The Vibe Coder's Web Design Guide

Everything a beginner needs to know to prompt AI tools like Stitch, Framer, or Webflow AI

Includes: Design Aesthetics · Scroll & Animation · Layouts · Navigation · Typography & Color · UI Patterns

How to Use This Guide

This guide is your cheat sheet for web design — written for vibe coders who build websites by prompting AI tools. You don't need to write code by hand. What you DO need is the vocabulary to tell the AI exactly what you want.

Each entry in this guide follows the same format:

What it is	A plain-English explanation of the concept
How it works	The visual mechanism — what makes it look or behave that way
AI Prompt	A copy-paste prompt you can give to Stitch, Framer AI, or any AI builder
Example Sites	Real websites that use this style so you can see it in action

TIP When prompting an AI tool, always combine 3 things: Aesthetic + Layout + Animation. Example: 'Build a glassmorphism hero section (aesthetic) with a centered two-column layout (layout) where the cards fade in on scroll (animation).' That level of detail gets you great results.

1. Design Aesthetics

Design aesthetics are the visual language of a website — the overall 'vibe' and personality. When you look at a website and think 'that looks futuristic' or 'that feels premium', you're reacting to its aesthetic. Understanding these styles lets you describe exactly what you want to an AI.

1.1 Glassmorphism

What it is

Glassmorphism makes UI elements look like frosted glass — semi-transparent, blurred, with a subtle border. Imagine a glass panel sitting in front of a colorful, blurry background. The panel lets some of the background bleed through while remaining readable.

How it works

The magic happens with two CSS properties: backdrop-filter:blur() which blurs everything behind the element, and a background-color set to rgba() (red, green, blue, alpha) where alpha is a low value like 0.15 making it partially transparent. A thin white border at low opacity completes the glass look. You always need a colorful, vibrant background behind the glass panels — otherwise there's nothing to blur.

backdrop-filter	CSS property that blurs the content behind an element
rgba()	Color with transparency — the 4th value (alpha) controls how see-through it is
Semi-transparent	Partially see-through — like looking through tinted glass
Best background	Purple/blue/pink gradients, abstract blobs, or colorful photos

AI PROMPT TO USE

"Build a glassmorphism SaaS landing page. Use a vivid purple-to-blue gradient background (#667eea to #764ba2). Create 3 feature cards that look like frosted glass panels — each card should have background: rgba(255,255,255,0.15), a backdrop-filter blur of 12px, a white border at 30% opacity, and a subtle drop shadow. Text should be white. Add a glass-style navbar at the top."

EXAMPLE WEBSITES TO STUDY

→ **Halo Lab on Dribbble —** Classic glassmorphism UI kit with frosted cards and vivid gradients *(dribbble.com/tags/glassmorphism)*

→ **Apple iCloud.com —** Subtle glass panels in login UI — very refined glassmorphism *(icloud.com)*

→ **Glassmorphism.com —** Interactive CSS generator to understand the effect visually *(glassmorphism.com)*

1.2 Neumorphism (Soft UI)

What it is

Neumorphism (new + skeuomorphism) creates buttons and cards that look like they are physically pushed out of or pressed into the background surface. Everything is the SAME color — the background and the element share an identical color. What creates the 3D illusion is a pair of shadows: one dark shadow from the bottom-right and one light (near-white) shadow from the top-left.

How it works

The trick is all in box-shadow. You apply two shadows simultaneously. The dark shadow simulates where light would NOT hit a raised surface. The light shadow simulates where light WOULD hit. Because both the background and the element are the exact same color (like #e0e5ec), the element appears to grow organically from the surface rather than sit on top of it.

box-shadow	CSS property — you can apply TWO shadows to one element separated by a comma
Concave	Pressed inward — feels like a pressed button (inner shadow)
Convex	Pushed outward — feels like a raised element (outer shadow)
Color rule	Background color and element color MUST match exactly

TIP Neumorphism looks best at low contrast. It's ideal for dark mode music players, control panels, and dashboard widgets. It struggles with accessibility on bright screens because the contrast is very low — don't use it for critical text or important CTAs.

AI PROMPT TO USE

"Create a neumorphism music player dashboard on a #e0e5ec background. Build raised circular play/pause buttons using dual box-shadows: box-shadow: 6px 6px 12px #b8bec7, -6px -6px 12px #ffffff. Add pressed (active) state with inset shadows: inset 4px 4px 8px #b8bec7, inset -4px -4px 8px #ffffff. Include a volume slider and track progress bar in the same neumorphic style."

EXAMPLE WEBSITES TO STUDY

→ **Neumorphism.io —** CSS generator — drag the sliders and watch the effect build. Best way to learn it. *(neumorphism.io)*

→ **Dribbble — Neumorphism UI —** Hundreds of beautiful Neumorphism UI examples for reference *(dribbble.com/tags/neumorphism)*

→ **UI8 Soft UI Kits —** Downloadable neumorphism design kits showing real app screens *(ui8.net/tags/neumorphism)*

1.3 Brutalism

What it is

Brutalist web design is intentionally raw, ugly-on-purpose, and anti-establishment. It rejects the polished corporate design of most tech companies and embraces thick black borders, stark primary colors (red, yellow, black), system fonts like monospace, and asymmetric or tilted layouts. It's the punk rock of web design.

How it works

No rounded corners. No gradients. No subtle shadows. Instead: solid 2-4px black borders on everything, backgrounds of pure yellow (#FFE600) or pure white, text in bold black, elements that are slightly rotated with CSS transform:rotate, and a general sense that someone typed the HTML in Notepad in 1998. Paradoxically, it takes skill to do brutalism WELL.

transform:rotate()	Tilts elements by a few degrees — a hallmark brutalism detail
Monospace font	Fixed-width font like Courier that looks like a typewriter
No border-radius	Sharp corners only — no rounded shapes anywhere
Primary colors	Pure red, yellow, blue, black, white — no pastels or muted tones

AI PROMPT TO USE

"Build a brutalist portfolio website with a pure white #FFFFFF background. Use bold black borders (3px solid #000) on every card and section. Typography should be black and oversized (hero headline at 80px, font-weight 900). Accent color is yellow (#FFE600) used for hover states and highlighted sections. Add slight rotation (transform: rotate(-1deg) to -2deg) on some image cards. Use a monospace font for body text. No gradients, no drop shadows, no border-radius anywhere."

EXAMPLE WEBSITES TO STUDY

→ **Gumroad.com —** The most famous modern brutalist website — product selling platform with thick borders *(gumroad.com)*

→ **Figma — Community Brutalism UI —** Search 'brutalism' in Figma Community for ready-made templates to study *(figma.com/community)*

→ **Hype4.academy —** Web design learning platform with a slightly brutalist aesthetic *(hype4.academy)*

1.4 Minimalism

What it is

Minimalism is the philosophy that content is the design. Everything that doesn't serve the content is removed. What remains is: generous white (or off-white) space, a single typeface, a monochromatic or near-monochromatic color palette, and zero decoration. The whitespace itself creates visual rhythm and breathing room.

How it works

Think of minimalism as removal. You start with a design and keep asking 'can this be removed without losing meaning?' Large padding and margins, a single font used at multiple weights (light, regular, bold), near-black text on near-white background, and content arranged with clear typographic hierarchy. No icons, no illustrations, no gradients — unless they serve a specific purpose.

Whitespace	Empty space used intentionally as a design element — not wasted space
Typographic hierarchy	Using size, weight, and spacing alone to show what's important
Monochromatic	Using one color at different shades — like all blacks, grays, and whites
Negative space	The empty areas around and between objects — as important as the objects themselves

AI PROMPT TO USE

"Design a minimalist photography portfolio homepage. Use #FAFAF9 (warm off-white) as the background. Typography only: a single font family (Cormorant Garamond or Playfair Display), hero headline at 72px weight 300, section titles at 32px. Color palette: near-black #111 for text, #888 for captions. No icons, no decorative elements, no gradients. Large padding on all sections (80px vertical). Image grid with generous white space between photos. Let the whitespace be the design."

EXAMPLE WEBSITES TO STUDY

→ **Awwwards — Minimalist Category —** Award-winning minimal websites — filter by 'minimalism' *(awwwards.com/websites/minimal)*

→ **Linear.app —** Project management tool — one of the cleanest, most minimal SaaS designs *(linear.app)*

→ **Typewolf.com —** Typography-focused minimal design inspiration, all real websites *(typewolf.com)*

1.5 Bento Grid

What it is

A bento grid is a layout (and aesthetic) inspired by a Japanese bento box — a grid of varied-size compartments, each showing a different feature or piece of content. Popularized by Apple's Mac product pages, it's now one of the most popular design trends for SaaS and product landing pages. Think of it as a mosaic of cards, each with its own design, creating a visual collage.

How it works

CSS Grid is the technical foundation. Some cards span 2 columns, others are tall and span 2 rows, some are small and square. Cards have different visual treatments: some dark (#111), some light (#F5F5F5), some with gradient backgrounds, some with product screenshots, some with large typography and a stat. The variety in size AND color is what makes it feel premium.

grid-column: span 2	Makes a card take up 2 columns of the grid instead of 1
grid-row: span 2	Makes a card take up 2 rows — creates a tall featured card
Varied treatments	Each card has its own background — dark, light, gradient, or image
Apple-style	Usually on a black or very dark gray page background for max contrast

TIP The key to a great bento grid is the mix of big and small. Don't make all cards the same size — you want 1-2 large featured cards (span 2 columns or 2 rows) and 3-4 smaller supporting cards. The large cards should hold your most impressive content.

AI PROMPT TO USE

"Build a bento grid feature section on a #0A0A0A (near-black) background. Create a responsive CSS Grid with 3 columns and auto rows. Include 6 cards of varying sizes: one large card spanning 2 columns showing a product screenshot, one tall card spanning 2 rows with a bold stat (like '10x faster'), and 4 smaller cards each highlighting one feature. Dark cards use #111111 background, light accent cards use #1A1A2E. Round all cards with 16px border-radius. Add hover scale effect (transform: scale(1.02))."

EXAMPLE WEBSITES TO STUDY

→ **Apple Mac product page —** The gold standard of bento grids — study how Apple varies card sizes and treatments *(apple.com/mac)*

→ **Linear.app homepage —** Dark bento-style feature grid for a SaaS product *(linear.app)*

→ **Framer.com —** Website builder with a beautiful bento grid on their homepage *(framer.com)*

→ **Dribbble — Bento Grid UI —** Search 'bento grid' for hundreds of design references *(dribbble.com/tags/bento)*

1.6 Dark Mode / Dark Luxury

What it is

Dark luxury uses deep charcoal or near-black backgrounds to create a premium, sophisticated atmosphere. Think of a high-end watch brand website or a premium SaaS product. It's not just dark mode — it's a deliberate aesthetic choice that uses darkness to make accent colors and product screenshots pop dramatically.

How it works

Background is near-black (#0A0A0A or #0D0D1A), NOT pure black (#000000) which can feel harsh. Cards and sections use slightly lighter darks (#111111, #1A1A2E) to create subtle depth. A single accent color — often electric blue (#0066FF), amber (#F59E0B), or soft purple (#8B5CF6) — is used sparingly for CTAs, icons, and highlights. Typography is pure white or

near-white (#F1F1EE). Thin borders at low opacity (rgba(255,255,255,0.08)) separate sections softly.

Near-black	#0A0A0A or #0D0D1A — not pure black, which can feel harsh on screens
Accent color	One bright color used ONLY for important elements — CTAs, icons, highlights
Border opacity	Very low opacity borders like rgba(255,255,255,0.08) for subtle separation
Subtle gradient	A very faint radial gradient in the background gives depth without being garish

AI PROMPT TO USE

"Build a dark luxury SaaS landing page. Background: #0D0D1A (deep dark blue-black). Primary text: #F1F1EE. Accent color: electric blue #0066FF for buttons and highlighted text. Cards use #111827 background with a 1px border at rgba(255,255,255,0.08). Add a very subtle radial gradient in the hero: a soft purple glow at 20% opacity centered behind the hero text. Clean sans-serif typography (Inter or Satoshi). CTA button: #0066FF with white text and a subtle glow effect (box-shadow: 0 0 30px rgba(0,102,255,0.4))."

EXAMPLE WEBSITES TO STUDY

→ **Vercel.com —** Near-perfect dark luxury SaaS design — study every detail of their homepage *(vercel.com)*

→ **Raycast.com —** Mac productivity app with stunning dark luxury aesthetic *(raycast.com)*

→ **Stripe.com —** Premium dark sections mixed with light sections — a masterclass *(stripe.com)*

1.7 Retro / Y2K

What it is

Retro and Y2K design draws nostalgia from the 90s and early 2000s internet — neon gradients, pixel art, bold typography, starfield backgrounds, and the visual energy of early web design before everyone agreed on 'good taste'. Ironically, Gen Z designers have made this one of the hottest aesthetics right now.

How it works

Key visual ingredients: hot pink (#FF00FF) and cyan (#00FFFF) or bright lime green (#AAFF00) as main colors, often on a black or very dark background. Pixel or display fonts with a technological or retro feel. Star/sparkle decorations scattered across the page. Bold italic type. Bright gradients that look unafraid. Sometimes actual animated GIF-style elements. Chrome/metallic text effects.

AI PROMPT TO USE

"Create a Y2K-inspired portfolio website on a #0A0A0A background. Color palette: hot pink #FF1493, electric cyan #00E5FF, and neon lime #AAFF00 as accents. Use a bold display font like Bebas Neue or Black Han Sans for headings. Add CSS sparkle/star decorations (small ✦ symbols) scattered on

the page. Hero headline in chrome metallic gradient text (silver to white). Sections separated by bright gradient dividers. Navigation links in bold uppercase with a neon hover underline effect."

EXAMPLE WEBSITES TO STUDY

→ **Y2K Aesthetic Community on Dribbble —** Search 'Y2K web design' for hundreds of references *(dribbble.com/tags/y2k)*

→ **Brat Girl Studio —** Real brand that fully commits to bright Y2K colors and energy *(bratgirlstudio.com)*

→ **Awwwards — Experimental category —** Find retro and experimental sites that push beyond conventional design *(awwwards.com)*

1.8 Claymorphism / 3D UI

What it is

Claymorphism makes UI elements look inflated, puffy, and three-dimensional — like they're made of clay or rubber. Buttons and cards appear to have physical volume and weight. This style creates a playful, friendly, and approachable feel, making it popular for apps targeting general consumers, productivity tools, and any product that wants to feel fun rather than corporate.

How it works

The 3D effect is created with multiple box-shadows: a bottom shadow in a darker version of the element's color (simulating the underside), and a larger soft glow shadow around the whole element. Candy colors (soft pastels or vivid saturated colors) are typical. High border-radius (24px+) on everything. When the button is clicked, the bottom shadow shrinks, making it feel like it's being physically pressed down.

Multi-layer shadow	Two or three box-shadows at once: a hard bottom one for depth, soft ambient glow
Candy colors	Bright but slightly soft colors — purple, coral, mint, sky blue
Press state	On click, bottom shadow disappears, element moves down by the shadow height
High border-radius	24-32px corners — very rounded, almost pill-shaped for small elements

AI PROMPT TO USE

"Design a claymorphism task management app UI on a soft #F0F4FF background. Create 3D-looking cards and buttons using multi-layer box-shadows: box-shadow: 0 8px 0 #4B44CC, 0 14px 20px rgba(108,99,255,0.35) for a raised purple button. Use candy colors: soft purple #6C63FF, coral #FF6B6B, mint #4ECDC4. All elements have border-radius: 20px minimum. Add pressed animation on buttons: transform: translateY(4px) with reduced shadow to simulate physical pressing. Background is a soft light blue-gray."

EXAMPLE WEBSITES TO STUDY

→ **Cuberto Agency — 3D UI Kits —** Premium 3D and claymorphism UI kits with real app screen examples *(cuberto.com)*

→ **Dribbble — Claymorphism —** Search 'clay UI' or 'claymorphism' for reference designs *(dribbble.com/tags/claymorphism)*

→ **Notion.so —** Subtle 3D elements in their marketing — good example of refined claymorphism *(notion.so)*

2. Scroll & Animation

Scroll and animation are what separate a static website from an experience. These techniques control how elements appear, move, and respond as the user scrolls through your page. Used well, they guide attention, reveal content at the right moment, and make a site feel premium. Used poorly, they're just distracting.

2.1 Parallax Scrolling

What it is

Parallax creates an illusion of depth by making different layers of the page scroll at different speeds. The background moves slowly while the foreground (text, buttons) moves at normal speed. This makes the page feel three-dimensional — like looking through a window where distant objects appear to move slower than close ones.

How it works

The simplest version uses CSS background-attachment: fixed on a background image, making it stay put while the content scrolls over it. More advanced parallax uses JavaScript to track how far the user has scrolled and applies a CSS transform: translateY() at a fraction of that scroll distance to each layer. Three layers minimum: far background (moves 20% of scroll speed), mid background (moves 50%), foreground text (moves 100%).

AI PROMPT TO USE

*"Build a parallax hero section with 3 scrolling layers. Background layer: a mountain/nature image that moves at 30% of scroll speed (translateY by 0.3 * scrollY). Middle layer: floating abstract blob shapes that move at 60% speed. Foreground layer: hero headline and CTA that scroll normally at 100%. Use JavaScript IntersectionObserver or requestAnimationFrame to update transforms on scroll. The depth illusion should be clearly visible when scrolling through the section."*

EXAMPLE WEBSITES TO STUDY

→ **Firewatch Game Website —** The most famous parallax website ever — multiple illustrated layers *(firewatchgame.com)*

→ **Webflow University — Parallax Tutorial —** No-code parallax scroll tutorial, great for understanding the mechanics *(university.webflow.com)*

→ **Awwwards — Parallax Sites —** Award-winning websites using advanced parallax effects *(awwwards.com/websites/parallax)*

2.2 Scroll-Triggered Animations

What it is

Elements are hidden (invisible and/or off-position) when the page loads. As the user scrolls down and a section enters the visible area of the screen (the 'viewport'), that element animates

into place. This is by far the most common animation technique on modern websites. The classic version is 'fade up' — element starts invisible and slightly below its final position, then fades in and slides up.

How it works

The IntersectionObserver JavaScript API watches when elements enter or leave the viewport. When an element enters, a CSS class is added to it (like .visible). That class triggers a CSS transition or animation that plays the reveal. The element starts with opacity:0 and transform:translateY(40px), then transitions to opacity:1 and translateY(0). The timing — how long the animation takes and its easing — determines whether it feels smooth or janky.

IntersectionObserver	A JavaScript API that fires a callback when an element enters/leaves the viewport
opacity: 0 → 1	Element goes from invisible to fully visible — the foundation of fade animations
translateY(40px) → 0	Element moves upward as it reveals — the 'rise' in fade-up
Stagger	Delaying each element slightly more than the previous — creates a cascading effect
AOS library	Animate On Scroll — a simple JavaScript library that handles all this automatically

AI PROMPT TO USE

"Build a features page where every card and section animates into view on scroll. Use IntersectionObserver to detect when each element is 20% visible. Apply these default start states: opacity:0 and transform:translateY(30px). On intersection, add class .visible which transitions to opacity:1, transform:translateY(0) with transition duration 0.6s and ease: cubic-bezier(0.4, 0, 0.2, 1). For 3-column feature grids, stagger each card by 0.1s extra delay (first card 0s, second 0.1s, third 0.2s)."

EXAMPLE WEBSITES TO STUDY

→ **Stripe.com —** Best-in-class scroll animations on every section — study how subtle and purposeful they are *(stripe.com)*

→ **AOS (Animate On Scroll) Demo —** Interactive demo of every scroll animation type this library supports *(michalsnik.github.io/aos)*

→ **GSAP ScrollTrigger Demos —** The most powerful scroll animation library — see what's possible *(gsap.com/scroll)*

2.3 Horizontal Scroll Sections

What it is

While the page scrolls vertically (up-down), a specific section 'hijacks' the scroll and moves the content sideways instead. So as you scroll DOWN with your mouse or finger, the content inside

that section slides from RIGHT to LEFT. It feels like flipping through cards or reading a horizontal timeline. Used a lot on portfolio sites, case study pages, and product showcases.

How it works

The section is 'pinned' in place on the screen while the user scrolls. As the user scrolls the required distance, the inner content (a row of cards or panels) moves horizontally. The total scrolling distance controls how far the horizontal movement goes — if you have 5 cards each 100vw wide, the section stays pinned for 500vh of scrolling. Typically implemented with GSAP ScrollTrigger's pin and horizontal scrub features.

AI PROMPT TO USE

"Create a horizontal scroll portfolio section that pins to the screen while the user scrolls vertically. The section should contain 5 project cards, each 80vw wide. As the user scrolls down through 500px of page height, the cards slide horizontally from right to left revealing each project. Use GSAP ScrollTrigger with pin:true and scrub:1 for a smooth tied-to-scroll feel. Each card has a project screenshot (top 60%), project title, brief description, and 'View Project' link. Add a horizontal progress bar at the bottom of the section."

EXAMPLE WEBSITES TO STUDY

→ **Active Theory Agency —** Award-winning agency with complex horizontal scroll portfolio *(activetheory.net)*

→ **Locomotive Scroll Examples —** Smooth scroll library with horizontal scroll demos *(locomotivemtl.github.io/locomotive-scroll)*

→ **Webflow — Horizontal Scroll Template —** No-code horizontal scroll examples you can study and clone *(webflow.com/templates)*

2.4 Sticky Sections / Pinning

What it is

An element stays fixed on the screen while the user scrolls, but ONLY within its parent section. Once the user scrolls past that section, the element releases and scrolls away normally. This is called 'pinning'. It's perfect for feature explanations where you want to show one product screenshot on the left while the user scrolls through a list of features on the right.

How it works

The simplest implementation uses CSS position:sticky top:0, which keeps the element at the top of the viewport as long as its parent is in view. A common layout has two columns: left column with position:sticky (the persistent visual — a phone mockup, screenshot, or graphic) and right column that scrolls normally (the feature list). As the user scrolls through features on the right, the visual on the left stays put and can change to reflect each feature.

AI PROMPT TO USE

"Design a SaaS feature explanation section with a sticky scroll layout. Left column (40% width): a phone mockup or browser window screenshot with position:sticky top:80px that stays in view. Right column (60% width): 4 feature blocks stacked vertically, each 400px tall, with icon, heading, and 2-line description. As the user scrolls through each feature block on the right, use IntersectionObserver to detect which block is most visible and update the left screenshot to show the corresponding product UI screen with a smooth crossfade transition."

EXAMPLE WEBSITES TO STUDY

→ **Notion.so Features page —** Classic sticky scroll — left screenshot stays while feature text scrolls right *(notion.so)*

→ **Linear.app —** Excellent sticky feature sections with animated product screenshots *(linear.app)*

→ **Lottiefiles.com —** Uses sticky panels with animated Lottie illustrations alongside scrolling text *(lottiefiles.com)*

2.5 Text Reveal on Scroll

What it is

Instead of the whole heading appearing at once, individual words or characters animate in one by one as the user scrolls. This is very popular on agency websites, portfolio sites, and anywhere that wants to create a dramatic, editorial feel. The text seems to 'write itself' or 'emerge' as the user scrolls through.

How it works

A JavaScript library called SplitType.js takes a heading like 'Hello World' and splits it into individual spans — one per word or even per character. Then GSAP or CSS animations target those spans with a stagger — each one delayed slightly more than the previous. The reveal effect itself can be: fade from opacity 0, slide up from below a clipping mask, or a blurry-to-sharp focus transition.

AI PROMPT TO USE

"Create a hero section with a word-by-word text reveal animation. The main headline should be split into individual word spans using a JavaScript loop. On page load (or scroll trigger), each word starts at opacity:0 and transform:translateY(20px), then transitions to opacity:1 and translateY(0). Apply a 0.08 second stagger between each word (word 1 at 0s, word 2 at 0.08s, word 3 at 0.16s, etc.). Use transition-timing-function: cubic-bezier(0.4,0,0.2,1) for a smooth, professional feel. Total animation duration per word: 0.5 seconds."

EXAMPLE WEBSITES TO STUDY

→ **Resn.co.nz —** New Zealand agency — stunning text reveal animations throughout *(resn.co.nz)*

→ **SplitType.js Demo —** Library documentation with live demos of text splitting and animation *(split-type.vercel.app)*

→ **Awwwards — Typography category —** Filter for sites winning awards for typography and text animation *(awwwards.com/websites/typography)*

2.6 Scroll Progress Indicator

What it is

A thin bar, usually at the very top of the page, that fills from left to right as the user reads down the page. When the user is at the top, the bar is empty. When they reach the bottom, the bar is full. It gives readers a sense of how far through the content they are, particularly useful on long blog posts and articles.

How it works

A fixed-position element sits at the top of the viewport (position:fixed, top:0, left:0, height:3-4px). Its width is calculated as a percentage: (scrollY / (document.body.scrollHeight - window.innerHeight)) * 100. This formula gives 0% when at the top and 100% when fully scrolled. A JavaScript scroll event listener updates the width in real-time. Using CSS transform:scaleX() instead of changing width is smoother on most devices.

AI PROMPT TO USE

"Add a reading progress bar to a blog post page. Create a fixed element: position:fixed, top:0, left:0, height:3px, width:100%, z-index:9999. Inside it, a div that starts at transform:scaleX(0) and transform-origin:left. On scroll event, calculate progress: scrolled / (totalHeight - viewportHeight). Update the inner div's scaleX to the progress value (0 to 1). Use a gradient fill from purple #667eea to pink #f64f59. Add a smooth transition of 0.1s on the scale for a slight lag that feels natural."

EXAMPLE WEBSITES TO STUDY

→ **Medium.com articles —** Medium uses a reading progress bar — the standard reference for this pattern *(medium.com)*

→ **CSS Tricks article on progress bars —** Technical breakdown of how to build one, with code examples *(css-tricks.com)*

→ **Any long-form blog —** Scroll down any long article on major blogs — many now have this built in *(smashingmagazine.com)*

2.7 Smooth Scroll / Lenis

What it is

By default, browsers scroll in a very mechanical, instant way — you flick the wheel and the page jumps. Smooth scroll replaces this with a physics-based momentum system: when you scroll, the page decelerates smoothly instead of stopping abruptly, like a heavy object slowly coming to rest. Lenis is the most popular modern JavaScript library for this. When you visit a website and think 'this scroll feels incredible, so buttery' — that's almost always Lenis.

How it works

Lenis works by overriding the browser's native scroll. It intercepts scroll input (mouse wheel, touch swipe, trackpad), applies its own easing and momentum math, and then moves the page via CSS transform instead of actual scrolling. This is smoother because CSS transforms are GPU-accelerated. Lenis is initialized with a few lines of JavaScript and runs on every animation frame using requestAnimationFrame. It integrates directly with GSAP ScrollTrigger — you just pass Lenis's scroll position to GSAP and all your scroll-triggered animations work as expected.

Lenis	The most popular smooth scroll JavaScript library — free and widely used
requestAnimationFrame	A browser API that fires a callback 60 times per second — how Lenis stays smooth
Easing	The math curve controlling deceleration — cubic-bezier controls the 'feel'
lerp	Linear interpolation — how Lenis blends current position toward target position
GSAP compatibility	Lenis passes its scroll position to GSAP so ScrollTrigger animations still work

TIP Smooth scroll is a FEEL upgrade, not a visual one. Users might not notice it consciously, but the site feels premium and polished. It's one of the cheapest ways to make a website feel high-end. However: never apply smooth scroll to pages with fixed-height modals or overflow:hidden sections — it can cause visual glitches.

AI PROMPT TO USE

*"Add Lenis smooth scrolling to this website. Import Lenis from CDN: <script src='https://unpkg.com/lenis@1.1.13/dist/lenis.min.js'></script>. Initialize with: const lenis = new Lenis({ duration: 1.2, easing: (t) => Math.min(1, 1.001 - Math.pow(2, -10 * t)), smooth: true }). Run in animation loop: function raf(time) { lenis.raf(time); requestAnimationFrame(raf); } requestAnimationFrame(raf). If GSAP ScrollTrigger is used on this page, also add: lenis.on('scroll', ScrollTrigger.update) and gsap.ticker.add((time) => { lenis.raf(time * 1000) })."*

EXAMPLE WEBSITES TO STUDY

→ **Lenis by Studio Freight — Demo —** The official Lenis demo page — feel the difference between native and smooth scroll *(lenis.darkroom.engineering)*

→ **Awwwards — Sites using Lenis —** Filter Awwwards sites by 'smooth scroll' to find the best implementations *(awwwards.com)*

→ **Locomotive Scroll —** Alternative smooth scroll library with its own parallax system — older but widely used *(locomotivemtl.github.io/locomotive-scroll)*

---

3. Layouts

Layout is how you organize information on the page — where things go and how they relate to each other. A great layout guides the reader's eye from the most important element to the least important, without them noticing. Layout decisions include: how many columns, what's in the center vs edges, how much space between elements, and what's above the fold (visible without scrolling).

3.1 F-Pattern Layout

What it is

Eye-tracking research has shown that when people read web pages, their eyes don't scan every word equally. Instead, they trace an F-shape: first they scan across the very top of the page (the top bar of the F), then they scan a shorter horizontal band a little lower (the middle bar of the F), then their eyes drift down the left edge of the page reading only the first few words of each line. Understanding this pattern helps you place your most important content where eyes naturally fall.

How it works

The practical takeaway is: put your most important content in the top horizontal band (the headline, navigation, hero message). Put your second most important element below that but still near the top-left. Your left margin and left edge of content are seen far more than the right. Content on the far right of the page is often never seen at all by users who are scanning rather than reading. This means left-aligned text, left-placed CTAs, and left-column navigation are all strategically placed where eyes actually go.

Above the fold	The top portion visible without scrolling — highest attention zone
Left edge priority	Left column and left-aligned text gets far more attention than right-aligned
Visual anchors	Bold text, images, and icons create stopping points that pull the eye
Z-pattern	For pages with less text (landing pages), eyes scan in a Z rather than F
Scanning vs reading	Most users scan first, read only if interested — design for scanners first

TIP The F-pattern is strongest on text-heavy pages like articles and search results. On landing pages with lots of imagery, a Z-pattern (top-left → top-right → bottom-left → bottom-right diagonal) is more common. For marketing pages: center your hero for visual impact, then switch to left-aligned layouts below the fold where reading begins.

AI PROMPT TO USE

"Design a long-form blog article layout following the F-pattern reading behavior. Full-width header with article title (left-aligned, not centered). Below: a two-column layout — left column 68% width for article body (left-aligned text, comfortable line length of 65-70 characters max, 18px body text). Right

column 28% width for a sticky table of contents sidebar that stays visible as you read. In the article body, use bold lead sentences at the start of each paragraph, subheadings every 200-300 words, and inline pull-quotes to create visual stopping points. Place key CTAs (email signup, related articles) in the left-aligned body column, not in the right sidebar."

EXAMPLE WEBSITES TO STUDY

→ **Medium.com — Article layout —** Clean left-aligned article layout designed around natural reading behavior *(medium.com)*

→ **Smashing Magazine —** Expert-level article layout with strategic left-aligned hierarchy *(smashingmagazine.com)*

→ **Nielsen Norman Group — F-Pattern research —** The original eye-tracking research paper that identified the F-pattern, with heatmap images *(nngroup.com/articles/f-shaped-pattern-reading-web-content)*

3.2 Hero + Feature Grid

What it is

The single most common layout for SaaS (Software as a Service) and product websites. At the top: a large, bold Hero section that states the main value proposition with a CTA button. Below: a grid of Feature cards (usually 3 columns, 2-3 rows) each explaining one capability of the product. This layout has been battle-tested and converts extremely well.

How it works

Hero section: full-width, center-aligned, bold H1 headline (max 8 words), one-sentence subtext, and a primary CTA button. Below this, often a strip of company logos ('Trusted by X and Y'). Then the feature grid: 3 equal columns, each card with an icon, a 3-5 word title, and a 1-2 sentence description. The grid can have 2-4 rows of cards depending on how many features you're highlighting.

AI PROMPT TO USE

"Build a complete SaaS landing page layout. Hero section: dark background (#0A0A0A), centered content, badge chip at top ('Now in Beta'), H1 headline at 64px bold with a gradient word, 1-line subtext, and two buttons side by side (primary: filled blue, secondary: outlined). Below: a row of 5-6 company logos in light gray. Then a feature grid: 3-column responsive CSS grid, each card has a small colored icon, bold feature title, 2-sentence description. 6 cards total. Below that: a testimonial quote. Then a pricing section. Finish with a CTA banner and footer."

EXAMPLE WEBSITES TO STUDY

→ **Vercel.com —** The reference standard for SaaS landing pages — hero + feature grid done perfectly *(vercel.com)*

→ **Tailwind UI Components —** Pre-built hero and feature grid components to study the structure *(tailwindui.com/components)*

→ **Landingfolio.com —** Gallery of 1000+ landing page designs for inspiration — all tagged by layout type *(landingfolio.com)*

3.3 Asymmetric / Split Layout

What it is

Instead of dividing the page into equal 50/50 halves, asymmetric layouts use unequal proportions — like 60/40, 65/35, or 70/30. One side gets the dominant visual (large image, video, illustration), while the other side gets supporting content (text, stats, CTA). The imbalance creates visual tension that makes the layout feel dynamic and intentional rather than sterile and corporate.

How it works

Using CSS Grid with non-equal column definitions: grid-template-columns: 3fr 2fr (60/40) or 2fr 1fr (67/33). The larger column holds the hero image or product screenshot. The smaller column holds the headline, description, and CTA. On mobile, both columns stack to 100% width. The key visual design principle here is contrast — a large, bold visual on one side creates focus, while the smaller text column creates breathing room and guides the eye to the CTA.

AI PROMPT TO USE

"Create a product landing page using a 60/40 asymmetric split layout. Left side (60%): full-height product screenshot or mockup with subtle shadow, slightly overlapping the right column. Right side (40%): vertically centered content with a small overline label (like 'NEW FEATURE'), H2 heading, 2-paragraph description, feature checklist with checkmarks, and a CTA button. Use CSS Grid: grid-template-columns: 3fr 2fr. Add a second split section below with the layout reversed (image on right, text on left) for visual variety."

EXAMPLE WEBSITES TO STUDY

→ **Notion.so Marketing pages —** Excellent use of asymmetric split layouts across their feature pages *(notion.so)*

→ **Dribbble — Split Layout —** Search 'split layout web design' for visual references *(dribbble.com)*

→ **Lapa.ninja —** Curated landing page designs — many use asymmetric splits beautifully *(lapa.ninja)*

3.4 Masonry Layout

What it is

Named after the way bricklayers stack bricks, a masonry layout arranges cards in columns where each card's height is determined by its content. Short items fill in below taller items in adjacent columns, rather than leaving empty space. This creates a Pinterest-style mosaic that feels organic and content-rich, making it perfect for photo galleries, blog posts, design portfolios, and social feeds.

How it works

The simplest CSS approach uses the CSS columns property: columns: 3 on the container makes content flow into 3 columns automatically, with each item taking only as much height as it needs. Items flow down the first column, then the second, then third. For more control, JavaScript libraries like Masonry.js or Packery calculate and position each item manually. The result is the same: a grid where nothing is wasted.

AI PROMPT TO USE

"Build a photography portfolio gallery using masonry layout. Use CSS columns: 3 with column-gap: 12px on the container. Each image card should be display:inline-block, width:100%, margin-bottom:12px. Images should be different heights (some portrait, some landscape) to show the masonry effect clearly. Add a hover overlay on each image showing a brief caption and a zoom icon. On mobile (under 768px), reduce to columns:2. Under 480px, reduce to columns:1. Include a category filter bar at the top (All, Architecture, Portraits, Nature) that filters which images show."

EXAMPLE WEBSITES TO STUDY

→ **Pinterest.com —** The masonry layout — this is exactly where the style became mainstream *(pinterest.com)*

→ **Unsplash.com —** Beautiful masonry photo grid — notice how differently-sized photos flow together *(unsplash.com)*

→ **Masonry.js documentation —** The classic JavaScript masonry library with demos and code examples *(masonry.desandro.com)*

3.5 Full-Page Scroll Sections

What it is

Each section of the website fills the entire screen (100% of the viewport height, called 100vh). When the user scrolls, the page snaps to the next full-screen section — like flipping between slides in a presentation. There's no partial scrolling; you're always viewing exactly one section at a time. This creates a very controlled, cinematic storytelling experience.

How it works

CSS makes this surprisingly simple. The container gets scroll-snap-type: y mandatory and overflow-y: scroll. Each child section gets height: 100vh and scroll-snap-align: start. The browser then handles the snapping automatically. For more features (navigation dots, transitions, callbacks), the fullPage.js library handles everything. Each section typically has one focused message, one visual, and one CTA.

AI PROMPT TO USE

"Build a 5-section fullpage scroll website using CSS scroll-snap. Container: height:100vh, overflow-y:scroll, scroll-snap-type:y mandatory. Each section: height:100vh, scroll-snap-align:start. Section 1: Dark hero with headline and CTA. Section 2: Feature explanation with image left, text right. Section 3: Statistics with 3 large numbers. Section 4: Testimonial quote centered with author. Section 5: Final CTA with newsletter signup. Add navigation dots on the right side: fixed position, 5 small dots, active dot is filled white, clicking a dot smoothly scrolls to that section using scrollIntoView."

EXAMPLE WEBSITES TO STUDY

→ **FullPage.js Examples —** The most popular fullpage scroll library — dozens of examples showing the pattern *(alvarotrigo.com/fullPage/examples)*

→ **Webflow — Full Page Templates —** No-code full-page scroll website templates *(webflow.com/templates)*

→ **Sony BE MOVED campaign —** Classic example of fullpage scroll storytelling for a product *(sony.com)*

3.6 Sidebar + Content Layout

What it is

A persistent navigation panel on the left side of the screen, with the main working area on the right. This is the universal layout for web applications, dashboards, and admin interfaces. The sidebar stays fixed as the main content area scrolls. The sidebar typically contains the app's main navigation, while the content area shows whatever page is currently selected.

How it works

The outer container uses display:flex or CSS Grid with two columns: a fixed-width sidebar (240-280px) and a flex-1 or fr content area that fills the remaining space. The sidebar gets position:sticky top:0 with height:100vh so it stays visible while the content scrolls. The sidebar contains the logo at top, navigation links in the middle (with icons), and user profile at the bottom. The content area has its own internal scrolling.

AI PROMPT TO USE

"Build a SaaS dashboard layout with a sidebar + content structure. Sidebar: 240px wide, dark background (#111827), fixed to left side, height:100vh. Contains: logo at top, navigation links with icons (Dashboard, Analytics, Projects, Settings, Team) with active state highlighted in blue, and user avatar + name at bottom. Main content area: flex-1, light gray background (#F9FAFB), has a top header bar with page title and action buttons. Content below shows: 4 stat cards in a row (Users, Revenue, Orders, Growth), then a data table. On mobile, sidebar collapses behind a hamburger menu."

EXAMPLE WEBSITES TO STUDY

→ **Vercel Dashboard —** The reference for modern SaaS dashboards — sidebar + content done perfectly *(vercel.com/dashboard)*

→ **Tailwind UI — Application UI —** Pre-built sidebar dashboard components showing all variations *(tailwindui.com/components#application-ui)*

→ **AdminLTE Demo —** Open-source admin dashboard template showing complex sidebar layouts *(adminlte.io)*

4. Navigation

Navigation is how users find their way around your website. Good navigation is invisible — users don't notice it because they always know where they are and how to get where they want. Bad navigation is noticed immediately because users get lost or frustrated. The navigation pattern you choose depends on the complexity of your site and the devices it's used on.

4.1 Sticky Navbar

What it is

A navigation bar that stays fixed to the top of the viewport as the user scrolls. Unlike a static nav that scrolls away with the page, a sticky nav is always accessible no matter how far down the user has scrolled. This is the most common and expected navigation pattern for marketing websites and landing pages.

How it works

position:sticky top:0 keeps the navbar at the top of the viewport when within its parent. A popular enhancement: the navbar starts transparent (showing the hero content behind it), and as the user scrolls even a few pixels, it transitions to a solid or frosted-glass background. This is detected by listening to the scroll event and checking if window.scrollY is greater than 0 (or some threshold like 80).

AI PROMPT TO USE

"Create a sticky navbar that is transparent at the very top of the page (background:transparent) and transitions to a white frosted-glass background when the user scrolls past 60px (background:rgba(255,255,255,0.85), backdrop-filter:blur(12px), box-shadow: 0 1px 20px rgba(0,0,0,0.08)). Navigation has: logo on the left, 5 nav links in the center, and a CTA button on the right. Add a 0.3s transition on background-color and backdrop-filter. On mobile (under 768px), hide the center links and show a hamburger menu icon instead."

EXAMPLE WEBSITES TO STUDY

→ **Apple.com —** The sticky frosted-glass navbar — the most copied navbar design in the world *(apple.com)*

→ **GitHub.com —** Clean sticky navbar with excellent mobile behavior *(github.com)*

→ **Framer.com —** Beautiful sticky nav with a smooth scroll-based transparency effect *(framer.com)*

4.2 Hamburger / Mobile Menu

What it is

On mobile screens, there isn't room for all navigation links. The hamburger menu (named after its resemblance to a hamburger when viewed from the side: bun, patty, bun = three horizontal

lines) hides all navigation behind a single icon. Tapping it reveals the full navigation — either as a fullscreen overlay, a side drawer, or a dropdown panel.

How it works

Three horizontal div elements (or a custom SVG) form the icon. When clicked, JavaScript toggles a class on a fullscreen overlay element (display:none → display:flex). The three lines animate into an X using CSS transforms: the top line rotates 45 degrees, the middle line fades to opacity:0, the bottom line rotates -45 degrees. The overlay menu typically appears with a fade or slide-in transition.

AI PROMPT TO USE

"Build a fullscreen hamburger menu for mobile. Three-line hamburger icon: each line is 24px wide, 2px tall, dark color, with 5px gap between them. When clicked, animate to X: top line rotates 45deg and translates down, middle line fades to opacity:0, bottom line rotates -45deg and translates up. The fullscreen menu overlay uses position:fixed, inset:0, background:#0A0A0A, z-index:9999. Navigation links appear centered, large (40px), white, with a stagger fade-in (each link 0.05s later). Include close button (X) in top right. Body scroll should be locked when menu is open (overflow:hidden on body)."

EXAMPLE WEBSITES TO STUDY

→ **Many agency websites —** Most creative agency sites use fullscreen hamburger menus — search 'agency portfolio' on Awwwards *(awwwards.com/websites/portfolio)*

→ **Codrops — Menu Animations —** Collection of creative hamburger menu animation techniques with code *(tympanus.net/codrops)*

→ **Cuberto.com —** Russian agency with beautiful fullscreen hamburger menu animations *(cuberto.com)*

4.3 Mega Menu

What it is

A mega menu is a large dropdown panel that appears when you hover over (or click) a navigation item. Unlike a simple dropdown with a list of links, a mega menu can contain multiple columns of links, featured images, highlighted promotions, icons, and even mini descriptions. They're used on complex websites where a single nav item leads to many sub-pages — common on e-commerce sites ('Shop' opens a panel with all categories), documentation sites, and large corporate websites.

How it works

When the user hovers a navigation link (or clicks on mobile), an absolutely-positioned full-width or wide panel appears below the nav bar. This panel is typically organized into 3-5 columns, each containing a category heading and 4-8 links below it. One column might be a 'Featured' panel with an image and a highlighted promotion. The panel appears with a subtle fade or slide-in animation. On desktop it's hover-triggered; on mobile, a click/tap opens it.

position:absolute	The mega menu panel is absolutely positioned below the nav bar
Hover trigger	Desktop: hover on nav link shows the panel. Mobile: tap/click instead.
Multi-column grid	Inside the panel, links are organized into 3-5 equal columns using CSS Grid
Featured column	One column with an image and CTA — draws attention to a promoted item
z-index	Mega menu must have high z-index (like 9000) to appear above all page content

AI PROMPT TO USE

"Create a mega menu for an e-commerce fashion website. Navbar has links: Women, Men, Kids, Sale. When hovering 'Women': a full-width panel drops down (background white, box-shadow: 0 20px 40px rgba(0,0,0,0.1), padding:40px). Panel contains a 4-column CSS grid: Column 1 'Clothing' with links (Dresses, Tops, Jeans, Jackets, Skirts). Column 2 'Accessories' with links (Bags, Shoes, Jewelry, Scarves). Column 3 'By Occasion' with links (Work, Weekend, Party, Holiday). Column 4: a featured image panel with a product photo, 'New Arrivals' label in a badge, and a 'Shop Now' button. Animate the panel: opacity 0 to 1, translateY(-8px) to 0, over 0.2s. Close when mouse leaves both the nav link and the panel."

EXAMPLE WEBSITES TO STUDY

→ **ASOS.com —** One of the best mega menus on the web — clean, fast, multi-column with imagery *(asos.com)*

→ **Apple.com navigation —** Refined mega menu for a product company — minimal but effective *(apple.com)*

→ **Shopify — Polaris Navigation docs —** How Shopify designs and implements mega menus in their design system *(polaris.shopify.com)*

4.4 Dot Navigation

What it is

A small set of dots (usually 4-8) positioned on the right or bottom side of the screen. Each dot represents one section or slide of the website. The dot corresponding to the currently-visible section is filled or larger than the others. Clicking a dot instantly jumps the page to that section. This pattern is almost exclusively used on fullpage scroll websites and slideshow-style presentations where the page snaps from section to section.

How it works

The dots are small circular elements (8-12px diameter) positioned with position:fixed on the right edge of the viewport. They're styled with a semi-transparent border by default and filled solid when active. As the user scrolls, JavaScript detects which section is currently in view (using IntersectionObserver or ScrollTrigger callbacks) and updates which dot has the 'active' class. Clicking a dot triggers a smooth scroll to that section using scrollIntoView() or GSAP's scrollTo plugin.

position:fixed right	Dots are fixed to the viewport so they stay visible as sections change
Active state	Active dot: larger (12px vs 8px), fully filled. Others: outlined or semi-transparent.
IntersectionObserver	Detects which section is currently most visible to update the active dot
Tooltip on hover	Hovering a dot shows the section name — helps users know where they're jumping
scrollIntoView()	JavaScript method to smoothly scroll to a specific element when a dot is clicked

AI PROMPT TO USE

"Add dot navigation to a 6-section fullpage scroll website. Create 6 dots using position:fixed, right:24px, top:50%, transform:translateY(-50%), displayed as a vertical column with 16px gap between dots. Default dot style: 8px width and height, border-radius:50%, border:2px solid rgba(255,255,255,0.6), background:transparent, cursor:pointer, transition all 0.3s. Active dot style: 12px width and height, background:white, border-color:white, box-shadow:0 0 8px rgba(255,255,255,0.5). On hover of any dot, show a tooltip to the left with the section name (absolute positioned, background:rgba(0,0,0,0.7), white text, padding:4px 10px, border-radius:4px). Use IntersectionObserver (threshold:0.5) to update active dot as sections scroll into view."

EXAMPLE WEBSITES TO STUDY

→ **FullPage.js examples —** Every fullPage.js demo uses dot navigation — click the dots to see the behavior *(alvarotrigo.com/fullPage/examples)*

→ **Apple — Special event pages —** Apple's product keynote microsites use dot nav for their fullpage sections *(apple.com/apple-events)*

→ **Dribbble — Dot Navigation UI —** Search 'dot navigation' for visual examples of different dot styles *(dribbble.com/tags/dot-navigation)*

4.5 Breadcrumbs

What it is

Breadcrumbs are a secondary navigation element showing the user's current location within the website's hierarchy. Named after the Hansel and Gretel fairy tale (dropping breadcrumbs to find your way home), they appear as a horizontal trail of links: Home > Category > Subcategory > Current Page. They let users understand exactly where they are and jump back to any parent level with one click. They're essential on e-commerce sites, documentation portals, and any website with more than 2 levels of depth.

How it works

Breadcrumbs are an ordered list (HTML ol element) of links separated by a visual divider — usually a forward slash (/), chevron (›), or angle bracket (>). The current page is the last item and is NOT a link (it's where you already are). All items before it are clickable links. The visual style is intentionally subtle — small font size (13-14px), muted color — so it doesn't compete

with the main page heading. Adding schema.org BreadcrumbList markup helps Google understand and display breadcrumbs in search results.

ol element	Breadcrumbs use an ordered list (ol) for correct HTML semantics
aria-label:breadcrumb	Accessibility attribute that tells screen readers this is a breadcrumb navigation
aria-current:page	Marks the current (last) item for screen readers — no link on this item
schema.org markup	Structured data that makes breadcrumbs appear in Google search results
Separator	The divider between items — usually CSS ::after content: '/' or '›'

AI PROMPT TO USE

"Add a breadcrumb navigation to a product detail page. HTML structure: <nav aria-label='Breadcrumb'> with li items separated by CSS-generated dividers. Style: font-size:13px, color:#888888. Each link (except last): color:#888, text-decoration:none, hover:color:#111 with 0.2s transition. Last item (current page): color:#111, font-weight:500, no link. Separator: CSS li + li::before { content: '›'; margin: 0 8px; color:#CCCCCC }. The breadcrumb path for this page: Home → Men's Clothing → Outerwear → Winter Jackets. Wrap in a container with 16px top and bottom padding. Also add JSON-LD schema.org BreadcrumbList markup in the head for SEO."

EXAMPLE WEBSITES TO STUDY

→ **Amazon.com product pages —** Every product page has breadcrumbs — the most-seen implementation in the world *(amazon.com)*

→ **MDN Web Docs —** Documentation portal with clean breadcrumb navigation showing deep page hierarchies *(developer.mozilla.org)*

→ **Zalando.com —** E-commerce with excellent breadcrumb implementation and schema markup *(zalando.com)*

4.6 Tab Navigation

What it is

Tabs are a horizontal row of clickable labels at the top of a content area, where clicking a tab swaps the content panel below it. Think of a physical filing cabinet with tabbed dividers — click a tab and you see what's in that folder. Tabs are perfect for organizing related content that users need to switch between without leaving the page: product specs vs reviews vs Q&A on a product page, or different dashboard views like Overview, Analytics, and Settings.

How it works

A row of button or anchor elements acts as the tab strip. The currently active tab is indicated visually: an underline border-bottom in the brand color, a filled background, bold text, or a

combination. Clicking a tab adds an 'active' class to that tab and shows the corresponding content panel while hiding all others. Content panels swap with either display:none/block toggling (instant) or an opacity/fade transition (smooth). Each tab and panel pair needs aria-selected, role:tab, aria-controls and role:tabpanel attributes for accessibility.

Active indicator	Underline, filled background, or bold text showing the selected tab
Panel swap	Other panels set to display:none or opacity:0 when their tab is inactive
role:tab / tabpanel	Accessibility attributes so keyboard and screen reader users can navigate
Keyboard navigation	Arrow keys should move between tabs — standard expected behavior
Animated indicator	A sliding underline that moves smoothly between tabs — very polished touch

TIP The sliding animated tab indicator is a great detail that makes tabs feel premium. Achieve it with a separate underline element (position:absolute, bottom:0) that moves via transform:translateX() when tabs are clicked. Calculate the target X position from the clicked tab's offsetLeft.

AI PROMPT TO USE

"Build a product detail page with tab navigation. Tab bar: 4 tabs — Overview, Specifications, Reviews (with count badge), FAQ. Tab strip styling: white background, bottom border 1px solid #E5E7EB, tabs have padding:14px 24px, font-size:15px, font-weight:500, color:#6B7280. Active tab: color:#111, border-bottom:2px solid #2563EB (brand blue), margin-bottom:-1px to overlap container border. Add a sliding animated indicator: a 2px blue line (position:absolute, bottom:-1px, transition:transform 0.25s ease, width matches the active tab) that slides across when tabs are clicked. Content panels: only the active panel is visible (others display:none). Each panel has fade-in: opacity 0 to 1 over 0.2s on activation."

EXAMPLE WEBSITES TO STUDY

→ **GitHub repository pages —** The Code / Issues / Pull Requests / Actions tab bar — used by millions of developers daily *(github.com)*

→ **Radix UI — Tabs component —** Fully accessible, animated tab component with code examples *(radix-ui.com/primitives/docs/components/tabs)*

→ **Amazon product pages —** Product Details / Reviews / Q&A tabs — high-traffic e-commerce tab implementation *(amazon.com)*

5. Typography & Color

Typography and color are the DNA of a brand's visual identity. They communicate personality before a single word is read. A headline in a bold geometric sans-serif communicates technology and confidence. The same headline in an elegant serif communicates luxury and tradition. Color operates the same way — independently of content, color creates emotional associations.

5.1 Type Scale & Hierarchy

What it is

A type scale is a system of predetermined font sizes that work harmoniously together. Instead of choosing random sizes (40px here, 22px there), you pick a scale ratio — like 1.25x or 1.414x — and multiply up and down from a base size. This creates visual harmony because every size is mathematically related to the others. Hierarchy means using size, weight, and color to communicate importance: the biggest, boldest text is the most important.

How it works

Start with a base body size of 16px or 18px. Multiply by your ratio for each level up. At 1.25x scale from 16px: body=16px, H4=20px, H3=25px, H2=31px, H1=39px, display=49px. For landing pages, you can be more aggressive — H1 at 56-80px, H2 at 36-48px. The clamp() CSS function creates fluid typography that smoothly scales between minimum and maximum sizes based on viewport width.

clamp(min, preferred, max)	Font size fluidly between min and max — clamp(40px, 6vw, 80px)
Font weight	100=thin, 400=normal, 600=semibold, 700=bold, 900=black
Letter-spacing	Negative values (-0.02em) on large headings makes them feel tighter and modern
Line-height	1.1-1.2 for headings, 1.6-1.7 for body text — critical for readability

AI PROMPT TO USE

"Define a complete typography system using CSS custom properties. --font-display: clamp(56px, 8vw, 96px), font-weight:800, letter-spacing:-0.04em, line-height:1.05. --font-h1: clamp(40px, 5vw, 64px), font-weight:700, letter-spacing:-0.02em. --font-h2: clamp(28px, 3.5vw, 44px), font-weight:600. --font-h3: 24px, font-weight:600. --font-body: 17px, font-weight:400, line-height:1.7. --font-small: 14px, font-weight:400, color: muted gray. Apply this system consistently throughout the page so the visual hierarchy is immediately clear."

EXAMPLE WEBSITES TO STUDY

→ **Typescale.com —** Interactive tool — choose a scale ratio and see all the sizes generated. Essential resource. *(typescale.com)*

→ **Fontjoy.com —** AI-powered font pairing tool — generates complementary font combinations *(fontjoy.com)*
→ **Typewolf.com —** Daily typography inspiration from real websites, with font identification *(typewolf.com)*

5.2 Font Pairing

What it is

Using two complementary fonts together — one for headings and display text, one for body text and UI. The heading font should have strong personality and visual impact. The body font should be highly readable at small sizes and in paragraphs. The two fonts should contrast but not clash. Using more than two fonts on a website almost always looks amateur.

Recommended pairings for different vibes

Editorial / Luxury	Playfair Display (headings) + DM Sans (body) — classic and refined
Modern Startup	Clash Display (headings) + Satoshi (body) — bold and contemporary
Tech / SaaS	Syne (headings) + Inter (body) — clean and functional
Experimental	Monument Extended (headings) + Neue Haas Grotesk (body) — striking
Friendly / Consumer	Nunito or Poppins (headings) + Source Sans Pro (body) — approachable

AI PROMPT TO USE

"Use these font pairings for this website: import Clash Display from Fontshare CDN (https://api.fontshare.com/v2/css?f[]=clash-display@700&display=swap) for all headings at font-weight:700. Import Satoshi from Fontshare (https://api.fontshare.com/v2/css?f[]=satoshi@400,500&display=swap) for body text at font-weight:400 for paragraphs and 500 for UI labels. Apply Clash Display to all h1, h2, h3 elements with letter-spacing:-0.03em. Apply Satoshi to body, p, button, input with line-height:1.6."

EXAMPLE WEBSITES TO STUDY

→ **Fontshare.com —** Free high-quality fonts specifically curated for web design — much better than Google Fonts *(fontshare.com)*
→ **Google Fonts — Pairings —** Each font page suggests popular pairings. Free and reliable. *(fonts.google.com)*
→ **Fonts In Use —** Real-world examples of font usage across websites, posters, and branding *(fontsinuse.com)*

5.3 Color Theory: The 60-30-10 Rule

What it is

The 60-30-10 rule is a simple color distribution formula borrowed from interior design. 60% of the visual area uses the dominant color (usually a neutral — white, off-white, light gray, or near-black). 30% uses the secondary color (slightly different tone for cards, sections, or backgrounds). 10% uses the accent color (your brand color — for CTAs, highlights, icons, and important UI elements).

How it works

The dominant 60% sets the overall mood. A near-white dominant color feels light and modern. A near-black dominant color feels premium and serious. The secondary 30% creates depth and section differentiation. The accent 10% draws the eye to what matters — your call-to-action button, important statistics, or links. If your accent appears more than 10% of the time, it loses its ability to draw attention.

Dominant (60%)	Your background color — neutral, sets the overall mood
Secondary (30%)	Card backgrounds, alternate sections, sidebar backgrounds
Accent (10%)	Brand color — ONLY for CTAs, highlights, active states, key icons
Tints & shades	Lighter/darker versions of one color — creates depth without adding new colors

AI PROMPT TO USE

"Design a website using the 60-30-10 color rule. Dominant color (60%): #FAFAF7 (warm off-white) for page backgrounds. Secondary color (30%): #F0EDE8 (slightly warmer gray) for cards, alternate sections, and footer. Accent color (10%): #2563EB (electric blue) ONLY for primary CTA buttons, active navigation links, highlighted text, and icons. Text: #111111 for headings, #555555 for body text, #888888 for captions. Never use the electric blue for anything decorative — only for actionable and important elements."

EXAMPLE WEBSITES TO STUDY

→ **Coolors.co —** Color palette generator — generate and lock colors until you find the perfect palette *(coolors.co)*

→ **Paletton.com —** Color theory-based palette builder showing complementary, triadic, and analogous colors *(paletton.com)*

→ **Huemint.com —** AI-powered brand color palette generator — generates full palettes instantly *(huemint.com)*

5.4 Gradient Text

What it is

Instead of a solid color for a heading or key word, the text itself is rendered as a gradient — flowing from one color to another. Extremely popular in AI/tech/SaaS websites because it adds visual interest to plain text without needing additional design elements. Often just one or two words in a headline get the gradient treatment, while the rest of the heading stays solid.

How it works

This is a CSS trick combining three properties. First, set background: linear-gradient(...) on the text element. Second, background-clip: text clips the gradient to the shape of the text characters. Third, -webkit-text-fill-color: transparent (or color: transparent) makes the actual text color invisible, revealing the gradient background showing through the text shape. Without all three, it doesn't work.

AI PROMPT TO USE

"Apply a gradient color effect to key words in the hero headline. The headline reads: 'Build websites that actually convert.' Make the word 'convert' a gradient from #667eea (purple-blue) to #f64f59 (coral-red). CSS: background: linear-gradient(135deg, #667eea 0%, #f64f59 100%); background-clip: text; -webkit-background-clip: text; -webkit-text-fill-color: transparent; color: transparent. Wrap that specific word in a span with this class. The rest of the headline should be white (on dark background). Add a second gradient word in the subheading using a different gradient (green to cyan: #11998e to #38ef7d)."

EXAMPLE WEBSITES TO STUDY

→ **Stripe.com —** Masterful use of subtle gradient text in headings — restrained and effective *(stripe.com)*

→ **Vercel.com —** Bold gradient text in the hero headline — electric and attention-grabbing *(vercel.com)*

→ **Gradient text CSS generator (CSS Gradient) —** Interactive generator for gradient text CSS — copy and paste ready *(cssgradient.io)*

5.5 Display / Hero Typography

What it is

Display typography means using text itself as the primary visual element of a design — the headline is so large, so bold, and so beautifully set that no hero image is needed. At 80px, 100px, or even larger, a single headline becomes a visual experience. This approach is popular on portfolio sites, creative agencies, editorial brands, and tech products that want to lead with confidence and personality rather than a generic stock photo.

How it works

The key CSS property is font-size set very large — typically using clamp(60px, 9vw, 120px) so it scales smoothly from mobile to desktop. Font-weight is usually 700 (bold) or 900 (black/extra-bold) for maximum visual weight. Letter-spacing is tightened with a negative value (-0.03em to -0.05em) which makes large display text feel more professional and modern —

loose letter-spacing on large text looks cheap. Line-height is reduced (0.9 to 1.1) because display text wraps to fewer lines and tight line height creates a compact, powerful block.

clamp(min, fluid, max)	clamp(60px, 9vw, 120px) — scales the font between min and max based on viewport
Negative letter-spacing	-0.03em to -0.05em on large headings — makes it feel tight and contemporary
Line-height < 1	0.9 to 1.05 for display text — tighter than body text, creates a compact visual mass
Font-weight 900	Black or Extra Bold weight — maximum visual impact at large sizes
Text wrapping	Let display headlines wrap naturally to 2-3 lines — don't force one line

TIP One of the most powerful display typography techniques: set one key word in a different color, gradient, or italic/outlined style. This creates a visual anchor and rhythm in the headline. Example: 'Design with' (regular white) 'purpose.' (gradient color). The contrast word should be the most important concept in your headline.

AI PROMPT TO USE

"Create a typographic hero section where the headline IS the design — no hero image needed. Background: #0A0A0A (near-black). Main headline: font-size clamp(64px, 9vw, 112px), font-weight:800, letter-spacing:-0.04em, line-height:0.95, color:#F1F1EE. The headline reads: 'We craft digital experiences that matter.' Make 'digital experiences' render in italic style AND a gradient text (from #667eea to #f64f59). Below the headline: a 1px white horizontal rule at 20% opacity, then a two-column layout — left: subtext paragraph (18px, muted #888, max-width:400px), right: two CTA buttons stacked vertically. No image anywhere on this hero. Let the typography breathe with 120px top and bottom padding."

EXAMPLE WEBSITES TO STUDY

→ **Awwwards — Typography category —** Award-winning sites where display typography is the main design element *(awwwards.com/websites/typography)*

→ **Obys Agency —** Ukrainian agency — incredible display typography used as the primary visual system *(obys.agency)*

→ **Semplice portfolio builder —** Portfolio sites built on Semplice frequently use display typography as hero *(semplice.com/inspiration)*

5.6 Dark / Light Mode Toggle

What it is

A toggle that lets users switch between two complete visual themes — a light version (white/off-white background, dark text) and a dark version (near-black background, light text). Many users prefer dark mode for night-time reading, reduced eye strain, or aesthetic preference. Offering both modes shows attention to user needs and is now considered a

standard feature on quality websites. The browser/OS also signals the user's preferred mode through a CSS media query, which your site can respect automatically.

How it works

The entire color system is built using CSS custom properties (variables) defined on the :root element. Two sets of values are defined — one for light mode, one for dark mode. Switching modes means updating the data-theme attribute on the HTML element (like <html data-theme='dark'>), which triggers a different set of CSS variable values. JavaScript saves the user's preference to localStorage so it persists between visits. The prefers-color-scheme media query lets you set the default based on the user's OS setting.

CSS custom properties	Variables like --color-bg, --color-text that change value when theme switches
data-theme attribute	<html data-theme='dark'> — the selector that triggers dark mode CSS values
prefers-color-scheme	CSS media query detecting if the user's OS is set to dark or light mode
localStorage	Browser storage that saves the user's theme choice across sessions
Transition	transition: background 0.3s, color 0.3s — smooth fade between modes, not instant jump

AI PROMPT TO USE

"Implement a complete dark/light mode system. In CSS, define on :root: --bg:#FFFFFF, --bg-secondary:#F5F5F5, --text:#111111, --text-muted:#666666, --border:#E5E7EB, --accent:#2563EB. Define [data-theme='dark']: --bg:#0D0D0D, --bg-secondary:#1A1A1A, --text:#F1F1EE, --text-muted:#888888, --border:#2A2A2A, --accent:#3B82F6. Apply to all elements: background:var(--bg), color:var(--text). Add body { transition: background 0.3s ease, color 0.3s ease }. Toggle button: on click, toggle data-theme='dark' on the html element and save to localStorage. On page load: check localStorage first, then check prefers-color-scheme media query as fallback. Toggle button shows sun icon in dark mode and moon icon in light mode."

EXAMPLE WEBSITES TO STUDY

→ **Linear.app —** Excellent dark/light mode toggle — notice the smooth transition and how every color adapts *(linear.app)*

→ **Josh W Comeau — Dark Mode Guide —** The most comprehensive guide to building dark mode without the dreaded flash of wrong theme *(joshwcomeau.com/react/dark-mode)*

→ **Rauno Freiberg's portfolio —** Designer at Linear — his portfolio has a beautifully implemented theme toggle *(rauno.me)*

6. UI Patterns

UI patterns are reusable solutions to common design problems. They've been tested by millions of users across thousands of websites, so users already know how they work. A card, a modal, a pricing table, an accordion — these patterns have established behaviors that users expect. Learning these patterns means you can describe exactly what component you need when prompting an AI builder.

6.1 Hero Section

What it is

The hero is the first thing a visitor sees when they land on your website — before any scrolling. It's your 3-second pitch. A great hero immediately communicates: what this is, who it's for, and why they should care. It typically contains a main headline (the biggest, most prominent text on the page), a supporting subheadline, and one clear call-to-action button.

How it works

Hero sections are typically full-width and either full-height (100vh) or tall enough to fill most of the screen. Content is either centered or left-aligned. The visual hierarchy is strict: headline dominates, subtext supports, CTA stands out. A visual element (product screenshot, illustration, abstract graphic) supports the message without competing with the text.

Above the fold	Everything visible before scrolling — this is your hero section's domain
Value proposition	The one sentence that explains what your product does and for whom
Primary CTA	One main action button — 'Get Started', 'Try Free', 'Book Demo'
Social proof	Small indicator of trust near the CTA: '2,000+ companies trust us' or star rating

AI PROMPT TO USE

"Build a hero section for a productivity SaaS. Layout: centered, full-width, dark background (#050505). Badge at top: small pill shape with text 'Now with AI • What's new →'. H1 headline at 72px bold: 'Do your best work,' — next line: 'in half the time.' with 'half the time' in gradient text. Subtext at 20px muted gray: 'The AI-powered workspace that thinks as fast as you do.' Two buttons: primary ('Start for free', blue filled) and secondary ('See how it works →', text only). Below buttons: small trust row '★★★★★ Loved by 10,000+ teams'. Below that: a product screenshot in a browser chrome mockup with a subtle glow effect."

EXAMPLE WEBSITES TO STUDY

→ **Landingfolio.com — Heroes —** 500+ hero section designs filtered by style — the best reference library *(landingfolio.com)*

→ **Saaspo.com —** SaaS landing page inspiration gallery — all hero sections catalogued *(saaspo.com)*

→ **Tailwind UI — Hero Sections —** Pre-built hero section code in multiple styles — study the structure *(tailwindui.com/components/marketing/sections/heroes)*

6.2 Pricing Table

What it is

A structured comparison of your product's different pricing tiers. Almost always 3 columns: a free/starter plan, a paid pro plan (the one you want most people to buy), and an enterprise plan. The pro plan is visually highlighted — larger, with a colored border, an elevated position, and a 'Most Popular' or 'Recommended' badge — to guide users toward it.

How it works

Three-column layout. Each column is a card containing: plan name, price (large and prominent), billing period, 1-line description, a checklist of included features, and a CTA button. The pro/featured plan card gets special visual treatment: a colored border (2px, not just 0.5px), a badge at the top, and sometimes a slight scale-up (transform:scale(1.05)) or extra vertical padding to make it literally bigger than the others.

AI PROMPT TO USE

"Build a 3-tier pricing table with monthly/annual toggle. Free tier: white card, $0/month, 5 features, outlined button. Pro tier (highlighted): colored border (#2563EB, 2px), transform:scale(1.04), 'Most Popular' badge in blue at top, $29/month (annual) / $39/month (monthly), 12 features, filled CTA button. Enterprise: white card, 'Custom pricing', 6 features, outlined button. All cards have: plan name (bold), price (48px), feature checklist with checkmark icons (green checks, gray crosses for excluded). The monthly/annual toggle animates the prices changing and shows 'Save 25%' badge next to the annual option."

EXAMPLE WEBSITES TO STUDY

→ **Tailwind UI — Pricing Sections —** Multiple pre-built pricing table layouts showing different structures *(tailwindui.com/components/marketing/sections/pricing)*

→ **Pricingpages.xyz —** Gallery of real SaaS pricing page screenshots and designs *(pricingpages.xyz)*

→ **Linear.app/pricing —** One of the cleanest pricing pages — study the layout and copy *(linear.app/pricing)*

6.3 Testimonials / Social Proof

What it is

A section showing quotes from real users of your product. Social proof is one of the most powerful conversion elements on a website — people trust other people's opinions more than they trust the company selling the product. The best testimonials are specific ('increased our

revenue by 40%') rather than generic ('great product, love it'). They should include: the quote, the person's photo, name, job title, and company.

How it works

Can be laid out as: a 3-column grid of cards, a single large featured quote, or an auto-scrolling marquee (infinite horizontal scroll). The marquee version is very popular currently — cards scroll continuously from right to left, often with two rows moving in opposite directions. Clicking stops the scroll. Star ratings above the quote add credibility.

AI PROMPT TO USE

"Create a testimonial section with a dual-row infinite marquee. Two rows of testimonial cards, each row auto-scrolling horizontally at different speeds. Row 1 scrolls left at 30px/s, Row 2 scrolls right at 20px/s. Each card (280px wide, white background, 12px border-radius, subtle shadow): star rating (5 gold stars), quote text in quotes, horizontal divider, avatar circle (40px, initials), name (bold 14px), role and company (12px gray). 8 cards per row duplicated for infinite loop (use CSS animation with keyframes translating X by -50%). Section background: light gray #F5F5F5."

EXAMPLE WEBSITES TO STUDY

→ **Senja.io —** Testimonial collection tool — their own website shows excellent testimonial patterns *(senja.io)*

→ **Testimonial.to —** Another testimonial widget product — great reference for modern testimonial UI *(testimonial.to)*

→ **Stripe.com — Customers page —** How a world-class company does social proof — mix of logos, quotes, and case studies *(stripe.com/customers)*

6.4 Accordion / FAQ

What it is

A list of questions where clicking any question expands it to reveal the answer, then clicking again collapses it. This pattern is perfect for FAQ sections, help documentation, and any situation where you have many pieces of related content that would be overwhelming if all shown at once. It lets users scan the questions quickly and only expand the ones they care about.

How it works

Each item has a visible question row (with a + or chevron icon on the right) and a hidden answer panel below it. The answer panel starts at max-height:0 and overflow:hidden. When the question is clicked, a class is toggled that changes max-height to a large value (or auto) and the panel reveals with a smooth CSS transition. The icon rotates 45 degrees (+ becomes ×) or 180 degrees (chevron flips). HTML also has a native details/summary element that does this without JavaScript.

AI PROMPT TO USE

"Build an FAQ section with 8 questions. Each FAQ item: full-width, white background, separated by a 1px #E5E7EB border. Question row: padding 20px, question text in font-weight:500, and a + icon on the right that rotates 45deg to × when open. Answer panel: starts height:0 overflow:hidden, transitions to auto height in 0.3s ease. Answer text is muted gray (#6B7280), 16px, with 20px padding. Only one question can be open at a time (opening a new one closes the previous). Add a search input at the top of the FAQ that filters questions as the user types."

EXAMPLE WEBSITES TO STUDY

→ **Webflow University — FAQ Templates —** No-code accordion FAQ examples with styling guidance *(university.webflow.com)*

→ **Headless UI (by Tailwind Labs) —** Accessible accordion components with animations — code examples *(headlessui.com)*

→ **Linear.app/pricing FAQ —** Simple, clean FAQ implementation at the bottom of their pricing page *(linear.app/pricing)*

6.5 Modal / Dialog

What it is

A modal (or dialog) is a popup overlay that appears over the page content, requiring the user to interact with it or dismiss it before returning to the main page. It's used for: confirmations ('Are you sure you want to delete this?'), forms (sign up, log in), image lightboxes (viewing a photo larger), video players, and onboarding flows. The background is darkened to direct focus to the modal.

How it works

An overlay div covers the entire screen (position:fixed, inset:0, background:rgba(0,0,0,0.5)). The modal panel sits centered on top of the overlay, usually achieved with display:flex, align-items:center, justify-content:center on the overlay. The modal panel itself is a white card with padding, shadow, and border-radius. It animates in with a scale + opacity transition. Clicking the overlay or pressing Escape closes the modal. Focus should be trapped inside the modal while it's open.

AI PROMPT TO USE

"Build a modal component that opens when a button is clicked. Overlay: position:fixed, inset:0, background:rgba(0,0,0,0.6), backdrop-filter:blur(4px), z-index:1000. Modal panel: background:white, border-radius:16px, padding:32px, max-width:480px, width:90%, box-shadow:0 25px 50px rgba(0,0,0,0.25). Animate in: transform:scale(0.92) opacity:0 → scale(1) opacity:1, 0.25s ease. Close on: X button top-right, clicking overlay, pressing Escape key. Include a sign-up form inside: heading 'Create your account', email input, password input, primary submit button, and a 'Sign in instead' link. Body scroll should lock when modal is open."

EXAMPLE WEBSITES TO STUDY

→ **Radix UI — Dialog —** The most accessible, well-implemented modal component with animation examples *(radix-ui.com/primitives/docs/components/dialog)*
→ **Flowbite — Modal Examples —** Multiple modal styles and variations with copy-paste code *(flowbite.com/docs/components/modal)*
→ **Dribbble — Modal Design —** Visual inspiration for creative modal and dialog designs *(dribbble.com/tags/modal)*

6.6 Card Component

What it is

A card is a self-contained rectangular container that groups related information about a single subject — a blog post, a product, a team member, a project. Cards are the most universal building block of web UI design. They create visual separation between items without requiring hard dividers, and they make complex pages scannable because each card is its own focused unit of content. The pattern comes from physical index cards and playing cards — a bounded, portable, self-explanatory information unit.

How it works

A basic card has: a visual area at the top (image, illustration, or colored block), a content area below with a title, description, optional metadata (date, author, tags, reading time), and sometimes a call-to-action (link, button). Cards are grouped in grids (2, 3, or 4 columns) and often have a hover state — a subtle lift effect (translateY(-4px) with increased shadow) that signals interactivity. The card itself is either fully clickable or has a specific CTA link.

Visual area	Top section of card: image, video thumbnail, illustration, or solid color block
Content area	Padded section below the visual: title, description, meta info, CTA
Hover lift	transform:translateY(-4px) + increased box-shadow on hover — signals the card is clickable
overflow:hidden	Applied to the card container — clips the image to the card's border-radius
aspect-ratio	e.g. aspect-ratio:16/9 on the image area keeps cards consistent height regardless of image

TIP Three things make or break a card grid: consistent image aspect ratios (use aspect-ratio:16/9 or 4/3 on the image container, not the image itself), comfortable internal padding (20-24px is standard), and a clear hover state. Without a hover state, cards feel static and users don't know they're clickable. A subtle lift + border highlight is usually enough.

AI PROMPT TO USE

"Build a blog post card grid with 3 columns. Each card: white background (#FFFFFF), border:1px solid #E5E7EB, border-radius:12px, overflow:hidden, cursor:pointer. Hover state:

transform:translateY(-4px), box-shadow:0 12px 30px rgba(0,0,0,0.1), border-color:#D0D0D0. All transitions: 0.25s ease. Card structure: image container (aspect-ratio:16/9, background:#F0F0F0 as placeholder) → content area with padding:20px containing: category tag (small pill, colored background, 11px uppercase text), H3 title (18px, font-weight:600, 2-line clamp with -webkit-line-clamp:2), excerpt paragraph (14px, muted gray, 3-line clamp), bottom meta row (author avatar 28px circle + name + date + reading time, all 13px muted). Entire card is wrapped in an <a> tag. On mobile (under 768px): 1 column. Tablet (768-1024px): 2 columns."

EXAMPLE WEBSITES TO STUDY

→ **Dribbble.com —** Dribbble's own project cards are a reference — hover effect, image, title, author meta *(dribbble.com)*

→ **Tailwind UI — Card Components —** Pre-built card components in multiple styles — study the markup and padding choices *(tailwindui.com/components/marketing/elements/cards)*

→ **Read.cv —** Portfolio platform with beautifully crafted card-based profile and post layouts *(read.cv)*

Quick Reference: Prompting Vocabulary

When prompting any AI tool (Stitch, Framer AI, v0.dev, Bolt, or similar), use this vocabulary to describe what you want with precision:

For aesthetics	glassmorphism / neumorphism / brutalism / minimalism / dark luxury / bento grid / claymorphism / Y2K
For feel	premium / editorial / playful / corporate / warm / clean / bold / refined / futuristic
For layout	hero + feature grid / bento / masonry / fullpage scroll / sidebar+content / split asymmetric
For animation	scroll-triggered fade-up / parallax / sticky scroll / text reveal / horizontal scroll / smooth lenis
For navigation	sticky frosted navbar / hamburger fullscreen / mega menu / dot navigation / tab navigation
For components	pricing table with toggle / testimonial marquee / FAQ accordion / sticky scroll feature / hero section / card grid
For typography	display headline / type scale / gradient text / font pairing / tight letter-spacing
For color	60-30-10 rule / near-black dark mode / single accent color / gradient background / monochromatic

TIP The perfect AI prompt combines: 1 aesthetic + 1 layout + 1 animation + specific colors and fonts. Example: 'Build a glassmorphism (aesthetic) hero section with a centered layout (layout) where feature cards fade up on scroll (animation), using a purple-to-blue gradient background, white frosted glass cards, and Clash Display font for the headline.'

— End of Guide —