FACULDADE DE TECNOLOGIA DE AMERICANA "MINISTRO RALPH BIASI"

Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas

Cesar Gabriel Araújo Lourenço Rodrigues

Isaque Torres da Silva

Mateus Matos Marquizeti

Rodrigo Vinícius Ventura de Souza

Movielub: A rede social de filmes

Americana, SP

2021

FACULDADE DE TECNOLOGIA DE AMERICANA "MINISTRO RALPH BIASI"
Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas

Cesar Gabriel Araújo Lourenço Rodrigues
Isaque Torres da Silva
Mateus Matos Marquizeti
Rodrigo Vinícius Ventura de Souza

Movielub: A rede social de filmes

Projeto de Conclusão da disciplina Trabalho de Graduação do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, sob a orientação do Prof. Dr. Kleber de Oliveira Andrade.

Área de concentração: Engenharia de Software

Americana, SP
2021

FICHA CATALOGRÁFICA – Biblioteca Fatec Americana - CEETEPS
Dados Internacionais de Catalogação-na-fonte

R612m RODRIGUES, César Gabriel Araújo Lourenço

Movielub: a rede social de filmes. / César Gabriel Araújo Lourenço Rodrigues, Isaque Torres da Silva, Mateus Matos Marquizeti, Rodrigo Vinicius Ventura de Souza. – Americana, 2021.
50f.

Monografia (Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas) - - Faculdade de Tecnologia de Americana – Centro Estadual de Educação Tecnológica Paula Souza

Orientadores: Prof. Dr. Kleber de Oliveira Andrade

1 Desenvolvimento de software I. SILVA, Isaque Torres da II. MARQUIZETI, Mateus Matos III. SOUZA, Rodrigo Vinicius Ventura de IV. ANDRADE, Kleber de Oliveira V. Centro Estadual de Educação Tecnológica Paula Souza – Faculdade de Tecnologia de Americana

CDU: 681.3.05

Cesar Gabriel Araújo Lourenço Rodrigues

Isaque Torres da Silva

Mateus Matos Marquizeti

Rodrigo Vinícius Ventura de Souza

Movielub: A rede social de filmes

Trabalho de graduação apresentado como exigência parcial para obtenção do título de Tecnólogo/Bacharel em Análise e Desenvolvimento de Sistemas pelo CEETEPS/Faculdade de Tecnologia – FATEC de Americana.

Área de concentração: Laboratório de Engenharia de Software.

Americana, 07 de dezembro de 2021.

Banca Examinadora:

Kleber de Oliveira Andrade

Doutor

Faculdade de Tecnologia – FATEC de Americana

Eduardo Antônio Vicentini

Mestre

Faculdade de Tecnologia – FATEC de Americana

Wagner Siqueira Cavalcante

Mestre

Faculdade de Tecnologia – FATEC de Americana

Resumo

No momento atual, foi notado que as pessoas se tornaram mais ansiosas e impacientes, assim, demorando horas para escolher um filme em plataformas de *streaming*, até mesmo deixando de assisti-lo pelo simples fato de não achar um título de seu agrado naquele momento. Por esses e outros motivos, o Movielub foi desenvolvido para auxiliar nesses momentos, uma aplicação *web*, com portabilidade para quase todos os dispositivos *mobiles*, e que fará o uso de um sistema de recomendações de filmes baseado no gosto pessoal do usuário, possibilitando a visualização do catálogo de favoritos de outros usuários, tornando-se uma rede social focada nos apaixonados por cinema e mídia audiovisual. O trabalho apresenta, primeiramente, o planejamento do projeto, e em seguida, o processo de desenvolvimento do aplicativo utilizando diversos recursos, como uma base de dados para sempre estar atualizado sobre as novidades do cinema, um sistema de autenticação, login de usuário, entre outros.

Palavras-Chave: Aplicativos, engenharia de *software*, *mobile*.

Sumário

Lista de Imagens

Lista de Tabelas

1. Introdução

A evolução exponencial da tecnologia, como um todo, mostra que nada é impossível para o ser humano. O que diria um cliente da Blockbuster[1] em 1995, se falasse a ele que é possível assistir filmes e séries a qualquer momento, em qualquer lugar, sem precisar alugar ou se preocupar com a devolução, ou a um telespectador de televisão, que seria acessível a qualquer pessoa, ter um catálogo variado e extenso, isento de propagandas e anúncios, sem se preocupar em que dia, ou em que hora um determinado filme vai ser transmitido ao vivo, ou por uma emissora.

Blockbuster nasceu em 1985, em Dallas, tendo como fundador David Cook. O cenário na época era promissor devido à grande circulação de videocassetes. Com isso, em pouco tempo a Blockbuster foi tomando espaço no mercado, em 1994, foi adquirida pela gigante Viacom[2] por US $8,4 bilhões, atingindo a marca de 4.500 lojas abertas. Após 6 anos, em 1999, a companhia foi eleita a 13° marca mais conhecida dos Estados Unidos (BLOCKBUSTER, 2020).

E é em 1997 que, Marc Randolph, Reed Hastings e Mitch Lowe, criam a famosa Netflix, na época não como uma plataforma de *streaming*, mas como serviço de entrega de DVDs pelos correios, mas esse cenário mudou em 2007, quando a Netflix, anuncia a sua plataforma de *streaming* (NETFLIX, 2017). Em contrapartida, a Blockbuster teve uma desvalorização no mercado.

Essa realidade não para somente em Blockbuster e Netflix, o mercado de *streaming*[3] está evoluindo e crescendo rapidamente, com isso deixando o modo de rede de televisão, como Rede Globo, SBT entre outras, obsoleto. Isso é visível no caso Rede Globo, que há pouco tempo anunciou o GloboPlay[4], a plataforma de *streaming* da rede de televisão Rede Globo, onde é possível encontrar séries originais, filmes, programas, jornais e entre outros (GLOBOPLAY, 2020).

[1] Blockbuster para mais acesse: < https://www.startse.com/noticia/startups/falencia-blockbuster-netflix >

[2] Viacom ou Video and Audio Communications foi um conglomerado de mídia estadunidense com várias empresas em todo o mundo de redes de televisão por assinatura e indústria cinematográfica

[3] Streaming é uma tecnologia de transmissão simultânea de dados de áudio e vídeo por meio da rede. Através do streaming não há necessidade de fazer o download do conteúdo, pois, na medida que, o software de streaming recebe os dados, eles são arquivados temporariamente na máquina e transmitidos ao usuário

[4] GloboPlay acesse: < https://globoplay.globo.com/ >

1.1 Objetivo

Esse trabalho tem como objetivo desenvolver um aplicativo para gerenciamento de filmes assistidos ou não pelo usuário, criar listas de filmes que pretende assistir, onde será possível compartilhar suas opiniões sobre os títulos assistidos, e indicá-los para seus seguidores, uma vez que o usuário poderá seguir e ser seguido por eles. E durante o processo de desenvolvimento, apresentar a aplicação da metodologia ágil SCRUM.

2. Projeto do Sistema

Segundo Gonçalves & Junior (2013, p. 9), o desenvolvimento de um *software* requer que sejam seguidas uma sequência de etapas que caracterizam o seu ciclo de vida a partir da aplicação de certa metodologia de desenvolvimento. Cada etapa da metodologia adotada, devolve parte do desenvolvimento do *software* em si, desde a sua concepção inicial até o produto, seguindo em uma linha geral, um fluxo de desenvolvimento baseado nas etapas de concepção, análise, projeto, implementação, teste, implantação e manutenção, onde cada etapa é uma continuação da anterior (AMADEU, GONÇALVES & TEIXEIRA JUNIOR, 2013, p. 11-12).

De acordo com ISO/ IEC/IEEE 12207 (2017, p. 3), entende-se como ciclo de vida a "estrutura contendo processos, atividade e tarefas envolvidas no desenvolvimento, operação e manutenção de um produto de *software*, abrangendo a vida do sistema desde a definição de seus requisitos até o término de seu uso".

2.1 Cronograma de Atividades

Para melhor organização das atividades, elas foram divididas em duas fases: A primeira delas, o planejamento, possui os seguintes itens a serem desenvolvidos:

- Planejamento das atividades utilizando a metodologia ágil SCRUM;
- Arquitetura do projeto;
- Modelagem do banco de dados.

E a segunda fase, que consiste em desenvolver o projeto da aplicação conforme o planejamento inicial, que possui os seguintes itens:

- Cadastro de usuário;
- Autenticação de usuário;
- Listagem de lançamentos de filmes;
- Recurso de pesquisa de filmes;
- Exibição das informações sobre o filme selecionado;
- Tela de filmes para assistir;
- Tela de filmes assistidos.

2.2 Metodologia de Desenvolvimento

De acordo com a definição de SCRUM realizada pelo site Desenvolvimento Ágil, no ambiente atual de desenvolvimento, os requisitos estão sujeitos a mudanças constantes, o que pode tornar a organização do processo de desenvolvimento complicada, tendo isso em mente, foi optado pela utilização da metodologia ágil SCRUM.

O SCRUM pode ser aplicado em quaisquer projetos, independentemente do tamanho. Ele possui a função de facilitar a organização do desenvolvimento em projetos onde os requisitos estão em mudanças frequentes, visando aumentar a produtividade.

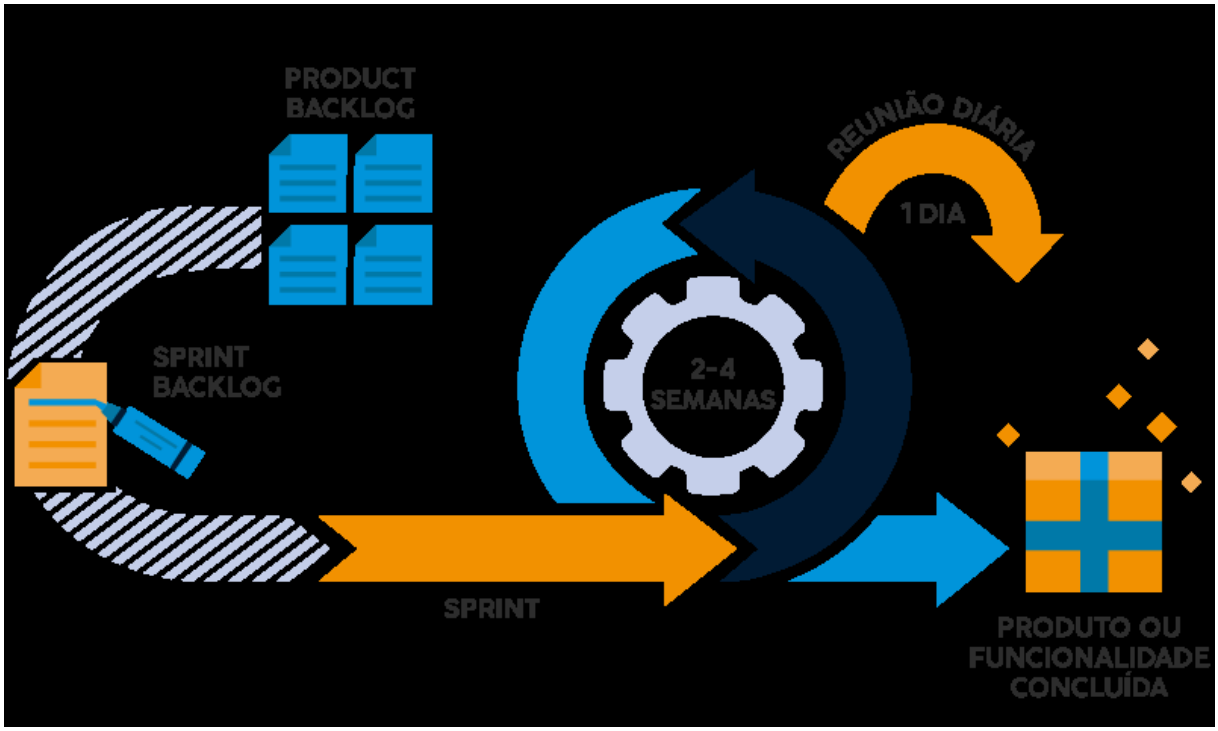
Os projetos são divididos em ciclos (tipicamente mensais), chamados de *Sprints*. O *Sprint* representa um *Time Box* (intervalo de tempo) dentro do qual um conjunto de atividades deve ser executado. Metodologias ágeis de desenvolvimento de *software* são iterativas, ou seja, o trabalho é dividido em iterações, que são chamadas de *Sprints* no caso do Scrum.

As funcionalidades a serem implementadas em um projeto são mantidas em uma lista que é conhecida como *Product Backlog*. No início de cada Sprint, faz-se um *Sprint Planning Meeting*, ou seja, uma reunião de planejamento na qual o *Product Owner* prioriza os itens do *Product Backlog* e a equipe seleciona as atividades que ela será capaz de implementar durante o *Sprint* que se inicia. As tarefas alocadas em um *Sprint* são transferidas do *Product Backlog* para o *Sprint Backlog*.

Durante o andamento da Sprint, diariamente faz-se uma reunião em que todos os membros da equipe devem participar e compartilhar suas tarefas e o que eles desenvolveram dentre as mesmas, para que todo o time consiga se situar em relação ao andamento do projeto.

Ao final de um Sprint, a equipe apresenta as funcionalidades implementadas em uma *Sprint Review Meeting*. Finalmente, faz-se uma *Sprint Retrospective* e a equipe parte para o planejamento do próximo *Sprint,* na figura 1 é possível ver como é o funcionamento do SCRUM.

Figura 1 - Funcionamento do SCRUM



Fonte: Tecnicon, 2020

2.3 Softwares Similares

Atualmente existem inúmeros serviços e aplicativos focados em filmes, séries e diversas outras mídias audiovisuais. Dentre eles. foram selecionados alguns para serem comparados com o projeto desenvolvido, o Movielub, como pode se ver na tabela a seguir.

Tabela 1 - Comparativo de funcionalidades de outros aplicativos em relação ao Movielub

Itens	Netflix	Amazon Prime Video	Cinemaniac	Movielub
Streaming de filmes	X	X		
Indicação de filmes	X	X	X	X
Aluguel ou compra de filmes				
Avaliações do filme	X	X	X	X
Lista de filmes para assistir	X	X	X	X
Indicação de filmes para amigos	X			X

Fonte: Elaborado pelo autor

2.4 Levantamento de Requisitos

A engenharia de requisitos (RE – *Requirements Engineering*) é o processo de descobrir, analisar, documentar e verificar requisitos de um sistema. Um requisito pode ser definido como uma descrição dos serviços fornecidos pelo sistema e as suas restrições operacionais (SOMMERVILLE, 2007). Tradicionalmente, os requisitos são divididos em dois tipos: requisitos funcionais e requisitos não funcionais.

2.4.1 Requisitos Funcionais

“Os requisitos funcionais descrevem o que o sistema deve fazer, isto é, definem a funcionalidade desejada do software” (SOMMERVILLE, 2019). A Tabela 2 apresenta os requisitos funcionais deste projeto.

Tabela 2 - Requisitos funcionais do projeto

Identificação	Requisitos Funcionais	Prioridade
RF1	Efetuar login	Alta
RF2	Efetuar cadastro	Alta
RF3	Efetuar recuperação de senha	Alta
RF4	Envio de email de confirmação de cadastro	Alta
RF5	Envio de email de recuperação de senha	Alta
RF6	Reenviar email de confirmação de cadastro ao realizar login	Alta
RF7	Efetuar cadastro de nova senha	Alta
RF8	Efetuar listagem de lançamentos	Alta
RF9	Pesquisar filmes	Alta
RF10	Exibição de informações do filme	Alta
RF11	Listagem de filmes a serem assistidos	Alta
RF12	Recomendar filmes	Alta

Fonte: Elaborado pelo autor

2.4.2 Requisitos Não Funcionais

“Os requisitos não funcionais são aqueles não diretamente relacionados às funções específicas fornecidas pelo sistema” (SOMMERVILLE, 2019). A Tabela 3 apresenta os requisitos não funcionais deste projeto.

Tabela 3 - Requisitos não funcionais do projeto

Identificação	Requisitos Não Funcionais	Prioridade
NF1	Aplicação *web* responsiva para acesso em quase todos os aparelhos *mobiles*	Alta
NF2	Interface limpa e de fácil entendimento	Alta

Fonte: Elaborado pelo autor

2.5 Recursos e Ferramentas

Essa seção é responsável por contemplar e descrever as ferramentas, programas e conceitos necessários para o desenvolvimento do aplicativo.

A aplicação *web* foi desenvolvida para suportar diferentes *browsers*[5], como Google Chrome, Safari, Opera, Firefox, Edge entre outros, pois o *framework* Angular dá essa possibilidade. Era necessário escolher uma IDE[6] *(Integrated Development Environment*) que desse a possibilidade de reunir a codificação tanto do *Back-End* como o *Front-End*. Foi escolhido o Visual Studio Code[7], juntamente utilizando o Git[8] e GitHub[9] para o versionamento do código, facilitando o desenvolvimento do *Front-End* utilizando o *framework* Angular[10], como já dito anteriormente, e o NodeJs[11] para o desenvolvimento *Back-End*.

Como banco de Dados de filmes foram escolhidos dois provedores, o *The Movie Database* (TMDB[12]) e o IMDb *(Internet Movie Database*), e para os dados dos

[5] Browsers é um navegador de internet, que permite a visualização de conteúdo, como uma página web. Para mais informações <https://conceito.de/browser>. Acesso em: 07/11/2021.

[6] IDE, do inglês Intergrated Development Environment ou Ambiente de Desenvolvimento integrado em português, é um software que auxilia no desenvolvimento de aplicações. Para saber mais acesse <https://www.treinaweb.com.br/blog/o-que-e-uma-ide-ambiente-de-desenvolvimento-integrado>. Acesso em: 07/11/2021.

[7] Visual Studio Code ou VSCode é um editor de código-fonte desenvolvido pela Microsoft. Disponível em: <https://code.visualstudio.com/docs>. Acesso em: 07/11/2021.

[8] Git é um sistema open-source que permite o controle de versionamento de códigos. Disponível em: <https://git-scm.com/docs>. Acesso em: 07/11/2021.

[9] GitHub é uma plataforma de hospedagem de código-fonte e arquivos com controle de versão usando o Git. Disponível em: <https://github.com/>. Acesso em: 07/11/2021.

[10] Angular é uma plataforma de aplicações web de código-fonte aberto e front-end baseado em TypeScript. Disponível em <https://angular.io/docs>. Acesso em: 07/11/2021.

[11] NodeJs é um software de código aberto, multiplataforma, baseado no interpretador V8 do Google e que permite a execução de códigos JavaScript fora de um navegador web. Disponível em <https://nodejs.org/en/docs/>. Acesso em: 07/11/2021.

[12] TMDB ou The Movie Database é uma base de dados contendo informações e detalhes sobre filmes e séries. Disponível em: <https://www.themoviedb.org/>. Acesso em: 07/11/2021.

usuários e outros dados da aplicação serão armazenados em um banco MySQL[13] e para o gerenciamento é utilizado o *MySQL Workbench*.

2.5.1 Angular

O Angular é um framework criado para desenvolver aplicações em diversas plataformas, mantido e desenvolvido pela Google*. Contém um vasto conjunto de bibliotecas poderosas para o desenvolvimento e evolução do aplicativo, possibilitando construir aplicações com uma qualidade e produtividade. Se originou da reescritura completa do antigo AngularJS e foi desenvolvido em TypeScript[14].

Segundo Longe, 2020, em um artigo para o Hostigator, Angular é um *framework* para desenvolver aplicativos web do tipo *single page application* (baseados em uma única página dinâmica). Ele utiliza o tipo de arquitetura MVC (*Model-View-Controller* ou Modelo-Visão-Controle), onde o Modelo gerencia a informação e recebe os comandos do Controle, a Visão é a representação visual da informação, e o Controle responde aos comandos e interage com o Modelo.

2.5.2 Visual Studio Code

O Visual Studio Code, também conhecido como VSCode, é uma IDE (Ambiente de Desenvolvimento Integrado) de código aberto da Microsoft, criada em 2015, para edição de códigos-fonte e desenvolvimento de aplicações. Possui integração com o Git, realce de sintaxe e muitas possibilidades para customização, além de poder realizar depurações e suporte a praticamente todas as linguagens de programação.

Além de ser totalmente gratuito, ainda no segundo semestre do ano do lançamento, durante o evento *Connect()*, o editor foi anunciado como *open-source*, e o código estava disponível no GitHub, o que permitiu que a comunidade técnica contribuísse para seu desenvolviment. Facilita a criação de extensões e novos recursos.

---

[13] MySQL é um sistema de gerenciamento de banco de dados, que utiliza a linguagem SQL como interface. Disponível em <https://www.mysql.com/>. Acesso em: 07/11/2021.

[14] TypeScript é uma linguagem de programação de código aberto desenvolvida pela Microsoft, se diferencia do JavaScript pela sua tipagem que é opcional ao seu antecessor. Disponível em: <https://www.typescriptlang.org/>. Acesso em: 07/11/202.

2.5.3 MySQL Workbench

O MySQL Workbench é uma ferramenta visual de *design* de banco de dados que integra desenvolvimento, administração, *design*, criação e manutenção de SQL em um único ambiente de desenvolvimento integrado para o sistema de anco de dados MySQL.

2.5.4 Git e Github

O Git é um dos sistemas de controle de versão mais utilizados no mundo de desenvolvimento de software. Ele é um projeto de código aberto desenvolvido em 2005 por Linus Torvalds[15], o criador do *kernel* do Linux, permitindo mais flexibilidade no fluxo de trabalho, segurança e desempenho.

Por sua vez o GitHub é uma plataforma de hospedagem de código-fonte e controle de versão, que se utiliza da tecnologia de versionamento Git, além do usuário poder contribuir com outros projetos de código aberto disponibilizados na plataforma.

2.5.5 NodeJs

NodeJs é uma tecnologia *open-source* usada para executar código JavaScript fora do navegador. Com ele, é possível criar e desenvolver aplicativos gerais da web, de sites a APIs e microsserviços. Isso se deve à combinação do ambiente de execução JavaScript fornecido pelo próprio Node.js e a interpretação e mecanismo de execução chamado V8* que existe no Google Chrome.

De acordo com André Brasil (Brasil, 2020), em seu artigo no King Host, “O Node.js é uma tecnologia, uma plataforma que utiliza o JavaScript como sintaxe. Através dele, é possível desenvolver pequenas e grandes aplicações. É de código aberto e possui uma ampla comunidade. O Node utiliza o NPM como gerenciador de pacotes e bibliotecas, que por sua vez é o maior ecossistema de bibliotecas *open source* do mundo”.

[15] Linus Benedict Torvalds é um engenheiro de software, nascido na Finlândia. Para saber mais acesse <https://www.techtudo.com.br/tudo-sobre/linus-torvalds.html>

2.5.6 Monday

Monday é um sistema operacional eficaz que permite gerenciar projetos e tarefas por meio de assistência de equipe, colaboração e prazos. Essa ferramenta é especialmente útil para organizações, que podem criar aplicativos de fluxo de trabalho personalizados para realizar planos ou tarefas diárias, onde também é possível aplicar várias metodologias, inclusive a metodologia ágil SCRUM[16].

Este produto foi originalmente usado como uma ferramenta interna da empresa israelense Wix[17,] e depois de passar por diversas atualizações, hoje se encontra bastante completa. Ela pode fornecer serviços para várias operações de negócios, como atividades de marketing, vendas, TI, roteiros de produtos, pesquisa e desenvolvimento, suporte ao cliente, planejamento e processos de negócios, recursos humanos, gerenciamento de eventos e até mesmo produção de mídia.

A segunda fase pode ser modificada para que mais itens sejam adicionados futuramente.

[16] SCRUM é uma metodologia de trabalho que facilita o desenvolvimento em equipe e a gestão dos projetos. Para saber mais acesse o site: <https://www.treasy.com.br/blog/scrum/>. Acesso em: 07/11/2021.

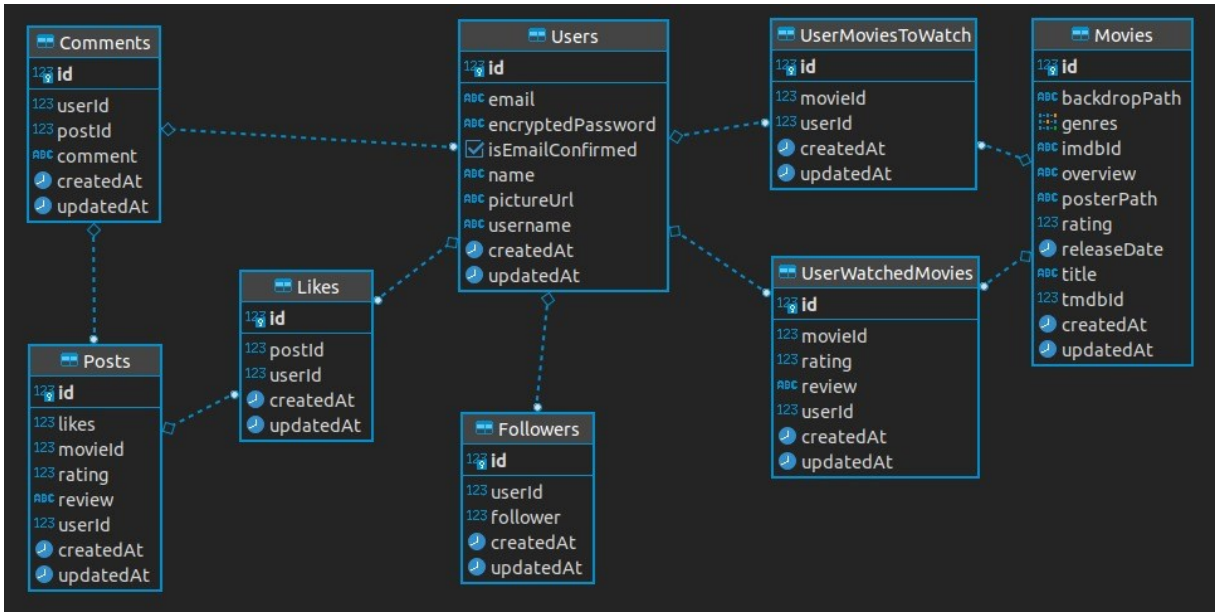
[17] Wix consiste em uma plataforma online de criação e edição de sites. Disponível em: <https://pt.wix.com/about/us>. Acesso em: 07/11/2021.

3. Modelagem do Banco de Dado

3.1 Diagrama de Entidade e Relacionamento

Diagrama Entidade Relacionamento (DER) é um modelo diagramático que descreve o modelo de dados de um sistema com alto nível de abstração. Ele é a principal representação do Modelo de Entidades e Relacionamentos. Sua maior aplicação é visualizar o relacionamento entre tabelas de um banco de dados, no qual as relações são construídas através da associação de um ou mais atributos destas tabelas (SOMMERVILLE, 2011). A Figura 2 apresenta o DER do sistema proposto.

Figura 2 - Diagrama de Entidade e Relacionamento do aplicativo Movielub



Fonte: Elaborado pelo autor

3.1.1 Dicionário de Dados

O Dicionário de Dados (DD) consiste numa lista organizada de todos os elementos de dados que são pertinentes ao sistema. As tabelas devem conter os seguintes campos: Entidade, Atributo, Classe, Domínio, Tamanho e Descrição. As tabelas de 4 a 11 apresentam as entidades do banco de dados do Movielub. A tabela 4 um exibe o Dicionário de dados referente ao Usuário.

Tabela 4 - Dicionário de Dados da Entidade *User*

Users				
Atributo	**Classe**	**Domínio**	**Tamanho**	**Descrição**
id	Determinante	Número	-	Identificador do usuário
name	Simples	Texto	50	Nome do usuário
username	Multivalorado	Texto	50	Nome do usuário dentro da plataforma
email	Simples	Texto	150	Endereço de email
encrypted_password	Simples	Texto	150	Senha criptografada
isEmailConfirmed	Simples	Booleana	-	Indica se o email foi confirmado
pictureUrl	Simples	Texto		Endereço da foto de perfil do usuário
created_at	Simples	Data	-	Data de criação
updated_at	Simples	Data	-	Data de alteração

Fonte: Elaborado pelo autor

A tabela 5 mostra o Dicionário de dados da tabela de Comentários.

Tabela 5 - Dicionário de Dados da Entidade *Comment*

Comments				
Atributo	**Classe**	**Domínio**	**Tamanho**	**Descrição**
id	Determinante	Número	-	Identificador do Comentário
postId	Estrangeira	Número	-	Identificador da publicação
userId	Estrangeira	Número	-	Identificador do usuário

Fonte: Elaborado pelo autor

A Tabela 6 mostra o Dicionário de dados da tabela de Seguidores.

Tabela 6 - Dicionário de Dados da Entidade *Follower*

Followers				
Atributo	**Classe**	**Domínio**	**Tamanho**	**Descrição**
id	Determinante	Número	-	Identificador da tabela seguidores
follower	Simples	Número	-	Quantidade de seguidores
userId	Estrangeira	Número	-	Identificador do usuário

Fonte: Elaborado pelo autor

A Tabela 7 mostra o Dicionário de dados da tabela de *Likes*.

Tabela 7 - Dicionário de Dados da Entidade *Like*

Likes				
Atributo	**Classe**	**Domínio**	**Tamanho**	**Descrição**
id	Determinante	Número	-	Identificador da tabela curtidas
post_id	Estrangeira	Número	-	Identificador da publicação
userId	Estrangeira	Número	-	Identificador do usuário

Fonte: Elaborado pelo autor

A Tabela 8 mostra o Dicionário de dados da tabela de filmes a serem assistidos.

Tabela 8 - Dicionário de Dados da Entidade *MovieToWatch*

MovieToWatch				
Atributo	**Classe**	**Domínio**	**Tamanho**	**Descrição**
id	Determinante	Número	-	Identificador da tabela
user_id	Estrangeira	Número	-	Identificador do usuário
movie_id	Estrangeira	Número	-	Identificador do filme

Fonte: Elaborado pelo autor

A Tabela 9 mostra o Dicionário de dados da tabela de Posts do usuário.

Tabela 9 - Dicionário de Dados da Entidade *Posts*

Posts				
Atributo	**Classe**	**Domínio**	**Tamanho**	**Descrição**
id	Determinante	Número	-	Identificador da publicação
likes	Simples	Número	-	Quantidade de curtidas
movie_id	Estrangeira	Número	-	Identificador do filme
rating	Simples	Número	-	Avaliação da publicação
review	Simples	Texto	300	Avaliação descritiva da publicação
user_id	Estrangeira	Número	-	Identificador do usuário

Fonte: Elaborado pelo autor

A Tabela 10 mostra o Dicionário de dados da tabela de filmes assistidos.

Tabela 10 - Dicionário de Dados da Entidade *WatchedMovie*

WatchedMovies				
Atributo	**Classe**	**Domínio**	**Tamanho**	**Descrição**
id	Determinante	Número	-	Identificador da publicação
movie_id	Estrangeira	Número	-	Identificador do filme
rating	Simples	Número	-	Avaliação da publicação
review	Simples	Texto	300	Avaliação descritiva da publicação
user_id	Estrangeira	Número	-	Identificador do usuário

Fonte: Elaborado pelo autor

E por último, a Tabela 7 mostra o Dicionário de dados da tabela de *Movies*.

Tabela 11 - Dicionário de Dados da Entidade *Movies*

Movies				
Atributo	**Classe**	**Domínio**	**Tamanho**	**Descrição**
id	Determinante	Número	-	Identificador do filme
title	Simples	Texto	100	Título do filme
rating	Simples	Número	-	Avaliação do filme
releaseDate	Simples	Data	-	Data de lançamento
posterPath	Simples	Texto	300	Caminho da imagem do poster
overview	Simples	Texto	300	Prévia
genres	Simples	Matriz de Texto	100	Gênero do filme
tmdbId	Simples	Número	-	Identificador do filme no provedor de dados TMDB (The Movie Database)
imdbId	Simples	Texto	-	Identificador do filme no provedor de dados IMDb (Internet Movie Database)
backdropPath	Simples	Texto	-	Caminho da imagem de fundo

Fonte: Elaborado pelo autor

3.2 Caso de Uso

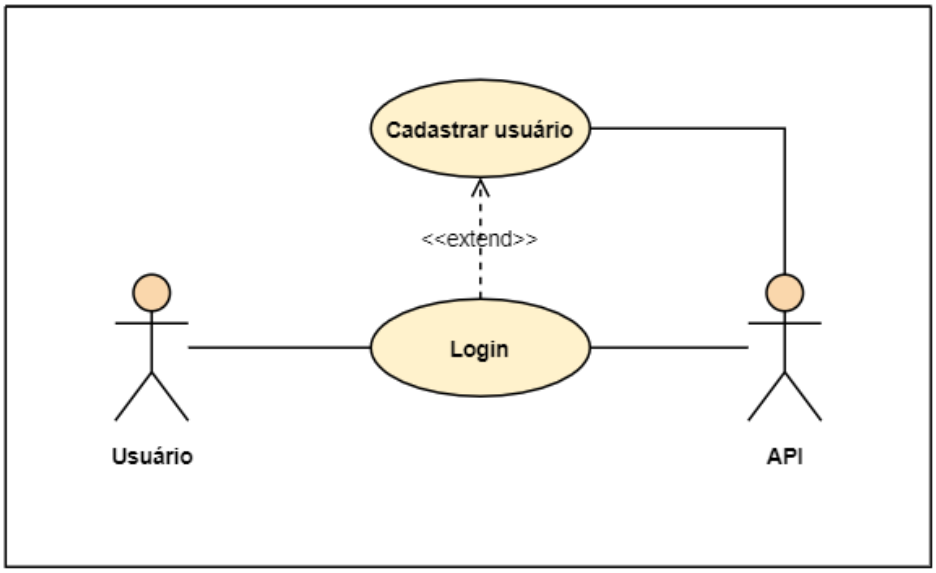
Diagramas de Caso de uso servem para descrever as funcionalidades de uma aplicação do ponto de vista do usuário. Dentro desses diagramas, possuem elementos que ajudam a entendê-lo, sendo eles os atores (bonecos), funcionalidades (elipses) e as relações (linhas).

No sistema em desenvolvimento, os atores envolvidos são: O usuário e a API, sendo:

- **Usuário:** Nesse caso, o ator é aquele que irá interagir com a aplicação, podendo utilizar suas funcionalidades.
- **API:** A API é o ator responsável pela comunicação da aplicação com o banco de dados, e outros serviços que possam ser implementados.

A figura 3 representa o caso de uso do login do usuário no sistema.

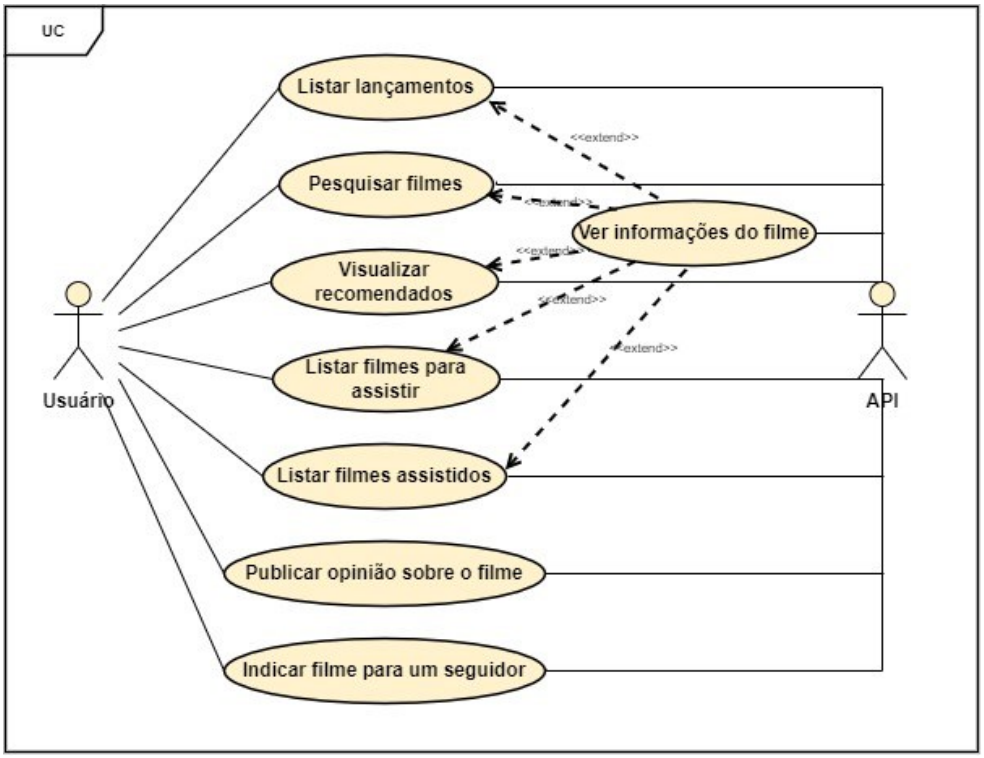
Figura 3 - Diagrama do caso de uso do login e cadastro de usuário



Fonte: Elaborado pelo autor

Já a figura 4, é a representação do que o usuário poderá realizar após estar dentro do sistema.

Figura 4 - Diagrama do caso de uso após o usuário estar dentro do sistema



Fonte: Elaborado pelo autor

3.3 Documentação do Caso de Uso de Cadastro e Login de Usuário

A tabela 12 será utilizada para documentar o passo-a-passo do item relacionado ao Cadastro do usuário:

Tabela 12 - Documentação do caso de uso do Cadastro de usuário

Nome do caso de uso	Cadastrar Usuário
Atores envolvidos	Usuário, API
Objetivo	Descrever os passos necessários para se cadastrar no sistema
Prioridade de desenvolvimento	Essencial
Ações do ator	**Ações do sistema**
1. O usuário clica em “Cadastrar”	
	2. O sistema leva o usuário até a página de cadastro
3. O usuário informa os dados necessários	
	4. O sistema valida os dados e envia para a API
	5. A API valida os dados novamente e insere os dados do usuário no banco.
	6. Após o cadastro, a API envia um email para o usuário aguardando confirmação.
7. o usuário acessa o email e confirma o cadastro.	
	8. O link do email o redireciona para uma página contento a mensagem de *feedback.*

Fonte: Elaborado pelo autor

4. Desenvolvimento

Nesse capítulo será tratado o desenvolvimento do projeto, onde será apresentado o passo-a-passo das atividades de cada etapa. O desenvolvimento inicial teve como principal objetivo codificar as funcionalidades mais básicas do aplicativo, tais como: a criação da interface de cadastro e autenticação, a busca e listagem de filmes, criação de perfil de usuário, interação com feed e serviços de recomendação de filmes.

4.1 Interfaces de Usuário (UI)

A sigla UI significa *User Interface* (em português, interface do usuário). Portanto, UI é a maneira como os usuários interagem com aplicativos, monitores, dispositivos ou software específicos. É responsável por orientar a interação do usuário e, ao mesmo tempo, realizar tarefas por meio da comunicação visual na interface. Essa dinâmica pode ser alcançada por meio de botões, menus, controle de voz e até mesmo gestos.

Ao projetar uma UI, é necessário antecipar as necessidades do usuário para criar um layout, uma interface, que seja de fácil utilização, intuitiva e possibilite que o usuário tenha uma boa experiência ao utilizar o seu aplicativo ou site.

4.2 Navegação da aplicação

O Angular fornece um esquema de rota e navegação completo, simples e fácil de usar, incluindo a realização de um esquema de proteção de rota seguro que foi utilizado em toda a nossa aplicação web. A utilização do roteamento do Angular, *Angular Routing*[18], que permite a navegação interpretar um URL do navegador como uma instrução para alterar a visualização. A figura 5 mostrara uma implantação do *Angular Routing* no projeto.

[18] *Angular Routing* é um módulo de roteamento que auxilia na navegação da aplicação web angular. Disponível em: <https://angular.io/guide/router >. Acesso em: 07/11/2021.

Figura 5 - Classe AuthRoutingModule

```
const ROUTES: Routes = [
  {
    path: '',
    component: AuthComponent,
    children: [
      {
        path: '',
        canActivate: [UnconfirmedGuard],
        component: FormLoginComponent,
      },
      {
        path: 'email-confirmation',
        canActivate: [UnconfirmedGuard],
        component: FormEmailConfirmationComponent,
      },
      {
        path: 'registration',
        canActivate: [UnconfirmedGuard],
        component: FormRegistrationComponent,
      },
      {
        path: 'recover-password',
        canActivate: [UnconfirmedGuard],
        component: RecoverPasswordComponent,
      },
      {
        path: 'reset-password',
        canActivate: [UnconfirmedGuard],
        component: ResetPasswordComponent,
      },
      {
        path: 'register',
        children: [
          {
            path: '',
            pathMatch: 'full',
            redirectTo: 'confirmation',
          },
          {
            path: 'confirmation',
            component: ConfirmatedEmailComponent,
          },
        ],
      },
    ],
  },
];
```

Fonte: Elaborado pelo autor.

Ao observar a figura 5 pode-se ver a proteção feitas nas rotas, de tal forma que um usuário não autenticado tentar acessar uma rota, é necessário um token[19] de seção, para emitir um erro, dizendo que o mesmo não tem acesso a tal rota. A classe que possui a responsabilidade de verificar o token de seção e informar se o usuário pode ou não acessar determinada rota é a *UnconfirmedGuard*. Ela pode ser visualizada com mais detalhes na figura 6.

[19] O Token de API é um identificador único que pode ser usado por apps ou sites para acessar API de terceiros ou serviços. Para saber acesse em: <https://www.atlassian.com/br/software/access/guide/api-token-controls>. Acesso em: 07/11/2021.

Figura 6 - Classe UnconfirmedGuard

```
@Injectable()
export class UnconfirmedGuard implements CanActivate, CanActivateChild {
  constructor(
    private router: Router,
    private sessionTokenStore: SessionTokenStore
  ) {}

  canActivate(): Observable<boolean> {
    return this.sessionTokenStore.getDecoded().pipe(
      op1: tap((token) => {
       if (!this.isValid(token)) {
         this.redirect();
        }
      }),
      DecodedSessionToken: map((token) => this.isValid(token))
    );
  }

  canActivateChild(): Observable<boolean> {
    return this.canActivate();
  }

  private isValid(token: DecodedSessionToken): boolean {
    return !token || !token.isEmailConfirmed;
  }

  private redirect(): void {
    this.router.navigateByUrl( url: '/m');
  }
}
```

Fonte: Elaborado pelo autor.

Após averiguação do token do usuário, é permitido o acesso a rota. Com isso o componente é chamado e se constrói a tela.

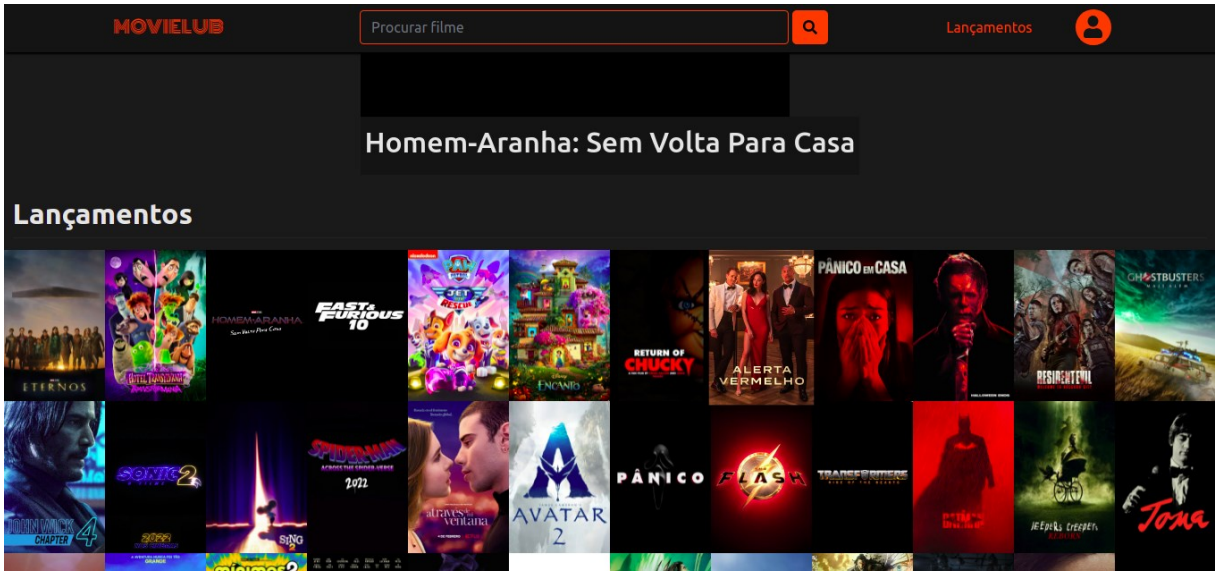
4.3 Utilização da API TMDB

Nesta aplicação é fundamental o uso de uma base de dados de filmes para que os usuários consigam avaliar, analisar as descrições, detalhes e futuramente receber recomendações personalizadas. Para isso foi escolhido a API TMDB, (The Movie Database em inglês), que possui um banco de dados de filmes e séries de TV.

A TMDB disponibiliza uma API completa para requisições, com o intuito de retornar os dados de filmes e séries. Um exemplo de uma implementação da API pode ser encontrado no Apêndice B.

Também é possível ver o retorno da API TMDB na tela lançamentos, que pode ser observada na figura 7, onde é feita uma requisição a API para retornar uma lista de filmes.

Figura 7 - Tela de início do site



Fonte: Elaborado pelo autor.

Após o retorno da API, o método httpClient(), que é responsável pela comunicação, devolve para o método listReleases() um objeto JSON[20], que contém a lista de filmes populares.

O método listReleases(), que pertence a classe TmdbClient, possui a finalidade de tratar o JSON recebido e devolver um objeto contendo uma lista de filmes do tipo *Movie*. A figura 8 mostra o método listReleases() sendo chamado pelo método assíncrono listReleases(), que pertence a classe movie.service, e também exibirá o mesmo interagindo com o httpClient().

[20] JSON, que significa JavaScript Objeto Notation, é uma formatação utilizado para estruturar dados para transmissão entre sistemas, muito comum em retornos e requisições HTTP. Para mais <https://www.json.org/json-en.html>. Acesso em: 07/11/2021.

Figura 8 - Fluxo para retornar lista de lançamentos de filmes

```
async listReleases({ page: pageNumber, userId }: { page: number , userId: User['id']}): Promise<MovieList> {
  const date = dayjs(new Date()).format('YYYY-MM-DD');
  const {
    page,
    results,
    totalPages,
    totalResults,
  } = await this.tmdbClient.listReleases( { date: { date, page: pageNumber });

  const ids = await this.listUserMovieIds(userId);

  return {
    page,
    results: results.map( callbackfn: (m: Movie) => this.setUserMovieStatus( imdbIds: ids,  movie: m)),
    totalPages,
    totalResults,
  };
}

async listReleases({ date, page: pageNumber }: MovieReleaseListOptions): Promise<MovieList> {
  try {
    const url = TmdbClient.buildUrl( path: 'discover/movie');
    const query = TmdbClient.buildQueryParams( queryParams: {
      'primary_release_date.gte': date,
      page: pageNumber,
      sort_by: 'popularity.desc',
    });
    const { body } = await this.httpClient.get( args: { query, url });

    const {
      page,
      results,
      total_pages,
      total_results,
    } = body;

    const movies = results.map((json: Record<string, any>) => (
      TmdbClient.parseMovie(json)
    ));

    return {
      page,
      results: movies,
      totalPages: total_pages,
      totalResults: total_results,
    };
  } catch (error) {···
  }
}
```

Fonte: Elaborado pelo autor.

Posteriormente a API retorna um objeto JSON contendo a lista de filmes populares e detalhes sobre a paginação para o *front-end* que é encarregado de montar o *layou*[21]1 da tela de lançamentos.

[21] O termo Layout é uma palavra inglesa, que significa plano, arranjo, esquema, design, projeto. Disponível em: <https://www.significados.com.br/layout/>. Acesso em: 07/11/2021.

4.4 Codificação da tela de informações detalhadas do filme

Antes de acessar a tela de informações detalhadas do filme, é necessário selecionar um filme, isso pode acontecer em diversas situações, como a tela de lançamentos, a tela de filmes assistidos, filmes para assistir, entre outras telas.

Ao selecionar um determinado filme, é executado o componente *movie-details.componet*.ts que requisita a função findMovieByTmdbId(), que possui como parâmetro o identificador único do filme na base de dados TMDB, que chama o método findById() da classe movies.service, passando o identificador como parâmetro, que por sua vez chama o método apiClient() importado da classe api.client, que é responsável pela comunicação a API do Movielub. Esse fluxo pode ser observado na figura 9.

Figura 9 - Fluxo para construir o componente de detalhes de filme

```
private findMovieByTmdbId(tmdbId: Movie['tmdbId']): void {
  this.isLoading = true;

  this.movieService.findById( id: tmdbId).subscribe(
    next: (movie) => {
      this.movie = movie;
      this.isLoading = false;
    },
    error: (err) => {
      this.handleError( error: err);
    }
  );
}

findById(id: number): Observable<Movie> {
  const path = [this.scope, id].join( separator: '/');

  return this.apiClient
    .get(path)
    .pipe( op1: map((resp) => this.movieFromJSON( json: resp)));
}

get(path: string, options: ApiClientOptions = {}): Observable<any> {
  const opts = this.defaultOptions(options);
  return this.http.get(this.apiUrl(path), { ...opts, responseType: 'json' });
}
```

Fonte: Elaborado pelo autor.

Após a requisição do frond-ent, a API começa a comunicação com a API do TMDB, com o intuíto de adquirir as informações do filme em questão.

O fluxo e a codificação podem ser analisados no Apêndice B, que mostra como trabalhar com a API do TMDB, e explica a lógica feita para adquirir as informações de um determinado filme.

Obtendo o sucesso, a API retorna um JSON contendo as informações daquele filme, para que, finalmente, o componente *movie-details.componet.ts* possa se construir, com isso formando a tela de detalhes de filme.

5. CONSIDERAÇÕES FINAIS

Esse trabalho teve como objetivo principal a elaboração de um aplicativo de recomendação e compartilhamento de filmes que, segundo os modelos contemporâneos de *streaming*, tem grande potencial de aceitação.

O primeiro passo desse trabalho foi a análise de viabilidade do projeto. Nesse momento foi realizado um amplo estudo de softwares similares no mercado, levantadas todas as características dos softwares e apontadas algumas características que se consideram relevantes no software. Após esse processo, realizou-se um escopo de como o aplicativo seria desde o *front-end* ao *back-end*. Um conjunto de 14 requisitos foram apontados e realizados estudos complementares de cada requisito e foram classificados de acordo com sua categorização “Funcionais” e “Não Funcionais”.

No processo de estruturação do projeto organizou-se o cronograma de acordo com o prazo e utilizamos a ferramenta Monday para auxiliar na distribuição das tarefas. Adotou-se nesse momento a metodologia de *sprints* para facilitar o desenvolvimento do software juntamente com a documentação.

De modo conjunto foram investigados alguns *frameworks* e, dentre várias opções no mercado, optou-se pelo Angular, que foi amplamente utilizado no processo de desenvolvimento, juntamente com o MySQL Workbench.

Durante os testes realizados exaustivamente verificou-se que as partes desenvolvidas funcionaram satisfatoriamente, permitindo que os objetivos propostos inicialmente fossem alcançados com sucesso.

Em razão das constantes modificações que a aplicação deverá sofrer durante seu uso por parte dos usuários, e por parte de novos filmes que forem adquiridos pela plataforma, torna-se necessária uma melhoria contínua. Também é interessante a implementação de um canal de suporte direto com os usuários para esclarecer dúvidas e coletar sugestões de melhorias, podendo tornar o software mais atraente e competitivo no mercado, em comparações as outros serviços.

A respeito da escalabilidade da aplicação propõe-se que futuramente possa-se oferecer espaço para inserir anúncios visando captar recursos para reinvestir em infraestrutura suficiente para tornar esta aplicação disponível para a sociedade por um custo acessível.

7 Referências

AMADEU, C. V.; GONÇALVES, P. R. & TEIXEIRA JUNIOR. **Análise e Projeto de Sistemas**. Batatais: Claretiano, 2013.

AMAZON PRIME VIDEO. Disponível em: <https://www.primevideo.com/>. Acesso em 12 out. 2021.

André Baltieri. Balta.io, **Angular: Rotas, Guardas e Navegação**. Disponível em: <https://balta.io/blog/angular-rotas-guardas-navegacao>. Acesso em 07 nov. 2021.

Brasil, André. **O que é o Node.js e quais são as suas características**, 2020. Disponível em: <https://king.host/wiki/artigo/o-que-e-o-node-js-e-quais-sao-as-suas-caracteristicas/>. Acesso em: 1 dez. 2020.

Brasil, Firefox. Disponível em <https://www.mozilla.org/pt-BR/firefox/new>. Acesso em 90 nov. 2021.

Brasil, Google Chrome. Disponível em <https://www.google.com/intl/pt-BR/chrome>. Acesso em 09 nov. 2021.

Brasil, Opera. Disponível em <https://www.opera.com/pt>. Acesso em 09 nov. 2021.

Brasil, Techtudo, **Linus Tornalds**. Disponível em: <https://www.techtudo.com.br/tudo-sobre/linus-torvalds.html>. Acesso em 20 out. 2021.

Conceito.De, **Conceito de browser**, 2016. Disponível em: <https://conceito.de/browser>. Acesso em 05 nov. 2021.

DevMedia, **Guia de Node.JS**. Disponível em: <https://www.devmedia.com.br/guia/node-js/40312>. Acesso em 10 out. 2021.

GitHub. Disponível em: <https://github.com/>. Acesso em 19 out. 2021.
GOOGLE PLAY. Disponível em: <https://play.google.com>. Acesso em 07 nov. 2021. https://wise.com/pt/help/articles/2958229/o-que-e-um-token-de-api

ISO/IEC/IEEE 12207. **Tecnologia da Informação – Processos de ciclo de vida de software**. Rio de Janeiro: ABNT. 1998.

João Pedro Voltarelli, TechTudo. **Como funciona Monday.com? Conheça recursos e preços da plataforma**, 22 out. 2020. Disponível em: <https://www.techtudo.com.br/listas/2020/10/como-funciona-mondaycom-conheca-recursos-e-precos-da-plataforma.ghtml>. Acesso em 05 nov. 2021.

JSON. Disponível em: <https://www.json.org/json-en.html>. Acesso em 07 nov. 2021.
Longen, Andrei. Hostinger Tutoriais, **O que é GitHub e Como Usá-lo**, 15 out. 201. Disponível em: <https://www.hostinger.com.br/tutoriais/o-que-github>. Acesso em 07 nov. 2021.

Longen, Andrei. **O Que é Angular? Guia para Iniciantes**, 2020. Disponível em: <https://www.hostinger.com.br/tutoriais/o-que-e-angular/>. Acesso em: 1 dez. 2020.

MJV Team, MJV. **UX e UI: entenda as diferenças e a aplicação em projetos**, 18 jan. 2021. Disponível em <https://www.mjvinnovation.com/pt-br/blog/diferencas-entre-ux-e-ui>. Acesso em 10 out. 2021.

Monday. Disponível em: <https://monday.com/lang/pt>. Acesso em 07 nov. 2021.

NETFLIX. Disponível em: <https://www.netflix.com/br/>. Acesso em 20 out. 2021.

NodeJs, **Documentação versão 14.18.1 LTS,** 2021. Disponível em: <https://nodejs.org/docs/latest-v14.x/api/process.html>. Acesso em 05 nov. 2021.

SCRUM. **Desenvolvimento Ágil**, 2020. Disponível em: <https://www.desenvolvimentoagil.com.br/scrum/> Acesso em: 2 dez. 2020.

Sommerville, I. (2019). **Engenharia de Software 10ª Edição**. São Paulo: Pearson Addison-Wesley.

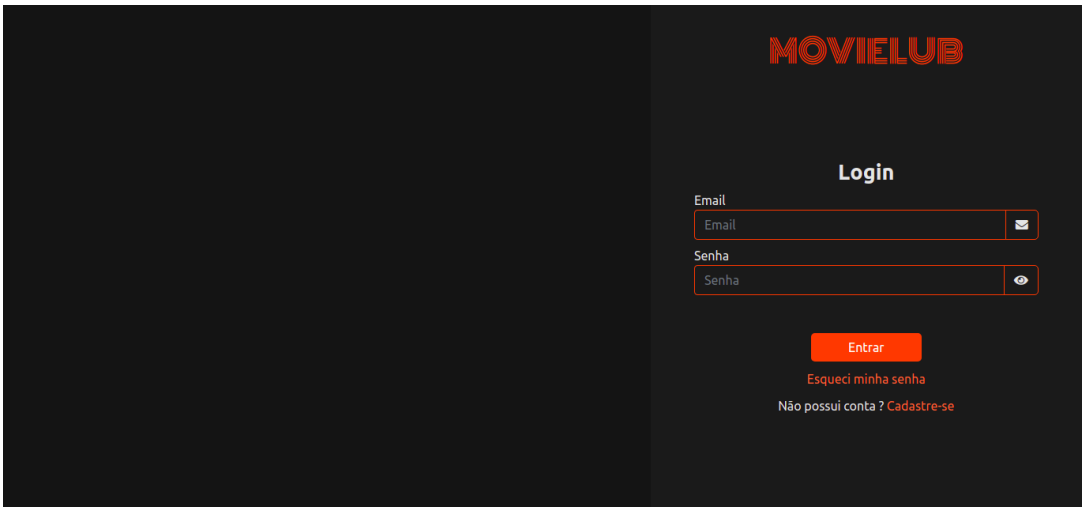
Visual Studio Code. **Documentação**, 2021. Disponível em <https://code.visualstudio.com/docs>. Acesso em 10 nov. 2021.

APÊNDICE A – Telas do Aplicativo

A.1 Tela de login

Ao iniciar o sistema, o usuário se depara com a tela de cadastro e autenticação (figura 10), na qual é mostrado o logotipo do sistema e os campos de e-mail e senha para realizar a autenticação do usuário. Se for o caso de um novo usuário logo embaixo se encontra o “Cadastre-se”, que ao clicar o usuário em questão é redirecionado para a tela de cadastro.

Figura 10 - Tela de início do site

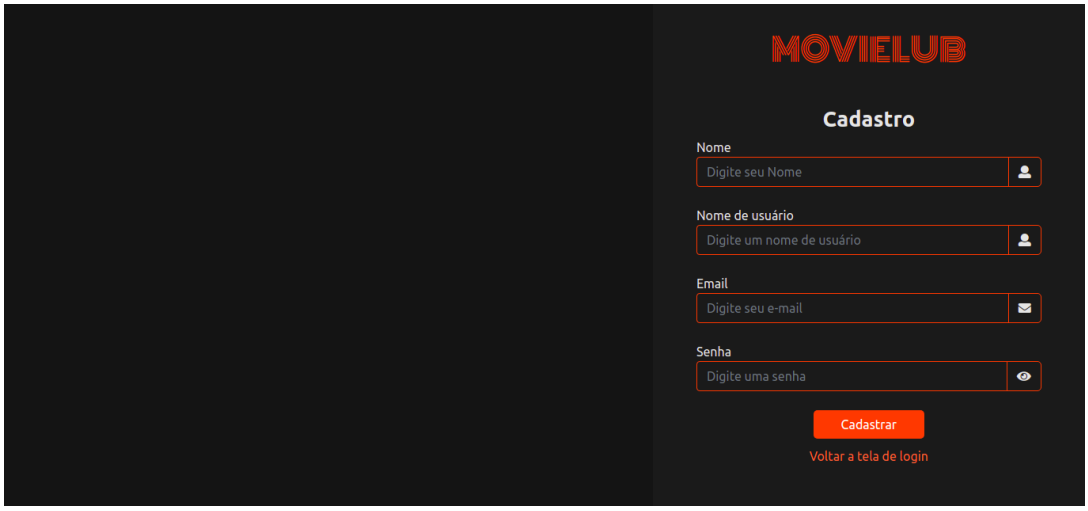


Fonte: Elaborado pelo autor.

A.2 Tela de cadastro

A tela de cadastro (figura 11) tem como objetivo cadastrar um novo usuário na plataforma Movielub. É necessário o nome completo, e-mail válido e uma senha alfanumérica com no mínimo seis caracteres. Após a inserção dos dados, o sistema realiza o envio de um e-mail para o correio eletrônico do usuário em questão, para que possa validá-lo.

Figura 11 - Tela de cadastro do site

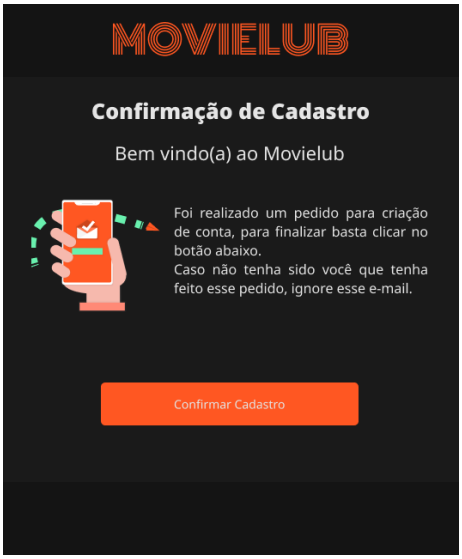


Fonte: Elaborado pelo autor.

A.3 Template do e-mail

Após a inserção dos dados na tela de cadastro, é enviada uma requisição do tipo POST para a API na rota (api/v1/users), que carrega as informações do usuário. Após isso, é executada uma verificação se o usuário é apto a realizar o cadastro. Caso for apto, é enviado um *template* de verificação, que pode ser encontrado na figura 12, no endereço de e-mail que o usuário informou.

Figura 12 - Código referente ao reenvio de e-mail

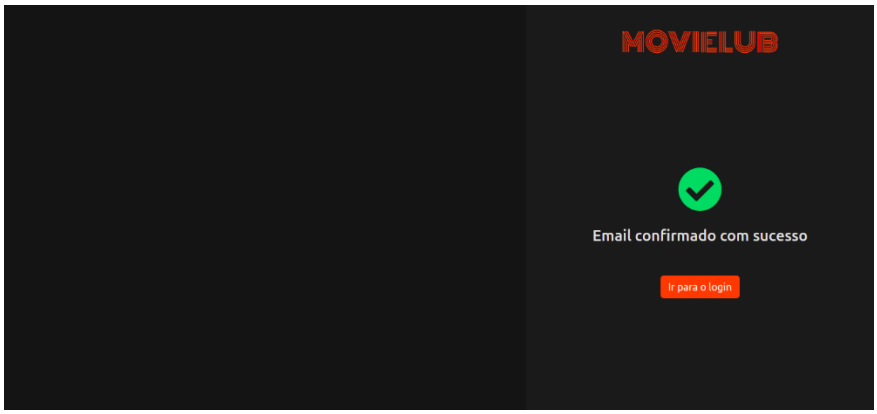


Fonte: Elaborado pelo autor.

A.4 Tela de e-mail confirmado

Após a confirmação do e-mail, o usuário é direcionado novamente para a tela de login/cadastro, onde poderá realizar o login no sistema. Na figura 13 podemos observar a tela de login após o e-mail confirmado.

Figura 13 - Tela de e-mail confirmado

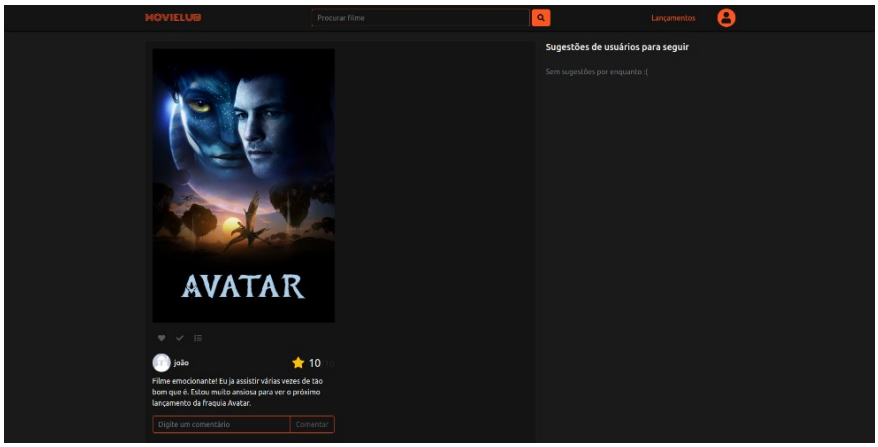


Fonte: Elaborado pelo autor.

A.5 Tela inicial

Caso confirmado o e-mail e realizado o login bem-sucedido, o usuário será retornado para a tela inicial do sistema, onde serão exibidos os filmes genéricos e as recomendações. A tela inicial do aplicativo (figura 14) exibe os filmes populares e as recomendações caso o login da tela inicial seja bem-sucedido, além de incluir o botão de menu e de busca.

Figura 14 - Tela inicial do site

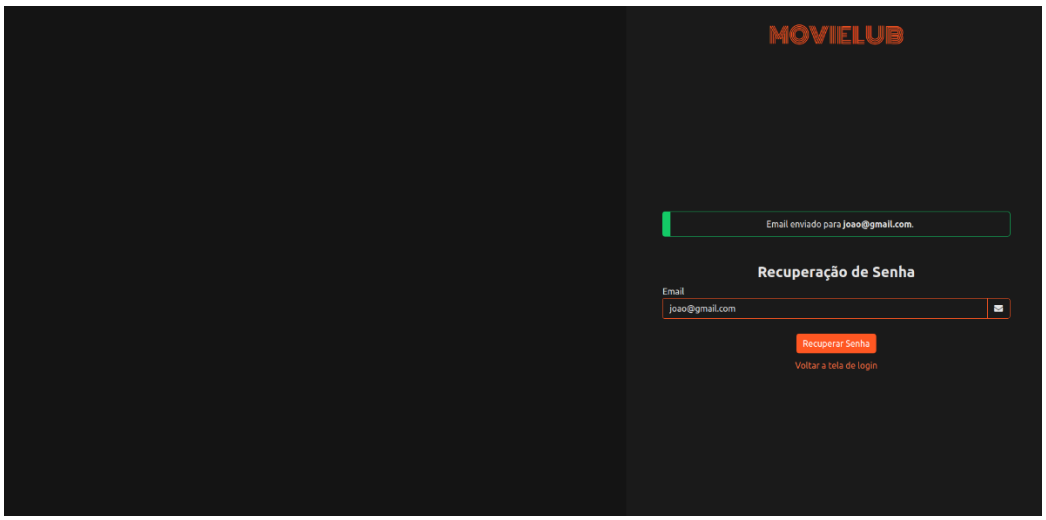


Fonte: Elaborado pelo autor.

A.6 Recuperação de Senha

A tela de recuperação de senha, localizada na figura 15, tem como objetivo redefinir a senha do usuário em questão. Ao solicitar a redefinição é necessário inserir o e-mail associado a conta que o usuário deseja recuperar. O *back-end* é responsável por fazer a verificação se o e-mail pertence a conta em questão. Se não for, é retornado um erro.

Figura 15 - Recuperação de Senha do site



Fonte: Elaborado pelo autor.

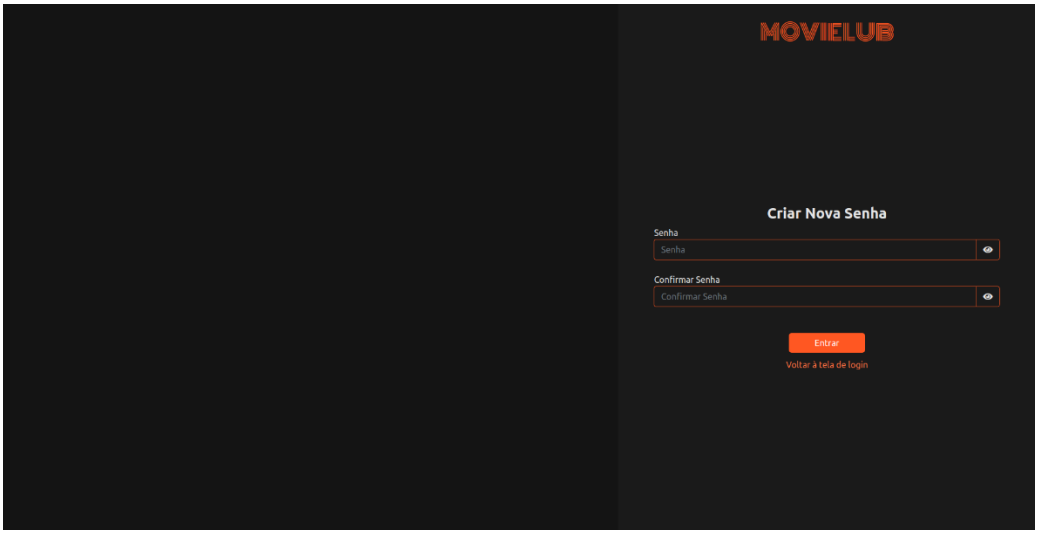
Caso a verificação ocorra com sucesso, é enviado um *template* com o link para a redefinição, que pode ser encontrado na figura 16, para o correio eletrônico do usuário.

Figura 16 - Template de recuperação de senha

Fonte: Elaborado pelo autor.

Ao clicar em “Definir uma nova Senha” no *template* envido pela Movielub, o usuário é redirecionado para a tela de redefinição de senha (figura 17), onde pode redefinir a senha atual.

Figura 17 - Tela de redefinição de senha do site



Fonte: Elaborado pelo autor.

Obtendo sucesso no procedimento anterior, é exibido para o usuário uma mensagem de confirmação, que pode ser vista na figura 18.

Figura 18 - Tela de sucesso da redefinição de senha do site

Fonte: Elaborado pelo autor.

A.7 Tela de perfil do usuário

A tela de perfil do usuário Movielub contém diversas informações, como a quantidade de filmes assistidos e que deseja assistir em breve, quantidade de seguidores, nome, foto de perfil e um ícone que contém a funcionabilidade de configurar dados da conta do usuário em questão. A figura 19 apresenta a tela de perfil do site.

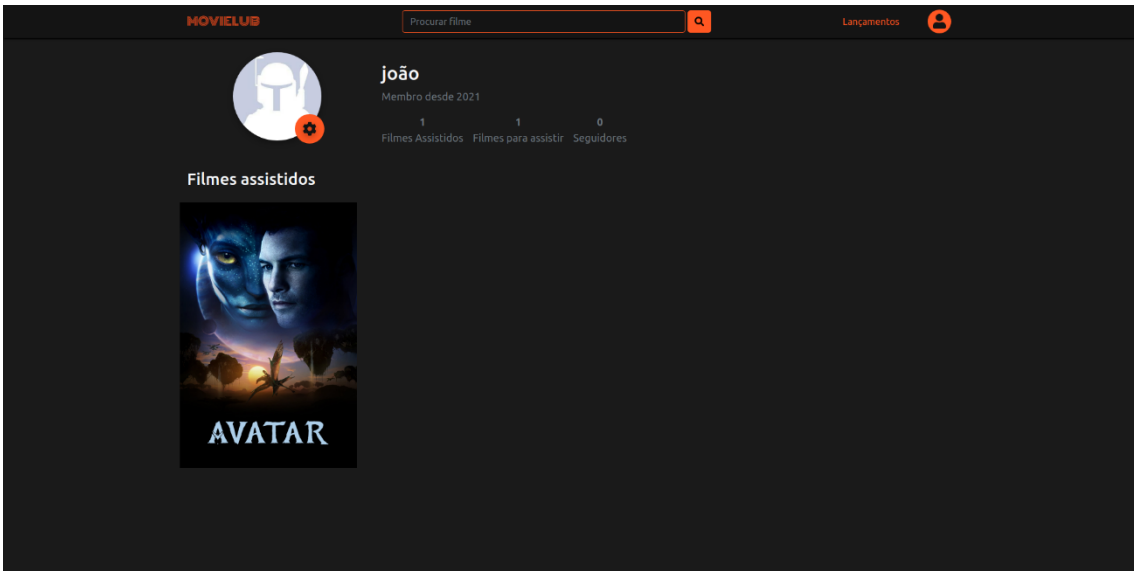
Figura 19 - Tela de perfil do site

Fonte: Elaborado pelo autor.

A.8 Tela de filmes de usuário

A tela de filmes do usuário tem como objetivo listar todos os filmes de seu interesse, juntamente com o histórico de filmes já assistidos. Na versão site, figura 20, encontra-se também um filtro ao lado esquerdo.

Figura 20 - Tela de filmes de usuário



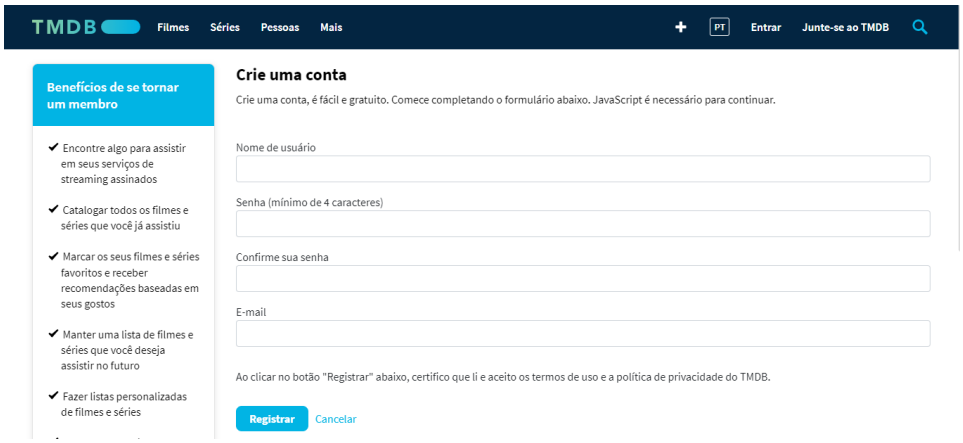
Fonte: Elaborado pelo autor

APÊNDICE B – Utilizando a API TMDB

TMDB é um banco de dados gratuito e de código aberto, onde se encontra filmes e séries de TV. Foi criado em 2008 por Travis Bell.

Para a utilização da API, é necessário primeiramente se registrar na plataforma, posteriormente efetuar o login. Localiza-se a seção API seguindo os seguintes passos: Clicar no menu *"*Configurações*"* de usa conta, onde se encontra um ícone com a inicial do nome, o que poderá ser visto na figura 21.

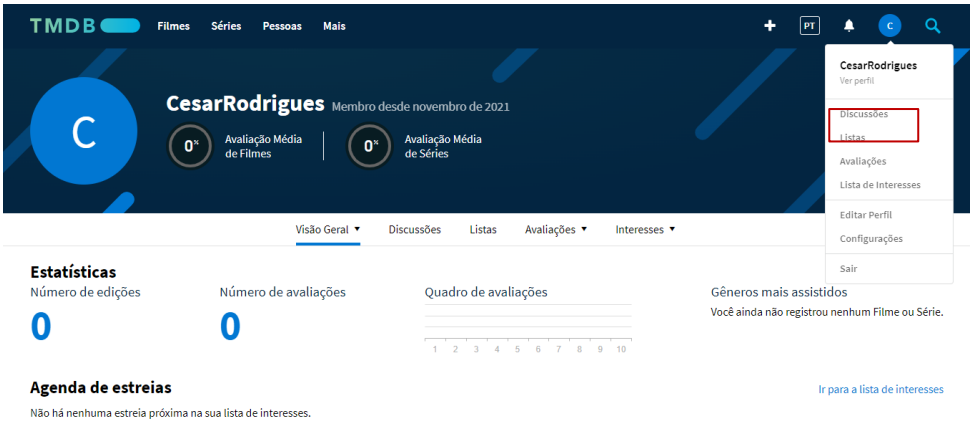
Figura 21 - Cadastro na plataforma TMDB



Fonte: Elaborado pelo autor.

A figura 22 mostrará como encontrar a seção *"Configurações".*

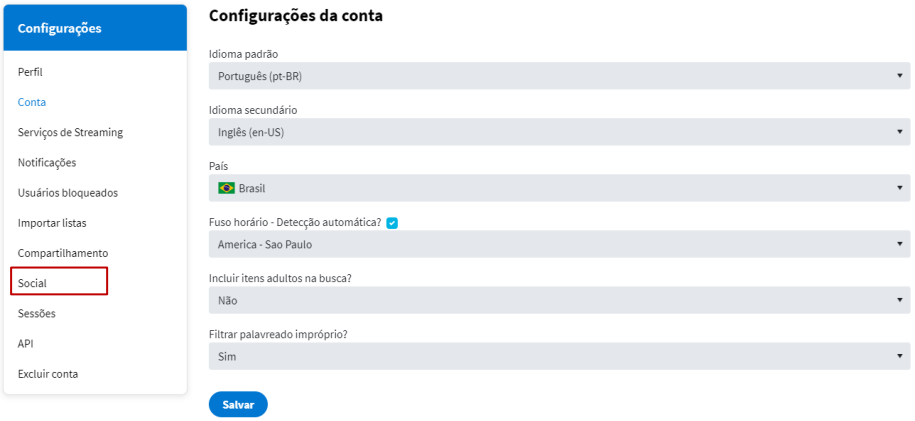
Figura 22 - Como encontrar a seção Configurações na plataforma TMDB



Fonte: Elaborado pelo autor.

Clica-se na seção “API” no menu ao lado esquerdo e, logo depois, clica-se em *“Criar”*, ao lado do botão *“Visão geral”*, para criar uma chave da API, isso pode ser visto na figura 23.

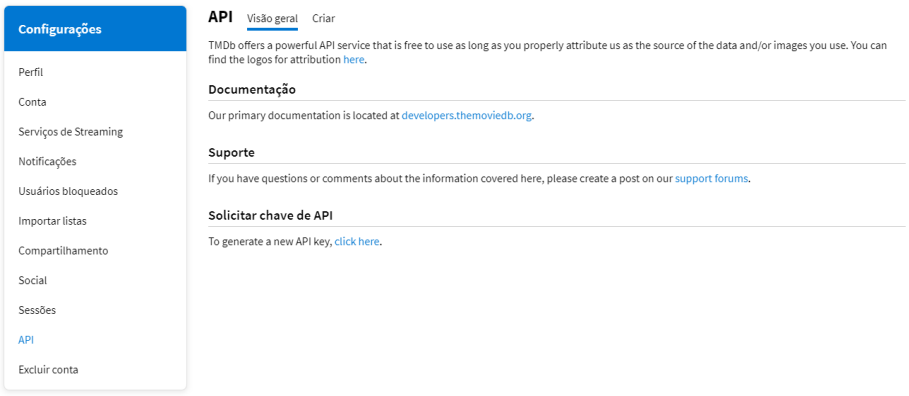
Figura 23 - Seção de Configurações da plataforma TMDB



Fonte: Elaborado pelo autor

Na figura 24 mostrara como encontrar a seção *“Visão geral”*.

Figura 24 - Seção sobre criação e visão geral da API na plataforma TMDB



Fonte: Elaborado pelo autor

Posteriormente, a aplicação será configurada para receber e fazer requisições HTTP da API, utilizando como padrão para o retorno da requisição o formato JSON. Ao acessar a página de desenvolvedores do TMDB* poderão ser encontradas todas as requisições que o serviço oferece.

São necessários seguintes passos para realizar uma requisição, que tem a função de retornar um determinado filme, utilizando o seu identificado único.

Primeiramente cria-se um Modelo de Filme, o que é necessário uma classe TypeScript denominada *Movie*, onde serão criados os atributos, construtor e algumas interfaces, como é possível observar na figura 25.

Figura 25 - Classe Movie TMDB

```
class Movie implements MovieAttributes {

  backdropPath?: string;
  genres: MovieGenre[];
  id?: number;
  imdbId?: string;
  imdbRating?: number;
  overview?: string;
  posterPath?: string;
  rating?: number;
  releaseDate?: string;
  title: string;
  tmdbId?: number;
  userMovieStatus?: UserMovieStatus;

  constructor(attr: MovieAttributes) {···
  }

  toJson(): Record<string, any> {···
  }
}

export enum UserMovieStatus {···
}

export interface MovieAttributes {···
}

export interface MovieList {···
}

export default Movie;
```

Fonte: Elaborado pelo autor

É preciso criar uma segunda classe, denominada *TmdbClient*, responsável pelo gerenciamento das requisições. Foi imprescindível a utilização de funções assíncronas, terminadas pelo *async* declarado no começo da função, fundamental para o processamento de requisições da API em conjunto com os outros processos da aplicação.

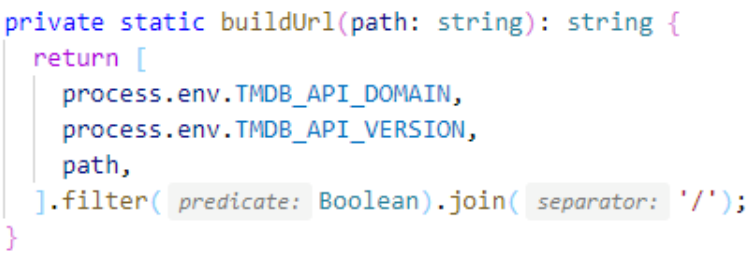
Observa-se o método *findByid,* na figura 26, que pertence a classe *TmdbClient* e tem a função de realizar uma requisição do método *Http GET*, solicitando a API do TMBD os dados do filme, é possível ver a utilização da assincronia, retornando uma *promise* do tipo *Movie.*

Figura 26 - Classe TmdbClient responsável pelas requisições a API TMBD

Fonte: Elaborado pelo autor

O método *findByid* possui um único parâmetro, do tipo numérico, que representa o identificador único do filme no banco de dados da TMDB. Ao instanciar o método passando um identificador, ele irá buscar a URL utilizando o método *buildUrl()*, que pode ser observado na figura 27, que tem a função de retornar a rota baseada nas variáveis de ambiente do projeto, acessada pelo objeto global *process.env*, já formatada para ser utilizada na requisição.

Figura 27 - – Método buildUrl() da classe TmdbClient



Fonte: Elaborado pelo autor

Posteriormente, são requisitados ao método buildQueryParams() os parâmetros, padrões necessários para a requisição, acessando novamente o objeto global *process.env** para a obtenção dos valores, como o *api_key*, que é utilizando para a autenticação. Isso pode ser analisado na figura 28.

Figura 28 - Método builQueryParams() da classe TmdbClient

```
private static buildQueryParams(queryParams?: Record<string, any>) {
  const defaultParams = {
    api_key: process.env.TMDB_API_KEY,
    language: process.env.TMDB_LANG,
    include_adult: process.env.TMDB_INCLUDE_ADULT,
  };

  return {
    ...defaultParams,
    ...queryParams,
  };
}
```

Fonte: Elaborado pelo autor

Após a obtenção da URL e dos parâmetros padrões, é chamado o método *get()* da classe *httpClient*, que efetua uma requisição para a API da TMDB na rota "https://api.themoviedb.org/3/movie/550?api_key=123", no caso do método que possui a função de retornar um determinado filme. Após o sucesso, o método *get()* retorna o corpo da requisição em questão, onde é esperado um objeto JSON.

Por fim, o método *parseMovie()* recebe como parâmetro o objeto JSON da requisição. O método possui a função de transformar o JSON retornado pela API TMDB em um objeto do tipo *Movie* que foi mencionado no início. Para essa conversão são necessárias algumas condições. O método e suas peculiaridade podem analisadas na figura 29.

Figura 29 - Método parseMovie() da classe TmdbClient

```
private static parseMovie(json: Record<string, any>): Movie {
  const {
    backdrop_path,
    genre_ids,
    genres,
    id,
    imdb_id,
    overview,
    poster_path,
    release_date,
    title,
    vote_average,
  } = json;

  return new Movie( attr: {
    backdropPath: TmdbClient.imagePathFromJSON( path: backdrop_path),
    genres: genres
      ? genres.map((genre: Record<string, any>) => MovieGenreMap.get( key: genre.id))
      : genre_ids.map((genreId: number) => MovieGenreMap.get( key: genreId)),
    imdbId: imdb_id,
    overview,
    posterPath: TmdbClient.imagePathFromJSON( path: poster_path),
    rating: vote_average,
    releaseDate: release_date,
    title,
    tmdbId: id,
  });
}
```

Fonte: Elaborado pelo autor

Essa, entre outras requisições, está presente em toda a aplicação. No caso de buscar um determinado filme pelo seu identificador único, é possível observar isso acontecendo quando um usuário seleciona um filme na seção *Filmes Assistidos* ou *Filmes para Assistir*, então o método é assinado ao clicar no filme escolhido pelo tal. Após isso, é possível comparar o retorno da API TMDB, na figura 30.

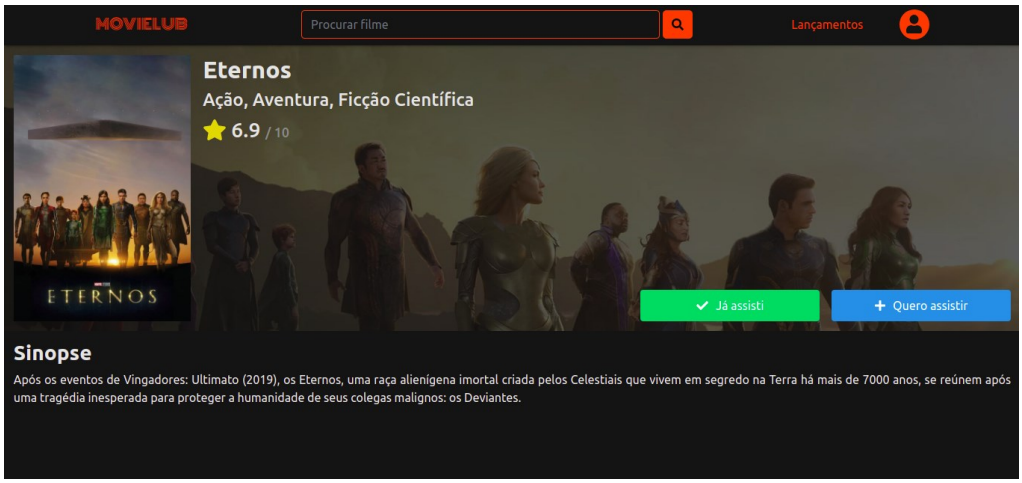
Figura 30 - Retorno do filme Eternals da API TMBD


```
"adult": false,
"backdrop_path": "/3G6wET9eLvYn3aoIj8NfQFhpYEB.jpg",
"belongs_to_collection": null,
"budget": 200000000,
"genres": [
  {
    "id": 28,
    "name": "Ação"
  },
  {
    "id": 12,
    "name": "Aventura"
  },
  {
    "id": 878,
    "name": "Ficção científica"
  }
],
"homepage": "https://www.marvel.com/movies/eternals",
"id": 524434,
"imdb_id": "tt9032400",
"original_language": "en",
"original_title": "Eternals",
"overview": "Após os eventos de Vingadores: Ultimato (2019), os Eternos, uma raça alienígena imortal criada pelos Celestiais...",
"popularity": 2026.37,
"poster_path": "/eDHOutMxLMgComnVhUK0Xfwunu5.jpg",
"production_companies": [
  {
    "id": 420,
    "logo_path": "/hUzeosd33nzE5MCNsZxCGEKTXaQ.png",
    "name": "Marvel Studios",
    "origin_country": "US"
  }
]
```


Fonte: Elaborado pelo autor

Com esses dados, pode-se obter os valores para preencher os dados necessários da tela que contém a descrição do filme em questão, que se encontra na figura 31.

Figura 31 - Tela de informações detalhadas sobre um filme demonstrado o uso da API TMDB



Fonte: Elaborado pelo autor