Planejamento do Frontend com JavaScript Puro

Visão Geral

Este documento detalha o planejamento do frontend do e-commerce utilizando JavaScript puro (vanilla), conforme solicitado. A arquitetura frontend será modular, responsiva e seguirá as tendências de design e UX identificadas na pesquisa de mercado, priorizando a experiência do usuário, performance e segurança.

Princípios de Design

1. Design Responsivo

- Abordagem mobile-first para todas as páginas
- Breakpoints para diferentes tamanhos de tela (mobile, tablet, desktop)
- Uso de unidades relativas (rem, %, vh/vw) para flexibilidade

2. Performance

- Carregamento assíncrono de recursos
- Lazy loading para imagens e componentes
- Minificação e compressão de assets
- Code splitting para carregamento sob demanda

3. Acessibilidade

- Conformidade com WCAG 2.1 nível AA
- Suporte a navegação por teclado
- Textos alternativos para imagens
- Contraste adequado para leitura

4. Usabilidade

- Feedback visual para interações
- Formulários com validação em tempo real
- Mensagens de erro claras e construtivas

- Microinterações para melhorar a experiência

Estrutura de Arquivos

```
/frontend
 /assets
  /images
   /banners
   /products
   /icons
   /logos
  /fonts
 /css
  /base
   reset.css
   typography.css
   variables.css
   animations.css
  /components
   header.css
   footer.css
   buttons.css
   forms.css
   cards.css
   modals.css
   carousel.css
   ...
  /layouts
   grid.css
   flex.css
  /pages
   home.css
   product-list.css
   product-detail.css
   cart.css
   checkout.css
   account.css
   ...
  main.css
 /js
  /core
   config.js
   api.js
   router.js
   store.js
   events.js
   utils.js
  /components
   /common
    header.js
```

```
    footer.js
    navigation.js
    search.js
    modal.js
    toast.js
    carousel.js
    ...
  /product
    product-card.js
    product-gallery.js
    product-filter.js
    product-review.js
    ...
  /cart
    cart-item.js
    cart-summary.js
    cart-actions.js
    ...
  /checkout
    address-form.js
    payment-form.js
    order-summary.js
    ...
  /user
    login-form.js
    register-form.js
    profile-form.js
    order-history.js
    ...
/pages
  home.js
  category.js
  product.js
  cart.js
  checkout.js
  account.js
  ...
/services
  auth-service.js
  product-service.js
  cart-service.js
  order-service.js
  payment-service.js
  user-service.js
  ...
/utils
  formatter.js
  validator.js
  storage.js
  dom.js
  animation.js
  ...
```

```
  main.js
/templates
  header.html
  footer.html
  product-card.html
  ...
index.html
404.html
robots.txt
sitemap.xml
manifest.json
```

Arquitetura JavaScript

1. Padrão de Arquitetura

Utilizaremos uma arquitetura baseada em componentes com o padrão MVC (Model-View-Controller) adaptado para JavaScript puro:

- **Model**: Gerenciamento de dados e estado da aplicação
- **View**: Templates HTML e renderização da interface
- **Controller**: Lógica de negócios e manipulação de eventos

2. Gerenciamento de Estado

Implementaremos um sistema simples de gerenciamento de estado global:

```
// store.js
const Store = (function() {
  let state = {
    user: null,
    cart: {
      items: [],
      total: 0
    },
    products: {
      featured: [],
      recent: []
    },
    ui: {
      isLoading: false,
      activeModal: null,
      notifications: []
    }
  };

  const listeners = [];
```

```
  function getState() {
    return {...state};
  }

  function setState(newState) {
    state = {...state, ...newState};
    notifyListeners();
  }

  function subscribe(listener) {
    listeners.push(listener);
    return () => {
      const index = listeners.indexOf(listener);
      if (index > -1) {
        listeners.splice(index, 1);
      }
    };
  }

  function notifyListeners() {
    listeners.forEach(listener => listener(state));
  }

  return {
    getState,
    setState,
    subscribe
  };
})();

export default Store;
```

3. Sistema de Roteamento

Implementaremos um roteador simples baseado em hash para navegação SPA:

```
// router.js
const Router = (function() {
  const routes = [];

  function addRoute(path, callback) {
    routes.push({
      path,
      callback
    });
  }

  function init() {
    window.addEventListener('hashchange', handleRouteChange);
```

```
    handleRouteChange();
  }

  function handleRouteChange() {
    const path = window.location.hash.slice(1) || '/';
    const route = routes.find(route => route.path === path ||
                              (route.path instanceof RegExp && route.path.test(path)));

    if (route) {
      route.callback(path);
    } else {
      // Rota 404
      window.location.hash = '/404';
    }
  }

  function navigate(path) {
    window.location.hash = path;
  }

  return {
    addRoute,
    init,
    navigate
  };
})();

export default Router;
```

4. Sistema de Eventos

Para comunicação entre componentes:

```
// events.js
const EventBus = (function() {
  const events = {};

  function on(eventName, callback) {
    if (!events[eventName]) {
      events[eventName] = [];
    }
    events[eventName].push(callback);

    return () => {
      off(eventName, callback);
    };
  }

  function off(eventName, callback) {
    if (events[eventName]) {
```

```
    events[eventName] = events[eventName].filter(cb => cb !== callback);
   }
  }

  function emit(eventName, data) {
   if (events[eventName]) {
    events[eventName].forEach(callback => {
     callback(data);
    });
   }
  }

  return {
   on,
   off,
   emit
  };
})();

export default EventBus;
```

5. Comunicação com API

Utilizaremos o Fetch API para comunicação com o backend:

```
// api.js
const API = (function() {
 const BASE_URL = '/api';

 async function request(endpoint, options = {}) {
  const defaultOptions = {
   headers: {
    'Content-Type': 'application/json',
    'Accept': 'application/json'
   },
   credentials: 'same-origin'
  };

  const token = localStorage.getItem('auth_token');
  if (token) {
   defaultOptions.headers['Authorization'] = `Bearer ${token}`;
  }

  const mergedOptions = {
   ...defaultOptions,
   ...options,
   headers: {
    ...defaultOptions.headers,
    ...options.headers
   }
```

```
    };

    try {
      const response = await fetch(`${BASE_URL}${endpoint}`, mergedOptions);

      if (!response.ok) {
        const error = await response.json();
        throw new Error(error.message || 'Ocorreu um erro na requisição');
      }

      return await response.json();
    } catch (error) {
      console.error('API Error:', error);
      throw error;
    }
  }

  function get(endpoint) {
    return request(endpoint);
  }

  function post(endpoint, data) {
    return request(endpoint, {
      method: 'POST',
      body: JSON.stringify(data)
    });
  }

  function put(endpoint, data) {
    return request(endpoint, {
      method: 'PUT',
      body: JSON.stringify(data)
    });
  }

  function del(endpoint) {
    return request(endpoint, {
      method: 'DELETE'
    });
  }

  return {
    get,
    post,
    put,
    delete: del
  };
})();

export default API;
```

Componentes Principais

1. Header

```
 // header.js
import Store from '../core/store.js';
import EventBus from '../core/events.js';

const Header = (function() {
  function render() {
    const headerElement = document.querySelector('header');
    if (!headerElement) return;

    const state = Store.getState();
    const cartItemCount = state.cart.items.reduce((total, item) => total +
item.quantity, 0);

    headerElement.innerHTML = `
      <div class="container">
        <div class="logo">
          <a href="#/">
            <img src="/assets/images/logos/logo.svg" alt="Logo da Loja">
          </a>
        </div>

        <div class="search-bar">
          <form id="search-form">
            <input type="text" placeholder="O que você está procurando?" id="search-
input">
            <button type="submit" aria-label="Buscar">
              <i class="icon-search"></i>
            </button>
          </form>
        </div>

        <nav class="main-nav">
          <ul>
            <li><a href="#/categories">Categorias</a></li>
            <li><a href="#/offers">Ofertas</a></li>
            <li><a href="#/new">Novidades</a></li>
          </ul>
        </nav>

        <div class="user-actions">
          <a href="#/account" class="user-icon">
            <i class="icon-user"></i>
            <span>${state.user ? state.user.name.split(' ')[0] : 'Entrar'}</span>
          </a>

          <a href="#/wishlist" class="wishlist-icon">
```

```
          <i class="icon-heart"></i>
        </a>

        <a href="#/cart" class="cart-icon">
          <i class="icon-cart"></i>
          <span class="cart-count">${cartItemCount}</span>
        </a>
      </div>

      <button class="mobile-menu-toggle" aria-label="Menu">
        <span></span>
        <span></span>
        <span></span>
      </button>
    </div>
  `;

  bindEvents();
}

function bindEvents() {
  const searchForm = document.getElementById('search-form');
  const mobileMenuToggle = document.querySelector('.mobile-menu-toggle');

  if (searchForm) {
    searchForm.addEventListener('submit', handleSearch);
  }

  if (mobileMenuToggle) {
    mobileMenuToggle.addEventListener('click', toggleMobileMenu);
  }

  // Inscrever-se para atualizações de estado
  Store.subscribe(state => {
    // Atualizar apenas o contador do carrinho para evitar re-renderização completa
    const cartCountElement = document.querySelector('.cart-count');
    if (cartCountElement) {
      const cartItemCount = state.cart.items.reduce((total, item) => total +
item.quantity, 0);
      cartCountElement.textContent = cartItemCount;
    }
  });
}

function handleSearch(event) {
  event.preventDefault();
  const searchInput = document.getElementById('search-input');
  if (searchInput && searchInput.value.trim()) {
    window.location.hash = `/search?q=$
{encodeURIComponent(searchInput.value.trim())}`;
  }
}
```

```
  function toggleMobileMenu() {
    document.body.classList.toggle('mobile-menu-open');
  }

  function init() {
    render();

    // Ouvir eventos que exigem atualização do header
    EventBus.on('user:login', () => render());
    EventBus.on('user:logout', () => render());
  }

  return {
    init
  };
})();

export default Header;
```

2. Product Card

```
// product-card.js
import EventBus from '../core/events.js';
import { formatCurrency } from '../utils/formatter.js';

const ProductCard = (function() {
  function create(product) {
    const element = document.createElement('div');
    element.className = 'product-card';
    element.dataset.productId = product.id;

    const discount = product.price_promotional ?
      Math.round((1 - product.price_promotional / product.price) * 100) : 0;

    element.innerHTML = `
      <div class="product-card__image">
        <a href="#/product/${product.id}">
          <img
            src="${product.images && product.images.length ? product.images[0].url : '/
assets/images/placeholder.jpg'}"
            alt="${product.name}"
            loading="lazy"
          >
        </a>
        ${discount > 0 ? `<span class="product-card__discount">-${discount}%</
span>` : ''}
        ${product.new ? '<span class="product-card__badge">Novo</span>' : ''}
        <button class="product-card__wishlist" aria-label="Adicionar aos favoritos">
          <i class="icon-heart"></i>
```

```
      </button>
    </div>

    <div class="product-card__content">
      <h3 class="product-card__title">
        <a href="#/product/${product.id}">${product.name}</a>
      </h3>

      <div class="product-card__rating">
        ${generateRatingStars(product.rating || 0)}
        <span class="product-card__rating-count">(${product.rating_count || 0})</
span>
      </div>

      <div class="product-card__price">
        ${product.price_promotional ?
          `<span class="product-card__price-old">${formatCurrency(product.price)}</
span>
           <span class="product-card__price-current">$
{formatCurrency(product.price_promotional)}</span>` :
          `<span class="product-card__price-current">$
{formatCurrency(product.price)}</span>`
        }
      </div>

      <button class="product-card__add-to-cart">
        Adicionar ao Carrinho
      </button>
    </div>
  `;

  bindEvents(element, product);

  return element;
}

function generateRatingStars(rating) {
  const fullStars = Math.floor(rating);
  const halfStar = rating % 1 >= 0.5;
  const emptyStars = 5 - fullStars - (halfStar ? 1 : 0);

  let starsHtml = '';

  for (let i = 0; i < fullStars; i++) {
    starsHtml += '<i class="icon-star-full"></i>';
  }

  if (halfStar) {
    starsHtml += '<i class="icon-star-half"></i>';
  }

  for (let i = 0; i < emptyStars; i++) {
```

```
      starsHtml += '<i class="icon-star-empty"></i>';
    }

    return starsHtml;
  }

  function bindEvents(element, product) {
    const addToCartButton = element.querySelector('.product-card__add-to-cart');
    const wishlistButton = element.querySelector('.product-card__wishlist');

    if (addToCartButton) {
      addToCartButton.addEventListener('click', (event) => {
        event.preventDefault();
        EventBus.emit('cart:add', {
          product_id: product.id,
          quantity: 1
        });
      });
    }

    if (wishlistButton) {
      wishlistButton.addEventListener('click', (event) => {
        event.preventDefault();
        EventBus.emit('wishlist:toggle', product.id);
        wishlistButton.classList.toggle('active');
      });
    }
  }

  return {
    create
  };
})();

export default ProductCard;
```

3. Cart Module

```
// cart-service.js
import API from '../core/api.js';
import Store from '../core/store.js';
import EventBus from '../core/events.js';

const CartService = (function() {
  function loadCart() {
    return API.get('/cart')
      .then(response => {
        Store.setState({
          cart: {
            items: response.items || [],
```

```
        total: response.total || 0
      }
    });
    return response;
  })
  .catch(error => {
    console.error('Error loading cart:', error);
    // Fallback para carrinho local se API falhar
    const localCart = getLocalCart();
    Store.setState({ cart: localCart });
    return localCart;
  });
}

function getLocalCart() {
  try {
    const cartData = localStorage.getItem('cart');
    if (cartData) {
      return JSON.parse(cartData);
    }
  } catch (e) {
    console.error('Error parsing local cart:', e);
  }

  return { items: [], total: 0 };
}

function saveLocalCart(cart) {
  try {
    localStorage.setItem('cart', JSON.stringify(cart));
  } catch (e) {
    console.error('Error saving local cart:', e);
  }
}

function addToCart(item) {
  return API.post('/cart/items', item)
    .then(response => {
      Store.setState({
        cart: {
          items: response.items || [],
          total: response.total || 0
        }
      });
      EventBus.emit('cart:updated', Store.getState().cart);
      return response;
    })
    .catch(error => {
      console.error('Error adding to cart:', error);
      // Fallback para carrinho local se API falhar
      const cart = getLocalCart();
```

```
      // Verificar se o produto já está no carrinho
      const existingItemIndex = cart.items.findIndex(i =>
        i.product_id === item.product_id &&
        (!item.variant_id || i.variant_id === item.variant_id)
      );

      if (existingItemIndex >= 0) {
        cart.items[existingItemIndex].quantity += item.quantity;
      } else {
        cart.items.push(item);
      }

      // Recalcular total (simplificado)
      cart.total = cart.items.reduce((total, item) => total + (item.price *
item.quantity), 0);

      saveLocalCart(cart);
      Store.setState({ cart });
      EventBus.emit('cart:updated', cart);
      return cart;
    });
  }

  function updateCartItem(itemId, quantity) {
    return API.put(`/cart/items/${itemId}`, { quantity })
      .then(response => {
        Store.setState({
          cart: {
            items: response.items || [],
            total: response.total || 0
          }
        });
        EventBus.emit('cart:updated', Store.getState().cart);
        return response;
      })
      .catch(error => {
        console.error('Error updating cart item:', error);
        // Fallback para carrinho local
        const cart = getLocalCart();
        const itemIndex = cart.items.findIndex(item => item.id === itemId);

        if (itemIndex >= 0) {
          if (quantity <= 0) {
            cart.items.splice(itemIndex, 1);
          } else {
            cart.items[itemIndex].quantity = quantity;
          }

          // Recalcular total
          cart.total = cart.items.reduce((total, item) => total + (item.price *
item.quantity), 0);
```

```
        saveLocalCart(cart);
        Store.setState({ cart });
        EventBus.emit('cart:updated', cart);
      }

      return cart;
    });
}

function removeCartItem(itemId) {
  return updateCartItem(itemId, 0);
}

function clearCart() {
  return API.delete('/cart')
    .then(response => {
      const emptyCart = { items: [], total: 0 };
      Store.setState({ cart: emptyCart });
      saveLocalCart(emptyCart);
      EventBus.emit('cart:updated', emptyCart);
      return response;
    })
    .catch(error => {
      console.error('Error clearing cart:', error);
      // Fallback para carrinho local
      const emptyCart = { items: [], total: 0 };
      saveLocalCart(emptyCart);
      Store.setState({ cart: emptyCart });
      EventBus.emit('cart:updated', emptyCart);
      return emptyCart;
    });
}

function init() {
  // Carregar carrinho ao inicializar
  loadCart();

  // Configurar listeners de eventos
  EventBus.on('cart:add', item => addToCart(item));
  EventBus.on('cart:update', ({ id, quantity }) => updateCartItem(id, quantity));
  EventBus.on('cart:remove', id => removeCartItem(id));
  EventBus.on('cart:clear', () => clearCart());
  EventBus.on('user:login', () => loadCart());
}

return {
  init,
  loadCart,
  addToCart,
  updateCartItem,
  removeCartItem,
  clearCart
```

```
  };
})();

export default CartService;
```

Páginas Principais

1. Home Page

```
// home.js
import API from '../core/api.js';
import ProductCard from '../components/product/product-card.js';
import Carousel from '../components/common/carousel.js';

const HomePage = (function() {
  function render() {
    const mainContent = document.getElementById('main-content');
    if (!mainContent) return;

    mainContent.innerHTML = `
      <section class="hero-banner">
        <div class="container">
          <div class="hero-carousel" id="hero-carousel">
            <div class="hero-carousel__loading">Carregando...</div>
          </div>
        </div>
      </section>

      <section class="featured-categories">
        <div class="container">
          <h2 class="section-title">Categorias em Destaque</h2>
          <div class="categories-grid" id="featured-categories">
            <div class="categories-grid__loading">Carregando...</div>
          </div>
        </div>
      </section>

      <section class="featured-products">
        <div class="container">
          <h2 class="section-title">Produtos em Destaque</h2>
          <div class="products-grid" id="featured-products">
            <div class="products-grid__loading">Carregando...</div>
          </div>
        </div>
      </section>

      <section class="new-arrivals">
        <div class="container">
```

```
        <h2 class="section-title">Novidades</h2>
        <div class="products-carousel" id="new-arrivals-carousel">
          <div class="products-carousel__loading">Carregando...</div>
        </div>
      </div>
    </section>

    <section class="special-offers">
      <div class="container">
        <h2 class="section-title">Ofertas Especiais</h2>
        <div class="products-grid" id="special-offers">
          <div class="products-grid__loading">Carregando...</div>
        </div>
      </div>
    </section>

    <section class="testimonials">
      <div class="container">
        <h2 class="section-title">O que nossos clientes dizem</h2>
        <div class="testimonials-carousel" id="testimonials-carousel">
          <div class="testimonials-carousel__loading">Carregando...</div>
        </div>
      </div>
    </section>

    <section class="newsletter">
      <div class="container">
        <div class="newsletter-box">
          <h3>Fique por dentro das novidades</h3>
          <p>Cadastre-se para receber ofertas exclusivas e novidades.</p>
          <form id="newsletter-form" class="newsletter-form">
            <input type="email" placeholder="Seu melhor e-mail" required>
            <button type="submit">Cadastrar</button>
          </form>
        </div>
      </div>
    </section>
  `;

  loadData();
  bindEvents();
}

function loadData() {
  // Carregar banners
  API.get('/banners?location=home')
    .then(banners => {
      renderBanners(banners);
    })
    .catch(error => {
      console.error('Error loading banners:', error);
    });
```

```
  // Carregar categorias em destaque
  API.get('/categories/featured')
    .then(categories => {
      renderFeaturedCategories(categories);
    })
    .catch(error => {
      console.error('Error loading featured categories:', error);
    });

  // Carregar produtos em destaque
  API.get('/products/featured')
    .then(products => {
      renderFeaturedProducts(products);
    })
    .catch(error => {
      console.error('Error loading featured products:', error);
    });

  // Carregar novidades
  API.get('/products/new')
    .then(products => {
      renderNewArrivals(products);
    })
    .catch(error => {
      console.error('Error loading new arrivals:', error);
    });

  // Carregar ofertas especiais
  API.get('/products/offers')
    .then(products => {
      renderSpecialOffers(products);
    })
    .catch(error => {
      console.error('Error loading special offers:', error);
    });

  // Carregar depoimentos
  API.get('/testimonials')
    .then(testimonials => {
      renderTestimonials(testimonials);
    })
    .catch(error => {
      console.error('Error loading testimonials:', error);
    });
}

function renderBanners(banners) {
  const carouselElement = document.getElementById('hero-carousel');
  if (!carouselElement) return;

  if (banners && banners.length) {
```

```
    const slides = banners.map(banner => `
      <div class="hero-carousel__slide">
        <a href="${banner.link || '#'}">
          <img
            src="${banner.image}"
            alt="${banner.title || 'Banner promocional'}"
            class="hero-carousel__image"
          >
          <div class="hero-carousel__content">
            ${banner.title ? `<h2 class="hero-carousel__title">${banner.title}</h2>` : ''}
            ${banner.subtitle ? `<p class="hero-carousel__subtitle">${banner.subtitle}</
p>` : ''}
            ${banner.button_text ? `<span class="btn">${banner.button_text}</
span>` : ''}
          </div>
        </a>
      </div>
    `).join('');

    carouselElement.innerHTML = slides;

    // Inicializar carrossel
    Carousel.init(carouselElement, {
      autoplay: true,
      interval: 5000,
      indicators: true,
      controls: true
    });
  } else {
    carouselElement.innerHTML = '<div class="hero-carousel__empty">Nenhum
banner disponível</div>';
  }
 }

 function renderFeaturedCategories(categories) {
  const categoriesElement = document.getElementById('featured-categories');
  if (!categoriesElement) return;

  if (categories && categories.length) {
    categoriesElement.innerHTML = categories.map(category => `
      <a href="#/category/${category.id}" class="category-card">
        <div class="category-card__image">
          <img src="${category.image}" alt="${category.name}">
        </div>
        <h3 class="category-card__title">${category.name}</h3>
      </a>
    `).join('');
  } else {
    categoriesElement.innerHTML = '<div class="categories-
grid__empty">Nenhuma categoria em destaque</div>';
  }
 }
```

```
function renderFeaturedProducts(products) {
  const productsElement = document.getElementById('featured-products');
  if (!productsElement) return;

  if (products && products.length) {
    productsElement.innerHTML = '';
    products.forEach(product => {
      productsElement.appendChild(ProductCard.create(product));
    });
  } else {
    productsElement.innerHTML = '<div class="products-grid__empty">Nenhum
produto em destaque</div>';
  }
}

function renderNewArrivals(products) {
  const carouselElement = document.getElementById('new-arrivals-carousel');
  if (!carouselElement) return;

  if (products && products.length) {
    carouselElement.innerHTML = '';

    const carouselTrack = document.createElement('div');
    carouselTrack.className = 'products-carousel__track';

    products.forEach(product => {
      const slide = document.createElement('div');
      slide.className = 'products-carousel__slide';
      slide.appendChild(ProductCard.create(product));
      carouselTrack.appendChild(slide);
    });

    carouselElement.appendChild(carouselTrack);

    // Inicializar carrossel
    Carousel.init(carouselElement, {
      slidesToShow: 4,
      slidesToScroll: 1,
      autoplay: true,
      interval: 5000,
      responsive: [
        {
          breakpoint: 1200,
          settings: {
            slidesToShow: 3
          }
        },
        {
          breakpoint: 768,
          settings: {
            slidesToShow: 2
```

```
          }
        },
        {
          breakpoint: 576,
          settings: {
            slidesToShow: 1
          }
        }
      ]
    });
  } else {
    carouselElement.innerHTML = '<div class="products-carousel__empty">Nenhuma novidade disponível</div>';
  }
}

function renderSpecialOffers(products) {
  const productsElement = document.getElementById('special-offers');
  if (!productsElement) return;

  if (products && products.length) {
    productsElement.innerHTML = '';
    products.forEach(product => {
      productsElement.appendChild(ProductCard.create(product));
    });
  } else {
    productsElement.innerHTML = '<div class="products-grid__empty">Nenhuma oferta especial disponível</div>';
  }
}

function renderTestimonials(testimonials) {
  const carouselElement = document.getElementById('testimonials-carousel');
  if (!carouselElement) return;

  if (testimonials && testimonials.length) {
    const slides = testimonials.map(testimonial => `
      <div class="testimonial-card">
        <div class="testimonial-card__rating">
          ${Array(5).fill(0).map((_, i) =>
            i < testimonial.rating ?
              '<i class="icon-star-full"></i>' :
              '<i class="icon-star-empty"></i>'
          ).join('')}
        </div>
        <p class="testimonial-card__text">${testimonial.text}</p>
        <div class="testimonial-card__author">
          <div class="testimonial-card__avatar">
            ${testimonial.avatar ?
              `<img src="${testimonial.avatar}" alt="${testimonial.name}">` :
              `<div class="testimonial-card__avatar-placeholder">${testimonial.name.charAt(0)}</div>`
```

```
          }
          </div>
          <div class="testimonial-card__info">
            <h4 class="testimonial-card__name">${testimonial.name}</h4>
            ${testimonial.location ? `<p class="testimonial-card__location">$
{testimonial.location}</p>` : ''}
          </div>
        </div>
      </div>
    `).join('');

    carouselElement.innerHTML = slides;

    // Inicializar carrossel
    Carousel.init(carouselElement, {
      slidesToShow: 3,
      slidesToScroll: 1,
      autoplay: true,
      interval: 6000,
      responsive: [
        {
          breakpoint: 992,
          settings: {
            slidesToShow: 2
          }
        },
        {
          breakpoint: 576,
          settings: {
            slidesToShow: 1
          }
        }
      ]
    });
  } else {
    carouselElement.innerHTML = '<div class="testimonials-
carousel__empty">Nenhum depoimento disponível</div>';
  }
 }

 function bindEvents() {
  const newsletterForm = document.getElementById('newsletter-form');

  if (newsletterForm) {
    newsletterForm.addEventListener('submit', function(event) {
      event.preventDefault();

      const emailInput = this.querySelector('input[type="email"]');
      if (emailInput && emailInput.value.trim()) {
        API.post('/newsletter/subscribe', { email: emailInput.value.trim() })
          .then(() => {
            alert('E-mail cadastrado com sucesso!');
```

```
            emailInput.value = '';
          })
          .catch(error => {
            console.error('Error subscribing to newsletter:', error);
            alert('Erro ao cadastrar e-mail. Por favor, tente novamente.');
          });
      }
    });
  }
 }

 return {
  render
 };
})();

export default HomePage;
```

2. Product Detail Page

```
// product.js
import API from '../core/api.js';
import EventBus from '../core/events.js';
import { formatCurrency } from '../utils/formatter.js';
import ProductGallery from '../components/product/product-gallery.js';
import ProductReview from '../components/product/product-review.js';

const ProductPage = (function() {
 let currentProduct = null;
 let selectedVariant = null;

 function render(productId) {
  const mainContent = document.getElementById('main-content');
  if (!mainContent) return;

  // Mostrar loader
  mainContent.innerHTML = `
   <div class="product-detail-loading">
    <div class="loader"></div>
    <p>Carregando produto...</p>
   </div>
  `;

  // Carregar dados do produto
  API.get(`/products/${productId}`)
   .then(product => {
    currentProduct = product;
    renderProductDetail(mainContent, product);
    loadRelatedProducts(product.id, product.category_id);
   })
```

```
    .catch(error => {
      console.error('Error loading product:', error);
      mainContent.innerHTML = `
        <div class="product-detail-error">
          <h2>Produto não encontrado</h2>
          <p>Desculpe, não foi possível carregar o produto solicitado.</p>
          <a href="#/" class="btn">Voltar para a página inicial</a>
        </div>
      `;
    });
}

function renderProductDetail(container, product) {
  // Estrutura básica da página
  container.innerHTML = `
    <div class="product-detail">
      <div class="container">
        <div class="breadcrumbs">
          <a href="#/">Home</a> &gt;
          <a href="#/category/${product.category_id}">${product.category_name}</
a> &gt;
          <span>${product.name}</span>
        </div>

        <div class="product-detail__grid">
          <div class="product-detail__gallery" id="product-gallery">
            <!-- Gallery will be rendered here -->
          </div>

          <div class="product-detail__info">
            <h1 class="product-detail__title">${product.name}</h1>

            <div class="product-detail__meta">
              <div class="product-detail__rating">
                ${generateRatingStars(product.rating || 0)}
                <span class="product-detail__rating-count">${product.rating_count || 0}
avaliações</span>
              </div>

              <div class="product-detail__sku">
                SKU: <span>${product.sku}</span>
              </div>

              ${product.stock > 0 ?
                `<div class="product-detail__stock product-detail__stock--in">
                  <i class="icon-check"></i> Em estoque
                </div>` :
                `<div class="product-detail__stock product-detail__stock--out">
                  <i class="icon-close"></i> Fora de estoque
                </div>`
              }
            </div>
```

```
            <div class="product-detail__price">
              ${product.price_promotional ?
                `<span class="product-detail__price-old">$
{formatCurrency(product.price)}</span>
                 <span class="product-detail__price-current">$
{formatCurrency(product.price_promotional)}</span>
                 <span class="product-detail__price-discount">
                   ${Math.round((1 - product.price_promotional / product.price) * 100)}%
OFF
                 </span>` :
                `<span class="product-detail__price-current">$
{formatCurrency(product.price)}</span>`
              }
            </div>

            <div class="product-detail__description">
              ${product.description_short || ''}
            </div>

            ${renderVariants(product)}

            <div class="product-detail__actions">
              <div class="product-detail__quantity">
                <button class="quantity-btn quantity-btn--minus" id="quantity-minus">-
</button>
                 <input type="number" id="quantity-input" value="1" min="1" max="$
{product.stock}">
                 <button class="quantity-btn quantity-btn--plus" id="quantity-plus">+</
button>
               </div>

               <button class="btn btn--primary btn--lg product-detail__add-to-cart"
id="add-to-cart-btn"
                 ${product.stock <= 0 ? 'disabled' : ''}>
                 <i class="icon-cart"></i> Adicionar ao Carrinho
               </button>

               <button class="btn btn--icon product-detail__wishlist" id="add-to-wishlist-
btn">
                 <i class="icon-heart"></i>
               </button>
            </div>

            <div class="product-detail__shipping">
              <h3>Calcular Frete</h3>
              <div class="shipping-calculator">
                <input type="text" id="shipping-cep" placeholder="Digite seu CEP"
maxlength="9">
                 <button id="calculate-shipping-btn">Calcular</button>
               </div>
               <div id="shipping-result"></div>
```

```
          </div>

          <div class="product-detail__share">
            <span>Compartilhar:</span>
            <a href="#" class="social-icon" aria-label="Compartilhar no Facebook">
              <i class="icon-facebook"></i>
            </a>
            <a href="#" class="social-icon" aria-label="Compartilhar no Twitter">
              <i class="icon-twitter"></i>
            </a>
            <a href="#" class="social-icon" aria-label="Compartilhar no WhatsApp">
              <i class="icon-whatsapp"></i>
            </a>
            <a href="#" class="social-icon" aria-label="Compartilhar por E-mail">
              <i class="icon-mail"></i>
            </a>
          </div>
        </div>
      </div>

      <div class="product-detail__tabs">
        <div class="tabs">
          <div class="tabs__header">
            <button class="tabs__button tabs__button--active" data-
tab="description">Descrição</button>
            <button class="tabs__button" data-tab="specifications">Especificações</
button>
            <button class="tabs__button" data-tab="reviews">
              Avaliações <span class="badge">${product.rating_count || 0}</span>
            </button>
          </div>

          <div class="tabs__content">
            <div class="tabs__panel tabs__panel--active" id="tab-description">
              ${product.description_long || 'Descrição não disponível.'}
            </div>

            <div class="tabs__panel" id="tab-specifications">
              ${renderSpecifications(product)}
            </div>

            <div class="tabs__panel" id="tab-reviews">
              <div id="reviews-container">
                <div class="reviews-loading">Carregando avaliações...</div>
              </div>
            </div>
          </div>
        </div>
      </div>

      <div class="product-detail__related">
        <h2 class="section-title">Produtos Relacionados</h2>
```

```
          <div class="products-grid" id="related-products">
            <div class="products-grid__loading">Carregando...</div>
          </div>
        </div>
      </div>
    </div>
  `;

  // Inicializar galeria
  const galleryElement = document.getElementById('product-gallery');
  if (galleryElement && product.images && product.images.length) {
    ProductGallery.init(galleryElement, product.images);
  }

  // Carregar avaliações
  loadReviews(product.id);

  // Vincular eventos
  bindEvents(product);
}

function generateRatingStars(rating) {
  const fullStars = Math.floor(rating);
  const halfStar = rating % 1 >= 0.5;
  const emptyStars = 5 - fullStars - (halfStar ? 1 : 0);

  let starsHtml = '';

  for (let i = 0; i < fullStars; i++) {
    starsHtml += '<i class="icon-star-full"></i>';
  }

  if (halfStar) {
    starsHtml += '<i class="icon-star-half"></i>';
  }

  for (let i = 0; i < emptyStars; i++) {
    starsHtml += '<i class="icon-star-empty"></i>';
  }

  return starsHtml;
}

function renderVariants(product) {
  if (!product.variants || !product.variants.length) {
    return '';
  }

  // Agrupar variantes por tipo de atributo
  const attributeGroups = {};

  product.variants.forEach(variant => {
```

```
    variant.attributes.forEach(attr => {
      if (!attributeGroups[attr.name]) {
        attributeGroups[attr.name] = {
          name: attr.name,
          values: []
        };
      }

      if (!attributeGroups[attr.name].values.some(v => v.value === attr.value)) {
        attributeGroups[attr.name].values.push({
          value: attr.value,
          label: attr.label || attr.value
        });
      }
    });
  });

  // Renderizar seletores de variantes
  let variantsHtml = '<div class="product-detail__variants">';

  Object.values(attributeGroups).forEach(group => {
    variantsHtml += `
      <div class="variant-group">
        <h3 class="variant-group__title">${group.name}:</h3>
        <div class="variant-group__options" data-attribute="${group.name}">
    `;

    group.values.forEach(value => {
      variantsHtml += `
        <button class="variant-option" data-value="${value.value}">
          ${value.label}
        </button>
      `;
    });

    variantsHtml += `
        </div>
      </div>
    `;
  });

  variantsHtml += '</div>';

  return variantsHtml;
}

function renderSpecifications(product) {
  if (!product.specifications || !product.specifications.length) {
    return '<p>Especificações não disponíveis.</p>';
  }

  let specsHtml = '<table class="specifications-table">';
```

```
    product.specifications.forEach(spec => {
      specsHtml += `
        <tr>
          <th>${spec.name}</th>
          <td>${spec.value}</td>
        </tr>
      `;
    });

    specsHtml += '</table>';

    return specsHtml;
  }

  function loadReviews(productId) {
    const reviewsContainer = document.getElementById('reviews-container');
    if (!reviewsContainer) return;

    API.get(`/products/${productId}/reviews`)
      .then(reviews => {
        if (reviews && reviews.length) {
          reviewsContainer.innerHTML = `
            <div class="reviews-summary">
              <div class="reviews-summary__rating">
                <div class="reviews-summary__average">${reviews.average_rating.toFixed(1)}</div>
                <div class="reviews-summary__stars">
                  ${generateRatingStars(reviews.average_rating)}
                </div>
                <div class="reviews-summary__count">${reviews.length} avaliações</div>
              </div>

              <div class="reviews-summary__distribution">
                ${[5, 4, 3, 2, 1].map(stars => {
                  const count = reviews.filter(r => Math.round(r.rating) === stars).length;
                  const percentage = reviews.length ? Math.round((count / reviews.length) * 100) : 0;

                  return `
                    <div class="rating-bar">
                      <div class="rating-bar__stars">${stars} <i class="icon-star-full"></i></div>
                      <div class="rating-bar__bar">
                        <div class="rating-bar__fill" style="width: ${percentage}%"></div>
                      </div>
                      <div class="rating-bar__percentage">${percentage}%</div>
                    </div>
                  `;
                }).join('')}
              </div>
            </div>
```

```
        <div class="reviews-list">
          ${reviews.map(review => ProductReview.render(review)).join('')}
        </div>

        <div class="reviews-form-container">
          <h3>Deixe sua avaliação</h3>
          <form id="review-form" class="review-form">
            <div class="form-group">
              <label>Sua nota:</label>
              <div class="rating-input" id="rating-input">
                <i class="icon-star-empty" data-rating="1"></i>
                <i class="icon-star-empty" data-rating="2"></i>
                <i class="icon-star-empty" data-rating="3"></i>
                <i class="icon-star-empty" data-rating="4"></i>
                <i class="icon-star-empty" data-rating="5"></i>
              </div>
            </div>

            <div class="form-group">
              <label for="review-title">Título:</label>
              <input type="text" id="review-title" required>
            </div>

            <div class="form-group">
              <label for="review-comment">Comentário:</label>
              <textarea id="review-comment" rows="5" required></textarea>
            </div>

            <div class="form-group">
              <label for="review-name">Nome:</label>
              <input type="text" id="review-name" required>
            </div>

            <div class="form-group">
              <label for="review-email">E-mail:</label>
              <input type="email" id="review-email" required>
            </div>

            <button type="submit" class="btn btn--primary">Enviar Avaliação</
button>
          </form>
        </div>
      `;

      // Vincular eventos do formulário de avaliação
      bindReviewFormEvents(productId);
    } else {
      reviewsContainer.innerHTML = `
        <div class="reviews-empty">
          <p>Este produto ainda não possui avaliações.</p>
          <p>Seja o primeiro a avaliar!</p>
```

```
          <div class="reviews-form-container">
            <h3>Deixe sua avaliação</h3>
            <form id="review-form" class="review-form">
              <div class="form-group">
                <label>Sua nota:</label>
                <div class="rating-input" id="rating-input">
                  <i class="icon-star-empty" data-rating="1"></i>
                  <i class="icon-star-empty" data-rating="2"></i>
                  <i class="icon-star-empty" data-rating="3"></i>
                  <i class="icon-star-empty" data-rating="4"></i>
                  <i class="icon-star-empty" data-rating="5"></i>
                </div>
              </div>

              <div class="form-group">
                <label for="review-title">Título:</label>
                <input type="text" id="review-title" required>
              </div>

              <div class="form-group">
                <label for="review-comment">Comentário:</label>
                <textarea id="review-comment" rows="5" required></textarea>
              </div>

              <div class="form-group">
                <label for="review-name">Nome:</label>
                <input type="text" id="review-name" required>
              </div>

              <div class="form-group">
                <label for="review-email">E-mail:</label>
                <input type="email" id="review-email" required>
              </div>

              <button type="submit" class="btn btn--primary">Enviar Avaliação</
button>
            </form>
          </div>
        </div>
      `;

      // Vincular eventos do formulário de avaliação
      bindReviewFormEvents(productId);
    }
  })
  .catch(error => {
    console.error('Error loading reviews:', error);
    reviewsContainer.innerHTML = '<div class="reviews-error">Erro ao carregar
avaliações.</div>';
  });
}
```

```
  function loadRelatedProducts(productId, categoryId) {
    const relatedContainer = document.getElementById('related-products');
    if (!relatedContainer) return;

    API.get(`/products/related?product_id=${productId}&category_id=${categoryId}`)
      .then(products => {
        if (products && products.length) {
          relatedContainer.innerHTML = '';

          products.forEach(product => {
            const productCard = document.createElement('div');
            productCard.className = 'product-card';
            productCard.innerHTML = `
              <div class="product-card__image">
                <a href="#/product/${product.id}">
                  <img
                    src="${product.images && product.images.length ?
product.images[0].url : '/assets/images/placeholder.jpg'}"
                    alt="${product.name}"
                    loading="lazy"
                  >
                </a>
                ${product.price_promotional ?
                  `<span class="product-card__discount">
                    -${Math.round((1 - product.price_promotional / product.price) * 100)}%
                  </span>` : ''
                }
                <button class="product-card__wishlist" aria-label="Adicionar aos
favoritos">
                  <i class="icon-heart"></i>
                </button>
              </div>

              <div class="product-card__content">
                <h3 class="product-card__title">
                  <a href="#/product/${product.id}">${product.name}</a>
                </h3>

                <div class="product-card__rating">
                  ${generateRatingStars(product.rating || 0)}
                  <span class="product-card__rating-count">(${product.rating_count ||
0})</span>
                </div>

                <div class="product-card__price">
                  ${product.price_promotional ?
                    `<span class="product-card__price-old">$
{formatCurrency(product.price)}</span>
                     <span class="product-card__price-current">$
{formatCurrency(product.price_promotional)}</span>` :
                    `<span class="product-card__price-current">$
```

```
{formatCurrency(product.price)}</span>`
                }
              </div>

              <button class="product-card__add-to-cart" data-product-id="$
{product.id}">
                Adicionar ao Carrinho
              </button>
            </div>
          `;

          relatedContainer.appendChild(productCard);
        });

        // Vincular eventos dos cards de produtos relacionados
        bindRelatedProductsEvents();
      } else {
        relatedContainer.innerHTML = '<div class="products-grid__empty">Nenhum
produto relacionado encontrado.</div>';
      }
    })
    .catch(error => {
      console.error('Error loading related products:', error);
      relatedContainer.innerHTML = '<div class="products-grid__error">Erro ao
carregar produtos relacionados.</div>';
    });
 }

 function bindEvents(product) {
  // Tabs
  const tabButtons = document.querySelectorAll('.tabs__button');
  tabButtons.forEach(button => {
    button.addEventListener('click', function() {
      const tabId = this.dataset.tab;

      // Remover classe ativa de todos os botões e painéis
      document.querySelectorAll('.tabs__button').forEach(btn => {
        btn.classList.remove('tabs__button--active');
      });

      document.querySelectorAll('.tabs__panel').forEach(panel => {
        panel.classList.remove('tabs__panel--active');
      });

      // Adicionar classe ativa ao botão e painel clicados
      this.classList.add('tabs__button--active');
      document.getElementById(`tab-${tabId}`).classList.add('tabs__panel--active');
    });
  });

  // Controle de quantidade
  const quantityInput = document.getElementById('quantity-input');
```

```
  const minusBtn = document.getElementById('quantity-minus');
  const plusBtn = document.getElementById('quantity-plus');

  if (minusBtn && plusBtn && quantityInput) {
    minusBtn.addEventListener('click', function() {
      const currentValue = parseInt(quantityInput.value);
      if (currentValue > 1) {
        quantityInput.value = currentValue - 1;
      }
    });

    plusBtn.addEventListener('click', function() {
      const currentValue = parseInt(quantityInput.value);
      const maxValue = parseInt(quantityInput.getAttribute('max'));

      if (currentValue < maxValue) {
        quantityInput.value = currentValue + 1;
      }
    });

    quantityInput.addEventListener('change', function() {
      const currentValue = parseInt(this.value);
      const maxValue = parseInt(this.getAttribute('max'));

      if (isNaN(currentValue) || currentValue < 1) {
        this.value = 1;
      } else if (currentValue > maxValue) {
        this.value = maxValue;
      }
    });
  }

  // Seleção de variantes
  const variantOptions = document.querySelectorAll('.variant-option');
  variantOptions.forEach(option => {
    option.addEventListener('click', function() {
      const attributeName = this.parentElement.dataset.attribute;
      const attributeValue = this.dataset.value;

      // Remover classe ativa de todas as opções do mesmo grupo
      this.parentElement.querySelectorAll('.variant-option').forEach(opt => {
        opt.classList.remove('variant-option--active');
      });

      // Adicionar classe ativa à opção clicada
      this.classList.add('variant-option--active');

      // Atualizar variante selecionada
      updateSelectedVariant();
    });
  });
```

```
// Adicionar ao carrinho
const addToCartBtn = document.getElementById('add-to-cart-btn');
if (addToCartBtn) {
  addToCartBtn.addEventListener('click', function() {
    if (product.stock <= 0) return;

    const quantity = parseInt(quantityInput.value) || 1;

    const cartItem = {
      product_id: product.id,
      variant_id: selectedVariant ? selectedVariant.id : null,
      quantity: quantity
    };

    EventBus.emit('cart:add', cartItem);

    // Mostrar confirmação
    const toast = document.createElement('div');
    toast.className = 'toast toast--success';
    toast.innerHTML = `
      <div class="toast__icon">
        <i class="icon-check"></i>
      </div>
      <div class="toast__content">
        <p>Produto adicionado ao carrinho!</p>
        <a href="#/cart" class="toast__link">Ver carrinho</a>
      </div>
      <button class="toast__close" aria-label="Fechar">
        <i class="icon-close"></i>
      </button>
    `;

    document.body.appendChild(toast);

    // Remover toast após 5 segundos
    setTimeout(() => {
      toast.classList.add('toast--hide');
      setTimeout(() => {
        document.body.removeChild(toast);
      }, 300);
    }, 5000);

    // Vincular evento de fechar toast
    toast.querySelector('.toast__close').addEventListener('click', function() {
      toast.classList.add('toast--hide');
      setTimeout(() => {
        document.body.removeChild(toast);
      }, 300);
    });
  });
}
```

```
    // Adicionar aos favoritos
    const wishlistBtn = document.getElementById('add-to-wishlist-btn');
    if (wishlistBtn) {
      wishlistBtn.addEventListener('click', function() {
        EventBus.emit('wishlist:toggle', product.id);
        this.classList.toggle('active');

        // Mostrar confirmação
        const toast = document.createElement('div');
        toast.className = 'toast toast--info';
        toast.innerHTML = `
          <div class="toast__icon">
            <i class="icon-heart"></i>
          </div>
          <div class="toast__content">
            <p>${this.classList.contains('active') ?
              'Produto adicionado aos favoritos!' :
              'Produto removido dos favoritos!'}</p>
            <a href="#/wishlist" class="toast__link">Ver favoritos</a>
          </div>
          <button class="toast__close" aria-label="Fechar">
            <i class="icon-close"></i>
          </button>
        `;

        document.body.appendChild(toast);

        // Remover toast após 5 segundos
        setTimeout(() => {
          toast.classList.add('toast--hide');
          setTimeout(() => {
            document.body.removeChild(toast);
          }, 300);
        }, 5000);

        // Vincular evento de fechar toast
        toast.querySelector('.toast__close').addEventListener('click', function() {
          toast.classList.add('toast--hide');
          setTimeout(() => {
            document.body.removeChild(toast);
          }, 300);
        });
      });
    }

    // Cálculo de frete
    const calculateShippingBtn = document.getElementById('calculate-shipping-
btn');
    if (calculateShippingBtn) {
      calculateShippingBtn.addEventListener('click', function() {
        const cepInput = document.getElementById('shipping-cep');
        const resultContainer = document.getElementById('shipping-result');
```

```
      if (cepInput && resultContainer && cepInput.value.trim()) {
        const cep = cepInput.value.trim().replace(/\D/g, '');

        if (cep.length !== 8) {
          resultContainer.innerHTML = '<div class="shipping-error">CEP inválido. Digite um CEP com 8 dígitos.</div>';
          return;
        }

        resultContainer.innerHTML = '<div class="shipping-loading">Calculando...</div>';

        API.get(`/shipping/calculate?cep=${cep}&product_id=${product.id}`)
          .then(options => {
            if (options && options.length) {
              let html = '<div class="shipping-options">';

              options.forEach(option => {
                html += `
                  <div class="shipping-option">
                    <div class="shipping-option__name">${option.name}</div>
                    <div class="shipping-option__price">${formatCurrency(option.price)}</div>
                    <div class="shipping-option__time">${option.time} dias úteis</div>
                  </div>
                `;
              });

              html += '</div>';
              resultContainer.innerHTML = html;
            } else {
              resultContainer.innerHTML = '<div class="shipping-error">Não foi possível calcular o frete para este CEP.</div>';
            }
          })
          .catch(error => {
            console.error('Error calculating shipping:', error);
            resultContainer.innerHTML = '<div class="shipping-error">Erro ao calcular o frete. Tente novamente.</div>';
          });
      }
    });
  }

  // Compartilhamento
  const shareLinks = document.querySelectorAll('.product-detail__share .social-icon');
  shareLinks.forEach(link => {
    link.addEventListener('click', function(event) {
      event.preventDefault();
```

```
      const url = encodeURIComponent(window.location.href);
      const title = encodeURIComponent(product.name);

      let shareUrl = '';

      if (this.querySelector('.icon-facebook')) {
        shareUrl = `https://www.facebook.com/sharer/sharer.php?u=${url}`;
      } else if (this.querySelector('.icon-twitter')) {
        shareUrl = `https://twitter.com/intent/tweet?url=${url}&text=${title}`;
      } else if (this.querySelector('.icon-whatsapp')) {
        shareUrl = `https://api.whatsapp.com/send?text=${title}%20${url}`;
      } else if (this.querySelector('.icon-mail')) {
        shareUrl = `mailto:?subject=${title}&body=${url}`;
      }

      if (shareUrl) {
        window.open(shareUrl, '_blank', 'width=600,height=400');
      }
    });
  });
}

function bindReviewFormEvents(productId) {
  const ratingInput = document.getElementById('rating-input');
  const reviewForm = document.getElementById('review-form');

  let selectedRating = 0;

  if (ratingInput) {
    const stars = ratingInput.querySelectorAll('i');

    stars.forEach(star => {
      // Hover
      star.addEventListener('mouseenter', function() {
        const rating = parseInt(this.dataset.rating);

        stars.forEach((s, index) => {
          if (index < rating) {
            s.className = 'icon-star-full';
          } else {
            s.className = 'icon-star-empty';
          }
        });
      });

      // Click
      star.addEventListener('click', function() {
        selectedRating = parseInt(this.dataset.rating);

        stars.forEach((s, index) => {
          if (index < selectedRating) {
            s.className = 'icon-star-full';
```

```
          } else {
            s.className = 'icon-star-empty';
          }
        });
      });
    });

    // Mouse leave
    ratingInput.addEventListener('mouseleave', function() {
      stars.forEach((s, index) => {
        if (index < selectedRating) {
          s.className = 'icon-star-full';
        } else {
          s.className = 'icon-star-empty';
        }
      });
    });
  }

  if (reviewForm) {
    reviewForm.addEventListener('submit', function(event) {
      event.preventDefault();

      if (selectedRating === 0) {
        alert('Por favor, selecione uma nota para o produto.');
        return;
      }

      const titleInput = document.getElementById('review-title');
      const commentInput = document.getElementById('review-comment');
      const nameInput = document.getElementById('review-name');
      const emailInput = document.getElementById('review-email');

      const reviewData = {
        product_id: productId,
        rating: selectedRating,
        title: titleInput.value.trim(),
        comment: commentInput.value.trim(),
        name: nameInput.value.trim(),
        email: emailInput.value.trim()
      };

      API.post('/products/reviews', reviewData)
        .then(response => {
          alert('Sua avaliação foi enviada com sucesso e será publicada após
moderação.');

          // Limpar formulário
          selectedRating = 0;
          reviewForm.reset();

          const stars = ratingInput.querySelectorAll('i');
```

```
          stars.forEach(s => {
            s.className = 'icon-star-empty';
          });
        })
        .catch(error => {
          console.error('Error submitting review:', error);
          alert('Erro ao enviar avaliação. Por favor, tente novamente.');
        });
    });
  }
}

function bindRelatedProductsEvents() {
  const addToCartButtons = document.querySelectorAll('.product-card__add-to-cart');
  const wishlistButtons = document.querySelectorAll('.product-card__wishlist');

  addToCartButtons.forEach(button => {
    button.addEventListener('click', function() {
      const productId = this.dataset.productId;

      EventBus.emit('cart:add', {
        product_id: productId,
        quantity: 1
      });

      // Mostrar confirmação
      const toast = document.createElement('div');
      toast.className = 'toast toast--success';
      toast.innerHTML = `
        <div class="toast__icon">
          <i class="icon-check"></i>
        </div>
        <div class="toast__content">
          <p>Produto adicionado ao carrinho!</p>
          <a href="#/cart" class="toast__link">Ver carrinho</a>
        </div>
        <button class="toast__close" aria-label="Fechar">
          <i class="icon-close"></i>
        </button>
      `;

      document.body.appendChild(toast);

      // Remover toast após 5 segundos
      setTimeout(() => {
        toast.classList.add('toast--hide');
        setTimeout(() => {
          document.body.removeChild(toast);
        }, 300);
      }, 5000);
```

```
      // Vincular evento de fechar toast
      toast.querySelector('.toast__close').addEventListener('click', function() {
        toast.classList.add('toast--hide');
        setTimeout(() => {
          document.body.removeChild(toast);
        }, 300);
      });
    });
  });

  wishlistButtons.forEach(button => {
    button.addEventListener('click', function() {
      const productCard = this.closest('.product-card');
      const productId = productCard.querySelector('.product-card__add-to-
cart').dataset.productId;

      EventBus.emit('wishlist:toggle', productId);
      this.classList.toggle('active');
    });
  });
 }

 function updateSelectedVariant() {
  if (!currentProduct || !currentProduct.variants || !
currentProduct.variants.length) {
    return;
  }

  // Obter atributos selecionados
  const selectedAttributes = {};
  document.querySelectorAll('.variant-group').forEach(group => {
    const attributeName = group.querySelector('.variant-
group__options').dataset.attribute;
    const selectedOption = group.querySelector('.variant-option--active');

    if (selectedOption) {
      selectedAttributes[attributeName] = selectedOption.dataset.value;
    }
  });

  // Encontrar variante correspondente
  const matchingVariant = currentProduct.variants.find(variant => {
    return variant.attributes.every(attr =>
      selectedAttributes[attr.name] === attr.value
    );
  });

  if (matchingVariant) {
    selectedVariant = matchingVariant;

    // Atualizar preço
    const priceElement = document.querySelector('.product-detail__price');
```

```
    if (priceElement) {
      if (matchingVariant.price_promotional) {
        priceElement.innerHTML = `
          <span class="product-detail__price-old">$
{formatCurrency(matchingVariant.price)}</span>
          <span class="product-detail__price-current">$
{formatCurrency(matchingVariant.price_promotional)}</span>
          <span class="product-detail__price-discount">
            ${Math.round((1 - matchingVariant.price_promotional /
matchingVariant.price) * 100)}% OFF
          </span>
        `;
      } else {
        priceElement.innerHTML = `
          <span class="product-detail__price-current">$
{formatCurrency(matchingVariant.price)}</span>
        `;
      }
    }

    // Atualizar estoque
    const stockElement = document.querySelector('.product-detail__stock');
    const addToCartBtn = document.getElementById('add-to-cart-btn');
    const quantityInput = document.getElementById('quantity-input');

    if (stockElement) {
      if (matchingVariant.stock > 0) {
        stockElement.className = 'product-detail__stock product-detail__stock--in';
        stockElement.innerHTML = '<i class="icon-check"></i> Em estoque';

        if (addToCartBtn) {
          addToCartBtn.disabled = false;
        }

        if (quantityInput) {
          quantityInput.max = matchingVariant.stock;
          if (parseInt(quantityInput.value) > matchingVariant.stock) {
            quantityInput.value = matchingVariant.stock;
          }
        }
      } else {
        stockElement.className = 'product-detail__stock product-detail__stock--out';
        stockElement.innerHTML = '<i class="icon-close"></i> Fora de estoque';

        if (addToCartBtn) {
          addToCartBtn.disabled = true;
        }

        if (quantityInput) {
          quantityInput.max = 0;
          quantityInput.value = 0;
        }
```

```
      }
    }

    // Atualizar imagem principal se a variante tiver imagem própria
    if (matchingVariant.image) {
      const galleryElement = document.getElementById('product-gallery');
      if (galleryElement) {
        const mainImage = galleryElement.querySelector('.product-gallery__main
img');
        if (mainImage) {
          mainImage.src = matchingVariant.image;
        }
      }
    }
  }
 }

 return {
  render
 };
})();

export default ProductPage;
```

Considerações de Segurança

1. Validação de Dados

Implementaremos validação rigorosa de todos os dados de entrada:

```
// validator.js
const Validator = (function() {
 function validateEmail(email) {
  const re = /^(([^<>()\[\]\\.,;:\s@"]+(\.[^<>()\[\]\\.,;:\s@"]+)*)|(".+"))@((\[[0-9]{1,3}\.
[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}])|(([a-zA-Z\-0-9]+\.)+[a-zA-Z]{2,}))$/;
  return re.test(String(email).toLowerCase());
 }

 function validatePassword(password) {
  // Mínimo 8 caracteres, pelo menos uma letra maiúscula, uma minúscula e um
número
  const re = /^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)[a-zA-Z\d]{8,}$/;
  return re.test(String(password));
 }

 function validateCPF(cpf) {
  cpf = cpf.replace(/[^\d]/g, '');
```

```
  if (cpf.length !== 11) return false;

  // Verificar se todos os dígitos são iguais
  if (/^(\d)\1+$/.test(cpf)) return false;

  // Validação do primeiro dígito verificador
  let sum = 0;
  for (let i = 0; i < 9; i++) {
    sum += parseInt(cpf.charAt(i)) * (10 - i);
  }

  let remainder = sum % 11;
  let digit1 = remainder < 2 ? 0 : 11 - remainder;

  if (parseInt(cpf.charAt(9)) !== digit1) return false;

  // Validação do segundo dígito verificador
  sum = 0;
  for (let i = 0; i < 10; i++) {
    sum += parseInt(cpf.charAt(i)) * (11 - i);
  }

  remainder = sum % 11;
  let digit2 = remainder < 2 ? 0 : 11 - remainder;

  return parseInt(cpf.charAt(10)) === digit2;
}

function validateCEP(cep) {
  const re = /^\d{5}-?\d{3}$/;
  return re.test(String(cep));
}

function validatePhone(phone) {
  const re = /^\(?(\d{2})\)?[-. ]?(\d{4,5})[-. ]?(\d{4})$/;
  return re.test(String(phone));
}

function sanitizeHTML(text) {
  const element = document.createElement('div');
  element.textContent = text;
  return element.innerHTML;
}

return {
  validateEmail,
  validatePassword,
  validateCPF,
  validateCEP,
  validatePhone,
  sanitizeHTML
};
```

```
})();

export default Validator;
```

2. Proteção contra XSS

Implementaremos medidas para prevenir ataques de Cross-Site Scripting:

- Sanitização de todos os dados de entrada
- Uso de `textContent` em vez de `innerHTML` quando possível
- Implementação de Content Security Policy (CSP)

3. Proteção contra CSRF

Para prevenir ataques de Cross-Site Request Forgery:

- Uso de tokens CSRF em formulários
- Verificação de origem das requisições
- Implementação de SameSite cookies

4. Segurança de Dados Sensíveis

- Nunca armazenar dados de cartão de crédito no frontend
- Uso de HTTPS para todas as comunicações
- Implementação de timeout para sessões inativas

Estratégias de Performance

1. Lazy Loading

Implementaremos carregamento sob demanda para imagens e componentes:

```
// Lazy loading de imagens
document.addEventListener('DOMContentLoaded', function() {
  const lazyImages = document.querySelectorAll('img[loading="lazy"]');

  if ('IntersectionObserver' in window) {
    const imageObserver = new IntersectionObserver(function(entries, observer) {
      entries.forEach(function(entry) {
        if (entry.isIntersecting) {
          const lazyImage = entry.target;
          lazyImage.src = lazyImage.dataset.src;
          lazyImage.removeAttribute('loading');
          imageObserver.unobserve(lazyImage);
```

```
        }
      });
    });

    lazyImages.forEach(function(lazyImage) {
      imageObserver.observe(lazyImage);
    });
  } else {
    // Fallback para navegadores que não suportam IntersectionObserver
    let active = false;

    const lazyLoad = function() {
      if (active === false) {
        active = true;

        setTimeout(function() {
          lazyImages.forEach(function(lazyImage) {
            if ((lazyImage.getBoundingClientRect().top <= window.innerHeight &&
lazyImage.getBoundingClientRect().bottom >= 0) &&
getComputedStyle(lazyImage).display !== 'none') {
              lazyImage.src = lazyImage.dataset.src;
              lazyImage.removeAttribute('loading');

              lazyImages = lazyImages.filter(function(image) {
                return image !== lazyImage;
              });

              if (lazyImages.length === 0) {
                document.removeEventListener('scroll', lazyLoad);
                window.removeEventListener('resize', lazyLoad);
                window.removeEventListener('orientationchange', lazyLoad);
              }
            }
          });

          active = false;
        }, 200);
      }
    };

    document.addEventListener('scroll', lazyLoad);
    window.addEventListener('resize', lazyLoad);
    window.addEventListener('orientationchange', lazyLoad);
  }
});
```

2. Debounce e Throttle

Para otimizar eventos frequentes como scroll e resize:

```
// utils/performance.js
const Performance = (function() {
  function debounce(func, wait, immediate) {
    let timeout;

    return function() {
      const context = this;
      const args = arguments;

      const later = function() {
        timeout = null;
        if (!immediate) func.apply(context, args);
      };

      const callNow = immediate && !timeout;

      clearTimeout(timeout);
      timeout = setTimeout(later, wait);

      if (callNow) func.apply(context, args);
    };
  }

  function throttle(func, limit) {
    let inThrottle;

    return function() {
      const context = this;
      const args = arguments;

      if (!inThrottle) {
        func.apply(context, args);
        inThrottle = true;

        setTimeout(() => {
          inThrottle = false;
        }, limit);
      }
    };
  }

  return {
    debounce,
    throttle
  };
})();

export default Performance;
```

3. Caching

Implementaremos estratégias de cache para reduzir requisições:

```
// cache.js
const Cache = (function() {
  const cache = {};

  function set(key, data, ttl = 60000) { // TTL em milissegundos, padrão 1 minuto
    cache[key] = {
      data,
      expiry: Date.now() + ttl
    };
  }

  function get(key) {
    const item = cache[key];

    if (!item) return null;

    if (Date.now() > item.expiry) {
      delete cache[key];
      return null;
    }

    return item.data;
  }

  function remove(key) {
    delete cache[key];
  }

  function clear() {
    Object.keys(cache).forEach(key => {
      delete cache[key];
    });
  }

  return {
    set,
    get,
    remove,
    clear
  };
})();

export default Cache;
```

Conclusão

O planejamento do frontend com JavaScript puro para o e-commerce foi projetado seguindo as melhores práticas de desenvolvimento web moderno, com foco em modularidade, performance, segurança e experiência do usuário.

A arquitetura baseada em componentes, com gerenciamento de estado centralizado e sistema de eventos, permitirá um desenvolvimento organizado e escalável, mesmo utilizando JavaScript puro conforme solicitado pelo cliente.

Os componentes e páginas detalhados neste documento cobrem as principais funcionalidades esperadas de um e-commerce completo, incluindo:

1. Navegação intuitiva e responsiva
2. Exibição atrativa de produtos
3. Detalhamento completo de produtos com variantes
4. Sistema de avaliações
5. Carrinho de compras funcional
6. Cálculo de frete
7. Compartilhamento em redes sociais
8. Lista de desejos

As considerações de segurança e performance garantirão que a aplicação seja não apenas funcional, mas também segura e rápida, proporcionando uma experiência de usuário de alta qualidade em todos os dispositivos.