<!DOCTYPE html>

<html lang="pt-br">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Login com Firebase</title>

<script src="https://cdn.tailwindcss.com"></script>

<style>

@import url('https://fonts.googleapis.com/css2?family=Inter:wght@400;600;700&display=swap');

body {

font-family: 'Inter', sans-serif;

background-color: #1e293b;

}

</style>

</head>

<body class="flex items-center justify-center min-h-screen p-4 text-white">

<div id="app" class="w-full max-w-sm p-8 space-y-6 bg-slate-800 rounded-xl shadow-lg border border-slate-700">

<h1 class="text-3xl font-bold text-center text-white mb-6">Autenticação Firebase</h1>

<!-- Mensagem de feedback do usuário -->

<div id="message-box" class="hidden p-4 rounded-md text-sm" role="alert"></div>

<div id="loading-spinner" class="hidden flex justify-center items-center">

<div class="animate-spin rounded-full h-8 w-8 border-b-2 border-white"></div>

</div>

<!-- Formulário de Login -->

<div id="login-form-container" class="space-y-4">

<div>

<label for="email" class="block text-sm font-medium text-gray-400">E-mail</label>

<input type="email" id="email" placeholder="seu-email@exemplo.com"

class="mt-1 block w-full px-3 py-2 bg-slate-700 border border-slate-600 rounded-md shadow-sm placeholder-gray-500 focus:outline-none focus:ring-blue-500 focus:border-blue-500 sm:text-sm">

</div>

<div>

<label for="password" class="block text-sm font-medium text-gray-400">Senha</label>

<input type="password" id="password" placeholder="Sua senha"

class="mt-1 block w-full px-3 py-2 bg-slate-700 border border-slate-600 rounded-md shadow-sm placeholder-gray-500 focus:outline-none focus:ring-blue-500 focus:border-blue-500 sm:text-sm">

</div>

<button id="login-button"

class="w-full flex justify-center py-2 px-4 border border-transparent rounded-md shadow-sm text-sm font-medium text-white bg-blue-600 hover:bg-blue-700 focus:outline-none focus:ring-2 focus:ring-offset-2 focus:ring-blue-500 transition-colors">

Entrar

</button>

<button id="signup-button"

class="w-full flex justify-center py-2 px-4 border border-transparent rounded-md shadow-sm text-sm font-medium text-white bg-green-600 hover:bg-green-700 focus:outline-none focus:ring-2 focus:ring-offset-2 focus:ring-green-500 transition-colors">

Criar Conta

</button>

</div>

<!-- Conteúdo do usuário após o login -->

<div id="auth-content" class="text-center space-y-4 hidden">

<p id="user-status" class="text-lg font-semibold text-green-400">Você está logado!</p>

<p id="user-id" class="break-all font-mono text-xs p-2 bg-slate-700 rounded-md shadow-inner"></p>

<p class="text-xs text-slate-400">Este é o seu ID de usuário. Em uma aplicação real, ele é usado para identificar seus dados no banco de dados.</p>

<button id="logout-button"

class="w-full flex justify-center py-2 px-4 border border-transparent rounded-md shadow-sm text-sm font-medium text-white bg-red-600 hover:bg-red-700 focus:outline-none focus:ring-2 focus:ring-offset-2 focus:ring-red-500 transition-colors">

Sair

</button>

</div>

<!-- Conteúdo de fallback para quando as variáveis do Firebase não estiverem disponíveis -->

<div id="fallback-message" class="hidden text-center p-4 rounded-md bg-red-500/20 text-red-400">

<p><strong>Erro de Carregamento</strong></p>

<p class="text-sm mt-2">O aplicativo não pôde se conectar ao Firebase. Isso pode acontecer ao executar o arquivo localmente.</p>

<p class="text-sm mt-2">Para funcionar, o site precisa ser hospedado em um ambiente web.</p>

</div>

</div>

<!-- Firebase SDKs -->

<script type="module">

import { initializeApp } from "<https://www.gstatic.com/firebasejs/11.6.1/firebase-app.js>";

import { getAuth, signInWithEmailAndPassword, onAuthStateChanged, signOut, createUserWithEmailAndPassword } from "<https://www.gstatic.com/firebasejs/11.6.1/firebase-auth.js>";

import { getFirestore, getDoc, doc, setDoc, setLogLevel } from "<https://www.gstatic.com/firebasejs/11.6.1/firebase-firestore.js>";

// Ativa o log de depuração do Firestore para ajudar a ver o que está acontecendo

setLogLevel('debug');

// Referências para os elementos da UI

const messageBox = document.getElementById('message-box');

const loadingSpinner = document.getElementById('loading-spinner');

const loginFormContainer = document.getElementById('login-form-container');

const loginButton = document.getElementById('login-button');

const signupButton = document.getElementById('signup-button');

const logoutButton = document.getElementById('logout-button');

const emailInput = document.getElementById('email');

const passwordInput = document.getElementById('password');

const authContent = document.getElementById('auth-content');

const userStatus = document.getElementById('user-status');

const userIdElem = document.getElementById('user-id');

const fallbackMessage = document.getElementById('fallback-message');

// Função para exibir mensagens na UI

function showMessage(message, type = 'info') {

messageBox.textContent = message;

messageBox.className = `p-4 rounded-md text-sm ${type === 'error' ? 'bg-red-500/20 text-red-400' : 'bg-green-500/20 text-green-400'}`;

messageBox.classList.remove('hidden');

}

// Configuração e inicialização do Firebase

const appId = typeof \_\_app\_id !== 'undefined' ? \_\_app\_id : 'default-app-id';

const firebaseConfig = JSON.parse(typeof \_\_firebase\_config !== 'undefined' ? \_\_firebase\_config : '{}');

let auth, db;

if (Object.keys(firebaseConfig).length > 0) {

fallbackMessage.classList.add('hidden'); // Esconde a mensagem de fallback

try {

const app = initializeApp(firebaseConfig);

auth = getAuth(app);

db = getFirestore(app);

// Mostra o spinner de carregamento enquanto o login é processado

loadingSpinner.classList.remove('hidden');

// Listener para o estado de autenticação (login/logout)

onAuthStateChanged(auth, async (user) => {

loadingSpinner.classList.add('hidden');

if (user) {

// Usuário logado

loginFormContainer.classList.add('hidden');

authContent.classList.remove('hidden');

userIdElem.textContent = user.uid;

showMessage('Login bem-sucedido!', 'success');

// Exemplo de como salvar/ler dados após o login

const userDocRef = doc(db, `artifacts/${appId}/users/${user.uid}/user-data/profile`);

await setDoc(userDocRef, {

lastLogin: new Date().toISOString(),

email: user.email

}, { merge: true });

console.log("Perfil do usuário salvo.");

} else {

// Usuário deslogado

loginFormContainer.classList.remove('hidden');

authContent.classList.add('hidden');

userIdElem.textContent = '';

showMessage('Faça login ou crie uma conta para continuar.', 'info');

}

});

} catch (error) {

console.error("Erro ao inicializar o Firebase:", error);

showMessage(`Erro fatal: ${error.message}`, 'error');

loadingSpinner.classList.add('hidden');

}

} else {

// Se as variáveis do Firebase não estiverem disponíveis, mostra a mensagem de fallback

console.error("Configuração do Firebase não encontrada.");

loadingSpinner.classList.add('hidden');

fallbackMessage.classList.remove('hidden');

}

// Evento de clique do botão de login

loginButton.addEventListener('click', async () => {

const email = emailInput.value.trim();

const password = passwordInput.value.trim();

if (!email || !password) {

showMessage('Por favor, preencha todos os campos.', 'error');

return;

}

loadingSpinner.classList.remove('hidden');

try {

await signInWithEmailAndPassword(auth, email, password);

} catch (error) {

loadingSpinner.classList.add('hidden');

console.error("Erro no login:", error);

let errorMessage = 'Ocorreu um erro no login.';

switch (error.code) {

case 'auth/user-not-found':

case 'auth/wrong-password':

errorMessage = 'E-mail ou senha inválidos.';

break;

case 'auth/invalid-email':

errorMessage = 'E-mail inválido.';

break;

default:

errorMessage = 'Erro no login. Tente novamente.';

}

showMessage(errorMessage, 'error');

}

});

// Evento de clique do botão de cadastro

signupButton.addEventListener('click', async () => {

const email = emailInput.value.trim();

const password = passwordInput.value.trim();

if (!email || !password) {

showMessage('Por favor, preencha todos os campos.', 'error');

return;

}

loadingSpinner.classList.remove('hidden');

try {

await createUserWithEmailAndPassword(auth, email, password);

showMessage('Conta criada com sucesso! Faça o login agora.', 'success');

} catch (error) {

loadingSpinner.classList.add('hidden');

console.error("Erro no cadastro:", error);

let errorMessage = 'Ocorreu um erro no cadastro.';

switch (error.code) {

case 'auth/email-already-in-use':

errorMessage = 'Este e-mail já está em uso.';

break;

case 'auth/weak-password':

errorMessage = 'A senha deve ter pelo menos 6 caracteres.';

break;

case 'auth/invalid-email':

errorMessage = 'E-mail inválido.';

break;

default:

errorMessage = 'Erro no cadastro. Tente novamente.';

}

showMessage(errorMessage, 'error');

}

});

// Evento de clique do botão de logout

logoutButton.addEventListener('click', async () => {

try {

await signOut(auth);

showMessage('Você saiu com sucesso.', 'success');

} catch (error) {

console.error("Erro ao fazer logout:", error);

showMessage('Erro ao sair. Tente novamente.', 'error');

}

});

</script>

</body>

</html>