**[DevSuperior logo](https://raw.githubusercontent.com/devsuperior/bds-assets/main/ds/devsuperior-logo-small.png) Semana Spring React - Episódio 1**

*Crie um app inédito para seu portfólio com as tecnologias mais demandadas do mercado*

**Realização**

[DevSuperior - Escola de programação](https://devsuperior.com.br/)

[DevSuperior no Instagram](https://instagram.com/devsuperior.ig) [DevSuperior no Youtube](https://youtube.com/devsuperior)

**Objetivos do projeto para esta aula**

* Criar projetos backend e frontend
* Salvar os projeto no Github em monorepo
* Montar o visual estático do front end

**AVISO: as aulas ficarão disponíveis somente até domingo às 23h59**

**AVISO: Instruções sobre certificado no Github do**[**aqui**](https://github.com/devsuperior/sds-dsmovie/tree/main/_certificado)

**Checklist**

**Passo: preparação**

**Dica: extensões do VS Code**

**Conferir Node (16 LTS) e Yarn**

[Guia de instalação das ferramentas](https://github.com/devsuperior/sds-dsmovie/tree/main/_instalacao)

node -v

yarn -v

Caso precise instalar o Yarn, faça o comando:

npm install --global yarn

**Design Figma**

<https://www.figma.com/file/hpQuzpGHq2MmrI87xnfMoT/DSMovie1>

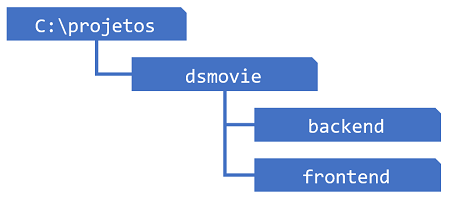
<https://www.figma.com/file/svCMhNqgpAZuN86w9IHJ4v/DSMovie2>

<https://www.figma.com/file/gEZYKqJJs2gEhIB6k9ksGB/DSMovie3>

<https://www.figma.com/file/hyovBMIxwrn2Bb5MZLrxHL/DSMovie4>

**Passo: criar projeto ReactJS**

* ATENÇÃO: esta será a estrutura de pastas que vamos criar:

[](https://raw.githubusercontent.com/devsuperior/bds-assets/main/sds/pastas-dsmovie.png)

yarn create react-app frontend --template typescript

OU:

npx create-react-app frontend --template typescript

IMPORTANTE: deletar subpasta .git

* *Lembrete: ver extensões e arquivos ocultos*

**Passo: criar projeto Spring Boot**

* Criar projeto Spring Boot no Spring Initializr com as seguintes dependências:
  + Web
  + JPA
  + H2
  + Postgres
  + Security
* Ajuste no arquivo pom.xml:

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-resources-plugin</artifactId>

<version>3.1.0</version>

</plugin>

* Botão direito no projeto -> Maven -> Update project (force update)

**Passo: salvar primeira versão no Github**

[Playlist de introdução e configuração do Git/Github](https://www.youtube.com/watch?v=KLG-jC1fh28&list=PLNuUvBZGBA8nDTr8QRMgoT_l3XNt_BbWj)

git config --global user.name <seu nome>

git config --global user.email <seu email>

git init

git add .

git commit -m "Project created"

git log --oneline

git remote “novo repositório do git”

git status

git remote -v

git branch -M main # trocar a branch

git push -u origin main # enviar para o git

**Passo: "limpar" o projeto ReactJS**

* Deletar arquivos não usados
* **COMMIT: Project clean**

**Passo: adicionar Bootstrap e CSS ao projeto**

* Bootstrap

yarn add bootstrap@5.1.3

* Arquivo index.css

@import url('https://fonts.googleapis.com/css2?family=Open+Sans:wght@400;700&display=swap');

:root {

--color-primary: #4D41C0;

}

\* {

box-sizing: border-box;

font-family: 'Open Sans', sans-serif;

}

html, body {

background-color: #E5E5E5;

}

a, a:hover {

text-decoration: none;

color: unset;

}

* Arquivo index.tsx

import 'bootstrap/dist/css/bootstrap.css';

import './index.css';

* **COMMIT: Bootstrap**

**Passo: Componente Navbar**

ATENÇÃO: no arquivo tsconfig.json: "baseUrl": "./src" (reinicie o app)

<header>

<nav className="container">

<div className="dsmovie-nav-content">

<h1>DSMovie</h1>

<a href="https://github.com/devsuperior" target="\_blank" rel="noreferrer">

<div className="dsmovie-contact-container">

<GithubIcon />

<p className="dsmovie-contact-link">/devsuperior</p>

</div>

</a>

</div>

</nav>

</header>

header {

height: 60px;

background-color: var(--color-primary);

display: flex;

align-items: center;

}

.dsmovie-nav-content {

display: flex;

align-items: center;

justify-content: space-between;

color: #fff;

}

.dsmovie-nav-content h1 {

font-size: 24px;

margin: 0;

font-weight: 700;

}

.dsmovie-contact-container {

display: flex;

align-items: center;

}

.dsmovie-contact-link {

margin: 0 0 0 10px;

font-size: 12px;

}

* **COMMIT: Navbar**

**Passo: Rotas**

* Instalar React Router DOM

yarn add react-router-dom@6.2.1 @types/react-router-dom@5.3.2

import {

BrowserRouter,

Routes,

Route

} from "react-router-dom";

import Listing from 'pages/Listing';

import Form from 'pages/Form';

import Navbar from "components/Navbar";

function App() {

return (

<BrowserRouter>

<Navbar />

<Routes>

<Route path="/" element={<Listing />} />

<Route path="/form">

<Route path=":movieId" element={<Form />} />

</Route>

</Routes>

</BrowserRouter>

);

}

export default App;

* **COMMIT: Routes**

**Passo: Tela de formulário**

const movie = {

id: 1,

image: "https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/jBJWaqoSCiARWtfV0GlqHrcdidd.jpg",

title: "The Witcher",

count: 2,

score: 4.5

};

<div className="dsmovie-form-container">

<img className="dsmovie-movie-card-image" src="url" alt="The Witcher" />

<div className="dsmovie-card-bottom-container">

<h3>"The Witcher"</h3>

<form className="dsmovie-form">

<div className="form-group dsmovie-form-group">

<label htmlFor="email">Informe seu email</label>

<input type="email" className="form-control" id="email" />

</div>

<div className="form-group dsmovie-form-group">

<label htmlFor="score">Informe sua avaliação</label>

<select className="form-control" id="score">

<option>1</option>

<option>2</option>

<option>3</option>

<option>4</option>

<option>5</option>

</select>

</div>

<div className="dsmovie-form-btn-container">

<button type="submit" className="btn btn-primary dsmovie-btn">Salvar</button>

</div>

</form >

<button className="btn btn-primary dsmovie-btn mt-3">Cancelar</button>

</div >

</div >

.dsmovie-form-container {

max-width: 480px;

margin: 40px auto;

padding: 20px;

}

.dsmovie-form {

width: calc(100% - 20px);

}

.dsmovie-form-group {

margin-bottom: 20px;

}

.dsmovie-form-group label {

font-size: 12px;

color: #aaa;

}

.dsmovie-form-btn-container {

display: flex;

justify-content: center;

}

.dsmovie-movie-card-image {

width: 100%;

border-radius: 8px 8px 0 0;

}

.dsmovie-card-bottom-container {

background-color: #fff;

display: flex;

flex-direction: column;

align-items: center;

padding: 10px 10px 20px 10px;

border-radius: 0 0 8px 8px;

}

.dsmovie-card-bottom-container h3 {

color: #4A4A4A;

text-align: center;

font-size: 14px;

font-weight: 700;

margin-bottom: 10px;

min-height: 40px;

}

.dsmovie-btn {

background-color: var(--color-primary);

font-size: 14px;

font-weight: 700;

height: 40px;

width: 180px;

display: flex;

align-items: center;

justify-content: center;

}

* **COMMIT: Form layout**

**Passo: Pagination**

<div className="dsmovie-pagination-container">

<div className="dsmovie-pagination-box">

<button className="dsmovie-pagination-button" disabled={true} >

<Arrow />

</button>

<p>{`${1} de ${3}`}</p>

<button className="dsmovie-pagination-button" disabled={false} >

<Arrow className="dsmovie-flip-horizontal" />

</button>

</div>

</div>

.dsmovie-pagination-container {

padding: 15px 0;

display: flex;

justify-content: center;

align-items: center;

}

.dsmovie-pagination-box {

width: 180px;

display: flex;

justify-content: space-between;

align-items: center;

}

.dsmovie-pagination-box form {

width: 100%;

display: flex;

}

.dsmovie-pagination-button {

width: 40px;

height: 40px;

border-radius: 4px;

background-color: #fff;

display: flex;

justify-content: center;

align-items: center;

border: 1px solid var(--color-primary);

cursor: pointer;

}

.dsmovie-pagination-button svg {

filter: brightness(0) saturate(100%) invert(26%) sepia(19%) saturate(7395%) hue-rotate(234deg) brightness(89%) contrast(92%);

}

.dsmovie-pagination-button:disabled {

border: 1px solid #c2c2c2;

cursor: unset;

}

.dsmovie-pagination-button:disabled svg {

filter: none;

}

.dsmovie-pagination-container p {

margin: 0;

font-size: 12px;

color: var(--color-primary);

}

.dsmovie-flip-horizontal {

transform: rotate(180deg);

}

* **COMMIT: Pagination layout**

**Passo: MovieCard**

**MovieStars**

<div className="dsmovie-stars-container">

<StarFull />

<StarFull />

<StarFull />

<StarHalf />

<StarEmpty />

</div>

.dsmovie-stars-container {

width: 130px;

display: flex;

justify-content: space-between;

}

.dsmovie-stars-container svg {

width: 22px;

height: auto;

}

**MovieScore**

<div className="dsmovie-score-container">

<p className="dsmovie-score-value">{score > 0 ? score.toFixed(1) : '-'}</p>

<MovieStars />

<p className="dsmovie-score-count">{count} avaliações</p>

</div>

.dsmovie-score-container {

display: flex;

flex-direction: column;

align-items: center;

}

.dsmovie-score-value {

margin: 0;

color: #FFBB3A;

font-size: 16px;

font-weight: 700;

}

.dsmovie-score-count {

font-size: 12px;

color: #989898;

margin: 4px 0 10px 0;

}

**MovieCard**

<div>

<img className="dsmovie-movie-card-image" src={movie.image} alt={movie.title} />

<div className="dsmovie-card-bottom-container">

<h3>{movie.title}</h3>

<MovieScore />

<div className="btn btn-primary dsmovie-btn">Avaliar</div>

</div>

</div>

* **COMMIT:**

**Passo: Navegação dos botões**

* **COMMIT: Navigation buttons**

**Filtrar por estrela**

mport { useEffect, useState } from 'react';

export default () => {

const [movie, setCars] = useMovie([]);

const [loading, setLoading] = useState(false);

const [star, setStar] = useState('');

  const getCars = async () => {

    setMovie([]);

    setLoading(true);

    // requisição

    let result = await fetch(`https://api.b7web.com.br/carros/api/carros?ano=${star}`)

    let json = await result.json();

    setLoading(false);

    if (json.error === '') {

      setCars(json.cars);

    } else {

      alert(json.error);

    }

  };

const handleYearChange = (e) => {

 // console.log(e.target.value);

  setYear(e.target.value);

}

useEffect(() => {

  getCars();

}, [year]);

  return(

    <div>

        <span>Selecione o ano do carro:</span>

        <select onChange={handleYearChange}>

          <option>2010</option>

          <option>2011</option>

          <option>2012</option>

          <option>2013</option>

          <option>2014</option>

        </select>

        <br />

        <button onClick={getCars}>Atualizar</button>

        <hr/>

        {loading === true &&

          <h2>Carregando as informações...</h2>

        }

      {cars.length === 0 && loading === false &&

        <h2>Nenhum carro encontrado!</h2>

      }

      <div className="container">

        {cars.map((item, index)=>(

          <div  key={index}>

            <h3>{item.brand} - {item.name}</h3>

            <img className="imagem" src={item.photo} width="200"/>

            <p>{item.year} - R$ {item.price}</p>

          </div>

        ))}

        </div>

    </div>

  );

}

# [DevSuperior logo](https://raw.githubusercontent.com/devsuperior/bds-assets/main/ds/devsuperior-logo-small.png) Semana Spring React - Episódio 2

Crie um app inédito para seu portfólio com as tecnologias mais demandadas do mercado

## Realização

[DevSuperior - Escola de programação](https://devsuperior.com.br/)

[DevSuperior no Instagram](https://instagram.com/devsuperior.ig) [DevSuperior no Youtube](https://youtube.com/devsuperior)

## Objetivos do projeto para esta aula

* Implementar o back end
* Modelo de domínio
* Acesso a banco de dados
* Estruturar o back end no padrão camadas
* Criar endpoints da API REST
* Implantação na nuvem

## AVISO: as aulas ficarão disponíveis somente até domingo às 23h59

## AVISO: Instruções sobre certificado no Github do [aqui](https://github.com/devsuperior/sds-dsmovie/tree/main/_certificado)

## Checklist

### Passo: configuração de segurança

@Configuration

@EnableWebSecurity

public class SecurityConfig extends WebSecurityConfigurerAdapter {

@Autowired

private Environment env;

@Override

protected void configure(HttpSecurity http) throws Exception {

if (Arrays.asList(env.getActiveProfiles()).contains("test")) {

http.headers().frameOptions().disable();

}

http.cors().and().csrf().disable();

http.sessionManagement().sessionCreationPolicy(SessionCreationPolicy.STATELESS);

http.authorizeRequests().anyRequest().permitAll();

}

@Bean

CorsConfigurationSource corsConfigurationSource() {

CorsConfiguration configuration = new CorsConfiguration().applyPermitDefaultValues();

configuration.setAllowedMethods(Arrays.asList("POST", "GET", "PUT", "DELETE", "OPTIONS"));

final UrlBasedCorsConfigurationSource source = new UrlBasedCorsConfigurationSource();

source.registerCorsConfiguration("/\*\*", configuration);

return source;

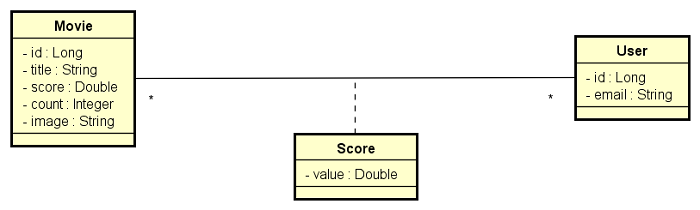
}

}

* **COMMIT: Security config**

### Passo: criar as entidades e o seed do banco

#### Modelo conceitual

[](https://raw.githubusercontent.com/devsuperior/bds-assets/main/sds/dsmovie-dominio.png)

#### application.properties

spring.profiles.active=test

spring.jpa.open-in-view=false

#### application-test.properties

# Dados de conexão com o banco H2

spring.datasource.url=jdbc:h2:mem:testdb

spring.datasource.username=sa

spring.datasource.password=

# Configuração do cliente web do banco H2

spring.h2.console.enabled=true

spring.h2.console.path=/h2-console

# Configuração para mostrar o SQL no console

spring.jpa.show-sql=true

spring.jpa.properties.hibernate.format\_sql=true

#### import.sql

INSERT INTO tb\_user(email) VALUES ('maria@gmail.com');

INSERT INTO tb\_user(email) VALUES ('joao@gmail.com');

INSERT INTO tb\_user(email) VALUES ('ana@gmail.com');

INSERT INTO tb\_user(email) VALUES ('lucia@gmail.com');

INSERT INTO tb\_user(email) VALUES ('joaquim@gmail.com');

INSERT INTO tb\_movie(score, count, title, image) VALUES (4.5, 2, 'The Witcher', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/jBJWaqoSCiARWtfV0GlqHrcdidd.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (3.3, 3, 'Venom: Tempo de Carnificina', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/vIgyYkXkg6NC2whRbYjBD7eb3Er.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (0, 0, 'O Espetacular Homem-Aranha 2: A Ameaça de Electro', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/u7SeO6Y42P7VCTWLhpnL96cyOqd.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (0, 0, 'Matrix Resurrections', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/hv7o3VgfsairBoQFAawgaQ4cR1m.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (0, 0, 'Shang-Chi e a Lenda dos Dez Anéis', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/cinER0ESG0eJ49kXlExM0MEWGxW.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (0, 0, 'Django Livre', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/2oZklIzUbvZXXzIFzv7Hi68d6xf.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (0, 0, 'Titanic', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/yDI6D5ZQh67YU4r2ms8qcSbAviZ.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (0, 0, 'O Lobo de Wall Street', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/cWUOv3H7YFwvKeaQhoAQTLLpo9Z.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (0, 0, 'Aves de Rapina: Arlequina e sua Emancipação Fantabulosa', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/jiqD14fg7UTZOT6qgvzTmfRYpWI.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (0, 0, 'Rogue One: Uma História Star Wars', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/6t8ES1d12OzWyCGxBeDYLHoaDrT.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (0, 0, 'Star Wars: A Guerra dos Clones', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/uK15I3sGd8AudO9z6J6vi0HH1UU.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (0, 0, 'Star Wars: Episódio I - A Ameaça Fantasma', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/36LnijfQCOC89rCMOhn2OINXROI.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (0, 0, 'Vingadores: Ultimato', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/7RyHsO4yDXtBv1zUU3mTpHeQ0d5.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (0, 0, 'Thor', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/cDJ61O1STtbWNBwefuqVrRe3d7l.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (0, 0, 'Cisne Negro', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/hqh5O4KssfJWI62HGAgrjHXbxhD.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (0, 0, 'O Silêncio dos Inocentes', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/mfwq2nMBzArzQ7Y9RKE8SKeeTkg.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (0, 0, 'Clube da Luta', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/hZkgoQYus5vegHoetLkCJzb17zJ.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (0, 0, 'Guerra Mundial Z', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/31VpBgUX5O4Z3dn5ZbX8HLqoXH3.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (0, 0, 'Harry Potter e as Relíquias da Morte - Parte 1', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/vcrgU0KaNj5mKUCIQSUdiQwTE9y.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (0, 0, 'Harry Potter e a Pedra Filosofal', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/lvOLivVeX3DVVcwfVkxKf0R22D8.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (0, 0, 'Alice no País das Maravilhas', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/qNdlZgz9yoSJ8f0YxQWfKGPoVV.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (0, 0, 'Animais Fantásticos e Onde Habitam', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/8Qsr8pvDL3s1jNZQ4HK1d1Xlvnh.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (0, 0, 'A Teoria de Tudo', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/kq2MHrRfH6RTfkvyDEmYLmGHE6U.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (0, 0, 'O Livro de Boba Fett', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/sjx6zjQI2dLGtEL0HGWsnq6UyLU.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (0, 0, 'O Último Duelo', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/4LrL40XecjGLRpX5I2gzMTUt04l.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (0, 0, 'Interestelar', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/rAiYTfKGqDCRIIqo664sY9XZIvQ.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (0, 0, 'Contato', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/yFkUPqBuUnbhYbQL8VFpTrAT9za.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (0, 0, 'Duna', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/jYEW5xZkZk2WTrdbMGAPFuBqbDc.jpg');

INSERT INTO tb\_movie(score, count, title, image) VALUES (0, 0, 'Aquaman', 'https://www.themoviedb.org/t/p/w533\_and\_h300\_bestv2/2cUsDz4TzFYHrKktT1bKHHQ7Cgm.jpg');

INSERT INTO tb\_score(movie\_id, user\_id, value) VALUES (1, 1, 5.0);

INSERT INTO tb\_score(movie\_id, user\_id, value) VALUES (1, 2, 4.0);

INSERT INTO tb\_score(movie\_id, user\_id, value) VALUES (2, 1, 3.0);

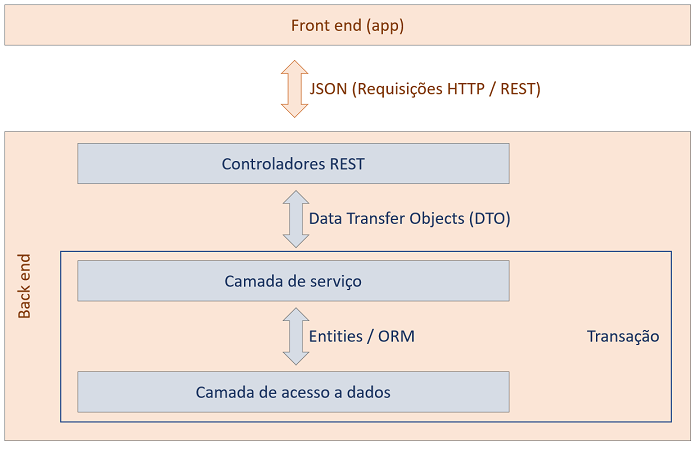
INSERT INTO tb\_score(movie\_id, user\_id, value) VALUES (2, 2, 3.0);

INSERT INTO tb\_score(movie\_id, user\_id, value) VALUES (2, 3, 4.0);

* **COMMIT: Domain model, database seed**

### Passo: Busca de filmes

#### Padrão camadas adotado

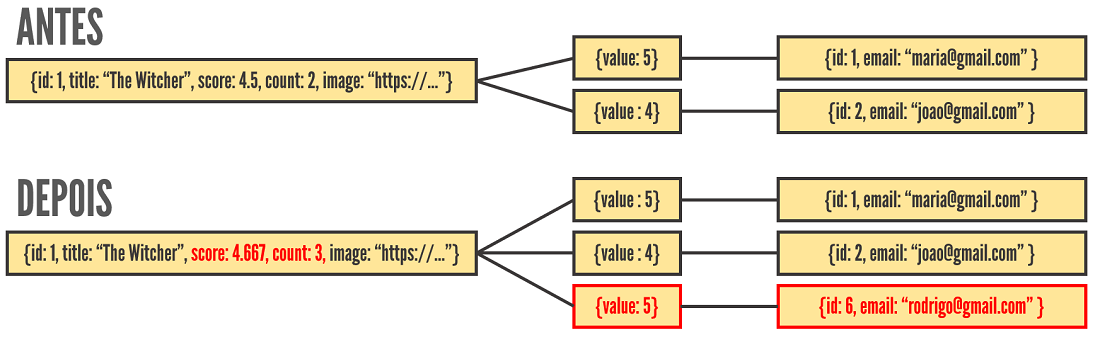
[](https://github.com/devsuperior/bds-assets/raw/main/sds/padrao-camadas.png)

* Criar repository
* Criar DTO
* Criar service
* Criar controller
* **COMMIT: Find movies**

### Passo: Salvar avaliação

#### Lógica:

1. Informar email, id do filme e valor da avaliação (1 a 5).
2. Recuperar usuário do banco de dados pelo email. Se o usuário não existir, insira no banco.
3. Salvar a avaliação do usuário para o dado filme.
4. Recalcular a avaliação média do filme e salvar no banco de dados.

[](https://raw.githubusercontent.com/devsuperior/bds-assets/main/sds/dsmovie-objs.png)

* **COMMIT: Save score**

### Passo: Validação no Postgres local

* Criar três perfis de projeto: test, dev, prod
* Gerar script SQL no perfil dev
* Testar projeto no banco Postgres local

#### application-dev.properties

#spring.jpa.properties.javax.persistence.schema-generation.create-source=metadata

#spring.jpa.properties.javax.persistence.schema-generation.scripts.action=create

#spring.jpa.properties.javax.persistence.schema-generation.scripts.create-target=create.sql

#spring.jpa.properties.hibernate.hbm2ddl.delimiter=;

spring.datasource.url=jdbc:postgresql://localhost:5432/dsmovie

spring.datasource.username=postgres

spring.datasource.password=1234567

spring.jpa.properties.hibernate.jdbc.lob.non\_contextual\_creation=true

spring.jpa.hibernate.ddl-auto=none

#### application-prod.properties

spring.datasource.url=${DATABASE\_URL}

#### system.properties

java.runtime.version=17

* **COMMIT: First homolog**

### Passo: Implantação no Heroku

* Criar app no Heroku
* Provisionar banco Postgres
* Definir variável APP\_PROFILE=prod
* Conectar ao banco via pgAdmin
* Criar seed do banco

heroku -v

heroku login

heroku git:remote –a lamberti-dsmovie

git remote -v

git subtree push --prefix backend heroku main

### Passo: implantação no Netlify

* Deploy básico
  + Base directory: frontend
  + Build command: yarn build
  + Publish directory: frontend/build
* Arquivo \_redirects

/\* /index.html 200

* Configurações adicionais
  + Site settings -> Domain Management: (colocar o nome que você quiser)
  + Deploys -> Trigger deploy

### Passo: Primeira requisição

* Instalar Axios

yarn add [axios@0.24.0](mailto:axios@0.24.0)

* Definir a BASE\_URL

export const BASE\_URL = process.env.REACT\_APP\_BACKEND\_URL ?? "http://localhost:8080";

* Definir os tipos Movie e MoviePage
* Fazer a requisição

export type Movie = {

id: number;

title: string;

score: number;

count: number;

image: string;

}

export type MoviePage = {

content: Movie[];

last: boolean;

totalPages: number;

totalElements: number;

size: number;

number: number;

first: boolean;

numberOfElements: number;

empty: boolean;

}

* **COMMIT: First request**

### Passo: React hooks: useState e useEffect

Hooks são funções cujo comportamento está vinculado ao estado e ao ciclo de vida do React a partir de componentes funcionais.

<https://pt-br.reactjs.org/docs/hooks-overview.html>

Hook: useState

Manter estado no componente

Hook: useEffect

Executar algo na instanciação ou destruição do componente, observar estado

* **COMMIT: useState, useEffect**

### Passo: Props

Props podem ser entendidas como argumentos do componente React.

<https://pt-br.reactjs.org/docs/components-and-props.html>

NOTA: em uma renderização dinâmica de coleção, cada elemento renderizado DEVE possuir um atributo key.

* **COMMIT: Props**

### Passo: useParams

* **COMMIT: useParams**

### Passo: Mostrar estrelinhas

// EX:

// getFills(3.5) => [1, 1, 1, 0.5, 0]

// getFills(4.1) => [1, 1, 1, 1, 0.5]

function getFills(score: number) {

const fills = [0, 0, 0, 0, 0];

const integerPart = Math.floor(score);

for (let i = 0; i < integerPart; i++) {

fills[i] = 1;

}

const diff = score - integerPart;

if (diff > 0) {

fills[integerPart] = 0.5;

}

return fills;

}

* **COMMIT: Show score**

### Passo: Pagination

* Controlar botão habilitado/desabilitado
* Trocar página ao clique do botão

const handlePageChange = (newNumber: number) => {

setPageNumber(newNumber);

}

* **COMMIT: Pagination**

### Passo: Salvando score, useNavigate

# ATENÇÃO: Coloque o [movieId] nas dependências o useEffect do FormCard:

useEffect(() => {

axios.get(`${BASE\_URL}/movies/${movieId}`)

.then(response => {

setMovie(response.data);

});

}, [movieId]);

Função para validar email

// https://stackoverflow.com/questions/46155/whats-the-best-way-to-validate-an-email-address-in-javascript

export function validateEmail(email: any) {

return String(email)

.toLowerCase()

.match(

/^(([^<>()[\]\\.,;:\s@"]+(\.[^<>()[\]\\.,;:\s@"]+)\*)|(".+"))@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\])|(([a-zA-Z\-0-9]+\.)+[a-zA-Z]{2,}))$/

);

}

Objeto de configuração da requisição Axios

const config: AxiosRequestConfig = {

baseURL: BASE\_URL,

method: 'PUT',

url: '/scores',

data: {

email: email,

movieId: movieId,

score: score

}

}

* **COMMIT: Save, useNavigate**

## PARABÉNS!

## [https://lambertidsmovie.netlify.app](https://lambertidsmovie.netlify.app/)

**Links complementares:**

[*https://dashboard.heroku.com/apps/lamberti-dsmovie/deploy/heroku-git*](https://dashboard.heroku.com/apps/lamberti-dsmovie/deploy/heroku-git)

*<https://github.com/LambertiMarcos/dsmovie>*

*<https://app.netlify.com/sites/lambertidsmovie/overview>*