import React, { useState, useEffect, useCallback, useMemo } from 'react';

import {

Heart, X, Save, User, FileText, Download, Share2, MessageSquare, Edit, Loader2, ArrowRightCircle, List, ArrowLeft, Send, Users, ThumbsUp, PlusCircle

} from 'lucide-react';

// --- 0. CONFIGURATION AND CONSTANTS ---

// Palette (Dark Theme)

const COLORS = {

black: '#0B0B12',

purple: '#6C2BD9', // Imperial Lilac

lilac: '#C8A2C8', // Soft accent

ink: '#EDEAFB', // Light text

card: '#111111',

border: '#222222',

};

// Common essay prompts grouped by focus

const PROMPT\_GROUPS = {

// Group 1: Academic/Policy Focused (DAAD, Fulbright)

ACADEMIC: [

{ value: 'academic\_fit', label: 'How does your academic background fit this program?' },

{ value: 'future\_goals', label: 'Describe your future career goals and how this opportunity helps.' },

{ value: 'policy\_impact', label: 'What specific policy impact do you hope to achieve?' },

],

// Group 2: Tech/Innovation Focused (MITACS, OpenAI)

TECH: [

{ value: 'tech\_contribution', label: 'Detail your most relevant technical/research contribution.' },

{ value: 'innovative\_challenge', label: 'Tell us about an innovative challenge you overcame in tech.' },

{ value: 'leadership\_exp', label: 'Detail your most relevant leadership experience.' },

],

// Group 3: General/Global (Chevening)

GLOBAL: [

{ value: 'why\_good\_fit', label: 'Why are you a good fit for this program?' },

{ value: 'why\_org', label: 'Why did you choose this specific organization or country?' },

{ value: 'personal\_challenge', label: 'Tell us about a significant personal challenge you overcame.' },

],

};

// Fixed list of quotes about originality, now including author

const ORIGINALITY\_QUOTES = [

{ quote: "The power of a single personality will always be greater than the power of circumstances.", author: "Marcus Aurelius" },

{ quote: "Originality is simply a pair of fresh eyes.", author: "Thomas W. Higginson" },

{ quote: "Be yourself; everyone else is already taken.", author: "Oscar Wilde" },

{ quote: "The secret to creativity is knowing how to hide your sources.", author: "Albert Einstein" },

{ quote: "The chief enemy of creativity is 'good' sense.", author: "Pablo Picasso" },

{ quote: "Originality is the best form of rebellion.", author: "Mike Sacks" },

{ quote: "Don't compromise yourself. You are all you've got.", author: "Janis Joplin" },

{ quote: "To be authentic, we must be willing to be vulnerable.", author: "Brené Brown" },

{ quote: "It takes courage to grow up and become who you really are.", author: "E.E. Cummings" },

{ quote: "The best way to predict the future is to create it.", author: "Peter Drucker" },

];

// Seed Data (Initial Opportunities)

const SEED\_OPPORTUNITIES = [

{ id: 1, title: 'DAAD Helmut-Schmidt Scholarship', org: 'DAAD', summary: 'Public policy master’s in Germany.', deadline: '2025-07-31T00:00:00.000Z', logoUrl: 'https://placehold.co/300x150/004B87/FFCC00/png?text=DAAD+SHIELD', promptGroup: 'ACADEMIC' },

{ id: 2, title: 'Chevening Scholarship', org: 'UK FCDO', summary: 'One-year master’s in the UK.', deadline: '2025-11-07T00:00:00.000Z', logoUrl: 'https://placehold.co/300x150/012169/FFFFFF/png?text=CHEVENING+LOGO', promptGroup: 'GLOBAL' },

{ id: 3, title: 'Fulbright Foreign Student', org: 'US State Dept', summary: 'Graduate study in the USA.', deadline: '2025-04-15T00:00:00.000Z', logoUrl: 'https://placehold.co/300x150/B31942/0A3161/png?text=FULBRIGHT+US', promptGroup: 'ACADEMIC' },

{ id: 4, title: 'MITACS Globalink Research Internship', org: 'Mitacs', summary: '12-week research in Canada.', deadline: '2025-10-01T00:00:00.000Z', logoUrl: 'https://placehold.co/300x150/008000/FFFFFF/png?text=MITACS+RESEARCH', promptGroup: 'TECH' },

{ id: 5, title: 'OpenAI Safety Scholars', org: 'OpenAI', summary: 'Scholars program on AI safety.', deadline: '2025-05-30T00:00:00.000Z', logoUrl: 'https://placehold.co/300x150/000000/FFFFFF/png?text=OPENAI+LOGO', promptGroup: 'TECH' }

];

// Seed data for Network Posts

const SEED\_POSTS = [

{

id: 'post1', authorId: 'user-b', authorName: 'Elena V.', authorPhotoUrl: 'https://placehold.co/100x100/6C2BD9/FFFFFF?text=EV',

timestamp: new Date(Date.now() - 86400000 \* 1).toISOString(),

content: "Just spent 6 hours refining my personal statement for the Chevening application. It's a marathon, not a sprint! Anyone else applying? Would love to connect and share tips. #Chevening #UKStudy",

imageUrl: 'https://placehold.co/600x400/111111/EDEAFB?text=Study+Session', likes: 15,

comments: [ { id: 'c1', authorName: 'Carlos M.', text: 'Great work Elena! I am also applying. The leadership essay is tough.' }, { id: 'c2', authorName: 'User', text: 'You got this! Keep up the great work.' } ]

},

{

id: 'post2', authorId: 'user-c', authorName: 'John D.', authorPhotoUrl: 'https://placehold.co/100x100/D92B2B/FFFFFF?text=JD',

timestamp: new Date(Date.now() - 86400000 \* 2).toISOString(),

content: "Found this amazing opportunity for a research internship in Canada! MITACS Globalink is perfect for anyone in STEM. Deadline is approaching, don't miss out! Check it out.",

imageUrl: null, opportunityLink: 'https://www.mitacs.ca/en/programs/globalink/globalink-research-internship', likes: 42, comments: []

},

];

// Local Storage Fallback Keys

const STORAGE\_KEYS = {

PROFILE: 'nabu:profile',

OPPORTUNITIES: 'nabu:opportunities',

SWIPES: 'nabu:swipes',

DRAFTS: 'nabu:drafts',

SHARES: 'nabu:shares',

POSTS: 'nabu:posts',

};

// --- 1. UTILITIES AND MOCKS ---

const generateUUID = () => crypto.randomUUID().slice(0, 12);

const generateSocialId = () => crypto.randomUUID().slice(0, 8);

const debounce = (func, delay) => {

let timeout;

return (...args) => {

clearTimeout(timeout);

timeout = setTimeout(() => func(...args), delay);

};

};

const formatDate = (isoString) => {

if (!isoString) return 'TBA';

try {

const date = new Date(isoString);

return date.toLocaleDateString('en-US', { year: 'numeric', month: 'long', day: 'numeric' });

} catch { return 'Invalid Date'; }

};

const timeAgo = (isoString) => {

const date = new Date(isoString);

const seconds = Math.floor((new Date() - date) / 1000);

let interval = seconds / 31536000; if (interval > 1) return Math.floor(interval) + "y ago";

interval = seconds / 2592000; if (interval > 1) return Math.floor(interval) + "mo ago";

interval = seconds / 86400; if (interval > 1) return Math.floor(interval) + "d ago";

interval = seconds / 3600; if (interval > 1) return Math.floor(interval) + "h ago";

interval = seconds / 60; if (interval > 1) return Math.floor(interval) + "m ago";

return Math.floor(seconds) + "s ago";

};

const mockEssayGenerator = (profile, opportunity, prompt) => {

const { title, org } = opportunity;

const { cv, ambitions, stories, name } = profile;

const storyList = Array.isArray(stories) ? stories.join('\n- ') : (stories || 'No stories provided.');

const ambitionsList = Array.isArray(ambitions) ? ambitions.join(', ') : (ambitions || 'No interests listed.');

return `ESSAY DRAFT (LOCAL MOCK)\n\nPrompt: "${prompt || 'General fit question'}"\n\nThe ${title} at ${org} is an ideal next step in my pursuit of my core interests in ${ambitionsList}.\n\nMy background in:\n${cv || 'No CV data provided.'}\nhas uniquely prepared me to address the core challenges of this opportunity. The personal narratives, such as:\n- ${storyList}\ndemonstrate my resilience and commitment.\n\nI am confident that I can contribute effectively.\n\n---\nThis is a suggestion. Make it yours.`;

};

const callGeminiApi = async (userQuery, systemPrompt) => {

const apiKey = "";

const apiUrl = `https://generativelanguage.googleapis.com/v1beta/models/gemini-2.5-flash-preview-05-20:generateContent?key=${apiKey}`;

const payload = {

contents: [{ parts: [{ text: userQuery }] }],

systemInstruction: { parts: [{ text: systemPrompt }] },

generationConfig: { temperature: 0.6 }

};

for (let i = 0, delay = 1000; i < 3; i++, delay \*= 2) {

try {

const response = await fetch(apiUrl, { method: 'POST', headers: { 'Content-Type': 'application/json' }, body: JSON.stringify(payload) });

const result = await response.json();

const essay = result.candidates?.[0]?.content?.parts?.[0]?.text;

if (essay) return essay;

throw new Error(result.error?.message || "API response lacked text content.");

} catch (error) {

console.error(`Attempt ${i + 1} failed:`, error);

if (i < 2) await new Promise(resolve => setTimeout(resolve, delay));

else throw new Error("API failed after multiple retries.");

}

}

throw new Error("Failed to call Gemini API.");

};

// --- 2. REUSABLE COMPONENTS ---

const OriginalityLockComponent = ({ onClose }) => {

const randomQuote = useMemo(() => ORIGINALITY\_QUOTES[Math.floor(Math.random() \* ORIGINALITY\_QUOTES.length)], []);

return (

<div className="fixed inset-0 z-[9999] flex items-center justify-center p-8 bg-black/95 text-center transition-opacity duration-300">

<div className="flex flex-col items-center justify-center p-12 w-full max-w-xl bg-[var(--nabu-card)] border-4 border-[var(--nabu-purple)] rounded-3xl shadow-[0\_0\_50px\_rgba(108,43,217,0.7)] animate-fadeIn space-y-6">

<Edit className="w-16 h-16 text-[var(--nabu-purple)]"/>

<h3 className="text-3xl font-bold text-[var(--nabu-purple)] font-serif">Acknowledge Originality</h3>

<div className="bg-[#1a1a1a] p-6 rounded-xl border border-[#2a2a2a] max-w-sm">

<p className="text-xl font-semibold text-[var(--nabu-ink)] mb-4">This is merely a suggestion. Make it yours.</p>

<p className="text-sm italic text-[var(--nabu-lilac)] mt-4 font-serif">"{randomQuote.quote}"</p>

<p className="text-xs text-gray-500 mt-1 font-semibold">— {randomQuote.author}</p>

</div>

<button onClick={onClose} className="py-3 px-8 text-lg font-semibold rounded-xl bg-[var(--nabu-purple)] text-[var(--nabu-ink)] hover:bg-opacity-90 transition-opacity focus:outline-none focus:ring-4 focus:ring-[var(--nabu-purple)]">I Got It</button>

</div>

</div>

);

};

const NabuLogo = ({ navigate }) => (

<button onClick={() => navigate('/network')} className="flex items-end font-extrabold text-3xl transition-colors text-[var(--nabu-ink)] hover:text-[var(--nabu-purple)] focus:outline-none focus:ring-2 focus:ring-[var(--nabu-purple)] rounded-md p-1">

NAB<div className="relative leading-none"><span className="text-4xl transition-colors">U</span><svg className="absolute top-5 left-[-3px] w-6 h-3 text-[var(--nabu-purple)] transition-colors" viewBox="0 0 24 10" fill="none" xmlns="http://www.w3.org/2000/svg"><path d="M1 6 C6 11, 18 11, 23 6" stroke="currentColor" strokeWidth="2" strokeLinecap="round"/></svg></div>

</button>

);

const Navbar = ({ navigate, currentPath }) => {

const navItems = [

{ path: '/network', label: 'Network', icon: Users },

{ path: '/profile/social', label: 'Profile', icon: User },

{ path: '/swipe', label: 'Opportunities', icon: Heart },

{ path: '/saved', label: 'Studio', icon: Save },

];

const showBackButton = currentPath !== '/' && currentPath !== '#/' && currentPath !== '/network';

const getIsActive = (path) => {

if (path === '/network' && (currentPath === '/' || currentPath === '#/' || currentPath === '/network')) return true;

if (path === '/swipe' && currentPath === '/swipe') return true;

if (path === '/profile/social' && (currentPath.startsWith('#/profile') || currentPath.startsWith('/profile'))) return true;

if (path === '/saved' && (currentPath === '/saved' || currentPath.startsWith('#/editor'))) return true;

return false;

};

return (

<nav className="fixed top-0 left-0 right-0 z-50 bg-[var(--nabu-black)] border-b border-[var(--nabu-border)] shadow-lg">

<div className="max-w-7xl mx-auto px-4 sm:px-6 lg:px-8 h-16 flex items-center justify-between">

<div className="flex items-center space-x-4">

{showBackButton && <button onClick={() => window.history.back()} className="p-2 rounded-lg text-[var(--nabu-lilac)] hover:bg-[#1f1f2a] hover:text-[var(--nabu-purple)] focus:outline-none focus:ring-2 focus:ring-[var(--nabu-purple)]" aria-label="Go back"><ArrowLeft className="w-6 h-6" /></button>}

<NabuLogo navigate={navigate} />

</div>

<div className="flex space-x-1 sm:space-x-2">

{navItems.map(({ path, label, icon: Icon }) => {

const isActive = getIsActive(path);

return (

<button key={path} onClick={() => navigate(path)} className={`flex items-center space-x-1 p-2 rounded-lg transition-all text-sm font-medium ${isActive ? 'bg-[var(--nabu-purple)] text-[var(--nabu-ink)] shadow-md' : 'text-[var(--nabu-lilac)] hover:bg-[#1f1f2a] hover:text-[var(--nabu-purple)]'} focus:outline-none focus:ring-2 focus:ring-[var(--nabu-purple)]`} aria-current={isActive ? 'page' : undefined}>

<Icon className="w-5 h-5" /><span className="hidden sm:inline">{label}</span>

</button>

);

})}

</div>

</div>

</nav>

);

};

const OpportunityCard = ({ opportunity, onClick, isSaved = false, actions }) => {

const { title, org, summary, deadline, logoUrl } = opportunity;

const cardContent = (

<div className="p-5 flex flex-col h-full">

{logoUrl && <div className="mb-4 rounded-lg overflow-hidden border border-[var(--nabu-border)]"><img src={logoUrl} alt={`${org} Logo`} className="w-full h-24 object-cover" onError={(e) => { e.target.onerror = null; e.target.src="https://placehold.co/300x150/1a1a1a/C8A2C8?text=Institution+Logo" }}/></div>}

<div className="flex items-center justify-between mb-2"><span className="px-3 py-1 text-xs font-semibold rounded-full bg-[var(--nabu-purple)] text-[var(--nabu-ink)]">{org}</span><span className="text-xs text-[var(--nabu-lilac)] font-mono"><span className="font-semibold text-[var(--nabu-ink)]">Deadline:</span> {formatDate(deadline)}</span></div>

<h3 className="text-xl font-bold text-[var(--nabu-ink)] mt-1 mb-3 font-serif">{title}</h3>

<p className="text-sm text-gray-400 flex-grow mb-4">{summary}</p>

{actions && <div className="flex justify-between space-x-4 mt-auto pt-3">{actions}</div>}

</div>

);

if (onClick) return <div onClick={onClick} onKeyDown={(e) => { if (e.key === 'Enter' || e.key === ' ') onClick(); }} role="button" tabIndex="0" aria-label={`Open editor for ${title}`} className={`w-full text-left rounded-xl transition-all duration-300 cursor-pointer bg-[var(--nabu-card)] border border-[var(--nabu-border)] hover:border-[var(--nabu-lilac)] shadow-md hover:shadow-lg focus:outline-none focus:ring-4 focus:ring-[var(--nabu-purple)] ${isSaved ? 'h-full' : 'min-h-[200px]'}`}>{cardContent}</div>;

return <div className="w-full h-full rounded-xl bg-[var(--nabu-card)] border border-[var(--nabu-border)] shadow-xl">{cardContent}</div>;

};

const BigActionButton = ({ icon: Icon, label, onClick, colorClass, hotkey }) => (

<button onClick={onClick} className={`flex flex-col items-center justify-center w-full p-4 rounded-xl text-center font-bold text-lg transition-all duration-200 shadow-xl border border-transparent ${colorClass} focus:outline-none focus:ring-4 focus:ring-offset-2 focus:ring-offset-[var(--nabu-black)]`} aria-label={label} data-hotkey={hotkey}>

<Icon className="w-8 h-8 sm:w-10 sm:h-10 mb-1" /><span className="text-sm sm:text-base">{label}</span>{hotkey && <span className="text-xs opacity-70 mt-1 hidden sm:inline-block">({hotkey})</span>}

</button>

);

const Toast = ({ message, type, removeToast }) => {

useEffect(() => { const timer = setTimeout(removeToast, 3000); return () => clearTimeout(timer); }, [removeToast]);

const colors = { success: 'bg-green-700 border-green-500', error: 'bg-red-700 border-red-500', info: 'bg-blue-700 border-blue-500' };

return <div className={`fixed bottom-4 right-4 p-4 rounded-xl shadow-2xl z-[9999] text-[var(--nabu-ink)] border-l-4 ${colors[type] || colors.info} animate-slideIn`}>{message}</div>;

};

const EmptyState = ({ title, message, ctaText, ctaAction }) => (

<div className="flex flex-col items-center justify-center p-10 text-center rounded-xl border border-dashed border-[var(--nabu-lilac)] bg-[#111111] max-w-lg mx-auto mt-12">

<ArrowRightCircle className="w-12 h-12 text-[var(--nabu-purple)] mb-4" /><h3 className="text-xl font-semibold text-[var(--nabu-ink)] mb-2">{title}</h3><p className="text-gray-400 mb-6">{message}</p>

{ctaAction && ctaText && <button onClick={ctaAction} className="px-6 py-2 text-sm font-medium rounded-lg bg-[var(--nabu-purple)] text-[var(--nabu-ink)] hover:bg-opacity-90 transition-opacity focus:outline-none focus:ring-2 focus:ring-[var(--nabu-purple)]">{ctaText}</button>}

</div>

);

const ShareBanner = ({ link, mode, onClose }) => {

const handleCopy = () => { navigator.clipboard.writeText(link).then(() => { alert('Collaboration link copied!'); }); };

return (

<div className="p-4 rounded-xl bg-[var(--nabu-purple)] bg-opacity-10 border border-[var(--nabu-purple)] text-[var(--nabu-ink)] mb-6">

<div className="flex justify-between items-center">

<Share2 className="w-5 h-5 text-[var(--nabu-purple)] mr-3 flex-shrink-0" /><div className="flex-grow"><p className="font-semibold text-sm mb-2">Share your draft: collaborate via comments or allow editing.</p><div className="flex items-center space-x-2"><input type="text" readOnly value={link} className="flex-grow p-2 text-xs rounded-lg bg-[#222] border border-[var(--nabu-purple)] text-[var(--nabu-ink)] font-mono truncate"/><button onClick={handleCopy} className="flex items-center px-3 py-2 text-xs font-medium rounded-lg bg-[var(--nabu-purple)] hover:bg-opacity-90 transition-opacity">Copy Link</button><button onClick={onClose} className="text-[var(--nabu-lilac)] hover:text-[var(--nabu-ink)] p-1 rounded-full"><X className="w-4 h-4" /></button></div><p className="mt-1 text-xs text-[var(--nabu-lilac)]">Mode: {mode === 'edit' ? 'Full Editing' : 'Comment Only'}</p></div>

</div>

</div>

);

};

// --- 3. GLOBAL STATE AND PERSISTENCE (CLIENT-SIDE) ---

const initialProfile = {

socialId: '', photoUrl: 'https://placehold.co/100x100/1a1a1a/C8A2C8?text=Photo',

name: '', description: '', cv: '', ambitions: [], stories: ['', '', '']

};

const initialOpportunities = SEED\_OPPORTUNITIES;

const App = () => {

const getRouteFromHash = () => window.location.hash.substring(1).split('?')[0] || '/';

const [route, setRoute] = useState(getRouteFromHash());

const navigate = (newPath) => {

const targetPath = newPath.startsWith('/') ? newPath : `/${newPath}`;

window.location.hash = targetPath;

setRoute(targetPath);

};

const [profile, setProfile] = useState(initialProfile);

const [opportunities, setOpportunities] = useState(initialOpportunities);

const [swipes, setSwipes] = useState({});

const [drafts, setDrafts] = useState({});

const [shareTokens, setShareTokens] = useState({});

const [posts, setPosts] = useState([]);

const [toast, setToast] = useState(null);

useEffect(() => {

const loadState = (key, defaultVal, setState) => {

const stored = localStorage.getItem(key);

if (stored) {

try { setState(JSON.parse(stored)); }

catch (e) { console.error(`Error loading state for ${key}`, e); }

} else if (key === STORAGE\_KEYS.OPPORTUNITIES) setState(SEED\_OPPORTUNITIES);

else if (key === STORAGE\_KEYS.POSTS) setState(SEED\_POSTS);

else if (key === STORAGE\_KEYS.PROFILE) {

const defaultDemoProfile = {

socialId: generateSocialId(),

photoUrl: 'https://placehold.co/100x100/1a1a1a/C8A2C8?text=User',

name: 'Alex Doe',

description: 'AI Ethics researcher, sustainable development advocate, and vintage synthesizer enthusiast.',

cv: 'Software Engineer with 5 years of experience in AI. PhD in Public Policy, specializing in ethical tech governance.',

ambitions: ['#Law', '#HumanRights', '#EnvironmentalSustainability', '#AI'],

stories: ["As a child, I saw how technology affected my community, which inspired me to study engineering to create ethical solutions.", "During my PhD, I faced a major research setback but learned the value of perseverance and collaboration to overcome it.", ""]

};

setState(defaultDemoProfile);

} else setState(defaultVal);

};

loadState(STORAGE\_KEYS.PROFILE, initialProfile, setProfile);

loadState(STORAGE\_KEYS.OPPORTUNITIES, initialOpportunities, setOpportunities);

loadState(STORAGE\_KEYS.SWIPES, {}, setSwipes);

loadState(STORAGE\_KEYS.DRAFTS, {}, setDrafts);

loadState(STORAGE\_KEYS.SHARES, {}, setShareTokens);

loadState(STORAGE\_KEYS.POSTS, [], setPosts);

const handleHashChange = () => {

const newRoute = getRouteFromHash();

if ((newRoute === '/' || newRoute === '#/') && !profile.name) navigate('/profile/social');

else setRoute(newRoute);

};

window.addEventListener('hashchange', handleHashChange);

if ((getRouteFromHash() === '/' || getRouteFromHash() === '#/') && !profile.name) navigate('/profile/social');

return () => window.removeEventListener('hashchange', handleHashChange);

}, []);

useEffect(() => { localStorage.setItem(STORAGE\_KEYS.PROFILE, JSON.stringify(profile)); }, [profile]);

useEffect(() => { localStorage.setItem(STORAGE\_KEYS.SWIPES, JSON.stringify(swipes)); }, [swipes]);

useEffect(() => { localStorage.setItem(STORAGE\_KEYS.DRAFTS, JSON.stringify(drafts)); }, [drafts]);

useEffect(() => { localStorage.setItem(STORAGE\_KEYS.OPPORTUNITIES, JSON.stringify(opportunities)); }, [opportunities]);

useEffect(() => { localStorage.setItem(STORAGE\_KEYS.SHARES, JSON.stringify(shareTokens)); }, [shareTokens]);

useEffect(() => { localStorage.setItem(STORAGE\_KEYS.POSTS, JSON.stringify(posts)); }, [posts]);

const showToast = useCallback((message, type = 'info') => { setToast({ message, type }); }, []);

const removeToast = useCallback(() => setToast(null), []);

const apiSaveProfile = (newProfile) => {

try { setProfile(newProfile); showToast('Profile Saved ✓', 'success'); }

catch (error) { console.error("Error saving profile:", error); showToast('Error saving profile.', 'error'); }

};

const apiPostSwipe = (opportunityId, action) => { setSwipes(prev => ({ ...prev, [opportunityId]: action })); };

const getSavedOpportunities = useMemo(() => {

const savedIds = Object.entries(swipes).filter(([, action]) => action === 'save').map(([id]) => parseInt(id));

return opportunities.filter(opp => savedIds.includes(opp.id)).sort((a, b) => new Date(a.deadline) - new Date(b.deadline));

}, [swipes, opportunities]);

const apiGenerateDraft = useCallback(async (opportunityId, prompt) => {

const opportunity = opportunities.find(o => o.id === opportunityId);

if (!opportunity) { showToast('Opportunity not found.', 'error'); return null; }

const systemPrompt = "Act as an expert scholarship essay writer... Your goal is to generate a compelling essay draft based on the user's profile... Maintain a formal yet personal tone.";

const userQuery = `Opportunity: ${opportunity.title} (${opportunity.org}). Summary: ${opportunity.summary}.\nCandidate Social Profile: Name: "${profile.name}", Description: "${profile.description}".\nCandidate Core Profile: CV: "${profile.cv}", Core Interests: "${Array.isArray(profile.ambitions) ? profile.ambitions.join(', ') : profile.ambitions}", Stories: "${Array.isArray(profile.stories) ? profile.stories.join('; ') : profile.stories}".\nEssay Question: "${prompt}"`;

try { return await callGeminiApi(userQuery, systemPrompt); }

catch (error) { console.error("Gemini API Error:", error); showToast('Gemini API Error. Using local generator.', 'error'); return mockEssayGenerator(profile, opportunity, prompt); }

}, [opportunities, profile, showToast]);

const apiCreateShare = (opportunityId, mode, draftText) => {

const token = generateUUID();

const shareLink = `${window.location.origin}${window.location.pathname}#/share/${token}?mode=${mode}`;

setShareTokens(prev => ({ ...prev, [token]: { opportunityId, mode, draftText, comments: [] } }));

return { url: shareLink, token };

};

const apiPostComment = (token, body) => {

const comment = { id: generateUUID(), body, createdAt: new Date().toISOString() };

setShareTokens(prev => {

const share = prev[token];

if (!share) return prev;

return { ...prev, [token]: { ...share, comments: [...share.comments, comment] } };

});

showToast('Comment added.', 'success');

};

const apiUpdateSharedDraft = (token, newDraftText) => {

setShareTokens(prev => {

const share = prev[token];

if (!share) return prev;

return { ...prev, [token]: { ...share, draftText: newDraftText } };

});

const opportunityId = shareTokens[token].opportunityId;

setDrafts(prev => ({ ...prev, [opportunityId]: newDraftText }));

showToast('Shared draft updated.', 'success');

};

const apiCreatePost = (content) => {

if (!content.trim()) return;

const newPost = { id: `post-${generateUUID()}`, authorId: profile.socialId, authorName: profile.name || 'User', authorPhotoUrl: profile.photoUrl, timestamp: new Date().toISOString(), content: content.trim(), imageUrl: null, likes: 0, comments: [] };

setPosts(prev => [newPost, ...prev]);

showToast('Post created!', 'success');

};

const apiLikePost = (postId) => { setPosts(prev => prev.map(post => post.id === postId ? { ...post, likes: post.likes + 1 } : post)); };

const apiAddCommentToPost = (postId, commentText) => {

if (!commentText.trim()) return;

const newComment = { id: `c-${generateUUID()}`, authorName: profile.name || 'User', text: commentText.trim() };

setPosts(prev => prev.map(post => post.id === postId ? { ...post, comments: [...post.comments, newComment] } : post));

};

const PageWrapper = ({ children }) => (

<div className="min-h-screen bg-[var(--nabu-black)] pt-16 text-[var(--nabu-ink)] font-inter">

<style>{`:root{--nabu-black:${COLORS.black};--nabu-purple:${COLORS.purple};--nabu-lilac:${COLORS.lilac};--nabu-ink:${COLORS.ink};--nabu-card:${COLORS.card};--nabu-border:${COLORS.border}}body{background-color:var(--nabu-black);color:var(--nabu-ink);font-family:'Inter',sans-serif}h1,h2,h3{font-family:'Playfair Display',serif}@keyframes slideIn{from{transform:translateX(100%);opacity:0}to{transform:translateX(0);opacity:1}}.animate-slideIn{animation:slideIn .3s ease-out forwards}@keyframes fadeIn{from{opacity:0;transform:translateY(-20px)}to{opacity:1;transform:translateY(0)}}.animate-fadeIn{animation:fadeIn .5s ease-out forwards}`}</style>

<main className="max-w-7xl mx-auto px-4 sm:px-6 lg:px-8 py-8">{children}</main>

{toast && <Toast {...toast} removeToast={removeToast} />}

</div>

);

const SocialProfilePage = () => {

const [localProfile, setLocalProfile] = useState(profile);

const [isEditing, setIsEditing] = useState(false);

useEffect(() => { if (!profile.name) setIsEditing(true); }, [profile.name]);

useEffect(() => { if (!localProfile.socialId) setLocalProfile(prev => ({ ...prev, socialId: generateSocialId() })); }, [localProfile.socialId]);

const handleChange = (e) => { setLocalProfile({ ...localProfile, [e.target.name]: e.target.value }); };

const handleSave = (e) => { e.preventDefault(); apiSaveProfile(localProfile); setIsEditing(false); };

const handleGoToCoreProfile = () => { apiSaveProfile(localProfile); setIsEditing(false); navigate('/profile/core'); };

const handleCopySocialId = () => { navigator.clipboard.writeText(localProfile.socialId).then(() => showToast('Social ID copied.', 'success')); };

return (

<div className="max-w-3xl mx-auto">

<div className="flex justify-between items-center mb-4">

<h2 className="text-3xl font-bold text-[var(--nabu-purple)]">Social NABU Profile</h2>

<div className="flex items-center space-x-3">

{isEditing ? <button onClick={handleSave} className="py-2 px-4 text-sm font-semibold rounded-lg bg-[var(--nabu-purple)] text-[var(--nabu-ink)] hover:bg-opacity-90">Save</button>

: <button onClick={() => setIsEditing(true)} className="py-2 px-4 text-sm font-semibold rounded-lg bg-[#222] border border-[#333] text-[var(--nabu-lilac)] hover:bg-[#333] flex items-center space-x-2"><Edit className="w-4 h-4" /><span>Edit</span></button>}

</div>

</div>

<p className="text-md text-gray-400 mb-8">This profile defines your personal brand for the <span className="font-bold text-[var(--nabu-ink)]">NABU network</span>.</p>

<form className="space-y-6">

<div className="flex flex-col sm:flex-row space-y-6 sm:space-y-0 sm:space-x-8 items-center">

<div className="flex flex-col items-center flex-shrink-0">

<img src={localProfile.photoUrl} alt="Profile" className="w-24 h-24 rounded-full object-cover border-4 border-[var(--nabu-lilac)]" onError={(e) => { e.target.onerror = null; e.target.src="https://placehold.co/100x100/1a1a1a/C8A2C8?text=User" }}/>

<div className="mt-4 text-center">

<label className="block text-sm font-medium text-[var(--nabu-lilac)] mb-1">Your NABU Social ID:</label>

<div className="flex items-center justify-center space-x-2">

<span className="font-mono text-lg text-[var(--nabu-ink)]">{localProfile.socialId}</span>

<button type="button" onClick={handleCopySocialId} className="text-[var(--nabu-purple)] hover:text-[var(--nabu-lilac)]" aria-label="Copy Social ID"><Share2 className="w-4 h-4" /></button>

</div>

</div>

</div>

<div className="flex-grow w-full space-y-4">

{isEditing ? (

<>

<input name="name" type="text" value={localProfile.name} onChange={handleChange} className="w-full p-3 rounded-xl bg-[#1a1a1a] border border-[#2a2a2a] focus:ring-2 focus:ring-[var(--nabu-purple)] text-lg font-bold" placeholder="Name / Nickname"/>

<textarea name="description" rows="3" value={localProfile.description} onChange={handleChange} className="w-full p-3 rounded-xl bg-[#1a1a1a] border border-[#2a2a2a] focus:ring-2 focus:ring-[var(--nabu-purple)] text-sm" placeholder="Description"></textarea>

</>

) : (

<div className="space-y-2">

<h3 className="text-2xl font-bold text-[var(--nabu-ink)] min-h-[3rem] flex items-center">{localProfile.name || <span className="text-gray-500">Name / Nickname</span>}</h3>

<p className="text-sm text-gray-400 min-h-[4rem]">{localProfile.description || <span className="text-gray-500">Description</span>}</p>

</div>

)}

</div>

</div>

<button type="button" onClick={handleGoToCoreProfile} className="w-full py-3 text-lg font-semibold rounded-xl bg-[var(--nabu-purple)] text-[var(--nabu-ink)] hover:bg-opacity-90 focus:outline-none focus:ring-4 focus:ring-[var(--nabu-purple)] flex items-center justify-center space-x-2">

<span>Go to Core Profile</span> <ArrowRightCircle className="w-5 h-5" />

</button>

</form>

</div>

);

};

const CoreProfilePage = () => {

const [localProfile, setLocalProfile] = useState(profile);

const [isEditing, setIsEditing] = useState(false);

const [tagInput, setTagInput] = useState('');

useEffect(() => { setLocalProfile(profile); if(!profile.cv) setIsEditing(true); }, [profile]);

const handleSave = () => { apiSaveProfile(localProfile); setIsEditing(false); };

const handleEdit = () => setIsEditing(true);

const handleStoryChange = (index, value) => {

const newStories = [...localProfile.stories];

newStories[index] = value;

setLocalProfile(p => ({ ...p, stories: newStories }));

};

const addStory = () => setLocalProfile(p => ({ ...p, stories: [...p.stories, ''] }));

const removeStory = (index) => setLocalProfile(p => ({ ...p, stories: p.stories.filter((\_, i) => i !== index) }));

const handleTagKeyDown = (e) => {

if (e.key === 'Enter' && tagInput.trim()) {

e.preventDefault();

const newTag = tagInput.trim().startsWith('#') ? tagInput.trim() : `#${tagInput.trim()}`;

if (!localProfile.ambitions.includes(newTag)) {

setLocalProfile(p => ({...p, ambitions: [...p.ambitions, newTag]}));

}

setTagInput('');

}

};

const removeTag = (tagToRemove) => setLocalProfile(p => ({...p, ambitions: p.ambitions.filter(tag => tag !== tagToRemove)}));

const handleDownloadCV = () => {

const cvContent = localProfile.cv || 'No CV content provided.';

const element = document.createElement("a");

const file = new Blob([cvContent], { type: 'text/plain' });

element.href = URL.createObjectURL(file);

element.download = `nabu-polished-cv.txt`;

document.body.appendChild(element);

element.click();

document.body.removeChild(element);

showToast('Polished CV downloaded.', 'success');

};

return (

<div className="max-w-3xl mx-auto">

<div className="flex justify-between items-center mb-4">

<h2 className="text-3xl font-bold text-[var(--nabu-purple)]">Core Application Profile</h2>

<div className="flex items-center space-x-3">

{isEditing ? <button onClick={handleSave} className="py-2 px-4 text-sm font-semibold rounded-lg bg-[var(--nabu-purple)] text-[var(--nabu-ink)] hover:bg-opacity-90">Save</button>

: <button onClick={handleEdit} className="py-2 px-4 text-sm font-semibold rounded-lg bg-[#222] border border-[#333] text-[var(--nabu-lilac)] hover:bg-[#333] flex items-center space-x-2"><Edit className="w-4 h-4" /><span>Edit</span></button>}

</div>

</div>

<p className="text-md text-gray-400 mb-8">This data is used by NABU's AI to match you with the best opportunities and generate tailored essays for your applications.</p>

<div className="space-y-8">

<div>

<label className="block text-sm font-medium text-[var(--nabu-lilac)] mb-2">Core Interests</label>

<div className={`p-4 rounded-xl bg-[#1a1a1a] border border-[#2a2a2a] min-h-[8rem] ${!isEditing ? 'cursor-text' : ''}`} onClick={isEditing ? undefined : handleEdit}>

<div className="flex flex-wrap gap-2">

{localProfile.ambitions.map((tag, i) => (

<div key={i} className="flex items-center bg-[var(--nabu-purple)] text-sm text-[var(--nabu-ink)] px-3 py-1 rounded-full">

<span>{tag}</span>

{isEditing && <button onClick={() => removeTag(tag)} className="ml-2 text-xs">&times;</button>}

</div>

))}

{isEditing && <input type="text" value={tagInput} onChange={e => setTagInput(e.target.value)} onKeyDown={handleTagKeyDown} placeholder="Add a tag and press Enter..." className="bg-transparent flex-grow focus:outline-none text-sm p-1"/>}

</div>

{!isEditing && localProfile.ambitions.length === 0 && <p className="text-gray-500 text-sm">#Law, #HumanRights, #AI...</p>}

</div>

</div>

<div>

<label htmlFor="cv" className="block text-sm font-medium text-[var(--nabu-lilac)] mb-2">CV / Background</label>

{isEditing ? <textarea id="cv" name="cv" rows="8" value={localProfile.cv} onChange={e => setLocalProfile({...localProfile, cv: e.target.value})} className="w-full p-3 rounded-xl bg-[#1a1a1a] border border-[#2a2a2a] focus:ring-2 focus:ring-[var(--nabu-purple)]" placeholder="List your achievements, education, and relevant experience."></textarea>

: <div className="p-4 rounded-xl bg-[#1a1a1a] border border-[#2a2a2a] min-h-[10rem] whitespace-pre-wrap text-sm text-gray-300 cursor-text" onClick={handleEdit}>{localProfile.cv || <p className="text-gray-500">Your professional and academic background...</p>}</div>}

<button type="button" onClick={handleDownloadCV} className="mt-3 px-4 py-2 text-sm font-medium rounded-lg bg-[#222] border border-[#333] text-[var(--nabu-lilac)] hover:bg-[#333] transition-colors flex items-center space-x-2">

<Download className="w-4 h-4" /> <span>Download Polished CV</span>

</button>

</div>

<div>

<label className="block text-sm font-medium text-[var(--nabu-lilac)] mb-2">Stories <span className="text-gray-400 font-normal">(memories or events from your life that help us understand who you are and your motivations)</span></label>

<div className="space-y-4">

{localProfile.stories.map((story, index) => (

<div key={index} className="relative">

{isEditing ? <textarea value={story} onChange={e => handleStoryChange(index, e.target.value)} rows="4" className="w-full p-3 rounded-xl bg-[#1a1a1a] border border-[#2a2a2a] focus:ring-2 focus:ring-[var(--nabu-purple)] resize-none" placeholder={`Story ${index + 1}`}></textarea>

: <div className="p-4 rounded-xl bg-[#1a1a1a] border border-[#2a2a2a] min-h-[6rem] text-sm text-gray-300 cursor-text" onClick={handleEdit}>{story || <p className="text-gray-500">{`Story ${index + 1}...`}</p>}</div>}

{isEditing && localProfile.stories.length > 1 && <button onClick={() => removeStory(index)} className="absolute top-2 right-2 text-gray-500 hover:text-red-500"><X className="w-4 h-4" /></button>}

</div>

))}

{isEditing && <button onClick={addStory} className="w-full flex items-center justify-center space-x-2 py-2 text-sm rounded-lg bg-[#222] border border-dashed border-[#333] text-[var(--nabu-lilac)] hover:bg-[#333]"><PlusCircle className="w-4 h-4" /><span>Add Story</span></button>}

</div>

</div>

</div>

</div>

);

};

const SwipePage = () => {

const availableOpportunities = useMemo(() => opportunities.filter(opp => !swipes.hasOwnProperty(opp.id)), [opportunities, swipes]);

const [currentIndex, setCurrentIndex] = useState(0);

const currentOpp = availableOpportunities[currentIndex];

const handleSwipe = useCallback((action) => {

if (!currentOpp) return; apiPostSwipe(currentOpp.id, action);

const card = document.getElementById('swipe-card');

if (card) {

const isSave = action === 'save', translateX = isSave ? '200%' : '-200%', rotateDeg = isSave ? '15deg' : '-15deg';

card.style.transition = 'transform 0.4s ease-out, opacity 0.4s ease-out';

card.style.transform = `translateX(${translateX}) rotate(${rotateDeg})`; card.style.opacity = '0';

setTimeout(() => { setCurrentIndex(p => p + 1); if (card) { card.style.transition = 'none'; card.style.transform = 'none'; card.style.opacity = '1'; } }, 400);

} else setCurrentIndex(p => p + 1);

}, [currentOpp]);

useEffect(() => {

const handleKeyDown = (e) => {

const isInputActive = /INPUT|TEXTAREA/.test(document.activeElement.tagName);

if (isInputActive) return;

if (e.key === 'ArrowLeft') handleSwipe('skip');

else if (e.key === 'ArrowRight') handleSwipe('save');

};

window.addEventListener('keydown', handleKeyDown);

return () => window.removeEventListener('keydown', handleKeyDown);

}, [handleSwipe]);

if (profile.cv.length < 50) return <EmptyState title="Complete Your Core Profile" message="Add your CV, ambitions, and stories to get personalized opportunity suggestions." ctaText="Go to Core Profile" ctaAction={() => navigate('/profile/core')} />;

if (!currentOpp) return <EmptyState title="All Opportunities Reviewed!" message="You've seen all available opportunities for now. Check back later or review your selections in the Studio." ctaText="Go to Studio" ctaAction={() => navigate('/saved')} />;

return (

<div className="flex flex-col items-center max-w-lg mx-auto">

<h2 className="text-3xl font-bold mb-4 text-center">Explore Opportunities</h2>

<div id="swipe-card" className="w-full h-[400px] mb-8 transition-all duration-300 transform-gpu will-change-transform"><OpportunityCard opportunity={currentOpp} /></div>

<div className="flex w-full space-x-6">

<BigActionButton icon={X} label="Skip" onClick={() => handleSwipe('skip')} colorClass="bg-red-900/50 text-red-400 hover:bg-red-800/70 focus:ring-red-500" hotkey="←" />

<BigActionButton icon={Save} label="Save" onClick={() => handleSwipe('save')} colorClass="bg-[var(--nabu-purple)] text-[var(--nabu-ink)] hover:bg-opacity-90 focus:ring-[var(--nabu-purple)]" hotkey="→" />

</div>

</div>

);

};

const SavedPage = () => {

if (getSavedOpportunities.length === 0) return <EmptyState title="Your Studio is Empty" message="Save opportunities you're interested in, and they will appear here. This is your workspace to build winning applications." ctaText="Find Opportunities" ctaAction={() => navigate('/swipe')} />;

return (

<div>

<h2 className="text-3xl font-bold mb-6">My Studio ({getSavedOpportunities.length})</h2>

<div className="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-3 gap-6">

{getSavedOpportunities.map(opp => <OpportunityCard key={opp.id} opportunity={opp} isSaved={true} onClick={() => navigate(`/editor/${opp.id}`)} actions={<button className="w-full px-4 py-2 text-sm font-semibold rounded-lg bg-[var(--nabu-lilac)] text-[var(--nabu-black)] hover:bg-opacity-90 flex items-center justify-center space-x-2"><FileText className="w-4 h-4" /> <span>Open in Editor</span></button>}/>)}

</div>

</div>

);

};

const EditorPage = () => {

const route = getRouteFromHash();

const match = route.match(/\/editor\/(\d+)/);

const opportunityId = match ? parseInt(match[1], 10) : null;

const opportunity = useMemo(() => opportunities.find(o => o.id === opportunityId), [opportunityId, opportunities]);

const availablePrompts = useMemo(() => (opportunity && PROMPT\_GROUPS[opportunity.promptGroup]) || PROMPT\_GROUPS.GLOBAL, [opportunity]);

const [draftText, setDraftText] = useState(drafts[opportunityId] || '');

const [selectedPrompt, setSelectedPrompt] = useState(availablePrompts[0]?.value || '');

const [isGenerating, setIsGenerating] = useState(false);

const [shareLink, setShareLink] = useState(null);

const [shareMode, setShareMode] = useState('comment');

const [showOriginalityPrompt, setShowOriginalityPrompt] = useState(false);

useEffect(() => { setDraftText(drafts[opportunityId] || ''); }, [opportunityId, drafts]);

useEffect(() => { if (availablePrompts.length > 0 && !selectedPrompt) setSelectedPrompt(availablePrompts[0].value); }, [availablePrompts, selectedPrompt]);

const saveDraft = useCallback(debounce((newText) => { setDrafts(p => ({ ...p, [opportunityId]: newText })); showToast('Draft saved locally.', 'info'); }, 1000), [opportunityId, showToast]);

const handleTextChange = (e) => { const newText = e.target.value; setDraftText(newText); saveDraft(newText); };

const handleGenerateClick = () => { if (opportunity && selectedPrompt) setShowOriginalityPrompt(true); };

const executeGeneration = async () => {

setShowOriginalityPrompt(false); setIsGenerating(true); setShareLink(null);

const promptText = availablePrompts.find(p => p.value === selectedPrompt)?.label || selectedPrompt;

try {

const essay = await apiGenerateDraft(opportunityId, promptText);

setDraftText(essay || ''); setDrafts(p => ({ ...p, [opportunityId]: essay }));

showToast('Draft generated ✓', 'success');

} catch (error) { console.error("Generation error:", error); showToast('Error generating draft.', 'error');

} finally { setIsGenerating(false); }

};

const handleDownload = () => {

const blob = new Blob([draftText], { type: 'text/plain' });

const url = URL.createObjectURL(blob);

const a = document.createElement('a');

a.href = url; a.download = `nabu-application-${opportunityId}.txt`; a.click();

URL.revokeObjectURL(url); showToast('File downloaded.', 'success');

};

const handleShare = (mode) => { const { url } = apiCreateShare(opportunityId, mode, draftText); setShareMode(mode); setShareLink(url); showToast('Collaboration link created.', 'success'); };

if (!opportunity) return <div><h2 className="text-3xl font-bold mb-6">Opportunity not found.</h2></div>;

const isControlsBlocked = isGenerating || showOriginalityPrompt;

return (

<div>

{showOriginalityPrompt && <OriginalityLockComponent onClose={executeGeneration} />}

<h2 className="text-3xl font-bold mb-2">{opportunity.title}</h2>

<p className="text-sm text-[var(--nabu-lilac)] mb-6">For: {opportunity.org} | Deadline: {formatDate(opportunity.deadline)}</p>

{shareLink && <ShareBanner link={shareLink} mode={shareMode} onClose={() => setShareLink(null)} />}

<div className={`bg-[var(--nabu-card)] p-4 rounded-xl border border-[var(--nabu-border)] mb-6 flex flex-col md:flex-row space-y-3 md:space-y-0 md:space-x-4 items-center ${isControlsBlocked ? 'opacity-50 pointer-events-none' : ''}`}>

<List className="w-5 h-5 text-[var(--nabu-lilac)] flex-shrink-0 hidden md:block" />

<select value={selectedPrompt} onChange={(e) => setSelectedPrompt(e.target.value)} className="flex-grow p-3 rounded-lg bg-[#222] border border-[#333] focus:ring-2 focus:ring-[var(--nabu-purple)] text-sm w-full md:w-auto" aria-label="Select Essay Prompt">{availablePrompts.map(p => <option key={p.value} value={p.value}>{p.label}</option>)}</select>

<div className="flex space-x-3 w-full md:w-auto"><button onClick={handleGenerateClick} disabled={isControlsBlocked} className={`px-4 py-2 text-sm font-semibold rounded-lg text-[var(--nabu-black)] transition-opacity w-full md:w-auto flex items-center justify-center space-x-2 ${isGenerating ? 'bg-[var(--nabu-lilac)] opacity-70 cursor-not-allowed' : 'bg-[var(--nabu-lilac)] hover:opacity-90'}`}>{isGenerating ? <Loader2 className="w-4 h-4 animate-spin" /> : <FileText className="w-4 h-4" />}<span>Generate Essay</span></button></div>

</div>

<>

<div className="flex justify-end space-x-3 mb-4"><button onClick={handleDownload} className="p-2 text-sm font-medium rounded-lg bg-[#222] border border-[#333] text-[var(--nabu-lilac)] hover:bg-[#333] flex items-center space-x-2" aria-label="Download"><Download className="w-4 h-4" /> <span>Download .txt</span></button><button onClick={() => handleShare('comment')} className="p-2 text-sm font-medium rounded-lg bg-[#222] border border-[#333] text-[var(--nabu-purple)] hover:bg-[#333] flex items-center space-x-2" aria-label="Share for Comments"><MessageSquare className="w-4 h-4" /> <span>Share (Comment)</span></button><button onClick={() => handleShare('edit')} className="p-2 text-sm font-medium rounded-lg bg-[#222] border border-[#333] text-[var(--nabu-purple)] hover:bg-[#333] flex items-center space-x-2" aria-label="Share for Editing"><Edit className="w-4 h-4" /> <span>Share (Edit)</span></button></div>

<div className="relative"><textarea value={draftText} onChange={handleTextChange} rows="25" readOnly={isControlsBlocked} className="w-full p-4 rounded-xl bg-[#1a1a1a] border border-[var(--nabu-purple)] focus:ring-2 focus:ring-[var(--nabu-purple)] text-sm font-mono resize-none" placeholder="The generated draft will appear here..." aria-label="Essay draft editor" /><div className="absolute bottom-3 right-3 text-xs text-[var(--nabu-lilac)] bg-black/50 p-1 rounded-md">This is a suggestion. Make it yours.</div></div>

</>

</div>

);

};

const SharePage = () => {

const route = getRouteFromHash();

const match = route.match(/\/share\/([a-zA-Z0-9-]+)/);

const token = match ? match[1] : null;

const shareData = shareTokens[token];

const [commentBody, setCommentBody] = useState('');

const [localDraftText, setLocalDraftText] = useState(shareData?.draftText || '');

const [isSaving, setIsSaving] = useState(false);

useEffect(() => { if (shareData) setLocalDraftText(shareData.draftText); }, [shareData]);

const opportunity = shareData ? opportunities.find(o => o.id === shareData.opportunityId) : null;

if (!shareData || !opportunity) return <EmptyState title="Invalid or Expired Link" message="This collaboration link may be incorrect or has expired." />;

const mode = shareData.mode;

const handlePostComment = (e) => { e.preventDefault(); if (commentBody.trim()) { apiPostComment(token, commentBody.trim()); setCommentBody(''); } };

const handleDraftUpdate = useCallback(debounce((newText) => { if (mode === 'edit') { setIsSaving(true); apiUpdateSharedDraft(token, newText); setTimeout(() => setIsSaving(false), 500); } }, 1500), [token, mode]);

const handleEditChange = (e) => { const newText = e.target.value; setLocalDraftText(newText); handleDraftUpdate(newText); };

return (

<div className="max-w-4xl mx-auto">

<h2 className="text-3xl font-bold mb-2">Collaborative Draft</h2><p className="text-lg text-[var(--nabu-purple)] font-semibold mb-4">{opportunity.title}</p>

<div className="p-4 rounded-xl bg-[#1a1a1a] border border-[var(--nabu-lilac)] mb-6 flex justify-between items-center"><p className="text-sm font-medium">Current Mode: <span className="font-bold uppercase text-[var(--nabu-ink)]">{mode}</span></p>{isSaving && mode === 'edit' && <p className="text-sm text-green-400 flex items-center"><Loader2 className="w-4 h-4 mr-2 animate-spin" /> Saving...</p>}</div>

<div className="grid grid-cols-1 lg:grid-cols-3 gap-6">

<div className="lg:col-span-2 relative">

<textarea value={localDraftText} onChange={handleEditChange} rows="20" readOnly={mode === 'comment'} className={`w-full p-4 rounded-xl bg-[#1a1a1a] border border-[var(--nabu-purple)] text-sm font-mono resize-none ${mode === 'edit' ? 'focus:ring-2 focus:ring-[var(--nabu-purple)]' : 'cursor-default opacity-80'}`} aria-label="Shared essay draft" />

{mode === 'comment' && <div className="absolute inset-0 flex items-center justify-center bg-black/70 rounded-xl"><MessageSquare className="w-6 h-6 mr-2 text-white" /><span className="text-lg font-bold text-white">READ-ONLY (Comments)</span></div>}

</div>

<div className="lg:col-span-1">

<h3 className="text-xl font-bold mb-3 flex items-center space-x-2"><MessageSquare className="w-5 h-5"/> <span>Comments</span></h3>

{mode === 'comment' && <form onSubmit={handlePostComment} className="mb-6 p-4 rounded-xl bg-[#1a1a1a] border border-[#2a2a2a]"><textarea value={commentBody} onChange={(e) => setCommentBody(e.target.value)} rows="3" className="w-full p-2 rounded-lg bg-[#222] border border-[#333] focus:ring-2 focus:ring-[var(--nabu-lilac)] text-sm resize-none" placeholder="Add a comment..."/><button type="submit" className="w-full mt-2 py-2 text-sm font-semibold rounded-lg bg-[var(--nabu-purple)] text-[var(--nabu-ink)] hover:bg-opacity-90">Post Comment</button></form>}

<div className="space-y-4 max-h-[400px] overflow-y-auto pr-2">{shareData.comments.length === 0 ? <p className="text-gray-500 italic">No comments yet.</p> : shareData.comments.map(c => <div key={c.id} className="p-3 bg-[#111] rounded-lg border-l-4 border-[var(--nabu-purple)]"><p className="text-xs text-gray-400 mb-1">{new Date(c.createdAt).toLocaleTimeString('en-US',{hour:'2-digit',minute:'2-digit'})}</p><p className="text-sm">{c.body}</p></div>)}</div>

</div>

</div>

</div>

);

};

const NetworkPage = () => {

const [newPostContent, setNewPostContent] = useState('');

const handleCreatePost = (e) => { e.preventDefault(); apiCreatePost(newPostContent); setNewPostContent(''); };

const PostCard = ({ post }) => {

const [commentText, setCommentText] = useState('');

const [showComments, setShowComments] = useState(false);

const handleCommentSubmit = (e) => { e.preventDefault(); apiAddCommentToPost(post.id, commentText); setCommentText(''); };

return (

<div className="bg-[var(--nabu-card)] border border-[var(--nabu-border)] rounded-2xl p-5 shadow-lg">

<div className="flex items-center mb-4"><img src={post.authorPhotoUrl} alt={post.authorName} className="w-12 h-12 rounded-full mr-4 border-2 border-[var(--nabu-purple)]" /><div><p className="font-bold text-[var(--nabu-ink)]">{post.authorName}</p><p className="text-xs text-gray-400">{timeAgo(post.timestamp)}</p></div></div>

<p className="mb-4 text-gray-300 whitespace-pre-wrap">{post.content}</p>

{post.imageUrl && <img src={post.imageUrl} alt="Post" className="rounded-lg mb-4 w-full object-cover" />}

{post.opportunityLink && <a href={post.opportunityLink} target="\_blank" rel="noopener noreferrer" className="block mb-4 p-3 bg-[#222] rounded-lg border border-[var(--nabu-purple)] hover:bg-[#333]"><p className="font-bold text-sm text-[var(--nabu-purple)]">Opportunity Link</p><p className="text-xs text-[var(--nabu-lilac)] truncate">{post.opportunityLink}</p></a>}

<div className="flex items-center justify-between text-gray-400 border-t border-[var(--nabu-border)] pt-3"><button onClick={() => apiLikePost(post.id)} className="flex items-center space-x-2 hover:text-[var(--nabu-purple)] transition-colors p-2 rounded-lg"><ThumbsUp className="w-5 h-5" /> <span>{post.likes} Likes</span></button><button onClick={() => setShowComments(!showComments)} className="flex items-center space-x-2 hover:text-[var(--nabu-purple)] transition-colors p-2 rounded-lg"><MessageSquare className="w-5 h-5" /> <span>{post.comments.length} Comments</span></button></div>

{showComments && <div className="mt-4 pt-4 border-t border-[var(--nabu-border)]">{post.comments.map(c => <div key={c.id} className="flex items-start space-x-3 mb-3"><p className="text-sm bg-[#222] p-2 rounded-lg"><strong className="text-[var(--nabu-lilac)]">{c.authorName}:</strong> {c.text}</p></div>)}<form onSubmit={handleCommentSubmit} className="flex items-center space-x-2 mt-4"><input type="text" value={commentText} onChange={e => setCommentText(e.target.value)} placeholder="Write a comment..." className="flex-grow p-2 text-sm rounded-lg bg-[#222] border border-[#333] focus:ring-1 focus:ring-[var(--nabu-purple)]" /><button type="submit" className="p-2 rounded-lg bg-[var(--nabu-purple)] text-[var(--nabu-ink)]"><Send className="w-4 h-4" /></button></form></div>}

</div>

);

};

return (

<div className="max-w-2xl mx-auto">

<h2 className="text-3xl font-bold mb-6 text-center">NABU Network</h2>

<div className="bg-[var(--nabu-card)] p-4 rounded-2xl border border-[var(--nabu-border)] mb-8 shadow-lg">

<form onSubmit={handleCreatePost} className="flex flex-col"><textarea value={newPostContent} onChange={e => setNewPostContent(e.target.value)} placeholder="Share your progress or an opportunity..." className="w-full p-3 rounded-xl bg-[#1a1a1a] border border-[#2a2a2a] focus:ring-2 focus:ring-[var(--nabu-purple)] mb-3 resize-none text-sm" rows="3" /><button type="submit" className="self-end px-6 py-2 text-sm font-semibold rounded-lg bg-[var(--nabu-purple)] text-[var(--nabu-ink)] hover:bg-opacity-90">Post</button></form>

</div>

<div className="space-y-6">{posts.map(post => <PostCard key={post.id} post={post} />)}</div>

</div>

);

};

const renderPage = () => {

const route = getRouteFromHash();

if (route === '/' || route === '#/' || route === '/profile') {

if (!profile.name) return <SocialProfilePage />;

return <NetworkPage />;

}

if (route === '/profile/social') return <SocialProfilePage />;

if (route === '/profile/core') return <CoreProfilePage />;

if (route === '/network') return <NetworkPage />;

if (route === '/swipe') return <SwipePage />;

if (route === '/saved') return <SavedPage />;

if (route.startsWith('/editor/')) return <EditorPage />;

if (route.startsWith('/share/')) return <SharePage />;

return <div><h2 className="text-3xl font-bold mb-6">404 - Page Not Found</h2></div>;

};

return (

<PageWrapper>

<Navbar navigate={navigate} currentPath={route} />

{renderPage()}

</PageWrapper>

);

};

export default App;