// Mobile Navigation Toggle

Const navToggle = document.getElementById(‘nav-toggle’);

Const navMenu = document.getElementById(‘nav-menu’);// Mobile Navigation Toggle

Const navToggle = document.getElementById(‘nav-toggle’);

Const navMenu = document.getElementById(‘nav-menu’);

Const navLinks = document.querySelectorAll(‘.nav-link’);

// Toggle mobile menu

navToggle.addEventListener(‘click’, () => {

navMenu.classList.toggle(‘active’);

// Animate hamburger bars

Const bars = navToggle.querySelectorAll(‘.bar’);

Bars[0].style.transform = navMenu.classList.contains(‘active’)

? ‘rotate(-45deg) translate(-5px, 6px)’

: ‘rotate(0) translate(0, 0)’;

Bars[1].style.opacity = navMenu.classList.contains(‘active’) ? ‘0’ : ‘1’;

Bars[2].style.transform = navMenu.classList.contains(‘active’)

? ‘rotate(45deg) translate(-5px, -6px)’

: ‘rotate(0) translate(0, 0)’;

});

// Close mobile menu when clicking on nav links

navLinks.forEach(link => {

link.addEventListener(‘click’, () => {

navMenu.classList.remove(‘active’);

// Reset hamburger bars

Const bars = navToggle.querySelectorAll(‘.bar’);

Bars[0].style.transform = ‘rotate(0) translate(0, 0)’;

Bars[1].style.opacity = ‘1’;

Bars[2].style.transform = ‘rotate(0) translate(0, 0)’;

});

});

// Smooth Scrolling for Navigation Links

Document.querySelectorAll(‘a[href^=”#”]’).forEach(anchor => {

Anchor.addEventListener(‘click’, function € {

e.preventDefault();

const target = document.querySelector(this.getAttribute(‘href’));

if (target) {

const headerOffset = 70;

const elementPosition = target.getBoundingClientRect().top;

const offsetPosition = elementPosition + window.pageYOffset – headerOffset;

window.scrollTo({

top: offsetPosition,

behavior: ‘smooth’

});

}

});

});

// Active Navigation Link on Scroll

Window.addEventListener(‘scroll’, () => {

Const sections = document.querySelectorAll(‘section[id]’);

Const scrollPos = window.pageYOffset + 100;

Sections.forEach(section => {

Const sectionTop = section.offsetTop;

Const sectionHeight = section.offsetHeight;

Const sectionId = section.getAttribute(‘id’);

If (scrollPos >= sectionTop && scrollPos < sectionTop + sectionHeight) {

// Remove active class from all nav links

navLinks.forEach(link => {

link.classList.remove(‘active’);

});

// Add active class to current section’s nav link

Const activeLink = document.querySelector(`.nav-link[href=”#${sectionId}”]`);

If (activeLink) {

activeLink.classList.add(‘active’);

}

}

});

});

// Navbar background change on scroll

Window.addEventListener(‘scroll’, () => {

Const navbar = document.querySelector(‘.navbar’);

If (window.scrollY > 50) {

Navbar.style.background = ‘rgba(243, 233, 220, 0.98)’;

Navbar.style.boxShadow = ‘0 2px 20px rgba(0, 0, 0, 0.1)’;

} else {

Navbar.style.background = ‘rgba(243, 233, 220, 0.95)’;

Navbar.style.boxShadow = ‘none’;

}

});

// Contact Form Handling

Const contactForm = document.getElementById(‘contact-form’);

contactForm.addEventListener(‘submit’, function€ {

e.preventDefault();

// Get form data

Const formData = new FormData(this);

Const name = formData.get(‘name’);

Const email = formData.get(‘email’);

Const subject = formData.get(‘subject’);

Const message = formData.get(‘message’);

// Simple validation

If (!name || !email || !subject || !message) {

showNotification(‘Please fill in all fields’, ‘error’);

return;

}

// Email validation

Const emailRegex = /^[^s@][+@[^s@]+.[^s@]+$/](mailto:+@[%5es@]+.[%5es@]+$/);

If (!emailRegex.test(email)) {

showNotification(‘Please enter a valid email address’, ‘error’);

return;

}

// Simulate form submission

Const submitBtn = this.querySelector(‘.btn-primary’);

Const originalText = submitBtn.innerHTML;

submitBtn.innerHTML = ‘<I class=”fas fa-spinner fa-spin”></i> Sending…’;

submitBtn.disabled = true;

// Simulate API call delay

setTimeout(() => {

showNotification(‘Thank you! Your message has been sent successfully. I’ll get back to you soon!’, ‘success’);

contactForm.reset();

submitBtn.innerHTML = originalText;

submitBtn.disabled = false;

}, 2000);

});

// Notification System

Function showNotification(message, type = ‘info’) {

// Remove existing notification

Const existingNotification = document.querySelector(‘.notification’);

If (existingNotification) {

existingNotification.remove();

}

// Create notification element

Const notification = document.createElement(‘div’);

Notification.className = `notification notification-${type}`;

Notification.innerHTML = `

<div class=”notification-content”>

<I class=”fas ${type === ‘success’ ? ‘fa-check-circle’ : type === ‘error’ ? ‘fa-exclamation-circle’ : ‘fa-info-circle’}”></i>

<span>${message}</span>

<button class=”notification-close” onclick=”this.parentElement.parentElement.remove()”>

<I class=”fas fa-times”></i>

</button>

</div>

`;

// Add styles

Notification.style.cssText = `

Position: fixed;

Top: 90px;

Right: 20px;

Background: ${type === ‘success’ ? ‘#10b981’ : type === ‘error’ ? ‘#ef4444’ : ‘#3b82f6’};

Color: white;

Padding: 15px 20px;

Border-radius: 10px;

Box-shadow: 0 10px 25px rgba(0, 0, 0, 0.2);

z-index: 1001;

max-width: 400px;

transform: translateX(100%);

transition: transform 0.3s ease;

`;

Notification.querySelector(‘.notification-content’).style.cssText = `

Display: flex;

Align-items: center;

Gap: 10px;

`;

Notification.querySelector(‘.notification-close’).style.cssText = `

Background: none;

Border: none;

Color: white;

Cursor: pointer;

Font-size: 16px;

Opacity: 0.8;

Transition: opacity 0.2s ease;

Margin-left: auto;

`;

// Add to DOM

Document.body.appendChild(notification);

// Animate in

setTimeout(() => {

notification.style.transform = ‘translateX(0)’;

}, 100);

// Auto remove after 5 seconds

setTimeout(() => {

if (notification.parentElement) {

notification.style.transform = ‘translateX(100%)’;

setTimeout(() => {

if (notification.parentElement) {

notification.remove();

}

}, 300);

}

}, 5000);

}

// Scroll Animation for Elements

Const observerOptions = {

Threshold: 0.1,

rootMargin: ‘0px 0px -50px 0px’

};

Const observer = new IntersectionObserver((entries) => {

Entries.forEach(entry => {

If (entry.isIntersecting) {

Entry.target.style.animation = ‘fadeInUp 0.6s ease forwards’;

Observer.unobserve(entry.target);

}

});

}, observerOptions);

// Add CSS animation keyframes

Const style = document.createElement(‘style’);

Style.textContent = `

@keyframes fadeInUp {

From {

Opacity: 0;

Transform: translateY(30px);

}

To {

Opacity: 1;

Transform: translateY(0);

}

}

.animate-on-scroll {

Opacity: 0;

Transform: translateY(30px);

}

.nav-link.active {

Color: var(--accent-1);

}

.nav-link.active::after {

Width: 100%;

}

`;

Document.head.appendChild(style);

// Initialize animations when DOM is loaded

Document.addEventListener(‘DOMContentLoaded’, () => {

// Elements to animate on scroll

Const animatedElements = document.querySelectorAll(`

.skill-card,

.project-card,

.education-card,

.highlight-item,

.about-text,

.contact-info,

.contact-form

`);

animatedElements.forEach(el => {

el.classList.add(‘animate-on-scroll’);

observer.observe(el);

});

});

// Typing Effect for Hero Section

Function typeWriter(element, text, speed = 50) {

Let I = 0;

Element.innerHTML = ‘’;

Function type() {

If (I < text.length) {

Element.innerHTML += text.charAt(i);

I++;

setTimeout(type, speed);

}

}

Type();

}

// Initialize typing effect when page loads

Window.addEventListener(‘load’, () => {

Const heroTitle = document.querySelector(‘.hero-title’);

If (heroTitle) {

Const originalText = heroTitle.textContent;

typeWriter(heroTitle, originalText, 80);

}

});

// Smooth reveal animation for sections

Const revealSections = () => {

Const sections = document.querySelectorAll(‘section’);

Sections.forEach(section => {

Const sectionTop = section.getBoundingClientRect().top;

Const sectionVisible = 150;

If (sectionTop < window.innerHeight – sectionVisible) {

Section.classList.add(‘section-visible’);

}

});

};

// Add section reveal styles

Const sectionStyles = document.createElement(‘style’);

sectionStyles.textContent = `

section {

opacity: 0;

transform: translateY(20px);

transition: all 0.6s ease;

}

Section.section-visible {

Opacity: 1;

Transform: translateY(0);

}

#hero {

Opacity: 1;

Transform: translateY(0);

}

`;

Document.head.appendChild(sectionStyles);

// Run on scroll and page load

Window.addEventListener(‘scroll’, revealSections);

Window.addEventListener(‘load’, revealSections);

// Parallax effect for hero section

Window.addEventListener(‘scroll’, () => {

Const scrolled = window.pageYOffset;

Const hero = document.querySelector(‘.hero’);

Const heroScroll = document.querySelector(‘.hero-scroll’);

If (hero) {

Const rate = scrolled \* -0.5;

Hero.style.transform = `translateY(${rate}px)`;

}

// Hide scroll indicator when scrolling

If (heroScroll) {

If (scrolled > 100) {

heroScroll.style.opacity = ‘0’;

} else {

heroScroll.style.opacity = ‘1’;

}

}

});

// Skills card hover effect enhancement

Document.addEventListener(‘DOMContentLoaded’, () => {

Const skillCards = document.querySelectorAll(‘.skill-card’);

skillCards.forEach(card => {

card.addEventListener(‘mouseenter’, () => {

card.style.transform = ‘translateY(-15px) scale(1.02)’;

});

Card.addEventListener(‘mouseleave’, () => {

Card.style.transform = ‘translateY(0) scale(1)’;

});

});

});

// Project cards interaction

Document.addEventListener(‘DOMContentLoaded’, () => {

Const projectCards = document.querySelectorAll(‘.project-card’);

projectCards.forEach(card => {

card.addEventListener(‘mouseenter’, () => {

card.style.transform = ‘translateY(-15px)’;

card.style.boxShadow = ‘0 20px 50px rgba(217, 111, 50, 0.2)’;

});

Card.addEventListener(‘mouseleave’, () => {

Card.style.transform = ‘translateY(0)’;

Card.style.boxShadow = ‘0 5px 15px rgba(0, 0, 0, 0.1)’;

});

});

});

// Add loading animation

Window.addEventListener(‘load’, () => {

Const loader = document.getElementById(‘loader’);

If (loader) {

Loader.style.display = ‘none’;

}

// Add entrance animation to hero content

Const heroContent = document.querySelector(‘.hero-content’);

If (heroContent) {

heroContent.style.animation = ‘fadeInUp 1s ease’;

}

});

Console.log(‘Portfolio website loaded successfully! 🚀’);

Console.log(‘Developed for Jayesh Kakade – Data Science Student’);

Const navLinks = document.querySelectorAll(‘.nav-link’);

// Toggle mobile menu

navToggle.addEventListener(‘click’, () => {

navMenu.classList.toggle(‘active’);

// Animate hamburger bars

Const bars = navToggle.querySelectorAll(‘.bar’);

Bars[0].style.transform = navMenu.classList.contains(‘active’)

? ‘rotate(-45deg) translate(-5px, 6px)’

: ‘rotate(0) translate(0, 0)’;

Bars[1].style.opacity = navMenu.classList.contains(‘active’) ? ‘0’ : ‘1’;

Bars[2].style.transform = navMenu.classList.contains(‘active’)

? ‘rotate(45deg) translate(-5px, -6px)’

: ‘rotate(0) translate(0, 0)’;

});

// Close mobile menu when clicking on nav links

navLinks.forEach(link => {

link.addEventListener(‘click’, () => {

navMenu.classList.remove(‘active’);

// Reset hamburger bars

Const bars = navToggle.querySelectorAll(‘.bar’);

Bars[0].style.transform = ‘rotate(0) translate(0, 0)’;

Bars[1].style.opacity = ‘1’;

Bars[2].style.transform = ‘rotate(0) translate(0, 0)’;

});

});

// Smooth Scrolling for Navigation Links

Document.querySelectorAll(‘a[href^=”#”]’).forEach(anchor => {

Anchor.addEventListener(‘click’, function € {

e.preventDefault();

const target = document.querySelector(this.getAttribute(‘href’));

if (target) {

const headerOffset = 70;

const elementPosition = target.getBoundingClientRect().top;

const offsetPosition = elementPosition + window.pageYOffset – headerOffset;

window.scrollTo({

top: offsetPosition,

behavior: ‘smooth’

});

}

});

});

// Active Navigation Link on Scroll

Window.addEventListener(‘scroll’, () => {

Const sections = document.querySelectorAll(‘section[id]’);

Const scrollPos = window.pageYOffset + 100;

Sections.forEach(section => {

Const sectionTop = section.offsetTop;

Const sectionHeight = section.offsetHeight;

Const sectionId = section.getAttribute(‘id’);

If (scrollPos >= sectionTop && scrollPos < sectionTop + sectionHeight) {

// Remove active class from all nav links

navLinks.forEach(link => {

link.classList.remove(‘active’);

});

// Add active class to current section’s nav link

Const activeLink = document.querySelector(`.nav-link[href=”#${sectionId}”]`);

If (activeLink) {

activeLink.classList.add(‘active’);

}

}

});

});

// Navbar background change on scroll

Window.addEventListener(‘scroll’, () => {

Const navbar = document.querySelector(‘.navbar’);

If (window.scrollY > 50) {

Navbar.style.background = ‘rgba(243, 233, 220, 0.98)’;

Navbar.style.boxShadow = ‘0 2px 20px rgba(0, 0, 0, 0.1)’;

} else {

Navbar.style.background = ‘rgba(243, 233, 220, 0.95)’;

Navbar.style.boxShadow = ‘none’;

}

});

// Contact Form Handling

Const contactForm = document.getElementById(‘contact-form’);

contactForm.addEventListener(‘submit’, function€ {

e.preventDefault();

// Get form data

Const formData = new FormData(this);

Const name = formData.get(‘name’);

Const email = formData.get(‘email’);

Const subject = formData.get(‘subject’);

Const message = formData.get(‘message’);

// Simple validation

If (!name || !email || !subject || !message) {

showNotification(‘Please fill in all fields’, ‘error’);

return;

}

// Email validation

Const emailRegex = /^[^s@][+@[^s@]+.[^s@]+$/](mailto:+@[%5es@]+.[%5es@]+$/);

If (!emailRegex.test(email)) {

showNotification(‘Please enter a valid email address’, ‘error’);

return;

}

// Simulate form submission

Const submitBtn = this.querySelector(‘.btn-primary’);

Const originalText = submitBtn.innerHTML;

submitBtn.innerHTML = ‘<I class=”fas fa-spinner fa-spin”></i> Sending…’;

submitBtn.disabled = true;

// Simulate API call delay

setTimeout(() => {

showNotification(‘Thank you! Your message has been sent successfully. I’ll get back to you soon!’, ‘success’);

contactForm.reset();

submitBtn.innerHTML = originalText;

submitBtn.disabled = false;

}, 2000);

});

// Notification System

Function showNotification(message, type = ‘info’) {

// Remove existing notification

Const existingNotification = document.querySelector(‘.notification’);

If (existingNotification) {

existingNotification.remove();

}

// Create notification element

Const notification = document.createElement(‘div’);

Notification.className = `notification notification-${type}`;

Notification.innerHTML = `

<div class=”notification-content”>

<I class=”fas ${type === ‘success’ ? ‘fa-check-circle’ : type === ‘error’ ? ‘fa-exclamation-circle’ : ‘fa-info-circle’}”></i>

<span>${message}</span>

<button class=”notification-close” onclick=”this.parentElement.parentElement.remove()”>

<I class=”fas fa-times”></i>

</button>

</div>

`;

// Add styles

Notification.style.cssText = `

Position: fixed;

Top: 90px;

Right: 20px;

Background: ${type === ‘success’ ? ‘#10b981’ : type === ‘error’ ? ‘#ef4444’ : ‘#3b82f6’};

Color: white;

Padding: 15px 20px;

Border-radius: 10px;

Box-shadow: 0 10px 25px rgba(0, 0, 0, 0.2);

z-index: 1001;

max-width: 400px;

transform: translateX(100%);

transition: transform 0.3s ease;

`;

Notification.querySelector(‘.notification-content’).style.cssText = `

Display: flex;

Align-items: center;

Gap: 10px;

`;

Notification.querySelector(‘.notification-close’).style.cssText = `

Background: none;

Border: none;

Color: white;

Cursor: pointer;

Font-size: 16px;

Opacity: 0.8;

Transition: opacity 0.2s ease;

Margin-left: auto;

`;

// Add to DOM

Document.body.appendChild(notification);

// Animate in

setTimeout(() => {

notification.style.transform = ‘translateX(0)’;

}, 100);

// Auto remove after 5 seconds

setTimeout(() => {

if (notification.parentElement) {

notification.style.transform = ‘translateX(100%)’;

setTimeout(() => {

if (notification.parentElement) {

notification.remove();

}

}, 300);

}

}, 5000);

}

// Scroll Animation for Elements

Const observerOptions = {

Threshold: 0.1,

rootMargin: ‘0px 0px -50px 0px’

};

Const observer = new IntersectionObserver((entries) => {

Entries.forEach(entry => {

If (entry.isIntersecting) {

Entry.target.style.animation = ‘fadeInUp 0.6s ease forwards’;

Observer.unobserve(entry.target);

}

});

}, observerOptions);

// Add CSS animation keyframes

Const style = document.createElement(‘style’);

Style.textContent = `

@keyframes fadeInUp {

From {

Opacity: 0;

Transform: translateY(30px);

}

To {

Opacity: 1;

Transform: translateY(0);

}

}

.animate-on-scroll {

Opacity: 0;

Transform: translateY(30px);

}

.nav-link.active {

Color: var(--accent-1);

}

.nav-link.active::after {

Width: 100%;

}

`;

Document.head.appendChild(style);

// Initialize animations when DOM is loaded

Document.addEventListener(‘DOMContentLoaded’, () => {

// Elements to animate on scroll

Const animatedElements = document.querySelectorAll(`

.skill-card,

.project-card,

.education-card,

.highlight-item,

.about-text,

.contact-info,

.contact-form

`);

animatedElements.forEach(el => {

el.classList.add(‘animate-on-scroll’);

observer.observe(el);

});

});

// Typing Effect for Hero Section

Function typeWriter(element, text, speed = 50) {

Let I = 0;

Element.innerHTML = ‘’;

Function type() {

If (I < text.length) {

Element.innerHTML += text.charAt(i);

I++;

setTimeout(type, speed);

}

}

Type();

}

// Initialize typing effect when page loads

Window.addEventListener(‘load’, () => {

Const heroTitle = document.querySelector(‘.hero-title’);

If (heroTitle) {

Const originalText = heroTitle.textContent;

typeWriter(heroTitle, originalText, 80);

}

});

// Smooth reveal animation for sections

Const revealSections = () => {

Const sections = document.querySelectorAll(‘section’);

Sections.forEach(section => {

Const sectionTop = section.getBoundingClientRect().top;

Const sectionVisible = 150;

If (sectionTop < window.innerHeight – sectionVisible) {

Section.classList.add(‘section-visible’);

}

});

};

// Add section reveal styles

Const sectionStyles = document.createElement(‘style’);

sectionStyles.textContent = `

section {

opacity: 0;

transform: translateY(20px);

transition: all 0.6s ease;

}

Section.section-visible {

Opacity: 1;

Transform: translateY(0);

}

#hero {

Opacity: 1;

Transform: translateY(0);

}

`;

Document.head.appendChild(sectionStyles);

// Run on scroll and page load

Window.addEventListener(‘scroll’, revealSections);

Window.addEventListener(‘load’, revealSections);

// Parallax effect for hero section

Window.addEventListener(‘scroll’, () => {

Const scrolled = window.pageYOffset;

Const hero = document.querySelector(‘.hero’);

Const heroScroll = document.querySelector(‘.hero-scroll’);

If (hero) {

Const rate = scrolled \* -0.5;

Hero.style.transform = `translateY(${rate}px)`;

}

// Hide scroll indicator when scrolling

If (heroScroll) {

If (scrolled > 100) {

heroScroll.style.opacity = ‘0’;

} else {

heroScroll.style.opacity = ‘1’;

}

}

});

// Skills card hover effect enhancement

Document.addEventListener(‘DOMContentLoaded’, () => {

Const skillCards = document.querySelectorAll(‘.skill-card’);

skillCards.forEach(card => {

card.addEventListener(‘mouseenter’, () => {

card.style.transform = ‘translateY(-15px) scale(1.02)’;

});

Card.addEventListener(‘mouseleave’, () => {

Card.style.transform = ‘translateY(0) scale(1)’;

});

});

});

// Project cards interaction

Document.addEventListener(‘DOMContentLoaded’, () => {

Const projectCards = document.querySelectorAll(‘.project-card’);

projectCards.forEach(card => {

card.addEventListener(‘mouseenter’, () => {

card.style.transform = ‘translateY(-15px)’;

card.style.boxShadow = ‘0 20px 50px rgba(217, 111, 50, 0.2)’;

});

Card.addEventListener(‘mouseleave’, () => {

Card.style.transform = ‘translateY(0)’;

Card.style.boxShadow = ‘0 5px 15px rgba(0, 0, 0, 0.1)’;

});

});

});

// Add loading animation

Window.addEventListener(‘load’, () => {

Const loader = document.getElementById(‘loader’);

If (loader) {

Loader.style.display = ‘none’;

}

// Add entrance animation to hero content

Const heroContent = document.querySelector(‘.hero-content’);

If (heroContent) {

heroContent.style.animation = ‘fadeInUp 1s ease’;

}

});

Console.log(‘Portfolio website loaded successfully! 🚀’);

Console.log(‘Developed for Jayesh Kakade – Data Science Student’);