

```
// Initialize the application
document.addEventListener('DOMContentLoaded', () => {
  // Load products and initialize
  loadProducts();
  initializeCart();
  initializeTheme();
  setupEventListeners();

  // Set up Formspree for notifications (replace with your ID)
  window.formspreeEndpoint = 'https://formspree.io/f/YOUR_FORM_ID_HERE';
});

// Product Data (15 products)
const products = [
  { id: 1, name: "Botanical Ceramic Mug", price: 24.99, desc: "Hand-painted with natural motifs", icon: "☕" },
  { id: 2, name: "Olive Wood Cutting Board", price: 39.99, desc: "Sustainable olive wood, natural finish", icon: "🪵" },
  { id: 3, name: "Sage Green Linen Napkins", price: 29.99, desc: "Set of 4, 100% organic linen", icon: "🍃" },
  { id: 4, name: "Terracotta Plant Pots", price: 34.99, desc: "Set of 3 different sizes", icon: "🪴" },
  { id: 5, name: "Bamboo Utensil Set", price: 19.99, desc: "Eco-friendly bamboo kitchen tools", icon: "🥬" },
  { id: 6, name: "Essential Oil Diffuser", price: 49.99, desc: "Ultrasonic with mood lighting", icon: "💧" },
  { id: 7, name: "Organic Cotton Tote Bag", price: 22.99, desc: "Large capacity, natural dye", icon: "🛍️" },
  { id: 8, name: "Herbal Tea Collection", price: 27.99, desc: "12 varieties of organic herbs", icon: "🍵" },
  { id: 9, name: "Recycled Glass Vase", price: 32.99, desc: "Hand-blown, unique patterns", icon: "🏺" },
  { id: 10, name: "Cork Yoga Mat", price: 44.99, desc: "Natural cork, non-slip surface", icon: "🧘" },
  { id: 11, name: "Beeswax Food Wraps", price: 18.99, desc: "Set of 3, reusable alternative", icon: "🍯" },
  { id: 12, name: "Hemp Rope Hammock", price: 89.99, desc: "Handwoven, supports 300 lbs", icon: "🌴" },
  { id: 13, name: "Natural Soy Candle", price: 26.99, desc: "Scented with essential oils", icon: "🕯️" },
  { id: 14, name: "Stoneware Dinner Set", price: 129.99, desc: "4-piece set, microwave safe", icon: "🍽️" },

```

```

    { id: 15, name: "Macramé Plant Hanger", price: 31.99, desc: "Hand-knotted, adjustable length", icon: "🌿" }
  ];

// Cart State
let cart = JSON.parse(localStorage.getItem('botanicalCart')) || [];

// Load Products
function loadProducts() {
  const productsGrid = document.getElementById('products-grid');

  productsGrid.innerHTML = products.map(product => `
    <div class="product-card" data-id="${product.id}">
      <div class="product-image">
        ${product.icon}
      </div>
      <h3 class="product-title">${product.name}</h3>
      <p class="product-description">${product.desc}</p>
      <div class="product-footer">
        <span class="product-price">${product.price.toFixed(2)}</span>
        <button class="btn-add-to-cart" onclick="addToCart(${product.id})">
          <i class="fas fa-plus"></i> Add to Cart
        </button>
      </div>
    </div>
  `).join('');
}

// Cart Functions
function initializeCart() {
  updateCartCount();
  renderCartItems();
}

function addToCart(productId) {
  const product = products.find(p => p.id === productId);
  const existingItem = cart.find(item => item.id === productId);

  if (existingItem) {
    existingItem.quantity += 1;
  } else {
    cart.push({
      ...product,
      quantity: 1
    });
  }
}

```

```

    });
}

updateCart();
showNotification(`${product.name} added to cart!`);
}

function removeFromCart(productId) {
    cart = cart.filter(item => item.id !== productId);
    updateCart();
}

function updateCart() {
    localStorage.setItem('botanicalCart', JSON.stringify(cart));
    updateCartCount();
    renderCartItems();
    updateCartTotal();
}

function updateCartCount() {
    const count = cart.reduce((total, item) => total + item.quantity, 0);
    document.getElementById('cart-count').textContent = count;
}

function renderCartItems() {
    const cartItems = document.getElementById('cart-items');
    const total = cart.reduce((sum, item) => sum + (item.price * item.quantity), 0);

    cartItems.innerHTML = cart.length > 0 ? cart.map(item => `
        <div class="cart-item">
            <div class="cart-item-info">
                <h4>${item.name}</h4>
                <p>Quantity: ${item.quantity}</p>
                <p class="cart-item-price">${(item.price *
item.quantity).toFixed(2)}</p>
            </div>
            <button class="cart-item-remove" onclick="removeFromCart(${item.id})">
                <i class="fas fa-trash"></i>
            </button>
        </div>
    `).join('') : '<p style="text-align: center; color: var(--text-secondary);">Your
cart is empty</p>';

    document.getElementById('cart-total').textContent = `${total.toFixed(2)}`;

```

```
}

function updateCartTotal() {
  const total = cart.reduce((sum, item) => sum + (item.price * item.quantity), 0);
  document.getElementById('cart-total').textContent = `$$${total.toFixed(2)}`;
}

// Theme Functions
function initializeTheme() {
  const savedTheme = localStorage.getItem('theme') || 'light';
  document.documentElement.setAttribute('data-theme', savedTheme);

  // Update theme link text
  const themeLink = document.getElementById('theme-link');
  if (themeLink) {
    themeLink.textContent = savedTheme === 'light' ? 'Switch to Dark Mode' :
'Switch to Light Mode';
  }
}

function toggleTheme() {
  const currentTheme = document.documentElement.getAttribute('data-theme');
  const newTheme = currentTheme === 'light' ? 'dark' : 'light';

  document.documentElement.setAttribute('data-theme', newTheme);
  localStorage.setItem('theme', newTheme);

  // Update theme link text
  const themeLink = document.getElementById('theme-link');
  if (themeLink) {
    themeLink.textContent = newTheme === 'light' ? 'Switch to Dark Mode' :
'Switch to Light Mode';
  }

  showNotification(`Switched to ${newTheme} mode`);
}

// Modal Functions
function openCheckoutModal() {
  if (cart.length === 0) {
    showNotification('Your cart is empty!', 'error');
    return;
  }
}
```

```

const modal = document.getElementById('checkout-modal');
const orderItems = document.getElementById('order-items');
const total = cart.reduce((sum, item) => sum + (item.price * item.quantity), 0);

orderItems.innerHTML = cart.map(item => `
  <div style="display: flex; justify-content: space-between; margin-bottom:
10px;">
    <span>${item.name} x${item.quantity}</span>
    <span>${(item.price * item.quantity).toFixed(2)}</span>
  </div>
`).join('');

document.getElementById('order-total').textContent = `${total.toFixed(2)}`;
modal.classList.add('active');
}

function closeCheckoutModal() {
  document.getElementById('checkout-modal').classList.remove('active');
}

// Form Submission
async function submitOrder(e) {
  e.preventDefault();

  const name = document.getElementById('name').value;
  const email = document.getElementById('email').value;
  const phone = document.getElementById('phone').value;

  // Prepare order data
  const orderData = {
    name,
    email,
    phone,
    items: cart.map(item => `${item.name} (x${item.quantity})`).join(', '),
    total: cart.reduce((sum, item) => sum + (item.price * item.quantity), 0),
    date: new Date().toLocaleString(),
    theme: document.documentElement.getAttribute('data-theme')
  };

  try {
    // Send notification to owner using Formspree
    const response = await fetch(window.formspreeEndpoint, {
      method: 'POST',
      headers: {

```

```

        'Content-Type': 'application/json',
        'Accept': 'application/json'
    },
    body: JSON.stringify({
        _subject: `New Order from ${name}`,
        name: `Customer: ${name}`,
        email: `Customer Email: ${email}`,
        phone: `Customer Phone: ${phone}`,
        order_details: `Items: ${orderData.items}`,
        order_total: `Total: ${orderData.total.toFixed(2)}`,
        order_date: `Date: ${orderData.date}`,
        theme_used: `Theme: ${orderData.theme}`
    })
});

if (response.ok) {
    // Success - clear cart and show confirmation
    cart = [];
    updateCart();
    closeCheckoutModal();
    showNotification('Order placed successfully! Owner has been notified.',
'success');

    // Reset form
    document.getElementById('checkout-form').reset();
} else {
    throw new Error('Failed to send notification');
}
} catch (error) {
    console.error('Error:', error);
    showNotification('Failed to send notification. Please try again.', 'error');
}
}

// UI Helper Functions
function showNotification(message, type = 'info') {
    // Remove existing notification
    const existingNotification = document.querySelector('.notification');
    if (existingNotification) {
        existingNotification.remove();
    }

    // Create notification
    const notification = document.createElement('div');

```

```

notification.className = `notification ${type}`;
notification.innerHTML = `
    <span>${message}</span>
    <button onclick="this.parentElement.remove()">&times;</button>
`;

// Add styles
notification.style.cssText = `
    position: fixed;
    top: 20px;
    right: 20px;
    background: ${type === 'error' ? '#ff4757' : type === 'success' ? '#2ed573'
: 'var(--color-accent)'};
    color: white;
    padding: 15px 20px;
    border-radius: 8px;
    display: flex;
    align-items: center;
    justify-content: space-between;
    gap: 15px;
    z-index: 9999;
    box-shadow: 0 5px 15px rgba(0,0,0,0.2);
    animation: slideIn 0.3s ease;
`;

notification.querySelector('button').style.cssText = `
    background: none;
    border: none;
    color: white;
    font-size: 1.5rem;
    cursor: pointer;
    line-height: 1;
`;

document.body.appendChild(notification);

// Auto-remove after 5 seconds
setTimeout(() => {
    if (notification.parentElement) {
        notification.remove();
    }
}, 5000);

// Add animation

```

```

const style = document.createElement('style');
style.textContent = `
    @keyframes slideIn {
        from { transform: translateX(100%); opacity: 0; }
        to { transform: translateX(0); opacity: 1; }
    }
`;
document.head.appendChild(style);
}

function scrollToProducts() {
    document.getElementById('products').scrollIntoView({
        behavior: 'smooth',
        block: 'start'
    });
}

// Event Listeners Setup
function setupEventListeners() {
    // Theme toggle
    document.getElementById('theme-switch').addEventListener('click', toggleTheme);
    document.getElementById('theme-link')?.addEventListener('click', (e) => {
        e.preventDefault();
        toggleTheme();
    });

    // Cart toggle
    document.getElementById('cart-icon').addEventListener('click', () => {
        document.getElementById('cart-sidebar').classList.add('active');
    });

    document.getElementById('close-cart').addEventListener('click', () => {
        document.getElementById('cart-sidebar').classList.remove('active');
    });

    // Checkout button
    document.getElementById('checkout-btn').addEventListener('click',
openCheckoutModal);

    // Clear cart
    document.getElementById('clear-cart').addEventListener('click', () => {
        cart = [];
        updateCart();
        showNotification('Cart cleared!', 'info');
    });
}

```



```

});

// Close modal
document.getElementById('close-modal').addEventListener('click',
closeCheckoutModal);

// Form submission
document.getElementById('checkout-form').addEventListener('submit',
submitOrder);

// Close modal when clicking outside
document.getElementById('checkout-modal').addEventListener('click', (e) => {
    if (e.target === document.getElementById('checkout-modal')) {
        closeCheckoutModal();
    }
});

// Close cart when clicking outside (on mobile)
document.addEventListener('click', (e) => {
    const sidebar = document.getElementById('cart-sidebar');
    const cartIcon = document.getElementById('cart-icon');

    if (sidebar.classList.contains('active') &&
        !sidebar.contains(e.target) &&
        !cartIcon.contains(e.target)) {
        sidebar.classList.remove('active');
    }
});

// Keyboard shortcuts
document.addEventListener('keydown', (e) => {
    if (e.key === 'Escape') {
        closeCheckoutModal();
        document.getElementById('cart-sidebar').classList.remove('active');
    }

    // Ctrl/Cmd + T to toggle theme
    if ((e.ctrlKey || e.metaKey) && e.key === 't') {
        e.preventDefault();
        toggleTheme();
    }
});
}

```

```
// Export functions for HTML onclick handlers  
window.addToCart = addToCart;  
window.removeFromCart = removeFromCart;  
window.scrollToProducts = scrollToProducts;
```