FRONTEND BOOTSTRAP LIBRARY

Elementor Pro Widget Alternative Snippets

A Developer's Reference Guide to Building Web Builder Elements from Scratch

PREPARED FOR:

Component Engineering Library

DATE OF ISSUE:

August 2026

REFERENCE MANUAL OVERVIEW

This reference manual compiles all **85 Elementor Pro widgets**, categorizing them systematically and providing clean, modular **React, JavaScript, and Tailwind CSS code snippets** for each. Frontend engineers can use this comprehensive guide as a checklist and code library to build customizable page builder components from the ground up, keeping visual consistency and high-performance execution in modern web stacks.

TABLE OF CONTENTS

LOOP & LISTING ELEMENTS

1. Loop Grid

STYLE SPEC: GRID · STACK: React + Tailwind CSS

Renders a customized grid layout by repeating a design template for each post or dynamic item.

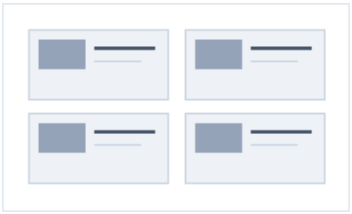
```
const LoopGrid = ({ items, renderCard }) => (
<div className="grid grid-cols-1 sm:grid-cols-2 md:grid-cols-3 gap-6">
{items.map((item, idx) => (
<div key={idx} className="border rounded-lg shadow-sm hover:shadow-md transition-shadow">
{renderCard(item)}
</div>
))}
</div>
);
```

2. Loop Carousel

STYLE SPEC: CAROUSEL · STACK: React + Tailwind CSS

A slider that repeats a custom template for each post, looping infinitely with swipe/click transitions.

```
const LoopCarousel = ({ items, renderCard }) => {
const [index, setIndex] = useState(0);
const next = () => setIndex((index + 1) % items.length);
const prev = () => setIndex((index - 1 + items.length) % items.length);
return (
<div className="relative flex items-center justify-between w-full">
<button onClick={prev} className="p-2 bg-white rounded-full shadow">←</button>
<div className="overflow-hidden w-full mx-4">
<div className="flex transition-transform duration-300" style={{ transform:
`translateX(-${index * 100}%)` }}>
{items.map((item, idx) => <div key={idx} className="min-w-full
p-2">{renderCard(item)}</div>)}
</div>
</div>
<button onClick={next} className="p-2 bg-white rounded-full shadow">→</button>
</div>
);
};
```

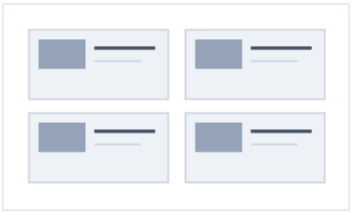


3. Posts

STYLE SPEC: GRID · STACK: React + Tailwind CSS

Displays a grid, list, or masonry layout of standard blog posts, with customizable layouts (Classic, Cards, etc.)

```
const PostsGrid = ({ posts }) => (
<div className="grid grid-cols-1 md:grid-cols-3 gap-6">
{posts.map(post => (
<article key={post.id} className="overflow-hidden rounded-lg border border-gray-100
shadow-sm">
<img src={post.image} alt={post.title} className="h-56 w-full object-cover" />
<div className="p-4 bg-white">
<h3 className="text-lg font-bold text-gray-900">{post.title}</h3>
<p className="mt-2 line-clamp-3 text-sm text-gray-500">{post.excerpt}</p>
</div>
</article>
))}
</div>
);
```

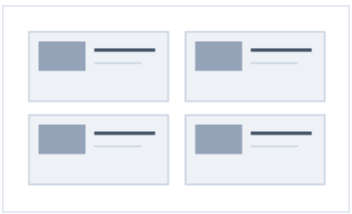


4. Portfolio

STYLE SPEC: GRID · STACK: React + Tailwind CSS

Displays portfolio items in a grid layout with taxonomy filters and smooth hover overlay animations.

```
const PortfolioGrid = ({ items }) => (
<div className="grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-3 gap-4">
{items.map(item => (
<div key={item.id} className="relative group overflow-hidden rounded-lg">
<img src={item.image} className="w-full h-64 object-cover transition-transform
group-hover:scale-105" />
<div className="absolute inset-0 bg-indigo-900/80 opacity-0 group-hover:opacity-100 flex
flex-col justify-center items-center transition-opacity">
<h3 className="text-white font-bold text-lg">{item.title}</h3>
<p className="text-indigo-200 text-sm">{item.category}</p>
</div>
</div>
))}
</div>
);
```



5. Gallery

STYLE SPEC: GRID · STACK: React + Tailwind CSS

A multi-layout image gallery with masonry settings, categories, filtering, and lightbox integration.

```
const GalleryGrid = ({ images }) => (
<div className="columns-1 sm:columns-2 md:columns-3 lg:columns-4 gap-4 space-y-4">
{images.map(img => (
<div key={img.id} className="break-inside-avoid overflow-hidden rounded-lg shadow-sm">
<img src={img.url} className="w-full object-cover hover:opacity-90 cursor-pointer" />
</div>
))}
</div>
);
```

6. Slides

STYLE SPEC: CAROUSEL · STACK: React + Tailwind CSS

A high-impact responsive slideshow banner with text overlays, CTA buttons, and custom background controls.

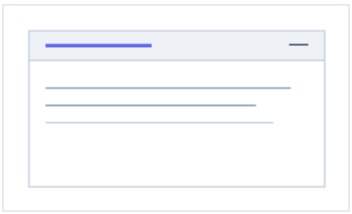
```
const HeroSlides = ({ slides }) => {
const [active, setActive] = useState(0);
return (
<div className="relative w-full h-[500px] overflow-hidden">
{slides.map((slide, idx) => (
<div key={idx} className={`absolute inset-0 transition-opacity duration-1000 flex
items-center justify-center ${active === idx ? 'opacity-100' : 'opacity-0'}`} style={{
backgroundImage: `url(${slide.image})`, backgroundSize: 'cover' }}>
<div className="absolute inset-0 bg-black/45"></div>
<div className="relative text-center text-white p-6">
<h2 className="text-4xl font-black mb-4">{slide.title}</h2>
<button className="bg-indigo-600 px-6 py-2 rounded-full">{slide.cta}</button>
</div>
</div>
))}
</div>
);
};
```

7. Carousel

STYLE SPEC: CAROUSEL · STACK: React + Tailwind CSS

A generic nested item carousel supporting standard images, cards, or custom widgets with pagination.

```
const SimpleCarousel = ({ children }) => (
<div className="flex overflow-x-auto snap-x snap-mandatory scrollbar-none gap-4">
{children.map((child, idx) => (
<div key={idx} className="snap-center shrink-0 w-80">
{child}
</div>
))}
</div>
);
```

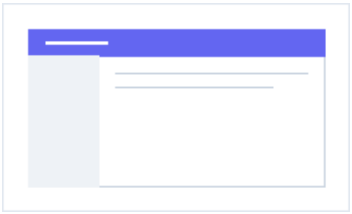


8. Taxonomy Filter

STYLE SPEC: INTERACTIVE · STACK: React + Tailwind CSS

An interactive button group that lets visitors filter posts or custom items dynamically by tag or category.

```
const TaxonomyFilter = ({ categories, activeCat, onFilter }) => (
<div className="flex flex-wrap gap-2 mb-6">
{categories.map(cat => (
<button key={cat} onClick={() => onFilter(cat)} className={`px-4 py-1.5 rounded-full text-sm
font-semibold transition-all ${activeCat === cat ? 'bg-indigo-600 text-white shadow-sm' :
'bg-gray-100 text-gray-600 hover:bg-gray-200'}`}>
{cat}
</button>
))}
</div>
);
```

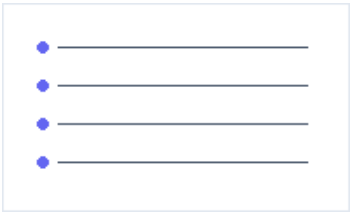


9. Template Widget

STYLE SPEC: THEME · STACK: React + Tailwind CSS

Enables the reuse of global saved templates or sections anywhere inside your pages.

```
const TemplateWidget = ({ renderTemplate, templateId }) => (
<div className="template-container border border-dashed border-indigo-200 rounded-md p-4
bg-indigo-50/10">
{renderTemplate(templateId)}
</div>
);
```



10. Sitemap

STYLE SPEC: LIST · STACK: React + Tailwind CSS

Generates a structured hierarchical list of pages, categories, and custom taxonomy links for navigation and SEO.

```
const SitemapList = ({ nodes }) => (
<ul className="space-y-2 text-gray-700">
{nodes.map(node => (
<li key={node.id} className="pl-4 border-l-2 border-gray-200">
<a href={node.url} className="hover:text-indigo-600 font-semibold">{node.title}</a>
{node.children && <SitemapList nodes={node.children} />}
</li>
))}
</ul>
);
```

FORM & INTERACTIVE ELEMENTS

11. Form

STYLE SPEC: FORM · STACK: React + Tailwind CSS

Creates multi-field, multi-step contact, feedback, or custom submission forms with webhooks and actions.

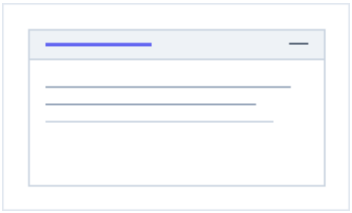
```
const MultiStepForm = () => {
const [step, setStep] = useState(1);
return (
<form className="max-w-md mx-auto p-6 bg-white border rounded-lg shadow-sm">
{step === 1 && (
<div className="space-y-4">
<label className="block text-sm font-medium">Name</label>
<input className="w-full p-2 border rounded-md" type="text" placeholder="John Doe" />
<button type="button" onClick={() => setStep(2)} className="w-full bg-indigo-600 text-white
p-2 rounded">Next</button>
</div>
)}
{step === 2 && (
<div className="space-y-4">
<label className="block text-sm font-medium">Email</label>
<input className="w-full p-2 border rounded-md" type="email" placeholder="john@example.com"
/>
<button type="submit" className="w-full bg-indigo-600 text-white p-2 rounded">Submit</button>
</div>
)}
</form>
);
};
```

12. Login

STYLE SPEC: FORM · STACK: React + Tailwind CSS

A front-end login form with customizable labels, redirect settings, and password recovery triggers.

```
const LoginForm = ({ onLogin }) => (
<form onSubmit={onLogin} className="space-y-4 max-w-sm p-6 bg-white border rounded-lg">
<div>
<label className="block text-sm font-semibold mb-1">Username/Email</label>
<input type="text" className="w-full p-2 border rounded-md focus:ring-2
focus:ring-indigo-500" />
</div>
<div>
<label className="block text-sm font-semibold mb-1">Password</label>
<input type="password" className="w-full p-2 border rounded-md focus:ring-2
focus:ring-indigo-500" />
</div>
<div className="flex justify-between items-center text-xs">
<label><input type="checkbox" className="mr-1" /> Remember Me</label>
<a href="#" className="text-indigo-600 hover:underline">Forgot Password?</a>
</div>
<button type="submit" className="w-full bg-indigo-600 text-white p-2 rounded-md
font-bold">Login</button>
</form>
);
```



13. Off-Canvas

STYLE SPEC: INTERACTIVE · STACK: React + Tailwind CSS

An interactive sliding panel or sidebar triggered by clicks/actions (useful for sliding navigation or carts).

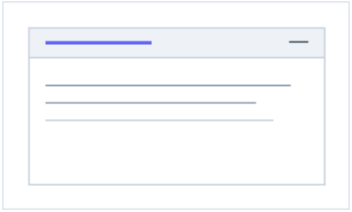
```
const OffCanvas = ({ isOpen, onClose, children }) => (
<div className={`fixed inset-0 z-50 overflow-hidden ${isOpen ? 'pointer-events-auto' :
'pointer-events-none'}`}>
<div className={`absolute inset-0 bg-black/45 transition-opacity duration-300 ${isOpen ?
'opacity-100' : 'opacity-0'}`} onClick={onClose}></div>
<div className={`absolute right-0 top-0 h-full w-80 bg-white p-6 shadow-xl
transition-transform duration-300 ${isOpen ? 'translate-x-0' : 'translate-x-full'}`}>
<button onClick={onClose} className="absolute top-4 right-4 text-xl">×</button>
<div className="mt-8">{children}</div>
</div>
</div>
);
```

14. Animated Headline

STYLE SPEC: HEADLINE · STACK: React + Tailwind CSS

Highlights text dynamically using circling, underlines, cross-outs, or typing effects.

```
const AnimatedHeadline = ({ highlightText, normalText }) => (
<h1 className="text-3xl font-extrabold text-slate-800">
{normalText}{" "}
<span className="relative inline-block">
<span className="absolute left-0 bottom-1 w-full h-3 bg-yellow-300/60 -z-10 transform
-rotate-1 skew-x-3"></span>
{highlightText}
</span>
</h1>
);
```

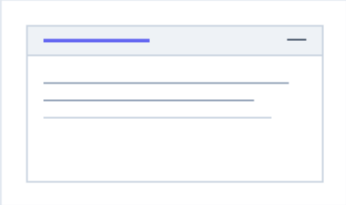


15. Hotspot

STYLE SPEC: INTERACTIVE · STACK: React + Tailwind CSS

Adds interactive pins or markers onto images which display descriptive tooltips on hover or click.

```
const ImageHotspot = ({ x, y, tooltip }) => (
<div className="absolute group" style={{ left: `${x}%`, top: `${y}%` }}>
<div className="w-4 h-4 bg-indigo-600 rounded-full animate-ping absolute inset-0"></div>
<div className="w-4 h-4 bg-indigo-600 rounded-full border border-white cursor-pointer
relative z-10"></div>
<div className="absolute bottom-6 left-1/2 -translate-x-1/2 w-48 p-2 bg-slate-900 text-white
text-xs rounded shadow-lg opacity-0 group-hover:opacity-100 transition-opacity duration-200
pointer-events-none">
{tooltip}
</div>
</div>
);
```

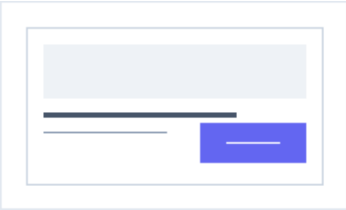


16. Flip Box

STYLE SPEC: INTERACTIVE · STACK: React + Tailwind CSS

An interactive card that flips 180 degrees on hover, revealing call-to-actions or descriptions on the back.

```
const FlipBox = ({ front, back }) => (
<div className="group w-64 h-64 [perspective:1000px]">
<div className="relative w-full h-full transition-transform duration-500
[transform-style:preserve-3d] group-hover:[transform:rotateY(180deg)]">
{/* Front */}
<div className="absolute inset-0 bg-indigo-50 border rounded-lg flex flex-col justify-center
items-center p-4 [backface-visibility:hidden]">
{front}
</div>
{/* Back */}
<div className="absolute inset-0 bg-indigo-600 text-white border rounded-lg flex flex-col
justify-center items-center p-4 [transform:rotateY(180deg)] [backface-visibility:hidden]">
{back}
</div>
</div>
</div>
);
```



17. Call to Action

STYLE SPEC: CTA · STACK: React + Tailwind CSS

High-impact boxes combining images, headings, button links, ribbon tags, and custom hover animations.

```
const CTAWidget = ({ title, desc, image, cta }) => (
<div className="relative group overflow-hidden rounded-xl border border-gray-100 shadow-md
h-80">
<img src={image} className="absolute inset-0 w-full h-full object-cover transition-transform
duration-500 group-hover:scale-105" />
<div className="absolute inset-0 bg-gradient-to-t from-black/85 via-black/45 to-transparent
flex flex-col justify-end p-6">
<h3 className="text-white text-xl font-bold mb-1">{title}</h3>
<p className="text-gray-200 text-sm mb-4">{desc}</p>
<button className="bg-indigo-600 hover:bg-indigo-700 text-white self-start px-4 py-2
rounded-md font-semibold">{cta}</button>
</div>
</div>
);
```

18. Media Carousel

STYLE SPEC: CAROUSEL · STACK: React + Tailwind CSS

A customized media carousel allowing users to slide through images and embed dynamic videos.

```
const MediaCarousel = ({ slides }) => (
<div className="flex gap-4 overflow-x-auto snap-x snap-mandatory scrollbar-thin">
{slides.map((slide, idx) => (
<div key={idx} className="snap-center shrink-0 w-80 h-48 relative rounded-lg
overflow-hidden">
<img src={slide.image} className="absolute inset-0 w-full h-full object-cover" />
{slide.isVideo && (
<div className="absolute inset-0 bg-black/30 flex justify-center items-center">
<button className="w-12 h-12 bg-white/80 rounded-full flex justify-center items-center
text-xl">▶</button>
</div>
)}
</div>
))}
</div>
);
```

19. Testimonial Carousel

STYLE SPEC: CAROUSEL · STACK: React + Tailwind CSS

Slideshow of client reviews, custom ratings, and profile cards with pagination controls.

```
const TestimonialCarousel = ({ testimonials }) => {
const [active, setActive] = useState(0);
return (
<div className="p-8 bg-gray-50 border rounded-xl max-w-lg text-center mx-auto">
<p className="text-lg italic text-slate-700">"{testimonials[active].text}"</p>
<div className="mt-6 flex flex-col items-center">
<img src={testimonials[active].avatar} className="w-12 h-12 rounded-full mb-2" />
<h4 className="font-bold">{testimonials[active].name}</h4>
<span className="text-sm text-gray-400">{testimonials[active].role}</span>
</div>
<div className="flex justify-center gap-2 mt-4">
{testimonials.map((_, i) => (
<button key={i} onClick={() => setActive(i)} className={`w-2.5 h-2.5 rounded-full ${active
=== i ? 'bg-indigo-600' : 'bg-gray-300'}`} />
))}
</div>
</div>
);
};
```

20. Reviews

STYLE SPEC: SOCIAL · STACK: React + Tailwind CSS

Integrates third-party review formatting, custom stars, rating counts, and badges.

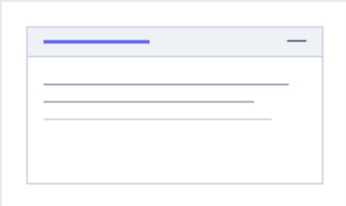
```
const CustomerReviews = ({ reviews }) => (
<div className="space-y-4">
{reviews.map(rev => (
<div key={rev.id} className="p-4 border rounded-lg bg-white">
<div className="flex justify-between mb-2">
<span className="font-bold">{rev.author}</span>
<div className="flex text-yellow-400">{'★'.repeat(rev.rating)}</div>
</div>
<p className="text-sm text-gray-600">{rev.body}</p>
</div>
))}
</div>
);
```

21. Table of Contents

STYLE SPEC: LIST · STACK: React + Tailwind CSS

Automatically parses page structure, listings headings and providing anchor link navigation scroll-to actions.

```
const TableOfContents = ({ selectors }) => {
const [headings, setHeadings] = useState([]);
useEffect(() => {
const els = Array.from(document.querySelectorAll(selectors.join(',')));
setHeadings(els.map(el => ({ text: el.innerText, id: el.id, level: el.tagName })));
}, [selectors]);
return (
<nav className="p-4 bg-gray-50 rounded-lg max-w-sm">
<h3 className="font-bold mb-3 text-slate-800">Table of Contents</h3>
<ul className="space-y-1">
{headings.map((h, i) => (
<li key={i} className={`text-sm ${h.level === 'H3' ? 'pl-4' : ''}`}>
<a href={`#${h.id}`} className="text-indigo-600 hover:underline">{h.text}</a>
</li>
))}
</ul>
</nav>
);
};
```

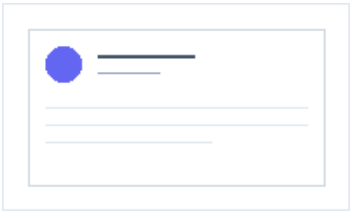


22. Countdown

STYLE SPEC: INTERACTIVE · STACK: React + Tailwind CSS

A time countdown timer widget displaying hours, minutes, and seconds until events or promotions.

```
const Countdown = ({ targetDate }) => {
const [timeLeft, setTimeLeft] = useState({ days: 0, hours: 0, minutes: 0, seconds: 0 });
// Calculate intervals dynamically via useEffect and setInterval
return (
<div className="flex gap-4">
{Object.entries(timeLeft).map(([unit, val]) => (
<div key={unit} className="flex flex-col items-center bg-indigo-50 border p-3 rounded-lg
min-w-[70px]">
<span className="text-2xl font-black text-indigo-700">{val}</span>
<span className="text-xs uppercase text-indigo-400">{unit}</span>
</div>
))}
</div>
);
};
```

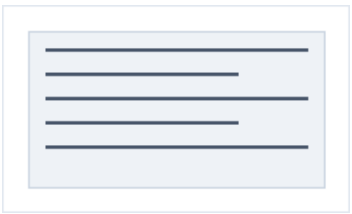


23. Share Buttons

STYLE SPEC: SOCIAL · STACK: React + Tailwind CSS

Custom SVG share icons for major channels with custom layouts, colors, and dynamic page sharing URLs.

```
const ShareButtons = ({ url, title }) => (
<div className="flex gap-2">
<button onClick={() =>
window.open(`https://twitter.com/intent/tweet?url=${url}&text=${title}`)}
className="bg-sky-500 text-white px-3 py-1.5 rounded-md text-sm font-semibold flex
items-center">
Share on X
</button>
<button onClick={() => window.open(`https://facebook.com/sharer/sharer.php?u=${url}`)}
className="bg-blue-600 text-white px-3 py-1.5 rounded-md text-sm font-semibold flex
items-center">
Share on FB
</button>
</div>
);
```

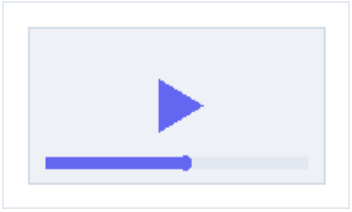


24. Blockquote

STYLE SPEC: TEXT · STACK: React + Tailwind CSS

Stylized blockquote module with left border accents, quote marks, and integrated tweet/share functionality.

```
const Blockquote = ({ quote, author, onTweet }) => (
<blockquote className="border-l-4 border-indigo-600 pl-4 py-2 my-4 italic text-lg
text-slate-700">
<p>"{quote}"</p>
<cite className="block not-italic text-sm font-bold text-slate-500 mt-2">— {author}</cite>
{onTweet && <button onClick={onTweet} className="text-xs text-indigo-500 hover:underline
mt-2">Tweet Quote</button>}
</blockquote>
);
```

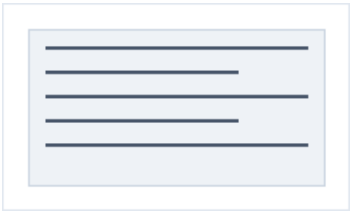


25. Lottie Widget

STYLE SPEC: VIDEO · STACK: React + Tailwind CSS

Enables the display of lightweight, high-performance vector Lottie animations with triggers (hover, scroll).

```
import Lottie from 'lottie-react';
const LottieWrapper = ({ animationData, loop = true }) => (
<div className="w-48 h-48 flex items-center justify-center">
<Lottie animationData={animationData} loop={loop} />
</div>
);
```

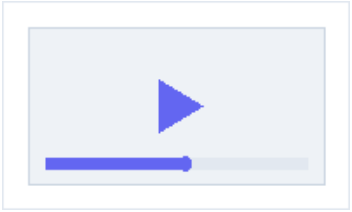


26. Code Highlight

STYLE SPEC: TEXT · STACK: React + Tailwind CSS

A beautifully formatted syntax highlighter block supporting multi-language themes and quick-copy utilities.

```
const CodeHighlight = ({ code, lang }) => {
const [copied, setCopied] = useState(false);
return (
<div className="relative rounded-lg overflow-hidden">
<button onClick={() => { navigator.clipboard.writeText(code); setCopied(true); }}
className="absolute top-2 right-2 bg-gray-800 text-white text-xs px-2 py-1 rounded">
{copied ? 'Copied' : 'Copy'}
</button>
<pre className="bg-slate-900 text-slate-100 p-4 font-mono text-sm overflow-x-auto">
<code>{code}</code>
</pre>
</div>
);
};
```

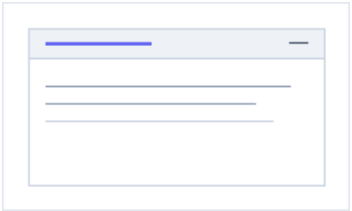


27. Video Playlist

STYLE SPEC: VIDEO · STACK: React + Tailwind CSS

Interactive sidebar playlist separating video playback sections, listing thumbnails, titles, and runtimes.

```
const VideoPlaylist = ({ videos, activeIdx, onSelect }) => (
<div className="border rounded-lg overflow-hidden flex flex-col md:flex-row h-96">
<iframe className="flex-1 bg-black" src={videos[activeIdx].url}></iframe>
<div className="w-full md:w-64 overflow-y-auto bg-gray-50 border-t md:border-t-0
md:border-l">
{videos.map((vid, idx) => (
<div key={vid.id} onClick={() => onSelect(idx)} className={`p-3 cursor-pointer border-b flex
gap-3 ${activeIdx === idx ? 'bg-indigo-50' : 'hover:bg-gray-100'}`}>
<img src={vid.thumb} className="w-16 h-10 object-cover rounded" />
<div className="text-xs">
<p className="font-bold line-clamp-2">{vid.title}</p>
<span className="text-gray-400">{vid.duration}</span>
</div>
</div>
))}
</div>
</div>
);
```

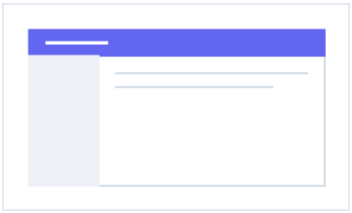


28. Progress Tracker

STYLE SPEC: INTERACTIVE · STACK: React + Tailwind CSS

An interactive indicator displaying the visual scroll progression percentage down the active reading page.

```
const ScrollProgress = () => {
const [width, setWidth] = useState(0);
useEffect(() => {
const onScroll = () => {
const scrolled = window.scrollY;
const height = document.documentElement.scrollHeight - window.innerHeight;
setWidth((scrolled / height) * 100);
};
window.addEventListener('scroll', onScroll);
return () => window.removeEventListener('scroll', onScroll);
}, []);
return <div className="fixed top-0 left-0 h-1.5 bg-indigo-600 transition-all z-50" style={{
width: `${width}%` }}></div>;
};
```

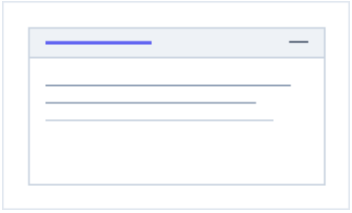


29. Menu

STYLE SPEC: THEME · STACK: React + Tailwind CSS

Responsive mega-menu dropdown structure with hover configurations, layouts, and desktop/mobile adapters.

```
const NavMenu = ({ items }) => {
const [open, setOpen] = useState(null);
return (
<nav className="flex gap-6 relative">
{items.map((item, idx) => (
<div key={idx} className="group" onMouseEnter={() => setOpen(idx)} onMouseLeave={() =>
setOpen(null)}>
<button className="text-slate-700 hover:text-indigo-600 font-medium
py-2">{item.label}</button>
{item.dropdown && open === idx && (
<div className="absolute left-0 mt-2 w-48 bg-white shadow-xl border rounded-md p-2 z-30">
{item.dropdown.map(d => <a key={d.href} href={d.href} className="block p-2 text-sm
hover:bg-gray-100">{d.label}</a>)}
</div>
)}
</div>
))}
</nav>
);
};
```

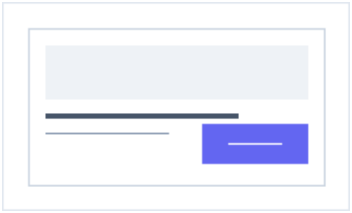


30. Floating Buttons

STYLE SPEC: INTERACTIVE · STACK: React + Tailwind CSS

Floating chat, contact, or trigger buttons aligned along page corners for support or fast communication.

```
const FloatingContact = () => (
<div className="fixed bottom-6 right-6 flex flex-col gap-3 z-40">
<a href="#" className="w-12 h-12 bg-green-500 rounded-full flex justify-center items-center
text-white shadow-lg hover:scale-105 transition-transform">
WA
</a>
<a href="#" className="w-12 h-12 bg-indigo-600 rounded-full flex justify-center items-center
text-white shadow-lg hover:scale-105 transition-transform">
Msg
</a>
</div>
);
```



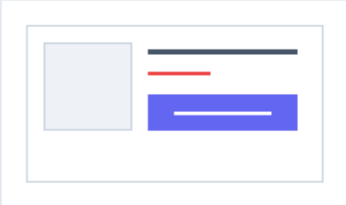
31. Off-Canvas Trigger

STYLE SPEC: BUTTON · STACK: React + Tailwind CSS

A stylized CTA button or trigger specifically built to open dynamic slider popups, side menus, or panels.

```
const OffCanvasTrigger = ({ onOpen, label = "Open Menu" }) => (
<button onClick={onOpen} className="flex items-center gap-2 bg-indigo-600 hover:bg-indigo-700
text-white px-5 py-2.5 rounded-lg shadow-sm font-semibold transition-all">
<span></span>
<span>{label}</span>
</button>
);
```

ECOMMERCE (WOOCOMMERCE) ELEMENTS



32. Product Title

STYLE SPEC: ECOMMERCE · STACK: React + Tailwind CSS

Theme builder tag displaying the current product's title dynamically inside catalog structures.

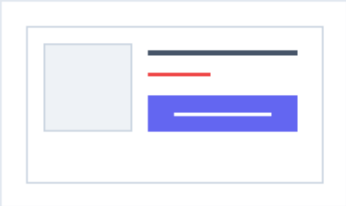
```
const ProductTitle = ({ title }) => (
<h1 className="text-3xl font-black text-slate-900 leading-tight tracking-tight mb-2">
{title}
</h1>
);
```

33. Product Breadcrumbs

STYLE SPEC: LIST · STACK: React + Tailwind CSS

Renders hierarchical categories showing category context paths for dynamic eCommerce pages.

```
const ProductBreadcrumbs = ({ category, title }) => (
<nav className="flex text-xs font-semibold text-gray-500 gap-1.5 uppercase tracking-wide
mb-4">
<a href="/shop" className="hover:text-indigo-600">Shop</a>
<span>/</span>
<a href={`/shop/${category}`} className="hover:text-indigo-600">{category}</a>
<span>/</span>
<span className="text-gray-400 truncate max-w-[120px]">{title}</span>
</nav>
);
```

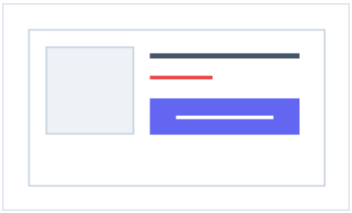


34. Product Images

STYLE SPEC: ECOMMERCE · STACK: React + Tailwind CSS

Multi-image thumbnail slider showing product views with zoom controls, custom borders, and layout skins.

```
const ProductImages = ({ images }) => {
const [main, setMain] = useState(images[0]);
return (
<div className="space-y-4">
<div className="aspect-square w-full border rounded-xl overflow-hidden bg-gray-50">
<img src={main} className="w-full h-full object-cover" />
</div>
<div className="flex gap-3">
{images.map((img, idx) => (
<button key={idx} onClick={() => setMain(img)} className={`w-16 h-16 border rounded-md
overflow-hidden bg-gray-50 ${main === img ? 'ring-2 ring-indigo-500' : ''}`}>
<img src={img} className="w-full h-full object-cover" />
</button>
))}
</div>
</div>
);
};
```

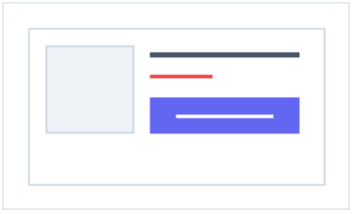


35. Product Price

STYLE SPEC: ECOMMERCE · STACK: React + Tailwind CSS

Pulls dynamic product prices, showing sale markdowns, currency indicators, and custom tags.

```
const ProductPrice = ({ price, salePrice }) => (
<div className="flex items-baseline gap-2.5 text-2xl font-bold mb-4">
{salePrice ? (
<>
<span className="text-red-500 font-extrabold">${salePrice}</span>
<span className="text-gray-400 text-lg line-through font-normal">${price}</span>
</>
) : (
<span className="text-slate-800">${price}</span>
)}
</div>
);
```

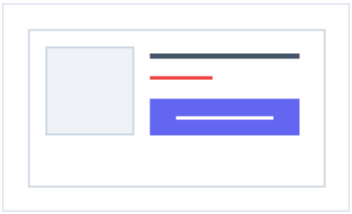


36. Add To Cart

STYLE SPEC: ECOMMERCE · STACK: React + Tailwind CSS

Configures input quantities, variant dropdown choices, stock notifications, and Checkout redirect buttons.

```
const AddToCart = ({ onAdd, initialStock = 10 }) => {
const [qty, setQty] = useState(1);
return (
<div className="flex items-center gap-4 mb-6">
<div className="flex border rounded-lg">
<button onClick={() => setQty(Math.max(1, qty - 1))} className="px-3 py-1.5
border-r">-</button>
<span className="px-4 py-1.5 font-bold flex items-center justify-center
min-w-[40px]">{qty}</span>
<button onClick={() => setQty(Math.min(initialStock, qty + 1))} className="px-3 py-1.5
border-l">+</button>
</div>
<button onClick={() => onAdd(qty)} className="flex-1 bg-indigo-600 hover:bg-indigo-700
text-white font-bold py-2 px-6 rounded-lg shadow-sm transition-all">
Add to Cart
</button>
</div>
);
};
```

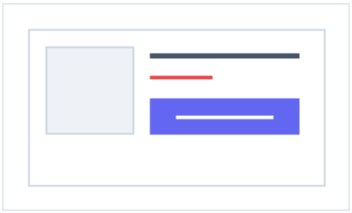


37. Product Rating

STYLE SPEC: ECOMMERCE · STACK: React + Tailwind CSS

Renders average review stars alongside numeric review volume tallies, linking to product review tabs.

```
const ProductRating = ({ rating, count }) => (
<div className="flex items-center gap-2 mb-3 text-sm text-gray-500">
<div className="flex text-yellow-400 text-base">
{Array.from({ length: 5 }).map((_, i) => (
<span key={i}>{i < Math.round(rating) ? '★' : '☆'}</span>
))}
</div>
<span className="font-medium">({count} reviews)</span>
</div>
);
```

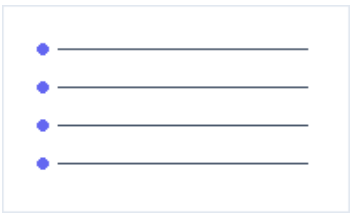


38. Product Stock

STYLE SPEC: ECOMMERCE · STACK: React + Tailwind CSS

Integrates back-end stocks data, showing green positive checkmarks, warnings, or Red Out Of Stock badges.

```
const ProductStock = ({ count }) => (
<div className="flex items-center gap-2 text-sm font-semibold mb-4">
{count > 0 ? (
<>
<span className="w-2 h-2 bg-green-500 rounded-full animate-pulse"></span>
<span className="text-green-600">In Stock ({count} units left)</span>
</>
) : (
<>
<span className="w-2 h-2 bg-red-500 rounded-full"></span>
<span className="text-red-600">Out of Stock</span>
</>
)}
</div>
);
```

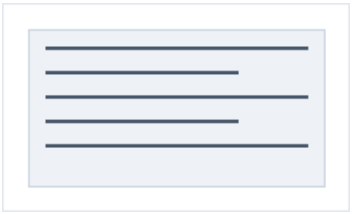


39. Product Meta

STYLE SPEC: LIST · STACK: React + Tailwind CSS

Meta details blocks listing category labels, tags, and SKUs.

```
const ProductMeta = ({ sku, category, tags }) => (
<div className="border-t border-b py-4 my-6 space-y-2 text-sm text-gray-500">
<div><span className="font-bold text-slate-800">SKU:</span> {sku}</div>
<div><span className="font-bold text-slate-800">Category:</span> {category}</div>
<div><span className="font-bold text-slate-800">Tags:</span> {tags.join(', ')}</div>
</div>
);
```

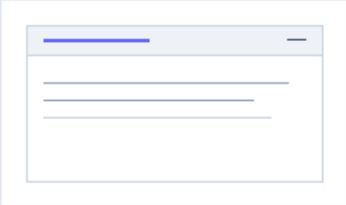


40. Short Description

STYLE SPEC: TEXT · STACK: React + Tailwind CSS

Extracts brief post/product excerpts, formatting them inside visual product listings.

```
const ShortDescription = ({ desc }) => (
<p className="text-slate-600 leading-relaxed text-sm mb-4">
{desc}
</p>
);
```

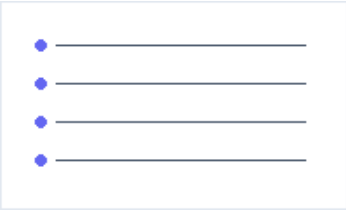


41. Product Data Tabs

STYLE SPEC: INTERACTIVE · STACK: React + Tailwind CSS

Responsive tab layout separating full product descriptions, technical parameters, and review panels.

```
const ProductDataTabs = ({ tabs }) => {
const [active, setActive] = useState(0);
return (
<div className="mt-10">
<div className="flex border-b border-gray-200 gap-8">
{tabs.map((tab, idx) => (
<button key={idx} onClick={() => setActive(idx)} className={`pb-3 font-bold text-sm
border-b-2 transition-all ${active === idx ? 'border-indigo-600 text-indigo-600' :
'border-transparent text-gray-400'}`}>
{tab.label}
</button>
))}
</div>
<div className="py-6 text-sm text-slate-600 leading-relaxed">{tabs[active].content}</div>
</div>
);
};
```



42. Additional Information

STYLE SPEC: LIST · STACK: React + Tailwind CSS

Displays custom dynamic attributes, product dimensions, weight, or technical properties in lists.

```
const AdditionalInfo = ({ attributes }) => (
<div className="divide-y divide-gray-100 text-sm border rounded-lg overflow-hidden">
{attributes.map(attr => (
<div key={attr.label} className="flex p-3 bg-white hover:bg-gray-50">
<span className="w-1/3 font-bold text-slate-800">{attr.label}</span>
<span className="w-2/3 text-slate-600">{attr.value}</span>
</div>
))}
</div>
);
```

43. Product Related

STYLE SPEC: CAROUSEL · STACK: React + Tailwind CSS

Analyzes taxonomies to compile carousel lists of similar products to encourage browsing.

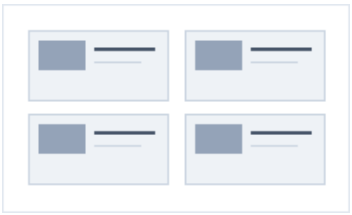
```
const RelatedProducts = ({ products, renderCard }) => (
<div className="mt-12">
<h3 className="text-xl font-bold mb-6 text-slate-800">Related Products</h3>
<div className="grid grid-cols-1 sm:grid-cols-2 md:grid-cols-4 gap-6">
{products.map(prod => renderCard(prod))}
</div>
</div>
);
```

44. Upsells

STYLE SPEC: CAROUSEL · STACK: React + Tailwind CSS

Displays premium upgrades or matching package recommendations directly in product listings.

```
const ProductUpsells = ({ upsellItems, renderCard }) => (
<div className="bg-indigo-50/20 p-6 border rounded-xl mt-8">
<h4 className="font-bold text-indigo-800 mb-4">Complete Your Set</h4>
<div className="flex gap-4 overflow-x-auto scrollbar-thin pb-2">
{upsellItems.map(item => (
<div key={item.id} className="w-56 shrink-0 bg-white rounded-lg p-2
shadow-sm">{renderCard(item)}</div>
))}
</div>
</div>
);
```



45. Products

STYLE SPEC: GRID · STACK: React + Tailwind CSS

A high-power query grid builder rendering full store inventory lists, complete with filter sorting.

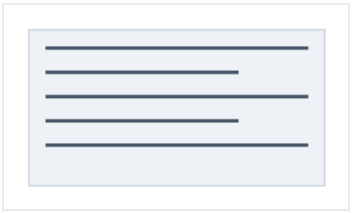
```
const ProductsQueryGrid = ({ products, onSort }) => (
<div>
<div className="flex justify-between items-center mb-6">
<span className="text-sm text-gray-500">Showing {products.length} products</span>
<select onChange={e => onSort(e.target.value)} className="p-2 border rounded-md text-sm">
<option value="latest">Latest</option>
<option value="price-asc">Price: Low to High</option>
</select>
</div>
<div className="grid grid-cols-2 md:grid-cols-4 gap-6">{products.map(p => <ProductCard
key={p.id} {...p} />)}</div>
</div>
);
```

46. Archive Products

STYLE SPEC: GRID · STACK: React + Tailwind CSS

Renders WooCommerce product list structures specifically matching search results or category taxonomies.

```
const ArchiveProducts = ({ archiveItems }) => (
<div className="grid grid-cols-1 sm:grid-cols-3 gap-6">
{archiveItems.length > 0 ? (
archiveItems.map(item => <ProductCard key={item.id} {...item} />)
) : (
<p className="col-span-full text-center text-gray-500 py-10">No products found matching
selection.</p>
)}
</div>
);
```

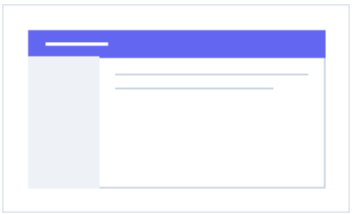


47. Archive Description

STYLE SPEC: TEXT · STACK: React + Tailwind CSS

Extracts descriptions or summary tags compiled specifically for active catalog archive indices.

```
const ArchiveDescription = ({ desc }) => (
<div className="bg-gray-50 border-l-4 border-indigo-500 p-4 mb-6">
<p className="text-slate-600 text-sm leading-relaxed">{desc}</p>
</div>
);
```

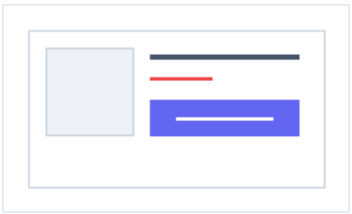


48. Archive Title (WooCommerce)

STYLE SPEC: THEME · STACK: React + Tailwind CSS

Theme builder tag dynamically listing titles matching currently active WooCommerce category archives.

```
const ArchiveTitle = ({ title }) => (
<div className="border-b pb-4 mb-8">
<h1 className="text-4xl font-extrabold text-slate-800 tracking-tight">{title}</h1>
</div>
);
```

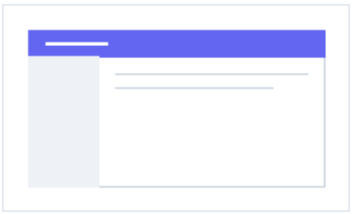


49. Custom Add To Cart

STYLE SPEC: ECOMMERCE · STACK: React + Tailwind CSS

Creates highly styled custom buttons that add specific quantities of specific items on-click.

```
const CustomAddToCart = ({ productId, label = "Buy Now", price }) => (
<button onClick={() => addToCart(productId, 1)} className="flex justify-between items-center
bg-slate-950 text-white font-bold py-3 px-6 rounded-lg shadow-md hover:bg-slate-900
transition-all w-full">
<span>{label}</span>
<span>${price}</span>
</button>
);
```



50. WooCommerce Pages

STYLE SPEC: THEME · STACK: React + Tailwind CSS

Embeds responsive full-container structures for system checkout, cart, or account workflows.

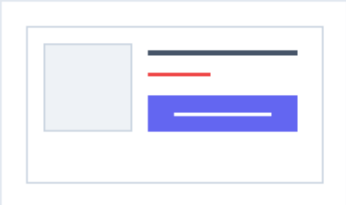
```
const WooPageContainer = ({ children }) => (
<div className="woocommerce-page max-w-7xl mx-auto px-4 py-10 sm:px-6 lg:px-8">
<div className="bg-white rounded-2xl shadow-sm border p-6 md:p-10">{children}</div>
</div>
);
```

51. Product Categories

STYLE SPEC: GRID · STACK: React + Tailwind CSS

Grid or card lists showing inventory categories with image banners and product count tags.

```
const ProductCategoriesGrid = ({ categories }) => (
<div className="grid grid-cols-2 md:grid-cols-3 lg:grid-cols-6 gap-4">
{categories.map(cat => (
<a href={cat.url} key={cat.id} className="group text-center flex flex-col items-center
bg-gray-50 p-4 rounded-xl border hover:border-indigo-500 transition-all">
<img src={cat.icon} className="w-12 h-12 mb-3 object-contain" />
<h4 className="font-bold text-sm text-slate-700 group-hover:text-indigo-600">{cat.name}</h4>
<span className="text-xs text-gray-400 mt-1">{cat.count} Items</span>
</a>
))}
</div>
);
```

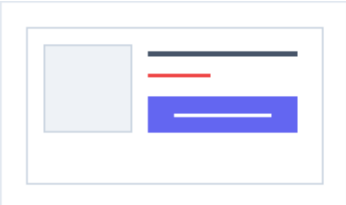


52. Menu Cart

STYLE SPEC: ECOMMERCE · STACK: React + Tailwind CSS

A header menu cart icon summarizing selected items, with hover flyouts for mini cart item views.

```
const MenuCart = ({ cartItems, onOpen }) => (
<button onClick={onOpen} className="relative p-2.5 text-slate-700 hover:text-indigo-600
transition-colors flex items-center">
<span></span>
{cartItems.length > 0 && (
<span className="absolute top-0 right-0 w-5 h-5 bg-indigo-600 text-white rounded-full
text-[10px] font-black flex items-center justify-center border border-white">
{cartItems.reduce((acc, i) => acc + i.qty, 0)}
</span>
)}
</button>
);
```

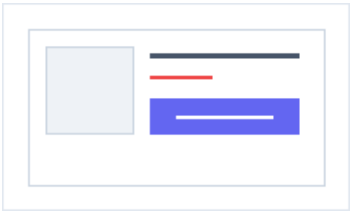


53. Cart

STYLE SPEC: ECOMMERCE · STACK: React + Tailwind CSS

Full page shopping cart interface displaying tabular product list selections, pricing additions, and CTA links.

```
const ShoppingCart = ({ items, onRemove, onUpdateQty }) => (
<div className="space-y-6">
{items.map(item => (
<div key={item.id} className="flex items-center justify-between border-b pb-4">
<img src={item.image} className="w-16 h-16 object-cover rounded" />
<div className="flex-1 ml-4">
<h4 className="font-bold">{item.name}</h4>
<p className="text-sm text-gray-400">${item.price} each</p>
</div>
<input type="number" value={item.qty} onChange={e => onUpdateQty(item.id,
parseInt(e.target.value))} className="w-16 border rounded p-1 mx-4 text-center" />
<button onClick={() => onRemove(item.id)} className="text-red-500
hover:underline">Remove</button>
</div>
))}
</div>
);
```

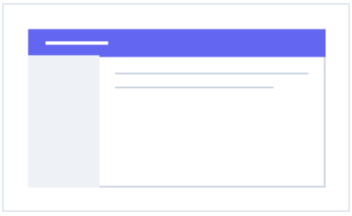


54. Checkout

STYLE SPEC: ECOMMERCE · STACK: React + Tailwind CSS

Handles transactional layouts for billing lists, coupon applications, shipping selections, and pay integrations.

```
const CheckoutLayout = ({ cartTotal, onPay }) => (
<div className="grid grid-cols-1 lg:grid-cols-3 gap-8">
<div className="lg:col-span-2 space-y-6">
<h3 className="text-xl font-bold border-b pb-2">Billing details</h3>
{/* Input Fields */}
</div>
<div className="bg-gray-50 p-6 border rounded-xl h-fit">
<h3 className="text-lg font-bold mb-4">Your order</h3>
<div className="flex justify-between border-b pb-3 mb-4"><span
className="font-semibold">Total</span><span className="font-bold
text-lg">${cartTotal}</span></div>
<button onClick={onPay} className="w-full bg-indigo-600 text-white py-3 rounded-lg font-bold
shadow-md hover:bg-indigo-700">Place Order</button>
</div>
</div>
);
```

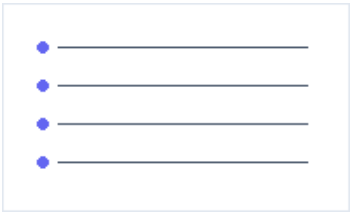


55. My Account

STYLE SPEC: THEME · STACK: React + Tailwind CSS

Multi-tab dashboards routing customers through order logs, address coordinates, or profile updates.

```
const AccountDashboard = () => {
const [tab, setTab] = useState('orders');
return (
<div className="flex gap-8">
<aside className="w-48 border-r pr-6 space-y-2">
{['Dashboard', 'Orders', 'Addresses'].map(t => (
<button key={t} onClick={() => setTab(t.toLowerCase())} className={`block w-full text-left
p-2 rounded text-sm font-bold ${tab === t.toLowerCase() ? 'bg-indigo-50 text-indigo-600' :
'text-gray-500 hover:bg-gray-50'}`}>{t}</button>
))}
</aside>
<main className="flex-1">{/* Render tab screen */}</main>
</div>
);
};
```

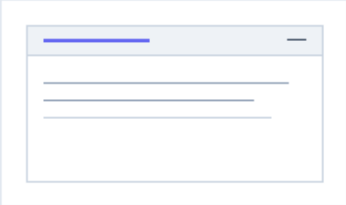


56. Purchase Summary

STYLE SPEC: LIST · STACK: React + Tailwind CSS

Renders transactional confirmation receipts detailing product IDs, billing terms, and shipping routes.

```
const PurchaseReceipt = ({ order }) => (
<div className="max-w-md bg-white border border-dashed rounded-2xl p-6 shadow-sm mx-auto">
<div className="text-center mb-6">
<span className="text-3xl">□</span>
<h2 className="text-xl font-black mt-2">Thank you!</h2>
</div>
<ul className="divide-y text-sm mb-4">
<li className="flex justify-between py-2"><span>Order ID</span><span
className="font-mono">#{order.id}</span></li>
<li className="flex justify-between py-2"><span>Total Paid</span><span
className="font-bold">${order.total}</span></li>
</ul>
<p className="text-xs text-center text-gray-400">A confirmation email has been
dispatched.</p>
</div>
);
```

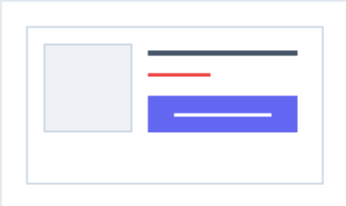


57. WooCommerce Notices

STYLE SPEC: INTERACTIVE · STACK: React + Tailwind CSS

Bespoke styled toast or top-bar warning indicators alert-notifying shoppers of dynamic catalog changes.

```
const WooCommerceNotice = ({ message, type = "success" }) => (
<div className={`p-4 border-l-4 rounded-r-md text-sm mb-4 shadow-sm flex items-center gap-3
${type === 'success' ? 'bg-green-50 border-green-500 text-green-700' : 'bg-red-50
border-red-500 text-red-700'}`}>
<span className="text-base">{type === 'success' ? '✓' : '⚠'}</span>
<span className="font-medium">{message}</span>
</div>
);
```

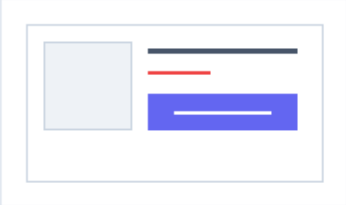


58. PayPal Button

STYLE SPEC: ECOMMERCE · STACK: React + Tailwind CSS

One-click PayPal checkout integration displaying customizable visual payment configurations.

```
const PayPalButton = ({ amount, onSuccess }) => {
// Render PayPal SDK Button script dynamically
return <div className="w-full max-w-[200px] h-10 bg-yellow-400 rounded-md font-semibold flex
items-center justify-center cursor-pointer shadow hover:bg-yellow-300">Pay with PayPal</div>;
};
```



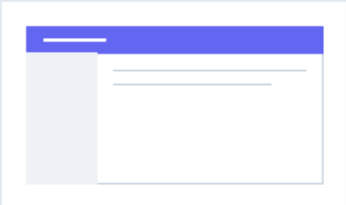
59. Stripe Button

STYLE SPEC: ECOMMERCE · STACK: React + Tailwind CSS

Renders fully secure Stripe card inputs, checking validations and executing payment intents.

```
const StripeButton = ({ price, onCharge }) => (
<button onClick={onCharge} className="w-full max-w-[200px] h-10 bg-[#5433FF]
hover:bg-[#4326dd] text-white font-bold rounded-md shadow flex items-center justify-center
gap-2">
<span></span>
<span>Stripe Quick Pay</span>
</button>
);
```

THEME BUILDER ELEMENTS

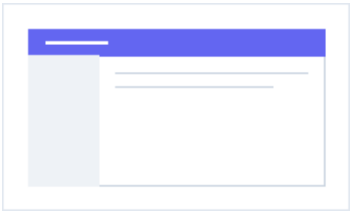


60. Site Logo

STYLE SPEC: THEME · STACK: React + Tailwind CSS

Theme builder widget dynamically extracting the corporate site branding logo image.

```
const SiteLogo = ({ logoUrl, homeUrl = "/" }) => (
<a href={homeUrl} className="inline-block hover:opacity-90 max-h-16 overflow-hidden">
<img src={logoUrl} alt="Site Logo" className="object-contain h-10" />
</a>
);
```

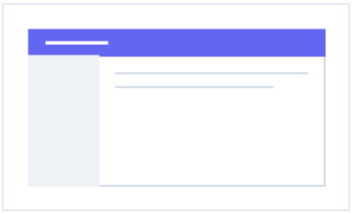


61. Site Title

STYLE SPEC: THEME · STACK: React + Tailwind CSS

Pulls current brand title metadata dynamically from back-end site configuration files.

```
const SiteTitle = ({ brandName, homeUrl = "/" }) => (
<a href={homeUrl} className="text-xl font-extrabold text-slate-800 hover:text-indigo-600
transition-colors uppercase tracking-wider">
{brandName}
</a>
);
```



62. Page Title

STYLE SPEC: THEME · STACK: React + Tailwind CSS

Displays titles matching active site document files, supporting page banners and grids.

```
const PageTitle = ({ title }) => (
<div className="bg-slate-50 py-10 px-6 border-b border-gray-100">
<h1 className="max-w-5xl mx-auto text-3xl font-black text-slate-800
tracking-tight">{title}</h1>
</div>
);
```

63. WordPress Menu

STYLE SPEC: LIST · STACK: React + Tailwind CSS

Compiles standard system navigation items into clean, list-style navigation headers.

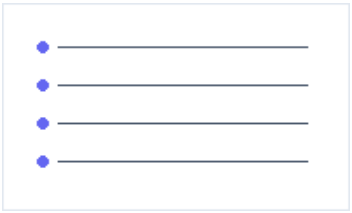
```
const WPMenu = ({ items }) => (
<ul className="flex gap-6 text-sm font-semibold text-slate-600">
{items.map(item => (
<li key={item.id}>
<a href={item.url} className="hover:text-indigo-600 transition-colors">{item.title}</a>
</li>
))}
</ul>
);
```

64. Search Form

STYLE SPEC: FORM · STACK: React + Tailwind CSS

Site-wide full-form search integration supporting input fields, submission buttons, and modal triggers.

```
const SearchForm = ({ onSearch }) => (
<form onSubmit={onSearch} className="flex items-center border rounded-full overflow-hidden
max-w-sm focus-within:ring-2 focus-within:ring-indigo-500 bg-white shadow-sm">
<input type="text" placeholder="Search our blog..." className="flex-1 px-4 py-2 border-none
outline-none text-sm" />
<button type="submit" className="bg-indigo-600 hover:bg-indigo-700 text-white px-5 py-2
text-sm font-bold">Search</button>
</form>
);
```



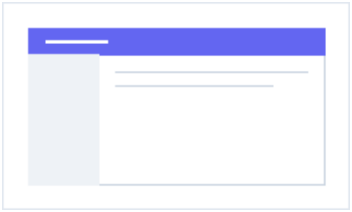
65. Breadcrumbs

STYLE SPEC: LIST · STACK: React + Tailwind CSS

Navigational trail listing directory paths for improved user experience and SEO indexation.

```
const Breadcrumbs = ({ paths }) => (
<nav className="flex text-xs font-semibold text-gray-400 gap-1">
{paths.map((p, idx) => (
<span key={idx} className="flex gap-1">
{idx > 0 && <span>/</span>}
{p.url ? <a href={p.url} className="hover:text-slate-700">{p.label}</a> : <span
className="text-slate-600">{p.label}</span>}
</span>
))}
</nav>
);
```

POST ELEMENTS ELEMENTS

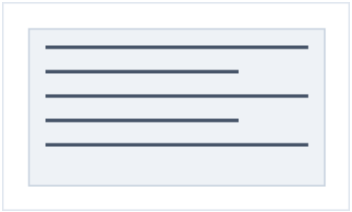


66. Post Title

STYLE SPEC: THEME · STACK: React + Tailwind CSS

Extracts current blog page title structures for standardized blog post layouts.

```
const PostTitle = ({ title }) => (
<h1 className="text-4xl font-extrabold text-slate-900 tracking-tight leading-tight mb-4">
{title}
</h1>
);
```

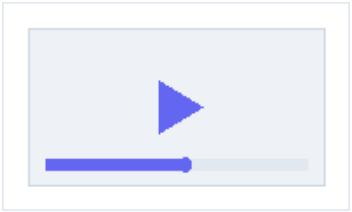


67. Post Excerpt

STYLE SPEC: TEXT · STACK: React + Tailwind CSS

Retrieves post excerpts, displaying meta descriptions or introductory summaries above contents.

```
const PostExcerpt = ({ excerpt }) => (
<p className="text-lg text-slate-500 font-medium leading-relaxed mb-6 border-l-2 pl-4
border-slate-300">
{excerpt}
</p>
);
```

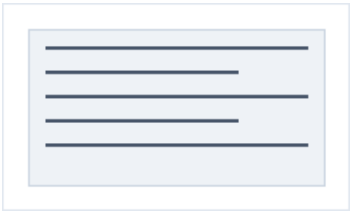


68. Featured Image

STYLE SPEC: MEDIA · STACK: React + Tailwind CSS

Dynamically references post header banner graphics, configuring height dimensions and aspect ratios.

```
const FeaturedImage = ({ src, alt = "Blog post banner" }) => (
<div className="w-full max-h-[450px] rounded-2xl overflow-hidden shadow-sm border bg-gray-50
mb-8">
<img src={src} alt={alt} className="w-full h-full object-cover" />
</div>
);
```



69. Post Content

STYLE SPEC: TEXT · STACK: React + Tailwind CSS

Dynamic layout grid rendering full article rich texts, images, and HTML embeds.

```
const PostContent = ({ htmlContent }) => (
<div className="prose max-w-none text-slate-800 leading-relaxed font-sans"
dangerouslySetInnerHTML={{ __html: htmlContent }} />
);
```

70. Author Box

STYLE SPEC: SOCIAL · STACK: React + Tailwind CSS

Author meta profile card listing avatars, biography summaries, and social media channels.

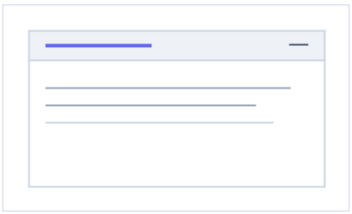
```
const AuthorBox = ({ name, bio, avatar, socials }) => (
<div className="flex gap-4 p-5 bg-gray-50 border rounded-2xl items-center max-w-xl">
<img src={avatar} className="w-16 h-16 rounded-full border shadow-sm" />
<div>
<h4 className="font-bold text-slate-800 text-lg">{name}</h4>
<p className="text-xs text-slate-500 mt-1 leading-relaxed">{bio}</p>
</div>
</div>
);
```

71. Post Comments

STYLE SPEC: FORM · STACK: React + Tailwind CSS

Visualizes site comment threads and customizable feedback text inputs for readers.

```
const PostComments = ({ comments, onComment }) => (
<div className="space-y-6 mt-10">
<h3 className="text-lg font-bold border-b pb-2">Comments ({comments.length})</h3>
<div className="space-y-4">{/* List out comment cards */}</div>
<form onSubmit={onComment} className="space-y-4">
<textarea placeholder="Post a comment..." className="w-full p-3 border rounded-lg h-24" />
<button type="submit" className="bg-indigo-600 text-white px-5 py-2 rounded-lg
font-bold">Submit</button>
</form>
</div>
);
```



72. Post Navigation

STYLE SPEC: INTERACTIVE · STACK: React + Tailwind CSS

Theme builder tag rendering simple Next and Previous post links on reading layouts.

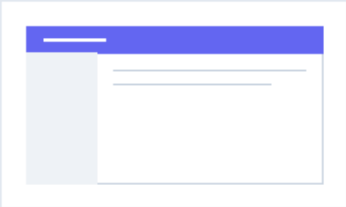
```
const PostNavigation = ({ prevPost, nextPost }) => (
<div className="flex justify-between border-t border-b py-4 my-8 text-sm">
{prevPost ? <a href={prevPost.url} className="hover:text-indigo-600 font-bold">← Previous
Post</a> : <div></div>}
{nextPost ? <a href={nextPost.url} className="hover:text-indigo-600 font-bold">Next Post
→</a> : <div></div>}
</div>
);
```

73. Post Info

STYLE SPEC: LIST · STACK: React + Tailwind CSS

Renders post metadatas showing publish dates, category paths, and reading times.

```
const PostInfo = ({ author, date, readTime }) => (
<div className="flex flex-wrap gap-4 text-xs font-semibold text-slate-400 mb-6 border-b
pb-4">
<div><span>By </span><span className="text-slate-700">{author}</span></div>
<div>•</div>
<div>{date}</div>
<div>•</div>
<div>{readTime} Read</div>
</div>
);
```

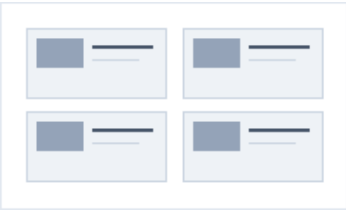


74. Archive Title

STYLE SPEC: THEME · STACK: React + Tailwind CSS

Dynamically outputs descriptive headlines for standard category listing pages.

```
const StandardArchiveTitle = ({ title }) => (
<div className="bg-gradient-to-r from-indigo-900 to-indigo-750 text-white p-8 rounded-2xl
mb-8">
<span className="text-xs font-bold uppercase tracking-wider text-indigo-200">Category
Archive</span>
<h1 className="text-3xl font-black mt-1">{title}</h1>
</div>
);
```



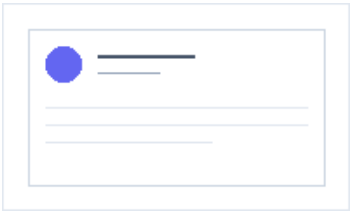
75. Archive Posts

STYLE SPEC: GRID · STACK: React + Tailwind CSS

Compiles standard category post lists, supporting pagination numbers and layout controls.

```
const ArchivePostsGrid = ({ posts }) => (
<div className="grid grid-cols-1 md:grid-cols-2 gap-6">
{posts.map(post => (
<div key={post.id} className="flex gap-4 border p-4 rounded-xl bg-white">
<img src={post.thumb} className="w-24 h-24 object-cover rounded-lg bg-gray-50" />
<div>
<h4 className="font-bold text-base hover:text-indigo-600 cursor-pointer">{post.title}</h4>
<p className="text-xs text-gray-400 mt-2 line-clamp-2">{post.excerpt}</p>
</div>
</div>
))}
</div>
);
```

SOCIAL & EMBEDS ELEMENTS

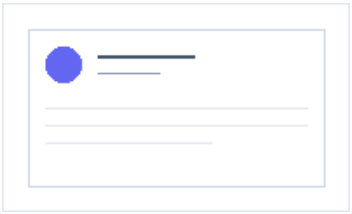


76. Facebook Button

STYLE SPEC: SOCIAL · STACK: React + Tailwind CSS

Embeds official Facebook Share or Like buttons linked directly to custom page URLs.

```
const FacebookButton = ({ url, layout = "standard" }) => {
// Loads SDK dynamically if not loaded, then initiates FB.XFBML.parse()
return <div className="fb-like" data-href={url} data-layout={layout}
data-action="like"></div>;
};
```

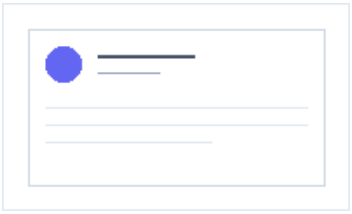


77. Facebook Comments

STYLE SPEC: SOCIAL · STACK: React + Tailwind CSS

Official Facebook comment iframe feeds linking visitor account discussions onto your pages.

```
const FacebookComments = ({ url, numPosts = 5 }) => (
<div className="fb-comments" data-href={url} data-numposts={numPosts}
data-width="100%"></div>
);
```



78. Facebook Embed

STYLE SPEC: SOCIAL · STACK: React + Tailwind CSS

Iframe embedding tool designed to easily embed target Facebook posts, reels, or videos.

```
const FacebookEmbed = ({ url }) => (
<div className="flex justify-center w-full my-4">
<iframe src={`https://www.facebook.com/plugins/post.php?href=${encodeURIComponent(url)}`}
width="500" height="350" scrolling="no" frameBorder="0" className="border-none
overflow-hidden"></iframe>
</div>
);
```

79. Facebook Page

STYLE SPEC: SOCIAL · STACK: React + Tailwind CSS

Official Facebook profile/page feed embedding tool, rendering cover visuals and CTAs.

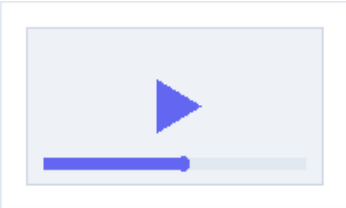
```
const FacebookPageEmbed = ({ pageUrl }) => (
<div className="w-full overflow-hidden max-w-[340px]">
<iframe src={`https://www.facebook.com/plugins/page.php?href=${encodeURIComponent(pageUrl)}&t
abs=timeline`} width="340" height="500" scrolling="no" frameBorder="0" className="border-none
overflow-hidden"></iframe>
</div>
);
```

80. Link in Bio

STYLE SPEC: LIST · STACK: React + Tailwind CSS

Multi-link dashboard widget built for bio listings on social media channels.

```
const LinkInBio = ({ avatar, name, links }) => (
<div className="flex flex-col items-center bg-slate-900 text-white min-h-[500px] p-8 max-w-sm
mx-auto rounded-3xl border">
<img src={avatar} className="w-20 h-20 rounded-full border-2 border-indigo-500 mb-3" />
<h3 className="font-bold text-lg mb-1">{name}</h3>
<div className="w-full space-y-3.5 mt-6">
{links.map(l => (
<a key={l.url} href={l.url} className="block w-full bg-slate-800 hover:bg-slate-750
text-center py-3 rounded-full font-bold text-sm border hover:border-indigo-500
transition-all">
{l.label}
</a>
))}
</div>
</div>
);
```

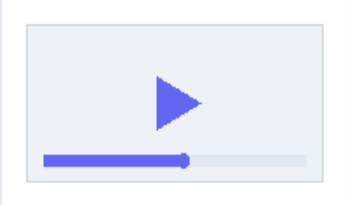


81. Sound Cloud

STYLE SPEC: VIDEO · STACK: React + Tailwind CSS

Audio player widget dynamically embedding SoundCloud songs or playlists via system URLs.

```
const SoundCloudEmbed = ({ trackUrl }) => (
<iframe width="100%" height="166" scrolling="no" frameBorder="no" allow="autoplay"
src={`https://w.soundcloud.com/player/?url=${encodeURIComponent(trackUrl)}`}></iframe>
);
```

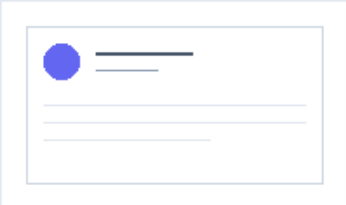


82. Spotify Embed

STYLE SPEC: VIDEO · STACK: React + Tailwind CSS

Iframe embed designed to easily pull Spotify music playlists, songs, or album covers.

```
const SpotifyEmbed = ({ playlistId }) => (
<iframe src={`https://open.spotify.com/embed/playlist/${playlistId}`} width="100%"
height="352" frameBorder="0" allow="autoplay; clipboard-write; encrypted-media; fullscreen;
picture-in-picture" className="rounded-2xl"></iframe>
);
```

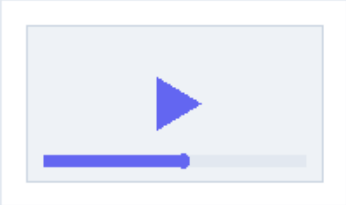


83. X (Twitter) Embed

STYLE SPEC: SOCIAL · STACK: React + Tailwind CSS

Embeds specific X/Twitter user timeline widgets, listing post structures and share icons.

```
const XEmbed = ({ tweetId }) => {
// Loads widget.js from platform.twitter.com and processes elements
return <div id={`tweet-${tweetId}`} className="flex justify-center my-4">{/* Embedded tweet
placeholder */}</div>;
};
```

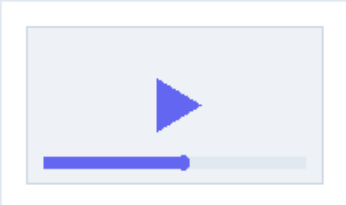


84. YouTube Embed

STYLE SPEC: VIDEO · STACK: React + Tailwind CSS

Custom YouTube iframe embedding widgets supporting auto-play adjustments and fullscreen overlays.

```
const YouTubeEmbed = ({ videoId, autoPlay = false }) => (
<div className="aspect-video w-full rounded-2xl overflow-hidden shadow-md bg-black">
<iframe src={`https://www.youtube.com/embed/${videoId}?autoplay=${autoPlay ? 1 : 0}`}
allow="accelerometer; autoplay; clipboard-write; encrypted-media; gyroscope;
picture-in-picture" allowFullScreen className="w-full h-full border-0"></iframe>
</div>
);
```



85. Vimeo Embed

STYLE SPEC: VIDEO · STACK: React + Tailwind CSS

Vimeo video player embedding frame supporting custom start times, colors, and visual loops.

```
const VimeoEmbed = ({ videoId }) => (
<div className="aspect-video w-full rounded-2xl overflow-hidden shadow-md bg-black">
<iframe src={`https://player.vimeo.com/video/${videoId}`} allow="autoplay; fullscreen;
picture-in-picture" allowFullScreen className="w-full h-full border-0"></iframe>
</div>
);
```