Ministry of Higher Education and Scientific Research - University of Mila
Institute of Mathematics and Computer Science - Department of Computer Science
Master 1 STIC – Advanced Web Development - 2025/2026

Practical Works 4 - The Final Polish & Admin Architecture

Objective: Complete the DZ-Shop Single Page Application by implementing advanced UI components, data filtering, an interactive Cart with quantity controls, and a secure Admin Dashboard for CRUD operations.

Deliverables for the end of TP4:

1. **Polished UI:** A responsive Hero Carousel, updated Navbar, and persistent Footer.
2. **Advanced Cart:** A cart page where users can modify quantities, remove items, and checkout via a Modal.
3. **Filtering Engine:** Category buttons on the Home page that instantly filter the product grid.
4. **Protected Admin Route:** A simulated Login page that unlocks a hidden Dashboard to Add/Delete products.

1. Global UI Enhancements (Footer & Navbar)

First, let's create a persistent Footer and modify our Navbar to accommodate the new pages.

A. Create the Footer (src/layouts/Footer.jsx):

```jsx
export default function Footer() {
  return (
    <footer className="bg-gray-900 text-white py-8 mt-12">
      <div className="max-w-6xl mx-auto px-4 grid grid-cols-1 md:grid-cols-3 gap-8 text-center md:text-left">
        <div>
          <h2 className="text-2xl font-bold text-orange-500">DZ-Shop</h2>
          <p className="text-gray-400 mt-2 text-sm">The best e-commerce platform in Mila, built with React & Tailwind v4.</p>
        </div>
        <div>
          <h3 className="font-semibold text-lg mb-2">Quick Links</h3>
          <ul className="text-gray-400 text-sm space-y-1">
            <li className="hover:text-orange-500 cursor-pointer">Home</li>
            <li className="hover:text-orange-500 cursor-pointer">Cart</li>
            <li className="hover:text-orange-500 cursor-pointer">Admin Login</li>
          </ul>
        </div>
        <div>
          <h3 className="font-semibold text-lg mb-2">Contact Us</h3>
          <p className="text-gray-400 text-sm">Email: contact@dzshop.dz</p>
          <p className="text-gray-400 text-sm">Phone: +213 555 00 00 00</p>
        </div>
      </div>
      <div className="border-t border-gray-800 mt-8 pt-4 text-center text-gray-500 text-xs">
        © 2026 University of Mila - Master STIC. All rights reserved.
      </div>
    </footer>
  );
}
```

B. Modify the Navbar (src/components/Navbar.jsx): Update your existing Navbar to include a link to the Login page.

```jsx
// Add this inside the flex container next to Cart
<Link to="/login" className="hover:text-orange-600 transition">Admin</Link>
```

2. The Hero Carousel (src/components/Carousel.jsx)

Instead of installing a heavy library, we will build a custom carousel using **useState** and **useEffect** to teach the React Lifecycle.

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

const banners = [
  "https://images.unsplash.com/photo-1607082348824-0a96f2a4b9da?q=80&w=2070&auto=format&fit=crop",
  "https://images.unsplash.com/photo-1483985988355-763728e1935b?q=80&w=2070&auto=format&fit=crop",
  "https://images.unsplash.com/photo-1490481651871-ab68de25d43d?q=80&w=2070&auto=format&fit=crop"
];

export default function Carousel() {
  const [current, setCurrent] = useState(0);

  // Auto-scroll logic
  useEffect(() => {
    const timer = setInterval(() => {
      setCurrent((prev) => (prev === banners.length - 1 ? 0 : prev + 1));
    }, 4000);
    return () => clearInterval(timer); // Cleanup on unmount
  }, []);

  return (
    <div className="relative w-full h-64 md:h-96 overflow-hidden bg-gray-900">
      {banners.map((img, index) => (
        <img
          key={index}
          src={img}
          alt={` Banner ${index}`}
          className={` absolute inset-0 w-full h-full object-cover transition-opacity duration-1000 ${
            index === current ? "opacity-100" : "opacity-0"
          }`}
        />
      ))}
      <div className="absolute inset-0 bg-black/40 flex items-center justify-center">
        <h2 className="text-white text-3xl md:text-5xl font-bold tracking-widest uppercase shadow-black drop-shadow-lg">
          Welcome to DZ-Shop
        </h2>
      </div>
    </div>
  );
}
```

3. Filtering Categories in Home.jsx

We will modify our **Home.jsx** to include the Carousel and a Category Filtering system.

```
import { useState, useEffect } from 'react';
import ProductCard from '../components/ProductCard';
import Carousel from '../components/Carousel';

export default function Home() {
  const [products, setProducts] = useState([]);
  const [loading, setLoading] = useState(true);
  const [category, setCategory] = useState('all'); // Filter state

  useEffect(() => {
    fetch('https://fakestoreapi.com/products')
      .then(res => res.json())
      .then(data => {
        setProducts(data);
        setLoading(false);
      });
  }, []);

  // Compute filtered products dynamically
```

```
  const filteredProducts = category === 'all'
    ? products
    : products.filter(p => p.category === category);

  const categories = ['all', "men's clothing", "women's clothing", "electronics", "jewelery"];

  if (loading) return <div className="text-center p-20 text-2xl">Loading store...</div>;

  return (
    <div>
      <Carousel />
      <div className="p-8 max-w-7xl mx-auto">

        {/* Category Filters */}
        <div className="flex flex-wrap gap-4 justify-center mb-8">
          {categories.map(cat => (
            <button
              key={cat}
              onClick={() => setCategory(cat)}
              className={` px-4 py-2 rounded-full capitalize font-semibold transition ${
                category === cat ? 'bg-orange-600 text-white' : 'bg-gray-200 text-gray-700 hover:bg-gray-300'
              } `}>
              {cat}
            </button>
          ))}
        </div>

        <div className="grid grid-cols-1 sm:grid-cols-2 md:grid-cols-3 lg:grid-cols-4 gap-8">
          {filteredProducts.map(item => <ProductCard key={item.id} product={item} />)}
        </div>
      </div>
    </div>
  );
}
```

4. Advanced Cart Logic & Checkout Modal

To allow users to modify item quantities and remove items, we first need to upgrade our "Brain" (the Context), and then update the UI in **Cart.jsx**.

A. Upgrade src/context/CartContext.jsx: Add the **removeFromCart**, **updateQuantity**, and **clearCart** functions to your existing Context, and make sure to expose them in the value prop.

```
  // ... existing addToCart logic ...

  // NEW: Remove item completely
  const removeFromCart = (id) => {
    setCart((prevCart) => prevCart.filter((item) => item.id !== id));
  };

  // NEW: Increase or Decrease quantity
  const updateQuantity = (id, amount) => {
    setCart((prevCart) => prevCart.map((item) => {
      if (item.id === id) {
        const newQuantity = item.quantity + amount;
        // Don't allow quantity to go below 1 (use remove button instead)
        return newQuantity > 0 ? { ...item, quantity: newQuantity } : item;
      }
      return item;
    }));
  };

  // NEW: Clear cart after checkout
  const clearCart = () => setCart([]);

  // ... existing total calculations ...
```

```
  return (
    <CartContext.Provider value={{
      cart, addToCart, removeFromCart, updateQuantity, clearCart, totalItems, totalPrice
    }}>
      {children}
    </CartContext.Provider>
  );
```

B. Upgrade src/pages/Cart.jsx: Now we consume these new functions. We will add + and - buttons for the quantity, a Remove button, and the Checkout Modal.

```
import { useState, useContext } from 'react';
import { CartContext } from '../context/CartContext';
import { Link } from 'react-router-dom';

export default function Cart() {
  const { cart, totalPrice, removeFromCart, updateQuantity, clearCart } = useContext(CartContext);
  const [isModalOpen, setIsModalOpen] = useState(false);

  const handleCheckout = () => setIsModalOpen(true);

  const confirmPurchase = () => {
    clearCart();
    setIsModalOpen(false);
    alert("Purchase successful! Thank you for choosing DZ-Shop.");
  };

  if (cart.length === 0) {
    return (
      <div className="p-20 text-center">
        <h2 className="text-3xl font-bold text-gray-800 mb-4">Your Cart is Empty</h2>
        <Link to="/" className="text-orange-600 font-bold hover:underline">← Go back to shopping</Link>
      </div>
    );
  }

  return (
    <div className="max-w-4xl mx-auto p-8 mt-10 relative">
      <h2 className="text-3xl font-bold mb-8 text-gray-800">Shopping Cart</h2>

      <div className="bg-white rounded-lg shadow-md p-6 mb-6">
        {cart.map((item) => (
          <div key={item.id} className="flex flex-col md:flex-row items-center justify-between border-b pb-4 mb-4 last:border-0
last:pb-0 last:mb-0 gap-4">

            {/* Product Info */}
            <div className="flex items-center gap-4 w-full md:w-2/5">
              <img src={item.image} alt={item.title} className="h-16 w-16 object-contain" />
              <h4 className="text-sm font-semibold line-clamp-2" title={item.title}>{item.title}</h4>
            </div>

            {/* Quantity Controls */}
            <div className="flex items-center justify-center gap-3 w-full md:w-1/5">
              <button
                onClick={() => updateQuantity(item.id, -1)}
                className="w-8 h-8 flex items-center justify-center bg-gray-200 rounded-full hover:bg-gray-300 font-bold text-gray-
600 transition"
              >
                -
              </button>
              <span className="font-bold w-6 text-center">{item.quantity}</span>
              <button
                onClick={() => updateQuantity(item.id, 1)}
                className="w-8 h-8 flex items-center justify-center bg-gray-200 rounded-full hover:bg-gray-300 font-bold text-gray-
600 transition"
              >
                +
              </button>
            </div>
```

```
        {/* Price & Remove Button */}
        <div className="flex items-center justify-between md:justify-end gap-6 w-full md:w-2/5 text-right">
          <span className="font-bold text-lg text-orange-600">${(item.price * item.quantity).toFixed(2)}</span>
          <button
            onClick={() => removeFromCart(item.id)}
            className="text-red-500 hover:text-red-700 text-sm font-bold uppercase tracking-wider bg-red-50 px-3 py-1 rounded hover:bg-red-100 transition"
          >
            Remove
          </button>
        </div>
      </div>
    ))}
  </div>

  <div className="bg-white rounded-lg shadow-md p-6 flex flex-col md:flex-row justify-between items-center gap-4">
    <span className="text-xl font-bold text-gray-800">Grand Total: <span className="text-3xl text-orange-600">${totalPrice.toFixed(2)}</span></span>
    <button onClick={handleCheckout} className="bg-green-600 text-white px-8 py-3 rounded-lg font-bold text-lg hover:bg-green-700 transition w-full md:w-auto">
      Checkout Now
    </button>
  </div>

  {/* THE CHECKOUT MODAL */}
  {isModalOpen && (
    <div className="fixed inset-0 bg-black/60 z-50 flex items-center justify-center backdrop-blur-sm">
      <div className="bg-white p-8 rounded-xl shadow-2xl max-w-sm w-full text-center">
        <h2 className="text-2xl font-bold text-gray-800 mb-4">Confirm Purchase</h2>
        <p className="text-gray-600 mb-6">Your credit card will be charged <strong className="text-orange-600">${totalPrice.toFixed(2)}</strong>. Proceed?</p>
        <div className="flex gap-4 justify-center">
          <button onClick={() => setIsModalOpen(false)} className="px-4 py-2 bg-gray-200 rounded font-bold hover:bg-gray-300 transition">Cancel</button>
          <button onClick={confirmPurchase} className="px-4 py-2 bg-orange-600 text-white rounded font-bold hover:bg-orange-700 transition">Confirm & Pay</button>
        </div>
      </div>
    </div>
  )}
 </div>
);
}
```

5. The Login Page & Admin Dashboard

A. Simple Login (src/pages/Login.jsx): To simulate authentication without a real backend, we will use a simple form that redirects upon a hardcoded success.

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

export default function Login() {
 const [email, setEmail] = useState('');
 const [password, setPassword] = useState('');
 const navigate = useNavigate();

 const handleLogin = (e) => {
  e.preventDefault();
  if (email === 'admin@dzshop.dz' && password === '123456') {
   localStorage.setItem('isAdmin', 'true');
   navigate('/admin');
  } else {
   alert("Invalid credentials! Try admin@dzshop.dz / 123456");
  }
 };
```

```
  return (
    <div className="flex justify-center items-center h-[70vh]">
      <form onSubmit={handleLogin} className="bg-white p-8 shadow-lg rounded-lg w-96">
        <h2 className="text-2xl font-bold mb-6 text-center text-orange-600">Admin Login</h2>
        <input
          type="email" placeholder="Email" required value={email} onChange={e => setEmail(e.target.value)}
          className="w-full border p-2 mb-4 rounded"
        />
        <input
          type="password" placeholder="Password" required value={password} onChange={e => setPassword(e.target.value)}
          className="w-full border p-2 mb-6 rounded"
        />
        <button className="w-full bg-orange-600 text-white py-2 rounded font-bold hover:bg-orange-700
transition">Login</button>
      </form>
    </div>
  );
}
```

B. Admin Dashboard (src/pages/Admin.jsx): This page demonstrates local CRUD (Create, Read, Update, Delete) architecture.

```
import { useState, useEffect } from 'react';
import { useNavigate } from 'react-router-dom';

export default function Admin() {
  const navigate = useNavigate();
  const [products, setProducts] = useState([]);

  // Route Protection Check
  useEffect(() => {
    if (localStorage.getItem('isAdmin') !== 'true') navigate('/login');

    // Fetch initial list
    fetch('https://fakestoreapi.com/products')
      .then(res => res.json())
      .then(data => setProducts(data));
  }, [navigate]);

  const deleteProduct = (id) => {
    setProducts(products.filter(p => p.id !== id));
    alert(` Product ${id} deleted locally! ` );
  };

  const handleLogout = () => {
    localStorage.removeItem('isAdmin');
    navigate('/login');
  };

  return (
    <div className="p-8 max-w-6xl mx-auto">
      <div className="flex justify-between items-center mb-8">
        <h1 className="text-3xl font-bold text-gray-800">Admin Dashboard</h1>
        <button onClick={handleLogout} className="bg-red-500 text-white px-4 py-2 rounded font-bold transition hover:bg-red-
600">Logout</button>
      </div>

      <div className="bg-white shadow-md rounded-lg overflow-x-auto">
        <table className="w-full text-left min-w-[600px]">
          <thead className="bg-gray-800 text-white">
            <tr>
              <th className="p-4">ID</th>
              <th className="p-4">Image</th>
              <th className="p-4">Title</th>
              <th className="p-4">Price</th>
              <th className="p-4 text-center">Actions</th>
            </tr>
          </thead>
          <tbody>
```

```
          {products.map(p => (
            <tr key={p.id} className="border-b hover:bg-gray-50">
              <td className="p-4">{p.id}</td>
              <td className="p-4"><img src={p.image} className="h-10 w-10 object-contain" alt="" /></td>
              <td className="p-4 text-sm truncate max-w-xs">{p.title}</td>
              <td className="p-4 font-bold text-orange-600">${p.price}</td>
              <td className="p-4 text-center flex gap-2 justify-center">
                <button className="bg-blue-500 text-white px-3 py-1 rounded text-xs hover:bg-blue-600
transition">Edit</button>
                <button onClick={() => deleteProduct(p.id)} className="bg-red-500 text-white px-3 py-1 rounded text-xs hover:bg-
red-600 transition">Delete</button>
              </td>
            </tr>
          ))}
        </tbody>
      </table>
    </div>
  </div>
 );
}
```

6. Final Assembly (src/App.jsx)

Ensure all pages and the new Footer are imported into your main router.

```
// src/App.jsx
import { BrowserRouter as Router, Routes, Route } from 'react-router-dom';
import { CartProvider } from './context/CartContext';
import Navbar from './components/Navbar';
import Footer from './layouts/Footer';
import Home from './pages/Home';
import ProductDetails from './pages/ProductDetails';
import Cart from './pages/Cart';
import Login from './pages/Login';
import Admin from './pages/Admin';

export default function App() {
 return (
  <CartProvider>
   <Router>
    <div className="min-h-screen flex flex-col bg-gray-50">
     <Navbar />
     {/* Main content takes up remaining space to push footer down */}
     <main className="flex-grow">
      <Routes>
       <Route path="/" element={<Home />} />
       <Route path="/product/:id" element={<ProductDetails />} />
       <Route path="/cart" element={<Cart />} />
       <Route path="/login" element={<Login />} />
       <Route path="/admin" element={<Admin />} />
      </Routes>
     </main>
     <Footer />
    </div>
   </Router>
  </CartProvider>
 );
}
```

Checklist:

- Can you filter products on the Home page?
- Does the Carousel rotate smoothly?
- Can you increase/decrease the quantity of items inside the Cart?
- Can you Remove an item entirely from the Cart?
- Does the Checkout Modal successfully empty the Cart?
- Can you login as admin@dzshop.dz and delete rows from the table?