми до списку збережених фільмів.

**Мета проекту:** створення веб-сайту для швидкого пошуку фільму, який допоможе вибрати кінострічку або телешоу для перегляду на дозвіллі з огляду на оцінки інших користувачів.

**Функціональні можливості:**

1) Перегляд підбірок найпопулярніших фільмів та фільмів із найвищим рейтингом

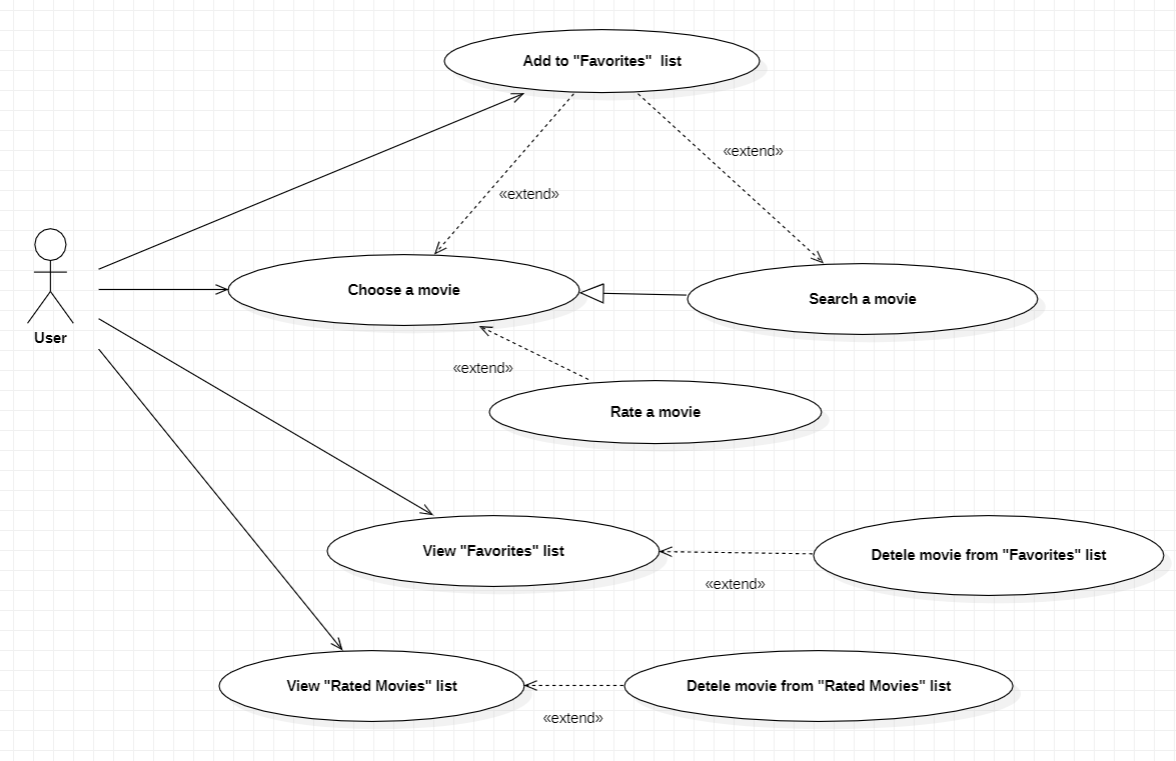
2) Перегляд загального переліку наявних фільмів та телешоу

3) Перегляд детальної інформації про фільм

4) Додавання/видалення фільмів зі списку збережених фільмів

5) Можливість оцінки фільму

6) Пошук фільма або телешоу за повною назвою або її частиною



**Встановлення проекту (для розробників)**

1. Спершу виконайте клонування репозиторію

2. Перейдіть до папки із проектом на вашому ПК

3. Встановіть або оновіть усі необхідні пакети за допомогою команди flutter pub get

4. Запустіть проект за допомогою команди flutter run та виберіть платформу, на якій проект запуститься.

**Основні класи проекту:**

**Папка *lib/shared/router/***

Класи MainRoutePath та MainRouteName містять статичні поля із назвами маршрутів та відповідними шляхами цих маршрутів для навігації по сайту.

Наприклад,

class MainRouteName {

   static const String films = 'films'; // Маршрут для фільмів

  …

}

class MainRoutePath {

static const String films = '/films'; // Шлях до маршруту фільмів

…..

}

Клас MainRouter відповідає за навігацію в додатку, визначаючи шляхи для різних сторінок і відображаючи їх відповідно до поточного стану додатку.

Клас MultiplatformPage надає методи для створення сторінок, які адаптовані до різних платформ.

Клас LayoutHelper виконує допоміжні функції для визначення типу екрану та кількості колонок для сітки на основі розміру екрану. Основна його мета - це спростити розробку інтерфейсу на Flutter, роблячи адаптивний дизайн більш доступним і простим для використання. Він надає методи для перевірки типу екрану (мобільний, планшетний, настільний) і отримання кількості колонок для сітки залежно від типу екрану.

**Папка *lib/shared/theme/***

Клас AppColor – це абстрактний клас для зберігання кольорів у змінних, які будуть використовуватися для стилізації віджетів.

**Папка *lib/shared/widgets/***

Класи OneAppBar, OneApp, OneButton, OneDrawer, OneImage, OneScaffold, OneShimmer відповідають за побудову різноманітних та привабливих віджетів, з яких складається інтерфейс користувача в застосунку Flutter.

**Папка *lib/presentation/favourite/pages/***

Клас FavoritePage є сторінкою, яка відображає список улюблених фільмів та серіалів. Він містить конструктор та метод build, який відповідає за побудову інтерфейсу сторінки з урахуванням поточного стану (в тому числі, помилок, ініціалізації та завантаження).

Клас FavouriteShowGrid є віджетом, який відображає сітку з улюбленими фільмами та серіалами. Він містить список елементів, які будуть відображені у сітці, та відповідає за їх відображення.

**Папка *lib/presentation/favourite/bloc/***

Клас FavoriteCubit є кубітом (частиною архітектури BLoC) і використовується для керування станом улюблених фільмів та телепередач. Метод **init()** ініціалізує стан кубіту, завантажуючи улюблені фільми та телешоу з репозиторію. Після завершення завантаження, стан оновлюється зі списками улюблених фільмів і телепередач, а також змінюється статус на "ініціалізований".У разі виникнення помилки під час завантаження, стан змінюється на "помилка" із повідомленням про помилку.

Клас FavoriteState представляє стан улюблених елементів (фільмів та телепередач). Містить властивості, такі як статус, повідомлення про помилку, списки улюблених фільмів та телепередач.Має фабричний метод **loading()**, який створює початковий стан під час завантаження, ініціалізуючи статус як "завантаження" і порожні списки фільмів та телепередач.

**Папка *lib/presentation/films/***

MoviesPage - це клас, який представляє сторінку з відображенням фільмів. Він управляє відображенням списку фільмів та їхніх деталей.

**Папка *lib/presentation/films/widgets/***

Клас OneMediaCard є віджето, який призначений для відображення картки медіа з деякою інформацією, такою як зображення, назва медіа, дата виходу, популярність та рейтинг. Головною метою цього класу є створення стилізованого віджету для представлення цих даних та обробки натискань на нього. Він дозволяє виконувати певні дії при натисканні на картку, наприклад, перехід на сторінку деталей медіа (MainRouteName.movieDetails).

**Папка *lib/presentation/films/bloc/***

Клас MoviesCubit є частиною програми на Flutter для керування списком фільмів. Основна його мета - обробка запитів користувача щодо пошуку фільмів за певною сторінкою та відображення стану цього пошуку в інтерфейсі користувача.

**Папка *lib/presentation/main/pages/***

Клас MainPage є сторінкою відображення головного екрану додатку. Основна його мета - показати список трендових фільмів та фільмів з найвищим рейтингом.

**Папка *lib/presentation/main/ pages /widgets/***

Клас MovieCards відповідає за відображення списку карток фільмів. Основна його мета полягає в тому, щоб відобразити список фільмів у формі горизонтального ListView, де кожен елемент списку представлений відповідною карткою фільму (OneMovieCard).

1. Конструктор: Приймає список фільмів (movies) і список улюблених фільмів (favouriteMovies).
2. Метод build: Відповідає за побудову віджету. В цьому методі використовується ListView.builder для відображення горизонтального списку фільмів. Кожен елемент списку створюється за допомогою віджету OneMovieCard, який приймає відповідний фільм і визначає, чи він є улюбленим.
3. Метод dispose: Викликається для видалення контролера прокрутки після закриття класу.

Клас OneMovieCard є частиною інтерфейсу користувацького інтерфейсу і представляє карточку для відображення інформації про один фільм. Основна мета класу - створити каркас для відображення інформації про фільм, такої як зображення, назва, рік випуску, популярність та рейтинг. Карточка також містить інтерактивні елементи, такі як кнопка "В обране", яка дозволяє користувачеві додавати або видаляти фільм зі списку обраних.

**Папка *lib/presentation/media\_details/pages/***

Клас MovieDetailsPage є сторінкою деталей фільмів або телешоу. Він відображає деталі конкретного фільму або телешоу залежно від типу контенту, який представлений у MediaDetailsState. Клас має такі основні функції:

1. Отримує стан даних через MediaDetailsCubit за допомогою BlocProvider.
2. Залежно від типу контенту (фільм або телешоу) відображає відповідні деталі.
3. Містить віджет MovieDetailsWidget, який відображає деталі фільму або телешоу.
4. Дозволяє користувачеві додавати фільми або телешоу в обране або змінювати їх рейтинг.

Клас MovieDetailsWidget відображає деталі фільму. Він містить різні віджети для показу інформації про фільм, такі як заголовок, зображення, рейтинг, країну, жанр, компанію, опис тощо. Також він має функціональність для оцінки фільму та додавання його до списку обраного. Клас розділений на частини для різних екранних розмірів, щоб забезпечити оптимальний дизайн для різних пристроїв.

**Папка *lib/presentation/search /pages/***

Клас MobileSearchPage є сторінкою пошуку у мобільному форматі. Він відображає текстове поле для введення запиту пошуку та результати пошуку. Крім того, він містить логіку для перевірки ширини екрану, і якщо вона більша за 1100, то відбувається перехід на домашню сторінку додатку.

Клас, OneResultCard, є віджетом, який призначений для відображення одного результату пошуку медіа. Його головна мета - відображати інформацію про медіа, яку він отримує через конструктор. Крім того, він включає обробник подій для натискання, який відправляє дані про обраний елемент медіа в кубіт деталей медіа, а потім переходить на екран деталей медіа за допомогою маршрутизатора.

Клас SearchPage виконує роль сторінки пошуку відображення результатів. Основна його мета - надати користувачу можливість ввести запит для пошуку та відображення результатів цього пошуку. Він має такі основні елементи:

1. TextEditingController: Це контролер для текстового поля, в якому користувач може ввести свій запит для пошуку.
2. SearchCubit: Цей клас використовується для управління станом пошуку. Клас SearchCubit містить логіку пошуку та управління станом.
3. Stack зі Stack: Користується для накладання різних елементів один на одного. У цьому випадку використовується для накладання текстового поля та карточки результатів пошуку.
4. OneTextField: Це власний віджет для текстового поля вводу. Використовується для введення запиту на пошук.
5. OneCard: Власний віджет карточки, що містить результати пошуку.
6. \_buildBody: Цей метод відповідає за відображення тіла сторінки в залежності від статусу пошуку. Якщо статус - помилка, відображається текст помилки. Якщо статус - ініціалізація, відображається контейнер з результатами пошуку. Якщо статус - інший, відображається індикатор завантаження.

Клас SearchResults виконує роль відображення результатів пошуку.

Клас SearchWrapper використовується для створення універсального контейнера, який має можливість відображення навігаційної панелі (для настільних екранів), вмісту сторінки, підтримки функціоналу пошуку та його відображення на настільному екрані.

**Папка *lib/presentation/tv /pages/***

Клас TVShowPage використовується для відображення списку телевізійних шоу. Основні функції цього класу включають:

1. Ініціалізація прокрутки та створення прослуховувача запитів на нові сторінки при пагінації.
2. Рендеринг списку телевізійних шоу за допомогою PagedGridView, що підтримує безкінечну пагінацію.
3. Кожен елемент списку представлений за допомогою OneMediaCard, який містить зображення телевізійного шоу та інші деталі.
4. Обробка випадків, коли немає елементів для відображення або коли завантажуються нові елементи.
5. Перехід до деталей обраного телевізійного шоу при натисканні на нього.

**Папка *lib /domain/entity /***

Клас MovieItem визначає об'єкт фільму. Цей клас допомагає створювати, ініціалізувати та перетворювати об'єкти фільмів для подальшої обробки в додатку.

Клас TVShowItem визначає модель телевізійного шоу з різними характеристиками та даними. Цей клас дозволяє представляти дані телевізійних шоу з різними джерелами даних та надає зручні методи для створення та отримання необхідних даних.

Клас SearchItem призначений для представлення елементів пошуку з інформацією про назву, тип контенту та посилання на зображення постера. Клас також містить методи для створення об'єктів з різних джерел та отримання URL зображення постера.

**Папка *lib/domain/repositories/***

Клас LocalRepository відповідає за взаємодію з локальним джерелом даних, щоб здійснювати операції збереження, вилучення та отримання даних про фільми та серіали. Основна мета класу - це забезпечення інтерфейсу для виконання таких операцій, приховуючи деталі взаємодії з локальним джерелом даних за допомогою LocalDatasource.

Основні поля класу:

\_datasource: Об'єкт типу LocalDatasource, який використовується для взаємодії з локальною базою даних.

Основні методи класу:

* saveAsFavouriteMovie: Збереження фільму як улюбленого.
* removeFromFavouriteMovie: Видалення фільму зі списку улюблених.
* getFavouriteMovies: Отримання списку улюблених фільмів.
* saveAsFavouriteTVShow: Збереження серіалу як улюбленого.
* removeFromFavouriteTVShow: Видалення серіалу зі списку улюблених.
* getFavouriteTVShows: Отримання списку улюблених серіалів.
* saveRatingMovie: Збереження рейтингу фільму.
* saveRatingTVShow: Збереження рейтингу серіалу.
* getRatingMovie: Отримання рейтингу фільму.
* getRatingTVShow: Отримання рейтингу серіалу.

Клас надає інтерфейс для роботи з фільмами та серіалами, включаючи додавання їх до списку улюблених, вилучення з цього списку та отримання їх рейтингу.

Клас RemoteRepository є частиною архітектури програми, яка взаємодіє з віддаленими даними, такими як фільми і телешоу, за допомогою API. Основною його метою є надання способу отримання різних категорій фільмів та телешоу, а також детальної інформації про кожен окремий елемент.

Основні поля цього класу:

* RemoteDatasource datasource: Це джерело даних для отримання інформації з віддаленого джерела, яке реалізує інтерфейс або клас для взаємодії з API.
* String apiKey: Це ключ API, який необхідно використовувати для отримання доступу до віддалених даних.

Основні методи цього класу:

* getTrendingMovies(): Отримання списку трендових фільмів.
* getTopRatedMovies(): Отримання списку фільмів з найвищим рейтингом.
* getAllMovies({required int page}): Отримання всіх фільмів з пагінацією.
* getAllTVShows({required int page}): Отримання всіх телешоу з пагінацією.
* search({required String query, required int page}): Пошук за запитом з пагінацією.
* getMovieDetail(int id): Отримання деталей фільму за його ID.
* getTVShowDetail(int id): Отримання деталей телешоу за його ID.

**Папка *lib /data/datasource/***

Rлас LocalDatasource відповідає за роботу з локальним сховищем даних для зберігання та отримання інформації про фільми та телесеріали у мобільному . Основна його мета - забезпечити можливість зберігання вибраних фільмів і телесеріалів, а також їх рейтингів у локальному сховищі, щоб користувач міг легко доступатися до цих даних навіть без підключення до Інтернету.

Основні поля класу:

* moviesBox: Сховище для зберігання об'єктів типу MovieModel.
* tvShowsBox: Сховище для зберігання об'єктів типу TVShowModel.
* ratingBox: Сховище для зберігання рейтингів.

Основні методи класу:

* saveAsFavouriteMovie: Збереження фільму у вибрані.
* removeFromFavouriteMovie: Видалення фільму з вибраних.
* getFavouriteMovies: Отримання списку вибраних фільмів.
* saveAsFavouriteTVShow: Збереження телесеріалу у вибрані.
* removeFromFavouriteTVShow: Видалення телесеріалу з вибраних.
* getFavouriteTVShows: Отримання списку вибраних телесеріалів.
* saveRatingMovie: Збереження рейтингу для фільму.
* saveRatingTVShow: Збереження рейтингу для телесеріалу.
* getRatingMovie: Отримання рейтингу для фільму.
* getRatingTVShow: Отримання рейтингу для телесеріалу.

Клас RemoteDatasource є абстрактним і визначає контракт для взаємодії з віддаленими даними через HTTP запити до API. Він використовує Retrofit, щоб спростити визначення екземплярів цього класу на основі інтерфейсу та автоматичної генерації коду.

**Папка *lib /data/model/local/***

Класи MovieModel та TVShowModel є моделями даних для представлення фільму та телешоу відповідно в додатку Flutter. Він є підтримкою для роботи з базою даних Hive, що є локальним ключовим-значенним сховищем у Flutter.

Основна суть цього класу полягає в тому, щоб мати структуру даних для збереження основних характеристик фільмів(телешоу), таких як ідентифікатор, назва, мова, оригінальна назва, огляд, посилання на постер, середній рейтинг, кількість голосів, дата виходу та популярність.

Анотація @HiveType(typeId: 0) вказує, що клас є Hive-типом з ідентифікатором 0, тобто його можна зберігати в базі даних Hive.

**Папка *lib /core/resources/***

Клас GlobalConstants є абстрактним і містить константи, які використовуються для доступу до API з фільмів і телешоу.

Клас DimensionConstants визначає абстрактні класи з константами для розмірів елементів, відступів і радіусів закруглення. Кожен клас містить різні константи з певними значеннями, що представляють різні розміри і параметри, які можуть використовуватися при розробці інтерфейсу користувача, особливо при налаштуванні відображення елементів, їх положення та вигляд.

**Папка *lib /core/services/***

Клас LocalStorageService відповідає за управління локальним сховищем даних у Flutter додатку. Основна його функція - ініціалізація та управління боксами (контейнерами) Hive, що є локальною базою даних у Flutter.

Основні методи та їх функціонал:

1. execute(Function() func, int count): Цей метод виконує функцію, яка передається як параметр, та відстежує кількість спроб її виконання. Якщо кількість спроб перевищує максимально допустиму кількість (maxRetries), генерується виняток про неуспішну ініціалізацію Hive.
2. init({int retryCount = 0}): Цей метод відповідає за ініціалізацію Hive. Спочатку він ініціалізує пакет Hive для використання у Flutter додатку, реєструє адаптери для моделей даних, та відкриває бокси для зберігання даних. Якщо ініціалізація не вдалася, виконується спроба повторної ініціалізації або видаляються бокси з диску.
3. openBox({int retryCount = 0}): Цей метод відповідає за відкриття боксів Hive для конкретних типів даних і реєстрацію їх у сервісі-локаторі. Якщо відкриття боксів не вдалося, виконується спроба повторного відкриття або видаляються бокси з диску.