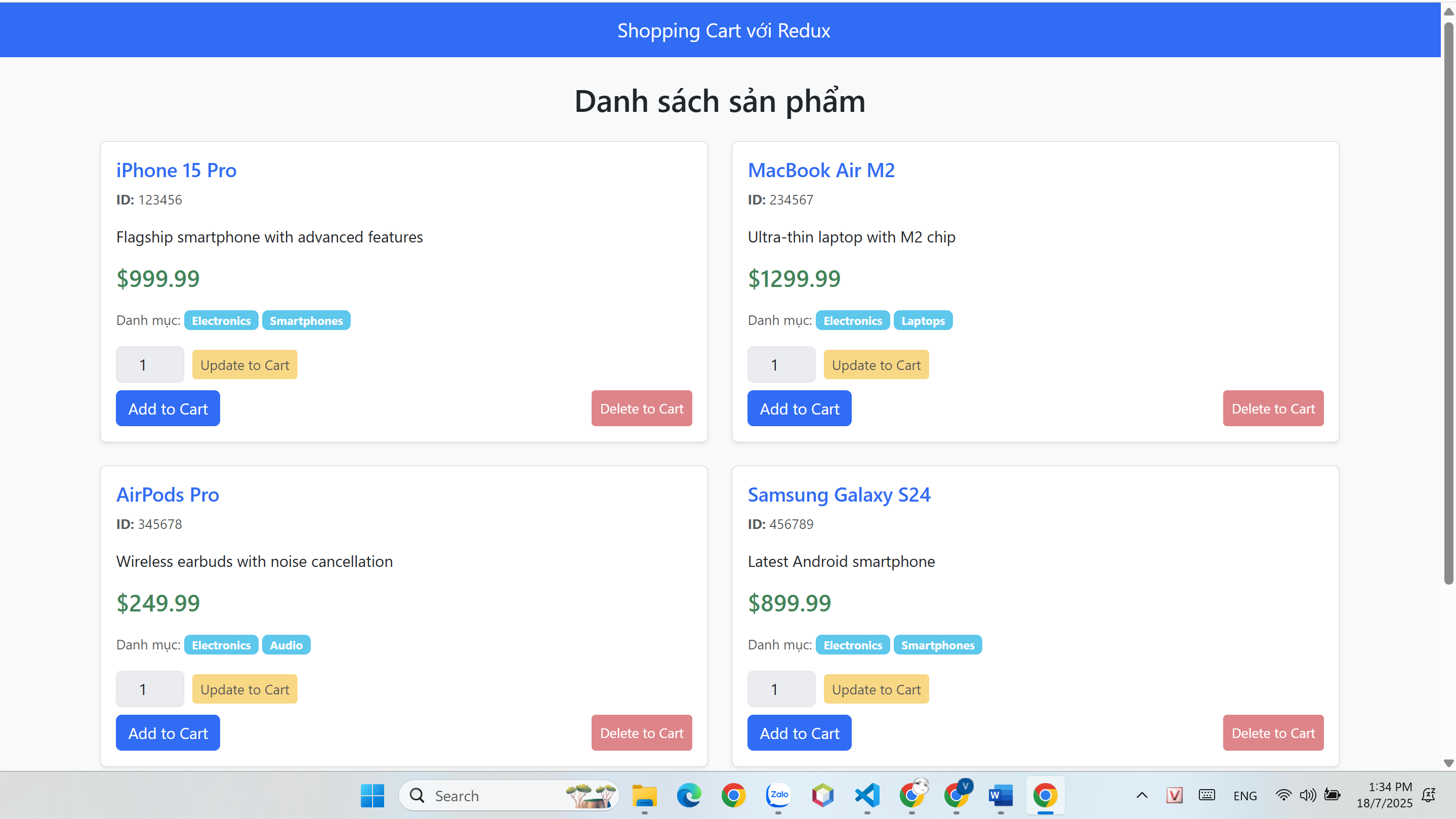
# Exercise 24: Introduction to Redux

## ProductList.js



import React, { useState } from 'react';

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

import { addToCart, updateCartQuantity, removeFromCart } from '../redux/actions';

import { Container, Row, Col, Card, Button, Badge, Form } from 'react-bootstrap';

const ProductList = () => {

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

  const cartItems = useSelector(state => state.cart);

  const dispatch = useDispatch();

  // State để lưu số lượng tạm thời cho mỗi sản phẩm

  const [quantities, setQuantities] = useState({});

  const handleAddToCart = (product) => {

    dispatch(addToCart(product));

    alert(`Đã thêm ${product.name} vào giỏ hàng!`);

  };

  const handleUpdateQuantity = (productId, quantity) => {

    if (quantity > 0) {

      dispatch(updateCartQuantity(productId, quantity));

      alert(`Đã cập nhật số lượng sản phẩm ${productId}!`);

    }

  };

  const handleRemoveFromCart = (productId) => {

    dispatch(removeFromCart(productId));

    alert(`Đã xóa sản phẩm ${productId} khỏi giỏ hàng!`);

  };

  const handleQuantityChange = (productId, value) => {

    setQuantities(prev => ({

      ...prev,

      [productId]: value

    }));

  };

  return (

    <Container className="mb-5">

      <h2 className="mb-4 text-center">Danh sách sản phẩm</h2>

      <Row>

        {products.map(product => {

          const cartItem = cartItems.find(item => item.id === product.id);

          const quantity = quantities[product.id] || (cartItem ? cartItem.quantity : 1);

          return (

            <Col key={product.id} md={6} lg={6} className="mb-4">

              <Card className="h-100 shadow-sm">

                <Card.Body className="d-flex flex-column">

                  <Card.Title className="text-primary">{product.name}</Card.Title>

                  <Card.Text className="text-muted small">

                    <strong>ID:</strong> {product.id}

                  </Card.Text>

                  <Card.Text className="flex-grow-1">

                    {product.description}

                  </Card.Text>

                  <Card.Text className="h4 text-success mb-3">

                    ${product.price}

                  </Card.Text>

                  <div className="mb-3">

                    <small className="text-muted">Danh mục: </small>

                    {product.catalogs.map(catalog => (

                      <Badge key={catalog} bg="info" className="me-1">

                        {catalog}

                      </Badge>

                    ))}

                  </div>

                  <div className="d-flex align-items-center mb-2">

                    <Form.Control

                      type="number"

                      min="1"

                      value={quantity}

                      onChange={(e) => handleQuantityChange(product.id, parseInt(e.target.value))}

                      style={{ width: '70px' }}

                      className="text-center me-2"

                      disabled={!cartItem} // Vô hiệu hóa nếu sản phẩm chưa có trong giỏ

                    />

                    <Button

                      variant="warning"

                      size="sm"

                      onClick={() => handleUpdateQuantity(product.id, quantity)}

                      disabled={!cartItem} // Vô hiệu hóa nếu sản phẩm chưa có trong giỏ

                    >

                      Update to Cart

                    </Button>

                  </div>

                  <div className="d-flex justify-content-between">

                    <Button

                      variant="primary"

                      onClick={() => handleAddToCart(product)}

                      disabled={cartItem} // Vô hiệu hóa nếu sản phẩm đã có trong giỏ

                    >

                      Add to Cart

                    </Button>

                    <Button

                      variant="danger"

                      size="sm"

                      onClick={() => handleRemoveFromCart(product.id)}

                      disabled={!cartItem} // Vô hiệu hóa nếu sản phẩm chưa có trong giỏ

                    >

                      Delete to Cart

                    </Button>

                  </div>

                </Card.Body>

              </Card>

            </Col>

          );

        })}

      </Row>

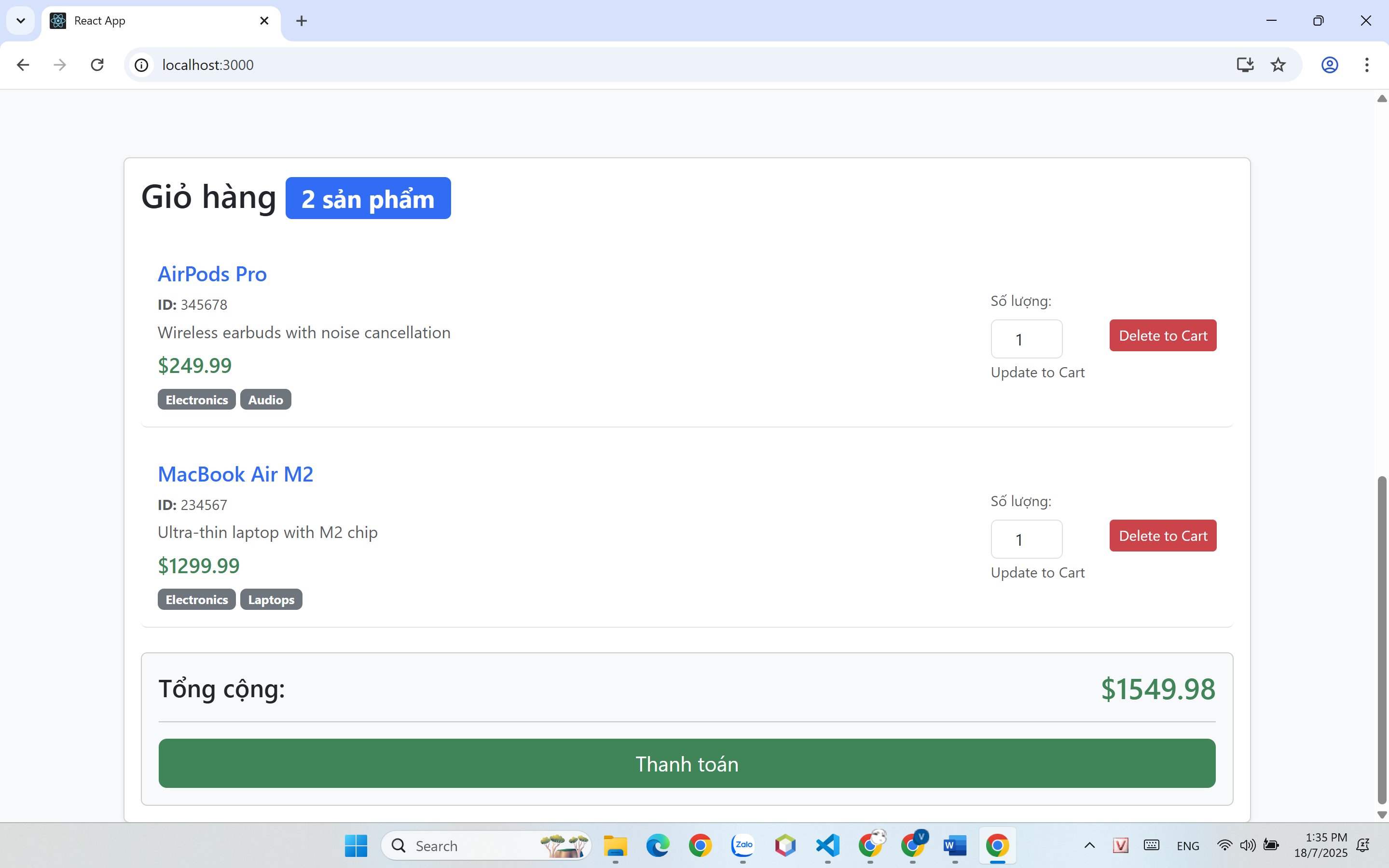
    </Container>

  );

};

export default ProductList;

## Cart.js



import React from 'react';

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

import { updateCartQuantity, removeFromCart } from '../redux/actions';

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

const Cart = () => {

  const cartItems = useSelector(state => state.cart);

  const dispatch = useDispatch();

  const handleUpdateQuantity = (productId, quantity) => {

    if (quantity > 0) {

      dispatch(updateCartQuantity(productId, quantity));

    }

  };

  const handleRemoveFromCart = (productId) => {

    dispatch(removeFromCart(productId));

    alert('Đã xóa sản phẩm khỏi giỏ hàng!');

  };

  const calculateTotal = () => {

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

  };

  if (cartItems.length === 0) {

    return (

      <Container>

        <Card className="shadow-sm">

          <Card.Body>

            <h2 className="mb-4">Giỏ hàng</h2>

            <Alert variant="info" className="text-center">

              <i className="fas fa-shopping-cart me-2"></i>

              Giỏ hàng của bạn đang trống

            </Alert>

          </Card.Body>

        </Card>

      </Container>

    );

  }

  return (

    <Container>

      <Card className="shadow-sm">

        <Card.Body>

          <h2 className="mb-4">

            Giỏ hàng

            <Badge bg="primary" className="ms-2">

              {cartItems.length} sản phẩm

            </Badge>

          </h2>

          {cartItems.map(item => (

            <Card key={item.id} className="mb-3 border-0 border-bottom">

              <Card.Body>

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

                  <Col md={8}>

                    <h5 className="text-primary">{item.name}</h5>

                    <p className="text-muted small mb-1">

                      <strong>ID:</strong> {item.id}

                    </p>

                    <p className="text-muted mb-2">{item.description}</p>

                    <p className="h5 text-success mb-2">${item.price}</p>

                    <div>

                      {item.catalogs.map(catalog => (

                        <Badge key={catalog} bg="secondary" className="me-1">

                          {catalog}

                        </Badge>

                      ))}

                    </div>

                  </Col>

                  <Col md={4}>

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

                      <div className="me-3">

                        <Form.Label className="small text-muted">Số lượng:</Form.Label>

                        <Form.Control

                          type="number"

                          min="1"

                          value={item.quantity}

                          onChange={(e) => handleUpdateQuantity(item.id, parseInt(e.target.value))}

                          style={{ width: '70px' }}

                          className="text-center"

                        />

                        <small className="text-muted">Update to Cart</small>

                      </div>

                      <Button

                        variant="danger"

                        size="sm"

                        onClick={() => handleRemoveFromCart(item.id)}

                        className="ms-2"

                      >

                        Delete to Cart

                      </Button>

                    </div>

                  </Col>

                </Row>

              </Card.Body>

            </Card>

          ))}

          <Card className="mt-4 bg-light">

            <Card.Body>

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

                <Col>

                  <h4 className="mb-0">Tổng cộng:</h4>

                </Col>

                <Col xs="auto">

                  <h3 className="mb-0 text-success">${calculateTotal()}</h3>

                </Col>

              </Row>

              <hr />

              <div className="d-grid">

                <Button variant="success" size="lg">

                  Thanh toán

                </Button>

              </div>

            </Card.Body>

          </Card>

        </Card.Body>

      </Card>

    </Container>

  );

};

export default Cart;

## App.js

import React from 'react';

import { Provider } from 'react-redux';

import store from './redux/store';

import ProductList from './components/ProductList';

import Cart from './components/Cart';

import { Container, Navbar } from 'react-bootstrap';

import 'bootstrap/dist/css/bootstrap.min.css';

const App = () => {

  return (

    <Provider store={store}>

      <div className="bg-light min-vh-100">

        <Navbar bg="primary" variant="dark" className="mb-4">

          <Container>

            <Navbar.Brand className="mx-auto">

              <i className="fas fa-shopping-cart me-2"></i>

              Shopping Cart với Redux

            </Navbar.Brand>

          </Container>

        </Navbar>

        <Container>

          <ProductList />

          <Cart />

        </Container>

      </div>

    </Provider>

  );

};

export default App;

## action.js

// Action Types - Định nghĩa các loại hành động

export const ADD\_TO\_CART = 'ADD\_TO\_CART';

export const UPDATE\_CART\_QUANTITY = 'UPDATE\_CART\_QUANTITY';

export const REMOVE\_FROM\_CART = 'REMOVE\_FROM\_CART';

// Action Creators - Các hàm tạo action

export const addToCart = (product) => ({

  type: ADD\_TO\_CART,

  payload: product

});

export const updateCartQuantity = (productId, quantity) => ({

  type: UPDATE\_CART\_QUANTITY,

  payload: { productId, quantity }

});

export const removeFromCart = (productId) => ({

  type: REMOVE\_FROM\_CART,

  payload: productId

});

## Reducers.js

import { ADD\_TO\_CART, UPDATE\_CART\_QUANTITY, REMOVE\_FROM\_CART } from './actions';

// Initial State - Trạng thái ban đầu

const initialState = {

  products: [

    {

      id: '123456',

      name: 'iPhone 15 Pro',

      price: 999.99,

      description: 'Flagship smartphone with advanced features',

      catalogs: ['Electronics', 'Smartphones']

    },

    {

      id: '234567',

      name: 'MacBook Air M2',

      price: 1299.99,

      description: 'Ultra-thin laptop with M2 chip',

      catalogs: ['Electronics', 'Laptops']

    },

    {

      id: '345678',

      name: 'AirPods Pro',

      price: 249.99,

      description: 'Wireless earbuds with noise cancellation',

      catalogs: ['Electronics', 'Audio']

    },

    {

      id: '456789',

      name: 'Samsung Galaxy S24',

      price: 899.99,

      description: 'Latest Android smartphone',

      catalogs: ['Electronics', 'Smartphones']

    }

  ],

  cart: []

};

// Reducer - Hàm xử lý thay đổi state

const cartReducer = (state = initialState, action) => {

  switch (action.type) {

    case ADD\_TO\_CART:

      // Kiểm tra xem sản phẩm đã có trong giỏ hàng chưa

      const existingItem = state.cart.find(item => item.id === action.payload.id);

      if (existingItem) {

        // Nếu đã có, tăng số lượng

        return {

          ...state,

          cart: state.cart.map(item =>

            item.id === action.payload.id

              ? { ...item, quantity: item.quantity + 1 }

              : item

          )

        };

      } else {

        // Nếu chưa có, thêm mới với quantity = 1

        return {

          ...state,

          cart: [...state.cart, { ...action.payload, quantity: 1 }]

        };

      }

    case UPDATE\_CART\_QUANTITY:

      return {

        ...state,

        cart: state.cart.map(item =>

          item.id === action.payload.productId

            ? { ...item, quantity: action.payload.quantity }

            : item

        )

      };

    case REMOVE\_FROM\_CART:

      return {

        ...state,

        cart: state.cart.filter(item => item.id !== action.payload)

      };

    default:

      return state;

  }

};

export default cartReducer;

## store.js

import { createStore } from 'redux';

import cartReducer from './reducers';

// Tạo Redux store

const store = createStore(

  cartReducer,

  // Thêm Redux DevTools Extension nếu có

  window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_\_ && window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_\_()

);

export default store;