[Descrição 1](#_Toc534829046)

[Perfil de Usuário 1](#_Toc534829047)

[Características 2](#_Toc534829048)

[UI Mocks 2](#_Toc534829049)

[**Tela de login** 2](#_Toc534829050)

[**Menu** 2](#_Toc534829051)

[**Tela Principal** 3](#_Toc534829052)

[**Tela de detalhes** 3](#_Toc534829053)

[**Widget** 3](#_Toc534829054)

[Considerações 4](#_Toc534829055)

[**Como será tratada a persistência de dados?** 4](#_Toc534829056)

[**Condições de contorno para UX.** 4](#_Toc534829057)

[**Bibliotecas utilizadas e suas funções.** 4](#_Toc534829058)

[**Implementação dos serviços utilizados.** 4](#_Toc534829059)

[Próximos passos: Tarefas requeridas 5](#_Toc534829060)

[**Tarefa 1: Iniciando o projeto** 5](#_Toc534829061)

[**Tarefa 2: Implementando a UI** 5](#_Toc534829062)

[**Tarefa 3: Comunicação** 5](#_Toc534829063)

[**Tarefa 4: Persistência de dados** 5](#_Toc534829064)

[**Tarefa 5: Acessibilidade** 6](#_Toc534829065)

[**Tarefa 6: Widget** 6](#_Toc534829066)

**GitHub Username**: AlanMunhoz

TMSearch

Descrição

O TMSearch é um aplicativo desenvolvido inteiramente na linguagem Java que permite a rápida visualização do lançamento de filmes e seus detalhes como avaliação, popularidade, entre outros. Sua interface simplificada permite favoritar filmes, compartilhar títulos e assistir a traillers.

Perfil de Usuário

Para todos aqueles apaixonados por filmes e que desejam estar sempre atualizados a respeito dos mesmos, este aplicativo traz diversas informações relevantes dentro de uma interface simples e agradável.

Características

* Side Menu com informações de login e entrada para as principais funções do aplicativo
* Viasualização de filmes mais populares
* Visualização de filmes mais bem avaliados
* Visualização de filmes em cartaz
* Visualização de filmes em lançamento
* Visualização de filmes favoritados
* Compartilhamento de filmes
* Visualização de traillers via intent implícita

UI Mocks

**Tela de login**

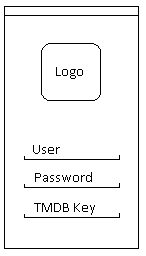


Figura 1: Login

A Figura 1 fará o login do usuário no Firebase Authentication, além disso, será utilizada para capturar a chave da API TMDB.

**Menu**



Figura 2: Side menu

A Figura 2 ilustra as opções disponíveis no menu, o ícone e a descrição para cada linha são descritos acima.

**Tela Principal**

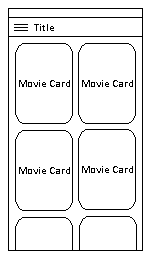


Figura 3: Tela Principal

Na Figura 3 temos a lista de filmes, onde a categoria será exibida no título da barra.

**Tela de detalhes**

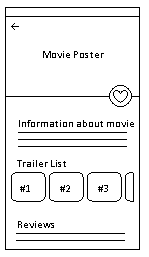


Figura 4: Detalhes

Na Figura 4 as informações referentes ao filme estarão sendo exibidas, dentre elas a imagem do poster, popularidade, avaliação entre outras. Há ainda a lista de trailers, que através de uma intent implícita chama uma aplicação para reproduzir o vídeo.

A *AppBar* exibe o poster do filme, sendo que a mesma será colapsada dando lugar a *Toolbar*.

**Widget**

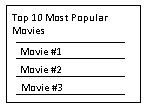


Figura 5: Widget

A Figura 5 mostra o widget, ele mostrará a lista dos 10 filmes mais populares.

Considerações

**Como será tratada a persistência de dados?**

A persistência de dados se dará aos filmes favoritados pelo usuário. Haverá ainda a persistência dos dados do usuário ao fazer login.

O Room será utilizado com Live Data e View Model. A estrutura será de duas tabelas (login e favoritos), a primeira respectivamente conterá os dados (login e senha), a segunda conterá os dados do filme favoritado.

**Condições de contorno para UX.**

No caso do usuário estar na tela principal e o botão retornar (*back button*) seja pressionado, nada ocorrerá, visto que neste caso o aplicativo finalizaria sua execução.

Na tela de login, será utilizada uma *Progress Bar* enquanto o login é efetuado, neste caso, ficará visível até que o serviço gere uma resposta ou então que o timeout de 10s seja atingido. Em caso de login inválido, uma mensagem via *toast* será exibida, onde os campos serão apagados e o que gerou o erro será destacado.

Em relação à conexão via internet, será utilizado um *broadcast receiver* para identificação do *status* da mesma. Neste caso, ocorrendo perda de conexão será exibida uma Snackbar indeterminada para o usuário até que a mesma se estabeleça novamente. Qualquer interação que necessite de comunicação não ocorrerá, sendo exibido um *toast* para lembrar o usuário da situação. A *Snackbar* se aplicará a todas as telas do aplicativo.

O layout em paisagem da tela principal será feito acrescentando-se mais duas colunas para os *cards* dos filmes, sendo duas colunas para orientação retrado. Neste caso não será feito dois layouts, mas sim o número de colunas será definido na pasta “values/dimens” para a orientação *landscape*.

O layout da tela de login ficará disponível apenas na orientação retrato, as demais terão a orientação paisagem.

**Bibliotecas utilizadas e suas funções.**

A exibição de imagens é de vital importância, visto a naturezado aplicativo. Desta forma, a biblioteca Picasso será utilizada para download e exibição das imagens.

Há ainda a necessidade da comunicação com a API The Movie DB, sendo que nesta, a biblioteca Retrofit será utilizada.

**Implementação dos serviços utilizados.**

Serão utilizados dois serviços, o Firebase Authentication e o Firebase Analytics.

O Firebase Authentication faz o cadastro e verificação do usuário por meio de e-mail e senha.

Já o Google Analytics envia logs de eventos importantes no sistema, como por exemplo a exibição de traillers e a interação de usuário ao compartilhar e favoritar.

Próximos passos: Tarefas requeridas

O Tema do aplicativo estenderá da AppCompat, usando transição simples entre atividades.

Todas as *strings* do aplicativos serão mantidas no arquivo *strings.xml*. Sendo que haverá suporte a alternância de layout RTL em todos os layouts.

O aplicativo estará equipado com uma assinatura, contida no repositório do projeto.

**Tarefa 1: Iniciando o projeto**

Primeiramente será criado um projeto partindo do layout básico de um side menu. Nele, o menu deve conter as opções citadas nos Mocks acima (Figura 2). Em seguida serão adicionadas as bibliotecas que o projeto irá utilizar.

Alguns passos posteriores:

* Configuração do Firebase Authentication
* Configuração do Firebase Analytics
* Suporte à criação de usuários autenticados e a geração de logs pelo Analytics.

**Tarefa 2: Implementando a UI**

Nesta etapa serão criadas duas Activities:

* A activity principal, que será utilizada para carregar os filmes de acordo com as opções do usuário
* A activity de detalhes, que exibirá os detalhes do filme selecionado pelo usuário

**Tarefa 3: Comunicação**

Esta etapa tratará da comunicação e da definição dos *end points* do projeto. A estrutura para comunicação será criada, a qual envolve a utilização da biblioteca Retrofit nas requests, onde a API The Movie DB será utilizada.

Haverá ainda a preocupação das chamadas em background, o que se fará através da *AsyncTask Loader*, não onerando assim a thread da interface gráfica e levando em consideração o ciclo de vida da *Activity*, evitando assim vazamento de memória.

**Tarefa 4: Persistência de dados**

Neste ponto o foco será a persistência de dados, onde os dados de login e os dados dos filmes favoritos serão salvos utilizando Room. LiveData e View Model serão utilizados para atualização eficiente do banco de dados.

Ao mover os dados da persistência para a visualização serão utilizados executors, onde se garantirá a execução em *thread background*.

**Tarefa 5: Acessibilidade**

O aplicativo tratará da acessibilidade por preencher devidamente o atributo *contentDescription* para *ImageViews*.

**Tarefa 6: Widget**

O widget será configurado e implementado nesta etapa, o mesmo conterá a lista dos 10 filmes mais populares. Ele será contruído com uma ListView como descrito na Figura 5.