<!DOCTYPE html>

<html lang="pt-BR">

<head>

<meta charset="UTF-8">

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

<title>Relatório de Inscrições em Tempo Real - Evento Dia dos Namorados</title>

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

<link href="https://fonts.googleapis.com/css2?family=Inter:wght@400;600;700&family=Playfair+Display:wght@700&display=swap" rel="stylesheet">

<style>

body {

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

background-color: #f9fafb;

}

.playfair {

font-family: 'Playfair Display', serif;

}

th {

background-color: #ef4444; /\* Red-500 \*/

color: white;

padding: 12px;

text-align: left;

position: sticky; /\* Para cabeçalho fixo ao rolar \*/

top: 0;

z-index: 10;

}

td {

padding: 12px;

border-bottom: 1px solid #e5e7eb; /\* Gray-200 \*/

}

tr:nth-child(even) {

background-color: #fde8e8; /\* Lighter red for even rows \*/

}

tr:hover {

background-color: #fecdd3; /\* Rose-200 for hover (corrigido de 'f розовый-200') \*/

}

.table-container {

max-width: 1200px;

margin: 0 auto;

padding: 20px;

background-color: white;

border-radius: 12px;

box-shadow: 0 10px 15px -3px rgba(0, 0, 0, 0.1), 0 4px 6px -2px rgba(0, 0, 0, 0.05);

max-height: 70vh; /\* Altura máxima para habilitar scroll da tabela \*/

overflow-y: auto; /\* Scroll vertical para a tabela \*/

}

.loading-message, .empty-message {

text-align: center;

padding: 20px;

font-size: 1.2em;

color: #555;

}

</style>

</head>

<body class="bg-rose-50 text-gray-800">

<script type="module">

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

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

import { getFirestore, collection, query as firestoreQuery, orderBy, onSnapshot, setLogLevel } from "<https://www.gstatic.com/firebasejs/11.6.1/firebase-firestore.js>"; // Mudado para onSnapshot

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

apiKey: "YOUR\_API\_KEY",

authDomain: "YOUR\_AUTH\_DOMAIN",

projectId: "YOUR\_PROJECT\_ID",

// storageBucket: "YOUR\_STORAGE\_BUCKET",

// messagingSenderId: "YOUR\_MESSAGING\_SENDER\_ID",

// appId: "YOUR\_APP\_ID"

};

const appId = typeof \_\_app\_id !== 'undefined' ? \_\_app\_id : 'valentines-event-2025';

let app, auth, db;

let unsubscribeRegistrations = null; // Para parar de ouvir quando necessário

try {

app = initializeApp(firebaseConfig);

auth = getAuth(app);

db = getFirestore(app);

setLogLevel('debug');

console.log("Firebase inicializado com sucesso para o relatório em tempo real.");

} catch (error) {

console.error("Erro na inicialização do Firebase para o relatório:", error);

const reportTableBody = document.getElementById('reportTableBody');

if (reportTableBody) {

reportTableBody.innerHTML = '<tr><td colspan="8" class="text-center text-red-500">Erro ao inicializar o sistema de relatório. Verifique a configuração do Firebase.</td></tr>';

}

}

function formatFirebaseTimestamp(timestamp) {

if (timestamp && timestamp.toDate) {

const date = timestamp.toDate();

return date.toLocaleString('pt-BR', {

day: '2-digit', month: '2-digit', year: 'numeric',

hour: '2-digit', minute: '2-digit'

});

}

return 'N/D';

}

function translatePaymentMethod(method) {

if (!method) return 'Não escolhido';

switch(method) {

case 'pix\_copied': return 'PIX (Chave Copiada)';

case 'card\_in\_person': return 'Cartão (Presencial)';

default: return method;

}

}

function translatePaymentStatus(status) {

// O status real de pagamento (pago/não pago) precisaria ser atualizado

// por um processo externo após a verificação do pagamento.

// Por enquanto, 'pending' significa que o usuário se inscreveu.

if (status === 'paid') return 'Pago (Confirmado)'; // Exemplo, se você atualizar esse status

if (status === 'pending') return 'Pendente';

return status || 'N/D';

}

function listenForRegistrations() {

const reportTableBody = document.getElementById('reportTableBody');

const loadingMessage = document.getElementById('loadingMessage');

const emptyMessage = document.getElementById('emptyMessage');

if (!db) {

console.error("Conexão com Firestore não estabelecida.");

if (loadingMessage) loadingMessage.style.display = 'none';

if (reportTableBody) reportTableBody.innerHTML = '<tr><td colspan="8" class="text-center text-red-500">Erro: Conexão com o banco de dados não estabelecida.</td></tr>';

return;

}

if (loadingMessage) loadingMessage.style.display = 'block';

if (emptyMessage) emptyMessage.style.display = 'none';

const registrationsCollectionPath = `/artifacts/${appId}/public/data/registrations\_valentines\_event\_2025`;

const q = firestoreQuery(collection(db, registrationsCollectionPath)); // orderBy('registeredAt', 'desc') pode exigir índice

// Cancela o listener anterior, se houver, para evitar múltiplos listeners

if (unsubscribeRegistrations) {

unsubscribeRegistrations();

}

unsubscribeRegistrations = onSnapshot(q, (querySnapshot) => {

if (loadingMessage) loadingMessage.style.display = 'none';

if (reportTableBody) reportTableBody.innerHTML = ''; // Limpa a tabela a cada atualização

if (querySnapshot.empty) {

if (emptyMessage) emptyMessage.style.display = 'block';

console.log("Nenhuma inscrição encontrada em tempo real.");

} else {

if (emptyMessage) emptyMessage.style.display = 'none';

let registrations = [];

querySnapshot.forEach((doc) => {

registrations.push({ id: doc.id, ...doc.data() });

});

// Ordenar no lado do cliente por data de registro (mais recente primeiro)

registrations.sort((a, b) => {

const dateA = a.registeredAt ? a.registeredAt.toMillis() : 0;

const dateB = b.registeredAt ? b.registeredAt.toMillis() : 0;

return dateB - dateA; // Descendente

});

registrations.forEach((data) => {

const row = reportTableBody.insertRow();

row.innerHTML = `

<td>${data.name || 'N/D'}</td>

<td>${data.email || 'N/D'}</td>

<td>${data.phone || 'N/D'}</td>

<td>${data.registrationType === 'casal' ? 'Casal' : 'Individual'}</td>

<td>R$ ${data.amount ? data.amount.toFixed(2) : '0.00'}</td>

<td>${translatePaymentMethod(data.paymentMethodChosen)}</td>

<td class="${data.paymentStatus === 'pending' ? 'text-orange-600' : (data.paymentStatus === 'paid' ? 'text-green-600' : 'text-gray-600')}">${translatePaymentStatus(data.paymentStatus)}</td>

<td>${formatFirebaseTimestamp(data.registeredAt)}</td>

`;

});

console.log("Relatório atualizado com novas inscrições/alterações.");

}

}, (error) => {

console.error("Erro ao ouvir inscrições em tempo real: ", error);

if (loadingMessage) loadingMessage.style.display = 'none';

if (reportTableBody) reportTableBody.innerHTML = `<tr><td colspan="8" class="text-center text-red-500">Erro ao carregar dados em tempo real: ${error.message}</td></tr>`;

});

}

onAuthStateChanged(auth, (user) => {

if (user) {

console.log("Usuário autenticado para relatório (UID):", user.uid);

listenForRegistrations();

} else {

signInAnonymously(auth)

.then(() => {

console.log("Login anônimo realizado para relatório.");

listenForRegistrations();

})

.catch((error) => {

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

if (document.getElementById('reportTableBody')) document.getElementById('reportTableBody').innerHTML = '<tr><td colspan="8" class="text-center text-red-500">Erro de autenticação para carregar o relatório.</td></tr>';

});

}

});

// Opcional: Limpar o listener quando a página for descarregada

window.addEventListener('beforeunload', () => {

if (unsubscribeRegistrations) {

unsubscribeRegistrations();

console.log("Listener de inscrições removido.");

}

});

</script>

<header class="bg-gray-800 text-white py-8">

<div class="container mx-auto px-6 text-center">

<h1 class="playfair text-3xl md:text-4xl font-bold">Relatório de Inscrições em Tempo Real</h1>

<p class="text-lg">Evento Dia dos Namorados - Assembleia de Deus - Ministério Amor de Deus</p>

</div>

</header>

<main class="container mx-auto px-4 sm:px-6 lg:px-8 py-12">

<div class="table-container">

<div id="loadingMessage" class="loading-message">Carregando inscrições...</div>

<div id="emptyMessage" class="emptyMessage" style="display:none;">Nenhuma inscrição encontrada.</div>

<table class="min-w-full divide-y divide-gray-200">

<thead>

<tr>

<th>Nome</th>

<th>Email</th>

<th>Telefone</th>

<th>Tipo</th>

<th>Valor</th>

<th>Pag. Escolhido</th>

<th>Status Pag.</th>

<th>Data Inscrição</th>

</tr>

</thead>

<tbody id="reportTableBody" class="bg-white divide-y divide-gray-200">

</tbody>

</table>

</div>

<p class="text-center mt-4 text-sm text-gray-600">Este relatório é atualizado automaticamente.</p>

</main>

<footer class="bg-gray-800 text-white py-10 text-center mt-12">

<p>&copy; ${new Date().getFullYear()} Assembleia de Deus - Ministério Amor de Deus.</p>

</footer>

</body>

</html>