

Guía de Optimización Responsiva y Desarrollo Continuo

Portfolio - Eduardo Téllez Valverde

1. FUNCIONALIDAD RESPONSIVA (Prioridad Inmediata)

1.1 Diagnóstico de Problemas Actuales

- **Header:** Solo tiene breakpoint en 768px, falta para tablets (1024px)
- **Hero Section:** Video background no optimizado para móvil
- **Grid Layouts:** Projects y Interests no tienen breakpoints intermedios
- **Custom Cursor:** Activo en móvil (debe desactivarse)
- **Modales:** No ajustan correctamente en pantallas pequeñas

1.2 Sistema de Breakpoints Recomendado

scss

```
// Definir en un archivo global o variables CSS
:root {
    --mobile-sm: 320px;
    --mobile: 480px;
    --tablet: 768px;
    --desktop: 1024px;
    --desktop-lg: 1440px;
}

/* Media queries estándar a implementar */
@media (max-width: 480px) { /* Móvil pequeño */}
@media (max-width: 768px) { /* Móvil/Tablet */}
@media (max-width: 1024px) { /* Tablet/Desktop pequeño */}
@media (min-width: 1440px) { /* Desktop grande */}
```

1.3 Acciones Específicas por Componente

Header.tsx / Header.module.css

- Añadir breakpoint intermedio (1024px)
- Reducir padding en móvil
- Ajustar tamaño del logo dinámicamente

- Mejorar animación del menú hamburguesa

Hero.tsx

- Desactivar video en móvil (< 768px)
- Usar imagen estática optimizada como fallback
- Ajustar tipografía responsiva:

css

```
.hero-main-title {
  font-size: clamp(1.5rem, 5vw, 3.5rem);
}

.hero-subtitle {
  font-size: clamp(0.875rem, 2vw, 1.25rem);
}
```

Projects.tsx & Interests.tsx

- Cambiar grid layout:

css

```
.projects-grid, .interests-grid {
  display: grid;
  grid-template-columns: repeat(auto-fit, minmax(280px, 1fr));
  gap: 1.5rem;
}

@media (max-width: 768px) {
  .projects-grid, .interests-grid {
    grid-template-columns: 1fr;
  }
}
```

CustomCursor.tsx

javascript

```
// Añadir detección de dispositivo táctil
useEffect(() => {
.. const isTouchDevice = 'ontouchstart' in window ||
..   navigator.maxTouchPoints > 0;
..
.. if (isTouchDevice) {
..   return; // No inicializar en dispositivos táctiles
.. }
.. // ... resto del código
}, []);
```

⚡ 2. EFICIENCIA Y PERFORMANCE

2.1 Optimización de Recursos

Lazy Loading de Videos

javascript

```

// Hero.tsx - Ejemplo de implementación
const [videoLoaded, setVideoLoaded] = useState(false);

useEffect(() => {
  const observer = new IntersectionObserver(
    ([entry]) => {
      if (entry.isIntersecting && !videoLoaded) {
        setVideoLoaded(true);
      }
    },
    { threshold: 0.1 }
  );
  if (videoRef.current) {
    observer.observe(videoRef.current);
  }
  return () => observer.disconnect();
}, [videoLoaded]);

// Renderizar video solo cuando sea visible
{videoLoaded && (
  <video autoPlay muted loop ref={videoRef}>
    <source src={`${process.env.NEXT_PUBLIC_BASE_PATH}/video2.mp4`} />
  </video>
)}

```

Optimización de Imágenes con Next.js

javascript

```
// Usar next/image en todos los componentes
import Image from 'next/image';

// Ejemplo para Projects.tsx
<Image
  ..src={`${basePath}/project-image.jpg`}
  ..alt="Project"
  ..width={400}
  ..height={300}
  ..loading="lazy"
  ..placeholder="blur"
  ..blurDataURL="data:image/jpeg;base64,...">
/>
```

2.2 Bundle Size y Code Splitting

Componentes Dinámicos

```
javascript

// En page.tsx o layout.tsx
import dynamic from 'next/dynamic';

const Music = dynamic(() => import('@/components/Music'), {
  loading: () => <div>Cargando sección de música...</div>,
  ..ssr: false // Si no necesita SSR
});

const MediaModal = dynamic(() => import('@/components/MediaModal'), {
  ..ssr: false
});
```

2.3 Memoria y Event Listeners

Fix para CustomCursor.tsx

```
javascript
```

```
useEffect(() => {
.. const interactiveElements = document.querySelectorAll('...');

.. // Crear funciones nombradas para poder removerlas
.. const handleEnter = () => cursor.classList.add('hover');
.. const handleLeave = () => cursor.classList.remove('hover');

.. interactiveElements.forEach(el => {
..   el.addEventListener('mouseenter', handleEnter);
..   el.addEventListener('mouseleave', handleLeave);
.. });

.. return () => {
..   // Limpiar correctamente
..   interactiveElements.forEach(el => {
..     el.removeEventListener('mouseenter', handleEnter);
..     el.removeEventListener('mouseleave', handleLeave);
..   });
.. };
}, []);
```

🚀 3. CONTINUACIÓN DE DESARROLLO

3.1 Estructura de Datos Centralizada

Crear **/src/data/portfolio.config.ts**

typescript

```
export const portfolioConfig = {
  personal: {
    name: 'Eduardo Téllez Valverde',
    title: 'Graphic content creator for data analysis',
    email: 'eduardotelv@gmail.com',
    phone: '525545321487',
    location: 'México'
  },
  ...
  projects: [
    {
      id: 1,
      title: 'Dashboard de Ventas Interactivo',
      description: '...',
      image: '/projects/dashboard.jpg',
      video: '/projects/dashboard-demo.mp4',
      tags: ['Tableau', 'Python', 'SQL'],
      links: {
        demo: 'https://...!',
        github: 'https://github.com/...'
      }
    }
  ],
  ...
  // ... más proyectos
},
...
music: {
  bands: [
    {
      name: 'Sahkil',
      role: 'Bajista',
      description: '...',
      spotify: 'https://open.spotify.com/...',
      social: {
        facebook: 'https://facebook.com/sahkilmexico',
        instagram: 'https://instagram.com/sahkilband'
      }
    }
  ]
}
};
```

3.2 Sistema de Formulario de Contacto

Implementación con API Route de Next.js

```
javascript

// /src/app/api/contact/route.ts
export async function POST(request: Request) {
  const body = await request.json();

  // Validación
  if (!body.email || !body.message) {
    return Response.json(
      { error: 'Campos requeridos faltantes' },
      { status: 400 }
    );
  }

  // Aquí integrar con servicio de email
  // Opciones: Resend, SendGrid, o EmailJS

  return Response.json({ success: true });
}
```

3.3 SEO y Metadata

En **/src/app/layout.tsx**

```
typescript
```

```
export const metadata: Metadata = {
  title: 'Eduardo Téllez - Data Visualization Portfolio',
  description: 'Transformo datos complejos en narrativas visuales',
  keywords: ['data visualization', 'dashboard', 'analytics'],
  openGraph: {
    title: 'Eduardo Téllez Portfolio',
    description: '...',
    images: ['/og-image.jpg'],
    locale: 'es_MX',
    type: 'website'
  },
  robots: {
    index: true,
    follow: true,
  }
};
```

3.4 Monitoreo y Analytics

Implementar Web Vitals

```
javascript
// /src/app/components/Analytics.tsx
'use client';

import { useReportWebVitals } from 'next/web-vitals';

export function Analytics() {
  useReportWebVitals((metric) => {
    // Enviar a tu servicio de analytics
    console.log(metric);
  });
}

return null;
}
```

4. CHECKLIST DE IMPLEMENTACIÓN

Fase 1: Responsividad (1-2 días)

- Implementar sistema de breakpoints global

- Ajustar Header para todos los dispositivos
- Optimizar Hero section para móvil
- Corregir grids de Projects e Interests
- Desactivar CustomCursor en móvil
- Probar en dispositivos reales

Fase 2: Performance (2-3 días)

- Implementar lazy loading para videos
- Optimizar todas las imágenes con next/image
- Configurar code splitting para componentes pesados
- Corregir memory leaks en event listeners
- Medir y optimizar Web Vitals

Fase 3: Contenido y Features (3-5 días)

- Crear archivo de configuración central
- Implementar formulario de contacto funcional
- Añadir contenido real (proyectos, media)
- Configurar SEO y metadata
- Implementar analytics básico

Fase 4: Pulido Final (2-3 días)

- Testing cross-browser
- Añadir loading states y error boundaries
- Documentar componentes
- Preparar para deployment
- Configurar GitHub Pages correctamente

5. CONFIGURACIÓN TÉCNICA RECOMENDADA

Variables de Entorno (.env.local)

```
bash
```

```
NEXT_PUBLIC_BASE_PATH=/Portfolio  
NEXT_PUBLIC_GA_ID=G-XXXXXXXXXXXX  
NEXT_PUBLIC_EMAIL_SERVICE_ID=service_xxxxx
```

Scripts de Package.json Actualizados

```
json

{
  "scripts": {
    "dev": "next dev",
    "build": "next build && next export",
    "start": "next start",
    "lint": "next lint",
    "test": "jest",
    "analyze": "ANALYZE=true next build"
  }
}
```

Next.config.js para GitHub Pages

```
javascript

/** @type {import('next').NextConfig} */
const nextConfig = {
  output: 'export',
  basePath: '/Portfolio',
  images: {
    unoptimized: true // Para GitHub Pages
  }
}

module.exports = nextConfig
```

6. MÉTRICAS DE ÉXITO

Performance Goals

- **LCP** (Largest Contentful Paint): < 2.5s
- **FID** (First Input Delay): < 100ms
- **CLS** (Cumulative Layout Shift): < 0.1
- **Lighthouse Score**: > 90

Responsividad

- Funcional en dispositivos desde 320px hasta 4K

- Sin scroll horizontal en ningún breakpoint
- Touch-friendly en móvil (min 44x44px para elementos clickeables)

User Experience

- Tiempo de carga inicial < 3s
 - Smooth scrolling en todas las secciones
 - Animaciones a 60fps
 - Zero errores en consola
-

🔍 7. PROBLEMAS ESPECÍFICOS A RESOLVER

Encoding UTF-8

```
html
<!-- En el head del documento -->
<meta charset="UTF-8">
```

Fix para caracteres especiales en archivos

- Guardar todos los archivos como UTF-8
- Buscar y reemplazar: "Ã" → "í", "Ã³" → "ó", etc.

Consistencia en Paths

```
javascript
// Crear un helper
// /src/utils/paths.ts
export const getAssetPath = (path: string) => {
  const basePath = process.env.NEXT_PUBLIC_BASE_PATH || "";
  return `${basePath}${path.startsWith("/") ? path : `/${path}`}`;
}

// Uso en componentes
import { getAssetPath } from '@/utils/paths';
<img src={getAssetPath('/images/hero.jpg')} />
```

NOTAS FINALES

Este documento está diseñado para ser implementado de manera progresiva. No intentes hacer todo de una vez. La prioridad es:

1. **Responsividad funcional** - El sitio debe verse bien en todos los dispositivos
2. **Performance básica** - Carga rápida y sin problemas de memoria
3. **Contenido real** - Reemplazar todos los placeholders
4. **Polish** - Detalles finales y optimizaciones

Cada fase construye sobre la anterior. No te saltes pasos y prueba exhaustivamente antes de pasar a la siguiente fase.

Recuerda: Un portfolio es tu carta de presentación. Mejor tener pocas features bien implementadas que muchas a medias.