

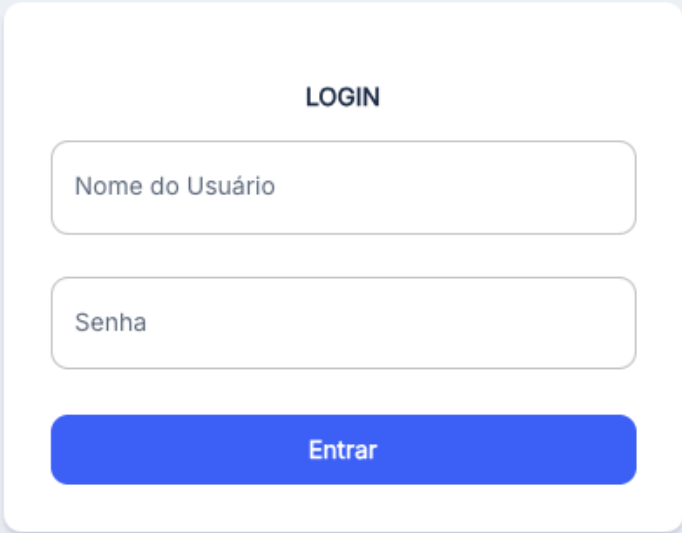
Link do repositório git: <https://github.com/iahgo/projeto-react>

Link da aplicação no Vercel: <https://projeto-react-six-alpha.vercel.app/>

Para rodar Local:

- 1- fazer o download do projeto no git
- 2- acessar o diretório frontend
- 3- yarn install e npm start
- 4- iniciar o backend

Foi desenvolvido um projeto em react que faz consultas a API-CARROS. O projeto se trata de um CRUD, onde o usuário faz login através de email e senha cadastrados na base de dados e é feita a requisição ao backend para validação da autenticação do usuário. Os dados são salvos em um token no localStorage e após fazer o logout os dados são destruídos por questões de segurança.

A screenshot of a login form titled "LOGIN" centered on a light purple background. The form is a white rounded rectangle containing two input fields: "Nome do Usuário" and "Senha". Below these fields is a blue button with the text "Entrar".

LOGIN

Nome do Usuário

Senha

Entrar

O login é feito através de requisição post com autenticação JWT, e o token fica salvo no localStorage enquanto o usuário estiver logado na plataforma.

Após realizar o login, é direcionado para a rota /cars, onde é possível cadastrar um novo carro, editar carros, deletar carros, ou detalhar e obter mais informação sobre os carros:

Gerenciamento de Carros

Cars

Adicionar Novo Carro

Modelo
Fabricante
Ano
Cor
Cavalos de Potência
País
Adicionar

- Golf - Volkswagen - 2020

Editar Deletar Detalhar

- Polo - Volkswagen - 2019

Editar Deletar Detalhar

- Passat - Volkswagen - 2021

Editar Deletar Detalhar

- Tiguan - Volkswagen - 2022

Editar Deletar Detalhar

- Fusion - Ford - 2021

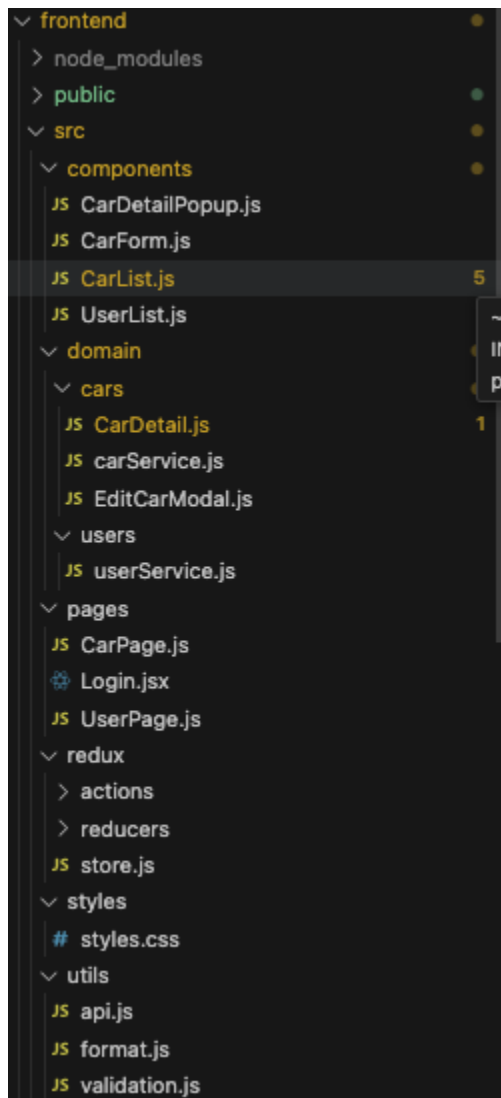
Editar Deletar Detalhar

- Focus - Ford - 2020

Editar Deletar Detalhar

PRINT DOS CÓDIGOS

ORGANIZAÇÃO DAS PASTAS E ARQUIVOS



COMPONENTS

CarForm - um componente que faz um formulário onde estão os inputs para criar um novo carro

```

const CarForm = ({ newCar, setNewCar, handleCreateCar }) => {
  >
  <input
    type="text"
    placeholder="Modelo"      You, ontem + commit inicial
    value={newCar.modelo}
    onChange={(e) => setNewCar({ ...newCar, modelo: e.target.value })}
  />
  <input
    type="text"
    placeholder="Fabricante"
    value={newCar.fabricante}
    onChange={(e) => setNewCar({ ...newCar, fabricante: e.target.value })}
  />
  <input
    type="number"
    placeholder="Ano"
    value={newCar.ano}
    onChange={(e) => setNewCar({ ...newCar, ano: e.target.value })}
  />
  <input
    type="text"
    placeholder="Cor"
    value={newCar.cor}
    onChange={(e) => setNewCar({ ...newCar, cor: e.target.value })}
  />
  <input
    type="number"
    placeholder="Cavalos de Potência"
    value={newCar.cavalosDePotencia}
    onChange={(e) => setNewCar({ ...newCar, cavalosDePotencia: e.target.value })}
  />
  <input
    type="text"
    placeholder="País"
    value={newCar.pais}
    onChange={(e) => setNewCar({ ...newCar, pais: e.target.value })}
  />
  <button type="submit">Adicionar</button>
</form>

```

CarDetailPopup - é uma janela dos detalhes do carro, a tela que aparece ao clicar em detalhar:

```

import React from 'react';

const CarDetailPopup = ({ carDetails, onClose }) => {
  if (!carDetails) return null;

  return (
    <div className="popup">
      <div className="popup-inner">
        <h2>Detalhes do Carro</h2>
        <p>Modelo: {carDetails.modelo}</p>
        <p>Fabricante: {carDetails.fabricante}</p>
        <p>Ano: {carDetails.ano}</p>
        <button onClick={onClose}>Fechar</button>
      </div>
    </div>
  );
};

export default CarDetailPopup;

```

DOMAIN - Aqui estão os domínios comerciais, carros e usuários. toda a lógica de chamadas a api começa por aqui:

CarService: onde estão os métodos que fazem a chamada ao backend através da rota api/carros

```

const BASE_URL = '/api/carros';

const handleResponse = async (response) => {
  if (!response.ok) {
    const error = await response.text();
    throw new Error(error || response.statusText);
  }
  const contentType = response.headers.get('content-type');
  if (contentType && contentType.includes('application/json')) {
    return await response.json();
  }
  return null;
};

const getAllCars = async () => {
  try {
    const response = await fetch(BASE_URL);
    return await handleResponse(response);
  } catch (error) {
    console.error('Erro ao buscar todos os carros:', error);
    throw error;
  }
};

const getCarById = async (id) => {
  try {
    const response = await fetch(`${BASE_URL}/${id}`);
    return await handleResponse(response);
  } catch (error) {
    console.error(`Erro ao buscar o carro com ID ${id}:`, error);
    throw error;
  }
};

const createCar = async (car) => {
  try {
    const response = await fetch(BASE_URL, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
      },
      body: JSON.stringify(car),
    });
  }
};

```

CarEdit: Formulário onde é alterados os dados antigos do carro e será feita a troca para os novos dados.

```

const EditCarModal = ({ car, onSave, onClose }) => {
  <input
    type="text"
    name="modelo"
    placeholder="Modelo"
    value={editedCar.modelo}
    onChange={handleChange}
  />
  <input
    type="text"
    name="fabricante"
    placeholder="Fabricante"
    value={editedCar.fabricante}
    onChange={handleChange}
  />
  <input
    type="number"
    name="ano"
    placeholder="Ano"
    value={editedCar.ano}
    onChange={handleChange}
  />
  <input
    type="text"
    name="cor"
    placeholder="Cor"
    value={editedCar.cor}
    onChange={handleChange}
  />
  <input
    type="number"
    name="cavalosDePotencia"
    placeholder="Cavalos de Potência"
    value={editedCar.cavalosDePotencia}
    onChange={handleChange}
  />
  <input
    type="text"
    name="pais"
    placeholder="País"
    value={editedCar.pais}
    onChange={handleChange}
  />

```

PAGES

Páginas: Carros, Login, Usuarios

Cada página contém os componentes relacionados a cada área de negócio. Por exemplo, dentro da page Car estão o formulário de carro, lista de carros, e o cabeçalho.

CarPage

```

You, há 4 minutos | 1 author (You)
import React from 'react';
import { useHistory } from 'react-router-dom';
import CarList from '../components/CarList';

const CarPage = () => {
  const history = useHistory();

  const handleNavigateToUsers = () => {
    history.push('/users');
  };

  const handleLogout = () => {
    localStorage.removeItem('token');
    history.push('/login');
  };

  const classButton = 'p-2 bg-blue-500 text-white';

  return (
    <div>
      <button onClick={
        Ver Usuários
      }>
      </button>
      <button onClick={handleLogout} className={classButton}>
        Logout
      </button>
      <h1>Gerenciamento de Carros</h1>
      <CarList />
    </div>
  );
};

export default CarPage;

```

```

const handleLogout: () => void
const handleLogout = () => {
  localStorage.removeItem('token');
  history.push('/login');
};

```

```
on: 'absolute', top: 10, right: 10 }>
```

You, ontem • commit inicial

Tela de Login onde é feita o formulário de login, autenticação e salvar o token no localStorage:


```
handleSubmit = async (e) => {
  e.preventDefault();
  const { email, password } = this.state;
  const { history, dispatchSaveEmail } = this.props;

  try {
    const BASE_URL = 'api/usuarios';
    const response = await fetch(`${BASE_URL}/login`, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
      },
      body: JSON.stringify({ email, password }),
    });

    if (!response.ok) {
      throw new Error('Email ou senha inválidos');
    }

    const data = await response.json();
    const { token } = data;
    localStorage.setItem('token', token);
    dispatchSaveEmail(email);
    history.push('/cars');
  } catch (error) {
    this.setState({ error: error.message });
  }
};
```

Formulário de login

```

render() {
  const { email, password, isDisabled, error } = this.state;
  const classButton = 'p-2 bg-blue-500 text-white';

  return (
    <div>
      <form onSubmit={ this.handleSubmit }>
        <input
          data-testid="email-input"
          placeholder="Email"
          type="text"
          name="email"
          value={ email }
          onChange={ this.handleChange }
          className="p-1 border"
        />
        <input
          data-testid="password-input"
          placeholder="Senha"
          type="password"
          name="password"
          value={ password }
          onChange={ this.handleChange }
          className="p-1 border"
        />
        <button
          type="submit"
          disabled={ isDisabled }
          className={ classButton }
        >
          Entrar
        </button>
      </form>
      { error && <p>{ error }</p> }
    </div>
  );
}

```

UTILS

No arquivo validation.js é feita validação de formulário, para preencher campos obrigatórios por exemplo:

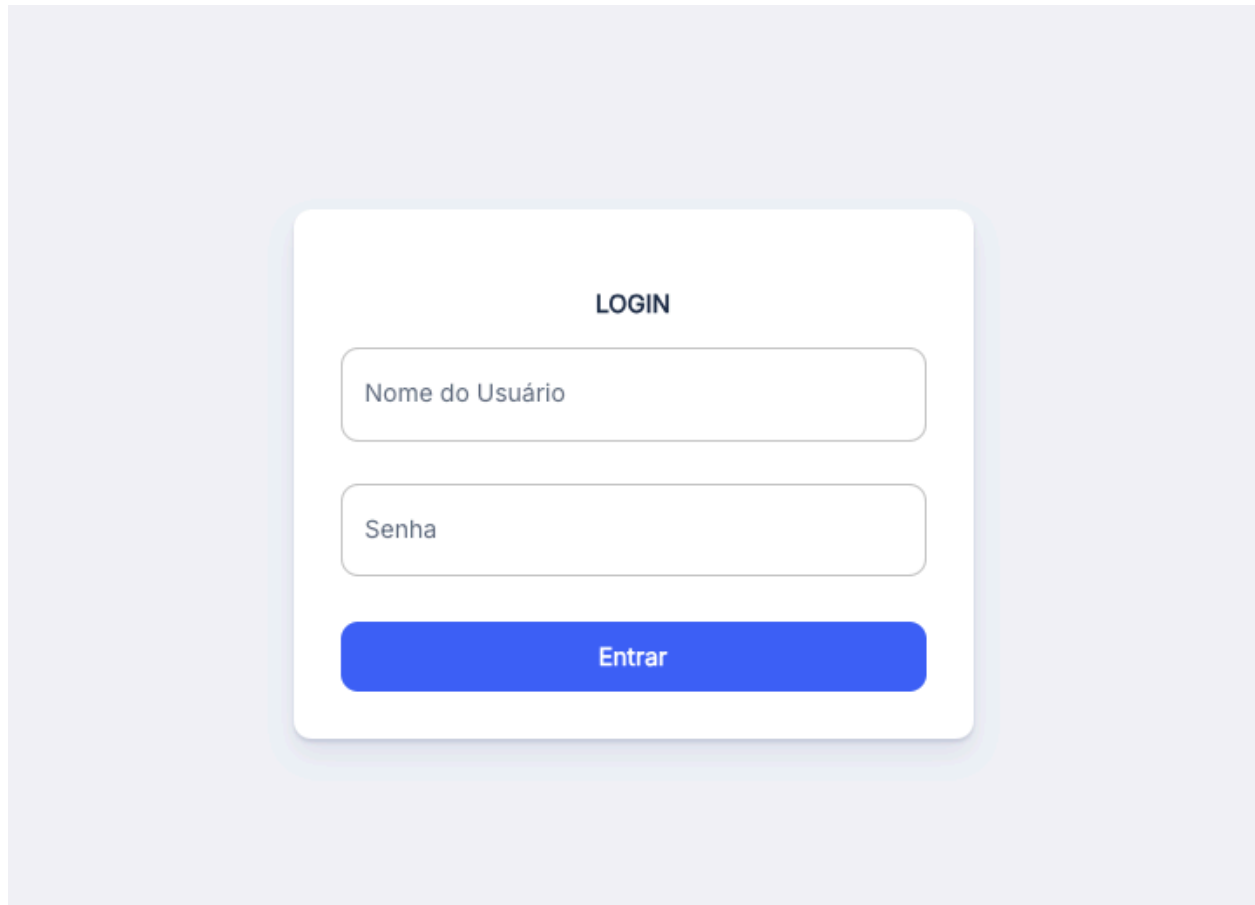
```

export const validateCar = (car) => {
  const errors = {};
  if (!car.modelo) errors.modelo = 'Modelo é obrigatório';
  if (!car.fabricante) errors.fabricante = 'Fabricante é obrigatório';
  if (!car.ano) errors.ano = 'Ano é obrigatório';
  return errors;
};

```

Imagens do projeto

Tela de Login



A login screen design featuring a light gray background. In the center is a white rounded rectangle with a subtle drop shadow. At the top of this rectangle is the word "LOGIN" in bold black text. Below it are two white input fields with rounded corners and thin gray borders. The first field is labeled "Nome do Usuário" and the second is labeled "Senha". At the bottom of the white rectangle is a solid blue button with rounded corners and the text "Entrar" in white.

LOGIN

Nome do Usuário

Senha

Entrar

LOGIN

Nome do usuário ou senha inválido!

Nome do Usuário

emailerrado@email.com

Senha

••••••••••

Entrar

Mensagem de login invalido, usuário ou senha errados.

Após fazer o login com sucesso, abrirá a tela carros:

Componente formulário criar novo carro

Gerenciamento de Carros

Cars

Adicionar Novo Carro

Lista todos os carros

- Golf - Volkswagen - 2020

[Editar](#)[Deletar](#)[Detalhar](#)

- Polo - Volkswagen - 2019

[Editar](#)[Deletar](#)[Detalhar](#)

- Passat - Volkswagen - 2021

[Editar](#)[Deletar](#)[Detalhar](#)

- Tiguan - Volkswagen - 2022

[Editar](#)[Deletar](#)[Detalhar](#)

- Fusion - Ford - 2021

[Editar](#)[Deletar](#)[Detalhar](#)

- Focus - Ford - 2020

[Editar](#)[Deletar](#)[Detalhar](#)

- Mustang - Ford - 2022

[Editar](#)[Deletar](#)[Detalhar](#)

- Corolla - Toyota - 2022

[Editar](#)[Deletar](#)[Detalhar](#)

- Camry - Toyota - 2021

[Editar](#)[Deletar](#)[Detalhar](#)

- Hilux - Toyota - 2020

[Editar](#)[Deletar](#)[Detalhar](#)

- Yaris - Toyota - 2022

[Editar](#)[Deletar](#)[Detalhar](#)

- RAV4 - Toyota - 2021

[Editar](#)[Deletar](#)[Detalhar](#)

- Uno - Fiat - 2020

[Editar](#)[Deletar](#)[Detalhar](#)

Componente de Editar um carro

Antes

- Golf - Volkswagen - 2020

Editar

Deletar

Detalhar

Ao clicar em editar

Gerenciamento de Carros

Cars

Adicionar Novo Carro

Modelo

Fabricante

Ano

Cor

Cavalos de Potência

País

Editar Carro

Golf GTI

Volkswagen

2025

Branco

380

Alemanha

Salvar

Cancelar

- Golf GTI - Volkswa

Editar

Deletar

- Polo - Volkswagen

Editar

Deletar

- Passat - Volkswag

Editar

Deletar

Detalhar

- Tiguan - Volkswagen - 2022

Editar

Deletar

Detalhar

Após editar, o carro mudou para Golf GTI, ano 2025, um belo upgrade hein 😂

- Golf GTI - Volkswagen - 2025

Editar

Deletar

Detalhar

Tela de Detalhar

Gerenciamento de Carros

Cars

Adicionar Novo Carro

Modelo

Fabricante

Ano

Cor

Cavalos de Potência

País

Golf GTI

Fabricante: Volkswagen

Ano: 2025

Cor: Branco

Cavalos de Potência: 380

País: Alemanha

Fechar

- Golf GTI - Volkswagen

Editar

Deletar

- Polo - Volkswagen

Editar

Deletar

Detalhar

- Passat - Volkswagen - 2021

Editar

Deletar

Detalhar

- Tiguan - Volkswagen - 2022

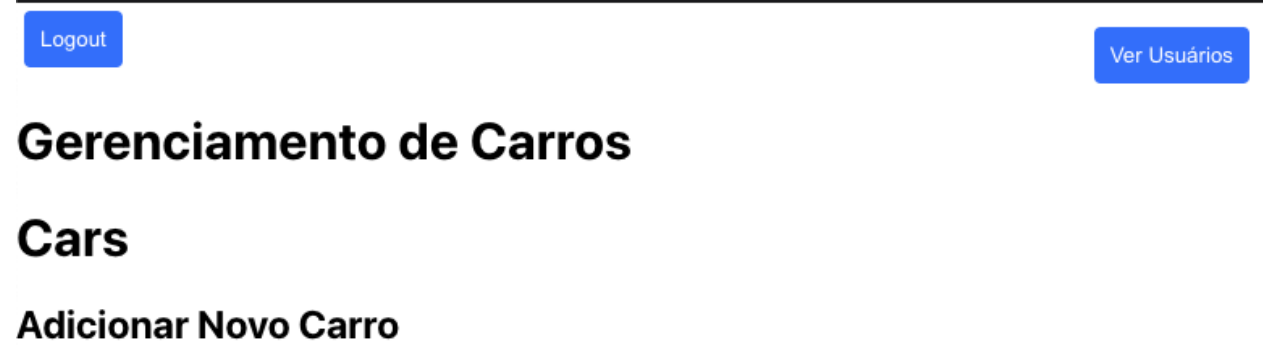
Editar

Deletar

Detalhar

- Fusion - Ford - 2021

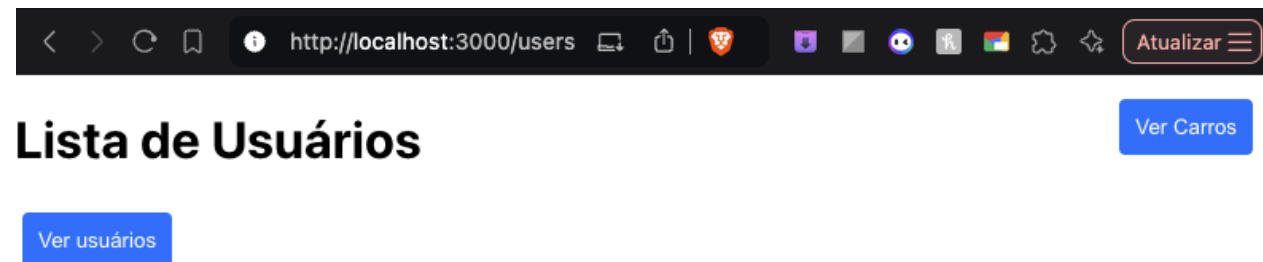
No topo da tela:



Botão para Logout, faz o logout e direciona para a tela de login novamente.

Ao clicar em usuários, será redirecionado para a tela dos Users, onde é possível visualizar todos os usuários cadastrados na API

Na rota /users



Lista de Usuários

Esconder usuários

- Wile E. Coyote
- Road Runner
- Daffy Duck
- Bugs Bunny
- Elmer Fudd