

Documentação

OpFlix

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Alameda Barão de Limeira, 539 – São Paulo/SP |  |  |  | sp.senai.br |

Sumário

1. [Resumo 2](#_Toc533767843)

[Objetivos](#_Toc533767844)

1. [Descrição do projeto 2](#_Toc533767845)

[Resumo do projeto](#_Toc533767846)

1. [Modelagem de Software 3](#_Toc533767847)

[Modelo Lógico](#_Toc533767848)

[Modelo Físico](#_Toc533767849)

[Modelo Conceitual](#_Toc533767850)

[Cronograma](#_Toc533767851)

1. [Funcionalidades 5](#_Toc533767852)

[Web](#_Toc533767853)

[Mobile](#_Toc533767854)

1. [Protótipos 6](#_Toc533767855)

[Web](#_Toc533767856)

[Mobile](#_Toc533767857)

1. [Front-End 7](#_Toc533767858)
2. [Mobile 8](#_Toc533767859)
3. [Arquitetura do Projeto 9](#_Toc533767860)
4. [Referências 10](#_Toc533767861)

[Links](#_Toc533767862)

[Livros](#_Toc533767863)

Resumo

Objetivos

Este documento tem como objetivo demonstrar todas as funcionalidades da plataforma OpFlix, que visa demonstrar informações sobre o lançamento de filmes e séries.

Descrição do projeto

O OpFlix tem como objetivo a de oferecer informações sobre o lançamento de filmes e séries.

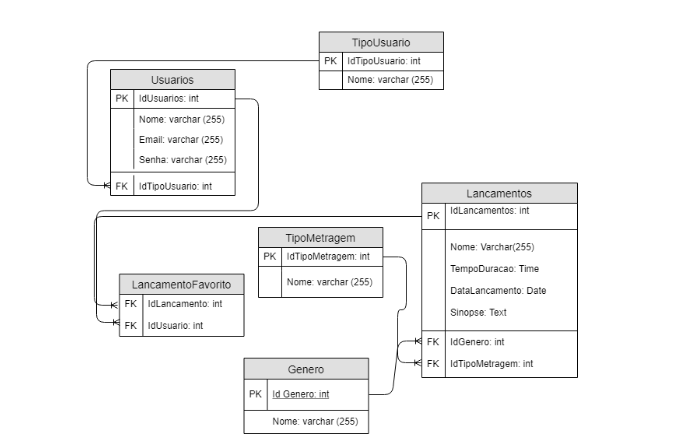
Resumo do projeto

O OpFlix oferece informações sobre o lançamento de filmes e séries e suas sinopses.

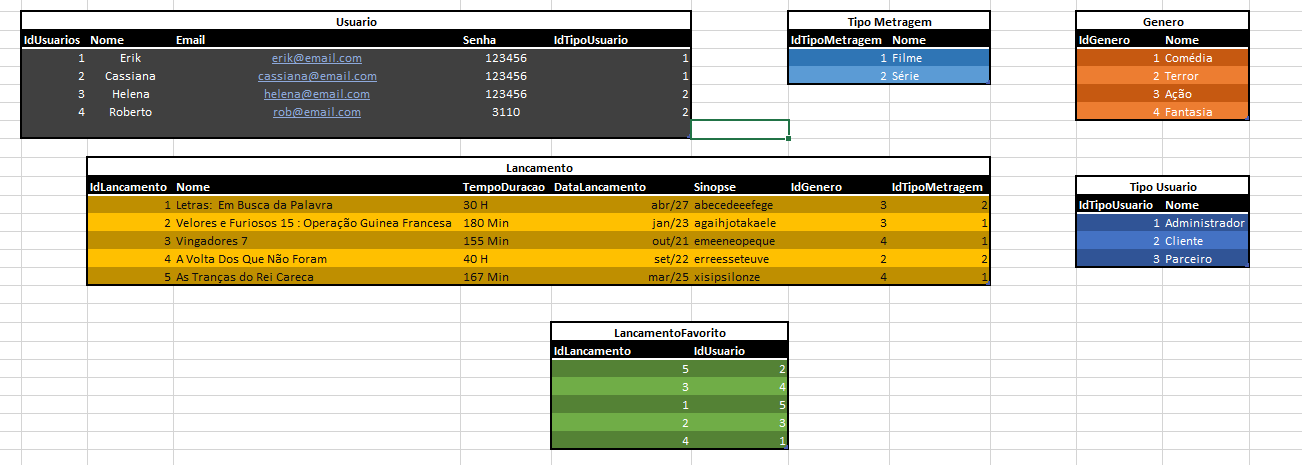
Modelagem de Software

Esta sessão tende a demonstrar o que será desenvolvido no banco de dados e quais são as suas representações.

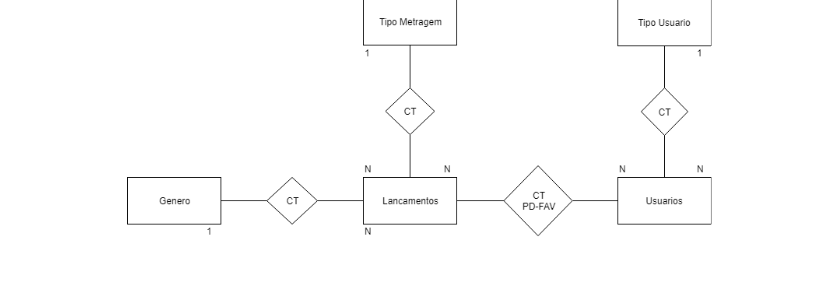
Modelo Lógico

O modelo Lógico é o modelo que mostra as entidades e seus relacionamentos e informações do tipo de dado de cada registro

Modelo Físico

O modelo Físico é o que mais se aproxima do Banco de Dados 

Modelo Conceitual

O modelo Conceitual que cria relações entre os elementos primários do Banco de Dados

Cronograma

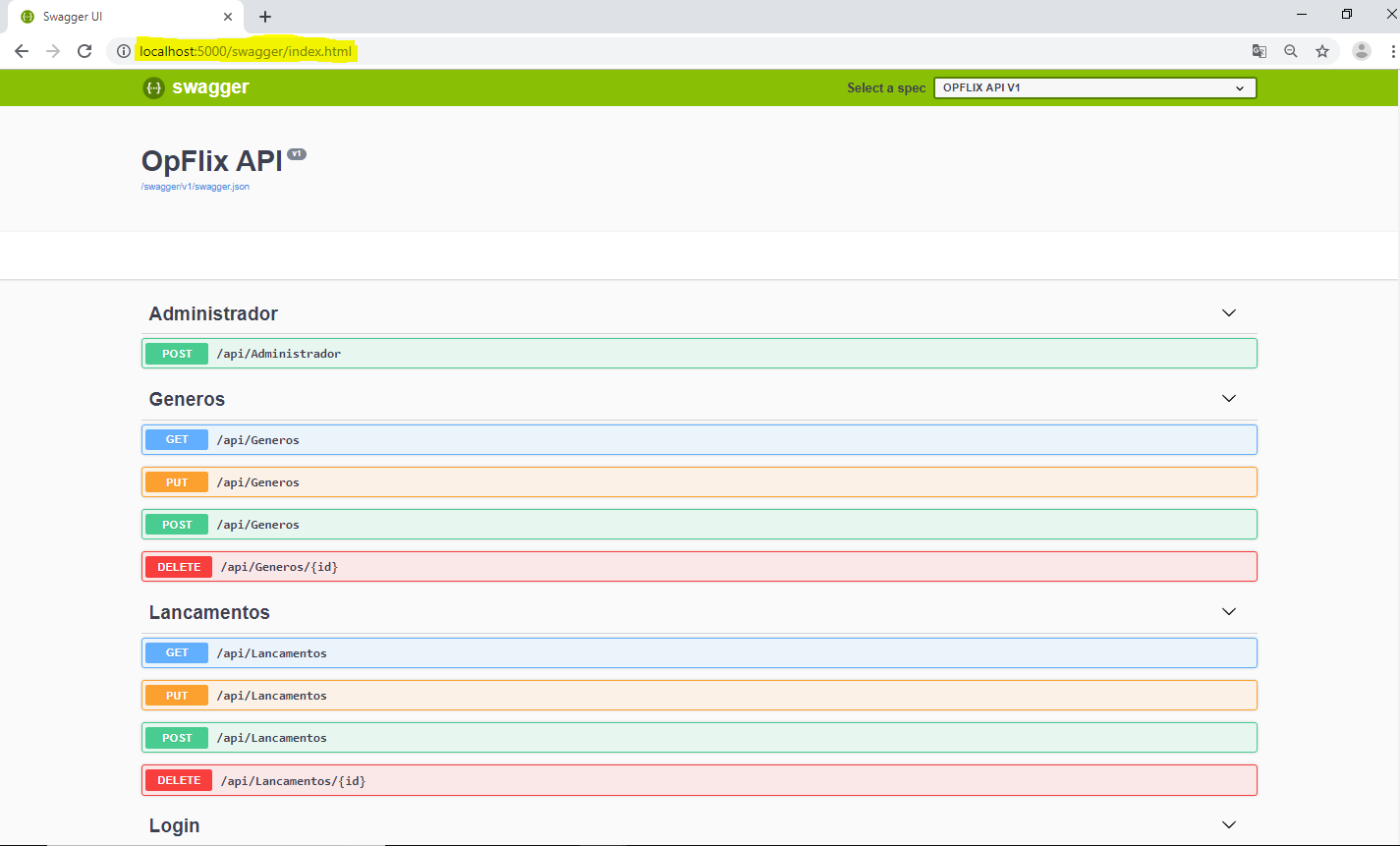
|  | Dia 1 | Dia 2 | Dia 3 | Dia 4 | Dia 5 | Dia 6 | Dia 7 | Dia 8 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Modelo Lógico** |  |  |  |  |  |  |  |  |
| **Modelo Físico** |  |  |  |  |  |  |  |  |
| Modelo Conceitual |  |  |  |  |  |  |  |  |

Funcionalidades

**Swagger**

**Para acessar o Swagger, primeiramente você deve adicionar o seguinte pacote: Swashbuckle.AspNetCore 4.0.1 .**

**Em seguida você deve executar a aplicação e acessar o seguinte link:** <http://localhost:5000/swagger/index.html>

****

Exportação e importação no Postman em outra máquina:

Para realizar a exportação basta clicar com o botão direito no projeto que você está fazendo e em seguida, apertar em “EXPORT” e escolher o local de exportação.

Para fazer a importação, aperte no botão “IMPORT” e escolha o local onde o arquivo do projeto.

**PACOTES DO NUGGET:**

**Já Vem com o projeto:**

Microsoft.AspNetCore.App

Microsoft.AspNetCore.Razor.Design

Microsoft.NETCore.App

**Swagger:**

Swashbuckle.AspNetCore 4.0.1

**EFCore:**

Microsoft.EntityFrameworkCore.SqlServer 2.1.11

Microsoft.EntityFrameworkCore.SqlServer.Design 1.1.6

Microsoft.EntityFrameworkCore.Tools 2.1.11

**Ferramentas Utilizadas:**

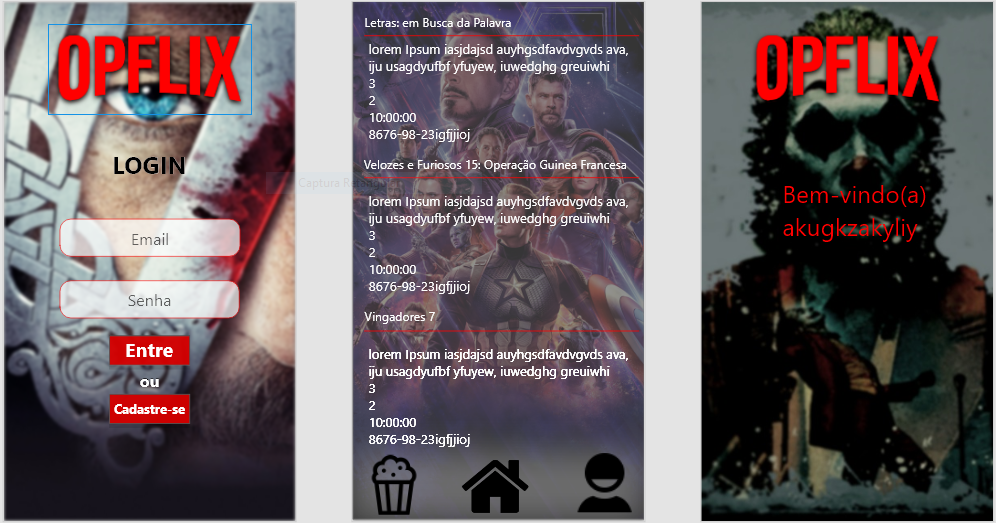
* Visual Studio 2017
* Postman
* Swagger
* SQL Server

**Realizar a criação de todo do banco e rodar o projeto de backend:**

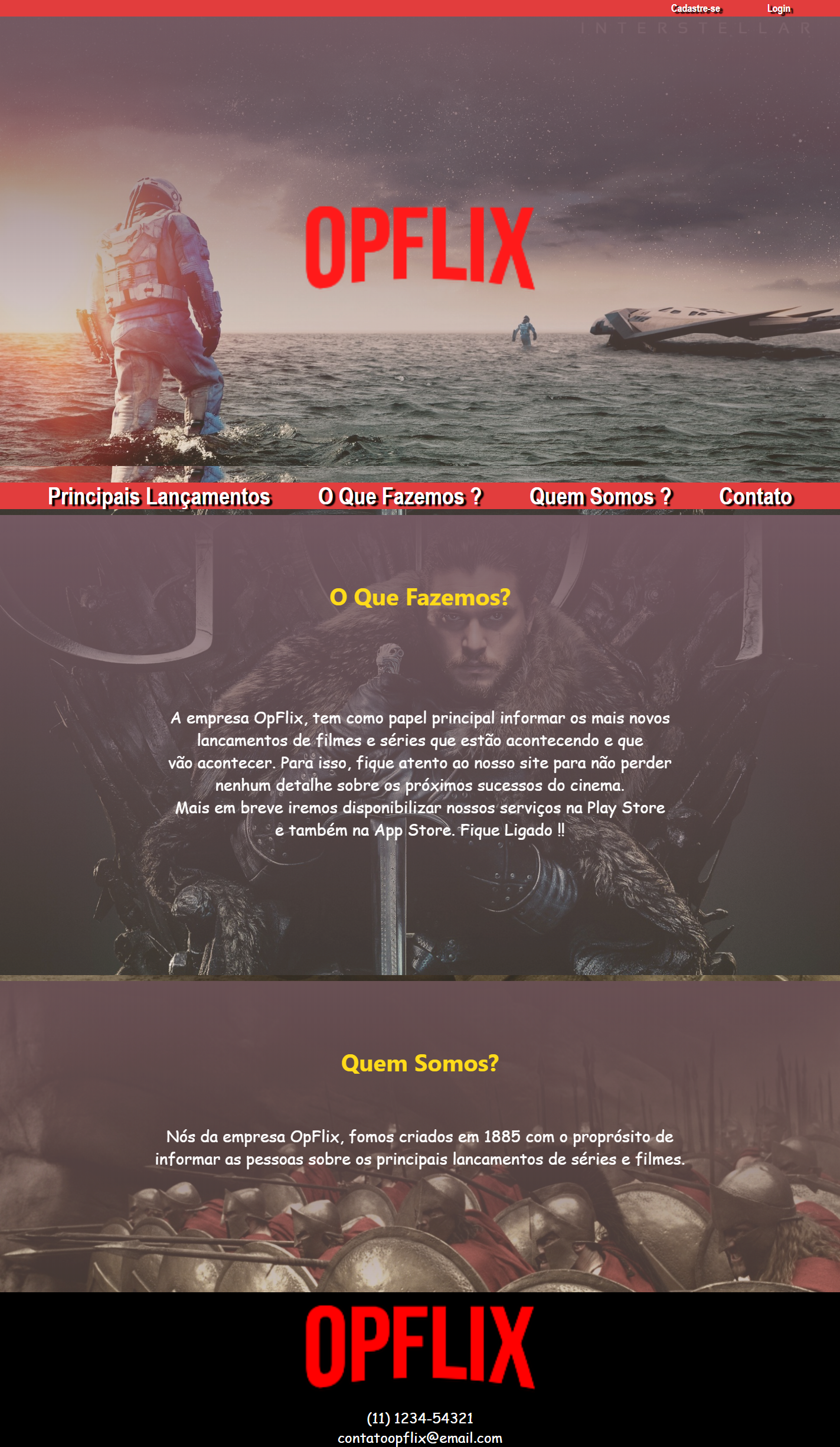
Execute todos os scripts do SQL, baixe todos os pacotes, que foram disponibilizados anteriormente e execute a aplicação.

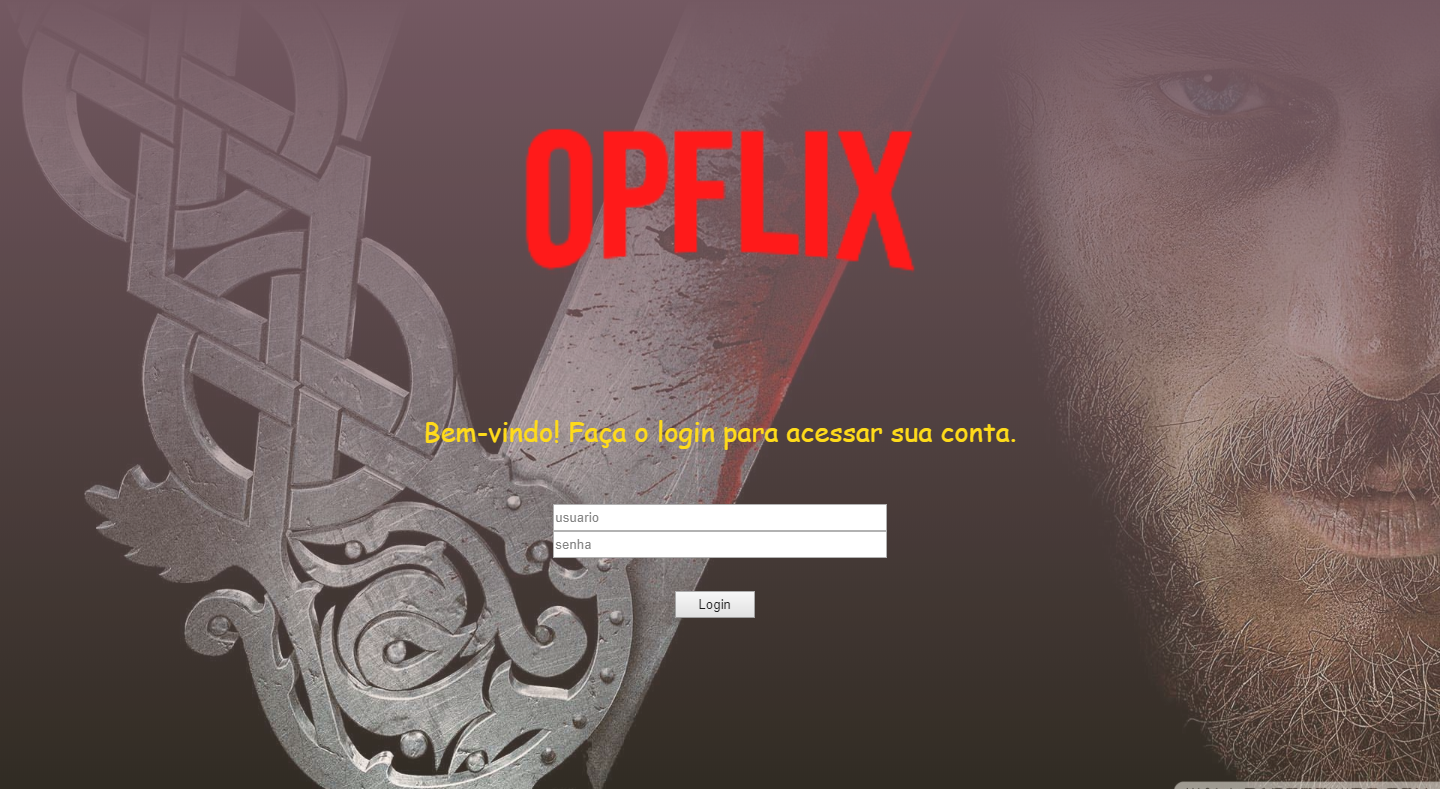
Protótipos

Mobile

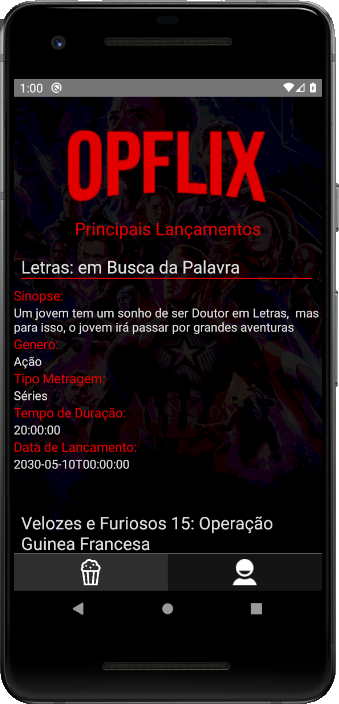


Front-End



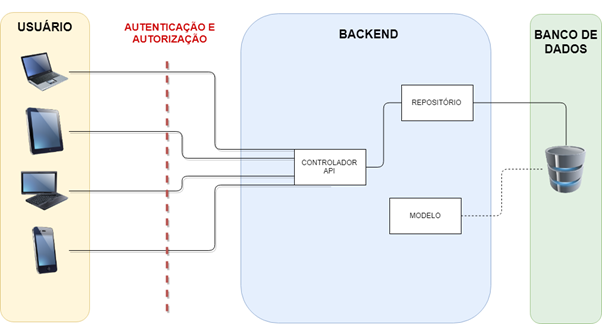


Mobile





Arquitetura do Projeto



Referências

Links

Livros