

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 sistema tem como objetivo demonstrar todas as funcionalidades da plataforma OpFlix, que visa demonstrar a uma plataforma onde você pode ver os últimos lançamentos do cinema e divulgação de sua página na Web por meio das Mídias Sociais.

Descrição do projeto

Esse sistema integrado com base na primeira parte em Banco de Dados tem como principais funcionalidades o cadastro de Filmes e Séries, definindo características distintas é a criação de Usuários.

Resumo do projeto

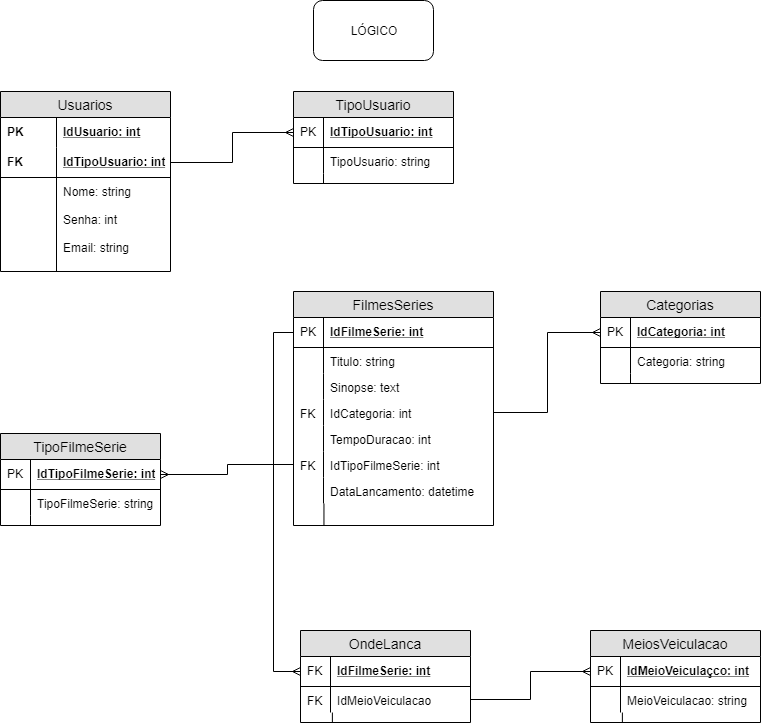
A primeira parte do projeto se baseia na criação de tabelas por meio do sistema de Banco de Dados com utilização no SQL Server Management é fácil visualização dos clientes.

Modelagem de Software

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

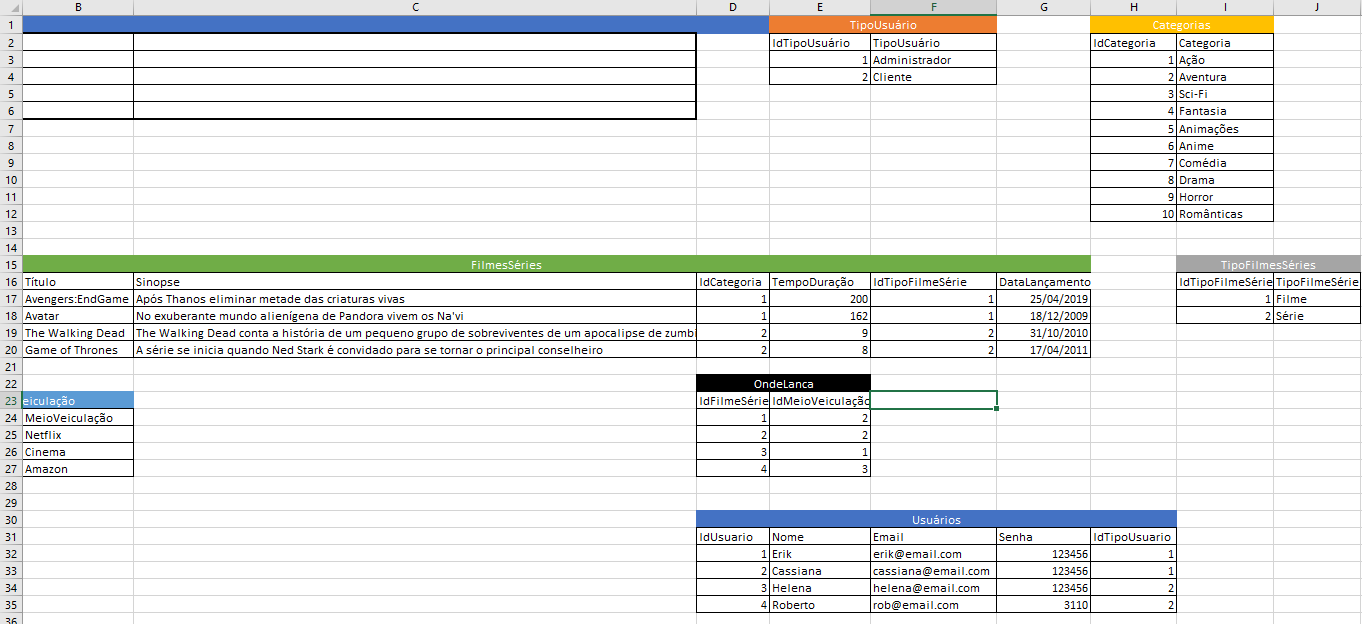
Modelo Lógico

O modelo lógico já leva em conta algumas limitações e implementa recursos como adequação de padrão e nomenclatura, define as chaves primárias e estrangeiras, normalização, integridade referencial, entre outras. Para o modelo lógico deve ser criado levando em conta os exemplos de modelagem de dados criados no modelo conceitual.



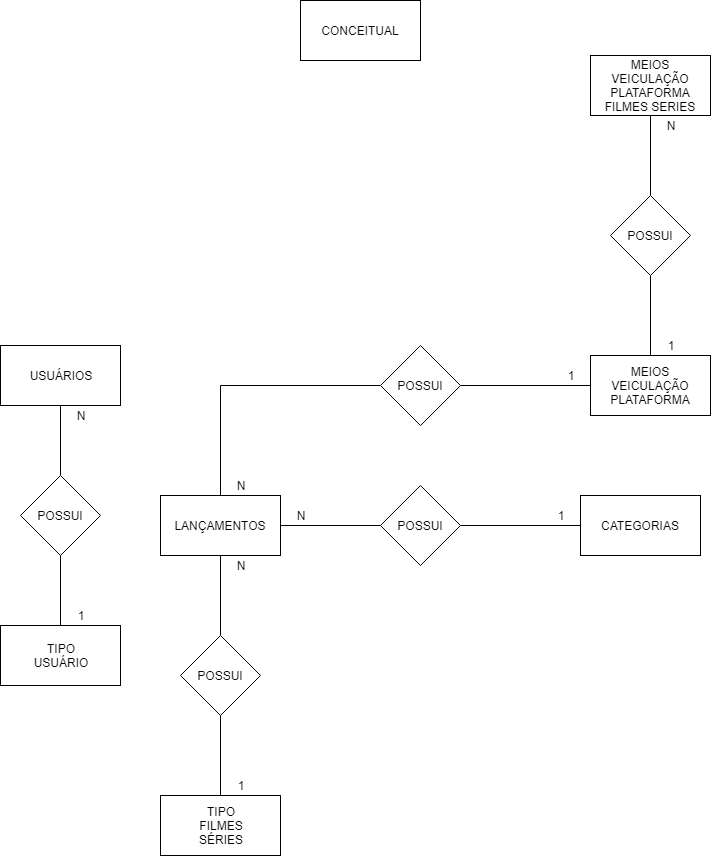
Modelo Físico

No modelo físico fazemos a modelagem física do modelo de banco de dados. Neste caso leva-se em conta as limitações impostas pelo SGBD escolhido e deve ser criado sempre com base nos exemplos de modelagem de dados produzidos no item anterior, modelo lógico.



Modelo Conceitual

A modelagem conceitual baseia-se no mais alto nível e deve ser usada para envolver o cliente, pois o foco aqui é discutir os aspectos do negócio do cliente e não da tecnologia. Os exemplos de modelagem de dados vistos pelo modelo conceitual são mais fáceis de compreender, já que não há limitações ou aplicação de tecnologia específica. O diagrama de dados que deve ser construído aqui é o Diagrama de Entidade e Relacionamento, onde deverão ser identificados todas as entidades e os relacionamentos entre elas. Este diagrama é a chave para a compreensão do modelo conceitual de dados.



Como executar

Abra os diagramas enviado por meio do Repositório no SQL Server Management na seguinte ordem M\_01\_BrunoHenriqueSilva\_DDL, M\_02\_BrunoHenriqueSilva\_DML, M\_03\_BrunoHenriqueSilva\_DQL.

Cronograma

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

Funcionalidades

Web

1. O administrador poderá cadastrar qualquer tipo de usuário (administrador ou cliente);

2. O administrador poderá cadastrar categorias (contendo nome);

3. O administrador poderá cadastrar o lançamento de um filme/série (contendo título, uma pequena sinopse, a categoria vinculada, tempo de duração, e se é filme ou série e a data de lançamento);

4. O cliente poderá visualizar todos os lançamentos publicados;

Administrador

5. Cadastro de novos usuários (pode deixar um endpoint público para cadastro de novos clientes e um outro endpoint aonde administradores cadastram outros administradores);

6. Criar, Atualizar e Deletar Lançamentos;

7. Cadastrar/Listar/Atualizar Categorias;

8. Cadastrar/Listar/Atualizar Plataformas/Mídias;

Para demonstração dessas e outras Funcionalidades é necessário o aplicativo de apoio chamado Postman onde na pasta Postman do Projeto OpFlix será disponibilizado por meio de GitHub no repositório como o link

<https://github.com/brunohesilva/2s2019-sprint-2-backend-OpFlix.git>

exportar a coleção é testar as funcionalidades listadas acima.

O acesso ao Postman se dá por meio do download do software por meio do site oficial, depois pegar a coleção exportada na Git descrito acima para poder fazer os testes.

O deploy do projeto é necessariamente no aplicativo Visual Studio onde na pasta com os projetos correspondentes abrir o projeto Senai.OpFlix.WebApi e apertar o as teclas Ctrl + F5 para iniciar a aplicação.

A importação do projeto no Postman se da por meio de abrir o aplicativo, depois clicar no botão Import depois no botão Chose Files e só selecionar a coleção disponibilizada no Git.

Os pacotes do Nugget utilizados foram Swashbuckle.AspNetCore 4.0.1, System.Data.SqlClient 4.6.1, Microsoft.EntityFrameworkCore.SqlServer 2.1.11, Microsoft.EntityFrameworkCore.SqlServer.Design 1.1.6, Microsoft.EntityFrameworkCore.Tools 2.1.11, Microsoft.AspNetCore.Authentication.JwtBearer 2.1.1, System.IdentityModel.Tokens.Jwt 5.5.0.

O projeto está arquitetado em um sistema Web API.

A criação do Banco de Dados se dá pela execução dos scripts enviados no SQL Server Management.

Mobile

1. O cliente poderá visualizar todos os lançamentos;

2. O cliente poderá realizar a ordenação por data de lançamento e/ou categoria;

Para a visualização da demais funcionalidades é necessário a utilização do software Android Studio pelos seguintes passos indo na opção configure AVD MANAGER, se você não tiver crie um emulador e clique no botão Play para iniciar o emulador

Para iniciação do projeto você precisa abrir CMD como Administrador, dar Cd na pasta do Projeto e depois colocar o seguinte comando C:\Users\\*CPF\*\AppData\Roaming\npm\react-native run-android

<https://github.com/brunohesilva/2s2019-sprint-4-mobile-OpFlix.git>

E necessária as seguinte dependecias na pasta do projeto por meio do CMD na pasta do projeto npm install react-navigation, npm install react-native-reanimated react-native-gesture-handler [react-native-screens@^1.0.0-alpha.23](mailto:react-native-screens@%5e1.0.0-alpha.23), npm install react-navigation-tabs, npm install --save react-navigation-stack, npm install jwt-decode

Colocar na pasta android do projeto no Visual Studio Code

android -> app -> build.gradle

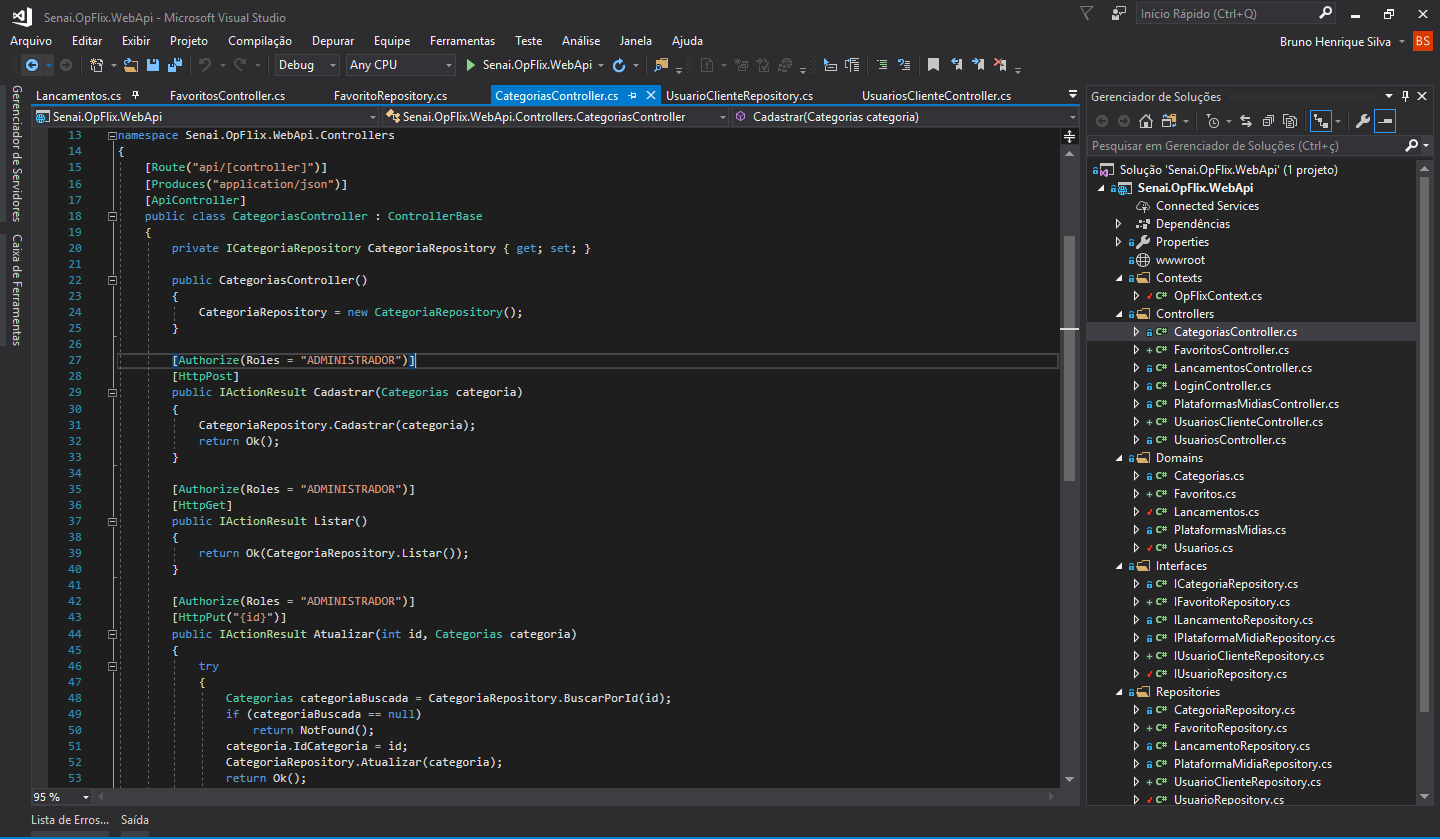
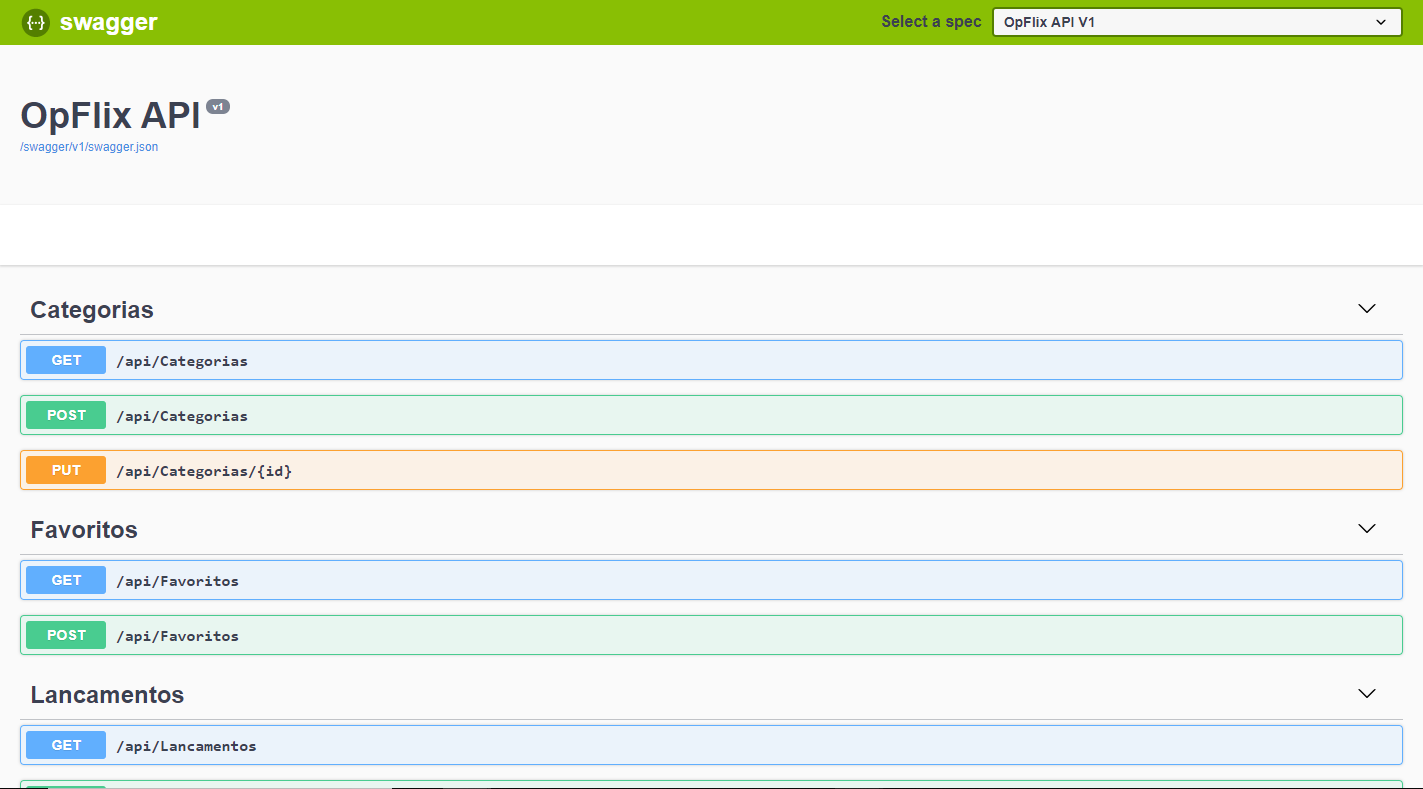
implementation 'androidx.appcompat:appcompat:1.1.0-rc01'

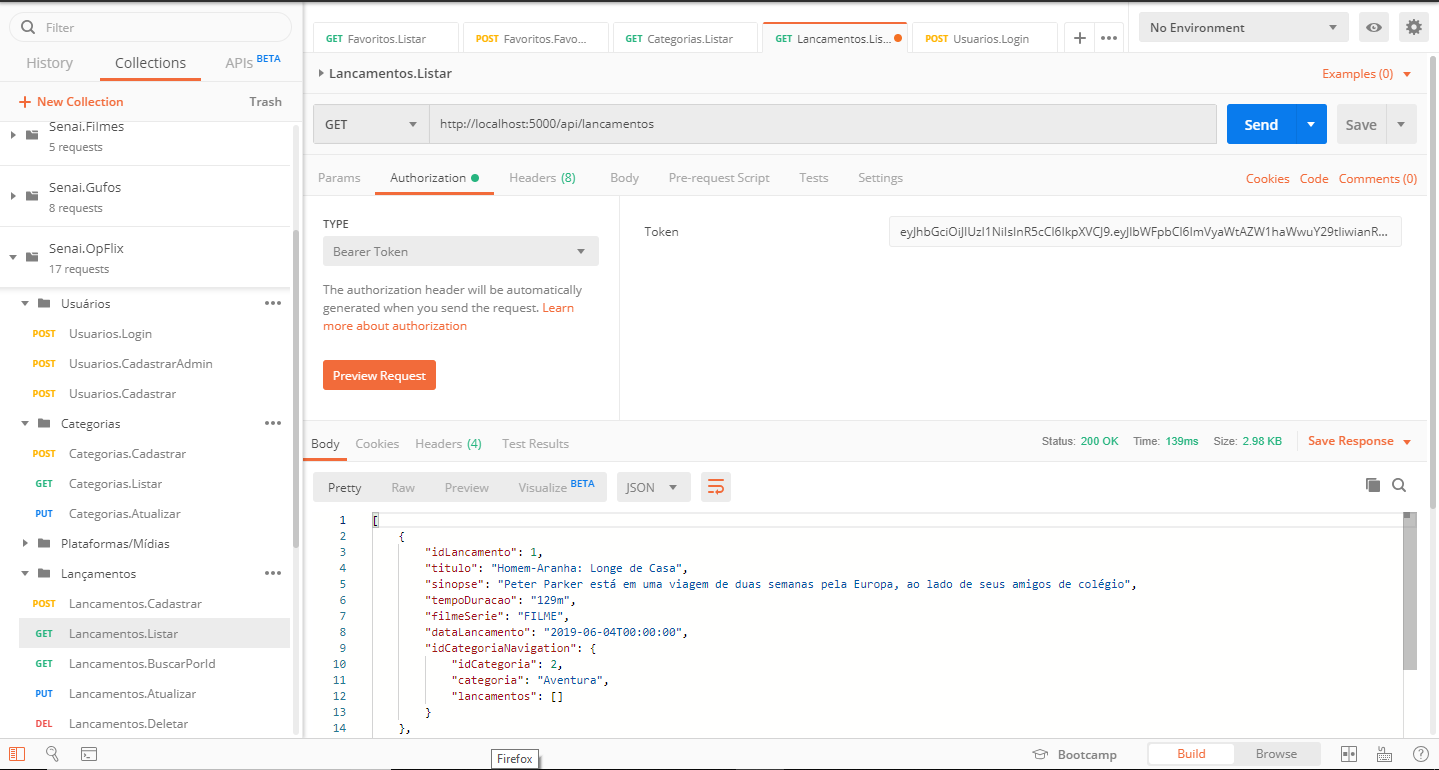
implementation 'androidx.swiperefreshlayout:swiperefreshlayout:1.1.0-alpha02'

Na secção dependências

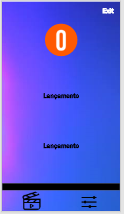
Protótipos

Web



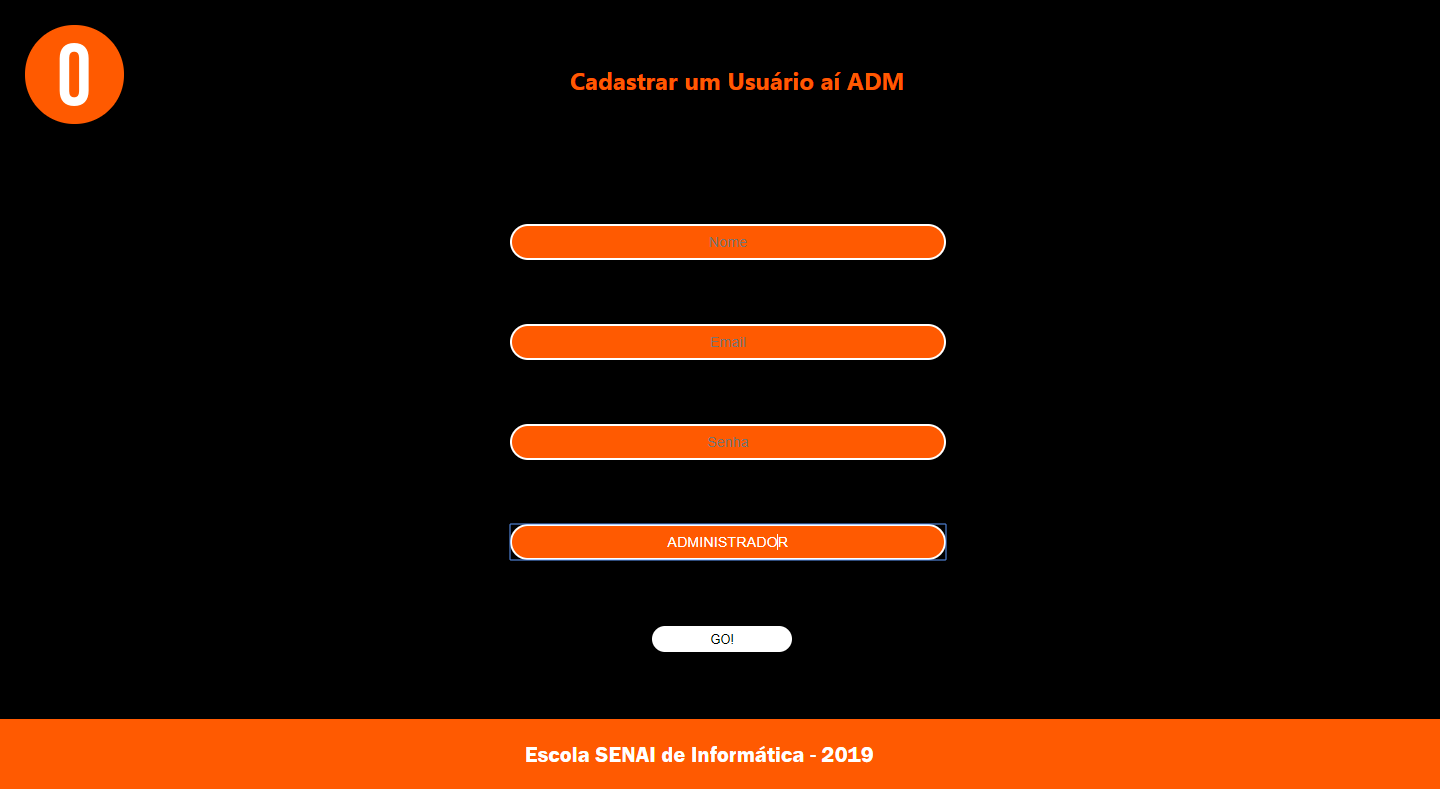


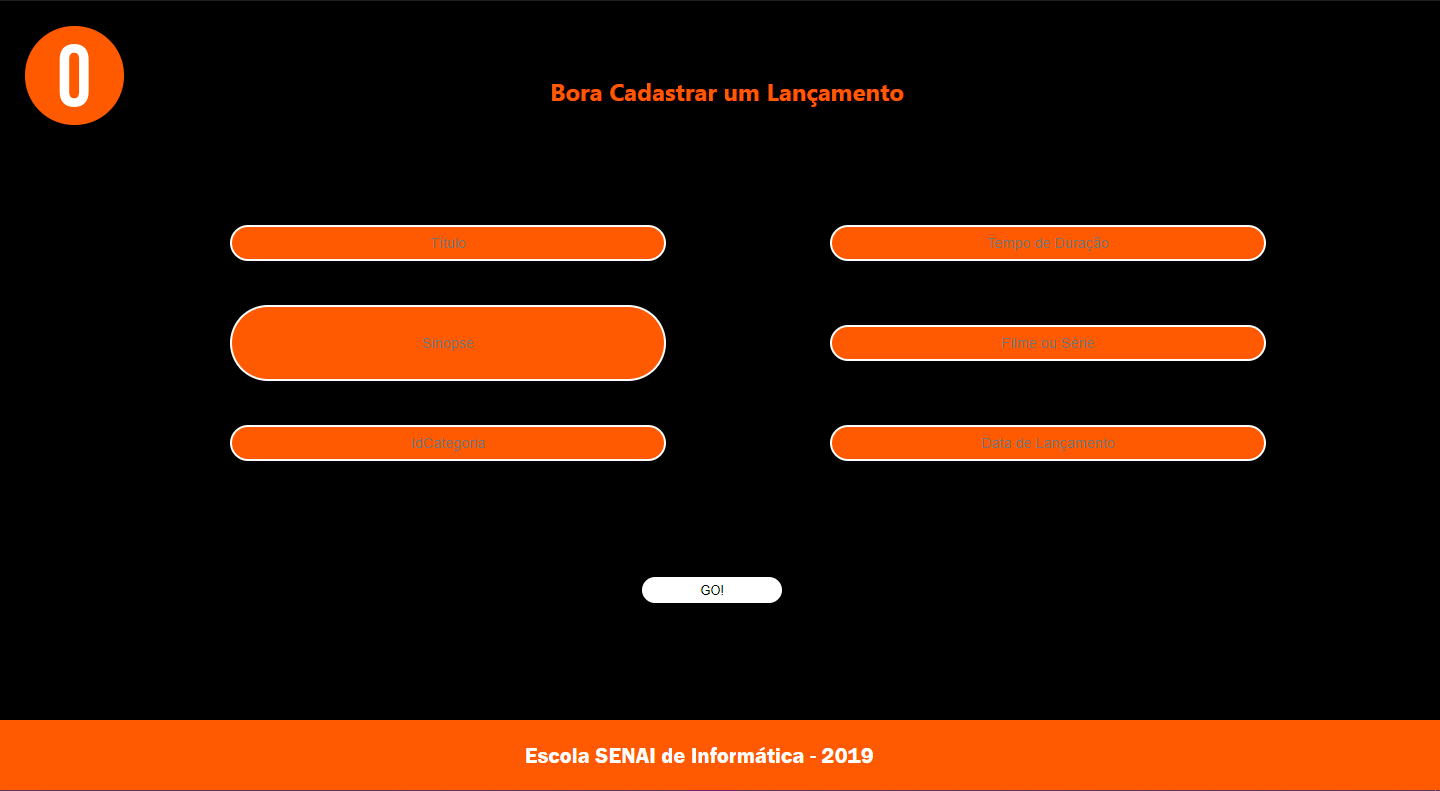
Mobile



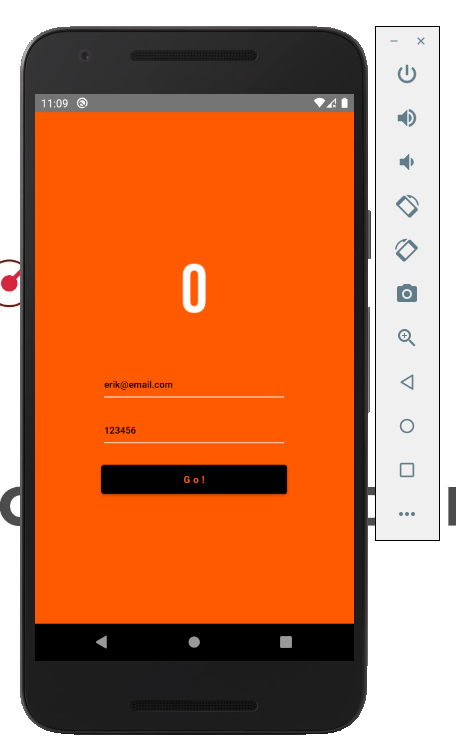
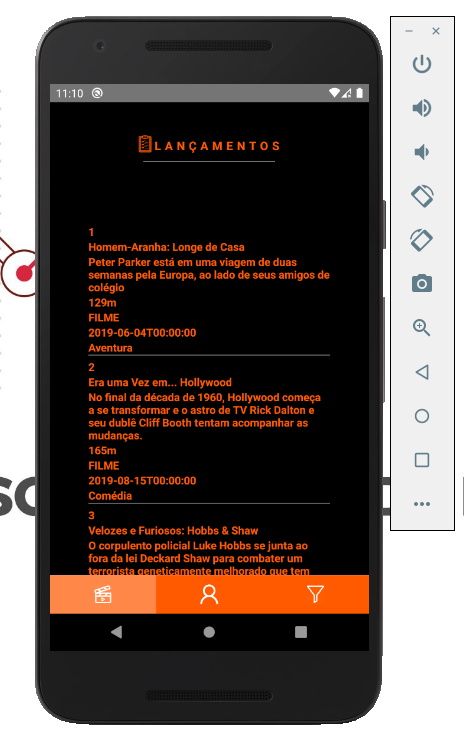
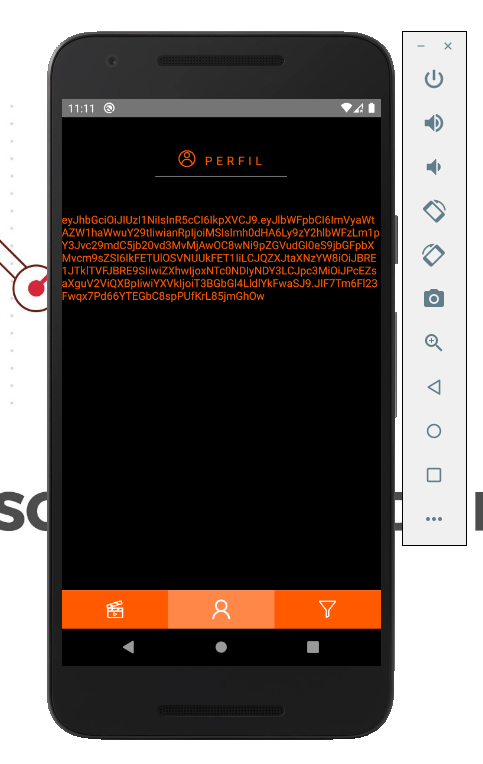
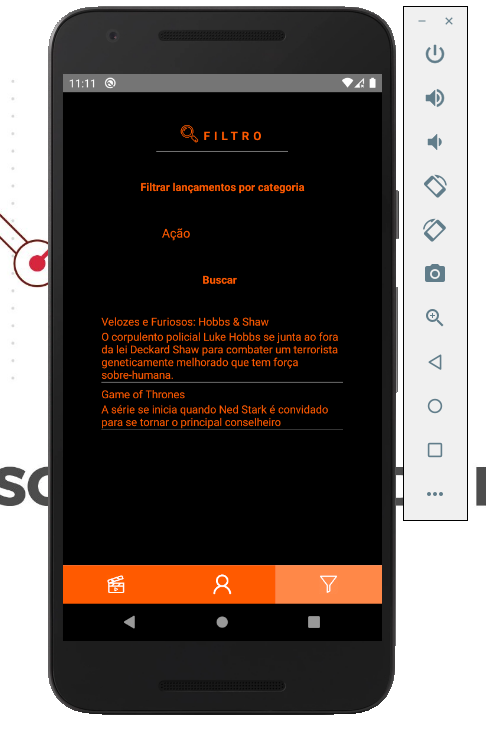


Front-End





Mobile



Arquitetura do Projeto

Referências

Links

Livros