Démo des Fonctionnalités JavaScript

Ce document présente les principales fonctionnalités JavaScript de la boutique en ligne de montres.

1. Recherche en temps réel (AJAX)

Fonctionnalité

Permet aux utilisateurs de rechercher des produits sans recharger la page, avec affichage des résultats au fur et à mesure de la saisie.

Code JavaScript (search.js)

javascript

```
/**
 * Initialise la recherche en direct sur la page boutique
 */
function initLiveSearch() {
    const searchInput = document.getElementById('search-sidebar');
    const searchResults = document.getElementById('search-results');

    if (!searchInput || !searchResults) return;

    // Ajouter un délai avant de déclencher la recherche
    let searchTimeout;

    searchInput.addEventListener('input', function() {
        const query = this.value.trim();

        // Effacer le timeout précédent
        clearTimeout(searchTimeout);

        // Si la requête est vide, masquer les résultats
        if (query.length === 0) {
            searchResults.innerHTML = '';
            searchResults.classList.add('hidden');
            return;
        }

        // Attendre 300ms avant de déclencher la recherche
        searchTimeout = setTimeout(() => {
            // Si la requête est trop courte, ne pas effectuer de recherche
            if (query.length < 2) return;

            // Afficher un indicateur de chargement
            searchResults.innerHTML = '<div class="p-4 text-center">Recherche en cours...</div>
            searchResults.classList.remove('hidden');

            // Effectuer une requête AJAX
            fetch(`search_ajax.php?query=${encodeURIComponent(query)}`)
                .then(response => response.json())
                .then(data => {
                    if (data.products.length === 0) {
                        // Aucun résultat
                        searchResults.innerHTML = '<div class="p-4 text-center">Aucun produit t
                    } else {
                        // Afficher les résultats
                        displaySearchResults(data.products, searchResults);
                    }
                })
```

```
                .catch(error => {
                    console.error('Erreur lors de la recherche:', error);
                    searchResults.innerHTML = '<div class="p-4 text-center text-red-600">Une er
                });
        }, 300);
    });
}
```

Backend PHP (search_ajax.php)

php

```php
<?php
// Traitement AJAX pour la recherche de produits
require_once 'includes/db.php';
require_once 'includes/functions.php';

// Vérifier que la requête contient un terme de recherche
if (!isset($_GET['query']) || empty($_GET['query'])) {
    echo json_encode(['products' => []]);
    exit;
}

// Nettoyer la requête
$query = clean($_GET['query']);

// Effectuer la recherche dans la base de données
$products = searchProducts($query);

// Renvoyer les résultats au format JSON
echo json_encode(['products' => $products]);
```

2. Gestion du panier (AJAX)

Fonctionnalité

Permet aux utilisateurs d'ajouter des produits au panier sans recharger la page, avec mise à jour dynamique du compteur de produits.

Code JavaScript (main.js)

javascript

```javascript
/**
 * Fonction pour ajouter un produit au panier via AJAX
 *
 * @param {number} productId ID du produit
 * @param {number} quantity Quantité à ajouter
 */
function addToCartAjax(productId, quantity = 1) {
    // Vérifier que la quantité est valide
    if (quantity <= 0) {
        quantity = 1;
    }

    // Créer les données à envoyer
    const formData = new FormData();
    formData.append('product_id', productId);
    formData.append('quantity', quantity);
    formData.append('add_to_cart_ajax', true);

    // Effectuer la requête AJAX
    fetch('add_to_cart_ajax.php', {
        method: 'POST',
        body: formData
    })
        .then(response => response.json())
        .then(data => {
            if (data.success) {
                // Mettre à jour l'affichage du panier
                updateCartCounter(data.cart_count);

                // Afficher un message de succès
                showNotification(data.message, 'success');
            } else {
                // Afficher un message d'erreur
                showNotification(data.message, 'error');
            }
        })
        .catch(error => {
            console.error('Erreur lors de l\'ajout au panier:', error);
            showNotification('Une erreur est survenue. Veuillez réessayer.', 'error');
        });
}

/**
 * Mettre à jour le compteur d'articles du panier
 *
 * @param {number} count Nombre d'articles dans le panier
```

```
 */
function updateCartCounter(count) {
    const counter = document.querySelector('.cart-counter');

    if (!counter) return;

    if (count > 0) {
        counter.textContent = count;
        counter.classList.remove('hidden');
    } else {
        counter.textContent = '0';
        counter.classList.add('hidden');
    }
}

/**
 * Afficher une notification à l'utilisateur
 *
 * @param {string} message Message à afficher
 * @param {string} type Type de notification (success, error, info)
 */
function showNotification(message, type = 'info') {
    // Créer l'élément de notification
    const notification = document.createElement('div');
    let bgColor = 'bg-blue-100 text-blue-700';

    if (type === 'success') {
        bgColor = 'bg-green-100 text-green-700';
    } else if (type === 'error') {
        bgColor = 'bg-red-100 text-red-700';
    }

    notification.className = `fixed top-4 right-4 ${bgColor} p-4 rounded shadow-md z-50 notific
    notification.innerHTML = message;

    // Ajouter la notification au document
    document.body.appendChild(notification);

    // Supprimer la notification après 3 secondes
    setTimeout(() => {
        notification.classList.add('fade-out');

        setTimeout(() => {
            document.body.removeChild(notification);
        }, 500);
```

```
    }, 3000);
}
```

Utilisation dans le HTML

```html
<button type="button" onclick="addToCartAjax(<?php echo $product['id']; ?>, document.getElement
    <i class="fas fa-shopping-cart mr-2"></i> Ajouter au panier
</button>
```

Backend PHP (add_to_cart_ajax.php)

php

```php
<?php
// Traitement AJAX pour l'ajout au panier
session_start();
require_once 'includes/db.php';
require_once 'includes/functions.php';

// Initialiser le panier dans la session s'il n'existe pas
if (!isset($_SESSION['cart'])) {
    $_SESSION['cart'] = [];
}

// Vérifier que la requête est de type POST
if ($_SERVER['REQUEST_METHOD'] !== 'POST' || !isset($_POST['add_to_cart_ajax'])) {
    echo json_encode([
        'success' => false,
        'message' => 'Requête invalide.'
    ]);
    exit;
}

// Récupérer les données du formulaire
$productId = isset($_POST['product_id']) ? (int)$_POST['product_id'] : 0;
$quantity = isset($_POST['quantity']) ? (int)$_POST['quantity'] : 1;

// Vérifier que la quantité est valide
if ($quantity <= 0) {
    $quantity = 1;
}

// Récupérer les détails du produit
$product = getProductById($productId);

// Vérifier si le produit existe
if (!$product) {
    echo json_encode([
        'success' => false,
        'message' => 'Le produit demandé n\'existe pas.'
    ]);
    exit;
}

// Ajouter le produit au panier...
// (suite du code pour ajouter au panier)

// Calculer le nombre total d'articles dans le panier
$cartItemCount = 0;
```

```php
foreach ($_SESSION['cart'] as $item) {
    $cartItemCount += $item['quantite'];
}

// Renvoyer une réponse de succès
echo json_encode([
    'success' => true,
    'message' => 'Le produit a été ajouté au panier.',
    'cart_count' => $cartItemCount
]);
```

3. Validation de formulaire côté client

Fonctionnalité

Vérifie les données saisies par l'utilisateur avant l'envoi du formulaire, pour améliorer l'expérience utilisateur et réduire les requêtes inutiles au serveur.

Code JavaScript (dans register.php)

javascript

```
document.addEventListener('DOMContentLoaded', function() {
    const form = document.getElementById('registerForm');

    form.addEventListener('submit', function(e) {
        let hasErrors = false;

        // Masquer tous les messages d'erreur
        document.querySelectorAll('.error-message').forEach(el => {
            el.classList.add('hidden');
            el.textContent = '';
        });

        // Validation du nom
        const nom = document.getElementById('nom');
        if (nom.value.trim() === '') {
            showError('nom', 'Le nom est requis.');
            hasErrors = true;
        }

        // Validation de l'email
        const email = document.getElementById('email');
        if (email.value.trim() === '') {
            showError('email', 'L\'email est requis.');
            hasErrors = true;
        } else if (!isValidEmail(email.value)) {
            showError('email', 'Format d\'email invalide.');
            hasErrors = true;
        }

        // Validation du mot de passe
        const password = document.getElementById('password');
        if (password.value === '') {
            showError('password', 'Le mot de passe est requis.');
            hasErrors = true;
        } else if (password.value.length < 6) {
            showError('password', 'Le mot de passe doit contenir au moins 6 caractères.');
            hasErrors = true;
        }

        // Validation de la confirmation du mot de passe
        const confirmPassword = document.getElementById('confirm_password');
        if (confirmPassword.value === '') {
            showError('confirm_password', 'La confirmation du mot de passe est requise.');
            hasErrors = true;
        } else if (confirmPassword.value !== password.value) {
            showError('confirm_password', 'Les mots de passe ne correspondent pas.');
```

```
            hasErrors = true;
        }

        // Empêcher la soumission du formulaire s'il y a des erreurs
        if (hasErrors) {
            e.preventDefault();
        }
    });

    // Fonction pour afficher un message d'erreur
    function showError(fieldId, message) {
        const errorEl = document.getElementById(`${fieldId}-error`);
        errorEl.textContent = message;
        errorEl.classList.remove('hidden');
    }

    // Fonction pour valider le format de l'email
    function isValidEmail(email) {
        const re = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
        return re.test(email);
    }
});
```

4. Gestion des quantités dans le panier

Fonctionnalité

Permet d'augmenter ou diminuer la quantité d'un produit dans le panier, avec mise à jour dynamique du total.

Code JavaScript (dans cart.php)

javascript

```
function decrementQuantity(productId) {
    const input = document.getElementById(`quantity-${productId}`);
    const value = parseInt(input.value, 10);
    if (value > 0) {
        input.value = value - 1;
        updateItemTotal(productId);
    }
}

function incrementQuantity(productId, max = 100) {
    const input = document.getElementById(`quantity-${productId}`);
    const value = parseInt(input.value, 10);
    if (value < max) {
        input.value = value + 1;
        updateItemTotal(productId);
    }
}

function updateItemTotal(productId) {
    const quantityInput = document.getElementById(`quantity-${productId}`);
    const priceEl = document.getElementById(`price-${productId}`);
    const totalEl = document.getElementById(`total-${productId}`);

    if (!quantityInput || !priceEl || !totalEl) return;

    const quantity = parseInt(quantityInput.value, 10);
    const price = parseFloat(priceEl.dataset.price);
    const total = quantity * price;

    totalEl.textContent = formatPrice(total);

    // Mettre à jour le total du panier
    updateCartTotal();
}

function updateCartTotal() {
    const subTotalEl = document.getElementById('subtotal');
    const totalEl = document.getElementById('total');
    let total = 0;

    // Calculer le total à partir de tous les sous-totaux
    document.querySelectorAll('[id^="total-"]').forEach(el => {
        const productId = el.id.replace('total-', '');
        const quantityInput = document.getElementById(`quantity-${productId}`);
        const priceEl = document.getElementById(`price-${productId}`);
```

```
        if (quantityInput && priceEl) {
            const quantity = parseInt(quantityInput.value, 10);
            const price = parseFloat(priceEl.dataset.price);
            total += quantity * price;
        }
    });

    // Mettre à jour les éléments d'affichage
    if (subTotalEl) subTotalEl.textContent = formatPrice(total);
    if (totalEl) totalEl.textContent = formatPrice(total);
}

function formatPrice(price) {
    return new Intl.NumberFormat('fr-FR', {
        style: 'currency',
        currency: 'EUR'
    }).format(price);
}
```

5. Filtrage des produits par catégorie

Fonctionnalité

Permet de filtrer les produits par catégorie sur la page boutique, sans recharger la page.

Code JavaScript (main.js)

javascript

```javascript
/**
 * Initialise les filtres de produits sur la page boutique
 */
function initProductFilters() {
    const categoryLinks = document.querySelectorAll('.category-filter');

    if (categoryLinks.length === 0) return;

    categoryLinks.forEach(link => {
        link.addEventListener('click', function(e) {
            e.preventDefault();

            const categoryId = this.dataset.category;
            const productGrid = document.querySelector('.product-grid');

            // Ajouter la classe active au lien cliqué
            document.querySelectorAll('.category-filter').forEach(l => {
                l.classList.remove('active');
            });
            this.classList.add('active');

            // Si on clique sur "Toutes les catégories", afficher tous les produits
            if (categoryId === 'all') {
                document.querySelectorAll('.product-card').forEach(card => {
                    card.style.display = 'block';
                });
                return;
            }

            // Sinon, filtrer les produits par catégorie
            document.querySelectorAll('.product-card').forEach(card => {
                if (card.dataset.category === categoryId) {
                    card.style.display = 'block';
                } else {
                    card.style.display = 'none';
                }
            });
        });
    });
}
```

Utilisation dans le HTML

```html
<div class="categories-filter mb-6">
    <a href="#" class="category-filter active" data-category="all">Toutes les catégories</a>
    <?php foreach ($categories as $category): ?>
        <a href="#" class="category-filter" data-category="<?php echo $category['id']; ?>">
            <?php echo $category['nom']; ?>
        </a>
    <?php endforeach; ?>
</div>

<div class="product-grid grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-3 gap-6">
    <?php foreach ($products as $product): ?>
        <div class="product-card bg-white rounded-lg shadow-md overflow-hidden" data-category="
            <!-- Contenu de la carte produit -->
        </div>
    <?php endforeach; ?>
</div>
```

Ces fonctionnalités JavaScript améliorent considérablement l'expérience utilisateur en rendant le site plus interactif et réactif, sans nécessiter de rechargement de page pour de nombreuses actions courantes.