

```

# PROMPT COMPLETO: OPTIMIZACIÓN DE ACCESIBILIDAD GRUPOTRANSERVICA.COM
## Objetivo: Llevar el puntaje de 88/100 a 100/100
---
## 🎯 MISIÓN CRÍTICA
Necesito que implementes todas las optimizaciones de accesibilidad necesarias
para llevar el sitio web grupotranservica.com de un puntaje de 88/100 a
100/100 en Google PageSpeed Insights - Accessibility Audit.
---
## 📊 SITUACIÓN ACTUAL
### Score Actual
- Accesibilidad: 88/100
- Objetivo: 100/100
### Problemas Identificados
#### 🔴 CRÍTICO 1: Botones sin nombres accesibles
Elementos afectados: 4 botones
```html
<!-- INCORRECTO - Sin aria-label -->
<button class="w-3 h-3 rounded-full transition-all duration-300 bg-[#155d29]
scale-125">
</button>
<button class="w-3 h-3 rounded-full transition-all duration-300 bg-gray-300
hover:bg-gray...">
</button>
<button class="p-2 rounded-full bg-white border border-gray-200
hover:border-[#155d29] ho...">
</button>
```
Impacto: Lectores de pantalla dicen "botón" sin contexto → Usuarios ciegos no
saben qué hace el botón.
---
#### 🟡 IMPORTANTE 2: Contraste insuficiente
Elementos afectados: 2 elementos
```html
<!-- Contraste inadecuado -->
<button class="bg-transervica-green text-white px-6 py-2 rounded-md text-sm
font-semibold..."
  aria-label="Solicitar cotización gratuita">
  QUOTE
</button>
<span class="text-white px-3 py-1 rounded-full text-sm font-medium bg-[#3ea30f]">
  280 tons
</span>
```
Ratio actual: < 4.5:1
Ratio requerido: ≥ 4.5:1 (texto normal) o ≥ 3:1 (texto grande)
---
#### 🟡 IMPORTANTE 3: Áreas táctiles pequeñas
Elementos afectados: 17 botones de navegación
```html
<!-- Muy pequeño: 8x8px (w-2 h-2 = 8px) -->
<button class="h-2 rounded-full transition-all duration-300 bg-gray-300 w-2
hover:bg-gray..."
  aria-label="Go to equipment 2">
</button>
<!-- También pequeño: 12x12px (w-3 h-3 = 12px) -->
<button class="w-3 h-3 rounded-full transition-all duration-300 bg-[#155d29]
scale-125">
</button>
```

```

```

**Tamaño actual:** 8-12px
**Tamaño mínimo recomendado:** 44x44px (48x48px ideal)
---
## 🛠️ SOLUCIONES DETALLADAS
### SOLUCIÓN 1: Agregar nombres accesibles a TODOS los botones
#### A. Botones de navegación de carrusel (dots)
**ANTES (Incorrecto):**
```html
<button class="w-3 h-3 rounded-full transition-all duration-300 bg-[#155d29]
scale-125">
</button>
```
**DESPUÉS (Correcto):**
```html
<button
  class="w-3 h-3 rounded-full transition-all duration-300 bg-[#155d29] scale-125"
  aria-label="Ir a slide 1 de equipos destacados"
  aria-current="true"
>
  <span class="sr-only">Slide 1 (activo)</span>
</button>
<button
  class="w-3 h-3 rounded-full transition-all duration-300 bg-gray-300
hover:bg-gray-500"
  aria-label="Ir a slide 2 de equipos destacados"
  aria-current="false"
>
  <span class="sr-only">Slide 2</span>
</button>
```
#### B. Botones de navegación anterior/siguiente
**ANTES (Incorrecto):**
```html
<button class="p-2 rounded-full bg-white border border-gray-200
hover:border-[#155d29] ho...">
  <!-- Solo icono sin texto -->
</button>
```
**DESPUÉS (Correcto):**
```html
<!-- Botón anterior -->
<button
  class="p-2 rounded-full bg-white border border-gray-200 hover:border-[#155d29]
hover:bg-gray-50 transition-all"
  aria-label="Equipo anterior"
  type="button"
>
  <svg class="w-5 h-5 text-gray-700" aria-hidden="true" fill="none" viewBox="0 0 24
24" stroke="currentColor">
    <path stroke-linecap="round" stroke-linejoin="round" stroke-width="2" d="M15
19 17 7 7" />
  </svg>
  <span class="sr-only">Ir al equipo anterior</span>
</button>
<!-- Botón siguiente -->
<button
  class="p-2 rounded-full bg-white border border-gray-200 hover:border-[#155d29]
hover:bg-gray-50 transition-all"
  aria-label="Equipo siguiente"

```

```

type="button"
>
<svg class="w-5 h-5 text-gray-700" aria-hidden="true" fill="none" viewBox="0 0 24
24" stroke="currentColor">
  <path stroke-linecap="round" stroke-linejoin="round" stroke-width="2" d="M9 5l7
7-7 7" />
</svg>
<span class="sr-only">Ir al equipo siguiente</span>
</button>
```

#### C. Clase de utilidad para screen readers
**Agregar a tu CSS global:**
```css
/* Screen reader only - Oculta visualmente pero accesible */
.sr-only {
  position: absolute;
  width: 1px;
  height: 1px;
  padding: 0;
  margin: -1px;
  overflow: hidden;
  clip: rect(0, 0, 0, 0);
  white-space: nowrap;
  border-width: 0;
}
/* Útil para focus visible */
.sr-only-focusable:focus {
  position: static;
  width: auto;
  height: auto;
  padding: inherit;
  margin: inherit;
  overflow: visible;
  clip: auto;
  white-space: normal;
}
```

---

#### SOLUCIÓN 2: Mejorar contraste de colores
#### A. Analizar colores actuales
```javascript
// Color actual problemático
const currentGreen = '#3ea30f'; // bg-[#3ea30f]
const currentDarkGreen = '#155d29'; // bg-[#155d29]
const white = 'ffffff';
// Calcular contraste (usar herramienta: https://contrast-ratio.com/)
// #3ea30f sobre blanco: 3.2:1 ❌ (necesita 4.5:1)
// #155d29 sobre blanco: 7.8:1 ✅
```

#### B. Ajustar colores para mejor contraste
**OPCIÓN 1: Oscurecer el verde claro**
```css
/* ANTES */
.bg-\[#3ea30f\] {
  background-color: #3ea30f; /* Ratio: 3.2:1 ❌ */
}
/* DESPUÉS - Verde más oscuro */
.bg-transervica-green-accessible {
  background-color: #2d8a0b; /* Ratio: 4.6:1 ✅ */
}

```

```

}
...

**OPCIÓN 2: Aumentar el tamaño del texto**
```html
<!-- Si el texto es ≥18px bold o ≥24px regular, solo necesita 3:1 -->
<span class="text-white px-3 py-1 rounded-full text-base font-bold bg-[#3ea30f]">
  280 tons
</span>
<!-- 18px bold + 3.2:1 = PASA ✅ -->
```

**OPCIÓN 3: Usar el verde oscuro que ya existe**
```html
<!-- Usar #155d29 que ya tiene buen contraste -->
<span class="text-white px-3 py-1 rounded-full text-sm font-medium bg-[#155d29]">
  280 tons
</span>
```

#### C. Recomendación para el botón QUOTE
**ANTES:**
```html
<button class="bg-transervica-green text-white px-6 py-2 rounded-md text-sm
font-semibold"
  aria-label="Solicitar cotización gratuita">
  QUOTE
</button>
```

**DESPUÉS - Solución 1 (Mejor contraste):**
```html
<button
  class="bg-[#155d29] text-white px-6 py-2 rounded-md text-sm font-semibold
hover:bg-[#0f4a1f] transition-colors focus:outline-none focus:ring-2
focus:ring-[#155d29] focus:ring-offset-2"
  aria-label="Solicitar cotización gratuita"
  type="button"
>
  QUOTE
</button>
```

**DESPUÉS - Solución 2 (Texto más grande):**
```html
<button
  class="bg-transervica-green text-white px-6 py-3 rounded-md text-base font-bold
hover:opacity-90 transition-opacity focus:outline-none focus:ring-2
focus:ring-transervica-green focus:ring-offset-2"
  aria-label="Solicitar cotización gratuita"
  type="button"
>
  QUOTE
</button>
```

---
#### SOLUCIÓN 3: Aumentar tamaño de áreas táctiles
#### A. Incrementar tamaño mínimo a 44x44px
**ANTES (8x8px - Muy pequeño):**
```html
<button class="h-2 rounded-full transition-all duration-300 bg-gray-300 w-2
hover:bg-gray-500"
  aria-label="Go to equipment 2">
</button>

```

```

```
**DESPUÉS (44x44px - Accesible):**
```html
<button
  class="w-11 h-11 rounded-full transition-all duration-300 bg-gray-300
  hover:bg-gray-500 focus:bg-gray-500 flex items-center justify-center
  focus:outline-none focus:ring-2 focus:ring-[#155d29] focus:ring-offset-2"
  aria-label="Ir al equipo 2"
  type="button"
>
  <!-- Indicador visual dentro del botón más grande -->
  <span class="w-2 h-2 rounded-full bg-current"></span>
</button>
```

**ALTERNATIVA - Mantener visual pequeño pero área táctil grande:**
```html
<button
  class="relative w-11 h-11 rounded-full transition-all duration-300
  hover:bg-gray-100 flex items-center justify-center focus:outline-none focus:ring-2
  focus:ring-[#155d29] focus:ring-offset-2"
  aria-label="Ir al equipo 2"
  type="button"
>
  <!-- El punto visual sigue siendo pequeño -->
  <span class="w-2 h-2 rounded-full bg-gray-300 transition-colors"></span>
</button>
<button
  class="relative w-11 h-11 rounded-full transition-all duration-300
  hover:bg-gray-100 flex items-center justify-center focus:outline-none focus:ring-2
  focus:ring-[#155d29] focus:ring-offset-2"
  aria-label="Ir al equipo 1 (activo)"
  aria-current="true"
  type="button"
>
  <!-- Punto activo -->
  <span class="w-2 h-2 rounded-full bg-[#155d29] scale-125 transition-all"></span>
</button>
```

#### B. Para botones de carousel (dots) - Componente completo
**Implementación moderna:**
```tsx
// CarouselDots.tsx
interface CarouselDotsProps {
  totalSlides: number;
  currentSlide: number;
  onSlideChange: (index: number) => void;
  ariaLabel?: string;
}

export default function CarouselDots({
  totalSlides,
  currentSlide,
  onSlideChange,
  ariaLabel = "Navegación de carrusel"
}: CarouselDotsProps) {
  return (
    <div
      className="flex gap-2 justify-center items-center"
      role="group"
      aria-label={ariaLabel}
    >

```

```

>
(Array.from({ length: totalSlides })).map((_, index) => {
  const isActive = index === currentSlide;

  return (
    <button
      key={index}
      onClick={() => onSlideChange(index)}
      className={`
        relative w-11 h-11 rounded-full transition-all duration-300
        flex items-center justify-center
        hover:bg-gray-100
        focus:outline-none focus:ring-2 focus:ring-[#155d29]
focus:ring-offset-2
        ${isActive ? 'bg-gray-50' : ''}
      `}
      aria-label={`Ir al equipo ${index + 1}`}
      aria-current={isActive}
      type="button"
    >
      {/* Indicador visual pequeño */}
      <span
        className={`
          w-2 h-2 rounded-full transition-all duration-300
          ${isActive
            ? 'bg-[#155d29] scale-125'
            : 'bg-gray-300 hover:bg-gray-400'}
        `}
        aria-hidden="true"
      />

      {/* Texto para screen readers */}
      <span className="sr-only">
        {isActive ? `Equipo ${index + 1} (activo)` : `Equipo ${index + 1}`}
      </span>
    </button>
  );
}
)}
</div>
);
}
...
**Uso:**
```tsx
<CarouselDots
  totalSlides={6}
  currentSlide={currentIndex}
  onSlideChange={setCurrentIndex}
  ariaLabel="Navegación de equipos destacados"
/>
...
---
### SOLUCIÓN 4: Focus States (Bonus - Mejora UX)
Aunque no está en el reporte, agregar estados de focus claros es crítico para
accesibilidad de teclado.
```css
/* Focus states para todos los elementos interactivos */
button:focus-visible,

```

```

a:focus-visible,
input:focus-visible,
select:focus-visible,
textarea:focus-visible {
  outline: 2px solid #155d29;
  outline-offset: 2px;
}
/* Focus ring con Tailwind */
.focus-ring {
  @apply focus:outline-none focus:ring-2 focus:ring-[#155d29] focus:ring-offset-2;
}
/* Usar en componentes */
<button className="... focus-ring">
  Click me
</button>
```

---
## 📋 CHECKLIST DE IMPLEMENTACIÓN
### FASE 1: Nombres Accesibles en Botones
- [ ] **Identificar todos los botones sin aria-label**
  ```bash
  # Buscar en código
  grep -r '<button' --include="*.tsx" --include="*.jsx" | grep -v 'aria-label'
  ```
- [ ] **Agregar aria-label a botones de carousel dots**
  - [ ] Botones de navegación (dots)
  - [ ] Estado activo con `aria-current="true"`
  - [ ] Texto descriptivo en `https://contrast-ratio.com/
  - [ ] Verificar #3ea30f sobre blanco
  - [ ] Verificar otros colores verdes
- [ ] **Ajustar colores**
  - [ ] OPCIÓN A: Oscurecer verde a #2d8a0b
  - [ ] OPCIÓN B: Aumentar tamaño de texto
  - [ ] OPCIÓN C: Usar verde oscuro #155d29
- [ ] **Actualizar botón QUOTE**
  - [ ] Cambiar color o tamaño
  - [ ] Verificar contraste ≥4.5:1
- [ ] **Actualizar badge "280 tons"**
  - [ ] Cambiar color o tamaño
  - [ ] Verificar contraste ≥4.5:1
### FASE 3: Áreas Táctiles
- [ ] **Aumentar tamaño de botones carousel**
  - [ ] Cambiar de `w-2 h-2` (8px) a `w-11 h-11` (44px)
  - [ ] Mantener indicador visual pequeño dentro
  - [ ] Agregar hover states
- [ ] **Aumentar tamaño de dots de navegación**
  - [ ] Cambiar de `w-3 h-3` (12px) a `w-11 h-11` (44px)
  - [ ] Centrar punto visual dentro del botón
  - [ ] Agregar focus ring
- [ ] **Crear componente CarouselDots reutilizable**
- [ ] Implementar con TypeScript

```

```

- [ ] Props para total slides, slide actual, callback
- [ ] Accesibilidad incluida
### FASE 4: Validación
- [ ] **Testing local**
  - [ ] Probar con lector de pantalla (NVDA/JAWS/VoiceOver)
  - [ ] Navegar solo con teclado (Tab, Enter, Space)
  - [ ] Verificar contraste con DevTools
- [ ] **Deploy a staging**
  - [ ] Build y deploy
  - [ ] PageSpeed Insights en staging
  - [ ] Objetivo: 100/100 en Accessibility
- [ ] **Producción**
  - [ ] Deploy final
  - [ ] PageSpeed Insights en producción
  - [ ] Verificar 100/100
---
## 🎯 RESULTADOS ESPERADOS
### Antes vs Después
Auditoría	Antes	Después
**Accessibility Score**	88/100	**100/100** ✅
Botones sin nombres	4 errores	0 errores ✅
Contraste insuficiente	2 errores	0 errores ✅
Áreas táctiles pequeñas	17 errores	0 errores ✅
### Impacto
✅ **Usuarios con discapacidad visual** pueden usar lectores de pantalla
✅ **Usuarios con movilidad reducida** pueden tocar botones fácilmente
✅ **Usuarios con discapacidad visual** leen texto con contraste adecuado
✅ **Cumplimiento WCAG 2.1 AA** (Web Content Accessibility Guidelines)
✅ **Mejor SEO** (Google considera accesibilidad)
✅ **Menor riesgo legal** (ADA compliance)
---
## 🔍 TESTING CON LECTORES DE PANTALLA
### Windows - NVDA (Gratis)
```
1. Descargar: https://www.nvaccess.org/download/
2. Instalar y abrir NVDA
3. Navegar a grupotranservica.com
4. Usar Tab para navegar
5. NVDA debe leer "Ir al equipo 1" en lugar de solo "botón"
```
### Mac - VoiceOver (Incluido)
```
1. Cmd + F5 para activar VoiceOver
2. Navegar a grupotranservica.com
3. Usar Tab para navegar
4. VoiceOver debe leer nombres descriptivos
```
### Chrome DevTools - Accessibility Tree
```
1. Abrir DevTools (F12)
2. Elements tab
3. Accessibility panel (abajo)
4. Verificar que cada botón tenga "Name"
```
---
## 📦 CÓDIGO COMPLETO DE EJEMPLO
Ver archivos adjuntos:
- `components/CarouselDots.accessible.tsx`

```



```

- `components/CarouselNavButtons.accessible.tsx`
- `styles/accessibility.css`
- `utils/contrast-checker.ts`
---
## ✅ VALIDACIÓN FINAL
### Herramientas de Testing
1. **PageSpeed Insights**
  - https://pagespeed.web.dev/
  - Verificar Accessibility: 100/100
2. **axe DevTools** (Extension de Chrome)
  - https://chrome.google.com/webstore/detail/axe-devtools
  - Ejecutar análisis automático
  - 0 errores
3. **WAVE** (WebAIM)
  - https://wave.webaim.org/
  - Análisis visual de accesibilidad
  - 0 errores
4. **Lighthouse en DevTools**
  - F12 → Lighthouse tab
  - Run accessibility audit
  - Score: 100
---
## 🎓 MEJORES PRÁCTICAS APRENDIDAS
### ✅ SIEMPRE hacer:
1. **Todos los botones deben tener nombres accesibles**
  - aria-label para botones con solo iconos
  - Texto visible o <span class="sr-only">
2. **Contraste mínimo 4.5:1** para texto normal
  - 3:1 para texto grande (≥18px bold, ≥24px regular)
  - Usar herramientas para verificar
3. **Áreas táctiles mínimas 44x44px**
  - Aunque el elemento visual sea pequeño
  - El área clickable/tappable debe ser grande
4. **Focus states visibles**
  - Nunca outline: none sin alternativa
  - Usar focus:ring de Tailwind
5. **ARIA solo cuando sea necesario**
  - Preferir HTML semántico
  - ARIA complementa, no reemplaza
### ❌ NUNCA hacer:
1. ❌ Botones sin texto accesible
2. ❌ Colores con bajo contraste
3. ❌ Áreas táctiles < 44x44px
4. ❌ outline: none sin alternativa
5. ❌ ARIA incorrecto o redundante
---
**¡Listo para implementar!** 🚀
Sigue el checklist paso a paso y alcanzarás 100/100 en Accessibility.

```