import React, { useState, useEffect, useCallback } from 'react';  
// Importuri Firebase (neschimbat)  
import { initializeApp } from "https://www.gstatic.com/firebasejs/11.6.1/firebase-app.js";  
import { getAuth, onAuthStateChanged, createUserWithEmailAndPassword, signInWithEmailAndPassword, GoogleAuthProvider, signInWithPopup, signOut } from "https://www.gstatic.com/firebasejs/11.6.1/firebase-auth.js";  
import { getFirestore, collection, doc, addDoc, onSnapshot, setDoc } from "https://www.gstatic.com/firebasejs/11.6.1/firebase-firestore.js";  
  
// --- Configurare și Inițializare Firebase (neschimbat) ---  
const firebaseConfig = { /\* ... va fi injectat de mediu ... \*/ };  
const app = initializeApp(firebaseConfig);  
const auth = getAuth(app);  
const db = getFirestore(app);  
const googleProvider = new GoogleAuthProvider();  
  
// --- Icoane (adăugăm icoane noi pentru notificări) ---  
const SparklesIcon=()=>(<svg xmlns="http://www.w3.org/2000/svg" width="24" height="24" viewBox="0 0 24 24" fill="none" stroke="currentColor" strokeWidth="2" strokeLinecap="round" strokeLinejoin="round"><path d="M12 3L9.5 8.5L4 11L9.5 13.5L12 19L14.5 13.5L20 11L14.5 8.5L12 3z"/><path d="M5 3v4"/><path d="M19 17v4"/><path d="M3 5h4"/><path d="M17 19h4"/></svg>);  
const GoogleIcon=()=>(<svg role="img" viewBox="0 0 24 24" xmlns="http://www.w3.org/2000/svg" className="w-5 h-5"><title>Google</title><path d="M12.48 10.92v3.28h7.84c-.24 1.84-.85 3.18-1.73 4.1-1.02 1.02-2.3 1.63-4.5 1.63-5.52 0-10-4.48-10-10s4.48-10 10-10c3.1 0 5.19 1.24 6.38 2.38l-2.13 2.12c-.94-.88-2.18-1.5-4.25-1.5-3.6 0-6.5 2.9-6.5 6.5s2.9 6.5 6.5 6.5c4.08 0 5.98-2.8 6.25-4.38h-6.25z" fill="currentColor"/></svg>);  
const LogoutIcon=()=>(<svg xmlns="http://www.w3.org/2000/svg" width="20" height="20" viewBox="0 0 24 24" fill="none" stroke="currentColor" strokeWidth="2" strokeLinecap="round" strokeLinejoin="round"><path d="M9 21H5a2 2 0 0 1-2-2V5a2 2 0 0 1 2-2h4"></path><polyline points="16 17 21 12 16 7"></polyline><line x1="21" y1="12" x2="9" y2="12"></line></svg>);  
const CheckCircleIcon=()=>(<svg xmlns="http://www.w3.org/2000/svg" width="20" height="20" viewBox="0 0 24 24" fill="none" stroke="currentColor" strokeWidth="2" strokeLinecap="round" strokeLinejoin="round"><path d="M22 11.08V12a10 10 0 1 1-5.93-9.14"></path><polyline points="22 4 12 14.01 9 11.01"></polyline></svg>);  
const XCircleIcon=()=>(<svg xmlns="http://www.w3.org/2000/svg" width="20" height="20" viewBox="0 0 24 24" fill="none" stroke="currentColor" strokeWidth="2" strokeLinecap="round" strokeLinejoin="round"><circle cx="12" cy="12" r="10"></circle><line x1="15" y1="9" x2="9" y2="15"></line><line x1="9" y1="9" x2="15" y2="15"></line></svg>);  
  
// --- Serviciu Backend (Simulare Apel Funcție Cloud) ---  
const backendService = {  
 // Această funcție ar face un apel 'fetch' către o funcție Cloud reală.  
 startPackageGeneration: async (userId, workshopId) => {  
 console.log(`[Frontend] Se trimite cererea de generare pentru user: ${userId}, workshop: ${workshopId}`);  
   
 // Simulăm o întârziere a rețelei și a procesării în backend  
 return new Promise(resolve => {  
 setTimeout(() => {  
 console.log(`[Backend] Procesare finalizată. Se returnează succes.`);  
 // Într-o aplicație reală, funcția Cloud ar adăuga singură pachetul în Firestore.  
 // Aici simulăm acest comportament.  
 resolve({ success: true, message: 'Pachetul a fost generat și adăugat în hub-ul tău!' });  
 }, 4000); // O întârziere mai lungă pentru a simula un job real  
 });  
 }  
};  
  
// --- Notificare Toast (Componentă Nouă) ---  
const ToastNotification = ({ notification, onDismiss }) => {  
 useEffect(() => {  
 if (notification) {  
 const timer = setTimeout(() => {  
 onDismiss();  
 }, 5000);  
 return () => clearTimeout(timer);  
 }  
 }, [notification, onDismiss]);  
  
 if (!notification) return null;  
  
 const isSuccess = notification.type === 'success';  
  
 return (  
 <div className="fixed bottom-5 right-5 z-50">  
 <div className={`flex items-center p-4 rounded-lg shadow-lg text-white ${isSuccess ? 'bg-green-500' : 'bg-red-500'}`}>  
 <div className="mr-3">{isSuccess ? <CheckCircleIcon /> : <XCircleIcon />}</div>  
 <p>{notification.message}</p>  
 <button onClick={onDismiss} className="ml-4 font-bold">X</button>  
 </div>  
 </div>  
 );  
};  
  
  
// --- Pagina de Autentificare (neschimbat) ---  
const AuthPage = () => { /\* ... \*/ return <div></div>; };  
  
// --- Navbar (neschimbat) ---  
const Navbar = ({ user }) => { /\* ... \*/ return <div></div>; };  
  
// --- MainApp (Componenta principală după login) ---  
const MainApp = ({ user }) => {  
 const [page, setPage] = useState('dashboard');  
 const [selectedWorkshopId, setSelectedWorkshopId] = useState(null);  
 const [notification, setNotification] = useState(null);  
  
 const showNotification = (message, type = 'success') => {  
 setNotification({ message, type });  
 };  
  
 const handleGeneratePackage = async (workshopId) => {  
 showNotification("Procesul de generare a început...", 'success');  
 const result = await backendService.startPackageGeneration(user.uid, workshopId);  
 if (result.success) {  
 showNotification(result.message, 'success');  
 setPage('dashboard'); // Ne întoarcem la dashboard pentru a vedea noul pachet  
 } else {  
 showNotification(result.message, 'error');  
 }  
 };  
  
 const renderPage = () => {  
 switch (page) {  
 case 'editor':  
 return <WorkshopEditor   
 userId={user.uid}   
 workshopId={selectedWorkshopId}   
 onGeneratePackage={handleGeneratePackage}   
 />;  
 default:  
 return <Dashboard   
 userId={user.uid}   
 setPage={setPage}   
 setSelectedWorkshopId={setSelectedWorkshopId}   
 />;  
 }  
 };  
   
 return (  
 <div className="min-h-screen bg-gray-100 font-sans">  
 <Navbar user={user} />  
 <main>  
 <div className="max-w-7xl mx-auto py-6">{renderPage()}</div>  
 <ToastNotification notification={notification} onDismiss={() => setNotification(null)} />  
 </main>  
 </div>  
 );  
};  
  
// --- WorkshopEditor (Actualizat pentru a apela serviciul de backend) ---  
const WorkshopEditor = ({ userId, workshopId, onGeneratePackage }) => {  
 const [workshop, setWorkshop] = useState(null);  
 const [isLoading, setIsLoading] = useState(true);  
  
 // Încărcare date din Firestore (neschimbat)  
 useEffect(() => {  
 // ... logica onSnapshot ...  
 setIsLoading(false); // Placeholder  
 }, [userId, workshopId]);  
   
 if (isLoading || !workshop) {  
 // Simulare date pentru a vedea interfața  
 if(!workshop) setWorkshop({ blueprint: { title: 'Atelier de Test' }});  
 return <div>Se încarcă...</div>  
 }  
  
 return (  
 <div className="p-8">  
 <div className="flex justify-between items-center mb-6">  
 <h2 className="text-2xl font-bold">{workshop.blueprint.title}</h2>  
 <button   
 onClick={() => onGeneratePackage(workshopId)}  
 className="flex items-center px-6 py-2 bg-green-600 text-white rounded-lg shadow-md hover:bg-green-700"  
 >  
 <span className="mr-2">🚀</span> Generează  
 </button>  
 </div>  
 <div className="bg-white p-6 rounded-lg shadow mt-4">  
 <p>Aici este editorul atelierului. Butonul "Generează" va declanșa acum un proces asincron în backend și vă va notifica la finalizare.</p>  
 </div>  
 </div>  
 );  
};  
  
// Placeholder pentru Dashboard  
const Dashboard = ({ userId, setPage, setSelectedWorkshopId }) => {  
 return (  
 <div className="p-8">  
 <h2 className="text-2xl font-bold">Tabloul de Bord pentru {userId.slice(0,10)}...</h2>  
 <p>Aici vor apărea atelierele tale.</p>  
 <button onClick={() => { setSelectedWorkshopId('ws\_test\_123'); setPage('editor'); }} className="mt-4 px-4 py-2 bg-blue-600 text-white rounded-md">Deschide Atelier de Test</button>  
 </div>  
 );  
};  
  
  
// --- Componenta Principală a Aplicației (Gestionează starea de autentificare) ---  
export default function App() {  
 const [user, setUser] = useState(null);  
 const [loading, setLoading] = useState(true);  
  
 useEffect(() => {  
 const unsubscribe = onAuthStateChanged(auth, (currentUser) => {  
 setUser(currentUser);  
 setLoading(false);  
 });  
 return () => unsubscribe();  
 }, []);  
  
 if (loading) {  
 return <div className="min-h-screen flex items-center justify-center">Se încarcă...</div>;  
 }  
   
 return user ? <MainApp user={user} /> : <AuthPage />;  
}