

How to Use this Template

1. Create a new document, and copy and paste the text from this template into your new document [Select All → Copy → Paste into new document]
 2. Name your document file: “**Capstone_Stage1**”
 3. Replace the text in green
-

[Description](#)

[Intended User](#)

[Features](#)

[User Interface Mocks](#)

[Screen 1](#)

[Screen 2](#)

[Key Considerations](#)

[How will your app handle data persistence?](#)

[Describe any corner cases in the UX.](#)

[Describe any libraries you'll be using and share your reasoning for including them.](#)

[Describe how you will implement Google Play Services.](#)

[Next Steps: Required Tasks](#)

[Task 1: Project Setup](#)

[Task 2: Implement UI for Each Activity and Fragment](#)

[Task 3: Your Next Task](#)

[Task 4: Your Next Task](#)

[Task 5: Your Next Task](#)

GitHub Username: ChrisMeyering

Trending Movies

Description

Are you tired of having to google “recent popular movies” or “best horror movies” over and over again? Powered by TMDB (The Movie Database) API, Trending Movies will allow you to have quick access to the most popular, top rated, most recent movies and more with just a few taps! If you are just curious as to which movies you might want to watch within the next few weeks, simply browse through your favorite genres and save the ones you want to watch to a list!

Intended User

This application is open to any age range. Recommended for teens and older, and movie enthusiasts in general!

Features

Uses internet:

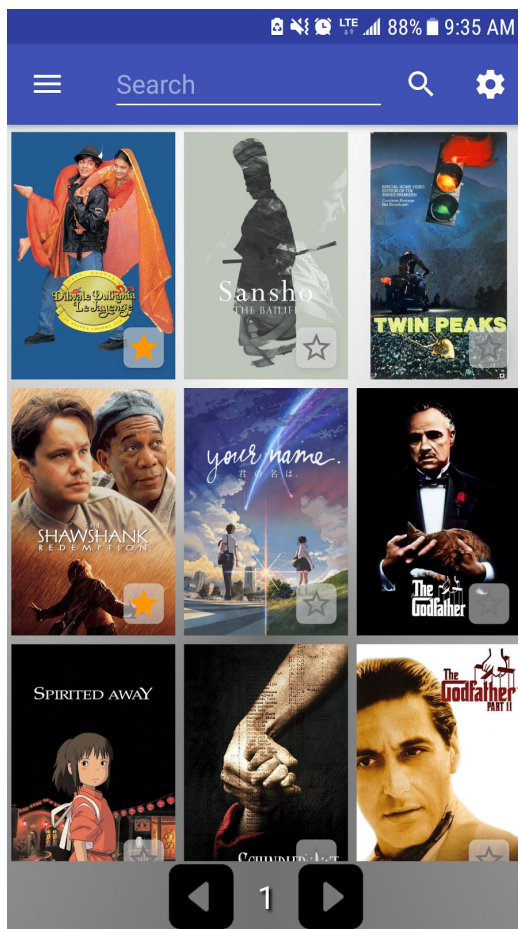
- Queries TMDb's API to keep an up-to-date list of movies to browse through
- Queries youtube to load trailers when available

Stores information:

- In order to reduce data usage, some information such as recently viewed movies and favorites will be saved to a local database through a Content Provider
- User can save select movies to a list of favorite movies
- Shared Preferences option to let user reduce network usage. If enabled, new query responses will be saved to a local database that the user can browse through (TODO: decide if posters should be saved or simply ID and title).

User Interface Mocks

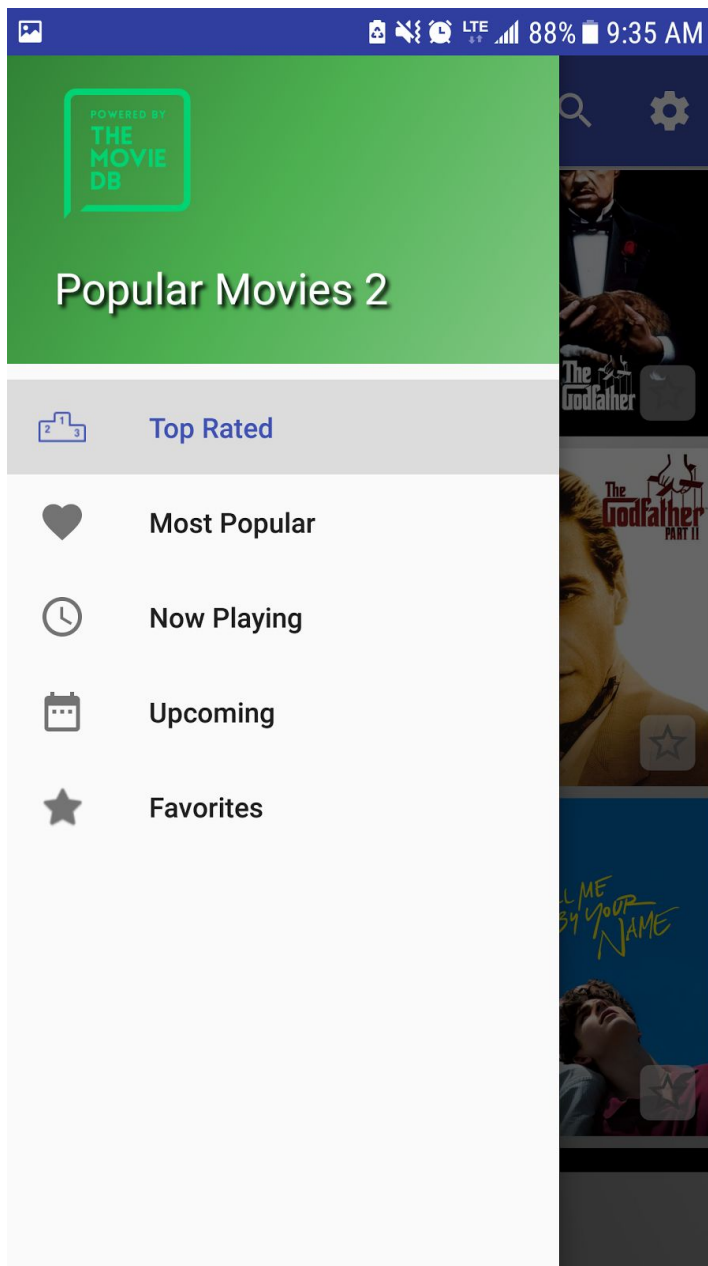
Small device -- Screen 1



This screen is the default layout for MainActivity. I intend to replace the space at the bottom (arrows & page number) with a "Load more" button that will scroll onto screen once the user reaches the bottom of the movie list.

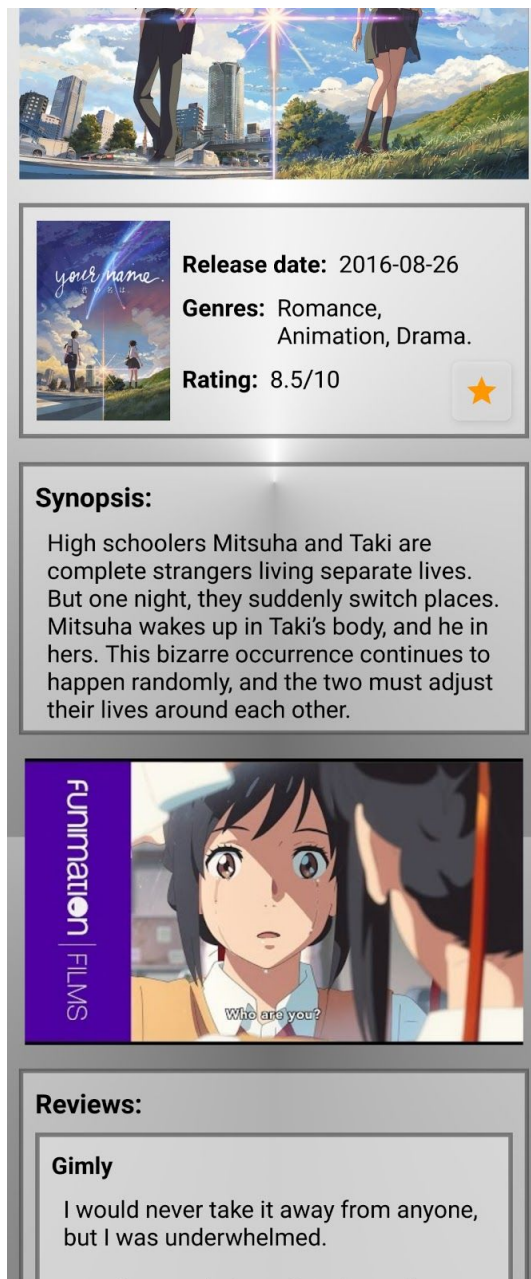
This activity allows user to search movies by name (search bar at the top), change the sorting preference (see next screenshot) and add movies to favorites list.

Small device -- Screen 2



This screen demonstrates a preview of the DrawerLayout I intend to use in order to let the user change the sorting criteria as well as browse favorite and recent movies (recent feature not yet implemented). I also intend to move the settings icon on the toolbar to the drawerlayout, or remove it entirely if I don't find a good use for it (possibly allow user to change color scheme? Might not because it is an unnecessary feature and will not bring much to the app).

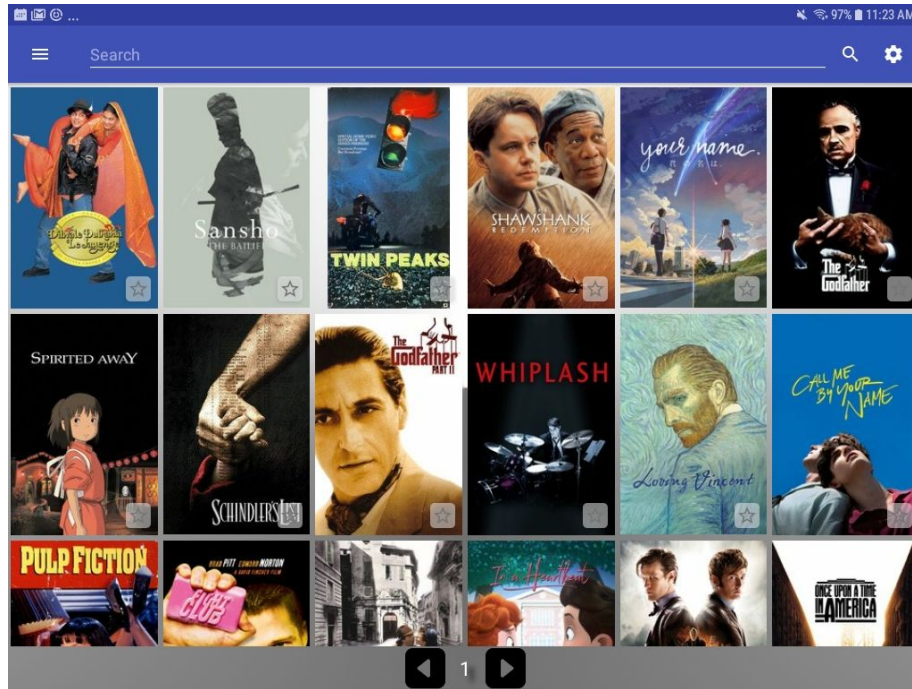
Small device -- Screen 3



This screen is the Detail screen. It will contain all the information about a specific movie. The title, backdrop image and favorite button (FAB?) will be inside of a CollapsingToolbarLayout. The NestedScrollView will contain everything else, including the movie poster, release date, rating, genres, synopsis, trailers and reviews.

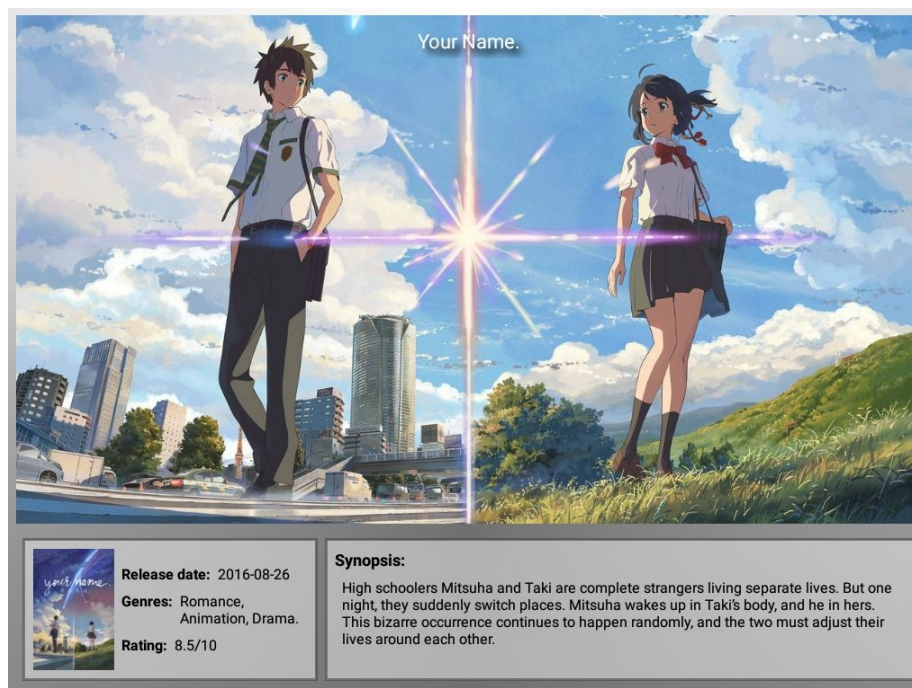
The large device photos below are very similar to their small device equivalents. I intend to make small changes to the layouts (such as the favorite button and the detail screen color scheme).

Large device -- Screen 1




Large screens have more videos per row.

Large device -- Screen 2



The detail activity is very similar to that of smaller screens. The only difference is that the details and the synopsis are next to each other instead of above.

Large device -- Screen 3





Release date: 2016-08-26
Genres: Romance, Animation, Drama.
Rating: 8.5/10

Synopsis:

High schoolers Mitsuha and Taki are complete strangers living separate lives. But one night, they suddenly switch places. Mitsuha wakes up in Taki's body, and he in hers. This bizarre occurrence continues to happen randomly, and the two must adjust their lives around each other.

Your Name. - Official English Dub Trailer



0:00 01:46 YouTube

Reviews:

Gimly

I would never take it away from anyone, but I was underwhelmed.

Final rating:★★ - Definitely not for me, but I sort of get the appeal.

BecauseImBatman

It has beautiful animation and beautiful characters. It is a funny, sweet and emotional roller coaster of a crowd-pleaser that manages to win your heart.

Reno

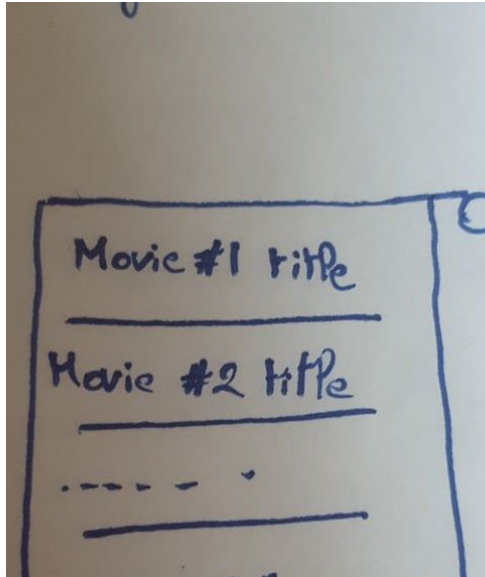
****They're no strangers, yet they've never met.****

Probably the most anticipated anime since the final film of Mr. Miyazaki. Because people know what these guys are capable of. This is one per cent sci-fi and 99 per cent fantasy. The concept is not new for us. From 'Freaky Friday' to 'The Lake House', even the recent American television film 'The Swap', you will remember a handful of names once you read this film's synopsis. Yet the film was so good, because of how well the theme was utilised.

We all know the technical brilliance of '5 Centimeters Per Second' and the beautiful romance from 'The Garden of Words'. This is an excellent mix of those two. If love either of them, or both, then you will love it as well. But if you ask me, my favourite is still 'The Garden of Words' from this director. I liked this film, enjoyed thoroughly, so liking it less than that means not a bad flick. Surely a film to recommend for anime fans, as well as for animation's (western audience).

The story of two teenagers, one from the quiet, forested mountain region and the other one from Tokyo. The plot revolves around a

Widget mock -- FavoritesWidget (min dimensions = 2x1)



When a specific movie in the widget is clicked on, the selected movie's detail page will open up.

Key Considerations

How will your app handle data persistence?

I intend to use

- Bundles (onSaveInstanceState) to ensure relevant data is preserved throughout the application's lifecycle.
- SharedPreferences to store user's customizations and preferences as well as any key-value pair variables
- A Content Provider to handle data persistence throughout this application
- A Job Dispatcher to schedule periodical updates of local data

Describe any edge or corner cases in the UX.

- I intend to animate transitions between activities and/or fragments using shared element transitions.
- If device is rotated or media is interrupted while user is watching a trailer, the video will resume where it left off in the same play/pause state.

Describe any libraries you'll be using and share your reasoning for including them.

Library	Reason
com.squareup.picasso:picasso	Load posters and backdrops into ImageViews from image URL
com.android.support:recyclerview	RecyclerView to efficiently display scrollable lists of movie posters/trailers of arbitrary length
com.android.support:design	FloatingActionButton CollapsingToolbarLayout/CoordinatorLayout NavigationDrawer
com.android.support.constraint:constraint-layout	ConstraintLayout to easily position items relative to one another
com.android.support:appcompat	Enable backwards compatibility to support devices running on older versions of Android
com.github.bumptech.glide:glide	Facilitate extraction of dominant and muted colors from an image.
com.google.android.gms:play-services-cast	Enable user to cast trailers to TV or other applicable receiver.
com.google.android.gms:play-services-ads	Free flavor only. 33% probability of showing an ad when transitioning to the details screen of a movie.

Describe how you will implement Google Play Services or other external services.

- I will use an IntentService to fetch data. If the database that is supposed to contain desired information (eg: movies sorted by most popular, most recent, etc...) is empty or non-existent, start the IntentService the IntentService will start a background task to query TMDB and populate the database with the response. All of these operations will be handled by a loader to ensure that everything loads smoothly in the background.
- I will use a Firebase JobDispatcher to periodically verify (and update if necessary) that information in databases is up-to-date -- these updates will only run on unmetered networks.

Next Steps: Required Tasks

Task 1: Project Setup

- Configure libraries

- Setup connection to TMDB
- Create databases

Task 2: Project Sync and Data Persistence

- Build Content Provider for databases (RecentIds, FavoriteIds, AllMovies, PopularIds, TrailerUrls)
- Make sure Content Provider handles all database operations correctly by implementing a Loader (CursorLoader) and verifying that the data is as expected.

Task 3: TMDB Queries

- Write a class containing TMDB data (base urls and extensions, keywords etc)
- Write functions to handle all desired query options (search by name or genre, most popular, by date, etc..)
- Translate JSON responses into POJO
- Implement all appropriate loaders

Task 3: Implement UI for Main Activity and its Fragments

- Build UI for MainActivity (tablet, phone, portrait and landscape (TV?))
- Build UI for MovieList Fragment
- Build UI for DrawerLayout (sort options, TMDB logo)

Task 4: Implement UI for Detail Activity and its Fragments

- Add CollapsingToolbarLayout with a FloatingActionButton
- Build UI for Trailers and Reviews Fragments
- Use ExoPlayer or YoutubePlayer? Setup MediaSession for player. Setup cast option for player
- Devices in Landscape:
 - Video player replaced by “Watch Trailers” button

■ Task 4.1: Implement Landscape Trailer Fragment UI -- Add database for trailer favorites?

Task 5: Animate Transitions

- Animate poster image from MainActivity to DetailsActivity when movie is selected
- Animate all other posters to slide off the screen

Task 6: Add Paid and Free Flavors

- Setup Google Play services for advertising in the free flavor (gradle: freeCompile)
- Create different layouts for both flavors