<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>DHK FOODIES</title>

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

<script src="https://unpkg.com/react@18/umd/react.production.min.js"></script>

<script src="https://unpkg.com/react-dom@18/umd/react-dom.production.min.js"></script>

<script src="https://unpkg.com/@babel/standalone/babel.min.js"></script>

<link rel="preconnect" href="https://fonts.googleapis.com">

<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

<link href="https://fonts.googleapis.com/css2?family=Inter:wght@400;500;600;700;800;900&display=swap" rel="stylesheet">

<style>

/\* Custom CSS for hero background animation \*/

@keyframes pulseBg {

0% { background-position: 0% 0%; }

50% { background-position: 100% 100%; }

100% { background-position: 0% 0%; }

}

.animate-pulse-bg {

animation: pulseBg 20s infinite alternate linear;

}

/\* Basic global styles for Inter font \*/

body {

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

margin: 0;

padding: 0;

}

/\* Modal specific styles \*/

.modal-overlay {

position: fixed;

top: 0;

left: 0;

right: 0;

bottom: 0;

background: rgba(0, 0, 0, 0.7);

display: flex;

justify-content: center;

align-items: center;

z-index: 1000;

}

.filter-sidebar-overlay {

position: fixed;

top: 0;

left: 0;

right: 0;

bottom: 0;

background: rgba(0, 0, 0, 0.6);

z-index: 1050;

transition: opacity 0.3s ease-in-out;

}

.filter-sidebar {

position: fixed;

top: 0;

left: 0;

bottom: 0;

width: 320px;

background: white;

box-shadow: 0 0 25px rgba(0,0,0,0.2);

transform: translateX(-100%);

transition: transform 0.3s ease-in-out;

z-index: 1100;

display: flex;

flex-direction: column;

}

.filter-sidebar.open {

transform: translateX(0);

}

.modal-content {

background: white;

padding: 2rem;

border-radius: 0.75rem; /\* rounded-xl \*/

max-width: 90%;

max-height: 90%;

overflow-y: auto;

position: relative;

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

transform: scale(0.95);

animation: modalPopIn 0.3s forwards cubic-bezier(0.25, 0.46, 0.45, 0.94);

}

@keyframes modalPopIn {

from { opacity: 0; transform: scale(0.8); }

to { opacity: 1; transform: scale(1); }

}

.modal-close-button {

position: absolute;

top: 1rem;

right: 1rem;

background: none;

border: none;

font-size: 1.5rem;

cursor: pointer;

color: #4a5568; /\* gray-700 \*/

}

.modal-close-button:hover {

color: #e53e3e; /\* red-600 \*/

}

/\* Splash screen specific styles \*/

.splash-screen {

position: fixed;

top: 0;

left: 0;

width: 100%;

height: 100%;

background: linear-gradient(to bottom right, #FF7E5F, #FEB47B); /\* Orange/Red gradient \*/

display: flex;

flex-direction: column;

justify-content: center;

align-items: center;

z-index: 2000; /\* Higher than modals \*/

color: white;

font-size: 2.5rem;

font-weight: bold;

transition: opacity 1s;

opacity: 1;

}

.splash-screen.hidden {

opacity: 0;

visibility: hidden;

}

.splash-screen-logo {

font-size: 4rem;

margin-bottom: 1rem;

animation: pulse 1.5s infinite alternate;

}

@keyframes pulse {

0% { transform: scale(1); }

100% { transform: scale(1.05); }

}

</style>

</head>

<body class="bg-gray-100">

<div id="root"></div>

<script type="text/babel">

const { useState, useEffect, useRef } = React;

// Helper function to create Lucide icons

const createIcon = (svgPath) => ({ size, ...props }) => (

<svg

xmlns="http://www.w3.org/2000/svg"

width={size || 24}

height={size || 24}

viewBox="0 0 24 24"

fill="none"

stroke="currentColor"

strokeWidth="2"

strokeLinecap="round"

strokeLinejoin="round"

{...props}

>

<path d={svgPath} />

</svg>

);

const Menu = createIcon("M4 12h16M4 6h16M4 18h16");

const Search = createIcon("M11 17.25a6.25 6.25 0 1 0 0-12.5 6.25 6.25 0 0 0 0 12.5ZM22 22l-4.1-4.1");

const ShoppingCart = createIcon("M6 2L3 6v14a2 2 0 0 0 2 2h14a2 2 0 0 0 2-2V6l-3-4H6Z M3 6h18 M16 10a2 2 0 0 1-4 0v-1a2 2 0 0 1 4 0v1Z");

const UserIcon = createIcon("M19 21v-2a4 4 0 0 0-4-4H9a4 4 0 0 0-4 4v2 M12 7a4 4 0 1 0 0-8 4 4 0 0 0 0 8Z");

const LogOut = createIcon("M9 21H5a2 2 0 0 1-2-2V5a2 2 0 0 1 2-2h4M17 17l5-5-5-5M22 12H9");

const CheckCircle = createIcon("M22 11.08V12a10 10 0 1 1-5.93-9.14 M22 4 12 14.01l-3-3");

const Truck = createIcon("M10 17H6a2 2 0 0 1-2-2V8a2 2 0 0 1 2-2h8a2 2 0 0 1 2 2v7a2 2 0 0 1-2 2h-1M12 18v2M12 10V6M6 18h1 M17 18h1 M20 12V8 M20 15h-2.5a2.5 2.5 0 0 0 0 5h2.5");

const Home = createIcon("M3 9l9-7 9 7v11a2 2 0 0 1-2 2H5a2 2 0 0 1-2-2z");

const Calendar = createIcon("M8 7V3M16 7V3M6 21h12a2 2 0 0 0 2-2V5a2 2 0 0 0-2-2H6a2 2 0 0 0-2 2v14a2 2 0 0 0 2 2Z M4 11h16");

const Package = createIcon("M16.5 8.5L22 12l-5.5 3.5 M2 12l5.5 3.5L13 12M7.5 8.5L13 12l5.5-3.5L13 5ZM12 22v-8");

const CreditCard = createIcon("M22 9V7a2 2 0 0 0-2-2H4a2 2 0 0 0-2 2v10a2 2 0 0 0 2 2h16a2 2 0 0 0 2-2v-2 M10 10h.01 M14 10h.01 M18 10h.01 M2 14h20");

const RupeeSign = createIcon("M6 3H7.5a2.5 2.5 0 0 1 2.5 2.5V11h4V5.5a2.5 2.5 0 0 1 2.5-2.5H18M6 7v14M18 7v14M6 13h12");

const Star = createIcon("M12 2l3.09 6.26L22 9.27l-5 4.87 1.18 6.88L12 18.25l-6.18 3.25L7 14.14l-5-4.87 8.91-1.01L12 2z");

const MapPin = createIcon("M12 2C8.13 2 5 5.13 5 9c0 5.25 7 13 7 13s7-7.75 7-13c0-3.87-3.13-7-7-7z M12 11.5a2.5 2.5 0 1 0 0-5 2.5 2.5 0 0 0 0 5z");

const Utensils = createIcon("M3 2v7c0 3.31 2.69 6 6 6h6c3.31 0 6-2.69 6-6V2M12 15v7M10 22h4");

const X = createIcon("M18 6 6 18M6 6l12 12");

const FilterIcon = createIcon("M22 3H2l8 9.46V19l4 2v-8.54L22 3z");

const Eye = createIcon("M2 12s3-7 10-7 10 7 10 7-3 7-10 7-10-7-10-7Z M12 15a3 3 0 1 0 0-6 3 3 0 0 0 0 6Z");

const EyeOff = createIcon("M9.9 4.24A9.94 9.94 0 0 1 12 2c7 0 10 7 10 7a13.13 13.13 0 0 1-1.67 2.68M6.61 6.61A3 3 0 0 0 12 15a3 3 0 0 0 2.39-.99M2 12s3 7 10 7a9.94 9.94 0 0 0 5.24-1.9M2 2l20 20");

const Phone = createIcon("M22 16.92v3a2 2 0 0 1-2.18 2 19.79 19.79 0 0 1-8.63-3.07 19.5 19.5 0 0 1-6-6 19.79 19.79 0 0 1-3.07-8.67A2 2 0 0 1 4.11 2h3a2 2 0 0 1 2 1.72 12.84 12.84 0 0 0 .7 2.81 2 2 0 0 1-.45 2.11L8.09 9.91a16 16 0 0 0 6 6l1.27-1.27a2 2 0 0 1 2.11-.45 12.84 12.84 0 0 0 2.81.7A2 2 0 0 1 22 16.92z");

const IdCard = createIcon("M12 22s8-4 8-10V5l-8-3-8 3v7c0 6 8 10 8 10z");

// RestaurantCard Component

const RestaurantCard = ({ restaurant, onViewMenu }) => (

<div className="bg-white rounded-xl shadow-lg overflow-hidden transform hover:scale-105 transition-transform duration-300 ease-in-out flex flex-col">

<img className="w-full h-48 object-cover" src={restaurant.image} alt={restaurant.name} onError={(e) => { e.target.onerror = null; e.target.src='https://placehold.co/600x400/CCCCCC/FFFFFF?text=Image+Error'; }} />

<div className="p-6 flex flex-col flex-grow">

<h3 className="font-bold text-xl mb-2 text-gray-800">{restaurant.name}</h3>

<p className="text-gray-600 text-sm mb-1 flex items-center"><Utensils size={16} className="mr-2 text-gray-500" /> {restaurant.cuisine}</p>

<p className="text-gray-600 text-sm mb-1 flex items-center"><MapPin size={16} className="mr-2 text-gray-500" /> {restaurant.address}</p>

<p className="text-gray-600 text-sm mb-4 flex items-center"><Star size={16} className="mr-2 text-yellow-500" fill="yellow" /> {restaurant.rating} / 5</p>

<button onClick={() => onViewMenu(restaurant)} className="mt-auto bg-orange-500 hover:bg-orange-600 text-white px-4 py-2 rounded-full text-sm font-semibold transition-colors duration-200">View Menu</button>

</div>

</div>

);

// FoodMenuItem Component (No Image)

const FoodMenuItem = ({ item, cartQuantity, onUpdateCartQuantity }) => {

const handleAdd = () => onUpdateCartQuantity(item.\_id, (cartQuantity || 0) + 1);

const handleRemove = () => onUpdateCartQuantity(item.\_id, (cartQuantity || 0) - 1);

return (

<div className="bg-white rounded-xl shadow-lg overflow-hidden transform hover:scale-105 transition-transform duration-300 ease-in-out flex flex-col p-6 h-full">

<div className="flex-grow">

<h3 className="font-bold text-xl mb-2 text-gray-800">{item.emoji} {item.name}</h3>

<p className="text-gray-600 text-base mb-4">{item.description}</p>

</div>

<div className="flex justify-between items-center mt-auto pt-4">

<span className="text-orange-500 font-bold text-lg">₹{item.price.toFixed(2)}</span>

{cartQuantity > 0 ? (

<div className="flex items-center space-x-2"><button onClick={handleRemove} className="bg-gray-200 text-gray-700 px-3 py-1 rounded-full hover:bg-gray-300 transition-colors duration-200" disabled={cartQuantity <= 0}>-</button><span className="font-medium text-lg text-gray-800">{cartQuantity}</span><button onClick={handleAdd} className="bg-gray-200 text-gray-700 px-3 py-1 rounded-full hover:bg-gray-300 transition-colors duration-200">+</button></div>

) : (

<button onClick={handleAdd} className="bg-orange-500 hover:bg-orange-600 text-white px-4 py-2 rounded-full text-sm font-semibold transition-colors duration-200">Add to Cart</button>

)}

</div>

</div>

);

};

// FilterSidebar Component

const FilterSidebar = ({ isOpen, onClose, onApplyFilters, availableCategories, initialFilters }) => {

const [selectedFilters, setSelectedFilters] = useState(initialFilters);

useEffect(() => {

if (isOpen) {

setSelectedFilters(initialFilters);

}

}, [initialFilters, isOpen]);

const handleCategoryChange = (category) => {

const newCategories = selectedFilters.categories.includes(category)

? selectedFilters.categories.filter(c => c !== category)

: [...selectedFilters.categories, category];

setSelectedFilters(prev => ({ ...prev, categories: newCategories }));

};

const handleRadioChange = (e) => {

const { name, value } = e.target;

setSelectedFilters(prev => ({ ...prev, [name]: value }));

};

const handleReset = () => {

const resetFilters = { diet: 'all', sortBy: 'relevance', categories: [] };

setSelectedFilters(resetFilters);

onApplyFilters(resetFilters);

onClose();

};

const handleApply = () => {

onApplyFilters(selectedFilters);

onClose();

};

return (

<>

{isOpen && <div className="filter-sidebar-overlay" onClick={onClose}></div>}

<div className={`filter-sidebar ${isOpen ? 'open' : ''}`}>

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

<div className="p-6 flex justify-between items-center border-b flex-shrink-0">

<h2 className="text-2xl font-bold text-gray-800 flex items-center"><X size={24} className="mr-3 cursor-pointer" onClick={onClose}/>FILTERS</h2>

<button onClick={handleReset} className="font-semibold text-orange-600 hover:text-orange-800">RESET</button>

</div>

<div className="flex-grow p-6 overflow-y-auto">

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

<div>

<h3 className="font-semibold text-gray-700 mb-3 text-sm uppercase tracking-wider">Sort By</h3>

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

<label className="flex items-center cursor-pointer"><input type="radio" name="sortBy" value="relevance" checked={selectedFilters.sortBy === 'relevance'} onChange={handleRadioChange} className="h-4 w-4 text-orange-600 focus:ring-orange-500 border-gray-300"/><span className="ml-2 text-gray-800">Relevance</span></label>

<label className="flex items-center cursor-pointer"><input type="radio" name="sortBy" value="price\_asc" checked={selectedFilters.sortBy === 'price\_asc'} onChange={handleRadioChange} className="h-4 w-4 text-orange-600 focus:ring-orange-500 border-gray-300"/><span className="ml-2 text-gray-800">Price: Low to High</span></label>

<label className="flex items-center cursor-pointer"><input type="radio" name="sortBy" value="price\_desc" checked={selectedFilters.sortBy === 'price\_desc'} onChange={handleRadioChange} className="h-4 w-4 text-orange-600 focus:ring-orange-500 border-gray-300"/><span className="ml-2 text-gray-800">Price: High to Low</span></label>

</div>

</div>

<hr/>

<div>

<h3 className="font-semibold text-gray-700 mb-3 text-sm uppercase tracking-wider">Dietary</h3>

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

<label className="flex items-center cursor-pointer"><input type="radio" name="diet" value="all" checked={selectedFilters.diet === 'all'} onChange={handleRadioChange} className="h-4 w-4 text-orange-600 focus:ring-orange-500 border-gray-300"/><span className="ml-2 text-gray-800">All</span></label>

<label className="flex items-center cursor-pointer"><input type="radio" name="diet" value="veg" checked={selectedFilters.diet === 'veg'} onChange={handleRadioChange} className="h-4 w-4 text-orange-600 focus:ring-orange-500 border-gray-300"/><span className="ml-2 text-gray-800">Veg Only</span></label>

<label className="flex items-center cursor-pointer"><input type="radio" name="diet" value="nonVeg" checked={selectedFilters.diet === 'nonVeg'} onChange={handleRadioChange} className="h-4 w-4 text-orange-600 focus:ring-orange-500 border-gray-300"/><span className="ml-2 text-gray-800">Non-Veg</span></label>

</div>

</div>

<hr/>

<div>

<h3 className="font-semibold text-gray-700 mb-3 text-sm uppercase tracking-wider">Categories</h3>

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

{availableCategories.map(category => (

<label key={category} className="flex items-center justify-between cursor-pointer">

<span className="text-gray-800 capitalize">{category}</span>

<input type="checkbox" checked={selectedFilters.categories.includes(category)} onChange={() => handleCategoryChange(category)} className="h-5 w-5 rounded text-orange-600 focus:ring-orange-500 border-gray-300"/>

</label>

))}

</div>

</div>

</div>

</div>

<div className="p-4 bg-white border-t flex-shrink-0">

<button onClick={handleApply} className="w-full bg-orange-500 hover:bg-orange-600 text-white font-bold py-3 px-4 rounded-full transition-colors">

APPLY FILTER

</button>

</div>

</div>

</div>

</>

);

};

// RestaurantMenuPage Component

const RestaurantMenuPage = ({ restaurant, cart, onUpdateCartQuantity, onBackToRestaurants }) => {

const categorizedMenu = restaurant.menu.reduce((acc, item) => {

const category = item.category || 'Other';

if (!acc[category]) acc[category] = [];

acc[category].push(item);

return acc;

}, {});

return (

<section className="container mx-auto py-8 md:py-12 px-4">

<button onClick={onBackToRestaurants} className="mb-6 flex items-center text-gray-700 hover:text-orange-500 transition-colors duration-200 font-medium"><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" className="mr-2"><path d="M15 18l-6-6 6-6"/></svg>Back to Restaurants</button>

<h2 className="text-4xl md:text-5xl font-bold text-gray-800 text-center mb-4">{restaurant.name}</h2>

<p className="text-lg text-gray-600 text-center mb-8">{restaurant.cuisine} - {restaurant.address}</p>

{Object.keys(categorizedMenu).map((category) => (

<div key={category} className="mb-10">

<h3 className="text-2xl md:text-3xl font-bold text-gray-800 mb-6 border-b-2 border-orange-400 pb-2">{category}</h3>

<div className="grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-3 xl:grid-cols-4 gap-6">

{categorizedMenu[category].map((item) => <FoodMenuItem key={item.\_id} item={item} cartQuantity={cart.find(cartItem => cartItem.\_id === item.\_id)?.quantity || 0} onUpdateCartQuantity={onUpdateCartQuantity} />)}

</div>

</div>

))}

</section>

);

};

// CartView Component

const CartView = ({ cartItems, onClose, onUpdateQuantity, onRemoveItem, onProceedToPayment }) => {

const calculateTotal = () => cartItems.reduce((total, item) => total + (item.price \* item.quantity), 0).toFixed(2);

return (

<div className="modal-overlay">

<div className="modal-content w-full md:w-3/4 lg:w-1/2">

<button onClick={onClose} className="modal-close-button"><X size={24} /></button>

<h2 className="text-3xl font-bold text-gray-800 mb-6 text-center">Your Cart</h2>

{cartItems.length === 0 ? <p className="text-gray-600 text-center text-lg">Your cart is empty.</p> : (

<div>

{cartItems.map((item) => (

<div key={item.\_id} className="flex flex-col sm:flex-row items-center justify-between border-b border-gray-200 py-4 last:border-b-0">

<div className="flex items-center space-x-4 mb-2 sm:mb-0 w-full sm:w-auto">

<span className="text-3xl">{item.emoji}</span>

<div><h4 className="font-semibold text-lg text-gray-800">{item.name}</h4><p className="text-gray-600">₹{item.price.toFixed(2)}</p></div>

</div>

<div className="flex items-center space-x-3 w-full sm:w-auto justify-center sm:justify-end">

<button onClick={() => onUpdateQuantity(item.\_id, item.quantity - 1)} disabled={item.quantity <= 1} className="bg-gray-200 text-gray-700 px-3 py-1 rounded-full hover:bg-gray-300 disabled:opacity-50 transition-colors duration-200">-</button>

<span className="font-medium text-lg text-gray-800">{item.quantity}</span>

<button onClick={() => onUpdateQuantity(item.\_id, item.quantity + 1)} className="bg-gray-200 text-gray-700 px-3 py-1 rounded-full hover:bg-gray-300 transition-colors duration-200">+</button>

<button onClick={() => onRemoveItem(item.\_id)} className="ml-4 text-red-500 hover:text-red-700 transition-colors duration-200">Remove</button>

</div>

</div>

))}

<div className="text-right mt-6 pt-4 border-t border-gray-200">

<p className="text-xl font-bold text-gray-800">Total: ₹{calculateTotal()}</p>

<button onClick={() => onProceedToPayment(calculateTotal())} className="mt-4 bg-orange-500 hover:bg-orange-600 text-white px-6 py-3 rounded-full text-lg font-semibold shadow-md transition-colors duration-200" disabled={cartItems.length === 0}>Proceed to Checkout</button>

</div>

</div>

)}

</div>

</div>

);

};

// AuthModal Component

const AuthModal = ({ onClose, onLoginSuccess }) => {

const [isLoginMode, setIsLoginMode] = useState(true);

const [username, setUsername] = useState('');

const [password, setPassword] = useState('');

const [confirmPassword, setConfirmPassword] = useState('');

const [phoneNumber, setPhoneNumber] = useState('');

const [message, setMessage] = useState('');

const [loading, setLoading] = useState(false);

const [showPassword, setShowPassword] = useState(false);

const [showConfirmPassword, setShowConfirmPassword] = useState(false);

const getUsers = () => JSON.parse(localStorage.getItem('mockUsers')) || [];

const saveUsers = (users) => localStorage.setItem('mockUsers', JSON.stringify(users));

const clearForm = () => {

setUsername('');

setPassword('');

setConfirmPassword('');

setPhoneNumber('');

setMessage('');

};

const handleAuth = async (e) => {

e.preventDefault();

setMessage('');

setLoading(true);

await new Promise(resolve => setTimeout(resolve, 1000));

const users = getUsers();

if (isLoginMode) {

if (!username || !password) {

setMessage('Username and password are required.');

setLoading(false);

return;

}

const userFound = users.find(user => user.username === username && user.password === password);

if (userFound) {

onLoginSuccess(username);

} else {

setMessage('Invalid username or password.');

}

} else {

if (!username || !password || !phoneNumber || !confirmPassword) {

setMessage('All fields are required for signup.');

setLoading(false);

return;

}

if (password !== confirmPassword) {

setMessage('Passwords do not match.');

setLoading(false);

return;

}

if (!/^\d{10}$/.test(phoneNumber)) {

setMessage('Please enter a valid 10-digit phone number.');

setLoading(false);

return;

}

if (users.some(user => user.username === username)) {

setMessage('Username already exists. Please choose another.');

} else {

const newUser = {

username,

password,

phoneNumber,

userId: 'DHK-' + Math.random().toString(36).substring(2, 9).toUpperCase(),

createdAt: new Date().toISOString()

};

saveUsers([...users, newUser]);

setMessage('Signup successful! You can now log in.');

onLoginSuccess(username);

}

}

setLoading(false);

};

return (

<div className="modal-overlay">

<div className="modal-content w-full md:w-1/2 lg:w-1/3">

<div className="flex justify-center mb-6 border-b border-gray-200">

<button onClick={() => { setIsLoginMode(true); clearForm(); }} className={`px-6 py-3 text-lg font-semibold rounded-t-lg transition-colors duration-200 ${isLoginMode ? 'bg-orange-500 text-white shadow-md' : 'bg-transparent text-gray-700 hover:text-orange-500'}`}>Login</button>

<button onClick={() => { setIsLoginMode(false); clearForm(); }} className={`px-6 py-3 text-lg font-semibold rounded-t-lg transition-colors duration-200 ${!isLoginMode ? 'bg-orange-500 text-white shadow-md' : 'bg-transparent text-gray-700 hover:text-orange-500'}`}>Sign Up</button>

</div>

<h2 className="text-3xl font-bold text-gray-800 mb-6 text-center">{isLoginMode ? 'Welcome Back!' : 'Join DHK FOODIES!'}</h2>

<form onSubmit={handleAuth} className="space-y-4">

<div><label className="block text-gray-700 text-sm font-bold mb-2" htmlFor="username">Username</label><input type="text" id="username" className="shadow appearance-none border rounded-lg w-full py-3 px-4 text-gray-700 leading-tight focus:outline-none focus:ring-2 focus:ring-orange-500" placeholder="Your username" value={username} onChange={(e) => setUsername(e.target.value)} required /></div>

{!isLoginMode && (<div><label className="block text-gray-700 text-sm font-bold mb-2" htmlFor="phoneNumber">Phone Number</label><input type="tel" id="phoneNumber" className="shadow appearance-none border rounded-lg w-full py-3 px-4 text-gray-700 leading-tight focus:outline-none focus:ring-2 focus:ring-orange-500" placeholder="10-digit mobile number" value={phoneNumber} onChange={(e) => setPhoneNumber(e.target.value)} required /></div>)}

<div className="relative">

<label className="block text-gray-700 text-sm font-bold mb-2" htmlFor="password">Password</label>

<input type={showPassword ? 'text' : 'password'} id="password" className="shadow appearance-none border rounded-lg w-full py-3 px-4 text-gray-700 mb-1 leading-tight focus:outline-none focus:ring-2 focus:ring-orange-500" placeholder="\*\*\*\*\*\*\*\*\*\*\*" value={password} onChange={(e) => setPassword(e.target.value)} required />

<button type="button" onClick={() => setShowPassword(!showPassword)} className="absolute inset-y-0 right-0 top-6 pr-3 flex items-center text-sm leading-5">

{showPassword ? <EyeOff size={20} className="text-gray-500"/> : <Eye size={20} className="text-gray-500"/>}

</button>

</div>

{!isLoginMode && (

<div className="relative">

<label className="block text-gray-700 text-sm font-bold mb-2" htmlFor="confirmPassword">Confirm Password</label>

<input type={showConfirmPassword ? 'text' : 'password'} id="confirmPassword" className="shadow appearance-none border rounded-lg w-full py-3 px-4 text-gray-700 mb-1 leading-tight focus:outline-none focus:ring-2 focus:ring-orange-500" placeholder="\*\*\*\*\*\*\*\*\*\*\*" value={confirmPassword} onChange={(e) => setConfirmPassword(e.target.value)} required />

<button type="button" onClick={() => setShowConfirmPassword(!showConfirmPassword)} className="absolute inset-y-0 right-0 top-6 pr-3 flex items-center text-sm leading-5">

{showConfirmPassword ? <EyeOff size={20} className="text-gray-500"/> : <Eye size={20} className="text-gray-500"/>}

</button>

</div>

)}

{message && <p className={`text-center ${message.includes('successful') ? 'text-green-600' : 'text-red-500'} text-sm font-medium`}>{message}</p>}

<button type="submit" className="bg-orange-500 hover:bg-orange-600 text-white font-bold py-3 px-4 rounded-full w-full focus:outline-none focus:shadow-outline transition-colors duration-200 flex items-center justify-center mt-4" disabled={loading}>{loading && <svg className="animate-spin -ml-1 mr-3 h-5 w-5 text-white" xmlns="http://www.w3.org/2000/svg" fill="none" viewBox="0 0 24 24"><circle className="opacity-25" cx="12" cy="12" r="10" stroke="currentColor" strokeWidth="4"></circle><path className="opacity-75" fill="currentColor" d="M4 12a8 8 0 018-8V0C5.373 0 0 5.373 0 12h4zm2 5.291A7.962 7.962 0 014 12H0c0 3.042 1.135 5.824 3 7.938l3-2.647z"></path></svg>}{loading ? 'Processing...' : (isLoginMode ? 'Login' : 'Sign Up')}</button>

</form>

</div>

</div>

);

};

// DeliveryAddressModal Component

const DeliveryAddressModal = ({ onClose, onSubmitAddress }) => {

const [fullName, setFullName] = useState('');

const [address, setAddress] = useState('');

const [city, setCity] = useState('');

const [state, setState] = useState('');

const [zipCode, setZipCode] = useState('');

const [error, setError] = useState('');

const handleSubmit = (e) => {

e.preventDefault();

if (!fullName || !address || !city || !state || !zipCode) {

setError('All fields are required.');

return;

}

setError('');

onSubmitAddress({ fullName, address, city, state, zipCode });

};

return (

<div className="modal-overlay">

<div className="modal-content w-full md:w-3/4 lg:w-1/2">

<button onClick={onClose} className="modal-close-button"><X size={24} /></button>

<h2 className="text-3xl font-bold text-gray-800 mb-6 text-center">Delivery Address</h2>

<form onSubmit={handleSubmit} className="space-y-4">

<div><label className="block text-gray-700 text-sm font-bold mb-2" htmlFor="fullName">Full Name</label><input type="text" id="fullName" className="shadow appearance-none border rounded-lg w-full py-3 px-4 text-gray-700 leading-tight focus:outline-none focus:ring-2 focus:ring-orange-500" value={fullName} onChange={(e) => setFullName(e.target.value)} required /></div>

<div><label className="block text-gray-700 text-sm font-bold mb-2" htmlFor="address">Address</label><input type="text" id="address" className="shadow appearance-none border rounded-lg w-full py-3 px-4 text-gray-700 leading-tight focus:outline-none focus:ring-2 focus:ring-orange-500" value={address} onChange={(e) => setAddress(e.target.value)} required /></div>

<div className="grid grid-cols-1 md:grid-cols-3 gap-4">

<div><label className="block text-gray-700 text-sm font-bold mb-2" htmlFor="city">City</label><input type="text" id="city" className="shadow appearance-none border rounded-lg w-full py-3 px-4 text-gray-700 leading-tight focus:outline-none focus:ring-2 focus:ring-orange-500" value={city} onChange={(e) => setCity(e.target.value)} required /></div>

<div><label className="block text-gray-700 text-sm font-bold mb-2" htmlFor="state">State</label><input type="text" id="state" className="shadow appearance-none border rounded-lg w-full py-3 px-4 text-gray-700 leading-tight focus:outline-none focus:ring-2 focus:ring-orange-500" value={state} onChange={(e) => setState(e.target.value)} required /></div>

<div><label className="block text-gray-700 text-sm font-bold mb-2" htmlFor="zipCode">Zip Code</label><input type="text" id="zipCode" className="shadow appearance-none border rounded-lg w-full py-3 px-4 text-gray-700 leading-tight focus:outline-none focus:ring-2 focus:ring-orange-500" value={zipCode} onChange={(e) => setZipCode(e.target.value)} required /></div>

</div>

{error && <p className="text-red-500 text-center text-sm mt-2">{error}</p>}

<button type="submit" className="mt-6 bg-orange-500 hover:bg-orange-600 text-white font-bold py-3 px-4 rounded-full w-full focus:outline-none focus:shadow-outline transition-colors duration-200">Proceed to Payment</button>

</form>

</div>

</div>

);

};

// PaymentModal Component

const PaymentModal = ({ onClose, totalAmount, onPaymentSuccess }) => {

const [selectedMethod, setSelectedMethod] = useState('card');

const [paymentProcessing, setPaymentProcessing] = useState(false);

const [paymentMessage, setPaymentMessage] = useState('');

const [cardDetails, setCardDetails] = useState({ number: '', expiry: '', cvv: '' });

const [upiNumber, setUpiNumber] = useState('');

const handlePayNow = async () => {

// Validation before proceeding

if (selectedMethod === 'card') {

if (!cardDetails.number || !cardDetails.expiry || !cardDetails.cvv) {

setPaymentMessage('Please fill all card details.');

return;

}

}

if (selectedMethod === 'upi') {

if (!upiNumber || !/^\d{10}$/.test(upiNumber)) {

setPaymentMessage('Please enter a valid 10-digit UPI number.');

return;

}

}

setPaymentProcessing(true);

setPaymentMessage('');

await new Promise(resolve => setTimeout(resolve, 2000));

setPaymentMessage('Payment successful! Redirecting to order confirmation...');

const orderId = 'DHK' + Date.now().toString().slice(-6) + Math.floor(Math.random() \* 100);

onPaymentSuccess({ orderId, total: totalAmount, paymentMethod: selectedMethod, timestamp: Date.now() });

};

const handleCardInputChange = (e) => {

const { name, value } = e.target;

setCardDetails(prev => ({ ...prev, [name]: value }));

};

return (

<div className="modal-overlay">

<div className="modal-content w-full md:w-2/3 lg:w-1/2">

<button onClick={onClose} className="modal-close-button"><X size={24} /></button>

<h2 className="text-3xl font-bold text-gray-800 mb-4 text-center">Complete Your Payment</h2>

<p className="text-center text-xl font-semibold text-orange-600 mb-8">Total Amount: ₹{totalAmount}</p>

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

{/\* Card Payment Option \*/}

<div className={`border rounded-lg p-4 transition-all ${selectedMethod === 'card' ? 'border-orange-500 ring-2 ring-orange-200' : 'border-gray-300'}`}>

<label className="flex items-center cursor-pointer">

<input type="radio" name="paymentMethod" value="card" checked={selectedMethod === 'card'} onChange={() => setSelectedMethod('card')} className="h-5 w-5 text-orange-600 focus:ring-orange-500" />

<span className="ml-3 font-semibold text-lg text-gray-800">Credit / Debit Card</span>

</label>

{selectedMethod === 'card' && (

<div className="mt-4 space-y-3 pl-8">

<input type="text" name="number" placeholder="Card Number" value={cardDetails.number} onChange={handleCardInputChange} className="shadow-sm appearance-none border rounded-lg w-full py-2 px-3 text-gray-700" />

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

<input type="text" name="expiry" placeholder="MM/YY" value={cardDetails.expiry} onChange={handleCardInputChange} className="shadow-sm appearance-none border rounded-lg w-full py-2 px-3 text-gray-700" />

<input type="text" name="cvv" placeholder="CVV" value={cardDetails.cvv} onChange={handleCardInputChange} className="shadow-sm appearance-none border rounded-lg w-full py-2 px-3 text-gray-700" />

</div>

</div>

)}

</div>

{/\* UPI Payment Option \*/}

<div className={`border rounded-lg p-4 transition-all ${selectedMethod === 'upi' ? 'border-orange-500 ring-2 ring-orange-200' : 'border-gray-300'}`}>

<label className="flex items-center cursor-pointer">

<input type="radio" name="paymentMethod" value="upi" checked={selectedMethod === 'upi'} onChange={() => setSelectedMethod('upi')} className="h-5 w-5 text-orange-600 focus:ring-orange-500" />

<span className="ml-3 font-semibold text-lg text-gray-800">UPI</span>

</label>

{selectedMethod === 'upi' && (

<div className="mt-4 pl-8">

<input type="tel" placeholder="Enter your 10-digit UPI Number" value={upiNumber} onChange={(e) => setUpiNumber(e.target.value)} maxLength="10" className="shadow-sm appearance-none border rounded-lg w-full py-2 px-3 text-gray-700" />

</div>

)}

</div>

{/\* COD Payment Option \*/}

<div className={`border rounded-lg p-4 transition-all ${selectedMethod === 'cod' ? 'border-orange-500 ring-2 ring-orange-200' : 'border-gray-300'}`}>

<label className="flex items-center cursor-pointer">

<input type="radio" name="paymentMethod" value="cod" checked={selectedMethod === 'cod'} onChange={() => setSelectedMethod('cod')} className="h-5 w-5 text-orange-600 focus:ring-orange-500" />

<span className="ml-3 font-semibold text-lg text-gray-800">Cash on Delivery</span>

</label>

{selectedMethod === 'cod' && (

<p className="mt-2 pl-8 text-gray-600">You can pay in cash at the time of delivery.</p>

)}

</div>

</div>

{paymentMessage && <p className={`text-center mt-4 font-medium ${paymentMessage.includes('successful') ? 'text-green-600' : 'text-red-500'}`}>{paymentMessage}</p>}

<button onClick={handlePayNow} className="mt-8 bg-orange-500 hover:bg-orange-600 text-white font-bold py-3 px-4 rounded-full w-full focus:outline-none focus:shadow-outline transition-colors duration-200 flex items-center justify-center" disabled={paymentProcessing}>

{paymentProcessing && <svg className="animate-spin -ml-1 mr-3 h-5 w-5 text-white" xmlns="http://www.w3.org/2000/svg" fill="none" viewBox="0 0 24 24"><circle className="opacity-25" cx="12" cy="12" r="10" stroke="currentColor" strokeWidth="4"></circle><path className="opacity-75" fill="currentColor" d="M4 12a8 8 0 018-8V0C5.373 0 0 5.373 0 12h4zm2 5.291A7.962 7.962 0 014 12H0c0 3.042 1.135 5.824 3 7.938l3-2.647z"></path></svg>}

{paymentProcessing ? 'Processing...' : `Pay ₹${totalAmount}`}

</button>

</div>

</div>

);

};

// OrderConfirmation Component

const OrderConfirmation = ({ orderDetails, onGoHome }) => {

const statusColors = {'Confirmed': 'text-blue-500', 'Preparing': 'text-yellow-500', 'Out for Delivery': 'text-purple-500', 'Delivered': 'text-green-500'};

return (

<section className="container mx-auto py-12 md:py-16 px-4 text-center min-h-[60vh] flex flex-col justify-center items-center">

<CheckCircle className="text-green-500 mb-6" size={80} />

<h2 className="text-4xl md:text-5xl font-bold text-gray-800 mb-4">Order Placed Successfully!</h2>

<p className="text-xl text-gray-700 mb-6">Thank you for your order, {orderDetails.deliveryAddress.fullName}!</p>

<div className="bg-white rounded-xl shadow-lg p-8 w-full max-w-md space-y-4 mb-8 text-left">

<h3 className="text-2xl font-bold text-gray-800 border-b pb-3 mb-3">Order Summary</h3>

<p className="text-lg text-gray-700 flex items-center"><Package size={20} className="mr-2" /> <strong>Order ID:</strong> {orderDetails.orderId}</p>

<p className="text-lg text-gray-700 flex items-center"><RupeeSign size={20} className="mr-2" /> <strong>Total Amount:</strong> ₹{orderDetails.total}</p>

<p className="text-lg text-gray-700 flex items-center"><CreditCard size={20} className="mr-2" /> <strong>Payment Method:</strong> {orderDetails.paymentMethod.toUpperCase()}</p>

{orderDetails.deliveryAddress && (

<div className="mt-4 pt-4 border-t border-gray-200">

<h4 className="text-xl font-bold text-gray-800 mb-2">Delivery Address:</h4>

<p className="text-gray-700">{orderDetails.deliveryAddress.fullName}</p>

<p className="text-gray-700">{orderDetails.deliveryAddress.address}</p>

<p className="text-gray-700">{orderDetails.deliveryAddress.city}, {orderDetails.deliveryAddress.state} - {orderDetails.deliveryAddress.zipCode}</p>

</div>

)}

<div className="flex items-center justify-center pt-4 border-t"><Truck className={`mr-2 ${statusColors[orderDetails.status]}`} size={32} /><p className="text-xl font-bold">Status: <span className={`${statusColors[orderDetails.status]}`}>{orderDetails.status}</span></p></div>

<p className="text-md text-gray-600 mt-2">Your order will be updated in your order history.</p>

</div>

<button onClick={onGoHome} className="bg-orange-500 hover:bg-orange-600 text-white px-8 py-3 rounded-full text-lg font-semibold shadow-md transition-colors duration-200 flex items-center"><Home size={20} className="mr-2" /> Back to Home</button>

</section>

);

};

// \*\*NEW\*\* ProfilePage Component

const ProfilePage = ({ userDetails }) => {

if (!userDetails) {

return (

<div className="flex justify-center items-center h-screen">

<p className="text-xl text-gray-500">Loading profile...</p>

</div>

);

}

const { username, phoneNumber, userId, createdAt } = userDetails;

// \*\*FIXED DATE HANDLING\*\*

const createdDate = createdAt ? new Date(createdAt).toLocaleDateString('en-GB', { day: 'numeric', month: 'long', year: 'numeric' }) : 'N/A';

const createdTime = createdAt ? new Date(createdAt).toLocaleTimeString('en-US', { hour: '2-digit', minute: '2-digit', hour12: true }) : '';

return (

<section className="container mx-auto py-12 md:py-16 px-4">

<h2 className="text-3xl md:text-4xl font-bold text-gray-800 text-center mb-10">Your Profile</h2>

<div className="max-w-md mx-auto bg-white rounded-xl shadow-lg p-8 space-y-6">

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

<UserIcon size={24} className="text-orange-500"/>

<div>

<p className="text-sm text-gray-500">Username</p>

<p className="text-lg font-semibold text-gray-800">{username}</p>

</div>

</div>

<hr/>

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

<IdCard size={24} className="text-orange-500"/>

<div>

<p className="text-sm text-gray-500">User ID</p>

<p className="text-lg font-semibold text-gray-800">{userId || 'N/A'}</p>

</div>

</div>

<hr/>

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

<Phone size={24} className="text-orange-500"/>

<div>

<p className="text-sm text-gray-500">Phone Number</p>

<p className="text-lg font-semibold text-gray-800">{phoneNumber}</p>

</div>

</div>

<hr/>

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

<Calendar size={24} className="text-orange-500"/>

<div>

<p className="text-sm text-gray-500">Member Since</p>

<p className="text-lg font-semibold text-gray-800">{createdDate} {createdTime}</p>

</div>

</div>

</div>

</section>

);

};

// SearchModal Component

const SearchModal = ({ onClose, restaurants, onSelectRestaurant }) => {

const [searchTerm, setSearchTerm] = useState('');

const [searchResults, setSearchResults] = useState([]);

useEffect(() => {

if (!searchTerm.trim()) {

setSearchResults([]);

return;

}

const lowerCaseSearchTerm = searchTerm.toLowerCase();

const filtered = restaurants.filter(

rest => rest.name.toLowerCase().includes(lowerCaseSearchTerm) || rest.cuisine.toLowerCase().includes(lowerCaseSearchTerm) || rest.address.toLowerCase().includes(lowerCaseSearchTerm)

);

setSearchResults(filtered);

}, [searchTerm, restaurants]);

return (

<div className="modal-overlay">

<div className="modal-content w-full md:w-3/4 lg:w-1/2">

<button onClick={onClose} className="modal-close-button"><X size={24} /></button>

<h2 className="text-3xl font-bold text-gray-800 mb-6 text-center">Search Restaurants</h2>

<input type="text" placeholder="Search by name, cuisine, or address..." className="shadow appearance-none border rounded-lg w-full py-3 px-4 text-gray-700 leading-tight focus:outline-none focus:ring-2 focus:ring-orange-500 mb-6" value={searchTerm} onChange={(e) => setSearchTerm(e.target.value)} />

{searchResults.length > 0 ? (

<div className="space-y-4 max-h-80 overflow-y-auto pr-2">

{searchResults.map(restaurant => (

<div key={restaurant.\_id} className="flex items-center space-x-4 p-3 bg-gray-50 rounded-lg hover:bg-gray-100 cursor-pointer transition-colors" onClick={() => { onSelectRestaurant(restaurant); onClose(); }}>

<img src={restaurant.image} alt={restaurant.name} className="w-16 h-16 object-cover rounded-md" onError={(e) => { e.target.onerror = null; e.target.src='https://placehold.co/64x64/CCCCCC/FFFFFF?text=Error'; }} />

<div><p className="font-semibold text-lg text-gray-800">{restaurant.name}</p><p className="text-gray-600 text-sm">{restaurant.cuisine} - {restaurant.address}</p></div>

</div>

))}

</div>

) : (searchTerm && <p className="text-center text-gray-600">No results found for "{searchTerm}".</p>)}

</div>

</div>

);

};

// Splash Screen Component

const SplashScreen = ({ isVisible }) => (

<div className={`splash-screen ${!isVisible ? 'hidden' : ''}`}>

<span className="splash-screen-logo">🍽️</span>

<h1 className="text-5xl font-extrabold tracking-wide">DHK FOODIES</h1>

<p className="text-lg mt-4 animate-pulse">Savor the Flavor, Delight Your Palate!</p>

</div>

);

// Main App component

function App() {

const [appVisible, setAppVisible] = useState(false);

const [showSplashScreen, setShowSplashScreen] = useState(true);

const [isAuthReady, setIsAuthReady] = useState(false);

const [isMobileMenuOpen, setIsMobileMenuOpen] = useState(false);

const [cart, setCart] = useState([]);

const [currentPage, setCurrentPage] = useState('home');

const [isCartOpen, setIsCartOpen] = useState(false);

const [isAuthModalOpen, setIsAuthModalOpen] = useState(false);

const [selectedRestaurant, setSelectedRestaurant] = useState(null);

const [isPaymentModalOpen, setIsPaymentModalOpen] = useState(false);

const [isDeliveryAddressModalOpen, setIsDeliveryAddressModalOpen] = useState(false);

const [currentDeliveryAddress, setCurrentDeliveryAddress] = useState(null);

const [currentCartTotal, setCurrentCartTotal] = useState(0);

const [orderPlacedDetails, setOrderPlacedDetails] = useState(null);

const [isLoggedIn, setIsLoggedIn] = useState(false);

const [currentUser, setCurrentUser] = useState(null);

const [currentUserDetails, setCurrentUserDetails] = useState(null);

const [userOrderHistory, setUserOrderHistory] = useState([]);

const [isSearchModalOpen, setIsSearchModalOpen] = useState(false);

const [isFilterModalOpen, setIsFilterModalOpen] = useState(false);

const [activeFilters, setActiveFilters] = useState({

diet: 'all',

sortBy: 'relevance',

categories: [],

});

// Mock Data

const mockRestaurants = [

{ \_id: 'rest\_001', name: 'Andhra Ruchulu', cuisine: 'South Indian', rating: 4.5, address: 'Hyderabad', image: 'https://images.pexels.com/photos/958545/pexels-photo-958545.jpeg?auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1', hasOffers: true, isNonVeg: true, menu: [ { \_id: 'item\_001', name: 'Gongura Mamsam', emoji: '🍖', description: 'Spicy mutton curry.', price: 420.00, category: 'Non-Veg Curries' }, { \_id: 'item\_002', name: 'Andhra Chicken Biryani', emoji: '🍚🍗', description: 'Spicy chicken biryani.', price: 380.00, category: 'Biryani' }, { \_id: 'item\_003', name: 'Pappu Charu', emoji: '🥣', description: 'Tangy lentil soup.', price: 150.00, category: 'Veg Curries' }, { \_id: 'item\_004', name: 'Ragi Mudde', emoji: '🫓', description: 'Finger millet balls.', price: 100.00, category: 'Breads' }, { \_id: 'item\_005', name: 'Boorelu', emoji: '🍩', description: 'Sweet fried dumplings.', price: 90.00, category: 'Desserts' }, { \_id: 'item\_006', name: 'Andhra Pulihora', emoji: '🍋🍚', description: 'Tangy tamarind rice.', price: 200.00, category: 'Rice Dishes' }, { \_id: 'item\_007', name: 'Fish Pulusu', emoji: '🐟🌶️', description: 'Sour fish curry.', price: 450.00, category: 'Seafood' }, { \_id: 'item\_008', name: 'Mirchi Bajji', emoji: '🌶️🥟', description: 'Chili fritters.', price: 70.00, category: 'Snacks' }, { \_id: 'item\_009', name: 'Filter Coffee', emoji: '☕', description: 'South Indian filter coffee.', price: 60.00, category: 'Drinks' }, { \_id: 'item\_101', name: 'Mutton Fry', emoji: '🍖🔥', description: 'Spicy dry mutton fry.', price: 450.00, category: 'Non-Veg Curries' } ]},

{ \_id: 'rest\_002', name: 'Rayalaseema Ruchulu', cuisine: 'Rayalaseema', rating: 4.3, address: 'Vijayawada', image: 'https://images.pexels.com/photos/1633578/pexels-photo-1633578.jpeg?auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1', hasOffers: false, isNonVeg: true, menu: [ { \_id: 'item\_010', name: 'Natukodi Pulusu', emoji: '🐔🌶️', description: 'Country chicken curry.', price: 480.00, category: 'Non-Veg Curries' }, { \_id: 'item\_011', name: 'Jonna Rotte', emoji: '🌾🫓', description: 'Sorghum flatbread.', price: 80.00, category: 'Breads' }, { \_id: 'item\_012', name: 'Ulavacharu Biryani', emoji: '🍚🍛', description: 'Horse gram biryani.', price: 400.00, category: 'Biryani'}, { \_id: 'item\_013', name: 'Peanut Chutney', emoji: '🥜🥣', description: 'Spicy peanut chutney.', price: 50.00, category: 'Accompaniments' }, { \_id: 'item\_014', name: 'Jalebi', emoji: '🥨🍯', description: 'Crispy, syrupy sweet.', price: 100.00, category: 'Desserts' }, { \_id: 'item\_015', name: 'Masala Chaas', emoji: '🥛🌿', description: 'Spiced buttermilk.', price: 70.00, category: 'Drinks' }, { \_id: 'item\_102', name: 'Ragi Sangati', emoji: '🌾🍚', description: 'Finger millet porridge.', price: 120.00, category: 'Breads' }, { \_id: 'item\_103', name: 'Brain Fry', emoji: '🧠🍳', description: 'Spicy lamb brain fry.', price: 350.00, category: 'Non-Veg Curries' }, { \_id: 'item\_104', name: 'Mutton Keema Balls', emoji: '🍖💣', description: 'Spiced minced mutton balls.', price: 400.00, category: 'Snacks' }, { \_id: 'item\_105', name: 'Badam Halwa', emoji: '🍮🌰', description: 'Almond sweet.', price: 150.00, category: 'Desserts' } ]},

{ \_id: 'rest\_003', name: 'Coastal Kitchen', cuisine: 'Seafood', rating: 4.6, address: 'Visakhapatnam', image: 'https://images.pexels.com/photos/7245465/pexels-photo-7245465.jpeg?auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1', hasOffers: true, isNonVeg: true, menu: [ { \_id: 'item\_016', name: 'Royyala Iguru', emoji: '🦐🌶️', description: 'Spicy prawn fry.', price: 550.00, category: 'Seafood' }, { \_id: 'item\_017', name: 'Chepala Pulusu', emoji: '🐟🥘', description: 'Tangy fish curry.', price: 480.00, category: 'Seafood' }, { \_id: 'item\_018', name: 'Kodi Pulao', emoji: '🍚🍗', description: 'Flavorful chicken pulao.', price: 370.00, category: 'Rice Dishes' }, { \_id: 'item\_019', name: 'Vegetable Kurma', emoji: '🥕🥦', description: 'Mixed vegetable curry.', price: 250.00, category: 'Veg Curries' }, { \_id: 'item\_020', name: 'Garelu', emoji: '🫘🍩', description: 'Savory lentil fritters.', price: 90.00, category: 'Snacks' }, { \_id: 'item\_021', name: 'Coconut Water', emoji: '🥥💧', description: 'Fresh tender coconut.', price: 80.00, category: 'Drinks' }, { \_id: 'item\_106', name: 'Crab Curry', emoji: '🦀🥘', description: 'Spicy crab masala.', price: 600.00, category: 'Seafood' }, { \_id: 'item\_107', name: 'Prawn Biryani', emoji: '🍚🦐', description: 'Aromatic prawn biryani.', price: 580.00, category: 'Biryani' }, { \_id: 'item\_108', name: 'Appam', emoji: '🥞', description: 'Soft rice pancake.', price: 60.00, category: 'Breads' }, { \_id: 'item\_109', name: 'Coconut Ladoo', emoji: '🥥🍬', description: 'Sweet coconut balls.', price: 110.00, category: 'Desserts' } ]},

{ \_id: 'rest\_004', name: 'Hyderabad House', cuisine: 'Hyderabadi', rating: 4.2, address: 'Guntur', image: 'https://images.pexels.com/photos/2474661/pexels-photo-2474661.jpeg?auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1', hasOffers: false, isNonVeg: true, menu: [ { \_id: 'item\_022', name: 'Mutton Kothmir', emoji: '🐑🌿', description: 'Mutton with coriander.', price: 460.00, category: 'Non-Veg Curries' }, { \_id: 'item\_027', name: 'Chicken 65', emoji: '🍗🌶️', description: 'Spicy fried chicken.', price: 290.00, category: 'Snacks' }, { \_id: 'item\_023', name: 'Bagara Baingan', emoji: '🍆🥘', description: 'Hyderabadi brinjal curry.', price: 260.00, category: 'Veg Curries' }, { \_id: 'item\_024', name: 'Double Ka Meetha', emoji: '🍞🍮', description: 'Bread pudding dessert.', price: 130.00, category: 'Desserts' }, { \_id: 'item\_025', name: 'Shahi Tukda', emoji: '👑🍰', description: 'Fried bread in milk.', price: 140.00, category: 'Desserts' }, { \_id: 'item\_026', name: 'Lassi', emoji: '🥛🥭', description: 'Sweet yogurt drink.', price: 80.00, category: 'Drinks' }, { \_id: 'item\_110', name: 'Mutton Biryani', emoji: '🍚🐑', description: 'Classic mutton biryani.', price: 500.00, category: 'Biryani' }, { \_id: 'item\_111', name: 'Haleem', emoji: '🍲', description: 'Slow-cooked meat stew.', price: 450.00, category: 'Non-Veg Curries' }, { \_id: 'item\_112', name: 'Mirchi ka Salan', emoji: '🌶️🥣', description: 'Chilli curry.', price: 180.00, category: 'Accompaniments' }, { \_id: 'item\_113', name: 'Qubani Ka Meetha', emoji: '🍑🍮', description: 'Apricot dessert.', price: 160.00, category: 'Desserts' } ]},

{ \_id: 'rest\_005', name: 'Guntur Spicy', cuisine: 'Spicy Andhra', rating: 4.7, address: 'Guntur', image: 'https://images.pexels.com/photos/12737656/pexels-photo-12737656.jpeg?auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1', hasOffers: true, isNonVeg: true, menu: [ { \_id: 'item\_028', name: 'Guntur Chicken Fry', emoji: '🌶️🐔', description: 'Fiery chicken dry fry.', price: 390.00, category: 'Non-Veg Curries' }, { \_id: 'item\_031', name: 'Spicy Egg Curry', emoji: '🥚🌶️', description: 'Eggs in spicy gravy.', price: 280.00, category: 'Non-Veg Curries' }, { \_id: 'item\_029', name: 'Red Chilli Dosa', emoji: '🌶️🥞', description: 'Spicy red chili dosa.', price: 140.00, category: 'Breads' }, { \_id: 'item\_030', name: 'Ghee Karam Dosa', emoji: '🥞🧈', description: 'Dosa with spicy powder.', price: 130.00, category: 'Breads' }, { \_id: 'item\_032', name: 'Lemon Soda', emoji: '🍋🥤', description: 'Refreshing lemon soda.', price: 60.00, category: 'Drinks' }, { \_id: 'item\_114', name: 'Mutton Kura', emoji: '🍖🌶️', description: 'Spicy Guntur mutton curry.', price: 480.00, category: 'Non-Veg Curries' }, { \_id: 'item\_115', name: 'Vankaya Pulusu', emoji: '🍆🥣', description: 'Spicy brinjal stew.', price: 220.00, category: 'Veg Curries' }, { \_id: 'item\_116', name: 'Onion Pakoda', emoji: '🧅🥨', description: 'Crispy onion fritters.', price: 100.00, category: 'Snacks' }, { \_id: 'item\_117', name: 'Mango Lassi', emoji: '🥭🥛', description: 'Sweet mango yogurt drink.', price: 90.00, category: 'Drinks' }, { \_id: 'item\_118', name: 'Pappu Annam', emoji: '🍚🥣', description: 'Lentil rice.', price: 150.00, category: 'Rice Dishes' } ]},

{ \_id: 'rest\_010', name: 'Biryani Point', cuisine: 'Biryani', rating: 4.8, address: 'Vijayawada', image: 'https://images.pexels.com/photos/5410403/pexels-photo-5410403.jpeg?auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1', hasOffers: false, isNonVeg: true, menu: [ { \_id: 'item\_045', name: 'Vijayawada Chicken Biryani', emoji: '🍚🍗', description: 'Famous spicy biryani.', price: 400.00, category: 'Biryani' }, { \_id: 'item\_046', name: 'Mutton Dum Biryani', emoji: '🍚🐑', description: 'Slow-cooked mutton.', price: 450.00, category: 'Biryani' }, { \_id: 'item\_047', name: 'Egg Biryani', emoji: '🍚🥚', description: 'Biryani with boiled eggs.', price: 320.00, category: 'Biryani' }, { \_id: 'item\_048', name: 'Paneer Biryani', emoji: '🍚🧀', description: 'Biryani with cottage cheese.', price: 340.00, category: 'Biryani' }, { \_id: 'item\_049', name: 'Mirchi Ka Salan', emoji: '🌶️🥣', description: 'Spicy chili gravy.', price: 80.00, category: 'Accompaniments' }, { \_id: 'item\_119', name: 'Fish Biryani', emoji: '🍚🐟', description: 'Flavorful fish biryani.', price: 480.00, category: 'Biryani' }, { \_id: 'item\_120', name: 'Special Chicken Biryani', emoji: '🌟🍚🍗', description: 'Chef\'s special biryani.', price: 450.00, category: 'Biryani' }, { \_id: 'item\_121', name: 'Veg Biryani', emoji: '🍚🥕', description: 'Mixed vegetable biryani.', price: 300.00, category: 'Biryani' }, { \_id: 'item\_122', name: 'Raita', emoji: '🥣🥒', description: 'Yogurt side dish.', price: 50.00, category: 'Accompaniments' }, { \_id: 'item\_123', name: 'Gulab Jamun', emoji: '🍮', description: 'Sweet milk solids balls.', price: 90.00, category: 'Desserts' } ]},

{ \_id: 'rest\_026', name: 'Veggie Delight', cuisine: 'Vegetarian', rating: 4.1, address: 'Tirupati', image: 'https://images.pexels.com/photos/1058277/pexels-photo-1058277.jpeg?auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1', hasOffers: true, isNonVeg: false, menu: [ { \_id: 'item\_124', name: 'Gutthi Vankaya Curry', emoji: '🍆🥘', description: 'Stuffed brinjal curry.', price: 250.00, category: 'Veg Curries' }, { \_id: 'item\_125', name: 'Goli Bajji', emoji: '🥟🧅', description: 'Fried lentil balls.', price: 70.00, category: 'Snacks' }, { \_id: 'item\_126', name: 'Jangiri', emoji: '🌸🍯', description: 'Flower-shaped sweet.', price: 130.00, category: 'Desserts' }, { \_id: 'item\_127', name: 'Paneer Butter Masala', emoji: '🧀🧈', description: 'Creamy paneer curry.', price: 280.00, category: 'Veg Curries' }, { \_id: 'item\_128', name: 'Veg Fried Rice', emoji: '🍚🥕', description: 'Stir-fried rice.', price: 200.00, category: 'Rice Dishes' }, { \_id: 'item\_129', name: 'Masala Dosa', emoji: '🥞🥔', description: 'Dosa with potato filling.', price: 120.00, category: 'Breads' }, { \_id: 'item\_130', name: 'Tomato Soup', emoji: '🥣🍅', description: 'Classic tomato soup.', price: 100.00, category: 'Soups' }, { \_id: 'item\_131', name: 'Mushroom 65', emoji: '🍄🌶️', description: 'Spicy fried mushrooms.', price: 240.00, category: 'Snacks' }, { \_id: 'item\_132', name: 'Vegetable Thali', emoji: '🍛', description: 'A full veg meal.', price: 300.00, category: 'Thali' }, { \_id: 'item\_133', name: 'Fresh Lime Soda', emoji: '🍋🥤', description: 'Sweet and salt soda.', price: 70.00, category: 'Drinks' } ]},

{ \_id: 'rest\_030', name: 'Filter Coffee', cuisine: 'Cafe', rating: 4.5, address: 'Visakhapatnam', image: 'https://images.pexels.com/photos/302899/pexels-photo-302899.jpeg?auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1', hasOffers: false, isNonVeg: false, menu: [{\_id: 'item\_134', name: 'Strong Filter Coffee', emoji: '☕☕', description: 'Robust traditional coffee.', price: 70.00, category: 'Drinks' }, { \_id: 'item\_135', name: 'Masala Chai', emoji: '☕🌶️', description: 'Spiced Indian tea.', price: 60.00, category: 'Drinks' }, { \_id: 'item\_136', name: 'Osmania Biscuit', emoji: '🍪', description: 'Hyderabadi shortbread.', price: 30.00, category: 'Snacks' }, { \_id: 'item\_137', name: 'Veg Puff', emoji: '🥐', description: 'Vegetable-filled pastry.', price: 50.00, category: 'Snacks' }, { \_id: 'item\_138', name: 'Samosa', emoji: '🥟', description: 'Potato-filled pastry.', price: 40.00, category: 'Snacks' }, { \_id: 'item\_139', name: 'Ginger Tea', emoji: '☕🌿', description: 'Tea with fresh ginger.', price: 65.00, category: 'Drinks' }, { \_id: 'item\_140', name: 'Black Coffee', emoji: '☕', description: 'Simple black coffee.', price: 50.00, category: 'Drinks' }, { \_id: 'item\_141', name: 'Cookies', emoji: '🍪', description: 'Assorted cookies.', price: 80.00, category: 'Desserts' }, { \_id: 'item\_142', name: 'Milkshake', emoji: '🥤🍫', description: 'Chocolate milkshake.', price: 120.00, category: 'Drinks' }, { \_id: 'item\_143', name: 'Bread Omelette', emoji: '🍞🍳', description: 'Simple bread omelette.', price: 90.00, category: 'Snacks' } ]}

];

const availableCategories = [...new Set(mockRestaurants.flatMap(r => r.menu.map(item => item.category)))];

const filteredRestaurants = mockRestaurants.filter(restaurant => {

const { diet, categories } = activeFilters;

if (diet === 'veg' && restaurant.isNonVeg) return false;

if (diet === 'nonVeg' && !restaurant.isNonVeg) return false;

if (categories.length > 0) {

const restaurantCategories = new Set(restaurant.menu.map(item => item.category));

const hasCategory = categories.some(cat => restaurantCategories.has(cat));

if (!hasCategory) return false;

}

return true;

}).map(r => {

const avgPrice = r.menu.reduce((acc, item) => acc + item.price, 0) / r.menu.length;

return { ...r, avgPrice };

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

const { sortBy } = activeFilters;

if (sortBy === 'price\_asc') return a.avgPrice - b.avgPrice;

if (sortBy === 'price\_desc') return b.avgPrice - a.avgPrice;

return 0; // relevance

});

// \*\*AUTH & DATA LOADING LOGIC\*\*

useEffect(() => {

const storedUsername = localStorage.getItem('loggedInUser');

if (storedUsername) {

const users = JSON.parse(localStorage.getItem('mockUsers')) || [];

const userDetails = users.find(user => user.username === storedUsername);

if (userDetails) {

setIsLoggedIn(true);

setCurrentUser(userDetails.username);

setCurrentUserDetails(userDetails);

}

}

setIsAuthReady(true);

}, []);

useEffect(() => {

if (isAuthReady) {

const timer = setTimeout(() => {

setShowSplashScreen(false);

if (isLoggedIn) {

setAppVisible(true);

} else {

setIsAuthModalOpen(true);

}

}, 1500);

return () => clearTimeout(timer);

}

}, [isAuthReady, isLoggedIn]);

useEffect(() => {

if (currentUser) {

const storedHistory = localStorage.getItem(`orderHistory\_${currentUser}`);

setUserOrderHistory(storedHistory ? JSON.parse(storedHistory) : []);

} else {

setUserOrderHistory([]);

}

}, [currentUser]);

useEffect(() => {

if (currentUser) {

localStorage.setItem(`orderHistory\_${currentUser}`, JSON.stringify(userOrderHistory));

}

}, [userOrderHistory, currentUser]);

useEffect(() => {

let intervalId;

if (currentPage === 'orders' && isLoggedIn) {

intervalId = setInterval(() => {

setUserOrderHistory(prevHistory => prevHistory.map(order => ({...order, status: getSimulatedOrderStatus(order.timestamp)})));

}, 10000);

}

return () => clearInterval(intervalId);

}, [currentPage, isLoggedIn]);

const getSimulatedOrderStatus = (timestamp) => {

const elapsedMinutes = (Date.now() - timestamp) / 60000;

if (elapsedMinutes < 0.25) return 'Confirmed';

if (elapsedMinutes < 1) return 'Preparing';

if (elapsedMinutes < 2) return 'Out for Delivery';

return 'Delivered';

};

// Handler Functions

const handleUpdateCartQuantity = (itemId, newQuantity) => {

setCart(prevCart => {

const itemToAdd = selectedRestaurant.menu.find(menuItem => menuItem.\_id === itemId);

if (!itemToAdd) return prevCart;

if (newQuantity <= 0) return prevCart.filter(item => item.\_id !== itemId);

const existingItem = prevCart.find(item => item.\_id === itemId);

if (existingItem) {

return prevCart.map(item => item.\_id === itemId ? { ...item, quantity: newQuantity } : item);

}

return [...prevCart, { ...itemToAdd, quantity: newQuantity }];

});

};

const handleLoginSuccess = (username) => {

const users = JSON.parse(localStorage.getItem('mockUsers')) || [];

const userDetails = users.find(user => user.username === username);

localStorage.setItem('loggedInUser', username);

setIsLoggedIn(true);

setCurrentUser(username);

setCurrentUserDetails(userDetails);

setIsAuthModalOpen(false);

setAppVisible(true);

setCurrentPage('home');

};

const handleLogout = () => {

localStorage.removeItem('loggedInUser');

setIsLoggedIn(false);

setCurrentUser(null);

setCurrentUserDetails(null);

setCart([]);

setAppVisible(false);

setCurrentPage('home');

setIsAuthModalOpen(true);

};

const handleProceedToDeliveryAddress = (total) => {

if (cart.length === 0) return;

setCurrentCartTotal(total);

setIsCartOpen(false);

setIsDeliveryAddressModalOpen(true);

};

const handleSubmitDeliveryAddress = (addressDetails) => {

setCurrentDeliveryAddress(addressDetails);

setIsDeliveryAddressModalOpen(false);

setIsPaymentModalOpen(true);

};

const handlePaymentSuccess = (paymentInfo) => {

const newOrder = {

...paymentInfo,

items: cart,

deliveryAddress: currentDeliveryAddress,

status: 'Confirmed'

};

setOrderPlacedDetails(newOrder);

setUserOrderHistory(prevHistory => [...prevHistory, newOrder]);

setCart([]);

setIsPaymentModalOpen(false);

setCurrentPage('orderConfirmation');

};

const renderContent = () => {

if (currentPage === 'orderConfirmation' && orderPlacedDetails) {

return <OrderConfirmation orderDetails={orderPlacedDetails} onGoHome={() => {setCurrentPage('home'); setOrderPlacedDetails(null);}} />;

}

if (currentPage === 'restaurantMenu' && selectedRestaurant) {

return <RestaurantMenuPage restaurant={selectedRestaurant} cart={cart} onUpdateCartQuantity={handleUpdateCartQuantity} onBackToRestaurants={() => {setSelectedRestaurant(null); setCurrentPage('menu');}} />;

}

switch (currentPage) {

case 'home': return <>

<section className="relative bg-gradient-to-r from-orange-400 to-red-500 text-white py-16 md:py-24 text-center overflow-hidden rounded-b-xl shadow-lg">

<div className="absolute inset-0 opacity-10 bg-cover bg-center animate-pulse-bg" style={{backgroundImage: "url('https://www.transparenttextures.com/patterns/food.png')"}}></div>

<div className="container mx-auto relative z-10 px-4">

<h2 className="text-4xl md:text-6xl font-bold mb-4 leading-tight drop-shadow-lg">Savor the Flavor, Delight Your Palate!</h2>

<p className="text-lg md:text-xl mb-8 max-w-2xl mx-auto opacity-95">Discover a world of delicious Indian dishes crafted with passion and served with a smile.</p>

<button onClick={() => setCurrentPage('menu')} className="bg-white text-orange-500 hover:bg-gray-100 px-8 py-3 rounded-full text-lg font-semibold shadow-xl transform transition-all duration-300 ease-in-out hover:scale-105">Explore Our Restaurants</button>

</div>

</section>

<section className="container mx-auto py-12 md:py-16 px-4">

<h2 className="text-3xl md:text-4xl font-bold text-gray-800 text-center mb-10">Popular Restaurants</h2>

<div className="grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-3 gap-8">{mockRestaurants.slice(0, 3).map(r => <RestaurantCard key={r.\_id} restaurant={r} onViewMenu={(restaurant) => {setSelectedRestaurant(restaurant); setCurrentPage('restaurantMenu');}} />)}</div>

<div className="text-center mt-10"><button onClick={() => setCurrentPage('menu')} className="bg-gray-200 text-gray-800 hover:bg-gray-300 px-8 py-3 rounded-full text-lg font-semibold shadow-md transition-colors">View All Restaurants</button></div>

</section>

</>;

case 'menu': return <section className="container mx-auto py-12 md:py-16 px-4">

<div className="flex justify-between items-center mb-10">

<h2 className="text-3xl md:text-4xl font-bold text-gray-800">All Restaurants</h2>

<button onClick={() => setIsFilterModalOpen(true)} className="flex items-center bg-white border border-gray-300 text-gray-700 px-4 py-2 rounded-lg hover:bg-gray-50 shadow-sm transition-colors">

<FilterIcon size={20} className="mr-2"/> Filter

</button>

</div>

{filteredRestaurants.length > 0 ?

<div className="grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-3 xl:grid-cols-4 gap-8">{filteredRestaurants.map(r => <RestaurantCard key={r.\_id} restaurant={r} onViewMenu={(restaurant) => {setSelectedRestaurant(restaurant); setCurrentPage('restaurantMenu');}} />)}</div> :

<p className="text-center text-gray-600 text-lg">No restaurants match your filters.</p>

}

</section>;

case 'contact': return <section className="container mx-auto py-12 md:py-16 px-4 text-center"><h2 className="text-3xl md:text-4xl font-bold text-gray-800 mb-6">Contact Us</h2><p className="text-lg text-gray-700 max-w-xl mx-auto">Have questions or feedback? We'd love to hear from you!</p><div className="mt-8 space-y-4"><p className="text-xl font-semibold text-gray-800">Email: <a href="mailto:hemanthheman86@gmail.com" className="text-orange-500 hover:underline">hemanthheman86@gmail.com</a></p><p className="text-xl font-semibold text-gray-800">Phone: <a href="tel:+917702807424" className="text-orange-500 hover:underline">+91 7702807424</a></p><p className="text-lg text-gray-600">Our support team is available Monday to Friday, 9 AM - 6 PM.</p></div></section>;

case 'orders': return <section className="container mx-auto py-12 md:py-16 px-4">

<h2 className="text-3xl md:text-4xl font-bold text-gray-800 mb-6 text-center">Your Orders</h2>

{!isLoggedIn ? <div className="text-center"><p className="text-lg text-gray-700">Please log in to view your orders.</p><button onClick={() => setIsAuthModalOpen(true)} className="mt-6 bg-orange-500 hover:bg-orange-600 text-white px-6 py-3 rounded-full text-lg font-semibold">Login</button></div> :

<div className="text-center">

<p className="text-2xl text-gray-800 font-semibold mb-8">Welcome, {currentUser}! Here are your orders:</p>

{userOrderHistory.length === 0 ? <p className="text-gray-600 text-lg">You have no past orders.</p> :

<div className="grid grid-cols-1 lg:grid-cols-2 gap-6 max-w-4xl mx-auto">

{userOrderHistory.slice().sort((a,b) => b.timestamp - a.timestamp).map(order => {

const statusColor = {'Confirmed': 'text-blue-600', 'Preparing': 'text-yellow-600', 'Out for Delivery': 'text-purple-600', 'Delivered': 'text-green-600'}[order.status] || 'text-gray-600';

return <div key={order.orderId} className="bg-white p-6 rounded-lg shadow-md text-left"><h3 className="text-xl font-bold text-gray-900 mb-2 flex items-center"><Package size={20} className="mr-2"/> Order #{order.orderId}</h3><p className="text-gray-700 mb-3">Date: {new Date(order.timestamp).toLocaleString()}</p><p className="font-semibold text-gray-800 mt-4">Items:</p><ul className="list-disc list-inside text-gray-600 ml-4">{order.items.map((item, idx) => <li key={idx}>{item.emoji} {item.name} x {item.quantity}</li>)}</ul><div className="flex items-center justify-between mt-4 pt-4 border-t"><div className="flex items-center"><Truck className={`mr-2 ${statusColor}`} size={24} /><p className={`font-semibold text-lg ${statusColor}`}>Status: {order.status}</p></div></div></div>

})}

</div>}

</div>}

</section>;

case 'profile': return <ProfilePage userDetails={currentUserDetails} />;

default: return null;

}

};

return (

<div className="min-h-screen bg-gray-100 font-sans">

<SplashScreen isVisible={showSplashScreen} />

{appVisible && (

<>

<header className="bg-white shadow-sm p-4 md:p-6 sticky top-0 z-50">

<nav className="container mx-auto flex justify-between items-center">

<div className="flex items-center space-x-2 cursor-pointer" onClick={() => setCurrentPage('home')}><img src="https://placehold.co/40x40/FF5733/FFFFFF?text=🍽️" alt="DHK Foodies Logo" className="rounded-full" /><h1 className="text-2xl md:text-3xl font-extrabold text-gray-800 tracking-tight">DHK FOODIES</h1></div>

<div className="hidden md:flex items-center space-x-8">{['home', 'menu', 'profile', 'orders', 'contact'].map(page => <a key={page} onClick={() => setCurrentPage(page)} className={`cursor-pointer capitalize ${currentPage.startsWith(page) ? 'text-orange-500' : 'text-gray-600'} hover:text-orange-500 font-medium`}>{page === 'menu' ? 'Restaurants' : page}</a>)}</div>

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

<Search className="text-gray-600 hover:text-orange-500 cursor-pointer" size={24} onClick={() => setIsSearchModalOpen(true)} />

<div className="relative cursor-pointer" onClick={() => setIsCartOpen(true)}><ShoppingCart className="text-gray-600 hover:text-orange-500" size={24} /><span className="absolute -top-2 -right-2 bg-red-500 text-white rounded-full text-xs w-5 h-5 flex items-center justify-center">{cart.reduce((sum, item) => sum + item.quantity, 0)}</span></div>

{isLoggedIn ? <div className="flex items-center space-x-2 group relative"><LogOut className="text-gray-600 hover:text-orange-500 cursor-pointer" size={24} onClick={handleLogout} /><span className="text-gray-700 font-medium hidden group-hover:block absolute top-full left-1/2 -translate-x-1/2 bg-gray-700 text-white text-xs px-2 py-1 rounded-md mt-1 whitespace-nowrap">Logout</span></div> : <UserIcon className="text-gray-600 hover:text-orange-500 cursor-pointer" size={24} onClick={() => setIsAuthModalOpen(true)} />}

</div>

<div className="md:hidden flex items-center space-x-4"><div className="relative cursor-pointer" onClick={() => setIsCartOpen(true)}><ShoppingCart className="text-gray-600 hover:text-orange-500" size={28} /><span className="absolute -top-2 -right-2 bg-red-500 text-white rounded-full text-xs w-5 h-5 flex items-center justify-center">{cart.reduce((sum, item) => sum + item.quantity, 0)}</span></div><button onClick={() => setIsMobileMenuOpen(o => !o)} className="text-gray-600 hover:text-orange-500"><Menu size={28} /></button></div>

</nav>

{isMobileMenuOpen && <div className="md:hidden mt-4 bg-white border-t py-4"><nav className="flex flex-col items-center space-y-3">

{['home', 'menu', 'profile', 'orders', 'contact'].map(page => <a key={page} onClick={() => {setCurrentPage(page); setIsMobileMenuOpen(false);}} className={`w-full text-center cursor-pointer capitalize ${currentPage.startsWith(page) ? 'text-orange-500' : 'text-gray-700'} hover:text-orange-500 font-medium py-2`}>{page === 'menu' ? 'Restaurants' : page}</a>)}

{isLoggedIn && (<div className="w-full border-t border-gray-200 mt-4 pt-4"><a onClick={() => { handleLogout(); setIsMobileMenuOpen(false); }} className="w-full text-center cursor-pointer text-red-500 hover:text-red-700 font-medium py-2 flex items-center justify-center"><LogOut size={20} className="mr-2"/>Logout</a></div>)}

</nav></div>}

</header>

{renderContent()}

<footer className="bg-gray-800 text-white py-8 md:py-10 mt-12">

<div className="container mx-auto text-center"><p>&copy; {new Date().getFullYear()} DHK FOODIES. All rights reserved.</p></div>

</footer>

</>

)}

{isAuthModalOpen && <AuthModal onClose={() => setIsAuthModalOpen(false)} onLoginSuccess={handleLoginSuccess} />}

{appVisible && isCartOpen && <CartView cartItems={cart} onClose={() => setIsCartOpen(false)} onUpdateQuantity={handleUpdateCartQuantity} onRemoveItem={(id) => setCart(c => c.filter(i => i.\_id !== id))} onProceedToPayment={handleProceedToDeliveryAddress} />}

{appVisible && isDeliveryAddressModalOpen && <DeliveryAddressModal onClose={() => setIsDeliveryAddressModalOpen(false)} onSubmitAddress={handleSubmitDeliveryAddress} />}

{appVisible && isPaymentModalOpen && <PaymentModal onClose={() => setIsPaymentModalOpen(false)} totalAmount={currentCartTotal} onPaymentSuccess={handlePaymentSuccess} />}

{appVisible && isSearchModalOpen && <SearchModal onClose={() => setIsSearchModalOpen(false)} restaurants={mockRestaurants} onSelectRestaurant={(r) => {setSelectedRestaurant(r); setCurrentPage('restaurantMenu');}} />}

{appVisible && <FilterSidebar isOpen={isFilterModalOpen} onClose={() => setIsFilterModalOpen(false)} onApplyFilters={setActiveFilters} availableCategories={availableCategories} initialFilters={activeFilters}/>}

</div>

);

}

ReactDOM.render(<App />, document.getElementById('root'));

</script>

</body>

</html>