# Assignment

## Header.js





import React, { useState } from 'react';

import {

  Navbar, Nav, Container, Badge, Button, NavDropdown

} from 'react-bootstrap';

import { Link, useNavigate } from 'react-router-dom';

import { useSelector, useDispatch } from 'react-redux';

import { logoutUser } from '../redux/slices/authSlice';

import { FaShoppingCart, FaUserCircle, FaSignOutAlt } from 'react-icons/fa';

const Header = () => {

  const user = useSelector(state => state.auth.user);

  const carts = useSelector(state => state.carts.items);

  const dispatch = useDispatch();

  const navigate = useNavigate();

  const cartItemCount = user ? carts.filter(item => item.userId === user.id).length : 0;

  const handleLogout = () => {

    dispatch(logoutUser());

    navigate('/login');

  };

  return (

    <Navbar bg="dark" variant="dark" expand="lg" sticky="top" className="shadow-sm">

      <Container fluid="lg">

        <Navbar.Brand as={Link} to="/" className="d-flex align-items-center">

          <span className="fw-bold">Fashion Shop</span>

        </Navbar.Brand>

        <Navbar.Toggle aria-controls="main-navbar" />

        <Navbar.Collapse id="main-navbar">

          <Nav className="me-auto">

            <Nav.Link as={Link} to="/" className="px-3">Trang Chủ</Nav.Link>

          </Nav>

          <Nav className="align-items-center">

            {user ? (

              <>

                <Nav.Link as={Link} to="/my-account" className="px-3">My Account</Nav.Link>

                <Nav.Link as={Link} to="/checkout" className="px-3">Checkout</Nav.Link>

                <Nav.Link as={Link} to="/wishlist" className="px-3">Wishlist</Nav.Link>

                <Nav.Link as={Link} to="/cart" className="position-relative px-3">

                  <FaShoppingCart size={20} />

                  {cartItemCount > 0 && (

                    <Badge pill bg="danger" className="position-absolute top-0 start-100 translate-middle" style={{ fontSize: '0.6rem' }}>

                      {cartItemCount}

                    </Badge>

                  )}

                </Nav.Link>

                <NavDropdown title={user.name || user.username} id="user-dropdown" align="end">

                  <NavDropdown.Item as={Link} to="/my-account">Account</NavDropdown.Item>

                  <NavDropdown.Item as={Link} to="/wishlist">Wishlist</NavDropdown.Item>

                  <NavDropdown.Item onClick={handleLogout}>Logout <FaSignOutAlt className="ms-1" /></NavDropdown.Item>

                </NavDropdown>

              </>

            ) : (

              <>

                <Nav.Link as={Link} to="/register" className="px-3">Register</Nav.Link>

                <Nav.Link as={Link} to="/login" className="px-3">Sign in</Nav.Link> {/\* Khi chưa login \*/}

                <Nav.Link as={Link} to="/checkout" className="px-3">Checkout</Nav.Link>

                <Nav.Link as={Link} to="/wishlist" className="px-3">Wishlist</Nav.Link>

                <Nav.Link as={Link} to="/cart" className="position-relative px-3">

                  <FaShoppingCart size={20} />

                  {cartItemCount > 0 && (

                    <Badge pill bg="danger" className="position-absolute top-0 start-100 translate-middle" style={{ fontSize: '0.6rem' }}>

                      {cartItemCount}

                    </Badge>

                  )}

                </Nav.Link>

              </>

            )}

          </Nav>

        </Navbar.Collapse>

      </Container>

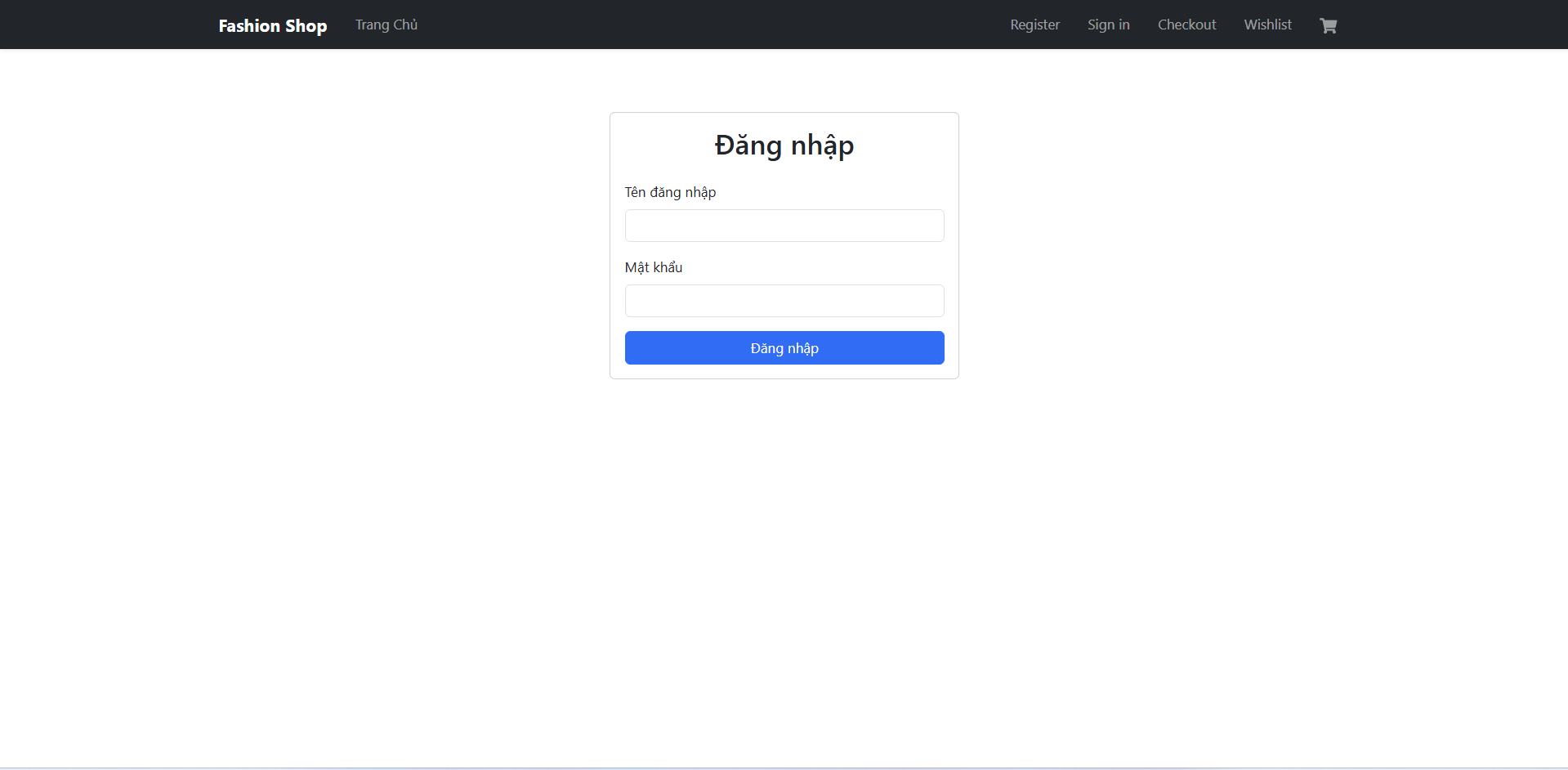
    </Navbar>

  );

};

export default Header;

## Login.js



import React, { useState } from 'react';

import { useNavigate, useLocation } from 'react-router-dom';

import { useDispatch } from 'react-redux';

import { loginUser } from '../redux/slices/authSlice';

import { Form, Button, Container, Card, Alert, Toast } from 'react-bootstrap';

const Login = () => {

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

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

  const [showError, setShowError] = useState(false);

  const dispatch = useDispatch();

  const navigate = useNavigate();

  const location = useLocation();

  const from = location.state?.from || '/';

  const handleSubmit = async (e) => {

    e.preventDefault();

    try {

      await dispatch(loginUser({ username, password })).unwrap();

      navigate(from, { replace: true });

      setShowError(false);

    } catch (err) {

      setShowError(true);

      setTimeout(() => setShowError(false), 3000);

    }

  };

  return (

    <Container className="mt-5">

      <Card className="mx-auto" style={{ maxWidth: '400px' }}>

        <Card.Body>

          <h2 className="text-center mb-4">Đăng nhập</h2>

          <Form onSubmit={handleSubmit}>

            <Form.Group className="mb-3">

              <Form.Label>Tên đăng nhập</Form.Label>

              <Form.Control

                type="text"

                value={username}

                onChange={e => setUsername(e.target.value)}

              />

            </Form.Group>

            <Form.Group className="mb-3">

              <Form.Label>Mật khẩu</Form.Label>

              <Form.Control

                type="password"

                value={password}

                onChange={e => setPassword(e.target.value)}

              />

            </Form.Group>

            <Button variant="primary" type="submit" className="w-100">

              Đăng nhập

            </Button>

          </Form>

          {showError && (

            <Toast

              show={showError}

              onClose={() => setShowError(false)}

              delay={3000}

              autohide

              style={{ position: 'absolute', top: '10px', right: '10px' }}

            >

              <Toast.Header>

                <strong className="me-auto">Error</strong>

              </Toast.Header>

              <Toast.Body>Đăng nhập thất bại. Vui lòng kiểm tra lại!</Toast.Body>

            </Toast>

          )}

        </Card.Body>

      </Card>

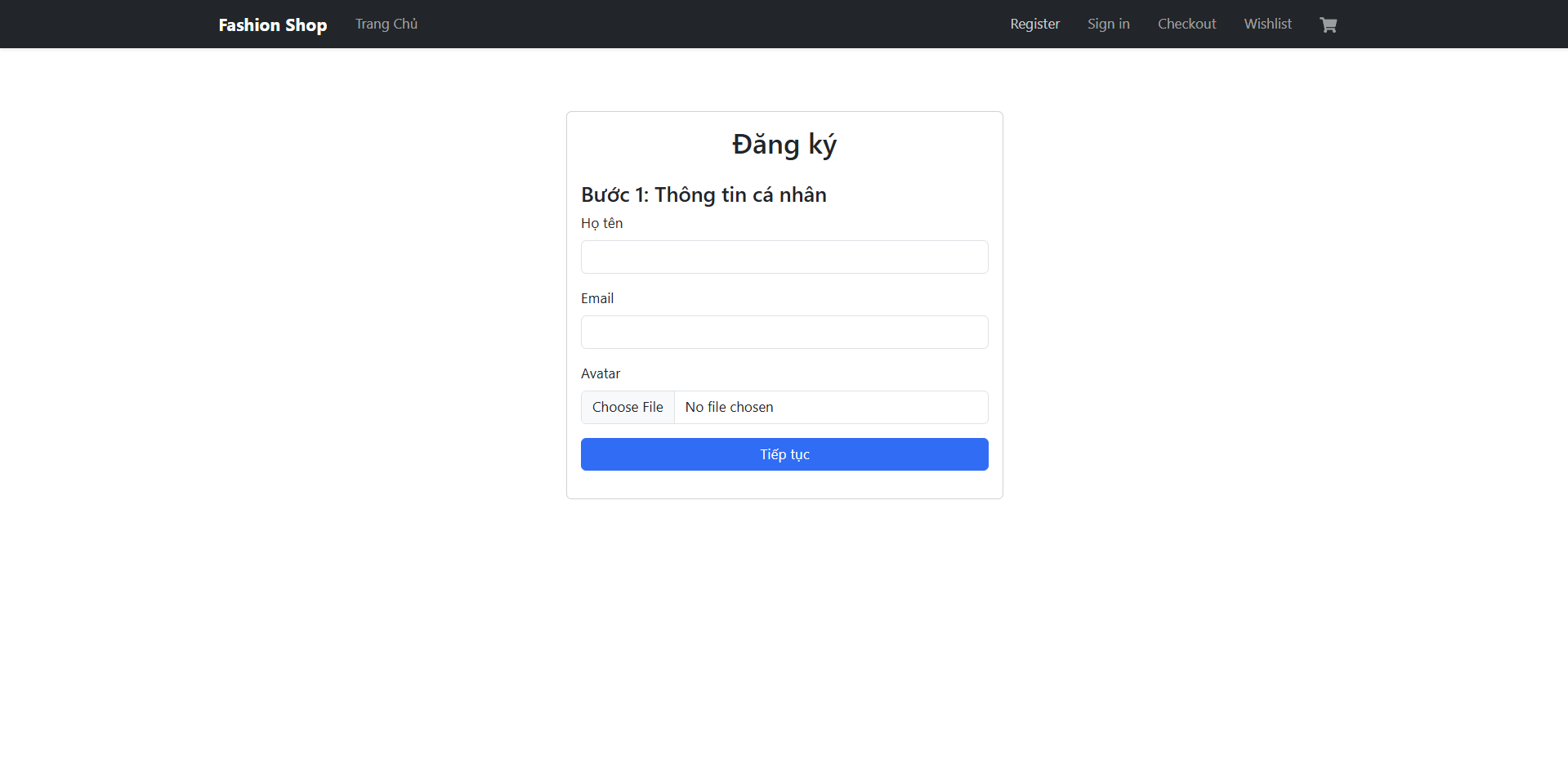
    </Container>

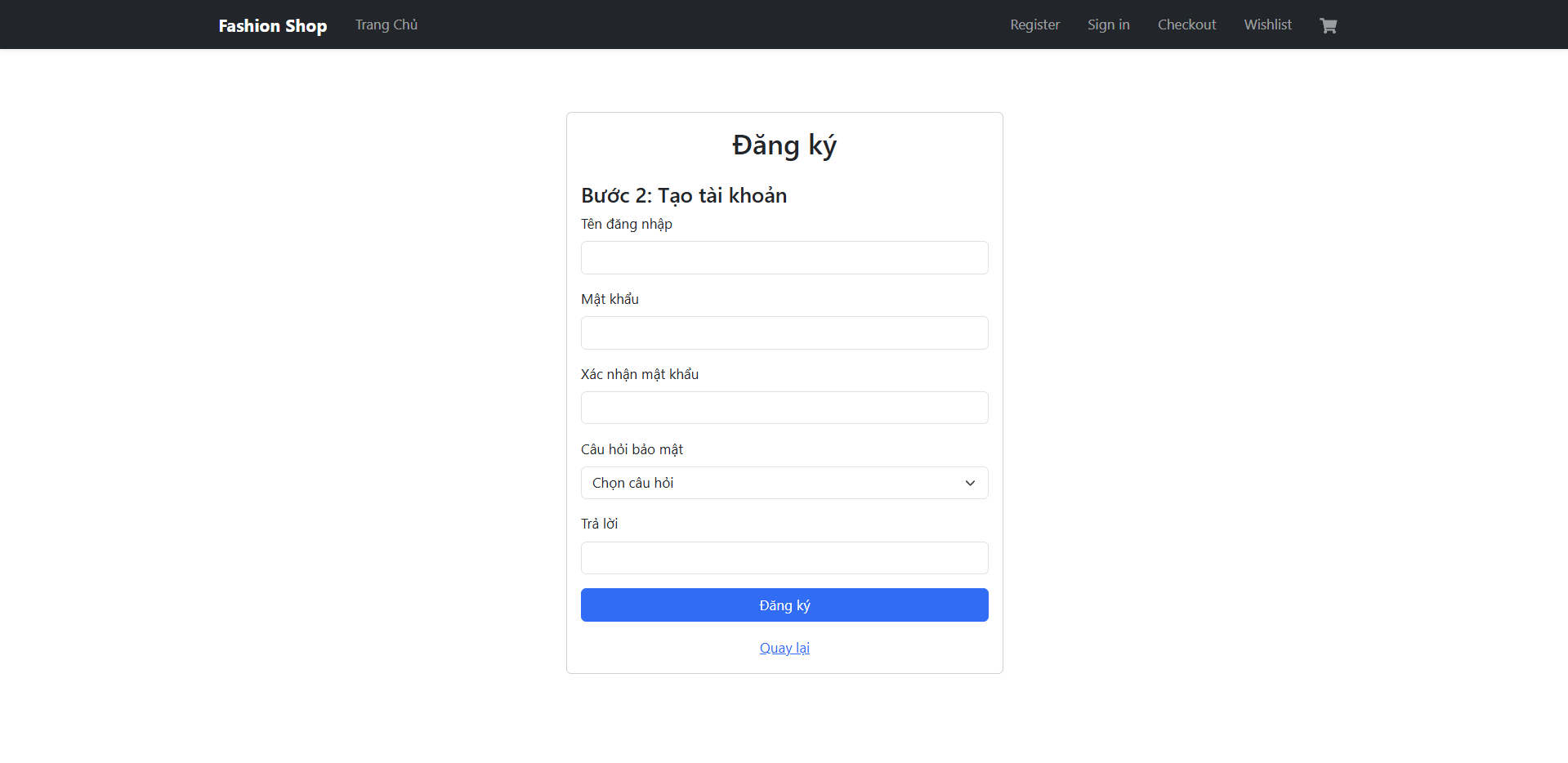
  );

};

export default Login;

## Register.js





import React, { useState } from 'react';

import { useNavigate } from 'react-router-dom';

import { useDispatch, useSelector } from 'react-redux';

import { registerUser } from '../redux/slices/authSlice';

import { Form, Button, Container, Card, Alert, Spinner } from 'react-bootstrap'; // Xóa Row, Col

const Register = () => {

  const [step, setStep] = useState(1);

  const [formData, setFormData] = useState({

    fullName: '',

    email: '',

    avatar: null,

    username: '',

    password: '',

    confirmPassword: '',

    secretQuestion: '',

    secretAnswer: ''

  });

  const [validationErrors, setValidationErrors] = useState({});

  const navigate = useNavigate();

  const dispatch = useDispatch();

  const { status, error } = useSelector((state) => state.auth);

  const isLoading = status === 'loading';

  const handleChange = (e) => {

    const { name, value, files } = e.target;

    if (name === 'avatar' && files[0]) {

      if (files[0].size > 2 \* 1024 \* 1024) {

        setValidationErrors({ ...validationErrors, avatar: 'File phải ≤ 2MB' });

        return;

      }

      if (!['image/jpeg', 'image/png'].includes(files[0].type)) {

        setValidationErrors({ ...validationErrors, avatar: 'Chỉ chấp nhận jpg/png' });

        return;

      }

      setFormData({ ...formData, [name]: files[0] });

    } else {

      setFormData({ ...formData, [name]: value });

    }

    if (validationErrors[name]) {

      setValidationErrors({ ...validationErrors, [name]: null });

    }

  };

  const validateForm = () => {

    const errors = {};

    if (step === 1) {

      if (!formData.fullName.trim()) errors.fullName = 'Vui lòng nhập họ tên';

      if (!formData.email.trim()) errors.email = 'Vui lòng nhập email';

      if (!formData.avatar) errors.avatar = 'Vui lòng chọn avatar';

    } else {

      if (!formData.username.trim() || formData.username.length < 4) errors.username = 'Tên đăng nhập phải ≥ 4 ký tự';

      if (!formData.password || formData.password.length < 6) errors.password = 'Mật khẩu phải ≥ 6 ký tự';

      if (formData.password !== formData.confirmPassword) errors.confirmPassword = 'Mật khẩu không khớp';

      if (!formData.secretQuestion) errors.secretQuestion = 'Vui lòng chọn câu hỏi bảo mật';

      if (!formData.secretAnswer.trim()) errors.secretAnswer = 'Vui lòng nhập câu trả lời';

    }

    setValidationErrors(errors);

    return Object.keys(errors).length === 0;

  };

  const handleSubmit = async (e) => {

    e.preventDefault();

    if (!validateForm()) return;

    if (step === 1) {

      setStep(2);

      return;

    }

    const formDataToSend = new FormData();

    Object.keys(formData).forEach(key => formDataToSend.append(key, formData[key]));

    formDataToSend.append('id', Date.now().toString());

    formDataToSend.append('role', 'user');

    try {

      await dispatch(registerUser(formDataToSend)).unwrap();

      navigate('/');

    } catch (err) {

      console.error('Registration failed:', err);

    }

  };

  return (

    <Container className="mt-5">

      <Card className="mx-auto" style={{ maxWidth: '500px' }}>

        <Card.Body>

          <h2 className="text-center mb-4">Đăng ký</h2>

          <Form onSubmit={handleSubmit}>

            {step === 1 ? (

              <>

                <h4>Bước 1: Thông tin cá nhân</h4>

                <Form.Group className="mb-3">

                  <Form.Label>Họ tên</Form.Label>

                  <Form.Control

                    type="text"

                    name="fullName"

                    value={formData.fullName}

                    onChange={handleChange}

                    isInvalid={!!validationErrors.fullName}

                  />

                  <Form.Control.Feedback type="invalid">{validationErrors.fullName}</Form.Control.Feedback>

                </Form.Group>

                <Form.Group className="mb-3">

                  <Form.Label>Email</Form.Label>

                  <Form.Control

                    type="email"

                    name="email"

                    value={formData.email}

                    onChange={handleChange}

                    isInvalid={!!validationErrors.email}

                  />

                  <Form.Control.Feedback type="invalid">{validationErrors.email}</Form.Control.Feedback>

                </Form.Group>

                <Form.Group className="mb-3">

                  <Form.Label>Avatar</Form.Label>

                  <Form.Control

                    type="file"

                    name="avatar"

                    onChange={handleChange}

                    isInvalid={!!validationErrors.avatar}

                  />

                  <Form.Control.Feedback type="invalid">{validationErrors.avatar}</Form.Control.Feedback>

                </Form.Group>

              </>

            ) : (

              <>

                <h4>Bước 2: Tạo tài khoản</h4>

                <Form.Group className="mb-3">

                  <Form.Label>Tên đăng nhập</Form.Label>

                  <Form.Control

                    type="text"

                    name="username"

                    value={formData.username}

                    onChange={handleChange}

                    isInvalid={!!validationErrors.username}

                  />

                  <Form.Control.Feedback type="invalid">{validationErrors.username}</Form.Control.Feedback>

                </Form.Group>

                <Form.Group className="mb-3">

                  <Form.Label>Mật khẩu</Form.Label>

                  <Form.Control

                    type="password"

                    name="password"

                    value={formData.password}

                    onChange={handleChange}

                    isInvalid={!!validationErrors.password}

                  />

                  <Form.Control.Feedback type="invalid">{validationErrors.password}</Form.Control.Feedback>

                </Form.Group>

                <Form.Group className="mb-3">

                  <Form.Label>Xác nhận mật khẩu</Form.Label>

                  <Form.Control

                    type="password"

                    name="confirmPassword"

                    value={formData.confirmPassword}

                    onChange={handleChange}

                    isInvalid={!!validationErrors.confirmPassword}

                  />

                  <Form.Control.Feedback type="invalid">{validationErrors.confirmPassword}</Form.Control.Feedback>

                </Form.Group>

                <Form.Group className="mb-3">

                  <Form.Label>Câu hỏi bảo mật</Form.Label>

                  <Form.Select

                    name="secretQuestion"

                    value={formData.secretQuestion}

                    onChange={handleChange}

                    isInvalid={!!validationErrors.secretQuestion}

                  >

                    <option value="">Chọn câu hỏi</option>

                    <option value="pet">Tên thú cưng đầu tiên của bạn là gì?</option>

                    <option value="school">Trường tiểu học của bạn là gì?</option>

                  </Form.Select>

                  <Form.Control.Feedback type="invalid">{validationErrors.secretQuestion}</Form.Control.Feedback>

                </Form.Group>

                <Form.Group className="mb-3">

                  <Form.Label>Trả lời</Form.Label>

                  <Form.Control

                    type="text"

                    name="secretAnswer"

                    value={formData.secretAnswer}

                    onChange={handleChange}

                    isInvalid={!!validationErrors.secretAnswer}

                  />

                  <Form.Control.Feedback type="invalid">{validationErrors.secretAnswer}</Form.Control.Feedback>

                </Form.Group>

              </>

            )}

            <Button

              variant="primary"

              type="submit"

              className="w-100 mb-3"

              disabled={isLoading}

            >

              {isLoading ? (

                <>

                  <Spinner as="span" animation="border" size="sm" role="status" aria-hidden="true" className="me-2" />

                  Đang xử lý...

                </>

              ) : step === 1 ? 'Tiếp tục' : 'Đăng ký'}

            </Button>

            {step > 1 && (

              <div className="text-center">

                <Button variant="link" onClick={() => setStep(1)} className="p-0">Quay lại</Button>

              </div>

            )}

            {error && <Alert variant="danger">{error}</Alert>}

          </Form>

        </Card.Body>

      </Card>

    </Container>

  );

};

export default Register;

## CarouselComponent.js



import React from 'react';

import Carousel from 'react-bootstrap/Carousel';

const CarouselComponent = () => {

  return (

    <Carousel interval={3000} pause="hover"> {/\* Autoplay 3s, pause on hover \*/}

      <Carousel.Item>

        <img

          src="/images/banner1.jpg"

          alt="Banner 1 - Phụ nữ cầm túi mua sắm"

          style={{ width: '100%', height: '250px', objectFit: 'cover' }}

        />

        <Carousel.Caption>

          <h3>Chào mừng đến với mua sắm</h3>

          <p>Khám phá phong cách thời trang của bạn!</p>

        </Carousel.Caption>

      </Carousel.Item>

      <Carousel.Item>

        <img

          src="/images/banner2.jpg"

          alt="Banner 2 - Trendy Fashion 50% off"

          style={{ width: '100%', height: '250px', objectFit: 'cover' }}

        />

        <Carousel.Caption>

          <h3>Giảm giá lên đến 50%</h3>

          <p>Đừng bỏ lỡ cơ hội mua sắm tuyệt vời!</p>

        </Carousel.Caption>

      </Carousel.Item>

      <Carousel.Item>

        <img

          src="/images/banner3.png"

          alt="Banner 3 - Brand Sale"

          style={{ width: '100%', height: '250px', objectFit: 'cover' }}

        />

        <Carousel.Caption>

          <h3>Thời trang thương hiệu</h3>

          <p>Khám phá các sản phẩm cao cấp ngay hôm nay.</p>

        </Carousel.Caption>

      </Carousel.Item>

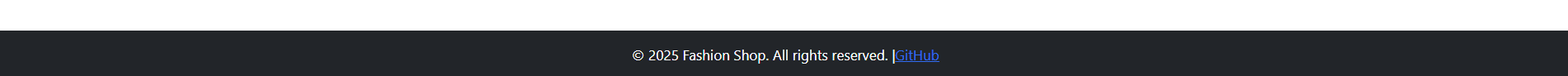
    </Carousel>

  );

};

export default CarouselComponent;

## Footer.js



import React from 'react';

import { Container } from 'react-bootstrap';

const Footer = () => {

  return (

    <footer className="bg-dark text-white text-center py-3 mt-5">

      <Container>

        <p className="mb-0">

          © 2025 Fashion Shop. All rights reserved. |

          <a href="https://github.com/Vnganm/Block3\_FER202" target="\_blank" rel="noopener noreferrer">GitHub</a>

        </p>

      </Container>

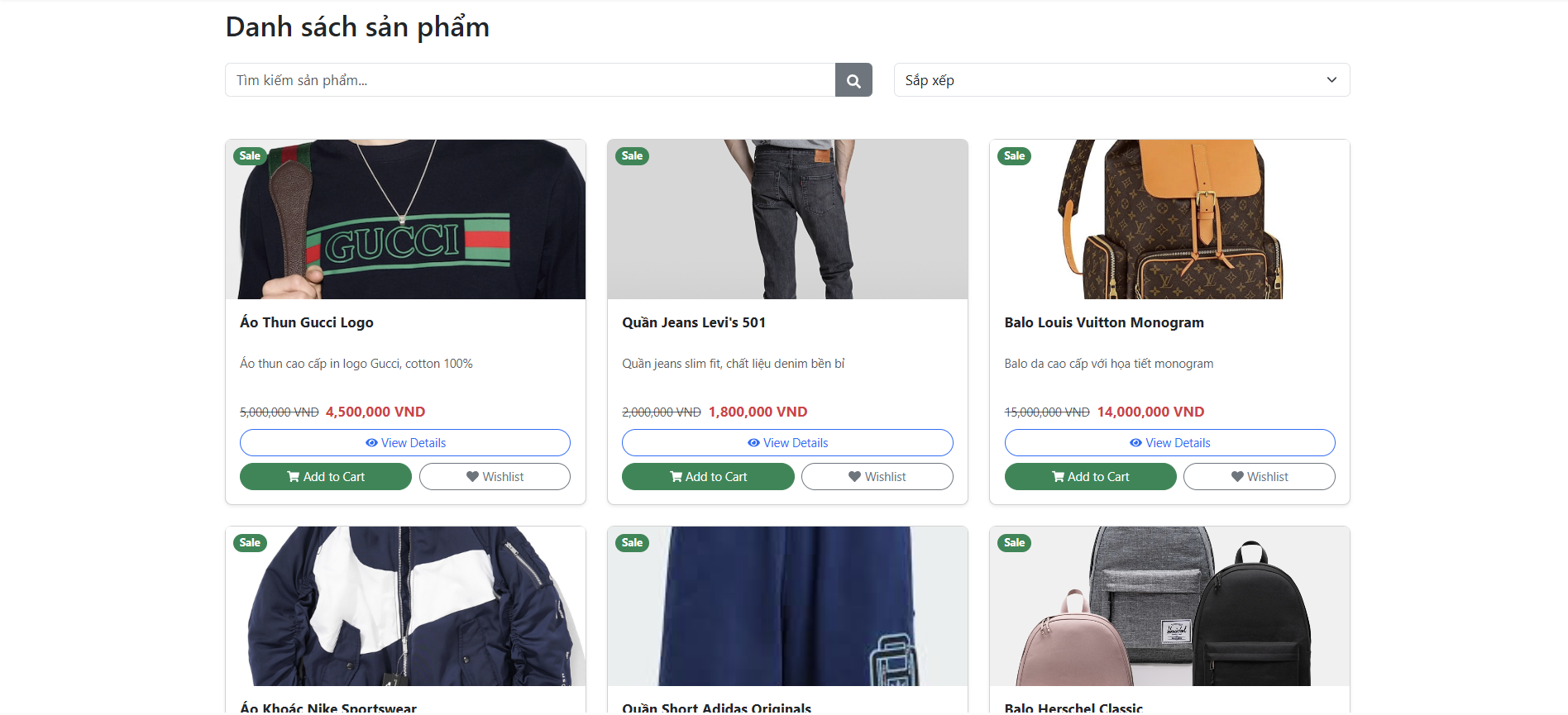
    </footer>

  );

};

export default Footer;

## ProductList.js



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

import { useSelector, useDispatch } from 'react-redux';

import { fetchProducts, addProduct, deleteProduct } from '../redux/slices/productSlice';

import Footer from '../components/Footer';

import { addToCart } from '../redux/slices/cartSlice';

import { Link } from 'react-router-dom';

import CarouselComponent from './CarouselComponent';

import ProductCard from './ProductCard';

import {

  Card,

  Button,

  Row,

  Col,

  Form,

  Container,

  Alert,

  Toast,

  InputGroup

} from 'react-bootstrap';

import { FaSearch, FaTrash, FaEdit, FaPlus } from 'react-icons/fa';

const ProductList = () => {

  const dispatch = useDispatch();

  const { items: products, error } = useSelector(state => state.products);

  const user = useSelector(state => state.auth.user);

  const [newProduct, setNewProduct] = useState({

    name: '',

    description: '',

    price: '',

    currentPrice: '',

    image: '',

    quantity: 0

  });

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

  const [filteredProducts, setFilteredProducts] = useState([]);

  const [previewImage, setPreviewImage] = useState(null);

  const [showSuccess, setShowSuccess] = useState(false);

  const [showCartAlert, setShowCartAlert] = useState(false);

  const [sortOption, setSortOption] = useState('default');

  useEffect(() => {

    dispatch(fetchProducts());

  }, [dispatch]);

  useEffect(() => {

    let tempProducts = products || [];

    if (searchTerm.trim() !== '') {

      tempProducts = tempProducts.filter(product =>

        product.name.toLowerCase().includes(searchTerm.toLowerCase()) ||

        product.description.toLowerCase().includes(searchTerm.toLowerCase())

      );

    }

    if (sortOption === 'name-asc') {

      tempProducts = [...tempProducts].sort((a, b) => a.name.localeCompare(b.name));

    } else if (sortOption === 'price-asc') {

      tempProducts = [...tempProducts].sort((a, b) => parseFloat(a.currentPrice) - parseFloat(b.currentPrice));

    } else if (sortOption === 'price-desc') {

      tempProducts = [...tempProducts].sort((a, b) => parseFloat(b.currentPrice) - parseFloat(a.currentPrice));

    }

    setFilteredProducts(tempProducts);

  }, [products, searchTerm, sortOption]);

  const handleSearch = (e) => {

    const value = e.target.value;

    setSearchTerm(value);

  };

  const handleAdd = (e) => {

    e.preventDefault();

    dispatch(addProduct({ ...newProduct, id: Date.now() }));

    setNewProduct({ name: '', description: '', price: '', currentPrice: '', image: '', quantity: 0 });

    setPreviewImage(null);

    setShowSuccess(true);

    setTimeout(() => setShowSuccess(false), 3000);

  };

  const handleDelete = (id) => {

    if (window.confirm('Bạn có chắc muốn xóa sản phẩm này?')) {

      dispatch(deleteProduct(id));

    }

  };

  const handleImageUpload = (e) => {

    const file = e.target.files[0];

    if (file.size > 2 \* 1024 \* 1024) {

      alert('File phải ≤ 2MB');

      return;

    }

    if (!['image/jpeg', 'image/png'].includes(file.type)) {

      alert('Chỉ chấp nhận jpg/png');

      return;

    }

    setNewProduct({ ...newProduct, image: file });

    setPreviewImage(URL.createObjectURL(file));

  };

  return (

    <>

      {/\* Carousel bên ngoài Container để full width \*/}

      <div className="full-width-carousel">

        <CarouselComponent />

      </div>

      {/\* Phần nội dung chính vẫn trong Container \*/}

      <Container>

        <div className="my-4">

          <h2>Danh sách sản phẩm</h2>

          <Row className="mb-5 mt-4">

            <Col md={7}>

              <InputGroup>

                <Form.Control

                  type="text"

                  placeholder="Tìm kiếm sản phẩm..."

                  value={searchTerm}

                  onChange={handleSearch}

                />

                <Button variant="secondary"><FaSearch /></Button>

              </InputGroup>

            </Col>

            <Col md={5}>

              <Form.Select value={sortOption} onChange={e => setSortOption(e.target.value)}>

                <option value="default">Sắp xếp</option>

                <option value="name-asc">Name A→Z</option>

                <option value="price-asc">Price Ascending</option>

                <option value="price-desc">Price Descending</option>

              </Form.Select>

            </Col>

          </Row>

          <Row xs={1} sm={2} md={3} className="g-4">

            {filteredProducts.map(product => (

              <Col key={product.id}>

                <ProductCard product={product} />

              </Col>

            ))}

          </Row>

          {user?.role === 'admin' && (

            <div className="mt-4">

              <h3>Thêm sản phẩm mới</h3>

              <Form onSubmit={handleAdd}>

                <Row>

                  <Col lg={4} md={6} sm={12}>

                    <Form.Group className="mb-3">

                      <Form.Label>Tên</Form.Label>

                      <Form.Control

                        type="text"

                        value={newProduct.name}

                        onChange={e => setNewProduct({ ...newProduct, name: e.target.value })}

                        required

                      />

                    </Form.Group>

                  </Col>

                  <Col md={6}>

                    <Form.Group className="mb-3">

                      <Form.Label>Mô tả</Form.Label>

                      <Form.Control

                        type="text"

                        value={newProduct.description}

                        onChange={e => setNewProduct({ ...newProduct, description: e.target.value })}

                        required

                      />

                    </Form.Group>

                  </Col>

                </Row>

                <Row>

                  <Col md={6}>

                    <Form.Group className="mb-3">

                      <Form.Label>Giá gốc</Form.Label>

                      <Form.Control

                        type="number"

                        value={newProduct.price}

                        onChange={e => setNewProduct({ ...newProduct, price: e.target.value })}

                        required

                      />

                    </Form.Group>

                  </Col>

                  <Col md={6}>

                    <Form.Group className="mb-3">

                      <Form.Label>Giá hiện tại</Form.Label>

                      <Form.Control

                        type="number"

                        value={newProduct.currentPrice}

                        onChange={e => setNewProduct({ ...newProduct, currentPrice: e.target.value })}

                        required

                      />

                    </Form.Group>

                  </Col>

                </Row>

                <Row>

                  <Col md={6}>

                    <Form.Group className="mb-3">

                      <Form.Label>Hình ảnh sản phẩm</Form.Label>

                      <Form.Control

                        type="file"

                        accept="image/\*"

                        onChange={handleImageUpload}

                        required

                      />

                      {previewImage && (

                        <div className="mt-2">

                          <img

                            src={previewImage}

                            alt="Preview"

                            style={{ maxWidth: '100%', maxHeight: '200px' }}

                            className="img-thumbnail"

                          />

                        </div>

                      )}

                    </Form.Group>

                  </Col>

                  <Col md={6}>

                    <Form.Group className="mb-3">

                      <Form.Label>Số lượng</Form.Label>

                      <Form.Control

                        type="number"

                        value={newProduct.quantity}

                        onChange={e => setNewProduct({ ...newProduct, quantity: parseInt(e.target.value) || 0 })}

                        required

                      />

                    </Form.Group>

                  </Col>

                </Row>

                <Button variant="success" type="submit" className="w-100 py-2">

                  <FaPlus className="me-2" /> Thêm sản phẩm

                </Button>

              </Form>

            </div>

          )}

        </div>

        <Toast

          show={showSuccess}

          onClose={() => setShowSuccess(false)}

          delay={3000}

          autohide

          style={{ position: 'absolute', top: '20px', right: '20px' }}

        >

          <Toast.Header>

            <strong className="me-auto">Thành công</strong>

          </Toast.Header>

          <Toast.Body>Thêm sản phẩm thành công!</Toast.Body>

        </Toast>

        <Toast

          show={showCartAlert}

          onClose={() => setShowCartAlert(false)}

          delay={3000}

          autohide

          style={{ position: 'absolute', top: '70px', right: '20px' }}

        >

          <Toast.Header>

            <strong className="me-auto">Thành công</strong>

          </Toast.Header>

          <Toast.Body>{user ? 'Thêm vào giỏ hàng thành công!' : 'Thêm vào wishlist thành công!'}</Toast.Body>

        </Toast>

        {error && <Alert variant="danger">{error}</Alert>}

      </Container>

      {/\* Footer bên ngoài Container để full width \*/}

      <div className="full-width-footer">

        <Footer />

      </div>

      {/\* Thêm CSS để đảm bảo full width \*/}

      <style jsx>{`

        .full-width-carousel,

        .full-width-footer {

          width: 100vw;

          position: relative;

          left: 50%;

          right: 50%;

          margin-left: -50vw;

          margin-right: -50vw;

        }

      `}</style>

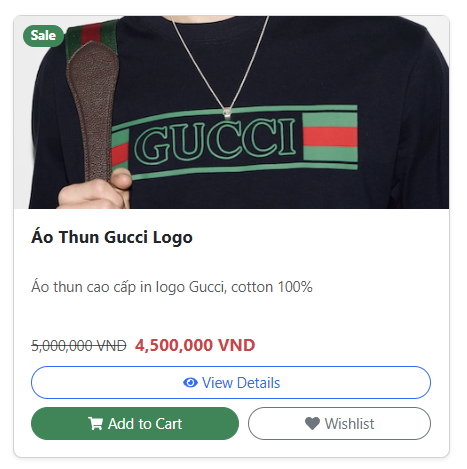
    </>

  );

};

export default ProductList;

## ProductCard.js



import React, { useState } from 'react';

import { useSelector, useDispatch } from 'react-redux';

import { Link, useLocation } from 'react-router-dom';

import { addToCart, addToWishlist } from '../redux/slices/cartSlice';

import { Card, Button, Badge, Toast, ToastContainer } from 'react-bootstrap';

import { FaShoppingCart, FaHeart, FaEye, FaInfoCircle, FaExclamationTriangle } from 'react-icons/fa';

const ProductCard = ({ product }) => {

  const dispatch = useDispatch();

  const user = useSelector(state => state.auth.user);

  const location = useLocation();

  const isInWishlist = user?.wishlist?.includes(product.id);

  const [showToast, setShowToast] = useState(false);

  const [toastConfig, setToastConfig] = useState({ type: 'success', message: '', title: '' });

  const showToastMessage = (type, title, message) => {

    setToastConfig({ type, title, message });

    setShowToast(true);

    setTimeout(() => setShowToast(false), 3000);

  };

  const handleAddToCart = () => {

    if (product.quantity > 0) {

      dispatch(addToCart(product))

        .then(() => showToastMessage('success', 'Success', 'Added to cart!'))

        .catch(() => showToastMessage('danger', 'Error', 'Failed to add to cart!'));

    } else {

      showToastMessage('danger', 'Error', 'Out of stock!');

    }

  };

  const handleWishlistAction = () => {

    if (!user) {

      showToastMessage('info', 'Information', 'Please sign in to save wishlist');

      const redirectUri = `${location.pathname}${location.search}`;

      setTimeout(() => window.location.href = `/login?redirect\_uri=${encodeURIComponent(redirectUri)}`, 1500);

      return;

    }

    if (!isInWishlist) {

      dispatch(addToWishlist(product.id))

        .then(() => showToastMessage('success', 'Success', 'Added to wishlist!'))

        .catch(() => showToastMessage('danger', 'Error', 'Failed to add to wishlist!'));

    }

  };

  const getToastIcon = () => {

    switch (toastConfig.type) {

      case 'success': return <FaHeart className="me-2" />;

      case 'info': return <FaInfoCircle className="me-2" />;

      case 'danger': return <FaExclamationTriangle className="me-2" />;

      default: return <FaInfoCircle className="me-2" />;

    }

  };

  const getToastBgStyle = () => {

    switch (toastConfig.type) {

      case 'success': return { backgroundColor: '#ffffff', border: '1px solid #28a745', color: '#28a745' };

      case 'info': return { backgroundColor: '#ffffff', border: '1px solid #17a2b8', color: '#17a2b8' };

      case 'danger': return { backgroundColor: '#ffffff', border: '1px solid #dc3545', color: '#dc3545' };

      default: return { backgroundColor: '#ffffff', border: '1px solid #17a2b8', color: '#17a2b8' };

    }

  };

  return (

    <>

      <Card

        className="rounded-3 shadow-sm h-100"

        style={{

          display: 'flex',

          flexDirection: 'column',

          transition: 'transform 0.3s ease'

        }}

        onMouseEnter={(e) => e.currentTarget.style.transform = 'translateY(-5px)'}

        onMouseLeave={(e) => e.currentTarget.style.transform = 'translateY(0)'}

      >

        <div className="position-relative" style={{ flexShrink: 0 }}>

          <Card.Img

            variant="top"

            src={product.image}

            alt={product.name}

            style={{

              height: '180px',

              objectFit: 'cover',

              width: '100%'

            }}

          />

          {product.quantity === 0 && (

            <Badge bg="danger" className="position-absolute top-0 end-0 m-2" pill>

              Out of Stock

            </Badge>

          )}

          {parseInt(product.price) > parseInt(product.currentPrice) && (

            <Badge bg="success" className="position-absolute top-0 start-0 m-2" pill>

              Sale

            </Badge>

          )}

        </div>

        <Card.Body className="d-flex flex-column justify-content-between p-3">

          <div>

            <Card.Title

              className="fw-bold mb-2"

              style={{

                fontSize: '1rem',

                height: '2.4rem',

                overflow: 'hidden',

                display: '-webkit-box',

                WebkitLineClamp: 2,

                WebkitBoxOrient: 'vertical'

              }}

            >

              {product.name}

            </Card.Title>

            <Card.Text

              className="text-muted mb-2"

              style={{

                fontSize: '0.85rem',

                height: '2.8rem',

                overflow: 'hidden',

                display: '-webkit-box',

                WebkitLineClamp: 2,

                WebkitBoxOrient: 'vertical'

              }}

            >

              {product.description}

            </Card.Text>

            <div className="mb-2">

              {parseInt(product.price) > parseInt(product.currentPrice) ? (

                <>

                  <span className="text-decoration-line-through text-muted me-2" style={{ fontSize: '0.85rem' }}>

                    {parseInt(product.price).toLocaleString()} VND

                  </span>

                  <span className="text-danger fw-bold" style={{ fontSize: '1rem' }}>

                    {parseInt(product.currentPrice).toLocaleString()} VND

                  </span>

                </>

              ) : (

                <span className="fw-bold" style={{ fontSize: '1rem' }}>

                  {parseInt(product.currentPrice).toLocaleString()} VND

                </span>

              )}

            </div>

          </div>

          <div className="d-flex flex-column gap-2">

            <Button

              variant="outline-primary"

              as={Link}

              to={`/product/${product.id}`}

              className="d-flex align-items-center justify-content-center gap-1 py-1 rounded-pill"

              style={{ fontSize: '0.85rem' }}

            >

              <FaEye size={14} /> View Details

            </Button>

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

              <Button

                variant="success"

                onClick={handleAddToCart}

                disabled={!user || product.quantity === 0}

                className="d-flex align-items-center justify-content-center gap-1 py-1 rounded-pill flex-fill"

                style={{ fontSize: '0.85rem' }}

              >

                <FaShoppingCart size={14} /> Add to Cart

              </Button>

              {user ? (

                isInWishlist ? (

                  <Button

                    variant="danger"

                    as={Link}

                    to="/wishlist"

                    className="d-flex align-items-center justify-content-center gap-1 py-1 rounded-pill flex-fill"

                    style={{ fontSize: '0.85rem' }}

                  >

                    <FaHeart size={14} /> In Wishlist

                  </Button>

                ) : (

                  <Button

                    variant="outline-secondary"

                    onClick={handleWishlistAction}

                    className="d-flex align-items-center justify-content-center gap-1 py-1 rounded-pill flex-fill"

                    style={{ fontSize: '0.85rem' }}

                  >

                    <FaHeart size={14} /> Wishlist

                  </Button>

                )

              ) : (

                <Button

                  variant="outline-secondary"

                  onClick={handleWishlistAction}

                  className="d-flex align-items-center justify-content-center gap-1 py-1 rounded-pill flex-fill"

                  style={{ fontSize: '0.85rem' }}

                >

                  <FaHeart size={14} /> Wishlist

                </Button>

              )}

            </div>

          </div>

        </Card.Body>

      </Card>

      <ToastContainer position="top-end" className="p-3" style={{ zIndex: 1050 }}>

        <Toast

          show={showToast}

          onClose={() => setShowToast(false)}

          delay={3000}

          autohide

          style={getToastBgStyle()}

        >

          <Toast.Header style={{ backgroundColor: 'transparent', borderBottom: 'none' }}>

            <strong className="me-auto" style={{ display: 'flex', alignItems: 'center' }}>

              {getToastIcon()}

              {toastConfig.title}

            </strong>

          </Toast.Header>

          <Toast.Body>

            {toastConfig.message}

          </Toast.Body>

        </Toast>

      </ToastContainer>

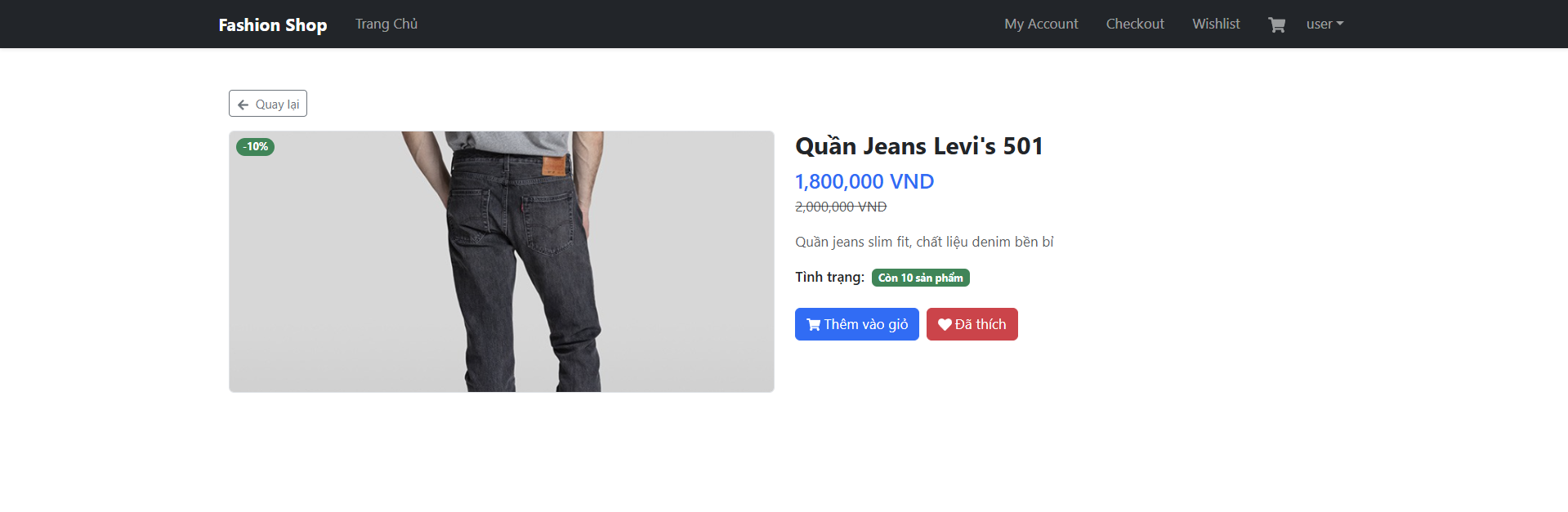
    </>

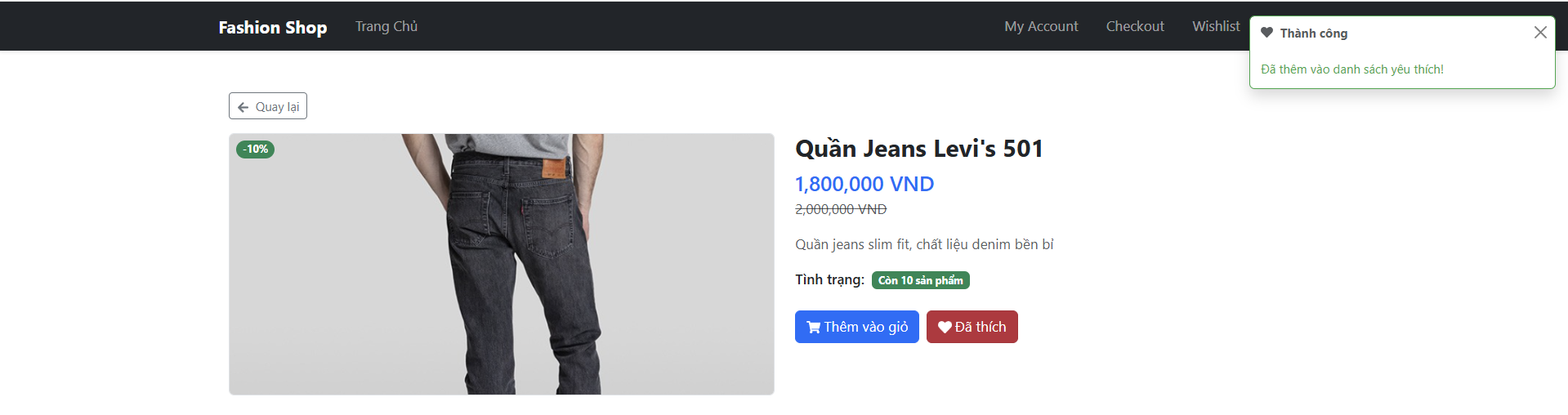
  );

};

export default ProductCard;

## ProductDetail.js





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

import { useParams, useNavigate, useLocation } from 'react-router-dom';

import { useSelector, useDispatch } from 'react-redux';

import { fetchProducts } from '../redux/slices/productSlice';

import { addToCart, addToWishlist } from '../redux/slices/cartSlice';

import {

  Container,

  Row,

  Col,

  Button,

  Spinner,

  Alert,

  Badge,

  Toast,

  ToastContainer,

  Image

} from 'react-bootstrap';

import {

  FaShoppingCart,

  FaHeart,

  FaInfoCircle,

  FaExclamationTriangle,

  FaEdit,

  FaArrowLeft

} from 'react-icons/fa';

const ProductDetail = () => {

  const { id } = useParams();

  const dispatch = useDispatch();

  const navigate = useNavigate();

  const location = useLocation();

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

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

  const products = useSelector(state => state.products.items);

  const user = useSelector(state => state.auth.user);

  const product = products.find(p => p.id === id);

  const isInWishlist = user?.wishlist?.includes(product?.id);

  const [showToast, setShowToast] = useState(false);

  const [toastConfig, setToastConfig] = useState({ type: 'success', message: '', title: '' });

  useEffect(() => {

    const loadProduct = async () => {

      try {

        await dispatch(fetchProducts()).unwrap();

        setLoading(false);

      } catch (err) {

        setError('Không thể tải thông tin sản phẩm');

        setLoading(false);

      }

    };

    loadProduct();

  }, [dispatch, id]);

  const showToastMessage = (type, title, message) => {

    setToastConfig({ type, title, message });

    setShowToast(true);

    setTimeout(() => setShowToast(false), 3000);

  };

  const handleAddToCart = () => {

    if (!user) {

      showToastMessage('info', 'Thông tin', 'Vui lòng đăng nhập để thêm vào giỏ hàng');

      const redirectUri = `${location.pathname}${location.search}`;

      setTimeout(() => window.location.href = `/login?redirect\_uri=${encodeURIComponent(redirectUri)}`, 1500);

      return;

    }

    if (!product || product.quantity <= 0) {

      showToastMessage('danger', 'Lỗi', 'Sản phẩm đã hết hàng!');

      return;

    }

    dispatch(addToCart({ ...product, quantity: 1 }))

      .then(() => {

        showToastMessage('success', 'Thành công', 'Đã thêm vào giỏ hàng!');

      })

      .catch(() => showToastMessage('danger', 'Lỗi', 'Không thể thêm vào giỏ hàng!'));

  };

  const handleAddToWishlist = () => {

    if (!user) {

      showToastMessage('info', 'Thông tin', 'Vui lòng đăng nhập để thêm vào danh sách yêu thích');

      const redirectUri = `${location.pathname}${location.search}`;

      setTimeout(() => window.location.href = `/login?redirect\_uri=${encodeURIComponent(redirectUri)}`, 1500);

      return;

    }

    if (!isInWishlist) {

      dispatch(addToWishlist(product.id))

        .then(() => showToastMessage('success', 'Thành công', 'Đã thêm vào danh sách yêu thích!'))

        .catch(() => showToastMessage('danger', 'Lỗi', 'Không thể thêm vào danh sách yêu thích!'));

    }

  };

  const getToastIcon = () => {

    switch (toastConfig.type) {

      case 'success': return <FaHeart className="me-2" />;

      case 'info': return <FaInfoCircle className="me-2" />;

      case 'danger': return <FaExclamationTriangle className="me-2" />;

      default: return <FaInfoCircle className="me-2" />;

    }

  };

  const getToastBgStyle = () => {

    switch (toastConfig.type) {

      case 'success': return { backgroundColor: '#ffffff', border: '1px solid #28a745', color: '#28a745' };

      case 'info': return { backgroundColor: '#ffffff', border: '1px solid #17a2b8', color: '#17a2b8' };

      case 'danger': return { backgroundColor: '#ffffff', border: '1px solid #dc3545', color: '#dc3545' };

      default: return { backgroundColor: '#ffffff', border: '1px solid #17a2b8', color: '#17a2b8' };

    }

  };

  if (loading) return (

    <Container className="my-4 text-center">

      <Spinner animation="border" role="status" variant="primary">

        <span className="visually-hidden">Đang tải...</span>

      </Spinner>

    </Container>

  );

  if (error) return (

    <Container className="my-4">

      <Alert variant="danger">{error}</Alert>

    </Container>

  );

  if (!product) return (

    <Container className="my-4">

      <Alert variant="warning">

        Không tìm thấy sản phẩm (ID: {id})

        <div className="mt-2">

          <Button variant="outline-primary" onClick={() => navigate('/')} size="sm">

            <FaArrowLeft className="me-1" /> Quay lại trang chủ

          </Button>

        </div>

      </Alert>

    </Container>

  );

  const discountPercentage = parseInt(product.price) > parseInt(product.currentPrice)

    ? Math.round((1 - parseInt(product.currentPrice) / parseInt(product.price)) \* 100)

    : 0;

  return (

    <Container className="my-4">

      <Button

        variant="outline-secondary"

        onClick={() => navigate(-1)}

        className="mb-3"

        size="sm"

      >

        <FaArrowLeft className="me-1" /> Quay lại

      </Button>

      <Row className="g-4">

        {/\* Hình ảnh sản phẩm \*/}

        <Col md={6}>

          <div className="position-relative">

            <Image

              src={product.image}

              alt={product.name}

              fluid

              className="rounded border"

              style={{ width: '100%', height: '300px', objectFit: 'cover' }}

            />

            {product.quantity === 0 && (

              <Badge bg="danger" className="position-absolute top-0 end-0 m-2" pill>

                Hết hàng

              </Badge>

            )}

            {discountPercentage > 0 && (

              <Badge bg="success" className="position-absolute top-0 start-0 m-2" pill>

                -{discountPercentage}%

              </Badge>

            )}

          </div>

        </Col>

        {/\* Thông tin sản phẩm \*/}

        <Col md={6}>

          <div>

            <h3 className="fw-bold mb-2">{product.name}</h3>

            <div className="mb-3">

              <h4 className="text-primary mb-1">

                {parseInt(product.currentPrice).toLocaleString()} VND

              </h4>

              {discountPercentage > 0 && (

                <div className="d-flex align-items-center">

                  <span className="text-decoration-line-through text-muted me-2">

                    {parseInt(product.price).toLocaleString()} VND

                  </span>

                </div>

              )}

            </div>

            <p className="text-muted mb-3">{product.description}</p>

            <div className="mb-3">

              <span className="fw-medium">Tình trạng: </span>

              <Badge bg={product.quantity > 0 ? 'success' : 'danger'} className="ms-1">

                {product.quantity > 0 ? `Còn ${product.quantity} sản phẩm` : 'Hết hàng'}

              </Badge>

            </div>

            {/\* Các nút hành động \*/}

            <div className="d-flex flex-wrap gap-2 mt-4">

              {user?.role === 'user' && product.quantity > 0 && (

                <Button

                  variant="primary"

                  onClick={handleAddToCart}

                  className="d-flex align-items-center"

                  disabled={product.quantity <= 0}

                >

                  <FaShoppingCart className="me-1" /> Thêm vào giỏ

                </Button>

              )}

              {user?.role === 'user' && (

                <Button

                  variant={isInWishlist ? 'danger' : 'outline-secondary'}

                  onClick={handleAddToWishlist}

                  className="d-flex align-items-center"

                >

                  <FaHeart className="me-1" />

                  {isInWishlist ? 'Đã thích' : 'Yêu thích'}

                </Button>

              )}

              {user?.role === 'admin' && (

                <Button

                  variant="outline-warning"

                  onClick={() => navigate(`/edit/${id}`)}

                  className="d-flex align-items-center"

                >

                  <FaEdit className="me-1" /> Chỉnh sửa

                </Button>

              )}

            </div>

          </div>

        </Col>

      </Row>

      <ToastContainer position="top-end" className="p-3" style={{ zIndex: 1050 }}>

        <Toast

          show={showToast}

          onClose={() => setShowToast(false)}

          delay={3000}

          autohide

          style={getToastBgStyle()}

        >

          <Toast.Header style={{ backgroundColor: 'transparent', borderBottom: 'none' }}>

            <strong className="me-auto" style={{ display: 'flex', alignItems: 'center' }}>

              {getToastIcon()}

              {toastConfig.title}

            </strong>

          </Toast.Header>

          <Toast.Body>

            {toastConfig.message}

          </Toast.Body>

        </Toast>

      </ToastContainer>

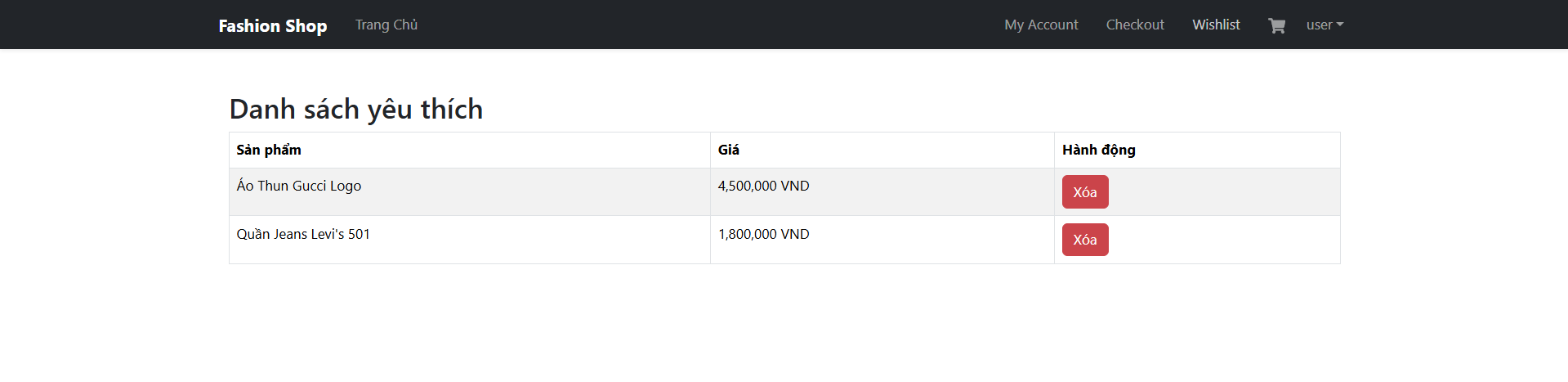
    </Container>

  );

};

export default ProductDetail;

## Wishlist.js



import React, { useEffect } from 'react';

import { useSelector, useDispatch } from 'react-redux';

import { Table, Button, Container } from 'react-bootstrap';

import { fetchProducts, removeFromWishlist } from '../redux/slices/productSlice';

const Wishlist = () => {

  const dispatch = useDispatch();

  const user = useSelector(state => state.auth.user);

  const products = useSelector(state => state.products.items);

  useEffect(() => {

    dispatch(fetchProducts());

  }, [dispatch]);

  const wishlistItems = user?.wishlist?.map(id => products.find(p => p.id === id)).filter(p => p) || [];

  const handleRemoveFromWishlist = (productId) => {

    if (user && user.wishlist) {

      const updatedWishlist = user.wishlist.filter(id => id !== productId);

      dispatch(removeFromWishlist({ userId: user.id, wishlist: updatedWishlist }));

    }

  };

  return (

    <Container className="mt-4">

      <h2>Danh sách yêu thích</h2>

      {wishlistItems.length === 0 ? (

        <p>Không có sản phẩm nào trong danh sách yêu thích.</p>

      ) : (

        <Table striped bordered hover>

          <thead>

            <tr>

              <th>Sản phẩm</th>

              <th>Giá</th>

              <th>Hành động</th>

            </tr>

          </thead>

          <tbody>

            {wishlistItems.map(product => (

              <tr key={product.id}>

                <td>{product.name}</td>

                <td>{parseInt(product.currentPrice).toLocaleString()} VND</td>

                <td>

                  <Button variant="danger" onClick={() => handleRemoveFromWishlist(product.id)}>Xóa</Button>

                </td>

              </tr>

            ))}

          </tbody>

        </Table>

      )}

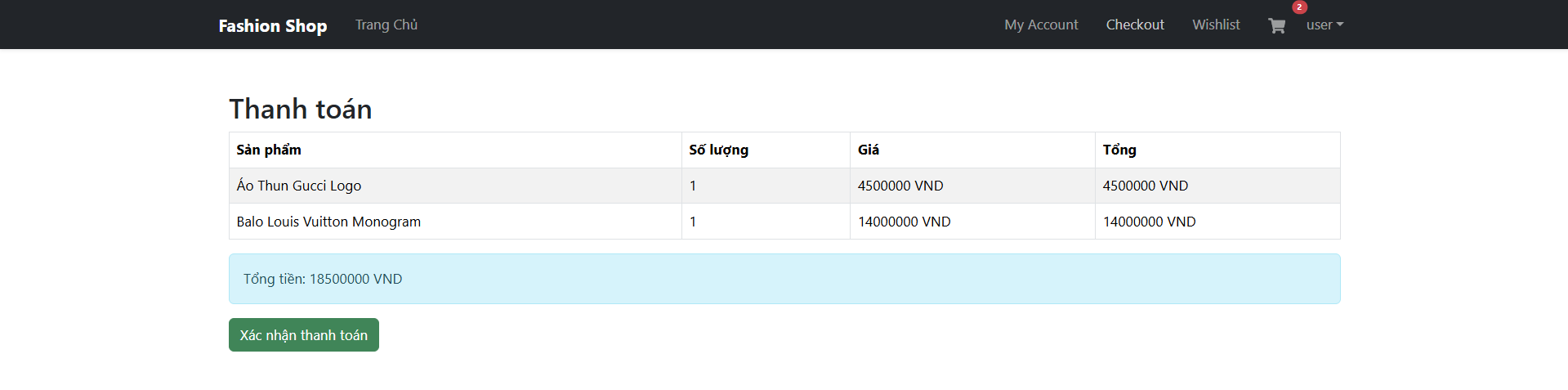
    </Container>

  );

};

export default Wishlist;

## Checkout.js



import React, { useEffect } from 'react';

import { useSelector, useDispatch } from 'react-redux';

import { fetchCarts, clearCart } from '../redux/slices/cartSlice';

import { fetchProducts } from '../redux/slices/productSlice';

import { Table, Button, Container, Alert } from 'react-bootstrap';

import axios from 'axios';

const Checkout = () => {

  const dispatch = useDispatch();

  const { items: carts } = useSelector(state => state.carts);

  const { items: products } = useSelector(state => state.products);

  const user = useSelector(state => state.auth.user);

  useEffect(() => {

    dispatch(fetchCarts());

    dispatch(fetchProducts());

  }, [dispatch]);

  const userCarts = carts.filter(item => item.userId === user?.id);

  const getTotal = () => {

    return userCarts.reduce((total, item) => {

      const product = products.find(p => p.id === item.productId);

      return total + (product ? parseInt(product.currentPrice) \* item.quantity : 0);

    }, 0);

  };

  const handleCheckout = async () => {

    const order = {

      id: Date.now().toString(),

      userId: user.id,

      items: userCarts.map(item => ({

        productId: item.productId,

        quantity: item.quantity

      })),

      total: getTotal(),

      date: new Date().toISOString()

    };

    await axios.post('http://localhost:3001/orders', order);

    await dispatch(clearCart(user.id));

    alert('Đặt hàng thành công!');

  };

  if (user?.role !== 'user') return <Alert variant="warning">Chỉ dành cho user.</Alert>;

  return (

    <Container className="mt-4">

      <h2>Thanh toán</h2>

      <Table striped bordered hover>

        <thead>

          <tr>

            <th>Sản phẩm</th>

            <th>Số lượng</th>

            <th>Giá</th>

            <th>Tổng</th>

          </tr>

        </thead>

        <tbody>

          {userCarts.map(item => {

            const product = products.find(p => p.id === item.productId);

            return (

              <tr key={item.id}>

                <td>{product?.name || 'Unknown'}</td>

                <td>{item.quantity}</td>

                <td>{product?.currentPrice || 0} VND</td>

                <td>{(product ? parseInt(product.currentPrice) \* item.quantity : 0)} VND</td>

              </tr>

            );

          })}

        </tbody>

      </Table>

      <Alert variant="info">Tổng tiền: {getTotal()} VND</Alert>

      <Button variant="success" onClick={handleCheckout}>Xác nhận thanh toán</Button>

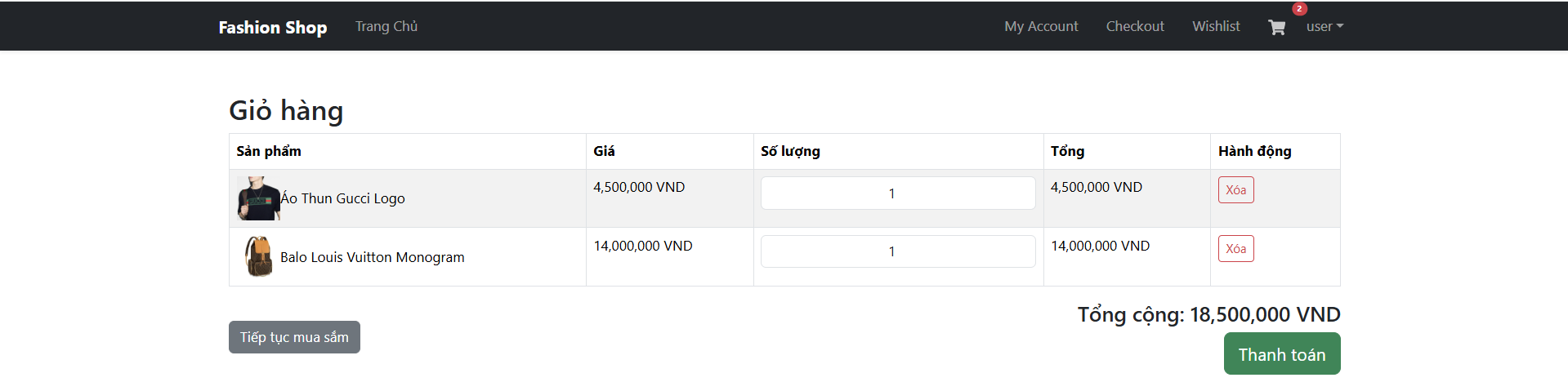
    </Container>

  );

};

export default Checkout;

## Cart.js



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

import { useSelector, useDispatch } from 'react-redux';

import { fetchCarts, updateCartItem, deleteCartItem, clearCart } from '../redux/slices/cartSlice';

import { fetchProducts } from '../redux/slices/productSlice';

import { Table, Button, Container, Form, Alert, Modal } from 'react-bootstrap';

import { useNavigate } from 'react-router-dom';

const Cart = () => {

  const dispatch = useDispatch();

  const navigate = useNavigate();

  const { items: carts } = useSelector(state => state.carts);

  const { items: products } = useSelector(state => state.products);

  const user = useSelector(state => state.auth.user);

  const [showCheckoutModal, setShowCheckoutModal] = useState(false);

  const [checkoutSuccess, setCheckoutSuccess] = useState(false);

  const [isDeleting, setIsDeleting] = useState(false);

  useEffect(() => {

    dispatch(fetchCarts());

    dispatch(fetchProducts());

  }, [dispatch]);

  const userCarts = carts.filter(item => item.userId === user?.id);

  const handleUpdate = (id, newQuantity, productId, oldQuantity) => {

    if (newQuantity > 0) {

      dispatch(updateCartItem({ id, quantity: newQuantity, productId, oldQuantity }));

    }

  };

  const handleDelete = async (id, productId, quantity) => {

    setIsDeleting(true);

    try {

      await dispatch(deleteCartItem({ id, productId, quantity })).unwrap();

    } catch (error) {

      console.error('Failed to delete item:', error);

    } finally {

      setIsDeleting(false);

    }

  };

  const getTotal = () => {

    return userCarts.reduce((total, item) => {

      const product = products.find(p => p.id === item.productId);

      return total + (product ? parseInt(product.currentPrice) \* item.quantity : 0);

    }, 0);

  };

  const handleCheckout = () => {

    setShowCheckoutModal(true);

  };

  const confirmCheckout = async () => {

    try {

      await dispatch(clearCart(user.id)).unwrap();

      setCheckoutSuccess(true);

      setShowCheckoutModal(false);

      setTimeout(() => setCheckoutSuccess(false), 3000);

    } catch (error) {

      console.error('Checkout failed:', error);

    }

  };

  if (!user) return <Alert variant="warning">Vui lòng đăng nhập để xem giỏ hàng.</Alert>;

  return (

    <Container className="mt-4">

      <h2>Giỏ hàng</h2>

      {userCarts.length === 0 ? (

        <Alert variant="info">Giỏ hàng trống.</Alert>

      ) : (

        <>

          <Table striped bordered hover>

            <thead>

              <tr>

                <th>Sản phẩm</th>

                <th>Giá</th>

                <th>Số lượng</th>

                <th>Tổng</th>

                <th>Hành động</th>

              </tr>

            </thead>

            <tbody>

              {userCarts.map(item => {

                const product = products.find(p => p.id === item.productId);

                const productPrice = product ? parseInt(product.currentPrice) : 0;

                const totalPrice = productPrice \* item.quantity;

                return (

                  <tr key={item.id}>

                    <td>

                      <div className="d-flex align-items-center">

                        <img src={product?.image} alt={product?.name} style={{ width: '50px', height: '50px' }} />

                        <span>{product?.name || 'Unknown'}</span>

                      </div>

                    </td>

                    <td>{productPrice.toLocaleString()} VND</td>

                    <td>

                      <Form.Control

                        type="number"

                        value={item.quantity}

                        min="1"

                        className="text-center"

                        onChange={e => handleUpdate(item.id, parseInt(e.target.value), item.productId, item.quantity)}

                      />

                    </td>

                    <td>{totalPrice.toLocaleString()} VND</td>

                    <td>

                      <Button

                        variant="outline-danger"

                        size="sm"

                        onClick={() => handleDelete(item.id, item.productId, item.quantity)}

                        disabled={isDeleting}

                      >

                        {isDeleting ? 'Đang xóa...' : 'Xóa'}

                      </Button>

                    </td>

                  </tr>

                );

              })}

            </tbody>

          </Table>

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

            <Button variant="secondary" onClick={() => navigate('/')}>Tiếp tục mua sắm</Button>

            <div className="text-end">

              <h4>Tổng cộng: {getTotal().toLocaleString()} VND</h4>

              <Button variant="success" size="lg" onClick={handleCheckout}>Thanh toán</Button>

            </div>

          </div>

        </>

      )}

      <Modal show={showCheckoutModal} onHide={() => setShowCheckoutModal(false)}>

        <Modal.Header closeButton>

          <Modal.Title>Xác nhận thanh toán</Modal.Title>

        </Modal.Header>

        <Modal.Body>

          <p>Bạn có chắc chắn muốn thanh toán đơn hàng này?</p>

          <p><strong>Tổng tiền: {getTotal().toLocaleString()} VND</strong></p>

        </Modal.Body>

        <Modal.Footer>

          <Button variant="secondary" onClick={() => setShowCheckoutModal(false)}>Hủy</Button>

          <Button variant="primary" onClick={confirmCheckout}>Xác nhận thanh toán</Button>

        </Modal.Footer>

      </Modal>

      {checkoutSuccess && (

        <Alert variant="success" className="mt-3">

          Thanh toán thành công!

        </Alert>

      )}

    </Container>

  );

};

export default Cart;