⚛️ Frontend React da Livraria — Parte 7

Vite + React + React Router + Axios

👨‍🏫 **Professor:** Fabricio Bizotto

📘 **Disciplina:** Desenvolvimento Web I

🎓 **Curso:** Ciência da Computação

📅 **Fase:** 4ª fase

Roteiro

- Stack e dependências
- Estrutura de pastas
- Configurando Vite + Proxy
- Axios (instância, interceptor de 401)
- Contexto de Autenticação (AuthContext)
- Protegendo rotas (PrivateRoute)
- Páginas: Login, Register, Home, Livros (CRUD)
- Serviços de API (authService, livrosService)
- Executando o frontend
- Resumo das mudanças

Stack e dependências

```
cd frontend
npm install
```

- React 18
- React Router DOM 6
- Axios 1.x
- Vite 5

> Projeto criado em `frontend/` e integrado ao backend via proxy.

Estrutura de pastas

```
frontend/
├── src/
│   ├── components/
│   │   ├── Header.jsx / Header.css
│   │   ├── LivroCard.jsx / LivroCard.css
│   │   ├── LivroForm.jsx / LivroForm.css
│   │   └── PrivateRoute.jsx
│   ├── contexts/
│   │   └── AuthContext.jsx
│   ├── pages/
│   │   ├── Login.jsx / Register.jsx / Auth.css
│   │   ├── Home.jsx / Home.css
│   │   └── Livros.jsx / Livros.css
│   ├── services/
│   │   ├── api.js
│   │   ├── authService.js
│   │   └── livrosService.js
│   ├── App.jsx / App.css
│   └── main.jsx / index.css
├── vite.config.js
├── index.html
└── package.json
```

Vite + Proxy para o backend

```
// frontend/vite.config.js
import { defineConfig } from 'vite'
import react from '@vitejs/plugin-react'

export default defineConfig({
  plugins: [react()],
  server: {
    port: 3000,
    proxy: {
      '/api': {
        target: 'http://localhost:3333',
        changeOrigin: true,
      }
    }
  }
})
```

> Todas as chamadas a `/api` são roteadas ao backend (porta 3333).

Axios: instância + interceptor 401

```
// frontend/src/services/api.js
import axios from 'axios';

const api = axios.create({
  baseURL: '/api',
  withCredentials: true,
  headers: { 'Content-Type': 'application/json' }
});

api.interceptors.response.use(
  (res) => res,
  (error) => {
    if (error.response?.status === 401) {
      const publicRoutes = ['/login', '/register'];
      const currentPath = window.location.pathname;
      if (!publicRoutes.includes(currentPath)) {
        window.location.href = '/login';
      }
    }
    return Promise.reject(error);
  }
);

export default api;
```

`withCredentials: true` mantém sessão (cookies) via proxy.

AuthContext: estado global de autenticação

```jsx
// frontend/src/contexts/AuthContext.jsx
import React, { createContext, useState, useContext, useEffect } from 'react';
import { authService } from '../services/authService';

const AuthContext = createContext({});

export const AuthProvider = ({ children }) => {
  const [user, setUser] = useState(null);
  const [loading, setLoading] = useState(true);

  useEffect(() => { checkAuth(); }, []);

  const checkAuth = async () => {
    try { setUser(await authService.getMe()); }
    catch { setUser(null); }
    finally { setLoading(false); }
  };

  const login = async (credentials) => {
    const data = await authService.login(credentials);
    setUser(data.user);
  };

  const register = async (userData) => {
    return authService.register(userData);
  };

  const logout = async () => {
    try { await authService.logout(); } finally { setUser(null); }
  };

  return (
    <AuthContext.Provider value={{ user, loading, login, register, logout, checkAuth }}>
      {children}
    </AuthContext.Provider>
  );
};

export const useAuth = () => useContext(AuthContext);
```

Protegendo rotas: PrivateRoute

```jsx
// frontend/src/components/PrivateRoute.jsx
import React from 'react';
import { Navigate } from 'react-router-dom';
import { useAuth } from '../contexts/AuthContext';

const PrivateRoute = ({ children }) => {
  const { user, loading } = useAuth();
  if (loading) return <div className="loading">Carregando...</div>;
  return user ? children : <Navigate to="/login" replace />;
};
export default PrivateRoute;
```

Rotas da aplicação

```
// frontend/src/App.jsx
import { BrowserRouter as Router, Routes, Route, Navigate } from 'react-router-dom'
import PrivateRoute from './components/PrivateRoute'
import Header from './components/Header'
import Login from './pages/Login'
import Register from './pages/Register'
import Home from './pages/Home'
import Livros from './pages/Livros'

<Router>
  <Header />
  <Routes>
    <Route path="/login" element={<Login />} />
    <Route path="/register" element={<Register />} />
    <Route path="/" element={<PrivateRoute><Home /></PrivateRoute>} />
    <Route path="/livros" element={<PrivateRoute><Livros /></PrivateRoute>} />
    <Route path="*" element={<Navigate to="/" replace />} />
  </Routes>
</Router>
```

Serviços de API

```
// frontend/src/services/authService.js
import api from './api';
export const authService = {
  register: (data) => api.post('/auth/register', data).then(r => r.data),
  login: (data) => api.post('/auth/login', data).then(r => r.data),
  logout: () => api.post('/auth/logout').then(r => r.data),
  getMe: () => api.get('/auth/me').then(r => r.data)
};
```

```
// frontend/src/services/livrosService.js
import api from './api';
export const livrosService = {
  listar: () => api.get('/livros').then(r => r.data),
  buscarPorId: (id) => api.get(`/livros/${id}`).then(r => r.data),
  criar: (livro) => api.post('/livros', livro).then(r => r.data),
  atualizar: (id, livro) => api.put(`/livros/${id}`, livro).then(r => r.data),
  remover: (id) => api.delete(`/livros/${id}`).then(r => r.data)
};
```

Página de Login (com redirecionamento se autenticado)

```
// frontend/src/pages/Login.jsx (trecho)
const { login, user } = useAuth();
useEffect(() => { if (user) navigate('/'); }, [user, navigate]);

const handleSubmit = async (e) => {
  e.preventDefault();
  setError(''); setLoading(true);
  try { await login(formData); navigate('/'); }
  catch (err) { setError(err.response?.data?.erro || 'Erro ao fazer login.'); }
  finally { setLoading(false); }
};
```

Página de Registro

```
// frontend/src/pages/Register.jsx (trecho)
const { register, user } = useAuth();
useEffect(() => { if (user) navigate('/'); }, [user, navigate]);

const handleSubmit = async (e) => {
  e.preventDefault();
  if (formData.password !== formData.confirmPassword) {
    setError('As senhas não coincidem'); return;
  }
  try {
    const { confirmPassword, ...data } = formData;
    await register(data);
    navigate('/login');
  } catch (err) {
    setError(err.response?.data?.erro || 'Erro ao criar conta.');
  }
};
```

Página de Livros (CRUD)

```
// frontend/src/pages/Livros.jsx (trecho)
const [livros, setLivros] = useState([]);
const carregarLivros = async () => {
  try { setLivros(await livrosService.listar()); }
  catch { setError('Erro ao carregar livros.'); }
};

const handleSubmit = async (formData) => {
  try {
    if (editingLivro) await livrosService.atualizar(editingLivro.id, formData);
    else await livrosService.criar(formData);
    setShowForm(false); setEditingLivro(null); carregarLivros();
  } catch (err) { setError(err.response?.data?.erro || 'Erro ao salvar livro.'); }
};
```

Executando o frontend

```
# Em um terminal (backend)
npm run dev          # porta 3333

# Em outro terminal (frontend)
cd frontend
npm install          # primeira vez
npm run dev          # porta 3000
```

- Acesse: http://localhost:3000
- Registre-se, faça login e gerencie seus livros

Resumo

- Frontend React com Vite + Router + Axios
- Sessões mantidas via proxy e `withCredentials`
- Rotas privadas com `PrivateRoute`
- Context API para estado de autenticação
- Páginas para autenticação e CRUD de livros

> Próximos passos: paginação, busca/filtro, feedback com toasts, testes.