**Desenvolvimento de um sistema web para stream de filmes por assinatura**

**Erik Kened Lopes Ferreira¹**

### Prof. Hylson Vescovi Netto (orientador)¹

¹Instituto Federal Catarinense – Campus Blumenau

Blumenau, SC – Brasil

### {erikkened51@gmail.com, hylson.vescovi@ifc.edu.br}

***Resumo***​*. Este trabalho descreve as atividades envolvidas no planejamento e desenvolvimento de uma aplicação web voltada para o fornecimento de filmes por meio de streaming, esta aplicação disponibilizará vários títulos a seus usuários os quais realizaram pagamento de assinatura mensal para ter acesso ao conteúdo do site.*

# 1. Introdução

Cada vez mais estamos conectados seja através de celulares, computadores, tablets ou televisores e cada vez mais buscamos por conteúdos que possam agregar a nossas vidas seja através de documentários, filmes inspiradores ou até mesmo sentir-se relaxado através de algum título qualquer.

Pode-se encontrar diversos títulos em plataformas pagas que aliás existem várias no mercado, mas, no entanto, de uma forma geral estamos limitados ao conteúdo já disponível na plataforma. Quando se quer ver algum título que não esteja previamente disponível nestas plataformas pode-se recorrer aos sites de distribuição duvidosa que sempre lhe colocam milhares de anúncios para que possa assistir algum título, mas nem sempre é possível encontrar o que se procura.

Diante da necessidade de economizar o tempo gasto nesta busca e ter uma plataforma que possibilite ao usuário solicitar os títulos desejados, foi desenvolvido a aplicação pipocando.

A aplicação aqui apresentada tem como objetivo principal de receber solicitações de títulos e disponibiliza-los no menor prazo possível. Para que desta forma os usuários não fiquem limitados ao conteúdo previamente disponível.

## 1.1. Tema/Problema

Todos os dias milhares de pessoas no mundo assistem a filmes sob demanda nas mais diversas plataformas, mas nem sempre é possível encontrar o título desejado somente é possível navegar e consumir o conteúdo previamente disponível.

Disponibilizar uma forma de interação com os usuários onde ele pode solicitar um título que não está previamente cadastrado, fornece maior liberdade ao cliente não limitando o conteúdo ao que a plataforma oferece, mas sim ao que o cliente deseja.

## 1.2. Objetivos Propostos/Solução dos Problemas

O objetivo deste trabalho é fornecer uma aplicação web para o fornecimento de filmes por meio de *stream*.

Os objetivos específicos são:

1. Pesquisar o referencial teórico para o desenvolvimento do sistema.
2. Levantar os requisitos funcionais e não funcionais do sistema.
3. Desenvolver o projeto do banco de dados relacional.
4. Implementar o sistema.
5. Realizar testes.

## 1.3. Escopo

Este projeto tem como objetivo entregar uma plataforma web para gerenciamento e exibição de filmes online. O sistema deverá oferecer os módulos de Filmes, administração, contato e solicitações. No módulo de administração existem os módulos de pagamento e assinaturas.

Todos os módulos podem consultar, criar e alterar registros. No módulo de filmes existe a funcionalidade de pesquisar títulos.

A solução proposta não prevê a cópia segura de dados, tal funcionalidade deve ser configurada separadamente.

## 1.4. Viabilidade do Projeto

A aplicação necessita de muito pouco processamento do dispositivo do usuário porem as tecnologias utilizadas limitam o funcionamento em dispositivos assim para seu correto funcionamento é necessária uma versão superior ao Nougat (7.0). Para um melhor carregamento na exibição dos títulos é importante que a velocidade de conexão com a internet do usuário seja superior a 5Mb/s.

## 1.5. Método de Trabalho

Para o desenvolvimento da ferramenta foi utilizado os conceitos de *Unified Process* (UP) ou processo unificado, no entanto, este processo não foi seguido fielmente, como um todo.

O processo unificado prevê um conjunto de atividades necessárias para transformar requisitos de usuário em um ​*software.* ​tal escolha ocorreu pela principal vantagem oferecida por este processo, ou seja, a de não ser necessário aguardar uma versão final do ​*software* ​para visualizar um resultado, tendo em vista que estes resultados são vistos, constantemente, devido ao modelo de desenvolvimento ser interativo e incremental. Cada interação realizada durante a etapa de desenvolvimento do projeto resulta em um incremento, e este significa uma nova funcionalidade no sistema (PRESSMAN, 2011).

De modo a compreender melhor a respeito do desenvolvimento deste trabalho, inicialmente, se faz necessário esclarecer o funcionamento da ​*internet*​.

A ​*internet* é uma rede largamente utilizada para realizar a comunicação entre todos os computadores e dispositivos com acesso a esta rede.

Os computadores conectados à ​*internet* interagem através de ​*requests* e *responses* (requisições e respostas) e, assim, cada requisição efetuada por um dispositivo na rede indica, além dos endereçamentos dos dispositivos do solicitante e do destinatário, também qual o conteúdo desejado.

A ​*web* é mantida por computadores que apresentam um processamento e disponibilidade superiores, e estes são chamados de ​*servers* ​ (servidores). Dessa forma, os servidores são responsáveis por receberem as requisições efetuadas pelos computadores convencionais, chamados de ​*clients* (clientes), bem como em retornar na forma de respostas.

A comunicação entre cliente e servidor é realizada utilizando o protocolo *HyperText Transfer Protocol* (HTTP) ou Protocolo de Transferência de Hipertexto, através de um navegador ​*web*​, responsável por enviar requisições, receber respostas e as traduzir, de modo que o usuário visualize no formato de documentos, de fotos, de vídeos e de imagens.

Conhecendo a respeito do funcionamento da comunicação entre computadores através da ​*web*​, pode-se entender o conceito de ​*web app* (aplicação ​*web*​), foco do trabalho em questão.

Uma aplicação ​*web* ​é como um ​*software,* porém ao invés de ser instalada, em um computador convencional, esta aplicação é configurada no servidor, de modo que seja possível acessá-la através da ​*internet*​, em que todo o processamento da aplicação é realizado no servidor, enquanto a apresentação visual é realizada no navegador de *internet* ​utilizado pelo usuário.

O funcionamento da aplicação está descrito no decorrer desta seção, que além de exibir as ferramentas utilizadas neste trabalho, também aborda os recursos que compõem a aplicação, de modo que a mesma esteja disponível para acesso pelo usuário.

## 1.6. Ferramentas

### 1.6.1. WAMP

WampServer (mais conhecido WAMP) é um [software](https://pt.wikipedia.org/wiki/Software) que efetua a instalação automática de um conjunto de softwares no [computador](https://pt.wikipedia.org/wiki/Computador), de modo a facilitar a configuração de um [software interpretador](https://pt.wikipedia.org/wiki/Interpretador) de [scripts](https://pt.wikipedia.org/wiki/Linguagem_de_computador_interpretada) local e um banco de dados no sistema [Windows](https://pt.wikipedia.org/wiki/Windows).

O WAMP é uma ferramenta que auxilia no desenvolvimento, permitindo aos programadores de websites testarem o trabalho em seus próprios computadores, sem necessitar de acesso à [Internet](https://pt.wikipedia.org/wiki/Internet) (WampServer, 2021).

### 1.6.2. PHP MyAdmin

PhpMyAdmin é uma ferramenta de software livre escrita em PHP, destinada a lidar com a administração do MySQL na web. PhpMyAdmin suporta uma ampla gama de operações no MySQL e MariaDB. As operações usadas com frequência (gerenciamento de bancos de dados, tabelas, colunas, relações, índices, usuários, permissões, etc.) podem ser realizadas por meio da interface do usuário, enquanto você ainda tem a capacidade de executar diretamente qualquer instrução SQL (PhpMyAdmin 2021).

### 1.6.2. Microsoft Visual Studio Code

O Visual Studio Code é um editor de código-fonte desenvolvido pela Microsoft para Windows, Linux e macOS. Ele inclui suporte para depuração, controle Git incorporado, realce de sintaxe, complementação inteligente de código, snippets e refatoração de código (Microsoft, 2021).

### 1.6.3. Astah Community 7.1

O Astah é uma ferramenta que possibilita a criação e modelagem dos diagramas UML - *Unified Modeling Language* ou Linguagem de Modelagem Unificada do projeto (ASTAH, 2021).

A ferramenta foi utilizada para o desenvolvimento do diagrama de Caso de Uso apresentado na seção 4.0 deste documento.

## 1.7. Front-end.

### 1.7.1. HTML 5

*HyperText Markup Language* – Linguagem de Marcação de Texto ou ainda HTML, é uma linguagem utilizada na construção de páginas ​*web* ​para publicação de conteúdo, na qual os documentos criados na linguagem são interpretados pelos navegadores e apresentados ao usuário, no formato de letras diferenciadas em cores, em imagens e em vídeos.

Em sua versão 5 foram incluídas novas ​*tags* para realização de marcações responsáveis por identificar o conteúdo presente na página, novos elementos para vídeos e imagens, elementos para desenhos, controles para formulários e suporte ao CSS3 (W3SCHOOLS, 2021).

### 1.7.2. CSS3

A ferramenta denominada de ​*Cascading Style Sheets* ​(CSS) ou​Folhas de Estilo em Cascata é uma linguagem utilizada para definir a aparência de documentos, que adotam as linguagens de marcação como, por exemplo, o HTML, tendo em vista que o CSS define como serão exibidos os elementos existentes em uma página da ​*web*​.

Em sua versão 3 foram incluídas novas propriedades, de forma a facilitar a criação de estilos para os componentes, realizar avanços na utilização das cores e textos e, ainda, possibilitar selecionar atributos HTML com os seletores entre outros (​W3SCHOOLS​, 2021).

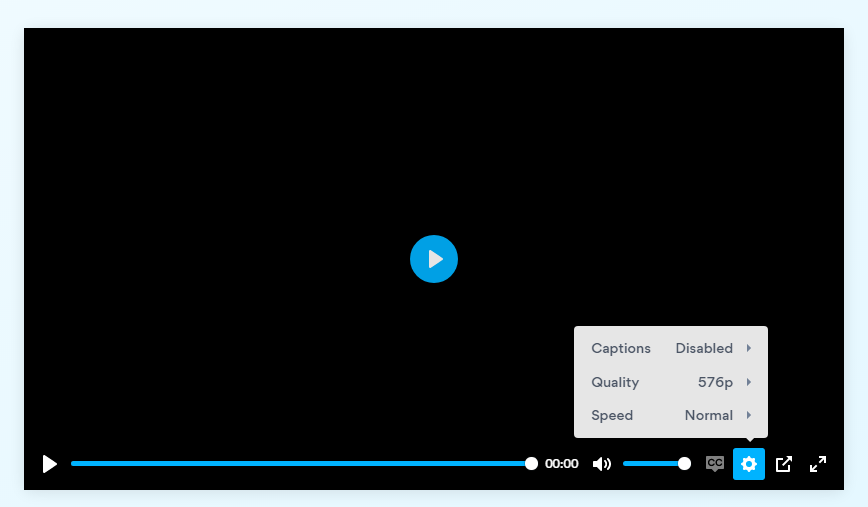
### 1.7.3. Angular cli 11.0.3

O Angular é um framework [*JavaScript*](#_1.7.6._JavaScript) código aberto, mantido pelo Google, que auxilia na execução de *single-page applications*. Seu objetivo é aumentar aplicativos que podem ser acessados por um navegador web, foi construído sob o padrão *model-view-view-model* (MVVM), em um esforço para facilitar tanto o desenvolvimento quanto o teste dos aplicativos. (Angular.io, 2021).

### 1.7.4. Plyr

O Plyr é um ​*framework* ​utilizado, também, para facilitar o desenvolvimento do *front-end*​ da aplicação, contém recursos para o controle de exibição de vídeos em diversos formatos, inclui a possibilidade de adicionar legendas e outras informações pertinentes a exibição do vídeo. (Plyr.io, 2021).

Alguns dos controles disponíveis no ​framework​ são controle de reprodução, controle de velocidade de reprodução, controle de volume, seleção de qualidade da mídia, seleção de legendas, colocar o player em tela cheia e restaurar ao tamanho normal (Figura 1).



**Figura 1. Plyr. Fonte: Plyr.io.**

### 1.7.5. Material Design Icons Iconfont 6.1

Material Design Icons Iconfont é um conjunto de arquivos pré-compilados, que possibilita a utilização de vários ícones e animações personalizados no projeto (Material.io, 2021).

Esta ferramenta foi utilizada para atender a todos os ícones do projeto.

### 1.7.6. JavaScript

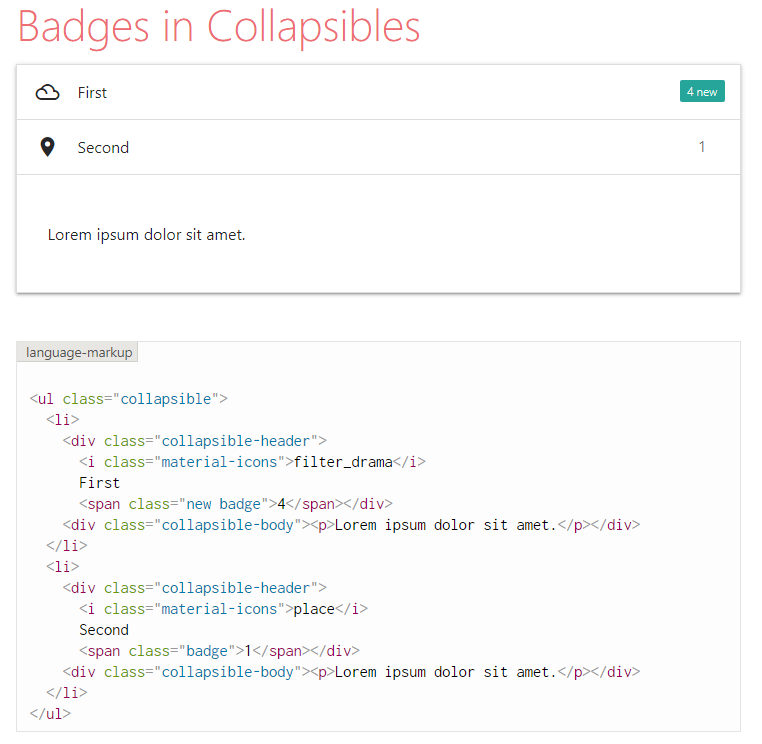
Considerada como uma linguagem de programação, o JavaScript executa no lado do usuário, ou seja, é processada pelo navegador, quando o usuário abre a página contendo a chamada do ​*script*​.

Com JavaScript se pode criar desde efeitos na página como, por exemplo, mostrar um ​*pop-up* ​em que se exibe uma mensagem ou conteúdo ou até mesmo coletar informações de itens de formulários e enviar para o ​*back-end* efetuar o processamento dessa informação e retornar com um determinado resultado (JAVASCRIPT, 2021).

### 1.7.7. Materialize CSS 1.0

O Materialize é um ​*framework,* ​cuja finalidade envolve facilitar o desenvolvimento do *front-end* da aplicação, possui vários componentes pré-compilados, responsáveis por implementar funcionalidades de forma fácil ao projeto, sendo a seguir exemplificado na figura que segue (MATERIALIZECSS, 2021).

O Materialize disponibiliza entre diversos outros elementos configuráveis menus expansíveis (Figura 2).

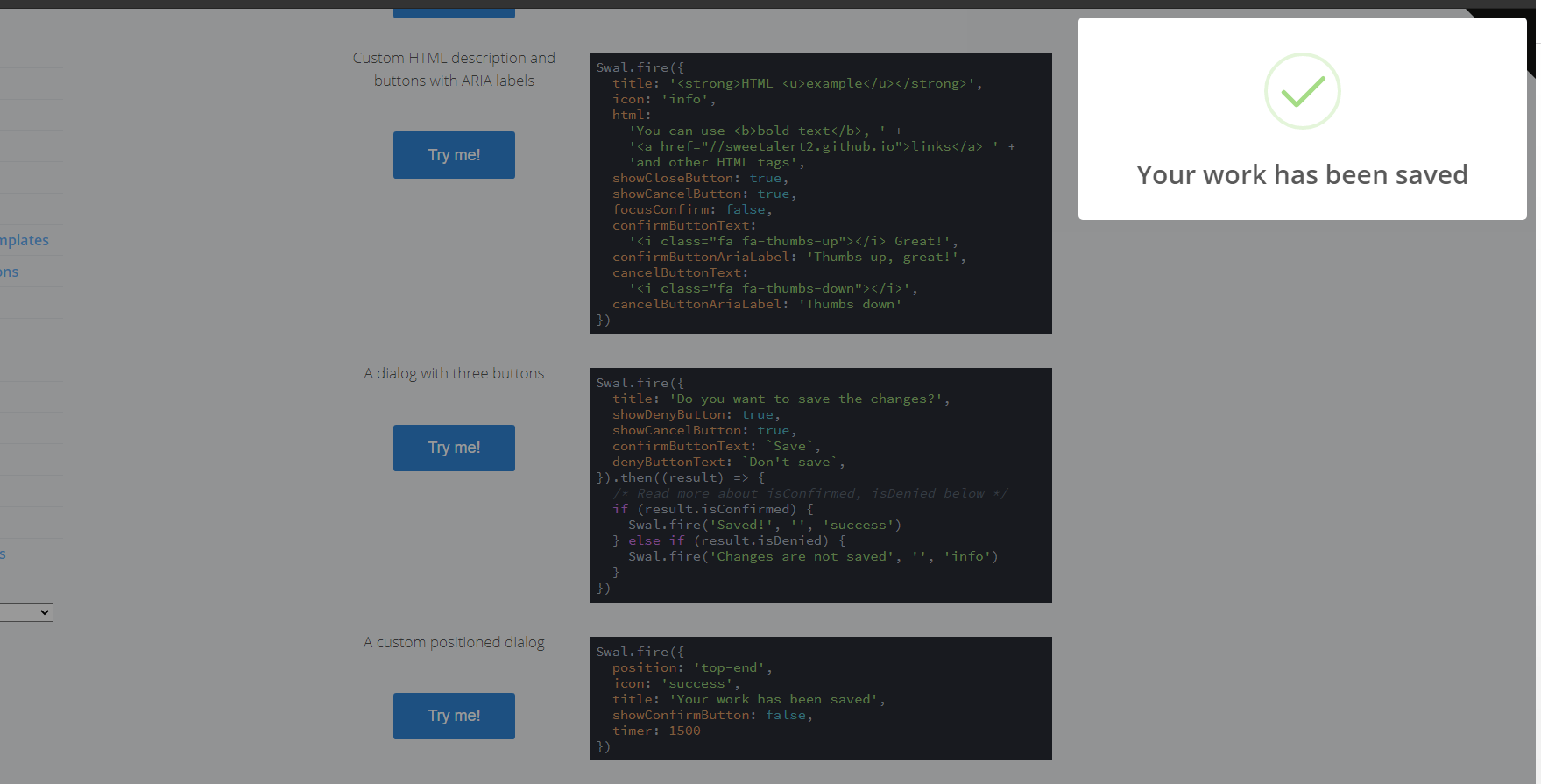


**Figura 2. Materialize. Fonte: MATERIALIZE.**

### 1.7.8. SweetAlert2 10.12.5

O SweetAlert2 é um ​*framework javascript* de código aberto*,* ​sua finalidade envolve otimizar a emissão de alertas ao usuário durante a utilização de aplicações web. (SWEETALERT2, 2021).

Todos os alertas e mensagens do SweetAlert2 podem conter um ícone central para descrever o evento da mensagem um título e ou uma mensagem (Figura3).



**Figura 3. Materialize. Fonte:** **SWEETALERT2.GITHUB.IO.**

## 1.8. Back-end

### 1.8.1. Node.js 14.15.1

Node.js é um software de [código aberto](https://pt.wikipedia.org/wiki/Open-source), [multiplataforma](https://pt.wikipedia.org/wiki/Cross-platform), que executa códigos [*JavaScript*](#_1.7.6._JavaScript)no backend/servidor e frontend/interface (dependendo apenas das bibliotecas e dos frameworks usados), baseado no [V8](https://pt.wikipedia.org/wiki/V8_(JavaScript)) interpretador de [*JavaScript*](#_1.7.6._JavaScript)em [C++](https://pt.wikipedia.org/wiki/C%2B%2B) do Google, mantido pela fundação Node.js em parceria com a [Linux Foundation](https://pt.wikipedia.org/wiki/Linux_Foundation) (NodeJS, 2021).

### 1.8.2. AWS-SDK 2.792.0

O AWS-SDK é um pacote de ferramentas fornecida pela Amazon para integração de aplicações NodeJs (item 1.8.1) com as ferramentas da companhia entre elas o AmazonS3 e o AmazonCloudWatch.

### 1.8.3. Express 4.17.1

O Express é um framework para aplicativo da web do Node.js mínimo e flexível que fornece um conjunto robusto de recursos para aplicativos web e móvel

(ExpressJs, 2021).

#### 1.8.3.1. DotEnv 8.2.0

DotEnv é uma biblioteca [*JavaScript*](#_1.7.6._JavaScript) que possibilita a utilização de um arquivo com variáveis de ambiente acessíveis por meio da utilização de ‘proccess.env’ como variável assim facilitando a publicação de uma aplicação em diversos ambientes de desenvolvimento como testes, desenvolvimento, produção entre outros.

#### 1.8.3.2. Cors 2.8.5

*Cross-Origin Resource Sharing* ou CORS é um mecanismo que permite que recursos restritos em uma página da web sejam recuperados por outro domínio fora do domínio ao qual pertence o recurso que será recuperado (Wikipédia, 2021).

A Biblioteca Cors implementa os recursos citados acima para aplicações em NodeJs.

#### 1.8.3.3. Mime-Types 2.1.27

A Biblioteca Mime-Types fornece recursos para capturar e ou alterar as extensões de arquivos em tempo real comumente utilizado para verificar extensões de arquivos no momento de *upload* e garantir que estejam eu formato aceitado pelo servidor.

### 1.8.4. Sequelize 6.3.5

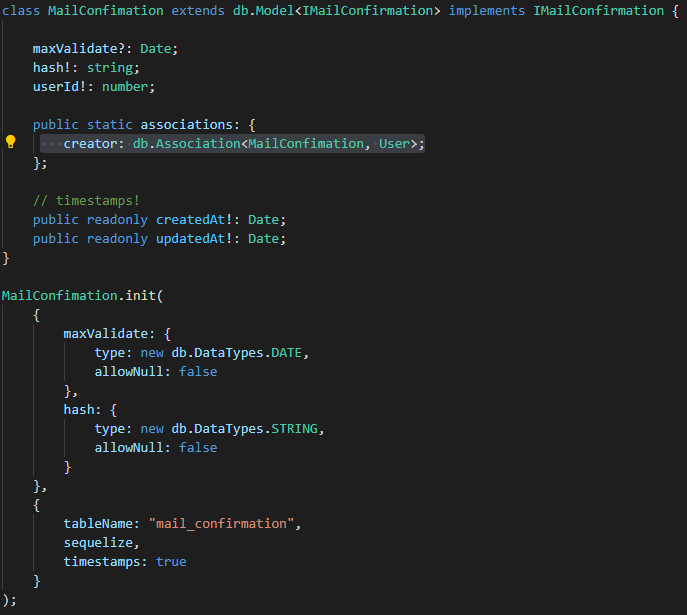
O Sequelize é um ORM é uma ferramenta utilizada para fazer a abstração entre a orientação a objetos do NodeJs e a camada de base de dados relacional proporcionando que consultas, alterações e inserções de dados sejam feitas por meio de objetos JSON e não por queries diretamente escritas em uma linguagem pré-definida.

#### 1.8.4.1. MySQL 2.2.5

MySql2 é uma biblioteca *javascript*para NodeJs que possibilita a comunicação de outras bibliotecas com os bancos de dados relacionais MySQL.

#### 1.8.4.2. Criação de tabelas com Sequelize

Para a construção das tabelas no banco de dados relacional MySQL utilizou-se de método de construção disponível na biblioteca Sequelize ORM (item1.8.4). Ao utilizar este método de construção temos a classe de manipulação do objeto diretamente relacionado com a tabela na base de dados, a forma de construção segue o padrão de objeto JSON onde pode-se informar o nome do atributo, se ele pode ser nulo, valor padrão entre outros parâmetros pertinentes a tabela que deve ser criada (Figura 4). É importante citar que para a correta criação da tabela é necessária uma interface do objeto relacional com os atributos necessários.



**Figura 4.Construção de tabelas com Sequelize ORM. Fonte: Elaborada pelo autor.**

#### 1.8.4.3. Relacionamento de tabelas com Sequelize

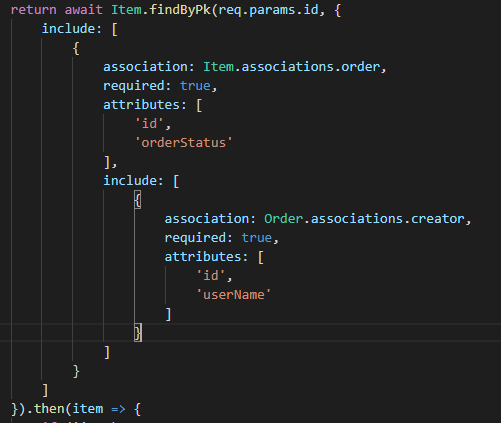
A biblioteca Sequelize (item1.8.4) permite que o relacionamento de tabelas seja feito de forma mais descritiva assim favorecendo o entendimento do relacionamento, este método de criação de relacionamento também proporciona a inserção de atributos como chave de relacionamento e ações como ‘onDelete’ de forma facilmente legível (Figura 5).



**Figura 5.Relacionamento de tabelas com Sequelize ORM. Fonte: Elaborada pelo autor.**

#### 1.8.4.3. Construção de queries com Sequelize

Como citado no (item 1.8.4) a biblioteca Sequelize proporciona a construção de consultas em sua base de dados em uma camada de abstração onde não é necessário escrever queries onde em um objeto Json se torna a consulta para diferentes bases de dados (Figura 6).



**Figura 6.Query com Sequelize ORM. Fonte: Elaborada pelo autor.**

### 1.8.5. Passport

Passaporte é middleware de autenticação para Node.js. Extremamente flexível e modular, o Passport pode ser inserido de forma discreta em qualquer aplicativo da web baseado no Express. Um conjunto abrangente de estratégias oferece suporte à autenticação usando um nome de usuário e senha, Facebook, Twitter e muito mais. (PASSPORTJS, 2021)

#### 1.8.5.1 Json Web Token 8.5.1

O JSON Web Token é um padrão da Internet para a criação de dados com assinatura opcional e/ou criptografia cujo *payload* contém o JSON que afirma algum número de declarações. Os *tokens* são assinados usando um segredo privado ou uma chave pública/privada.

#### 1.8.5.2 Passport JWT 4.0.0

Passport JWT é uma extensão para a biblioteca Passport (item 1.8.5), que possibilita a integração com JWT (item 1.8.5.1).

### 1.8.6. NodeMailer 6.4.16

Nodemailer é um módulo para aplicativos Node.js para permitir um envio de e-mail fácil como bolo. O projeto começou em 2010, quando não havia uma opção sensata para enviar mensagens de e-mail, hoje é a solução que a maioria dos usuários de Node.js recorrem por padrão (Nodemailer, 2021).

### 1.8.7. Multer 1.4.2

Multer é um modulo javascript para ser usado como *midleware* em requisições com estrutura ‘multipart/form-data’ possibilitando o upload de arquivos de forma simples e prática.

#### 1.8.7.1. MulterS3 2.9.0

Multer S3 é uma extensão para o módulo Multer (item 1.8.7) que possibilita upload de arquivos diretamente para a Amazon S3.

## 1.9. Banco de dados

### 1.9.1 MySQL

O MySQL é um sistema de gerenciamento de banco de dados, que utiliza a linguagem SQL como interface. É atualmente um dos sistemas de gerenciamento de bancos de dados mais populares da Oracle Corporation, com mais de 10 milhões de instalações pelo mundo (MySQL, 2021).

# 2. Trabalhos Correlatos Existentes

Pendente.

# 3. Requisitos

Esta seção apresenta os requisitos funcionais e não funcionais utilizados para o desenvolvimento do projeto.

## 3.1. Requisitos Funcionais

Esta seção apresenta os requisitos funcionais utilizados para o desenvolvimento do projeto.

RF001 – O sistema deve manter o cadastro de filmes.

RF002 – O sistema deve manter o cadastro de pessoas.

RF003 – O sistema deve manter o cadastro de assinaturas.

RF004 – O sistema deve manter o cadastro de classificações de filmes.

RF005 – O sistema deve manter o cadastro de categorias de filmes.

RF006 – O sistema deve controlar o acesso por perfis de usuário.

RF007 – O sistema deve controlar o acesso aos filmes através das assinaturas.

RF008 – O sistema somente pode ser acessado por usuários cadastrados e autenticados.

RF009 – O sistema deve permitir solicitar novos filmes.

RF010 – O sistema deve permitir a qualquer pessoa possa se cadastrar na plataforma.

## 3.2. Requisitos Não Funcionais

Esta seção apresenta os requisitos não funcionais utilizados para o desenvolvimento do projeto.

RNF001 – O sistema deve permitir busca de filmes por título e ou descrição.

RNF002 – O sistema deve permitir ao usuário visualizar seu perfil.

RNF003 – O sistema deve permitir contatos de usuários cadastrados ou não.

RNF004 – O sistema deve permitir ver usuários cadastrados (perfil admin).

RNF005 – O sistema ser responsivo e funcionar em diversas plataformas.

RNF006 – O sistema deve permitir ao usuário deixar mensagens de contato.

RNF007 – O sistema deve permitir ao usuário alterar sua senha.

# 4. Diagramas UML

A UML é uma linguagem-padrão para a elaboração da estrutura de projetos de software. Ela poderá ser empregada para a visualização, a especificação, a construção e a documentação de artefatos que façam uso de sistemas complexos de software (Wikipédia, 2021).

## 4.1 Diagrama de casos de uso

Pendente.

**Figura 7.Diagrama de casos de uso. Fonte: Elaborada pelo autor.**

## 4.2 Diagrama de classes

Pendente.

**Figura 8.Diagrama de classes. Fonte: Elaborada pelo autor.**

## 4.3 Diagrama de atividades

Pendente.

**Figura 9.Diagrama de atividades. Fonte: Elaborada pelo autor.**

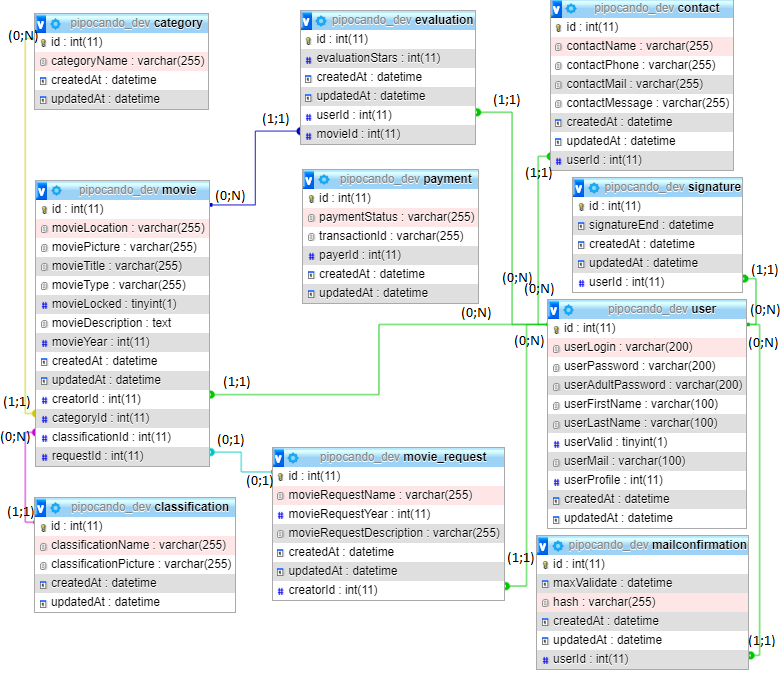
# 5. Modelagem de dados

Esta seção apresenta o modelo do banco de dados relacional e seu dicionário de dados.

## 5.1. Modelo Entidade Relacionamento

O Modelo Entidade Relacionamento (MER) ou somente Entidade Relacionamento (ER) é utilizado para representar o banco de dados. Neste modelo são definidas entidades, relacionamentos, atributos, chaves primárias e chaves estrangeiras responsáveis por manter a integridade das informações presentes no banco de dados (​SMARTDRAW​, 2021).

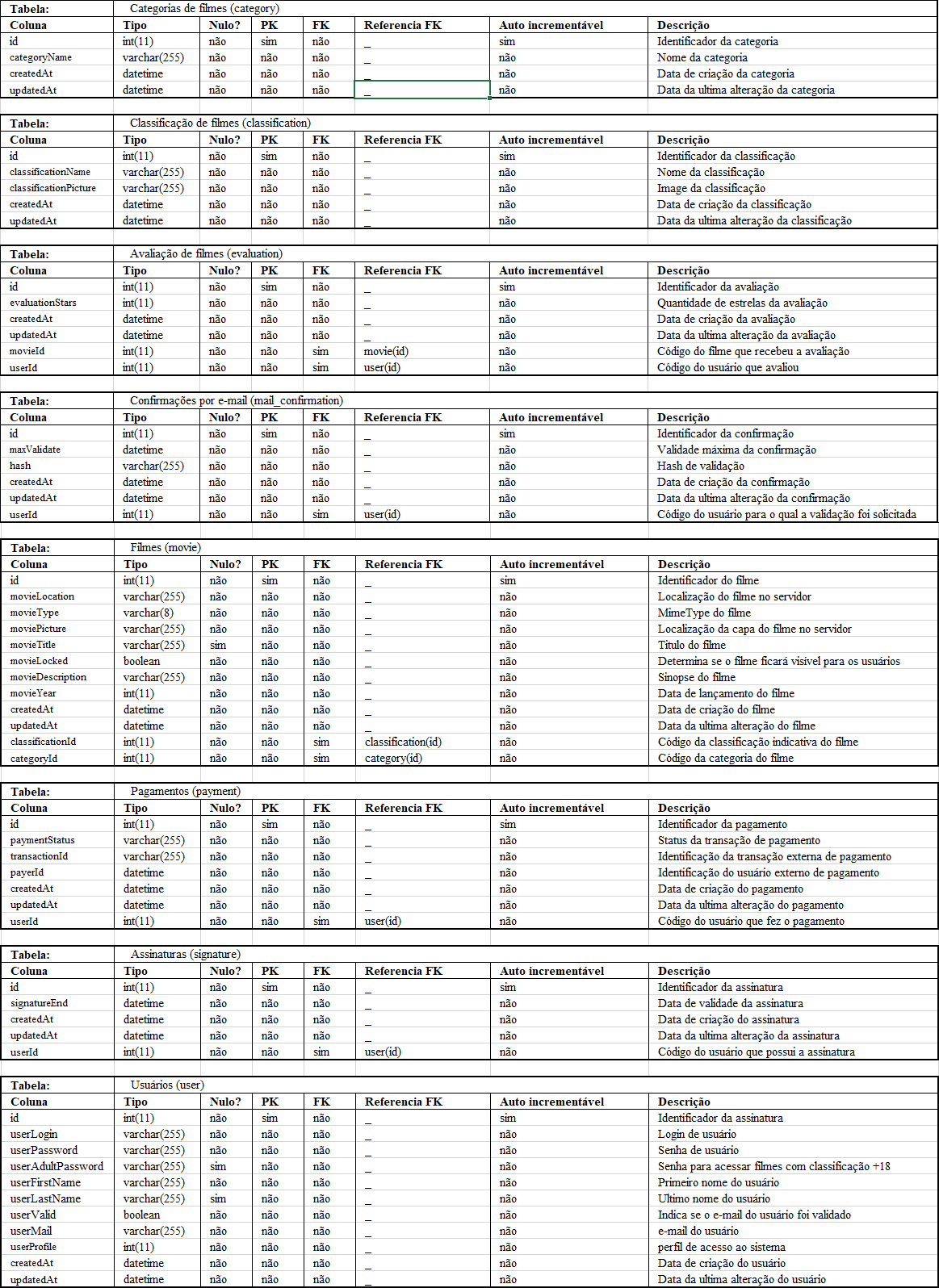
O MER da aplicação apresenta as tabelas e os relacionamentos do banco de dados. As informações das tabelas são: colunas, chaves, cardinalidade e índices (Figura 4).



**Figura 10. MER. Fonte: Elaborada pelo autor.**

## 5.2. Dicionário de dados

Esta seção apresenta o dicionário de dados do banco de dados da aplicação, utilizado para descrever cada atributo, valor suportado, tamanho limite, se o campo oferece auto incremento, restrições de chave e nulidade.



**Figura 11. Dicionário de dados. Fonte: Elaborada pelo autor.**

**Figura 12. Dicionário de dados. Fonte: Elaborada pelo autor.**

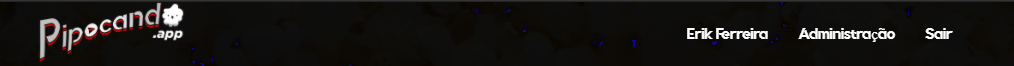
**Figura 13. Dicionário de dados. Fonte: Elaborada pelo autor.**

# 6. Projeto da Interface

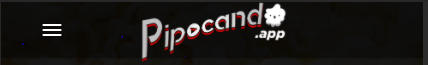
A interface de usuário foi desenvolvida com intuito de facilitar o acesso do usuário as informações pertinentes ao seu perfil.

## 6.1. Header.

O *header* (cabeçalho) da página fornece acesso direto as funcionalidades disponíveis para o usuário conforme o perfil, por padrão apresenta o nome do usuário e a opção de sair (Figura 14), caso o acesso seja feito via dispositivo móvel as opções ficam disponíveis através de um botão de menu e não diretamente dispostas (Figura 15).



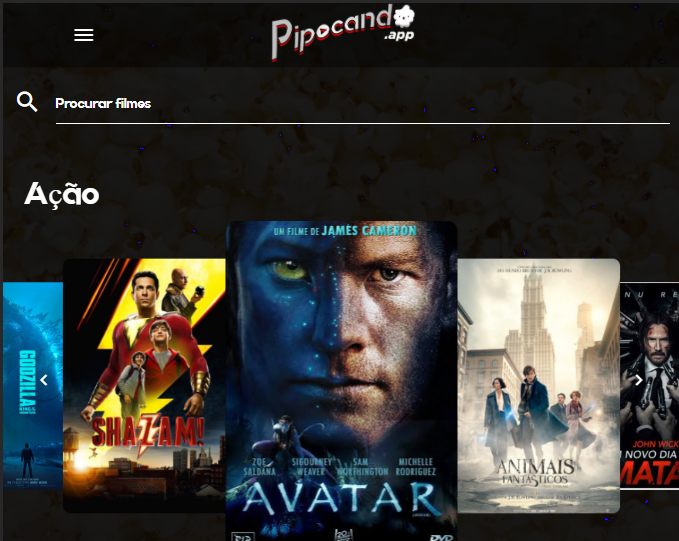
**Figura 14. Header em desktops. Fonte: Elaborada pelo autor.**



**Figura 15. Header em dispositivos móveis. Fonte: Elaborada pelo autor.**

## 6.2. Visualização de todos os filmes.

A página de visualização de filmes se divide por categorias e ordena a exibição dos títulos por data da última alteração no título todos os usuários cadastrados tem acesso a esta página, independentemente se tenham ou não assinaturas ativas. Ainda na página de visualização existe a barra de pesquisa universal que possibilita aos usuários buscarem algum título na plataforma (Figura 16).

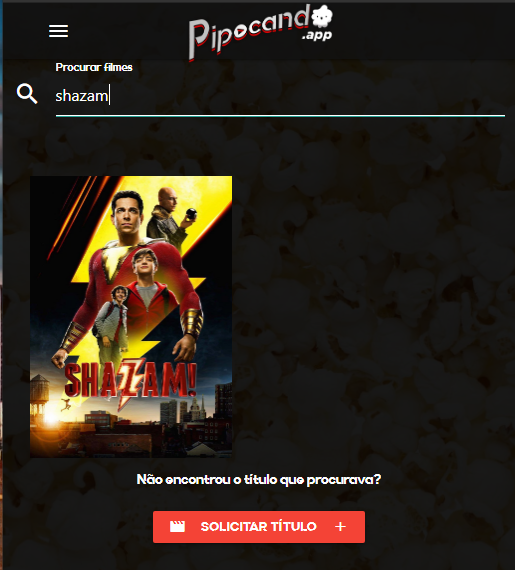


**Figura 16.Página de exibição de filmes. Fonte: Elaborada pelo autor.**

## 6.3. Pesquisa de filmes.

A página de pesquisa aparece sempre que algum valor for digitado na barra de pesquisa presente na tela de visualização de filmes (item 6.2), sempre aparecerá já com os resultados correspondentes ao valor da pesquisa e desaparece assim que o valor na barra for vazio.

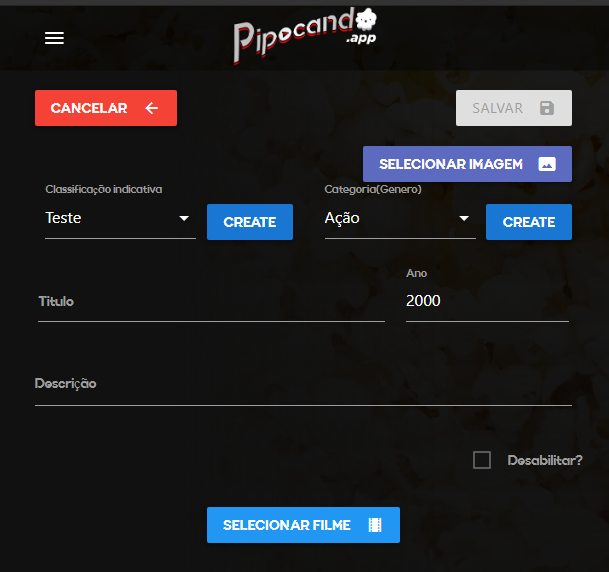
Ainda na tela de pesquisa de filmes pode se encontrar o botão de solicitação de filmes (Figura 17).



**Figura 17.Página de pesquisa de filmes. Fonte: Elaborada pelo autor.**

## 6.4. Cadastro de filmes.

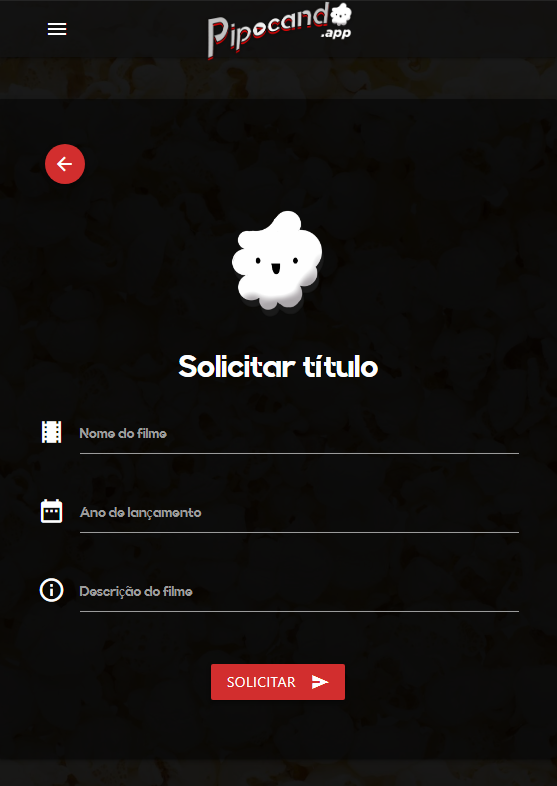
A página de cadastro de filmes é acessada através da página de Gestão de filmes (item 6.6). E fornece ao usuário (Admin) os campos necessários para o cadastro de um título na plataforma. Dentre elas estão criação e ou seleção de uma categoria de filmes, criação e ou seleção de uma nova classificação de filmes, título do filme, imagem de capa, descrição do filme, ano de lançamento, disponibilidade do título e a seleção do filme que deve ter formato MP4 (Figura 18).



**Figura 18.Página de cadastro de filmes. Fonte: Elaborada pelo autor.**

## 6.5. Solicitação de filmes.

Todos os usuários cadastrados na plataforma que possuam uma assinatura ativa podem solicitar filmes o acesso à página de solicitação se dá através da página de pesquisa (item 6.3), assim quando o filme procurado não for encontrado o usuário pode solicitar inserindo as informações necessárias que são nome do filme, data de lançamento e uma descrição opcional (Figura 19).



**Figura 19.Página de solicitação de filmes. Fonte: Elaborada pelo autor.**

## 6.6. Gestão de filmes.

A página de gestão de filmes é acessada através da página de administração que está disponível no header das páginas (item 6.1) caso o usuário possua perfil de administrador, e apresenta algumas informações básicas sobre o filme como nome e data de cadastro, assim como as opções de editar, excluir, visualizar títulos ou cadastrar um novo (figura 20).



**Figura 21.Página de gestão de filmes. Fonte: Elaborada pelo autor.**

## 6.9. Visualização de filme.

A página de visualização de filmes pode ser acessada de diversas formas, sendo algumas delas através da página de pesquisa (item 6.3), através da página de visualização geral (item 6.2) ou através da página de gestão de filmes (item 6.6).

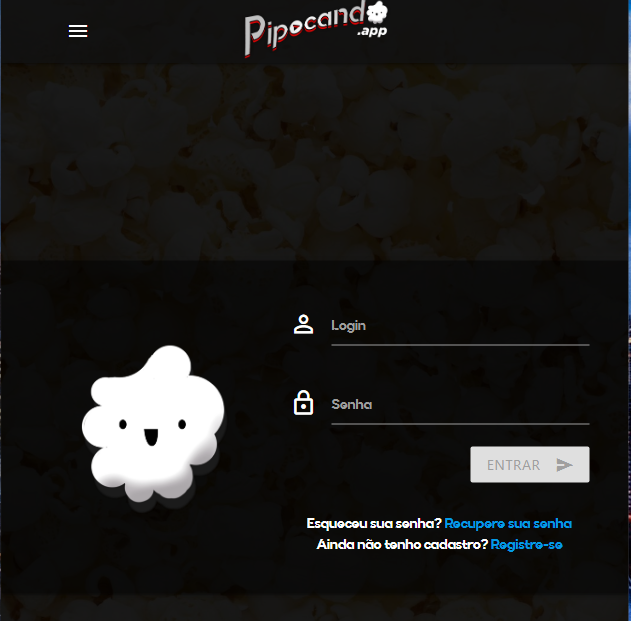
Disponíveis na página de visualização de filmes estão o nome do filme, data de lançamento do filme nos cinemas, descrição do filme, avaliação (CNDB) do filme, imagem de capa e o player de reprodução do filme (item 1.7.4) para que seja executado (Figura 22).



**Figura 22.Página de exibição de filmes. Fonte: Elaborada pelo autor.**

## 6.10. Tela de login.

A tela de login é a primeira tela apresentada ao usuário tendo em vista que não foi criada uma home page para a aplicação, dispõe links diretos para as telas de cadastro (item 6.13) e tela de recuperação de senha (item 6.11) e de campos básicos para o login sendo eles usuário e senha (Figura 23).



**Figura 23.Página de login. Fonte: Elaborada pelo autor.**

## 6.11. Tela de pedido de recuperação de senha

Através desta o usuário pode solicitar uma hash para executar a alteração de sua senha caso tenha esquecido ou somente a queira modificar.

Esta tela dispõe de um link para a tela de registro de usuário e um campo que o usuário deve inserir seu e-mail informado no cadastro (Figura 24).



**Figura 24.Página de solicitação de mudança de senha. Fonte: Elaborada pelo autor.**

## 6.12. Tela de alteração de senha.

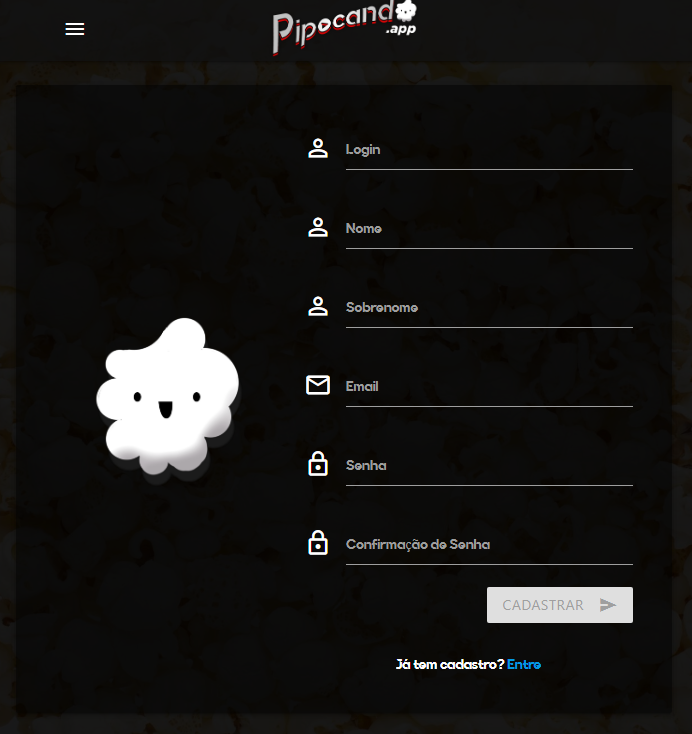
A alteração de senha é acessível através do e-mail enviado ao usuário após a solicitação de alteração de senha, e dispõe de campos de senha e confirmação de senha (Figura 25).



**Figura 24.Página de alteração de senha. Fonte: Elaborada pelo autor.**

## 6.13. Tela de Cadastro.

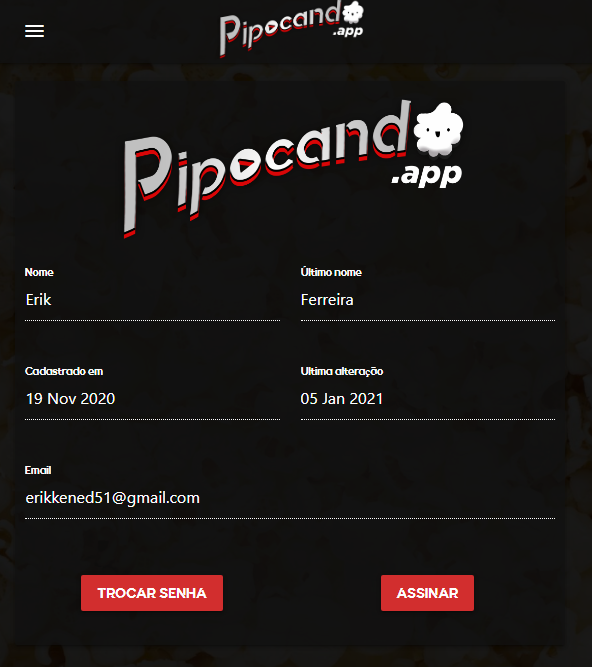
A tela de cadastro dispõe dos campos básicos para um usuário estar cadastrado na plataforma sendo eles login que é único e será validado, nome de usuário, sobrenome do usuário como campo opcional, e-mail que também será único e validado, senha e confirmação (Figura 25).



**Figura 25.Página de cadastro. Fonte: Elaborada pelo autor.**

## 6.14. Tela de visualização de usuário.

Usuários autenticado podem clicar em seu nome na barra de navegação (item 6.1) e visualizar o seu perfil podendo alterar seus dados cadastrais e solicitar alteração de senha, caso o usuário não possua assinatura ativa na página de perfil ficará disponível um botão para que o usuário faça sua assinatura (Figura 26).



**Figura 26.Página de visualização de perfil. Fonte: Elaborada pelo autor.**

## 6.15. Tela de pagamento.

Sempre que um usuário que não possui assinatura valida tentar acessar uma área que precise de assinatura como os itens 6.9 e 6.5, será redirecionado para a página de pagamento, como o sistema não foi integrado com uma plataforma de pagamento de modo a ter uma cobrança em formato transparente, a tela apenas conta com um link de cobrança e cabe ao gestor do sistema conferir a assinatura com o seu meio de preferência, nesse caso foi utilizado a *pagSeguro* como plataforma de pagamentos.



**Figura 27.Página de pagamentos. Fonte: Elaborada pelo autor.**

# 7. Resultados Obtidos

Após o desenvolvimento obteve-se uma aplicação funcional para registro e manutenção de filmes, incluindo o controle de acesso ao conteúdo por meio de assinaturas.

O sistema possibilita a escolha entre armazenar os dados de conteúdo localmente ou em nuvem ficando a escolha do gerenciador de conteúdo onde armazenar os dados.

Todos os requisitos listados foram atendidos. A primeira versão do sistema encontra-se disponível para uso.

# 8. Considerações Finais e Trabalhos Futuros

Após o seu completo desenvolvimento o projeto aqui citado mostrou-se bastante eficaz no controle de acesso e gestão do conteúdo, não foi envolvido no sistema uma plataforma de pagamento tendo apenas uma tabela para registro manual dos pagamentos. Este é um dos pontos que podem ser melhorados adicionar uma plataforma de pagamentos diretamente na aplicação.

Uma das maiores dificuldades encontradas foi a requisição do stream para a Amazon por meio do backend, pois é preciso solicitar um pequeno pedaço dos filmes por vez para garantir a segurança das mídias disponíveis no site.

De uma forma geral para facilitar a aplicação das regras de controle de acesso à mídia é recomendado o uso de um servidor local onde toda a interação da aplicação com o conteúdo é controlada através da própria aplicação e não dependendo de terceiros.

# Referências Bibliográficas

Pendente.