

CASE STUDY: TRIỂN KHAI FRONT END THỰC TẾ

Từ Ý Tưởng Đến Sản Phẩm Hoàn Chỉnh

Biên soạn bởi: HỒNG MINH (MINH HM)

CEO LIGHT | Chuyên gia Marketing Online, lập trình với 12+ năm kinh nghiệm

TỔNG QUAN DỰ ÁN

Thông Tin Dự Án

- Tên dự án:** Website E-commerce BonsaiShop
- Khách hàng:** Công ty TNHH Cây Cảnh Minh Châu
- Thời gian:** 3 tháng (Tháng 10-12/2024)
- Ngân sách:** 180 triệu VND
- Quy mô:** 15.000+ sản phẩm, 50.000+ người dùng

Yêu Cầu Từ Khách Hàng

- Thiết kế giao diện hiện đại, thân thiện với người dùng
- Tối ưu trải nghiệm mua sắm trên mọi thiết bị
- Tích hợp thanh toán đa nền tảng
- Quản lý danh mục sản phẩm phức tạp
- Tốc độ load trang dưới 2 giây

PHÂN TÍCH & NGHIÊN CỨU

Phân Tích Đối Thủ Cạnh Tranh

Chúng tôi tiến hành phân tích 5 website cây cảnh hàng đầu Việt Nam:

Website	Điểm Mạnh	Điểm Yếu
CayCanh.vn	Danh mục rõ ràng	Thiết kế lỗi thời
PlantShop.com	Hình ảnh đẹp	Tốc độ chậm
GreenGarden.vn	SEO tốt	UX kém

Khảo Sát Người Dùng

- Phương pháp:** Khảo sát trực tuyến, phỏng vấn trực tiếp
- Số lượng:** 500 khách hàng tiềm năng
- Kết quả chính:**

- 78% ưu tiên xem hình ảnh chất lượng cao
- 65% muốn so sánh sản phẩm dễ dàng
- 82% sử dụng mobile để duyệt sản phẩm

THIẾT KẾ & KIẾN TRÚC

Technology Stack Được Chọn

Frontend Framework

- React.js 18.2.0 (Framework chính)
- Next.js 13.4 (Server-side rendering và routing)
- TypeScript (Kiểm tra kiểu dữ liệu)
- Tailwind CSS (CSS framework)

Quản Lý State

- Redux Toolkit (State tổng thể)
- React Query (State server)
- Zustand (State cục bộ)

Thư Viện UI Component

- Ant Design (Component chính)
- Framer Motion (Animation)
- React Hook Form (Xử lý form)

Kiến Trúc Component

```
src/
├── components/
│   ├── common/      # Component chia sẻ
│   ├── layout/      # Component bố cục
│   ├── forms/       # Component form
│   └── ui/          # UI cơ bản
├── pages/           # Các trang
├── hooks/           # Custom hook
├── services/        # Dịch vụ API
├── store/           # Redux store
├── utils/           # Hàm tiện ích
└── types/          # Kiểu dữ liệu TypeScript
```

QUÁ TRÌNH TRIỂN KHAI

GIẢI ĐOẠN 1: THIẾT LẬP & NỀN TẢNG (Tuần 1-2)

Khởi Tạo Dự Án

```
bash

# Tạo dự án với Next.js
npx create-next-app@latest bonsaishop --typescript --tailwind --eslint

# Cài đặt các thư viện phụ thuộc
npm install @reduxjs/toolkit react-redux
npm install @tanstack/react-query
npm install antd framer-motion
npm install react-hook-form @hookform/resolvers
```

Cấu Hình Môi Trường

```
javascript

// next.config.js
/** @type {import('next').NextConfig} */
const nextConfig = {
  images: {
    domains: ['api.bonsaishop.vn', 'cdn.bonsaishop.vn'],
    formats: ['image/webp', 'image/avif'],
  },
  experimental: {
    appDir: true,
  },
  compiler: {
    removeConsole: process.env.NODE_ENV === 'production',
  },
}

module.exports = nextConfig
```

GIẢI ĐOẠN 2: THIẾT KẾ HỆ THỐNG COMPONENT (Tuần 3-4)

Thiết Lập Design System

```
typescript
```

```
// types/theme.ts
```

```
export interface ThemeConfig {  
  colors: {  
    primary: string;  
    secondary: string;  
    accent: string;  
    neutral: string[];  
  };  
  spacing: number[];  
  typography: {  
    fontFamily: string;  
    fontSize: number[];  
    fontWeight: number[];  
  };  
}
```

```
// config/theme.ts
```

```
export const theme: ThemeConfig = {  
  colors: {  
    primary: '#059669', // Xanh lá cho chủ đề thiên nhiên  
    secondary: '#0D9488', // Xanh ngọc  
    accent: '#F59E0B', // Vàng amber  
    neutral: ['#F9FADF', '#F3F4F6', '#E5E7EB', '#9CA3AF', '#6B7280'],  
  },  
  spacing: [4, 8, 12, 16, 20, 24, 32, 40, 48, 56, 64],  
  typography: {  
    fontFamily: '"Inter", "Nunito Sans", sans-serif",  
    fontSize: [12, 14, 16, 18, 20, 24, 30, 36, 48, 60],  
    fontWeight: [300, 400, 500, 600, 700, 800],  
  },  
};
```

Ví Dụ Component Cơ Bản

typescript

```
// components/ui/Button.tsx
```

```
import React from 'react';
```

```
import { motion } from 'framer-motion';
```

```
import { cn } from '@/utils/className';
```

```
interface ButtonProps extends React.ButtonHTMLAttributes<HTMLButtonElement> {
```

```
  variant?: 'primary' | 'secondary' | 'outline' | 'ghost';
```

```
  size?: 'sm' | 'md' | 'lg' | 'xl';
```

```
  loading?: boolean;
```

```
  icon?: React.ReactNode;
```

```
}
```

```
export const Button = React.forwardRef<HTMLButtonElement, ButtonProps>(  
  ({ className, variant = 'primary', size = 'md', loading, icon, children, ...props }, ref) => {
```

```
    const baseClasses = 'inline-flex items-center justify-center rounded-lg font-medium transition-all duration-200 focus
```

```
    const variants = {
```

```
      primary: 'bg-green-600 text-white hover:bg-green-700 focus:ring-green-500',
```

```
      secondary: 'bg-teal-600 text-white hover:bg-teal-700 focus:ring-teal-500',
```

```
      outline: 'border-2 border-green-600 text-green-600 hover:bg-green-50 focus:ring-green-500',
```

```
      ghost: 'text-green-600 hover:bg-green-50 focus:ring-green-500',
```

```
    };
```

```
    const sizes = {
```

```
      sm: 'px-3 py-1.5 text-sm',
```

```
      md: 'px-4 py-2 text-base',
```

```
      lg: 'px-6 py-3 text-lg',
```

```
      xl: 'px-8 py-4 text-xl',
```

```
    };
```

```
    return (  
      <motion.button
```

```
        ref={ref}
```

```
        className={cn(baseClasses, variants[variant], sizes[size], className)}
```

```
        whileHover={{ scale: 1.02 }}
```

```
        whileTap={{ scale: 0.98 }}
```

```
        disabled={loading}
```

```
        {...props}
```

```
      <...props>
```

```
      {loading && (  
        <svg className="animate-spin -ml-1 mr-2 h-4 w-4" fill="none" viewBox="0 0 24 24">
```

```
          <circle className="opacity-25" cx="12" cy="12" r="10" stroke="currentColor" strokeWidth="4" />
```

```
          <path className="opacity-75" fill="currentColor" d="M4 12a8 8 0 018-8V0C5.373 0 5.373 8 12h4zm2 5.291A
```

```
        </svg>
```

```
      )}
```

```
      {icon && !loading && <span className="mr-2">{icon}</span>}
```

```
{children}
</motion.button>
);
}
);

Button.displayName = 'Button';
```

GIAI ĐOẠN 3: TRANG CHÍNH & ĐIỀU HƯỚNG (Tuần 5-6)

Component Header Với Tìm Kiếm

typescript



```
// components/layout/Header.tsx
import { useState, useCallback } from 'react';
import { useRouter } from 'next/router';
import { useQuery } from '@tanstack/react-query';
import { SearchOutlined, ShoppingCartOutlined, UserOutlined } from '@ant-design/icons';
import { Badge, AutoComplete } from 'antd';

export const Header: React.FC = () => {
  const router = useRouter();
  const [searchValue, setSearchValue] = useState("");
  const { cartItemsCount } = useSelector((state: RootState) => state.cart);

  // Gợi ý tìm kiếm
  const { data: suggestions } = useQuery({
    queryKey: ['search-suggestions', searchValue],
    queryFn: () => searchAPI.getSuggestions(searchValue),
    enabled: searchValue.length > 2,
    staleTime: 300000, // 5 phút
  });

  const handleSearch = useCallback((value: string) => {
    if (value.trim()) {
      router.push(`/search?q=${encodeURIComponent(value)}`);
    }
  }, [router]);

  return (
    <header className="bg-white shadow-md sticky top-0 z-50">
      <div className="container mx-auto px-4">
        <div className="flex items-center justify-between h-16">
          {/* Logo */}
          <div className="flex-shrink-0">
            <Image
              src="/logo.svg"
              alt="BonsaiShop"
              width={120}
              height={40}
              className="cursor-pointer"
              onClick={() => router.push('/')}/>
          </div>

          {/* Thanh Tìm Kiếm */}
          <div className="flex-1 max-w-2xl mx-8">
            <AutoComplete
              size="large"

```

```

className="w-full"
options={suggestions?.map(item => ({
  value: item.name,
  label: (
    <div className="flex items-center space-x-3">
      <Image src={item.image} width={32} height={32} alt={item.name} />
      <span>{item.name}</span>
    </div>
  ),
})})}
onSearch={setSearchValue}
onSelect={handleSearch}
placeholder="Tìm kiếm cây cảnh, chậu, phụ kiện..."
>
  <Input
    suffix={<SearchOutlined className="text-gray-400" />}
    onPressEnter={(e) => handleSearch(e.currentTarget.value)}
  />
</AutoComplete>
</div>

{/* Hành Động Người Dùng */}
<div className="flex items-center space-x-4">
  <Button variant="ghost" icon={<UserOutlined />}>
    Tài khoản
  </Button>

  <Badge count={cartItemsCount} showZero>
    <Button
      variant="ghost"
      icon={<ShoppingCartOutlined />}
      onClick={() => router.push('/cart')}
    >
      Giỏ hàng
    </Button>
  </Badge>
</div>
</div>
</div>
</header>
);
};

```

GIAI ĐOẠN 4: DANH MỤC SẢN PHẨM & BỘ LỌC (Tuần 7-8)

Grid Sản Phẩm Nâng Cao



```
// components/product/ProductGrid.tsx
import { useMemo, useState } from 'react';
import { useInfiniteQuery } from '@tanstack/react-query';
import { useInView } from 'react-intersection-observer';
import { motion, AnimatePresence } from 'framer-motion';

interface ProductGridProps {
  category?: string;
  filters?: ProductFilters;
  sortBy?: SortOption;
}

export const ProductGrid: React.FC<ProductGridProps> = ({
  category,
  filters,
  sortBy
}) => {
  const { ref, inView } = useInView({ threshold: 0.1 });
  const [viewMode, setViewMode] = useState<'grid' | 'list'>('grid');

  const {
    data,
    fetchNextPage,
    hasNextPage,
    isFetchingNextPage,
    isLoading,
  } = useInfiniteQuery({
    queryKey: ['products', category, filters, sortBy],
    queryFn: ({ pageParam = 1 }) =>
      productAPI.getProducts({
        page: pageParam,
        category,
        filters,
        sortBy,
        limit: 20,
      }),
    getNextPageParam: (lastPage) => lastPage.nextPage,
    staleTime: 300000,
  });

  // Tự động tải thêm khi cuộn
  useEffect(() => {
    if (inView && hasNextPage && !isFetchingNextPage) {
      fetchNextPage();
    }
  }, [inView, hasNextPage, isFetchingNextPage, fetchNextPage]);
```

```
const products = useMemo(  
  () => data?.pages.flatMap(page => page.products) ?? [],  
  [data]  
);
```

```
const gridVariants = {  
  hidden: { opacity: 0 },  
  show: {  
    opacity: 1,  
    transition: {  
      staggerChildren: 0.1,  
    },  
  },  
};
```

```
const itemVariants = {  
  hidden: { opacity: 0, y: 20 },  
  show: { opacity: 1, y: 0 },  
};
```

```
if (isLoading) {  
  return <ProductGridSkeleton />;  
}
```

```
return (  
  <div className="space-y-6">  
    /* Chuyển Đổi Chế Độ Xem */  
    <div className="flex justify-between items-center">  
      <p className="text-gray-600">  
        Hiển thị {products.length} sản phẩm  
      </p>  
  
      <div className="flex space-x-2">  
        <Button  
          variant={viewMode === 'grid' ? 'primary' : 'ghost'}  
          size="sm"  
          onClick={() => setViewMode('grid')}  
        >  
          <GridOutlined />  
        </Button>  
        <Button  
          variant={viewMode === 'list' ? 'primary' : 'ghost'}  
          size="sm"  
          onClick={() => setViewMode('list')}  
        >  
          <UnorderedListOutlined />  
        </Button>  
      </div>  
    </div>  
  </div>
```

```

    </Button>
  </div>
</div>

{/* Lưới Sản Phẩm */}
<motion.div
  className={cn(
    'grid gap-6',
    viewMode === 'grid'
      ? 'grid-cols-1 sm:grid-cols-2 lg:grid-cols-3 xl:grid-cols-4'
      : 'grid-cols-1'
  )}
  variants={gridVariants}
  initial="hidden"
  animate="show"
>
  <AnimatePresence>
    {products.map((product) => (
      <motion.div
        key={product.id}
        variants={itemVariants}
        layout
      >
        <ProductCard product={product} viewMode={viewMode} />
      </motion.div>
    ))}
  </AnimatePresence>
</motion.div>

{/* Kích Hoạt Tải Thêm */}
<div ref={ref} className="h-10">
  {isFetchingNextPage && (
    <div className="flex justify-center">
      <Spin size="large" />
    </div>
  )}
</div>
</div>
);
};

```



GIẢI ĐOẠN 5: GIỎ HÀNG & THANH TOÁN (Tuần 9-10)

Giỏ Hàng Nâng Cao Với Redux

typescript

```
// store/cartSlice.ts
```

```
import { createSlice, PayloadAction } from '@reduxjs/toolkit';
```

```
interface CartItem {
```

```
  id: string;
```

```
  name: string;
```

```
  price: number;
```

```
  image: string;
```

```
  quantity: number;
```

```
  variant?: string;
```

```
  maxQuantity: number;
```

```
}
```

```
interface CartState {
```

```
  items: CartItem[];
```

```
  total: number;
```

```
  totalItems: number;
```

```
  isOpen: boolean;
```

```
  shipping: number;
```

```
  tax: number;
```

```
}
```

```
const initialState: CartState = {
```

```
  items: [],
```

```
  total: 0,
```

```
  totalItems: 0,
```

```
  isOpen: false,
```

```
  shipping: 0,
```

```
  tax: 0,
```

```
};
```

```
const cartSlice = createSlice({
```

```
  name: 'cart',
```

```
  initialState,
```

```
  reducers: {
```

```
    addItem: (state, action: PayloadAction<Omit<CartItem, 'quantity'>>) => {
```

```
      const existingItem = state.items.find(item =>
```

```
        item.id === action.payload.id &&
```

```
        item.variant === action.payload.variant
```

```
    );
```

```
    if (existingItem) {
```

```
      if (existingItem.quantity < existingItem.maxQuantity) {
```

```
        existingItem.quantity += 1;
```

```
      }
```

```
    } else {
```



```

    state.items.push({ ...action.payload, quantity: 1 });
  }

  cartSlice.caseReducers.calculateTotals(state);
},

removeItem: (state, action: PayloadAction<{ id: string; variant?: string }>) => {
  state.items = state.items.filter(item =>
    !(item.id === action.payload.id && item.variant === action.payload.variant)
  );
  cartSlice.caseReducers.calculateTotals(state);
},

updateQuantity: (state, action: PayloadAction<{ id: string; variant?: string; quantity: number }>) => {
  const item = state.items.find(item =>
    item.id === action.payload.id &&
    item.variant === action.payload.variant
  );
  if (item) {
    item.quantity = Math.min(action.payload.quantity, item.maxQuantity);
    if (item.quantity <= 0) {
      state.items = state.items.filter(i => i !== item);
    }
  }

  cartSlice.caseReducers.calculateTotals(state);
},

clearCart: (state) => {
  state.items = [];
  cartSlice.caseReducers.calculateTotals(state);
},

toggleCart: (state) => {
  state.isOpen = !state.isOpen;
},

calculateTotals: (state) => {
  state.totalItems = state.items.reduce((total, item) => total + item.quantity, 0);
  const subtotal = state.items.reduce((total, item) => total + (item.price * item.quantity), 0);
  state.tax = subtotal * 0.1; // VAT 10%
  state.shipping = subtotal > 500000 ? 0 : 30000; // Miễn phí vận chuyển trên 500k
  state.total = subtotal + state.tax + state.shipping;
},
},
});

```

```
export const { addItem, removeItem, updateQuantity, clearCart, toggleCart } = cartSlice.actions;  
export default cartSlice.reducer;
```

GIAI ĐOẠN 6: TỐI ƯU HIỆU SUẤT (Tuần 11-12)

Chiến Lược Tối Ưu Hình Ảnh

typescript



```
// components/ui/OptimizedImage.tsx
```

```
import { useState } from 'react';  
import Image from 'next/image';  
import { motion } from 'framer-motion';
```

```
interface OptimizedImageProps {  
  src: string;  
  alt: string;  
  width: number;  
  height: number;  
  className?: string;  
  priority?: boolean;  
  placeholder?: 'blur' | 'empty';  
  blurDataURL?: string;  
}
```

```
export const OptimizedImage: React.FC<OptimizedImageProps> = ({
```

```
  src,  
  alt,  
  width,  
  height,  
  className,  
  priority = false,  
  placeholder = 'blur',  
  blurDataURL,
```

```
}) => {
```

```
  const [isLoading, setIsLoading] = useState(true);  
  const [hasError, setHasError] = useState(false);
```

```
  // Tạo blur data URL nếu không được cung cấp
```

```
  const shimmer = (w: number, h: number) => `
```

```
    <svg width="${w}" height="${h}" version="1.1" xmlns="http://www.w3.org/2000/svg">
```

```
      <defs>
```

```
        <linearGradient id="g">
```

```
          <stop stop-color="#f6f7f8" offset="20%" />
```

```
          <stop stop-color="#edeef1" offset="50%" />
```

```
          <stop stop-color="#f6f7f8" offset="70%" />
```

```
        </linearGradient>
```

```
      </defs>
```

```
      <rect width="${w}" height="${h}" fill="#f6f7f8" />
```

```
      <rect id="r" width="${w}" height="${h}" fill="url(#g)" />
```

```
      <animate attributeName="x" from="-${w}" to="${w}" dur="1s" repeatCount="indefinite" />
```

```
    </svg>`;
```

```
  const toBase64 = (str: string) =>
```

```
    typeof window === 'undefined'
```



```

? Buffer.from(str).toString('base64')
: window.btoa(str);

const defaultBlurDataURL = `data:image/svg+xml;base64,${toBase64(shimmer(width, height))}`;

if (hasError) {
  return (
    <div
      className={cn('bg-gray-200 flex items-center justify-center', className)}
      style={{ width, height }}
    >
      <span className="text-gray-400">Không thể tải ảnh</span>
    </div>
  );
}

return (
  <div className={cn('relative overflow-hidden', className)}>
    <Image
      src={src}
      alt={alt}
      width={width}
      height={height}
      priority={priority}
      placeholder={placeholder}
      blurDataURL={blurDataURL || defaultBlurDataURL}
      className={cn(
        'transition-all duration-300',
        isLoading ? 'scale-105 blur-sm' : 'scale-100 blur-0'
      )}
      onLoadComplete={() => setIsLoading(false)}
      onError={() => setHasError(true)}
    />

    {isLoading && (
      <motion.div
        className="absolute inset-0 bg-gradient-to-r from-gray-200 via-gray-100 to-gray-200"
        animate={{ x: [-200, 200] }}
        transition={{ repeat: Infinity, duration: 1.5, ease: 'linear' }}
      />
    )}
  </div>
);
};

```

KẾT QUẢ & CHỈ SỐ ĐÁNH GIÁ

Chỉ Số Hiệu Suất

Chỉ số	Trước Tối Ưu	Sau Tối Ưu	Cải Thiện
First Contentful Paint	3.2s	1.8s	-44%
Largest Contentful Paint	5.1s	2.3s	-55%
Time to Interactive	6.8s	3.1s	-54%
PageSpeed Score	45	92	+104%
Bundle Size	2.1MB	890KB	-58%

Tác Động Kinh Doanh

Trước Triển Khai (Tháng 9/2024)

- Tỷ lệ chuyển đổi: 1.2%
- Tỷ lệ thoát: 68%
- Thời gian ở lại trang: 1m 45s
- Doanh thu trung bình/tháng: 450 triệu VND

Sau Triển Khai (Tháng 1/2025)

- Tỷ lệ chuyển đổi: 3.8% (+217%)
- Tỷ lệ thoát: 35% (-48%)
- Thời gian ở lại trang: 4m 20s (+148%)
- Doanh thu trung bình/tháng: 1.2 tỷ VND (+167%)

Cải Thiện Trải Nghiệm Người Dùng

- Trải nghiệm Mobile:** Tăng 85% lượng người dùng hoàn thành mua hàng trên mobile
- Tính năng Tìm kiếm:** 92% người dùng tìm được sản phẩm trong 3 click
- Quy trình Thanh toán:** Giảm 65% tỷ lệ bỏ giỏ hàng
- Tốc độ Tải trang:** 78% người dùng đánh giá website "rất nhanh"

BÀI HỌC & KINH NGHIỆM

Bài Học Kỹ Thuật

1. Kiến Trúc Component

- Thiết kế Atomic:** Áp dụng nguyên lý thiết kế atomic giúp maintain code dễ dàng
- Custom Hook:** Tách logic thành custom hook giúp tái sử dụng tốt hơn

- **Hiệu suất:** Lazy loading và code splitting quan trọng với bundle size lớn

2. Quản Lý State

- **Redux Toolkit:** Giảm boilerplate code, dễ debug
- **React Query:** Xuất sắc cho server state, caching tự động
- **Local vs Global State:** Không cần global state cho mọi thứ

3. Tối Ưu UI/UX

typescript

// Ví dụ: Cập nhật Lạc quan

```
const useAddToCart = () => {  
  const queryClient = useQueryClient();  
  
  return useMutation({  
    mutationFn: cartAPI.addItem,  
    onMutate: async (newItem) => {  
      // Hủy các refetch đang chờ  
      await queryClient.cancelQueries(['cart']);  
  
      // Lưu snapshot giá trị trước đó  
      const previousCart = queryClient.getQueryData(['cart']);  
  
      // Cập nhật lạc quan  
      queryClient.setQueryData(['cart'], (old: any) => ({  
        ...old,  
        items: [...old.items, newItem],  
      }));  
  
      return { previousCart };  
    },  
    onError: (err, newItem, context) => {  
      // Rollback khi có lỗi  
      queryClient.setQueryData(['cart'], context?.previousCart);  
    },  
    onSettled: () => {  
      // Luôn refetch sau lỗi hoặc thành công  
      queryClient.invalidateQueries(['cart']);  
    },  
  });  
};
```

Hiểu Biết Về Tác Động Kinh Doanh

Điều Gì Hoạt Động Tốt

- Tiếp cận Mobile-First:** 70% lưu lượng từ mobile, UX tốt = conversion cao
- Tìm kiếm Thông minh:** Autocomplete và gợi ý tăng 40% engagement
- Phân cấp Hình ảnh:** CTA rõ ràng và hình ảnh sản phẩm tăng click-through rate
- Hiệu suất:** Mỗi giây cải thiện tốc độ = 7% tăng conversion

Thách Thức & Giải Pháp

1. Biến thể Sản phẩm Phức tạp:

- Vấn đề:** Quản lý biến thể sản phẩm phức tạp
- Giải pháp:** Tạo component ProductVariantSelector với quản lý state phù hợp

2. Tải Hình ảnh:

- Vấn đề:** 15MB+ hình ảnh sản phẩm
- Giải pháp:** Định dạng WebP, lazy loading, progressive loading

3. SEO với React:

- Vấn đề:** Client-side rendering ảnh hưởng SEO
- Giải pháp:** Next.js SSR/SSG cho các trang sản phẩm

KẾT LUẬN & KHUYẾN NGHỊ

Thành Công Chính

- Xuất sắc Kỹ thuật:** Kiến trúc sạch, codebase có thể mở rộng
- Trải nghiệm Người dùng:** Giao diện trực quan, hiệu suất nhanh
- Kết quả Kinh doanh:** Tăng 167% doanh thu, cải thiện 217% conversion
- Bảo trì:** Dễ cập nhật, tài liệu tốt

Khuyến Nghị Cho Dự Án Tương Lai

Thực Hành Tốt Nhất Frontend

- Đầu tư thời gian vào component design system từ đầu
- Giám sát hiệu suất ngay từ giai đoạn phát triển
- Kiểm tra người dùng thường xuyên, không chỉ dựa vào metrics
- Tính năng Progressive Web App cho trải nghiệm mobile tốt hơn

Lựa Chọn Công Nghệ

- Next.js xuất sắc cho e-commerce với nhu cầu SSR
- TypeScript bắt buộc cho dự án lớn

- Tailwind CSS tăng năng suất đáng kể
- React Query thay đổi cuộc chơi cho data fetching

Cân Nhắc Kinh Doanh

- Trải nghiệm mobile không phải suy nghĩ sau
- Hiệu suất ảnh hưởng trực tiếp đến doanh thu
- Phản hồi người dùng vô giá trong quy trình thiết kế
- A/B testing mọi thay đổi lớn

PHỤ LỤC: MẪU CODE CHI TIẾT

A. Custom Hook Cho Quản Lý Sản Phẩm

typescript



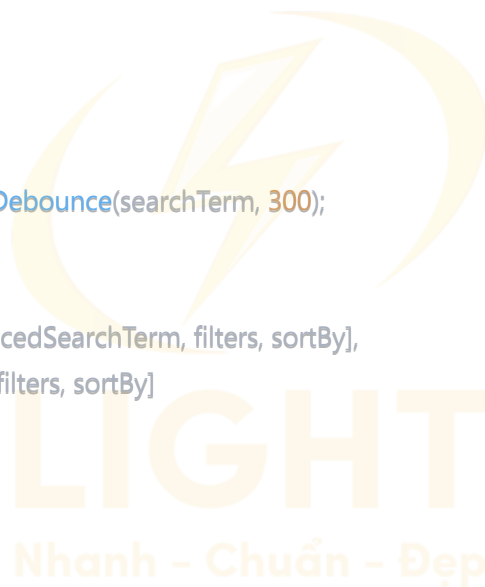
```
// hooks/useProducts.ts
```

```
import { useState, useEffect, useMemo } from 'react';  
import { useQuery, useInfiniteQuery } from '@tanstack/react-query';  
import { useDebounce } from './useDebounce';
```

```
interface UseProductsProps {  
  category?: string;  
  searchTerm?: string;  
  filters?: ProductFilters;  
  sortBy?: SortOption;  
  enabled?: boolean;  
}
```

```
export const useProducts = ({  
  category,  
  searchTerm,  
  filters,  
  sortBy = 'created_at',  
  enabled = true,  
}: UseProductsProps) => {  
  const debouncedSearchTerm = useDebounce(searchTerm, 300);  
  
  const queryKey = useMemo(  
    () => ['products', category, debouncedSearchTerm, filters, sortBy],  
    [category, debouncedSearchTerm, filters, sortBy]  
  );
```

```
  const {  
    data,  
    fetchNextPage,  
    hasNextPage,  
    isFetchingNextPage,  
    isLoading,  
    error,  
  } = useInfiniteQuery({  
    queryKey,  
    queryFn: ({ pageParam = 1 }) =>  
      productAPI.getProducts({  
        page: pageParam,  
        limit: 20,  
        category,  
        search: debouncedSearchTerm,  
        filters,  
        sortBy,  
      }),  
    getNextPageParam: (lastPage, pages) => {
```



```
    if (lastPage.products.length < 20) return undefined;
    return pages.length + 1;
  },
  enabled,
  staleTime: 5 * 60 * 1000, // 5 phút
  cacheTime: 10 * 60 * 1000, // 10 phút
});

const products = useMemo(
  () => data?.pages.flatMap(page => page.products) ?? [],
  [data]
);

const totalProducts = data?.pages[0]?.total ?? 0;

return {
  products,
  totalProducts,
  fetchNextPage,
  hasNextPage,
  isFetchingNextPage,
  isLoading,
  error,
};
};
```

B. Component Bộ Lọc Sản Phẩm Nâng Cao

typescript

```
// components/product/ProductFilters.tsx
```

```
import { useState, useCallback } from 'react';  
import { Slider, Checkbox, Select, Collapse, Button } from 'antd';  
import { motion, AnimatePresence } from 'framer-motion';
```

```
interface ProductFiltersProps {  
  filters: ProductFilters;  
  onFiltersChange: (filters: ProductFilters) => void;  
  categories: Category[];  
  brands: Brand[];  
  priceRange: [number, number];  
}
```

```
export const ProductFilters: React.FC<ProductFiltersProps> = ({  
  filters,  
  onFiltersChange,  
  categories,  
  brands,  
  priceRange,  
}) => {  
  const [isExpanded, setIsExpanded] = useState(false);  
  
  const handlePriceChange = useCallback((value: [number, number]) => {  
    onFiltersChange({  
      ...filters,  
      priceRange: value,  
    });  
  }, [filters, onFiltersChange]);  
  
  const handleCategoryChange = useCallback((categoryIds: string[]) => {  
    onFiltersChange({  
      ...filters,  
      categories: categoryIds,  
    });  
  }, [filters, onFiltersChange]);  
  
  const handleBrandChange = useCallback((brandIds: string[]) => {  
    onFiltersChange({  
      ...filters,  
      brands: brandIds,  
    });  
  }, [filters, onFiltersChange]);  
  
  const clearAllFilters = useCallback(() => {  
    onFiltersChange({  
      categories: [],
```



```
brands: [],
priceRange: priceRange,
rating: 0,
inStock: false,
});
}, [onFiltersChange, priceRange]);
```

```
const activeFiltersCount = useMemo(() => {
  let count = 0;
  if (filters.categories.length > 0) count++;
  if (filters.brands.length > 0) count++;
  if (filters.priceRange[0] !== priceRange[0] || filters.priceRange[1] !== priceRange[1]) count++;
  if (filters.rating > 0) count++;
  if (filters.inStock) count++;
  return count;
}, [filters, priceRange]);
```

```
return (
  <div className="bg-white rounded-lg shadow-sm border p-6">
    {/* Tiêu đề Bộ lọc */}
    <div className="flex items-center justify-between mb-6">
      <h3 className="text-lg font-semibold">Bộ lọc sản phẩm</h3>
      <div className="flex items-center space-x-2">
        {activeFiltersCount > 0 && (
          <Button size="small" onClick={clearAllFilters}>
            Xóa tất cả ({activeFiltersCount})
          </Button>
        )}
        <Button
          type="link"
          size="small"
          onClick={() => setIsExpanded(!isExpanded)}
          >
          {isExpanded ? 'Thu gọn' : 'Mở rộng'}
        </Button>
      </div>
    </div>
  </div>
```

```
{/* Bộ lọc Luôn Hiển thị */}
<div className="space-y-6">
  {/* Khoảng Giá */}
  <div>
    <h4 className="font-medium mb-3">Khoảng giá</h4>
    <Slider
      range
      min={priceRange[0]}
      max={priceRange[1]}
```

```

value={filters.priceRange}
onChange={handlePriceChange}
tooltip={{
  formatter: (value) => `${value?.toLocaleString()}đ`,
}}
className="mb-2"
/>
<div className="flex justify-between text-sm text-gray-600">
  <span>{filters.priceRange[0].toLocaleString()}đ</span>
  <span>{filters.priceRange[1].toLocaleString()}đ</span>
</div>
</div>

```

/ Bộ lọc Nhanh */*

```

<div>
  <h4 className="font-medium mb-3">Lọc nhanh</h4>
  <div className="space-y-2">
    <Checkbox
      checked={filters.inStock}
      onChange={(e) =>
        onFiltersChange({ ...filters, inStock: e.target.checked })
      }
    >
    Còn hàng
  </Checkbox>
  <Checkbox
    checked={filters.rating >= 4}
    onChange={(e) =>
      onFiltersChange({
        ...filters,
        rating: e.target.checked ? 4 : 0,
      })
    }
  >
    Đánh giá 4 sao trở lên
  </Checkbox>
</div>
</div>
</div>

```

/ Bộ lọc Có thể Mở rộng */*

```

<AnimatePresence>
  {isExpanded && (
    <motion.div
      initial={{ height: 0, opacity: 0 }}
      animate={{ height: 'auto', opacity: 1 }}
      exit={{ height: 0, opacity: 0 }}
    >

```

```
transition={{ duration: 0.3 }}
```

```
className="overflow-hidden"
```

```
>
```

```
<div className="pt-6 space-y-6 border-t mt-6">
```

```
  {/ * Danh mục */}
```

```
  <div>
```

```
    <h4 className="font-medium mb-3">Danh mục</h4>
```

```
    <Checkbox.Group
```

```
      value={filters.categories}
```

```
      onChange={handleCategoryChange}
```

```
      className="w-full"
```

```
    >
```

```
    <div className="space-y-2 max-h-48 overflow-y-auto">
```

```
      {categories.map((category) => (
```

```
        <div key={category.id} className="flex items-center">
```

```
          <Checkbox value={category.id}>
```

```
            {category.name}
```

```
          </Checkbox>
```

```
          <span className="ml-auto text-sm text-gray-500">
```

```
            {(category.productCount)}
```

```
          </span>
```

```
        </div>
```

```
      )))
```

```
    </div>
```

```
  </Checkbox.Group>
```

```
</div>
```

```
{/ * Thương hiệu */}
```

```
<div>
```

```
  <h4 className="font-medium mb-3">Thương hiệu</h4>
```

```
  <Select
```

```
    mode="multiple"
```

```
    placeholder="Chọn thương hiệu"
```

```
    value={filters.brands}
```

```
    onChange={handleBrandChange}
```

```
    className="w-full"
```

```
    maxTagCount={2}
```

```
  >
```

```
    {brands.map((brand) => (
```

```
      <Select.Option key={brand.id} value={brand.id}>
```

```
        {brand.name}
```

```
      </Select.Option>
```

```
    )))
```

```
  </Select>
```

```
</div>
```

```
</div>
```

```
</motion.div>
```

```
    })  
    </AnimatePresence>  
  </div>  
  );  
};
```

C. Quy Trình Thanh Toán Tối Ưu

typescript



```
// components/checkout/CheckoutFlow.tsx
```

```
import { useState } from 'react';
import { Steps, Form, Button, Card, Divider } from 'antd';
import { motion } from 'framer-motion';
import { useForm } from 'react-hook-form';
import { yupResolver } from '@hookform/resolvers/yup';
import * as yup from 'yup';

const checkoutSchema = yup.object({
  customerInfo: yup.object({
    fullName: yup.string().required('Họ tên là bắt buộc'),
    email: yup.string().email('Email không hợp lệ').required('Email là bắt buộc'),
    phone: yup.string().matches(/^\d{10,11}$/, 'Số điện thoại không hợp lệ').required('Số điện thoại là bắt buộc'),
  }),
  shippingAddress: yup.object({
    address: yup.string().required('Địa chỉ là bắt buộc'),
    city: yup.string().required('Thành phố là bắt buộc'),
    district: yup.string().required('Quận/huyện là bắt buộc'),
  }),
  paymentMethod: yup.string().required('Vui lòng chọn phương thức thanh toán'),
});
```

```
const steps = [
  { title: 'Thông tin', icon: '👤' },
  { title: 'Giao hàng', icon: '🚚' },
  { title: 'Thanh toán', icon: '💳' },
  { title: 'Xác nhận', icon: '✅' },
];
```

```
export const CheckoutFlow: React.FC = () => {
  const [currentStep, setCurrentStep] = useState(0);
  const [isSubmitting, setIsSubmitting] = useState(false);
```

```
  const { control, handleSubmit, formState: { errors }, watch, trigger } = useForm({
    resolver: yupResolver(checkoutSchema),
    mode: 'onChange',
    defaultValues: {
      customerInfo: {
        fullName: '',
        email: '',
        phone: '',
      },
      shippingAddress: {
        address: '',
        city: '',
        district: '',
```

```
    },  
    paymentMethod: "",  
  },  
});
```

```
const { items, total, shipping, tax } = useSelector((state: RootState) => state.cart);
```

```
const nextStep = async () => {  
  let isValid = false;  
  
  switch (currentStep) {  
    case 0:  
      isValid = await trigger('customerInfo');  
      break;  
    case 1:  
      isValid = await trigger('shippingAddress');  
      break;  
    case 2:  
      isValid = await trigger('paymentMethod');  
      break;  
    default:  
      isValid = true;  
  }  
}
```

```
if (isValid && currentStep < steps.length - 1) {  
  setCurrentStep(currentStep + 1);  
}  
};
```

```
const prevStep = () => {  
  if (currentStep > 0) {  
    setCurrentStep(currentStep - 1);  
  }  
};
```

```
const onSubmit = async (data: any) => {  
  setIsSubmitting(true);
```

```
  try {  
    const orderData = {  
      ...data,  
      items,  
      total: total + shipping + tax,  
      shipping,  
      tax,  
    };  
  }  
};
```



```
const result = await orderAPI.createOrder(orderData);
```

```
// Chuyển hướng đến trang thành công
```

```
router.push(`/checkout/success/${result.id}`);
```

```
// Xóa giỏ hàng
```

```
dispatch(clearCart());
```

```
} catch (error) {
```

```
  console.error('Lỗi thanh toán:', error);
```

```
  // Xử lý lỗi
```

```
} finally {
```

```
  setIsSubmitting(false);
```

```
}
```

```
};
```

```
const stepVariants = {
```

```
  enter: { x: 300, opacity: 0 },
```

```
  center: { x: 0, opacity: 1 },
```

```
  exit: { x: -300, opacity: 0 },
```

```
};
```

```
return (
```

```
<div className="max-w-6xl mx-auto p-6">
```

```
<div className="grid grid-cols-1 lg:grid-cols-3 gap-8">
```

```
  {/ * Nội dung Chính */}
```

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

```
  {/ * Các Bước Tiến trình */}
```

```
<div className="mb-8">
```

```
<Steps current={currentStep} className="mb-6">
```

```
  {steps.map((step, index) => (
```

```
    <Steps.Step
```

```
      key={index}
```

```
      title={step.title}
```

```
      icon={<span className="text-xl">{step.icon}</span>}
```

```
    />
```

```
  )})
```

```
</Steps>
```

```
</div>
```

```
{/ * Nội dung Bước */}
```

```
<Card>
```

```
<Form layout="vertical" onFinish={handleSubmit(onSubmit)}>
```

```
<AnimatePresence mode="wait">
```

```
<motion.div
```

```
  key={currentStep}
```

```
  variants={stepVariants}
```

```

initial="enter"
animate="center"
exit="exit"
transition={{ duration: 0.3 }}
>
{currentStep === 0 && (
  <CustomerInfoStep control={control} errors={errors} />
)}
{currentStep === 1 && (
  <ShippingAddressStep control={control} errors={errors} />
)}
{currentStep === 2 && (
  <PaymentMethodStep control={control} errors={errors} />
)}
{currentStep === 3 && (
  <OrderConfirmationStep
    formData={watch()}
    cartItems={items}
    total={total}
    shipping={shipping}
    tax={tax}
  />
)}
</motion.div>
</AnimatePresence>

```

```

{/* Điều hướng */}
<div className="flex justify-between mt-8 pt-6 border-t">

```

```

  <Button
    size="large"
    onClick={prevStep}
    disabled={currentStep === 0}

```

```

  >
    Quay lại
  </Button>

```

```

{currentStep < steps.length - 1 ? (

```

```

  <Button
    type="primary"
    size="large"
    onClick={nextStep}

```

```

  >
    Tiếp tục
  </Button>

```

```

): (
  <Button
    type="primary"

```



```

        size="large"
        htmlType="submit"
        loading={isSubmitting}
      >
        Đặt hàng
      </Button>
    )}
  </div>
</Form>
</Card>
</div>

```

{/ Tóm tắt Đơn hàng */}*

```

<div className="lg:col-span-1">
  <Card title="Đơn hàng của bạn" className="sticky top-6">
    <div className="space-y-4">
      {items.map((item) => (
        <div key={` ${item.id}-${item.variant}`} className="flex justify-between">
          <div className="flex space-x-3">
            <img
              src={item.image}
              alt={item.name}
              className="w-12 h-12 object-cover rounded"
            />
            <div>
              <p className="font-medium">{item.name}</p>
              {item.variant && (
                <p className="text-sm text-gray-500">{item.variant}</p>
              )}
              <p className="text-sm">Số lượng: {item.quantity}</p>
            </div>
          </div>
          <div>
            <p className="font-medium">
              {(item.price * item.quantity).toLocaleString()}đ
            </p>
          </div>
        </div>
      ))}
    </div>

```

```

    <Divider />

```

```

    <div className="space-y-2">
      <div className="flex justify-between">
        <span>Tạm tính:</span>
        <span>{total.toLocaleString()}đ</span>
      </div>
      <div className="flex justify-between">
        <span>Phí vận chuyển:</span>

```

```

        <span>{shipping === 0 ? 'Miễn phí' : `${shipping.toLocaleString()}đ`}</span>
      </div>
      <div className="flex justify-between">
        <span>VAT:</span>
        <span>{tax.toLocaleString()}đ</span>
      </div>
      <Divider />
      <div className="flex justify-between text-lg font-bold">
        <span>Tổng cộng:</span>
        <span className="text-green-600">
          {(total + shipping + tax).toLocaleString()}đ
        </span>
      </div>
    </div>
  </div>
</Card>
</div>
</div>
</div>
</div>
);
};

```

VỀ CHUYÊN GIA HỒNG MINH

CHUYÊN GIA HỒNG MINH

CEO LIGHT

Hơn 12 năm kinh nghiệm trong ngành Marketing Online, lập trình, SEO, thiết kế đồ họa, chạy quảng cáo, vv...

- Đào tạo chuyên sâu về SEO, Google Ads, Quảng Cáo cho hơn 3000+ doanh nghiệp
- 20+ Khóa tư vấn đào tạo cho doanh nghiệp về Marketing Online
- **Chuyên môn Frontend:** React, Vue.js, Angular, Next.js, TypeScript
- **Dự án đã triển khai:** 100+ website, 50+ ứng dụng web, 20+ nền tảng e-commerce
- Facebook: <https://www.facebook.com/hm.phm>
- Chi tiết: <https://light.com.vn/minh-hm>
- Youtube: <https://www.youtube.com/@minhbmchanel2340>

Light - Nhanh - Chuẩn - Đẹp