// Hero Section

const heroImages = [

{

url: "/images/Haleakala\_Maui.jpg", // Update with your image path

alt: "View of the Milky Way and observatory from top of Haleakala",

location: "Maui, Hawaii",

},

{

url: "/images/Holland\_MA.jpg", // Update with your image path

alt: "Northern lights above the lake, Holland, MA",

location: "Holland, Massachusetts",

},

{

url: "/images/Fish\_River\_Canyon\_Namibia.jpg", // Update with your image path

alt: "Sunset over Fish River Canyon, Namibia",

location: "Fish River Canyon, Namibia",

},

];

// Gallery Items

const galleryItems = [

{

id: 1,

title: "Santorini Sunset",

location: "Santorini, Greece",

category: "islands",

region: "Europe",

imageUrl: "/images/santorini.jpg", // Update with your image path

description: "Breathtaking views of the Aegean Sea from the white-washed buildings of Santorini.",

likes: 1243,

isFeatured: true,

},

{

id: 2,

title: "Bali Beach Paradise",

location: "Bali, Indonesia",

category: "beaches",

region: "Asia",

imageUrl: "/images/bali.jpg", // Update with your image path

description: "Crystal clear waters and golden sands of Bali's most beautiful beaches.",

likes: 982,

isFeatured: true,

},

{

id: 3,

title: "Swiss Alps",

location: "Zermatt, Switzerland",

category: "mountains",

region: "Europe",

imageUrl: "/placeholder.jpg", // Update with your image path

description: "Majestic mountain peaks and scenic valleys in the heart of the Swiss Alps.",

likes: 756,

isFeatured: false,

},

{

id: 4,

title: "Tokyo Skyline",

location: "Tokyo, Japan",

category: "cities",

region: "Asia",

imageUrl: "/placeholder.jpg", // Update with your image path

description: "The vibrant and futuristic skyline of Tokyo at night.",

likes: 1102,

isFeatured: false,

},

{

id: 5,

title: "Tuscan Countryside",

location: "Tuscany, Italy",

category: "countryside",

region: "Europe",

imageUrl: "/placeholder.jpg", // Update with your image path

description: "Rolling hills and cypress trees in the picturesque Tuscan countryside.",

likes: 845,

isFeatured: false,

},

{

id: 6,

title: "Maldives Overwater Bungalows",

location: "Maldives",

category: "islands",

region: "Asia",

imageUrl: "/placeholder.jpg", // Update with your image path

description: "Luxury overwater bungalows in the crystal clear waters of the Maldives.",

likes: 1567,

isFeatured: true,

},

{

id: 7,

title: "Grand Canyon",

location: "Arizona, USA",

category: "mountains",

region: "North America",

imageUrl: "/placeholder.jpg", // Update with your image path

description: "The vast and colorful landscape of the Grand Canyon.",

likes: 932,

isFeatured: false,

},

{

id: 8,

title: "Paris Streets",

location: "Paris, France",

category: "cities",

region: "Europe",

imageUrl: "/placeholder.jpg", // Update with your image path

description: "Charming streets and cafes in the romantic city of Paris.",

likes: 1089,

isFeatured: false,

},

{

id: 9,

title: "Amalfi Coast",

location: "Amalfi, Italy",

category: "beaches",

region: "Europe",

imageUrl: "/placeholder.jpg", // Update with your image path

description: "Colorful cliffside villages along the stunning Amalfi Coast.",

likes: 1345,

isFeatured: true,

},

{

id: 10,

title: "New Zealand Landscapes",

location: "South Island, New Zealand",

category: "countryside",

region: "Oceania",

imageUrl: "/placeholder.jpg", // Update with your image path

description: "Dramatic landscapes and untouched nature in New Zealand's South Island.",

likes: 876,

isFeatured: false,

},

{

id: 11,

title: "Kyoto Gardens",

location: "Kyoto, Japan",

category: "countryside",

region: "Asia",

imageUrl: "/placeholder.jpg", // Update with your image path

description: "Serene and meticulously maintained traditional Japanese gardens in Kyoto.",

likes: 765,

isFeatured: false,

},

{

id: 12,

title: "Machu Picchu",

location: "Cusco Region, Peru",

category: "mountains",

region: "South America",

imageUrl: "/placeholder.jpg", // Update with your image path

description: "The ancient Incan citadel set against a backdrop of stunning mountain scenery.",

likes: 1432,

isFeatured: true,

},

];

// Initialize the page

document.addEventListener('DOMContentLoaded', function() {

// Set current year in footer

document.getElementById('current-year').textContent = new Date().getFullYear();

// Initialize hero section

initHero();

// Initialize gallery

initGallery();

// Initialize filters

initFilters();

});

// Hero Section Functions

function initHero() {

const heroSlidesContainer = document.querySelector('.hero-slides');

const heroIndicatorsContainer = document.querySelector('.hero-indicators');

const heroLocationContainer = document.querySelector('.hero-location');

const heroText = document.querySelector('.hero-text');

// Create hero slides

heroImages.forEach((image, index) => {

const slide = document.createElement('div');

slide.className = `hero-slide ${index === 0 ? 'active' : ''}`;

slide.style.backgroundImage = `url(${image.url})`;

heroSlidesContainer.appendChild(slide);

// Create indicators

const indicator = document.createElement('button');

indicator.className = `hero-indicator ${index === 0 ? 'active' : ''}`;

indicator.setAttribute('aria-label', `View slide ${index + 1}`);

indicator.addEventListener('click', () => setActiveSlide(index));

heroIndicatorsContainer.appendChild(indicator);

});

// Set initial location

heroLocationContainer.innerHTML = `

<span>${heroImages[0].location}</span>

`;

// Add loaded class to hero text

setTimeout(() => {

heroText.classList.add('loaded');

}, 100);

// Auto rotate slides

let currentSlide = 0;

const slideInterval = setInterval(() => {

currentSlide = (currentSlide + 1) % heroImages.length;

setActiveSlide(currentSlide);

}, 5000);

function setActiveSlide(index) {

// Update slides

const slides = document.querySelectorAll('.hero-slide');

slides.forEach((slide, i) => {

slide.classList.toggle('active', i === index);

});

// Update indicators

const indicators = document.querySelectorAll('.hero-indicator');

indicators.forEach((indicator, i) => {

indicator.classList.toggle('active', i === index);

});

// Update location

heroLocationContainer.innerHTML = `

<span>${heroImages[index].location}</span>

`;

// Update current slide index

currentSlide = index;

}

}

// Gallery Functions

function initGallery() {

const galleryGrid = document.getElementById('gallery-grid');

const modal = document.getElementById('image-modal');

const modalClose = document.getElementById('modal-close');

const modalImage = document.getElementById('modal-image');

const modalTitle = document.getElementById('modal-title');

const modalLocation = document.getElementById('modal-location');

const modalDescription = document.getElementById('modal-description');

const modalCategory = document.getElementById('modal-category');

const modalRegion = document.getElementById('modal-region');

const modalLikes = document.getElementById('modal-likes');

const modalLikeBtn = document.getElementById('modal-like-btn');

const modalSaveBtn = document.getElementById('modal-save-btn');

let likedItems = [];

let savedItems = [];

// Render gallery items

renderGalleryItems();

// Close modal when clicking the close button

modalClose.addEventListener('click', () => {

modal.classList.remove('open');

});

// Close modal when clicking outside the content

modal.addEventListener('click', (e) => {

if (e.target === modal) {

modal.classList.remove('open');

}

});

// Like button in modal

modalLikeBtn.addEventListener('click', () => {

const itemId = parseInt(modalLikeBtn.dataset.itemId);

toggleLike(itemId);

updateModalLikeButton(itemId);

});

// Save button in modal

modalSaveBtn.addEventListener('click', () => {

const itemId = parseInt(modalSaveBtn.dataset.itemId);

toggleSave(itemId);

updateModalSaveButton(itemId);

});

function renderGalleryItems(filteredItems = galleryItems) {

galleryGrid.innerHTML = '';

filteredItems.forEach(item => {

const galleryItem = document.createElement('div');

galleryItem.className = 'gallery-item';

galleryItem.innerHTML = `

<div class="gallery-image-container">

<img src="${item.imageUrl}" alt="${item.title}" class="gallery-image">

<div class="gallery-overlay"></div>

<div class="gallery-gradient"></div>

${item.isFeatured ? '<div class="gallery-featured">Featured</div>' : ''}

<div class="gallery-actions">

<button class="gallery-action-btn ${likedItems.includes(item.id) ? 'liked' : ''}" data-action="like" data-id="${item.id}">

<i class="${likedItems.includes(item.id) ? 'fas' : 'far'} fa-heart"></i>

<span class="sr-only">Like</span>

</button>

<button class="gallery-action-btn ${savedItems.includes(item.id) ? 'active' : ''}" data-action="save" data-id="${item.id}">

<i class="${savedItems.includes(item.id) ? 'fas' : 'far'} fa-bookmark"></i>

<span class="sr-only">Save</span>

</button>

</div>

</div>

<div class="gallery-content">

<h3 class="gallery-title">${item.title}</h3>

<div class="gallery-location">

<i class="fas fa-map-marker-alt"></i>

${item.location}

</div>

<p class="gallery-description">${item.description}</p>

<div class="gallery-likes">

<i class="fas fa-heart"></i>

${item.likes + (likedItems.includes(item.id) ? 1 : 0)} likes

</div>

</div>

`;

// Open modal when clicking

galleryItem.addEventListener('click', () => {

openModal(item);

});

// Handle like and save button clicks

const actionButtons = galleryItem.querySelectorAll('.gallery-action-btn');

actionButtons.forEach(button => {

button.addEventListener('click', (e) => {

e.stopPropagation(); // Prevent modal from opening

const action = button.dataset.action;

const itemId = parseInt(button.dataset.id);

if (action === 'like') {

toggleLike(itemId);

renderGalleryItems(getFilteredItems());

} else if (action === 'save') {

toggleSave(itemId);

renderGalleryItems(getFilteredItems());

}

});

});

galleryGrid.appendChild(galleryItem);

});

// Show message if no items

if (filteredItems.length === 0) {

const noResults = document.createElement('div');

noResults.className = 'no-results';

noResults.innerHTML = '<p>No destinations found matching your criteria. Try adjusting your filters.</p>';

galleryGrid.appendChild(noResults);

}

}

function openModal(item) {

modalImage.src = item.imageUrl;

modalImage.alt = item.title;

modalTitle.textContent = item.title;

modalLocation.textContent = item.location;

modalDescription.textContent = item.description;

modalCategory.textContent = item.category;

modalRegion.textContent = item.region;

modalLikes.textContent = item.likes + (likedItems.includes(item.id) ? 1 : 0);

// Set data attributes for like and save buttons

modalLikeBtn.dataset.itemId = item.id;

modalSaveBtn.dataset.itemId = item.id;

// Update button states

updateModalLikeButton(item.id);

updateModalSaveButton(item.id);

// Open modal

modal.classList.add('open');

}

function updateModalLikeButton(itemId) {

const isLiked = likedItems.includes(itemId);

const likeIcon = modalLikeBtn.querySelector('i');

if (isLiked) {

likeIcon.className = 'fas fa-heart';

modalLikeBtn.classList.add('liked');

} else {

likeIcon.className = 'far fa-heart';

modalLikeBtn.classList.remove('liked');

}

// Update likes count

const item = galleryItems.find(item => item.id === itemId);

if (item) {

modalLikes.textContent = item.likes + (isLiked ? 1 : 0);

}

}

function updateModalSaveButton(itemId) {

const isSaved = savedItems.includes(itemId);

const saveIcon = modalSaveBtn.querySelector('i');

if (isSaved) {

saveIcon.className = 'fas fa-bookmark';

modalSaveBtn.classList.add('active');

modalSaveBtn.querySelector('span').textContent = 'Saved';

} else {

saveIcon.className = 'far fa-bookmark';

modalSaveBtn.classList.remove('active');

modalSaveBtn.querySelector('span').textContent = 'Save';

}

}

function toggleLike(itemId) {

const index = likedItems.indexOf(itemId);

if (index === -1) {

likedItems.push(itemId);

} else {

likedItems.splice(index, 1);

}

}

function toggleSave(itemId) {

const index = savedItems.indexOf(itemId);

if (index === -1) {

savedItems.push(itemId);

} else {

savedItems.splice(index, 1);

}

}

}

// Filter Functions

function initFilters() {

const searchInput = document.getElementById('search-input');

const clearSearchBtn = document.getElementById('clear-search');

const categoryButtons = document.querySelectorAll('.category-btn');

const dropdownToggle = document.querySelector('.dropdown-toggle');

const dropdownItems = document.querySelectorAll('.dropdown-item');

const activeFiltersContainer = document.getElementById('active-filters');

let activeCategory = 'all';

let searchQuery = '';

let activeFilters = [];

// Search input event

searchInput.addEventListener('input', (e) => {

searchQuery = e.target.value.trim().toLowerCase();

clearSearchBtn.classList.toggle('hidden', !searchQuery);

renderGalleryItems(getFilteredItems());

updateActiveFilters();

});

// Clear search button

clearSearchBtn.addEventListener('click', () => {

searchInput.value = '';

searchQuery = '';

clearSearchBtn.classList.add('hidden');

renderGalleryItems(getFilteredItems());

updateActiveFilters();

});

// Category buttons

categoryButtons.forEach(button => {

button.addEventListener('click', () => {

categoryButtons.forEach(btn => btn.classList.remove('active'));

button.classList.add('active');

activeCategory = button.dataset.category;

renderGalleryItems(getFilteredItems());

updateActiveFilters();

});

});

// Dropdown toggle

dropdownToggle.addEventListener('click', () => {

dropdownToggle.parentElement.classList.toggle('open');

});

// Close dropdown when clicking outside

document.addEventListener('click', (e) => {

if (!e.target.closest('.dropdown')) {

document.querySelector('.dropdown').classList.remove('open');

}

});

// Dropdown items (regions)

dropdownItems.forEach(item => {

item.addEventListener('click', (e) => {

e.preventDefault();

const region = item.dataset.region;

addFilter(region);

document.querySelector('.dropdown').classList.remove('open');

renderGalleryItems(getFilteredItems());

updateActiveFilters();

});

});

function addFilter(filter) {

if (!activeFilters.includes(filter)) {

activeFilters.push(filter);

}

}

function removeFilter(filter) {

activeFilters = activeFilters.filter(f => f !== filter);

renderGalleryItems(getFilteredItems());

updateActiveFilters();

}

function updateActiveFilters() {

activeFiltersContainer.innerHTML = '';

if (activeFilters.length === 0) {

return;

}

const filtersLabel = document.createElement('span');

filtersLabel.className = 'active-filters-label';

filtersLabel.textContent = 'Active filters:';

activeFiltersContainer.appendChild(filtersLabel);

activeFilters.forEach(filter => {

const filterTag = document.createElement('div');

filterTag.className = 'filter-tag';

filterTag.innerHTML = `

${filter}

<button>

<i class="fas fa-times"></i>

<span class="sr-only">Remove ${filter} filter</span>

</button>

`;

filterTag.querySelector('button').addEventListener('click', () => {

removeFilter(filter);

});

activeFiltersContainer.appendChild(filterTag);

});

if (activeFilters.length > 1) {

const clearAllBtn = document.createElement('button');

clearAllBtn.className = 'clear-filters';

clearAllBtn.textContent = 'Clear all';

clearAllBtn.addEventListener('click', () => {

activeFilters = [];

renderGalleryItems(getFilteredItems());

updateActiveFilters();

});

activeFiltersContainer.appendChild(clearAllBtn);

}

}

// Make getFilteredItems available globally

window.getFilteredItems = getFilteredItems;

}

// Get filtered gallery items based on current filters

function getFilteredItems() {

const searchInput = document.getElementById('search-input');

const activeCategory = document.querySelector('.category-btn.active').dataset.category;

const searchQuery = searchInput.value.trim().toLowerCase();

const activeFilters = Array.from(document.querySelectorAll('.filter-tag')).map(tag =>

tag.textContent.trim().replace('Remove filter', '').trim()

);

return galleryItems.filter(item => {

// Filter by category

if (activeCategory !== 'all' && item.category !== activeCategory) {

return false;

}

// Filter by search query

if (searchQuery && !item.title.toLowerCase().includes(searchQuery) &&

!item.location.toLowerCase().includes(searchQuery) &&

!item.description.toLowerCase().includes(searchQuery)) {

return false;

}

// Filter by region

if (activeFilters.length > 0 && !activeFilters.includes(item.region)) {

return false;

}

return true;

});

}

// Helper function to render gallery items

function renderGalleryItems(filteredItems = galleryItems) {

const galleryGrid = document.getElementById('gallery-grid');

// This function is implemented in the initGallery function

// We're just calling it here to make it accessible globally

if (typeof window.renderGalleryItems === 'function') {

window.renderGalleryItems(filteredItems);

}

}