KATHFORD
INTERNATIONAL COLLEGE OF ENGINEERING & MANAGEMENT | AFFILIATED TO TRIBHUVAN UNIVERSITY

Lab Report 3

JavaScript Arrays, Objects and DOM manipulation

Course: CSC381 - E-Commerce

Submitted by:

Parakram Kharel
Roll No: 24

Kathford International College of Engineering and Management
Affiliated to Tribhuvan University

August 30, 2025

1. Objective

To implement dynamic cart management functionality for an e-commerce web application using JavaScript arrays, objects, and DOM manipulation techniques, enabling users to add, modify, and remove items from their shopping cart with real-time updates and persistent storage capabilities.

2. Tools and Technologies Used

Technology	Purpose
HTML5	Structure and semantic markup for product displays
CSS3	Styling, animations, and responsive design
JavaScript	Price calculations and interactive functionality
VS Code	Code editor and development environment
Web Browser Developer Tools	Debugging and performance testing

3. Theory / Background

JavaScript enables dynamic web functionality through data manipulation and user interactions. In e-commerce applications, JavaScript transforms static product displays into interactive shopping experiences with real-time cart management and interface updates.

Lab 3 focuses on JavaScript arrays, objects, and DOM manipulation for cart functionality. Arrays store and manage cart items using methods like `push()`, `splice()`, and `reduce()`. Objects structure product and cart data, while DOM manipulation updates the interface dynamically and localStorage provides cart persistence.

4. Page Layout Design

4.1 Enhanced Product Card Structure

Each interactive product card contains:

- Product Image with dynamic badges (NEW, SALE, HOT)
- Product Title and Description
- Enhanced Price Section with:
  - Current price display with proper formatting

  - Original price with strikethrough for sale items
  - Automatic discount percentage calculation
  - Savings amount display
- Interactive Quantity Section featuring:
  - Increment/decrement buttons with validation
  - Number input field (1–10 range)
  - Real-time total calculation display
- Action buttons with visual feedback

4.2 Cart Management System

- Dynamic cart overlay with professional modal design
- Real-time cart item management using JavaScript arrays
- Individual item quantity adjustment and removal functionality
- Comprehensive cart totals including subtotal, tax (13%), and shipping
- Cart persistence using localStorage for session continuity
- Mobile-responsive cart interface with backdrop blur effects

4.3 JavaScript Array and Object Implementation

- Product data management using arrays of objects
- Cart items stored as objects with properties (id, name, price, quantity)
- Array methods for data manipulation
- DOM manipulation for dynamic content updates and real-time interface changes

5. Code Snippets

5.1 HTML Code: Product Card with Cart Integration

```
<div class="product-card" data-category="electronics" data-product-id="1">
  <div class="product-image">
    <img src="assets/1-Premium_Headphones.jpg" alt="Premium Headphones">
    <div class="product-badge">NEW</div>
  </div>
  <div class="product-info">
    <h3>Premium Headphones</h3>
    <p class="product-desc">
      High-quality wireless headphones with noise cancellation
    </p>
    <div class="price-section">
      <div class="price-tag" data-original-price="8999">NPR 8,999</div>
    </div>
    <div class="quantity-section">
      <label>Quantity:</label>
      <div class="quantity-selector">
        <button class="qty-btn minus" onclick="changeQuantity(this,
        -1)">-</button>
        <input type="number" class="qty-input" value="1" min="1" max="10"
        onchange="updateProductTotal(this)">
        <button class="qty-btn plus" onclick="changeQuantity(this,
        1)">+</button>
      </div>
      <div class="product-total">
        Total: NPR <span class="total-amount">8,999</span>
      </div>
    </div>
    <div class="product-actions">
      <button class="add-cart-btn" data-price="8999"
      onclick="addToCart(this)">
        Add to Cart
      </button>
      <button class="details-btn">View Details</button>
    </div>
  </div>
</div>

```

5.2 JavaScript Code: Product Data Array

```
const products = [
{
    id: 1,
    name: "Premium Headphones",
    description: "High-quality wireless headphones with noise
    cancellation",
    price: 8999,
    category: "electronics",
    image: "assets/1-Premium_Headphones.jpg",
    badge: "NEW"
  },
{
    id: 2,
    name: "Sport Sneakers",
    description: "Comfortable running shoes perfect for daily workouts",
    price: 4599,
    originalPrice: 6999,
    category: "fashion",
    image: "assets/2-Sport_Sneakers.jpg",
    badge: "SALE"
  }
];
```

5.3 JavaScript Code: Cart Management with Arrays

```
let cartItems = [];
let cartTotal = 0;
let nextCartItemId = 1;

function addToCart(button) {
    const productCard = button.closest('.product-card');
    const productId =
    parseInt(productCard.getAttribute('data-product-id'));
    const quantity =
    parseInt(productCard.querySelector('.qty-input').value);

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

    const existingItemIndex = cartItems.findIndex(item => item.productId
    === productId);

    if (existingItemIndex !== -1) {
        cartItems[existingItemIndex].quantity += quantity;
```

```
            cartItems[existingItemIndex].totalPrice =
            cartItems[existingItemIndex].quantity *
            cartItems[existingItemIndex].unitPrice;
        } else {
            const cartItem = {
                id: nextCartItemId++,
                productId: productId,
                name: product.name,
                unitPrice: product.price, quantity: quantity,
                totalPrice: product.price * quantity,
                image: product.image, category: product.category
              };
            cartItems.push(cartItem);
        }

        updateAllCartDisplays();
        saveCartToStorage();
    }
```

5.4 JavaScript Code: Array Methods for Cart Operations

```
    function removeFromCart(cartItemId) {
        const itemIndex = cartItems.findIndex(item => item.id ===
        cartItemId);
        if (itemIndex !== -1) {
            cartItems.splice(itemIndex, 1);
            updateAllCartDisplays();
            saveCartToStorage();
        }
    }

    function updateCartTotals() {
        const subtotal = cartItems.reduce((total, item) => total +
        item.totalPrice, 0);
        const tax = subtotal * 0.13; // 13% VAT
        const shipping = 0;
        const finalTotal = subtotal + tax + shipping;

        cartTotal = finalTotal;
        document.getElementById('total-price').textContent =
        formatPrice(finalTotal);
    }
```

5.5 JavaScript Code: DOM Manipulation for Cart Display

```
function renderCartOverlayItems() {
    const container = document.getElementById('cart-overlay-items');

    if (cartItems.length === 0) {
        container.innerHTML = '<div class="empty-cart-overlay">Your cart
        is empty</div>';
        return;
      }

    const itemsHTML = cartItems.map(item => `
    <div class="overlay-cart-item">
      <div class="overlay-cart-item-image">
        <img src="${item.image}" alt="${item.name}">
      </div>
      <div class="overlay-cart-item-details">
        <div class="overlay-cart-item-name">${item.name}</div>
        <div class="overlay-cart-item-price">NPR
        ${formatPrice(item.unitPrice)} each</div>
        <div class="overlay-cart-item-quantity">
          <button onclick="updateCartItemQuantity(${item.id},
          ${item.quantity - 1})">-</button>
          <input type="number" value="${item.quantity}" min="1" max="10">
          <button onclick="updateCartItemQuantity(${item.id},
          ${item.quantity + 1})">+</button>
        </div>
        <div class="overlay-cart-item-total">Total: NPR
        ${formatPrice(item.totalPrice)}</div>
      </div>
    </div>
    `).join('');

    container.innerHTML = itemsHTML;
  }
```

6. Output / Screenshots

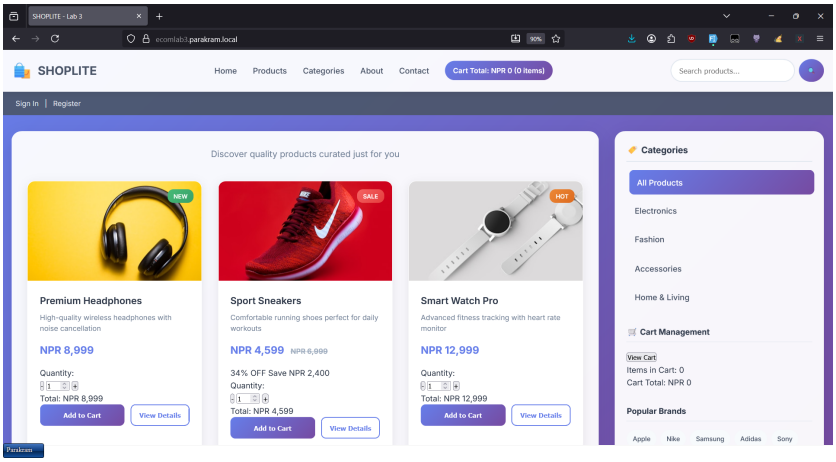

Figure 1: Product Catalog.

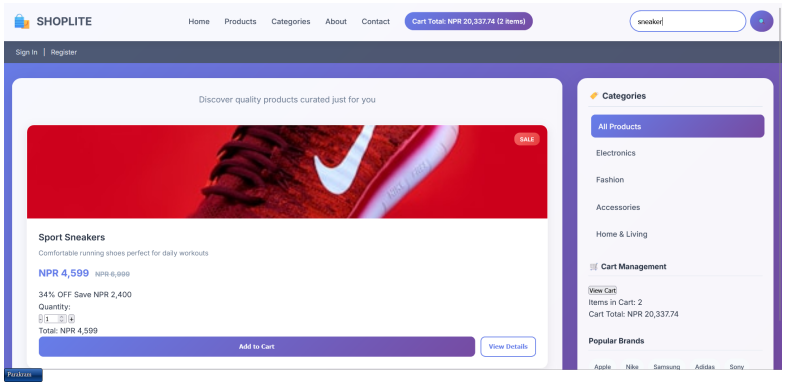

Figure 2: Product Search.

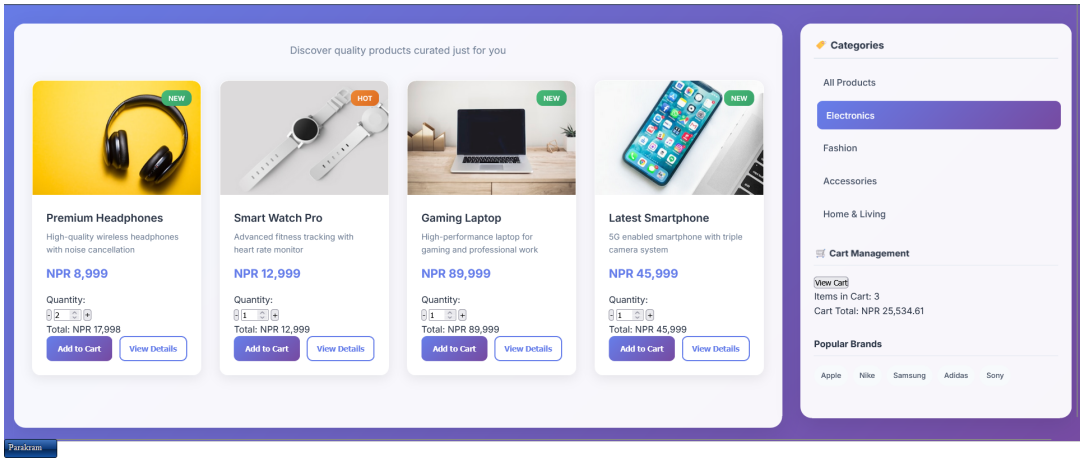

Figure 3: Electronics Category.

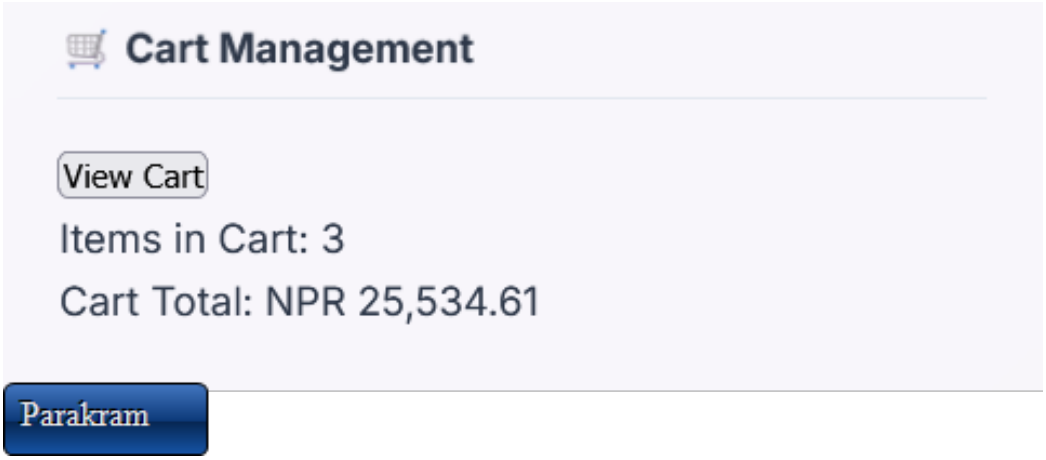

Figure 4: Cart Info in Sidebar.

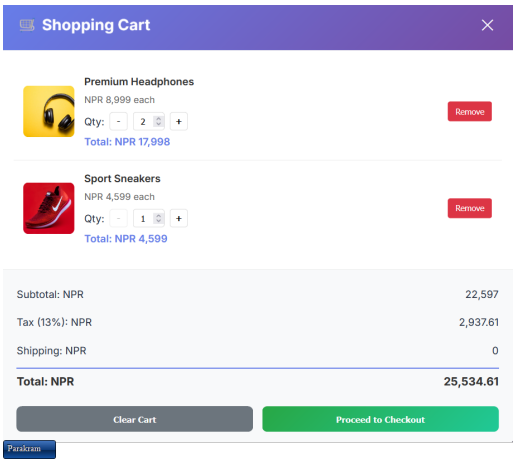

Figure 5: View Cart.

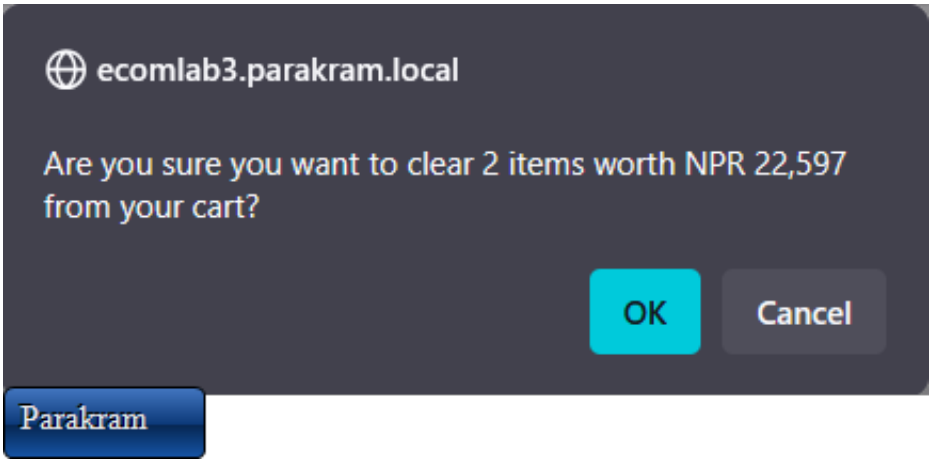

Figure 6: Clear Cart Confirmation.

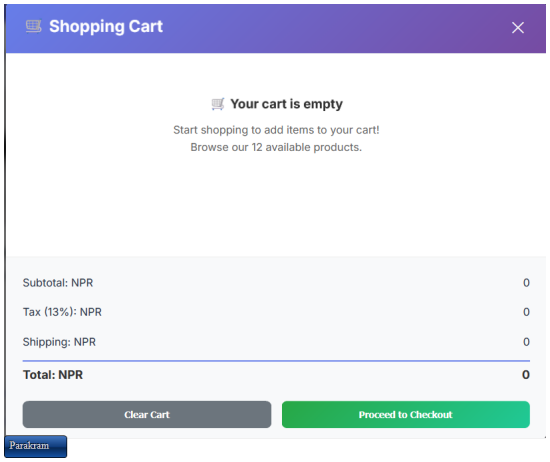

Figure 7: Empty Cart.

7. Result

The e-commerce application successfully implemented comprehensive cart management using JavaScript arrays and objects. Features include dynamic cart operations with real-time item addition, modification, and removal using array methods like push(), splice(), and reduce(). The cart overlay displays items with persistent localStorage storage, automatic total calculations including tax and shipping, and complete DOM manipulation for seamless user interactions across all devices.

8. Conclusion

This lab demonstrated advanced JavaScript programming concepts through practical e-commerce cart implementation. Students mastered array and object manipulation, learned essential array methods for data operations, and implemented sophisticated DOM manipulation techniques. The project established strong foundations in data structures, event-driven programming, and modern web storage, transforming static product displays into fully functional shopping cart systems with professional-grade user experience.

9. References

- MDN Array Methods Documentation
- MDN DOM Manipulation Guide
- MDN Web Storage API
- Modern JavaScript Array Methods Tutorial