import { useState, useEffect, useRef } from ‘react’

interface SideCartProps { isOpen: boolean; onClose: () => void; pricing: { currency: string; paypalClientId: string; }; selectedPackage: ‘1day’ | ‘2days’ | null; onPackageChange: (pkg: ‘1day’ | ‘2days’ | null) => void; }

// PayPal component for side cart const PayPalButton = ({ amount, description, isVisible }: { amount: number; description: string; isVisible: boolean }) => { const containerRef = useRef(null);

useEffect(() => { // Clear previous button if (containerRef.current) { containerRef.current.innerHTML = ’’; }

if (window.paypal && amount > 0 && isVisible) {  
 // Small delay to ensure DOM is ready  
 const timer = setTimeout(() => {  
 if (containerRef.current) {  
 window.paypal.Buttons({  
 createOrder: (data: any, actions: any) => {  
 return actions.order.create({  
 purchase\_units: [{  
 amount: {  
 value: amount.toString(),  
 currency\_code: 'USD'  
 },  
 description: description  
 }]  
 })  
 },  
 onApprove: async (data: any, actions: any) => {  
 try {  
 const order = await actions.order.capture()  
 console.log('Order successful:', order)  
 alert('Payment successful! You will receive a confirmation email.')  
 } catch (error) {  
 console.error('Payment error:', error)  
 alert('Payment error. Please try again.')  
 }  
 },  
 onError: (err: any) => {  
 console.error('PayPal error:', err)  
 alert('PayPal error. Please try again.')  
 }  
 }).render(containerRef.current)  
 }  
 }, 200);  
  
 return () => clearTimeout(timer);  
}

}, [amount, description, isVisible])

if (amount === 0) return null

return

}

const SideCart = ({ isOpen, onClose, pricing, selectedPackage, onPackageChange }: SideCartProps) => { const [quantity, setQuantity] = useState(2) // Default to 2 days as requested

// Initialize quantity based on selected package useEffect(() => { if (selectedPackage === ‘1day’) { setQuantity(1) } else if (selectedPackage === ‘2days’) { setQuantity(2) } else if (isOpen && !selectedPackage) { // Default to 2 days when opening cart without specific selection setQuantity(2) } }, [selectedPackage, isOpen])

// Calculate price based on quantity const getPrice = (qty: number) => { if (qty === 2) return 100 if (qty === 1) return 60 return 0 }

// Get package name based on quantity const getPackageName = (qty: number) => { if (qty === 2) return ‘2-Day Pass’ if (qty === 1) return ‘1-Day Pass’ return ‘No selection’ }

const currentPrice = getPrice(quantity) const packageName = getPackageName(quantity)

const updateQuantity = (newQty: number) => { if (newQty >= 0 && newQty <= 2) { setQuantity(newQty) } }

return ( <> {/\* Overlay \*/} {isOpen && (

)}  
  
 {/\* Side Cart \*/}  
 <div className={`fixed right-0 top-0 h-full w-full sm:w-96 max-w-full bg-gray-900 border-l border-gray-700 z-50 transform transition-transform duration-300 ease-in-out flex flex-col ${  
 isOpen ? 'translate-x-0' : 'translate-x-full'  
 }`}>  
 {/\* Header \*/}  
 <div className="flex items-center justify-between p-4 sm:p-6 border-b border-gray-700 flex-shrink-0">  
 <h2 className="text-lg sm:text-xl font-bold text-white">Your Reservation</h2>  
 <button   
 onClick={onClose}  
 className="text-gray-400 hover:text-white transition-colors p-1"  
 >  
 <svg className="w-5 h-5 sm:w-6 sm:h-6" fill="none" stroke="currentColor" viewBox="0 0 24 24">  
 <path strokeLinecap="round" strokeLinejoin="round" strokeWidth={2} d="M6 18L18 6M6 6l12 12" />  
 </svg>  
 </button>  
 </div>  
  
 {/\* Scrollable Content \*/}  
 <div className="flex-1 overflow-y-auto">  
 <div className="p-4 sm:p-6 space-y-4 sm:space-y-6">  
 {/\* Event Info \*/}  
 <div className="bg-gray-800 rounded-lg p-4">  
 <h3 className="font-bold text-white mb-2">Jun Fan Jeet Kune Do Seminar</h3>  
 <p className="text-gray-400 text-sm">August 30-31, 2025</p>  
 <p className="text-gray-400 text-sm">Warrior Strength Martial Arts, Bellevue WA</p>  
 </div>  
  
 {/\* Package Selection \*/}  
 <div className="space-y-4">  
 <h4 className="font-bold text-white">Select Package</h4>  
   
 {/\* Quantity Selector \*/}  
 <div className="flex items-center justify-between bg-gray-800 rounded-lg p-4">  
 <div>  
 <p className="font-medium text-white">{packageName}</p>  
 {quantity > 0 && (  
 <p className="text-sm text-gray-400">  
 {quantity === 2 ? 'Complete 2-day experience' : 'Single day access'}  
 </p>  
 )}  
 </div>  
   
 <div className="flex items-center space-x-3">  
 <button   
 onClick={() => updateQuantity(quantity - 1)}  
 className="w-8 h-8 rounded-full bg-gray-700 hover:bg-gray-600 flex items-center justify-center text-white transition-colors"  
 disabled={quantity === 0}  
 >  
 <svg className="w-4 h-4" fill="none" stroke="currentColor" viewBox="0 0 24 24">  
 <path strokeLinecap="round" strokeLinejoin="round" strokeWidth={2} d="M20 12H4" />  
 </svg>  
 </button>  
   
 <span className="w-8 text-center font-bold text-white">{quantity}</span>  
   
 <button   
 onClick={() => updateQuantity(quantity + 1)}  
 className="w-8 h-8 rounded-full bg-yellow-400 hover:bg-yellow-300 flex items-center justify-center text-black transition-colors"  
 disabled={quantity === 2}  
 >  
 <svg className="w-4 h-4" fill="none" stroke="currentColor" viewBox="0 0 24 24">  
 <path strokeLinecap="round" strokeLinejoin="round" strokeWidth={2} d="M12 6v6m0 0v6m0-6h6m-6 0H6" />  
 </svg>  
 </button>  
 </div>  
 </div>  
  
 {/\* Quick Selection Buttons \*/}  
 <div className="grid grid-cols-2 gap-3">  
 <button   
 onClick={() => setQuantity(1)}  
 className={`p-3 rounded-lg border-2 transition-all ${  
 quantity === 1   
 ? 'border-yellow-400 bg-yellow-400/10 text-yellow-400'   
 : 'border-gray-600 hover:border-gray-500 text-gray-300'  
 }`}  
 >  
 <div className="text-sm font-medium">1 Day</div>  
 <div className="text-xs">{pricing.currency}60</div>  
 </button>  
   
 <button   
 onClick={() => setQuantity(2)}  
 className={`p-3 rounded-lg border-2 transition-all ${  
 quantity === 2   
 ? 'border-yellow-400 bg-yellow-400/10 text-yellow-400'   
 : 'border-gray-600 hover:border-gray-500 text-gray-300'  
 }`}  
 >  
 <div className="text-sm font-medium">2 Days</div>  
 <div className="text-xs">{pricing.currency}100</div>  
 <div className="text-xs text-green-400">Save {pricing.currency}20</div>  
 </button>  
 </div>  
 </div>  
  
 {/\* Price Summary \*/}  
 <div className="bg-gray-800 rounded-lg p-4 space-y-3">  
 <div className="flex justify-between">  
 <span className="text-gray-400">Package</span>  
 <span className="text-white">{packageName}</span>  
 </div>  
   
 {currentPrice > 0 && (  
 <>  
 <div className="flex justify-between">  
 <span className="text-gray-400">Quantity</span>  
 <span className="text-white">{quantity} day{quantity > 1 ? 's' : ''}</span>  
 </div>  
   
 <div className="border-t border-gray-700 pt-3">  
 <div className="flex justify-between items-center">  
 <span className="text-lg font-bold text-white">Total</span>  
 <span className="text-2xl font-bold text-yellow-400">  
 {pricing.currency}{currentPrice}  
 </span>  
 </div>  
 </div>  
 </>  
 )}  
 </div>  
  
 {/\* Empty Cart Message \*/}  
 {currentPrice === 0 && (  
 <div className="text-center py-8">  
 <div className="text-6xl mb-4">🛒</div>  
 <p className="text-gray-400 mb-4">Your cart is empty</p>  
 <p className="text-sm text-gray-500">Select 1 or 2 days to continue</p>  
 </div>  
 )}  
 </div>  
 </div>  
  
 {/\* Footer - Always visible when cart has items \*/}  
 {currentPrice > 0 && (  
 <div className="flex-shrink-0 p-4 sm:p-6 border-t border-gray-700 space-y-3 sm:space-y-4 bg-gray-900">  
 <div className="text-xs text-gray-400 text-center">  
 Secure payment via PayPal • SSL encrypted  
 </div>  
   
 <PayPalButton   
 amount={currentPrice}   
 description={`${packageName} - Jun Fan JKD Seminar`}  
 isVisible={isOpen && currentPrice > 0}  
 />  
  
 <div className="text-center">  
 <div className="flex items-center justify-center space-x-4 text-xs text-gray-500">  
 <span className="flex items-center">  
 <svg className="w-4 h-4 mr-1 text-green-400" fill="currentColor" viewBox="0 0 20 20">  
 <path fillRule="evenodd" d="M5 9V7a5 5 0 0110 0v2a2 2 0 012 2v5a2 2 0 01-2 2H5a2 2 0 01-2-2v-5a2 2 0 012-2zm8-2v2H7V7a3 3 0 016 0z" clipRule="evenodd" />  
 </svg>  
 Secure  
 </span>  
 <span className="flex items-center">  
 <svg className="w-4 h-4 mr-1 text-blue-400" fill="currentColor" viewBox="0 0 20 20">  
 <path fillRule="evenodd" d="M6.267 3.455a3.066 3.066 0 001.745-.723 3.066 3.066 0 013.976 0 3.066 3.066 0 001.745.723 3.066 3.066 0 012.812 2.812c.051.643.304 1.254.723 1.745a3.066 3.066 0 010 3.976 3.066 3.066 0 00-.723 1.745 3.066 3.066 0 01-2.812 2.812 3.066 3.066 0 00-1.745.723 3.066 3.066 0 01-3.976 0 3.066 3.066 0 00-1.745-.723 3.066 3.066 0 01-2.812-2.812 3.066 3.066 0 00-.723-1.745 3.066 3.066 0 010-3.976 3.066 3.066 0 00.723-1.745 3.066 3.066 0 012.812-2.812zm7.44 5.252a1 1 0 00-1.414-1.414L9 10.586 7.707 9.293a1 1 0 00-1.414 1.414l2 2a1 1 0 001.414 0l4-4z" clipRule="evenodd" />  
 </svg>  
 Verified  
 </span>  
 </div>  
 </div>  
 </div>  
 )}  
 </div>  
</>

) }

export default SideCart