

```
# 🎬 PROTÓTIPO INTERATIVO COMPLETO – PÁGINA  
PORTFOLIO SHOWCASE  
## Site: portfoliodanilo.com  
## Sistema: Ghost Design System  
## Documento Canônico – Estrutura + Motion + Interação +  
Parallax Lerp  
## Versão: 2.0 – COM ANIMAÇÕES PARALLAX
```

🎯 OBJETIVO DO PROTÓTIPO

Criar a ****página Portfolio Showcase completa**** com:

- Hero Section com vídeo em loop
- Grid de projetos com ****Parallax Lerp**** (scroll suave)
- Modal/Página Interna de Projeto (2 tipos)
- Sistema de animação editorial silencioso
- Navegação fluida e contextual
- Coerência total com Ghost System

🔨 ESTRUTURA DA PÁGINA

📂 Hierarquia de Componentes

```

```
PortfolioShowcase
 └── Hero Section (Video Loop)
 ├── Video Background
 ├── Overlay Gradient
 ├── Título "portfólio showcase"
 └── CTA "vamos trabalhar juntos"
 └── Projects Gallery (Parallax Lerp)
 ├── Gallery Container (fixed)
 ├── Gallery Track (animated)
 └── ProjectCard[] (clicáveis + parallax)
 └── Card Image Wrapper (parallax interno)
 └── Clients Brands
 └── Contact
 └── Footer
 └── PortfolioModal (quando ativo)
 ├── Backdrop
 └── ModalContainer
 ├── CloseButton
 ├── ProjectContent (Tipo A ou B)
 ├── MainMedia
 ├── ProjectTitle
 └── ProjectMeta
```

```
 | └ SecondaryContent (galeria/texto)
`-- AnimatePresence
...

```

## # \*\*2. DESIGN SYSTEM\*\*

### ### 2.1 Color Palette

| Token           | Value     | Uso                                                       |
|-----------------|-----------|-----------------------------------------------------------|
| bluePrimary     | `#0048ff` | Cor primária da marca, CTAs, links, elementos interativos |
| blueAccent      | `#4fe6ff` | Destaques secundários, brilhos "ghost"/atmosfera          |
| purpleDetails   | `#8705f2` | Pequenos detalhes e highlights                            |
| pinkDetails     | `#f501d3` | Pequenos detalhes, ênfases pontuais                       |
| background      | `#040013` | Fundo escuro principal                                    |
| backgroundLight | `#f0f0f0` | Seções claras (forms, blocos alternados)                  |
| text            | `#fcffff` | Texto principal em fundo escuro                           |
| textInverse     | `#0e0e0e` | Texto em fundos claros                                    |
| textEmphasis    | `#2E85F2` | Palavras destacadas no meio do texto                      |
| textHighlight   | `#4fe6ff` | Destaques curtos, intros breves                           |
| textSecondary   | `#a1a3a3` | Infos secundárias, metadata                               |
| neutral         | `#b0d3a`  | Gradientes, fundos sutis                                  |
| neutralLight    | `#F5F5F5` | Fundos de seções secundárias                              |

> Obs: `textEmphasis` estava com `#2E85F2` e `textHilight` com typo – normalizei para `textHighlight`.

### ### 2.2 Typography

**Fonte primária:** TT Norms Pro (self-hosted, fallback: `ui-sans-serif, system-ui`)

Tokens de texto **responsivos** (usando `clamp`) para manter coerência em todos os breakpoints:

| Token   | Mobile (~<640px) | Desktop (~≥1024px) | Peso    | Uso                                                           |
|---------|------------------|--------------------|---------|---------------------------------------------------------------|
| display | 2.5rem (40px)    | 4.5rem (72px)      | Black   | Frases grandes no meio da página, não-semânticas (Big Phrase) |
| h1      | 2rem (32px)      | 3.5rem (56px)      | Bold    | Hero headlines, títulos principais                            |
| h2      | 1.5rem (24px)    | 2.5rem (40px)      | Bold    | Títulos de seção                                              |
| h3      | 1.25rem (20px)   | 1.75rem (28px)     | Medium  | Títulos de cards, subtítulos                                  |
| body    | 1rem (16px)      | 1.125rem (18px)    | Regular | Texto corrido                                                 |
| small   | 0.875rem (14px)  | 0.875rem (14px)    | Reg/Med | Labels, legendas                                              |
| micro   | 0.75rem (12px)   | 0.75rem (12px)     | Mono    | Tags, infos de sistema                                        |

#### Tokens em CSS com `clamp()`

```
[`css
:root {
 --font-display: clamp(2.5rem, 5vw, 4.5rem);
 --font-h1: clamp(2rem, 4vw, 3.5rem);
 --font-h2: clamp(1.5rem, 3vw, 2.5rem);
 --font-h3: clamp(1.25rem, 2vw, 1.75rem);
 --font-body: clamp(1rem, 1.2vw, 1.125rem);
 --font-small: 0.875rem;
 --font-micro: 0.75rem;
}

body {
 font-family: "TT Norms Pro", ui-sans-serif, system-ui, -apple-system,
 BlinkMacSystemFont,
 "Segoe UI", sans-serif;
}

.display-text {
 font-size: var(--font-display);
 font-weight: 700;
 line-height: 1.1;
}

.h1 {
 font-size: var(--font-h1);
 font-weight: 700;
 line-height: 1.1;
}

.h2 {
 font-size: var(--font-h2);
```

```

 font-weight: 700;
 line-height: 1.15;
}

.h3 {
 font-size: var(--font-h3);
 font-weight: 500;
 line-height: 1.2;
}

body {
 font-size: var(--font-body);
 font-weight: 400;
 line-height: 1.5;
}

.small {
 font-size: var(--font-small);
}

.micro {
 font-size: var(--font-micro);
 font-family: ui-monospace, SFMono-Regular, Menlo, Monaco, Consolas,
 "Liberation Mono", "Courier New",
 monospace;
}

```

Versão conceitual em Tailwind

```

// tailwind.config.js
module.exports = {
 theme: {
 extend: {
 fontFamily: {
 sans: ['"TT Norms Pro"', "ui-sans-serif", "system-ui"],
 },
 fontSize: {
 display: [
 "clamp(2.5rem, 5vw, 4.5rem)",
 { lineHeight: "1.1", fontWeight: "700" },
],
 h1: [
 "clamp(2rem, 4vw, 3.5rem)",
 { lineHeight: "1.1", fontWeight: "700" },
],
 h2: [
 "clamp(1.5rem, 3vw, 2.5rem)",
 { lineHeight: "1.15", fontWeight: "700" },
],
 h3: [
 "clamp(1.25rem, 2vw, 1.75rem)",
 { lineHeight: "1.2", fontWeight: "500" },
],
 body: [
 "clamp(1rem, 1.2vw, 1.125rem)",

```

```

 { lineHeight: "1.5", fontWeight: "400" },
],
 small: ["0.875rem", { lineHeight: "1.4" }],
 micro: ["0.75rem", { lineHeight: "1.4" }],
},
},
},
} ;'

```

## ## 2.3 Spacing & Grid

### Container

- max-width: 1680px
- Padding horizontal: clamp(24px, 5vw, 96px)

### Ritmo Vertical

- Seções: py-16 md:py-24
- Componentes: gap-8 md:gap-12
- Elementos internos: gap-4 md:gap-6

### Grid (Tailwind)

- Mobile (até md):
- Layout: 1 coluna (grid-cols-1 ou flex flex-col)
- w-full
- Alinhamento:
- text-center para todos os textos
- items-center e justify-center para stacks verticais (flex-col)
- Tablet (md:):
- Cards em md:grid-cols-2
- Hero / destaque podem continuar 1 coluna
- Textos podem voltar a text-left se fizer sentido
- Desktop (lg:+):
- Distribuição customizada por seção
- Textos geralmente alinhados à esquerda para leitura longa

Regra de alinhamento para mobile (base do sistema):

Breakpoint padrão: < 768px

Regra:

- Todos os títulos (display, h1, h2, h3), parágrafos e CTAs usam text-align: center.
- Componentes em coluna usam align-items: center.
- Imagens e ícones principais centralizados (margin-inline: auto).

Exemplo padrão de seção:

```
<section className="flex flex-col items-center text-center md:items-start
md:text-left">
 {/* conteúdo */}
</section>
```

---

## ## 2.4 Animation Principles

Filosofia: animações orgânicas e intencionais, nunca gratuitas.

Core Library: Framer Motion

Diretrizes:

- Animar apenas transform e opacity (performance)
- Easing: cubic-bezier(0.22, 1, 0.36, 1) (easeOutExpo)
- Duração: 300–700ms na maioria das transições
- Stagger: 60–120ms entre elementos sequenciais
- Respeitar prefers-reduced-motion: desabilitar animações não essenciais

Padrões comuns:

```
// Scroll reveal
<motion.div
 initial={{ opacity: 0, y: 24 }}
 whileInView={{ opacity: 1, y: 0 }}
 transition={{ duration: 0.6, ease: [0.22, 1, 0.36, 1] }}>
</>

// Hover (botões, cards)
<motion.button
 whileHover={{ scale: 1.02, y: -2 }}
 transition={{ duration: 0.3 }}>
</>

// Staggered children
const variants = {
 hidden: { opacity: 0 },
 visible: {
 opacity: 1,
 transition: { staggerChildren: 0.08 },
 },
};
```

Em mobile, como tudo é centralizado e o fluxo é vertical, as entradas preferenciais vêm de baixo (y: 24 → 0) acompanhando o scroll.

---

## ## 2.5 Display Text / Big Phrases (Frases em destaque)

Frases grandes no meio da página, com grande destaque visual, mas sem função de título semântico.

Token: display

Diretrizes de uso:

- Quando usar:

- Frases de impacto, statements da marca, quotes, promessas fortes de seção.
- Semântica:
- Usar como `<p>` ou `<span>` com classe específica:
- `className="display-text"` ou `className="text-display"`
- Exemplo:

```
<p className="text-display">
 Construímos experiências digitais que parecem magia, mas são guiadas por
 dados.
</p>
```

- Alinhamento:
- Mobile: sempre centralizado, com largura limitada:
- Ex.: `className="text-display max-w-2xl mx-auto text-center"`
- Desktop: pode ser centralizado ou seguir a grid da seção (recomendado manter centralizado em blocos de destaque).
- Espaçamento:
- Mais respiro que títulos normais:
- Ex.: `mt-16 mb-12` (ajustar conforme a seção).
- Cores:
- Base: `text (#fcffff)`
- Palavras-chave com `textEmphasis` e `textHighlight`.

Exemplo em JSX/Tailwind:

```
<section className="py-16 flex flex-col items-center text-center">
 <p className="text-display max-w-2xl mx-auto">
 Criamos produtos que parecem
 magia, mas são construídos
 com
 engenharia séria.
 </p>
</section>
```

## ## 2.6 Global Assets

Logos:

- Favicon:  
`https://aymuvxysygrwoiccsjgxj.supabase.co/storage/v1/object/public/logo_site/Favicon.svg``
- Favicon Light:  
`https://aymuvxysygrwoiccsjgxj.supabase.co/storage/v1/object/public/logo_site/FaviconLight.svg``
- Logo Light (full):  
`https://aymuvxysygrwoiccsjgxj.supabase.co/storage/v1/object/public/logo_site/LogoLight.svg``
- Logo Dark (full):  
`https://aymuvxysygrwoiccsjgxj.supabase.co/storage/v1/object/public/logo_site/LogoDark.svg``

## ## 2.7 Fonts:

```
- font-family: 'TT Norms Pro';
src:
 url('https://aymuvxysygrwoicsjgxj.supabase
.co/storage/v1/object/public/assets/fonts/TT%20Norms%20Pro%20Thin
.woff2') format('woff2');
font-weight: 100;
font-style: normal;
font-display: swap;

- font-face {
font-family: 'TT Norms Pro';
src:
 url('https://aymuvxysygrwoicsjgxj.supabase
.co/storage/v1/object/public/assets/fonts/TT%20Norms%20Pro%20Light
.woff2') format('woff2');
font-weight: 300;
font-style: normal;
font-display: swap;

- font-face {
font-family: 'TT Norms Pro';
src:
 url('https://aymuvxysygrwoicsjgxj.supabase
.co/storage/v1/object/public/assets/fonts/TT%20Norms%20Pro%20Regular
.woff2') format('woff2');
font-weight: 400;
font-style: normal;
font-display: swap;

- font-face {
font-family: 'TT Norms Pro';
src:
 url('https://aymuvxysygrwoicsjgxj.supabase
.co/storage/v1/object/public/assets/fonts/TT%20Norms%20Pro%20Medium
.woff2') format('woff2');
font-weight: 500;
font-style: normal;
font-display: swap;

- font-face {
font-family: 'TT Norms Pro';
src:
 url('https://aymuvxysygrwoicsjgxj.supabase
.co/storage/v1/object/public/assets/fonts/TT%20Norms%20Pro%20Bold
.woff2') format('woff2');
font-weight: 700;
font-style: normal;
font-display: swap;

- font-face {
font-family: 'TT Norms Pro';
```

```

src:
 url('https://aymuvxysygrwoicsjgxj.supabase
 .co/storage/v1/object/public/assets/fonts/TT%20Norms%20Pro%20Black
 .woff2') format('woff2');
font-weight: 900;
font-style: normal;
font-display: swap;
}

/* Fonte Mono para Tags */
@font-face {
 font-family: 'PPSupplyMono';
 src: url('https://assets.codepen.io/7558/PPSupplyMono-Variation.woff2')
 format('woff2');
 font-weight: 100 900;
 font-style: normal;
 font-display: swap;
}

```

## Videos:

- Manifesto Video:  
`<https://aymuvxysygrwoicsjgxj.supabase.co/storage/v1/object/public/project-videos/VIDEO-APRESENTACAO-PORTFOLIO.mp4>`

## Client Logos:

- 12 monochromatic SVG logos: `client1.svg` through `client12.svg`
- Base URL: `https://aymuvxysygrwoicsjgxj.supabase.co/storage/v1/o

-----

## # \*\* HERO SECTION – VÍDEO LOOPING

```

Estrutura HTML/JSX
```tsx
<section className="hero-section relative h-screen overflow-hidden">
  <video
    autoComplete
    loop
    muted
    playsInline
    className="absolute inset-0 w-full h-full object-cover"
  >
    <source src="[URL_DO_VIDEO]" type="video/mp4" />
  </video>

  <div className="overlay absolute inset-0 bg-gradient-to-b from-black/60
    via-black/40 to-black/60" />

  <div className="content relative h-full flex flex-col items-center
    justify-center text-white px-4">
    <h1 className="text-4xl md:text-6xl font-bold mb-6 text-center">
      <span className="text-blue-400">portfólio</span> showcase
    </h1>
  </div>
</section>

```

```
</h1>
<button className="cta bg-blue-500 hover:bg-blue-600 text-white px-8 py-3 rounded-full">
  vamos trabalhar juntos →
</button>
</div>
</section>
```

```

### ### Especificações Técnicas

#### \*\*Vídeo:\*\*

- `autoPlay`: inicia automaticamente
- `loop`: repete infinitamente
- `muted`: sem som (necessário para autoplay)
- `playsInline`: reproduz inline em mobile
- `object-fit: cover`: cobre toda a área

#### \*\*Overlay:\*\*

- Gradient vertical: `from-black/60 via-black/40 to-black/60`
- Garante legibilidade do texto sobre qualquer vídeo

#### \*\*Título:\*\*

- "portfólio" em azul (`text-blue-400`)
- "showcase" em branco
- Centralizado vertical e horizontalmente

#### \*\*CTA:\*\*

- Cor: `bg-blue-500`
- Hover: `hover:bg-blue-600`
- Border radius: `rounded-full`
- Transição suave: `transition-all duration-300`
- Efeito scale no hover: `hover:scale-105`

----

## ## 🎨 GALLERY COM PARALLAX LERP

### ### 🧠 Conceito do Parallax Lerp

O sistema usa **Linear Interpolation (Lerp)** para criar:

- Scroll suave e fluido
- Movimento parallax independente em cada imagem
- Performance otimizada com `requestAnimationFrame`

### ### 📈 Estrutura HTML/CSS

```
```html
<section class="gallery" ref={galleryRef}>
  <div class="gallery-track" ref={trackRef}>
    <div class="card" ref={cardRef}>
      <div class="card-image-wrapper">
        
      </div>
    </div>
  </div>
</section>
```

```
</div>
<div class="card-overlay">
  <!-- Conteúdo -->
</div>
</div>
<!-- Mais cards... -->
</div>
</section>
```

```

### ### 🎨 CSS Crítico

```
``css
.gallery {
 /* Height será setado dinamicamente via JS */
}

.gallery-track {
 position: fixed;
 top: 0;
 left: 0;
 display: grid;
 grid-template-columns: repeat(3, 1fr);
 gap: 0.25rem;
 padding: 0.25rem;
 will-change: transform;
}

.card {
 height: 400px;
 overflow: hidden;
 border-radius: 8px;
 cursor: pointer;
 transition: transform 0.3s ease;
}

.card:hover {
 transform: translateY(-4px);
 box-shadow: 0 12px 24px rgba(0, 0, 0, 0.08);
}

.card-image-wrapper {
 height: 135%; /* 35% maior que o card para criar espaço de parallax */
 will-change: transform;
}

.card-image-wrapper img {
 width: 100%;
 height: 100%;
 object-fit: cover;
}

/* Responsivo */
@media (max-width: 1024px) {
```

```

.gallery-track {
 grid-template-columns: repeat(2, 1fr);
}
}

@media (max-width: 640px) {
 .gallery-track {
 grid-template-columns: repeat(1, 1fr);
 }
}
```

```

🌐 JavaScript – Sistema Parallax Lerp

```

```javascript
// Refs e variáveis
const galleryRef = useRef(null);
const trackRef = useRef(null);
const cardsRef = useRef([]);
const rafRef = useRef(null);
const startYRef = useRef(0);
const endYRef = useRef(0);
const easing = 0.05; // Suavidade do lerp (quanto menor, mais suave)

// Função Lerp
const lerp = (start, end, t) => start * (1 - t) + end * t;

// Função Parallax para cada card
const parallax = (cardElement) => {
 const wrapper = cardElement.querySelector('.card-image-wrapper');
 if (!wrapper) return;

 // Diferença entre altura do card e altura da imagem
 const diff = cardElement.offsetHeight - wrapper.offsetHeight;

 // Posição do card na viewport
 const { top } = cardElement.getBoundingClientRect();

 // Progresso (0 = topo da tela, 1 = fundo da tela)
 const progress = top / window.innerHeight;

 // Posição Y do parallax
 const yPos = diff * progress;

 wrapper.style.transform = `translateY(${yPos}px)';
};

// Ativa parallax em todos os cards
const activateParallax = () => {
 cardsRef.current.forEach(card => {
 if (card) parallax(card);
 });
};

```

```

// Update Loop principal
const updateScroll = () => {
 if (!galleryRef.current || !trackRef.current) return;

 // Lerp entre posição atual e posição alvo
 startYRef.current = lerp(startYRef.current, endYRef.current, easing);

 // Atualiza altura da galeria (para criar espaço de scroll)
 galleryRef.current.style.height = `${trackRef.current.clientHeight}px`;

 // Move o track
 trackRef.current.style.transform = `translateY(-${startYRef.current}px)`;

 // Ativa parallax em cada card
 activateParallax();

 // Continua o loop
 rafRef.current = requestAnimationFrame(updateScroll);

 // Para o loop quando chegar muito perto do target
 if (Math.abs(startYRef.current - window.scrollY) < 0.1) {
 cancelAnimationFrame(rafRef.current);
 }
};

// Handler do scroll
const startScroll = () => {
 endYRef.current = window.scrollY;
 if (rafRef.current) cancelAnimationFrame(rafRef.current);
 rafRef.current = requestAnimationFrame(updateScroll);
};

// Setup dos event listeners
useEffect(() => {
 const handleScroll = () => startScroll();
 const handleResize = () => updateScroll();

 // Inicializa
 updateScroll();

 window.addEventListener('scroll', handleScroll);
 window.addEventListener('resize', handleResize);

 return () => {
 window.removeEventListener('scroll', handleScroll);
 window.removeEventListener('resize', handleResize);
 if (rafRef.current) cancelAnimationFrame(rafRef.current);
 };
}, []);
```

```

🔍 Como Funciona o Parallax Lerp

1. **Gallery Container** (`galleryRef`):

- Tem altura dinâmica baseada no conteúdo
- Cria espaço para scroll natural

2. ****Gallery Track**** (`trackRef`):

- `position: fixed` → fica fixo na tela
- Animado via `transform: translateY()`
- Lerp cria movimento suave

3. ****Cada Card**:**

- Wrapper da imagem tem 135% de altura
- Movimento parallax independente
- Baseado na posição do card na viewport

4. ****Loop de Animação**:**

- `requestAnimationFrame` garante 60fps
- Lerp interpola entre posição atual e target
- Para automaticamente quando chega no destino

🧙 PROJECT CARD – ANATOMIA COMPLETA

```
### Estrutura Visual
```tsx
<div className="card" onClick={onClick}>
 <div className="card-image-wrapper">

 </div>

 <div className="card-overlay">
 <h3>{project.title}</h3>
 <div className="card-meta">
 {project.client}
 •
 {project.year}
 </div>
 <div className="card-tags">
 {project.tags.map(tag => (
 {tag}
))}
 </div>
 </div>
</div>
```

```

Estados do Card

```
#### Default
```css
.card {
 transform: none;
}

.card-overlay {
```

```

 opacity: 0;
}

```
#### Hover
```
.css
.card:hover {
 transform: translateY(-4px);
 box-shadow: 0 12px 24px rgba(0, 0, 0, 0.08);
}

.card-overlay {
 opacity: 1;
 background: linear-gradient(to top,
 rgba(0, 0, 0, 0.9) 0%,
 rgba(0, 0, 0, 0.5) 50%,
 transparent 100%
);
}
```

```

Active (clique)
 - Trigger modal/página interna
 - Card permanece visível no fundo
 - Backdrop escurece a página

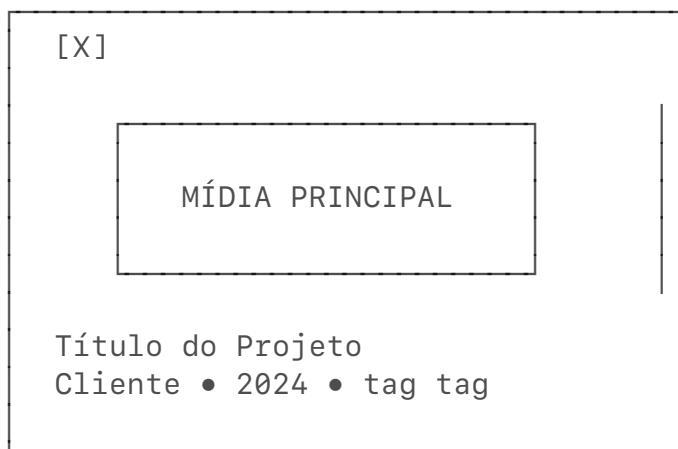
🎭 MODAL / PÁGINA INTERNA – TIPOS

A TIPO A – ZOOM VIEWER

Quando usar:

- Projeto visual simples
- Uma peça principal forte
- Foco em observação

Layout:



```

**\*\*Código:\*\***

```
```tsx
<div className="modal-type-a">
  <div className="media-container">
    <img src={project.image} alt={project.title} />
  </div>

  <div className="info-container">
    <h2>{project.title}</h2>

    <div className="meta">
      <span>{project.client}</span>
      <span>•</span>
      <span>{project.year}</span>
    </div>

    <div className="tags">
      {project.tags.map(tag => (
        <span key={tag}>{tag}</span>
      ))}
    </div>
  </div>
````
```

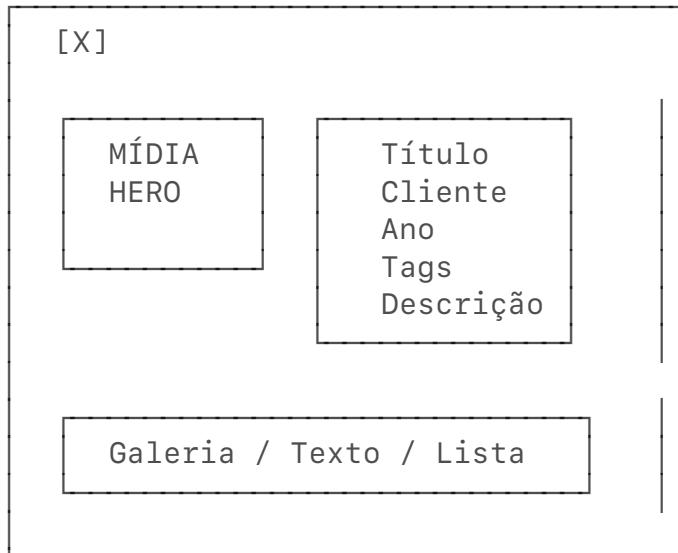
---  
### **B** TIPO B – PÁGINA INTERNA DE PROJETO

**\*\*Quando usar:\*\***

- Projeto complexo
- Múltiplas entregas
- Contexto necessário

**\*\*Layout:\*\***

```



```

\*\*Código:\*\*

```
```tsx
<div className="modal-type-b">
  <div className="hero-section">
    <div className="hero-media">
      <img src={project.image} alt={project.title} />
    </div>

    <div className="hero-info">
      <h2>{project.title}</h2>

      <div className="meta">
        <span>{project.client}</span>
        <span>•</span>
        <span>{project.year}</span>
      </div>

      <div className="tags">
        {project.tags.map(tag => (
          <span key={tag}>{tag}</span>
        )))
      </div>

      <p className="description">
        {project.description}
      </p>
    </div>
  </div>

  {project.gallery && (
    <div className="gallery-section">
      {project.gallery.map((img, idx) => (
        <img key={idx} src={img} alt={`${project.title} - ${idx + 1}`} />
      )))
    </div>
  )}
</div>
```

```

## 🎬 ANIMAÇÃO – TIMELINE CANÔNICO DO MODAL

### ⏹️ ABERTURA DO MODAL

#### T = 0ms – Estado Inicial

```
```ts
// Backdrop
opacity: 0
```

```
// Modal Container
opacity: 0
scale: 0.98
y: 12px
```

```
// Conteúdo interno  
visibility: hidden  
opacity: 0  
```
```

```

T = 0 → 180ms – Backdrop Aparece
```ts  
backdrop {  
  opacity: 0 → 1  
  transition: linear 180ms  
}  
```
```

#### \*\*CSS/Framer Motion:\*\*

```
```tsx  
<motion.div  
  className="backdrop"  
  initial={{ opacity: 0 }}  
  animate={{ opacity: 1 }}  
  exit={{ opacity: 0 }}  
  transition={{ duration: 0.18, ease: 'linear' }}  
/>  
```
```

```

T = 120 → 380ms – Container Aparece
```ts  
modalContainer {  
  opacity: 0 → 1  
  scale: 0.98 → 1  
  y: 12 → 0  
  transition: cubic-bezier(0.22, 1, 0.36, 1) 260ms  
  delay: 120ms  
}  
```
```

#### \*\*CSS/Framer Motion:\*\*

```
```tsx  
<motion.div  
  className="modal-container"  
  initial={{ opacity: 0, scale: 0.98, y: 12 }}  
  animate={{ opacity: 1, scale: 1, y: 0 }}  
  exit={{ opacity: 0, scale: 0.98, y: 8 }}  
  transition={{  
    opacity: { duration: 0.26 },  
    scale: { duration: 0.26, ease: [0.22, 1, 0.36, 1] },  
    y: { duration: 0.26, ease: [0.22, 1, 0.36, 1] }  
  }}  
/>  
```
```

```

T = 380 → 520ms – Pausa Consciente
- **Nenhuma animação**
- Usuário reconhece contexto
- Estabilização visual
- Container está visível mas conteúdo ainda não
```

```

T = 520 → 760ms – Mídia Principal
```

```
```ts  
mainMedia {  
    opacity: 0 → 1  
    transition: ease-out 240ms  
    delay: 520ms  
}  
// ✗ Sem translate  
// ✗ Sem scale  
// Apenas presença  
```
```

```
Implementação:
```

```
```tsx  
<motion.div  
    className="main-media"  
    initial={{ opacity: 0 }}  
    animate={{ opacity: 1 }}  
    transition={{  
        delay: 0.52,  
        duration: 0.24,  
        ease: 'easeOut'  
    }}  
>  
    <img src={project.image} alt={project.title} />  
</motion.div>  
```
```

```

T = 760 → 960ms – Título
```

```
```ts  
projectTitle {  
    opacity: 0 → 1  
    y: 6 → 0  
    duration: 200ms  
    delay: 760ms  
}  
```
```

```
Implementação:
```

```
```tsx  
<motion.h2  
    initial={{ opacity: 0, y: 6 }}
```

```
animate={{ opacity: 1, y: 0 }}
transition={{
  delay: 0.76,
  duration: 0.2
}}
>
  {project.title}
</motion.h2>
```

```

```
T = 960 → 1120ms – Meta Informações
```ts
projectMeta {
  opacity: 0 → 1
  y: 4 → 0
  duration: 160ms
  delay: 960ms
}
```

```

#### \*\*Implementação:\*\*

```
```tsx
<motion.div
  className="meta"
  initial={{ opacity: 0, y: 4 }}
  animate={{ opacity: 1, y: 0 }}
  transition={{
    delay: 0.96,
    duration: 0.16
  }}
>
  <span>{project.client}</span>
  <span>•</span>
  <span>{project.year}</span>
</motion.div>
```

```

#### #### T = 1120 → 1500ms – Conteúdo Secundário

```
```ts
// Galeria, texto, bullets
secondaryContent {
  opacity: 0 → 1
  y: 8 → 0
  stagger: 80ms
  duration: 200ms
  delay: 1120ms (base)
}
```

```

#### \*\*Implementação com Stagger:\*\*

```
```tsx
```

```
{project.gallery?.map((img, idx) => (
  <motion.img
    key={idx}
    src={img}
    initial={{ opacity: 0, y: 8 }}
    animate={{ opacity: 1, y: 0 }}
    transition={{
      delay: 1.12 + (idx * 0.08),
      duration: 0.2
    }}
  />
))}
```

```
### ⏱ ESTADO IDLE (T > 1500ms)
```

Após entrada completa:

- Nenhuma animação contínua
- Nada flutua
- Nada pulsa
- Foco total em leitura
- Scroll interno habilitado
- Parallax do fundo está pausado (body overflow hidden)

```
### ⏱ FECHAMENTO DO MODAL
```

```
#### T = 0 → 180ms – Container Sai
```ts
modalContainer {
 opacity: 1 → 0
 scale: 1 → 0.98
 y: 0 → 8
 transition: ease-in 180ms
}
```
``
```

```
#### T = 0 → 150ms – Backdrop Sai
```ts
backdrop {
 opacity: 1 → 0
 transition: linear 150ms
}
```
``
```

Implementação:

```
```tsx
<AnimatePresence>
 {selectedProject && (
```

```
<PortfolioModal
 project={selectedProject}
 onClose={() => setSelectedProject(null)}
/>
)}
</AnimatePresence>
```
---
```

🌐 INTERAÇÃO – FLUXO COMPLETO

1 Usuário rola a página

- Parallax lerp ativo
- Cards se movem suavemente
- Imagens internas fazem parallax independente

2 Usuário passa mouse sobre card

```
```ts
onMouseEnter={() => setIsHovered(true)}
```

```
// CSS aplicado
```

```
.card-overlay {
 opacity: 0 → 1
 backdrop-filter: blur(4px)
}
```

### ### 3 Usuário clica em um card

```
```ts
onClick={() => setSelectedProject(project)}
```

```
// Ações
```

1. Estado atualizado
2. Modal renderizado via Portal
3. Scroll da página bloqueado (body overflow: hidden)
4. Foco move para o modal

4 Modal/Página Interna abre

- Backdrop aparece (0→180ms)
- Container aparece (120→380ms)
- Pausa (380→520ms)
- Conteúdo se revela em sequência (520→1500ms)
- Scroll interno disponível após 1500ms

5 Usuário lê/explora o projeto

- Scroll interno disponível
- Botão fechar sempre visível (fixed position)
- ESC funciona
- Click no backdrop funciona

6 Usuário fecha modal

Gatilhos:

- Click no backdrop
- Click no botão [X]
- Tecla ESC

Código:

```
```tsx
const handleClose = () => {
 setSelectedProject(null);
 // Body overflow restaurado automaticamente no useEffect cleanup
};

// ESC handler
useEffect(() => {
 const handleEscape = (e) => {
 if (e.key === 'Escape') handleClose();
 };
 window.addEventListener('keydown', handleEscape);
 return () => window.removeEventListener('keydown', handleEscape);
}, []);

// Backdrop click
<div onClick={(e) => {
 if (e.target === e.currentTarget) handleClose();
}} />
```

```

Resultado:

- Modal fecha com animação reversa
- Foco retorna ao card original
- Scroll da página é restaurado
- Parallax volta a funcionar

IMPLEMENTAÇÃO REACT COMPLETA

Estado Global

```
```tsx
const [selectedProject, setSelectedProject] = useState<Project | null>(null);
```

```

Hero Section

```
```tsx
<section className="relative h-screen overflow-hidden">
 <video
 autoPlay
 loop
 muted
 playsInline
 className="absolute inset-0 w-full h-full object-cover"
 >
```

```

```

>
  <source src="video.mp4" type="video/mp4" />
</video>

<div className="absolute inset-0 bg-gradient-to-b from-black/60
via-black/40 to-black/60" />

<div className="relative h-full flex flex-col items-center justify-center
text-white px-4">
  <h1 className="text-4xl md:text-6xl font-bold mb-6 text-center">
    <span className="text-blue-400">portfólio</span> showcase
  </h1>
  <button className="bg-blue-500 hover:bg-blue-600 text-white px-8 py-3
rounded-full transition-all duration-300 transform hover:scale-105
flex items-center gap-2">
    vamos trabalhar juntos
    <span className="text-xl">→</span>
  </button>
</div>
</section>
```

Gallery com Parallax
```tsx
<section ref={galleryRef} className="gallery">
  <div
    ref={trackRef}
    className="gallery-track fixed grid grid-cols-1 md:grid-cols-2
    lg:grid-cols-3 gap-1 p-1"
  >
    {projects.map((project, index) => (
      <ProjectCard
        key={project.id}
        ref={(el) => (cardsRef.current[index] = el)}
        project={project}
        onClick={() => setSelectedProject(project)}
      />
    )))
  </div>
</section>
```

Modal com Portal
```tsx
import { createPortal } from 'react-dom';

{selectedProject && createPortal(
  <PortfolioModal
    project={selectedProject}
    onClose={() => setSelectedProject(null)}
  />,
  document.body
)}
```

```

```
useEffect - Parallax Setup
```tsx
useEffect(() => {
  const handleScroll = () => startScroll();
  const handleResize = () => updateScroll();

  updateScroll();
  window.addEventListener('scroll', handleScroll);
  window.addEventListener('resize', handleResize);

  return () => {
    window.removeEventListener('scroll', handleScroll);
    window.removeEventListener('resize', handleResize);
    if (rafRef.current) cancelAnimationFrame(rafRef.current);
  };
}, []);
```

```

```
useEffect - Modal Body Lock
```tsx
useEffect(() => {
  if (selectedProject) {
    document.body.style.overflow = 'hidden';
    return () => {
      document.body.style.overflow = '';
    };
  }
}, [selectedProject]);
```

```

---

## ## 🎨 ESTILO – CSS/TAILWIND COMPLETO

```
Hero Section
```css
.hero-section {
  position: relative;
  height: 100vh;
  overflow: hidden;
}

.hero-section video {
  position: absolute;
  inset: 0;
  width: 100%;
  height: 100%;
  object-fit: cover;
}

.hero-section .overlay {
  position: absolute;
  inset: 0;
  background: linear-gradient(

```

```
        to bottom,
        rgba(0, 0, 0, 0.6) 0%,
        rgba(0, 0, 0, 0.4) 50%,
        rgba(0, 0, 0, 0.6) 100%
    );
}

.hero-section .content {
    position: relative;
    height: 100%;
    display: flex;
    flex-direction: column;
    align-items: center;
    justify-content: center;
    color: white;
    padding: 1rem;
}
```
Gallery & Track
```css
.gallery {
    /* Height dinâmica via JS */
}

.gallery-track {
    position: fixed;
    top: 0;
    left: 0;
    width: 100%;
    display: grid;
    grid-template-columns: repeat(3, 1fr);
    gap: 0.25rem;
    padding: 0.25rem;
    will-change: transform;
}

@media (max-width: 1024px) {
    .gallery-track {
        grid-template-columns: repeat(2, 1fr);
    }
}

@media (max-width: 640px) {
    .gallery-track {
        grid-template-columns: repeat(1, 1fr);
    }
}
```
Card
```css
.card {
    position: relative;
    height: 400px;
}
```

```
overflow: hidden;
background: #f3f4f6;
border-radius: 0.5rem;
cursor: pointer;
transition: transform 0.3s ease, box-shadow 0.3s ease;
}

.card:hover {
  transform: translateY(-4px);
  box-shadow: 0 12px 24px rgba(0, 0, 0, 0.08);
}

.card-image-wrapper {
  position: absolute;
  inset: 0;
  height: 135%;
  will-change: transform;
}

.card-image-wrapper img {
  width: 100%;
  height: 100%;
  object-fit: cover;
}

.card-overlay {
  position: absolute;
  inset: 0;
  background: linear-gradient(
    to top,
    rgba(0, 0, 0, 0.9) 0%,
    rgba(0, 0, 0, 0.5) 50%,
    transparent 100%
  );
  display: flex;
  flex-direction: column;
  justify-content: flex-end;
  padding: 1.5rem;
  opacity: 0;
  transition: opacity 0.3s ease;
}

.card:hover .card-overlay {
  opacity: 1;
}

```
Modal
```css
.modal-backdrop {
  position: fixed;
  inset: 0;
  background: rgba(0, 0, 0, 0.6);
  backdrop-filter: blur(4px);
  z-index: 50;
}
```

```
display: flex;
align-items: center;
justify-content: center;
padding: 1rem;
}

.modal-container {
  position: relative;
  width: 100%;
  max-width: 1200px;
  max-height: 90vh;
  background: white;
  border-radius: 1.5rem;
  padding: 2rem;
  overflow-y: auto;
  overscroll-behavior: contain;
}

@media (min-width: 768px) {
  .modal-container {
    padding: 3rem;
  }
}

.close-button {
  position: absolute;
  top: 1rem;
  right: 1rem;
  width: 2.5rem;
  height: 2.5rem;
<function_calls>
<invoke name="artifacts">
<parameter name="command">update</parameter>
<parameter name="id">prototipo_completo_final</parameter>
<parameter name="old_str">  position: absolute;
  top: 1rem;
  right: 1rem;
  width: 2.5rem;
  height: 2.5rem;</parameter>
<parameter name="new_str">  position: absolute;
  top: 1rem;
  right: 1rem;
  width: 2.5rem;
  height: 2.5rem;
  border-radius: 50%;
  background: rgba(0, 0, 0, 0.05);
  display: flex;
  align-items: center;
  justify-content: center;
  transition: background 0.2s ease;
  cursor: pointer;
  z-index: 10;
}

.close-button:hover {
```

```
    background: rgba(0, 0, 0, 0.1);  
}  
````
```

## ## RESPONSIVO – BREAKPOINTS

```
Desktop ($\geq 1024\text{px}$)
```css  
.gallery-track {  
  grid-template-columns: repeat(3, 1fr);  
  gap: 0.25rem;  
  padding: 0.25rem;  
}
```

```
.modal-container {  
  max-width: 1200px;  
  padding: 3rem;  
}
```

```
.card {  
  height: 400px;  
}  
````
```

```
Tablet (768px – 1023px)
```

```
```css  
.gallery-track {  
  grid-template-columns: repeat(2, 1fr);  
}
```

```
.modal-container {  
  max-width: 90vw;  
  padding: 2rem;  
}
```

```
.card {  
  height: 350px;  
}  
````
```

```
Mobile (<768px)
```

```
```css  
.gallery-track {  
  grid-template-columns: repeat(1, 1fr);  
  gap: 0.5rem;  
  padding: 0.5rem;  
}
```

```
.modal-container {  
  width: 95vw;  
  padding: 1.5rem;  
  border-radius: 1rem;
```

```
}

.card {
  height: 300px;
}

.hero-section h1 {
  font-size: 2rem;
}

.hero-section button {
  font-size: 0.875rem;
  padding: 0.625rem 1.5rem;
}
```

```

## ## PROIBIÇÕES ABSOLUTAS

### ### Na Página Grid

-  Animações agressivas
-  Autoplay de áudio
-  Carrosséis automáticos não controláveis
-  Parallax exagerado (>150% de movimento)
-  Scroll hijacking

### ### No Hero

-  Vídeo com som (mesmo muted=false)
-  Autoplay sem controles
-  Vídeo muito pesado (>10MB)
-  Ausência de fallback para imagem

### ### No Modal/Página Interna

-  Animação por scroll interno
-  Parallax dentro do modal
-  Blur decorativo excessivo
-  Spring / bounce
-  Entrada simultânea de tudo
-  Linguagem de landing page
-  CTAs promocionais
-  Popups dentro de popups

----

## ## ACESSIBILIDADE

```
Modal
```tsx
<div
  role="dialog"
  aria-modal="true"
  aria-labelledby="project-title"
  aria-describedby="project-description"
>
  <h2 id="project-title">{project.title}</h2>
  <p id="project-description">{project.description}</p>
</div>
```

Foco
```tsx
useEffect(() => {
  if (selectedProject) {
    const closeButton = document.querySelector('.close-button');
    closeButton?.focus();

    // Salva elemento focado anterior
    const previousFocus = document.activeElement;

    return () => {
      // Restaura foco ao fechar
      previousFocus?.focus();
    };
  }
}, [selectedProject]);
```

Teclado
- `ESC` fecha modal
- `Tab` navega elementos internos
- `Shift + Tab` navegação reversa
- `Enter` ou `Space` ativa botões

Screen Readers
```tsx
<button
  aria-label="Fechar visualização do projeto"
  onClick={onClose}
>
  <X aria-hidden="true" />
</button>

<img
  src={project.image}
  alt={`Projeto ${project.title} - ${project.client}`}
  loading="lazy"
/>
```

Reduced Motion
```tsx
```

```
const prefersReducedMotion = window.matchMedia(
  '(prefers-reduced-motion: reduce)'
).matches;

const transition = prefersReducedMotion
  ? { duration: 0 }
  : { duration: 0.26, ease: [0.22, 1, 0.36, 1] };
```

```

## ## ⚡ PERFORMANCE

### ### Otimizações Críticas

#### #### 1. Lazy Loading de Imagens

```
```tsx
![{project.title}]({project.image})
```
```

```

2. will-change

```
```css
.gallery-track {
 will-change: transform;
}

.card-image-wrapper {
 will-change: transform;
}

/* Remover will-change após animação */
.modal-container.animation-complete {
 will-change: auto;
}
```
```

```

#### #### 3. requestAnimationFrame

```
```javascript
// Cancela RAF quando não necessário
if (Math.abs(startYRef.current - window.scrollY) < 0.1) {
  cancelAnimationFrame(rafRef.current);
}
```
```

```

4. Debounce em Resize

```
```javascript
let resizeTimeout;
const handleResize = () => {
 clearTimeout(resizeTimeout);
```
```

```

```

resizeTimeout = setTimeout(() => {
 updateScroll();
}, 100);
};

5. Portal para Modal
```tsx
import { createPortal } from 'react-dom';

// Renderiza no final do body, evitando reflows
createPortal(<Modal />, document.body)
```

6. Overscroll Contain
```css
.modal-container {
  overscroll-behavior: contain;
}
```

7. Image Optimization
- WebP com fallback para JPEG
- Srcset para diferentes resoluções
- Tamanho adequado (não usar imagens gigantes)

```tsx
![{project.title}]({project.image})
```

```

## ## MÉTRICAS DE PERFORMANCE

```

Targets
- **FCP (First Contentful Paint)**: <1.5s
- **LCP (Largest Contentful Paint)**: <2.5s
- **TTI (Time to Interactive)**: <3.5s
- **CLS (Cumulative Layout Shift)**: <0.1
- **FPS durante scroll**: 60fps
- **Parallax lag**: <16ms

Como Medir
```javascript

```

```

// FPS Monitor
let lastTime = performance.now();
let frames = 0;

function measureFPS() {
  const now = performance.now();
  frames++;

  if (now >= lastTime + 1000) {
    const fps = Math.round((frames * 1000) / (now - lastTime));
    console.log(`FPS: ${fps}`);
    frames = 0;
    lastTime = now;
  }
}

requestAnimationFrame(measureFPS);
}

measureFPS();
```

```

----

## ## TESTES RECOMENDADOS

### ### Funcionalidade

1.  Abrir/fechar modal múltiplas vezes
2.  Testar todos os gatilhos de fechamento (ESC, backdrop, botão)
3.  Scroll interno em conteúdos longos
4.  Navegação por teclado completa
5.  Parallax funciona em todos os cards
6.  Hover states em todos os cards
7.  Click em cards diferentes

### ### Performance

1.  Verificar FPS durante scroll (deve ser 60fps)
2.  Testar em dispositivos mais lentos
3.  Medir tempo de carregamento de imagens
4.  Validar sem memory leaks (abrir/fechar modal 50x)
5.  Testar com 50+ cards na galeria
6.  Verificar uso de CPU durante parallax

### ### Acessibilidade

1.  Testar com screen reader (NVDA/JAWS)
2.  Navegar apenas com teclado
3.  Testar com prefers-reduced-motion
4.  Validar contraste de cores (WCAG AA)

5.  Testar com zoom 200%
6.  Validar foco visível em todos elementos

#### ### Responsivo

1.  Testar em mobile (320px – 768px)
2.  Testar em tablet (768px – 1024px)
3.  Testar em desktop (1024px+)
4.  Testar rotação de tela
5.  Testar em diferentes navegadores
6.  Testar touch vs mouse interactions

---

## ## CHECKLIST DE VALIDAÇÃO COMPLETO

#### ### Hero Section

- [ ] Vídeo carrega e faz loop corretamente
- [ ] Overlay garante legibilidade do texto
- [ ] CTA tem hover state claro
- [ ] Responsivo em todos os tamanhos
- [ ] Performance ok (vídeo <10MB)

#### ### Grid de Projetos

- [ ] Cards respondem a hover suavemente
- [ ] Parallax lerp funciona em todos os cards
- [ ] Imagens carregam progressivamente
- [ ] Layout responsivo funciona
- [ ] Performance fluida em 60fps
- [ ] Scroll é natural (não hijacked)

#### ### Modal/Página Interna

- [ ] Abertura silenciosa e orientada
- [ ] Pausa perceptível após container (380–520ms)
- [ ] Mídia aparece antes do texto
- [ ] Título antes dos detalhes
- [ ] Conteúdo secundário não compete
- [ ] Fechamento rápido e discreto
- [ ] Scroll interno funciona
- [ ] Não parece landing page

#### ### Interação

- [ ] Click no card abre modal correto
- [ ] ESC fecha modal
- [ ] Click no backdrop fecha modal
- [ ] Click no botão [X] fecha modal
- [ ] Foco retorna ao card original
- [ ] Scroll da página bloqueado durante modal
- [ ] Parallax pausado durante modal
- [ ] Parallax retoma após fechar modal

#### ### Acessibilidade

- [ ] `role="dialog"` presente
- [ ] `aria-modal="true"` presente
- [ ] `aria-label` em botões
- [ ] Foco gerenciado corretamente
- [ ] Screen reader compatível
- [ ] Navegação por teclado completa
- [ ] prefers-reduced-motion respeitado

#### ### Ghost System

- [ ] Não parece landing page
- [ ] Mantém contexto do portfólio
- [ ] Leitura confortável
- [ ] Animação serve à leitura
- [ ] Coerente com página SOBRE
- [ ] Silencioso e editorial
- [ ] Foco no conteúdo, não no efeito

----

## ## RESULTADO ESPERADO

O usuário deve:

1.  Ver hero impactante mas não invasivo
2.  Rolar a página com parallax suave e natural
3.  Ver grid de projetos organizado e convidativo
4.  Sentir curiosidade ao hover nos cards
5.  Clicar naturalmente para explorar
6.  Experimentar abertura calma e orientada
7.  Ler conteúdo sem distrações
8.  Fechar modal e voltar exatamente onde estava
9.  Continuar explorando outros projetos
10.  Sentir continuidade, não ruptura

\*\*O modal não é um destino – é uma extensão natural da página.\*\*

\*\*O parallax não é um show – é um guia visual sutil.\*\*

----

## ## PRINCÍPIOS FINAIS

> \*\*"A tecnologia serve à experiência, não o contrário."\*\*

Cada elemento deste protótipo foi pensado para:

- \*\*Guiar\*\* sem distrair
- \*\*Revelar\*\* sem chocar
- \*\*Animar\*\* sem exagerar
- \*\*Impressionar\*\* pela clareza, não pelo excesso

#### ### Ghost System em Ação

1. \*\*Presença sem peso\*\* – Hero forte mas não opressivo

2. **\*\*Movimento com propósito\*\*** – Parallax guia o olhar
  3. **\*\*Revelação gradual\*\*** – Modal respeita o tempo de leitura
  4. **\*\*Retorno natural\*\*** – Nada se perde ao fechar
- 

## ## DADOS DE EXEMPLO

```
Estrutura de Projeto
```typescript
interface Project {
    id: number;
    title: string;
    client: string;
    year: string;
    tags: string[];
    image: string;
    type: 'A' | 'B';
    description?: string;
    gallery?: string[];
    deliverables?: string[];
    links?: {
        label: string;
        url: string;
    }[];
}
```

Exemplo de Projeto Tipo A
```typescript
{
    id: 1,
    title: 'Visual Identity',
    client: 'Tech Corp',
    year: '2024',
    tags: ['Branding', 'Design'],
    image: 'https://example.com/image.jpg',
    type: 'A',
    description: 'Complete visual identity redesign for a tech startup.'
}
```

Exemplo de Projeto Tipo B
```typescript
{
    id: 2,
    title: 'Garoto – Nestlé',
    client: 'Nestlé',
    year: '2023',
    tags: ['Packaging', 'Campaign'],
    image: 'https://example.com/hero.jpg',
    type: 'B',
    description: 'Embalagens especiais GAROTO para páscoa com identidade renovada.'
}
```

```

gallery: [
  'https://example.com/gallery-1.jpg',
  'https://example.com/gallery-2.jpg',
  'https://example.com/gallery-3.jpg'
],
deliverables: [
  'Redesign de embalagens',
  'Campanha digital',
  'Materiais PDV',
  'Guidelines de marca'
],
links: [
  { label: 'Ver campanha completa', url: 'https://example.com' }
]
}
```

```

## ## 🤖 PROMPT EXECUTOR – AGENT COPILOT

```md

Você deve implementar a Página Portfolio Showcase completa conforme este protótipo canônico.

Arquivos a criar/modificar:

- PortfolioShowcase.tsx (página principal)
- ProjectCard.tsx (card do grid)
- PortfolioModal.tsx (modal/página interna)
- ProjectContent.tsx (conteúdo interno: Tipo A e B)
- useParallax.ts (hook customizado para parallax)

Objetivo:

Sistema completo de portfólio com hero em vídeo, grid de projetos com parallax lerp, e visualização modal seguindo Ghost System.

Ações obrigatórias:

1. HERO SECTION:

- Video background em loop (autoPlay, muted, playsInline)
- Overlay gradient (from-black/60 via-black/40 to-black/60)
- Título "portfólio showcase" (portfólio em azul)
- CTA "vamos trabalhar juntos" com hover

2. GALLERY COM PARALLAX LERP:

- Grid responsivo: 3 cols (desktop) → 2 (tablet) → 1 (mobile)
- Sistema de scroll suave com lerp (easing: 0.05)
- Track fixo com translateY animado
- Cards com parallax interno independente
- Hover states nos cards
- requestAnimationFrame para 60fps

3. MODAL/PÁGINA INTERNA:

- Tipos A (Zoom Viewer) e B (Página Interna)

- Timeline de animação canônico:
 - * Backdrop: 0→180ms (linear)
 - * Container: 120→380ms (ease-out custom)
 - * Pausa: 380→520ms
 - * Mídia: 520→760ms
 - * Título: 760→960ms
 - * Meta: 960→1120ms
 - * Secundário: 1120→1500ms (stagger 80ms)

4. INTERAÇÕES:

- Click no card abre modal
- ESC / backdrop / botão fecha modal
- Body overflow bloqueado durante modal
- Foco gerenciado (vai para fechar, retorna ao card)
- Parallax pausado durante modal

5. PERFORMANCE:

- Lazy load de imagens
- will-change apenas no necessário
- Portal para modal
- overscroll-contain no modal
- Cancelar RAF quando não necessário

6. ACESSIBILIDADE:

- role="dialog" e aria-modal="true"
- aria-label em botões
- Navegação por teclado
- prefers-reduced-motion

Regras de implementação:

- Usar Framer Motion + AnimatePresence
- Usar refs para gallery, track e cards
- Implementar lerp corretamente
- Parallax baseado em getBoundingClientRect
- Criar Portal para modal (document.body)
- Gerenciar foco com useEffect
- Respeitar prefers-reduced-motion
- Lazy load de imagens
- Não adicionar efeitos além do especificado
- Não usar animações por scroll interno no modal
- Não criar linguagem de landing page
- Não usar spring/bounce

Estrutura de pastas sugerida:

```
```
src/
 components/
 portfolio/
 PortfolioShowcase.tsx
 HeroSection.tsx
```

```
ProjectCard.tsx
ProjectsGallery.tsx
PortfolioModal.tsx
ProjectContentTypeA.tsx
ProjectContentTypeB.tsx
hooks/
 useParallax.ts
 useBodyLock.ts
types/
 project.ts
data/
 projects.ts
````
```

Critérios de aceite:

- Hero com vídeo loop funciona corretamente
 - Grid responsivo e performático
 - Parallax lerp suave em 60fps
 - Modal abre/fecha conforme timeline
 - Foco retorna ao card original
 - Acessibilidade completa
 - Coerente com Ghost System
 - Leitura confortável e sem distrações
 - Performance validada (60fps durante scroll)
- ````

REFERÊNCIAS TÉCNICAS

Parallax Lerp Original

- CodePen: <https://codepen.io/danilonovaisv/pen/VYjejyb>
- Técnica: Vertical Scrolling Parallax com Lerp
- Autor: danilonovaisv

Ghost Design System

- Filosofia: Presença sem peso
- Motion: Editorial e silencioso
- Foco: Conteúdo sobre efeito

Framer Motion

- Docs: <https://www.framer.com/motion/>
- AnimatePresence para mount/unmount
- Variants para animações complexas

Performance

- Web Vitals: <https://web.dev/vitals/>
- requestAnimationFrame:
<https://developer.mozilla.org/en-US/docs/Web/API/window/requestAnimationFrame>

VERSIONAMENTO

****v2.0** – Dezembro 2024**

-  Hero Section com vídeo loop
-  Parallax Lerp implementado
-  Timeline de modal atualizado
-  Performance otimizada
-  Acessibilidade completa

****v1.0** – Novembro 2024**

- Grid básico de projetos
- Modal simples
- Sem parallax

GLOSSÁRIO

****Lerp**:** Linear Interpolation – técnica de suavização que interpola entre valor atual e valor alvo

****Easing**:** Taxa de suavização do lerp (0.05 = muito suave, 0.5 = quase instantâneo)

****RAF**:** requestAnimationFrame – API do navegador para animações em 60fps

****Parallax**:** Efeito visual onde elementos se movem em velocidades diferentes criando profundidade

****Modal**:** Janela sobreposta que requer interação antes de retornar ao conteúdo principal

****Portal**:** Técnica React para renderizar componentes fora da hierarquia DOM do pai

****Ghost System**:** Sistema de design focado em presença sutil e conteúdo prioritário

****FIM DO PROTÓTIPO INTERATIVO COMPLETO****</parameter>