


We are having a problem billing your account. Please [update your payment method](#) or call your payment provider for details on why the transaction failed.

You can [contact support](#) with any questions.



AsahioO docs: add DESIGN_SYSTEM.md full frontend design guide for reuse in ra...

⋮

✓

3976d20 · 10 minutes ago 🕒

682 lines (535 loc) · 22.5 KB

Preview Code Blame

📄

👤

Raw

📄

📄

✎

⌵

⋮

Sistema de Diseño Frontend — Dajosh Whatshome

Guía completa para replicar la identidad visual y los patrones de interfaz del Catálogo en cualquier otro proyecto (ej. el proyecto de Rifas). Todo lo que está aquí está extraído del código real.

1. Identidad Visual — La Filosofía del Diseño

El diseño del catálogo sigue el estilo "**Lujo Accesible**": se ve sofisticado y premium, pero sin ser frío ni corporativo. Las decisiones visuales clave son:

- **Fondos cálidos (no blanco puro):** El fondo es #F7F5F2 , un blanco marfil que se siente acogedor.
- **Texto casi negro (no negro puro):** #1A1A1A en lugar de #000000 — menos contraste agresivo, más elegante.
- **Un solo color de acento — dorado:** #C8A96E . Se usa con moderación: solo en lo más importante (botones primarios, ítems activos, highlights). Nunca como color de fondo

grande.

- **Tipografía mixta:** Titulares en serif (Playfair Display) mezclada con cuerpo en sans-serif (DM Sans). Esta combinación editorial es la firma más reconocible del diseño.
- **Sin sombras pesadas:** Las sombras (shadow-sm , shadow-md) son escuetas. El énfasis se logra con bordes y cambios de fondo, no con elevación.

2. Tokens de Color

Definidos en `src/app/globals.css` con el bloque `@theme` de Tailwind v4. Para replicarlos en otro proyecto, usar **exactamente los mismos nombres de token** para que los componentes copiados funcionen sin cambios.

```
/* globals.css – copiar completo en el proyecto nuevo */
@import "tailwindcss";

@theme {
  /* Fondos */
  --color-brand-bg:      #F7F5F2; /* Fondo general, marfil cálido */
  --color-brand-surface: #FFFFFF; /* Tarjetas, modales, sidebar, inputs */

  /* Texto */
  --color-brand-text:    #1A1A1A; /* Texto principal */
  --color-brand-muted:   #8A8A8A; /* Texto secundario, placeholders, labels */

  /* Acento y Énfasis */
  --color-brand-accent:  #C8A96E; /* Dorado – CTA, activos, highlights */
  --color-brand-border:  #E8E4DF; /* Bordes y divisores */

  /* Semánticos */
  --color-brand-sale:    #C84B4B; /* Alertas, precios de oferta, agotados */
  --color-brand-sold:    #D0CCC7; /* Indicador de producto sin stock */

  /* Tipografías */
  --font-serif: var(--font-playfair);
  --font-sans:  var(--font-dm-sans);
}

@layer base {
  body {
    background-color: var(--color-brand-bg);
    color: var(--color-brand-text);
    -webkit-tap-highlight-color: transparent; /* sin parpadeo gris en touch */
  }
}
```

```
@layer utilities {
  /* Scrollbar invisible (chips horizontales, etc.) */
  .no-scrollbar { scrollbar-width: none; }
  .no-scrollbar::-webkit-scrollbar { display: none; }

  /* Safe Area para dispositivos con notch/home bar */
  .safe-bottom { padding-bottom: env(safe-area-inset-bottom, 0px); }
  .safe-top { padding-top: env(safe-area-inset-top, 0px); }
}
```

Usos de cada color

Token	Usos típicos
brand-bg	body , min-h-screen , inputs, chips inactivos
brand-surface	Tarjetas, modales, sidebar, paneles de formulario
brand-text	Títulos, texto de párrafo, botones primarios rellenos
brand-muted	Labels de formulario, texto auxiliar, iconos inactivos
brand-accent	Botón activo, dots de nav activa, borders de énfasis, íconos de acento
brand-border	border de tarjetas, divisores <hr> , separadores de sección
brand-sale	Badge de "agotado", precio rebajado, alertas importantes

3. Tipografía

Las fuentes se cargan desde Google Fonts con `next/font/google` en `src/app/layout.tsx` .
No se deben importar via `<link>` en HTML — `next/font` las optimiza automáticamente.

```
// src/app/layout.tsx
import { Playfair_Display, DM_Sans } from "next/font/google";

const playfair = Playfair_Display({
  subsets: ["latin"],
  variable: "--font-playfair", // ← expone como variable CSS
});

const dmSans = DM_Sans({
  subsets: ["latin"],
  variable: "--font-dm-sans",
});
```

```
// En el body:
<body className={` ${playfair.variable} ${dmSans.variable} font-sans antialiased`
```

Jerarquía tipográfica

Nivel	Clases Tailwind	Uso
H1 de página	font-serif text-3xl md:text-4xl	Título del dashboard, páginas admin
H1 del hero	font-serif text-2xl sm:text-4xl md:text-5xl lg:text-6xl leading-tight	Portada del catálogo
H2 de sección	font-serif text-2xl	Secciones dentro de página
H3 de card/modal	font-serif text-lg font-semibold	Encabezados de tarjetas, modales
Cuerpo normal	text-base leading-relaxed	Párrafos, descripciones
Texto auxiliar	text-sm text-brand-muted	Subtítulos, hints, fechas
Etiquetas de form	text-sm font-medium text-brand-text/90	Labels de inputs
Microcopy	text-xs text-brand-muted	Hints debajo de inputs
Badge/Tag	text-xs font-semibold uppercase tracking-widest	Badges, chips
Nav móvil	text-[10px] font-medium leading-none	Labels en bottom tab bar

Combinación editorial (patrón del hero)

El efecto de firma del diseño: título principal en serif + subtítulo en *italic light* de color muted.

```
<h2 className="font-serif text-4xl text-brand-text">
  Descubre tu estilo{" "}
  <span className="italic font-light text-brand-muted">
    con nuestra selección.
```



```
</span>
</h2>
```

4. Espaciado y Contenedores

Se usa un sistema consistente de contenedores centrados con padding lateral responsivo:

```
// Contenedor estándar (páginas públicas)
<div className="max-w-7xl mx-auto px-4 sm:px-6 lg:px-8">

// Contenedor de páginas admin
<div className="max-w-6xl mx-auto">           // dashboard
<div className="max-w-4xl mx-auto pb-10"> // formularios

// Hero section
<div className="max-w-5xl mx-auto px-4 sm:px-6 lg:px-8 text-center">
```

Sistema de padding de páginas

Contexto	Móvil	Desktop
Layout admin	p-4 pb-24	md:p-10
Sección hero	pt-8 pb-10	md:pt-20 md:pb-24
Paneles de formulario	p-6	md:p-8

pb-24 en móvil deja espacio para el bottom tab bar de 64px (h-16) + padding de seguridad.

5. Componentes Base — Recetas Copiables

5.1 Tarjeta / Panel (card)

El patrón usado en todo el sistema para agrupar contenido:

```
// Tarjeta estándar
<div className="bg-brand-surface border border-brand-border rounded-xl shadow-sm
...
</div>
```

```
// Tarjeta con hover (interactiva)
<div className="bg-brand-surface border border-brand-border rounded-xl shadow-sm
    hover:border-brand-text/30 hover:shadow-md transition-all p-6">
    ...
</div>

// Tarjeta con acento (destacada)
<div className="bg-brand-accent/5 border border-brand-accent/20 rounded-xl p-6
    hover:bg-brand-accent/10 hover:border-brand-accent transition-all">
    ...
</div>

// Métrica KPI (con ícono background)
<div className="bg-brand-surface p-6 rounded-2xl border border-brand-border shadow-sm
    relative overflow-hidden group hover:border-brand-text/20 transition-colors">
    {/* Ícono decorativo de fondo */}
    <div className="absolute top-0 right-0 p-4 opacity-5 group-hover:opacity-10 transition-opacity">
        <SomeIcon className="w-16 h-16" />
    </div>
    <div className="text-brand-muted text-sm font-medium mb-3 uppercase tracking-wide">
        Label
    </div>
    <div className="text-4xl font-serif text-brand-text">42</div>
</div>
```

Valores de border-radius usados:

- rounded-lg — botones, inputs, chips inactivos
- rounded-xl — paneles de formulario, tarjetas de admin
- rounded-2xl — tarjetas métricas dashboard, modales principales
- rounded-full — chips de categoría, pills de badge, avatares, botones circulares

5.2 Botones

```
// Botón primario (relleno oscuro – acción principal)
<button className="bg-brand-text text-brand-surface px-6 py-2.5 rounded-lg font-medium
    hover:bg-black transition-colors shadow-sm">
    Guardar
</button>

// Botón primario con loading
<button disabled={loading} className="bg-brand-text text-brand-surface px-8 py-3
    font-medium hover:bg-black transition-colors">
```



```

disabled:opacity-50 flex items-center gap-1
{loading && <div className="w-4 h-4 border-2 border-brand-surface/30 border-t-l
  Guardar Ajustes
</button>

// Botón de acento (dorado, acciones secundarias importantes)
<button className="bg-brand-accent text-white px-6 py-2.5 rounded-lg font-medium
  hover:bg-brand-accent/90 transition-colors shadow-sm">

  WhatsApp
</button>

// Botón outline (cancelar, acciones neutras)
<button className="px-4 py-2 text-sm font-medium text-brand-muted hover:text-bran
  rounded-lg border border-brand-border hover:border-brand-text,

  Cancelar
</button>

// Botón destructivo (peligro – eliminar)
<button className="bg-red-500 text-white px-4 py-2 text-sm font-medium rounded-l
  hover:bg-red-600 transition-all shadow-sm">

  Eliminar
</button>

// Botón ícono (cuadrado pequeño)
<button className="p-2 text-brand-muted hover:text-brand-text rounded-full
  hover:bg-brand-bg transition-colors">
  <X className="w-4 h-4" />
</button>

// Botón de nav con flecha (action list item)
<Link href="/" className="flex items-center p-5 bg-brand-surface border border-br
  rounded-xl hover:border-brand-text hover:shadow-md tra
  <div className="flex-1">
    <h3 className="font-medium text-brand-text mb-1">Título</h3>
    <p className="text-sm text-brand-muted">Descripción corta</p>
  </div>
  <div className="w-10 h-10 rounded-full bg-brand-bg flex items-center justify-c
    text-brand-muted group-hover:bg-brand-text group-hover:text-br
    <ArrowRight className="w-5 h-5 transition-transform group-hover:translate-x:
  </div>
</Link>

```

5.3 Inputs de Formulario

Todos los inputs siguen el mismo patrón: fondo semi-transparente de `brand-bg`, borde en `brand-border`, foco en `brand-accent`.



```
// Input de texto estándar
<div>
  <label className="block text-sm font-medium mb-1.5 text-brand-text/90">
    Campo
  </label>
  <input
    className="w-full p-2.5 border border-brand-border rounded
      focus:border-brand-accent focus:ring-1 focus:ring-brand-accent
      transition-colors bg-brand-bg/50 outline-none"
  />
  <p className="text-xs text-brand-muted mt-1">Texto de ayuda opcional</p>
</div>

// Input con ícono izquierdo (redes sociales, búsqueda)
<div className="relative">
  <div className="absolute inset-y-0 left-0 pl-3 flex items-center pointer-events-none">
    <SomeIcon className="h-4 w-4 text-brand-muted" />
  </div>
  <input className="w-full pl-9 p-2.5 border border-brand-border rounded
    focus:border-brand-accent focus:ring-1 focus:ring-brand-accent
    transition-colors bg-brand-bg/50 outline-none" />
</div>

// Textarea
<textarea
  rows={3}
  className="w-full p-2.5 border border-brand-border rounded
    focus:border-brand-accent focus:ring-1 focus:ring-brand-accent
    transition-colors bg-brand-bg/50 outline-none resize-none leading-relaxed"
/>
```



Nota importante: Los inputs usan `rounded` (sin `-lg`, `-xl`). Esto es intencional — los botones tienen `rounded-lg` para distinguirse visualmente de los campos.

5.4 Encabezado de Sección con Divisor

Patrón para separar secciones dentro de una card de formulario:

```
<h2 className="text-lg font-serif text-brand-text mb-6 flex items-center gap-2 border-bottom">
  <SomeIcon className="w-5 h-5 text-brand-muted" />
  Nombre de la Sección
</h2>
```



Variante del admin sidebar (divisor centrado con texto):

```
<div className="flex items-center gap-2 px-3">
  <div className="h-px bg-brand-border flex-1" />
  <span className="text-xs font-semibold text-brand-muted tracking-widest uppercase">
    Herramientas
  </span>
  <div className="h-px bg-brand-border flex-1" />
</div>
```



5.5 Badge / Pill / Tag

```
// Badge de acento (destacado – colección nueva, nuevo, etc.)
<div className="inline-flex items-center gap-2 px-3 py-1 rounded-full
  bg-brand-accent/10 text-brand-accent text-xs font-medium
  uppercase tracking-widest">
  <Sparkles className="w-3.5 h-3.5" />
  <span>Nueva Colección</span>
</div>

// Badge de estado (borrador)
<span className="text-xs font-medium px-2 py-0.5 rounded-full bg-brand-muted/10">
  Borrador
</span>

// Badge de agotado/error
<span className="text-xs font-medium px-2 py-0.5 rounded-full bg-brand-sale/10">
  Agotado
</span>
```



5.6 Toast (Notificaciones)

El componente `Toast.tsx` usa un **singleton global**: se puede llamar desde cualquier lugar del código (dentro o fuera de componentes React) con `showToast()`.

Instalación en el nuevo proyecto:

1. Copiar `src/components/ui/Toast.tsx` al nuevo proyecto.
2. Agregar `<ToastContainer />` en el `layout.tsx` raíz (junto al `{children}`).
3. Usar en cualquier archivo:

```
import { showToast } from '@components/ui/Toast';

// Éxito
showToast('¡Rifa creada exitosamente!', 'success');

// Error
showToast('Hubo un problema al guardar.', 'error');

// Información
showToast('Los cambios se guardarán en borrador.', 'info');
```

Estilos por tipo:

Tipo	Background	Borde	Ícono
success	bg-emerald-50	border-emerald-200	CheckCircle verde
error	bg-red-50	border-red-200	AlertTriangle rojo
info	bg-brand-accent/10	border-brand-accent/20	Info dorado

Las notificaciones se auto-descartan a los **4000ms** y pueden cerarse manualmente. Se apilan verticalmente desde la esquina superior derecha.

5.7 Modal de Confirmación

El componente `ConfirmModal.tsx` reemplaza el `window.confirm()` nativo.

```
// Uso en cualquier componente
const [confirm, setConfirm] = useState(false);

<ConfirmModal
  open={confirm}
  title="¿Eliminar esta rifa?"
  message="Esta acción no se puede deshacer. La rifa y todos sus datos serán eli
  confirmLabel="Sí, eliminar"
  cancelLabel="Cancelar"
  danger={true} // ← pone el botón de confirmar en rojo
  onConfirm={() => { /* acción */ setConfirm(false); }}
  onCancel={() => setConfirm(false)}
/>
```



Sin danger : el botón de confirmar usa `bg-brand-text text-brand-surface` .

Con danger : el botón usa `bg-red-500 text-white hover:bg-red-600` .

El ícono del modal también cambia: dorado sin danger, rojo con danger.

5.8 Chips de Filtro (Categorías)

El patrón de filtros horizontales con scroll invisible. Se puede reutilizar para cualquier lista de filtros (estados de rifa, tipos de premio, etc.).

```
// Chip activo: fondo brand-text animado con layoutId
// Chip inactivo: borde brand-border, fondo brand-surface

<div className="w-full overflow-x-auto no-scrollbar py-4 border-b border-brand-b
      bg-brand-bg/95 backdrop-blur-md sticky top-14 sm:top-20 z-10">
  <div className="max-w-7xl mx-auto px-4 sm:px-6 lg:px-8">
    <div className="flex space-x-2.5 pb-2">
      {items.map((item) => {
        const isActive = activeId === item.id;
        return (
          <motion.button
            key={item.id}
            whileHover={{ scale: 1.05 }}
            whileTap={{ scale: 0.95 }}
            onClick={() => setActive(item.id)}
            className={clsx(
              "relative whitespace-nowrap px-5 py-2 rounded-full text-sm font-me
              isActive ? "text-brand-surface shadow-md" : "bg-brand-surface text
            )}
          >
            {isActive && (
              <motion.div
                layoutId="activeCategoryBg" // ← anima el fondo entre chips
                className="absolute inset-0 bg-brand-text rounded-full"
                transition={{ type: "spring", stiffness: 300, damping: 30 }}
              />
            )}
            <span className={clsx("relative z-10", isActive && "font-semibold")}>
              {item.nombre}
            </span>
          </motion.button>
        );
      })}
    </div>
```



```
</div>  
</div>
```

6. Patrones de Animación (Framer Motion)

6.1 Entrada escalonada ("stagger") — para listas de cards/secciones

El patrón más usado en el proyecto: las cards aparecen en cascada con delay entre sí.

```
// 1. Definir variantes  
const containerVariants = {  
  hidden: { opacity: 0 },  
  show: {  
    opacity: 1,  
    transition: { staggerChildren: 0.1 }, // 100ms entre cada hijo  
  },  
};  
  
const itemVariants = {  
  hidden: { opacity: 0, y: 20 },  
  show: {  
    opacity: 1,  
    y: 0,  
    transition: { type: "spring" as const, stiffness: 300, damping: 24 },  
  },  
};  
  
// 2. Usarlos en JSX  
<motion.div variants={containerVariants} initial="hidden" animate="show">  
  <motion.div variants={itemVariants}> Card 1 </motion.div>  
  <motion.div variants={itemVariants}> Card 2 </motion.div>  
  <motion.div variants={itemVariants}> Card 3 </motion.div>  
</motion.div>
```



6.2 Entrada de modal / bottom sheet

```
// Modal centrado (desktop)  
<motion.div  
  initial={{ opacity: 0, scale: 0.9, y: 20 }}  
  animate={{ opacity: 1, scale: 1, y: 0 }}  
  exit={{ opacity: 0, scale: 0.9, y: 20 }}  
  transition={{ type: "spring", stiffness: 400, damping: 30 }}>
```



```

    className="bg-brand-surface rounded-2xl shadow-2xl border border-brand-border"
  >

  // Bottom sheet (móvil)
  <motion.div
    initial={{ y: "100%" }}
    animate={{ y: 0 }}
    exit={{ y: "100%" }}
    transition={{ type: "spring", damping: 30, stiffness: 300 }}
    className="fixed bottom-0 left-0 right-0 bg-brand-surface rounded-t-2xl safe-b
  >

```

6.3 Backdrop (fondo oscuro de modales)

```

<motion.div
  initial={{ opacity: 0 }}
  animate={{ opacity: 1 }}
  exit={{ opacity: 0 }}
  onClick={onClose}
  className="fixed inset-0 bg-black/30 backdrop-blur-sm z-[60]"
/>

```



6.4 Notificación Toast (entrada desde la derecha)

```

<motion.div
  initial={{ opacity: 0, x: 80, scale: 0.95 }}
  animate={{ opacity: 1, x: 0, scale: 1 }}
  exit={{ opacity: 0, x: 80, scale: 0.95 }}
  transition={{ type: "spring", stiffness: 400, damping: 30 }}
>

```



6.5 Indicador activo "píldora deslizante" (layoutId)

El efecto de que la píldora de fondo se *desliza* de un elemento activo al siguiente, sin código manual de posición:

```

// Funciona para tabs, chips de categoría, nav del sidebar
{isActive && (
  <motion.div
    layoutId="uniqueIdPerGroup" // único por grupo de elementos
    className="absolute inset-0 bg-brand-text rounded-full" // o rounded-lg
    transition={{ type: "spring", stiffness: 350, damping: 30 }}
  >

```



```

  />
}}

```

Grupos en el proyecto:

- `layoutId="activeCategoryBg"` — chips de categoría del catálogo
- `layoutId="sidebarActive"` — nav del sidebar en desktop
- `layoutId="bottomNavDot"` — punto debajo del tab activo en móvil

6.6 Feedback táctil en botones (`whileTap`)

```

// Chips y botones que se presionan
<motion.button whileHover={{ scale: 1.05 }} whileTap={{ scale: 0.95 }}>

// Botones de nav (tab bar móvil) — con Tailwind nativo
<button className="... active:scale-90 transition-all">

// Touch rápido en tarjetas
<div className="... active:scale-[0.98] transition-all">

```



6.7 Entrada simple con delay manual

Para elementos que no están en un stagger pero deben entrar uno a uno:

```

<motion.div initial={{ opacity: 0, y: 20 }} animate={{ opacity: 1, y: 0 }} trans:
  Elemento 1
</motion.div>
<motion.div initial={{ opacity: 0, y: 20 }} animate={{ opacity: 1, y: 0 }} trans:
  Elemento 2
</motion.div>
<motion.div initial={{ opacity: 0, y: 20 }} animate={{ opacity: 1, y: 0 }} trans:
  Elemento 3
</motion.div>

```



7. Configuración del Proyecto (para el proyecto nuevo)

package.json — dependencias a instalar

```
npm install framer-motion lucide-react clsx @supabase/ssr @supabase/supabase-js
```



next.config.ts — dominio de imágenes

Si el proyecto nuevo también usa Supabase Storage para imágenes, añadir el dominio en la whitelist:

```
images: {  
  remotePatterns: [  
    {  
      protocol: 'https',  
      hostname: 'qhiufyedablidveileth.supabase.co', // mismo proyecto Supabase  
    }  
  ],  
},
```



tsconfig.json — alias de importación

```
{  
  "compilerOptions": {  
    "paths": {  
      "@/*": [ "./src/*" ]  
    }  
  }  
}
```



8. Patrones de Layout Responsivo

8.1 Mobile-First breakpoints usados

El proyecto usa los breakpoints estándar de Tailwind, pero con preferencia a **cubrir móvil primero**:

```
default → móvil (< 640px)  
sm:      → 640px+ (tablet pequeña, algunos chips y grids)  
md:      → 768px+ (cambio mayor: sidebar vs bottom nav, tabla vs cards)  
lg:      → 1024px+ (grids de 3 columnas)
```



8.2 Grid de productos

```
<div className="grid grid-cols-2 md:grid-cols-3 lg:grid-cols-4 gap-x-2.5 gap-y-5
```



8.3 Layout con sidebar (admin)

```
<div className="min-h-screen flex flex-col md:flex-row bg-brand-bg font-sans">
  <AdminSidebar />
  <main className="flex-1 p-4 pb-24 md:p-10 overflow-auto">
    {children}
  </main>
</div>
```

8.4 Header height

El header tiene altura diferente en móvil y desktop. Los elementos sticky que dependen del header deben usar los mismos valores:

```
// Header en sí
<header className="h-14 sm:h-20">

// Sticky que va debajo del header
<div className="sticky top-14 sm:top-20 z-10">
```

9. Checklist de Implementación para el Proyecto de Rifas

Para aplicar este diseño en el proyecto de rifas, seguir este orden:

- ☐ Copiar `globals.css` (o agregar el bloque `@theme`) con todos los tokens
- ☐ Configurar las fuentes en `layout.tsx` (Playfair + DM Sans con `next/font`)
- ☐ Copiar `Toast.tsx` + agregar `<ToastContainer />` en el layout raíz
- ☐ Copiar `ConfirmModal.tsx` para reemplazar `window.confirm()`
- ☐ Aplicar `bg-brand-bg font-sans` en el body/layout raíz
- ☐ Usar `bg-brand-surface + border brand-border + rounded-xl` en todas las tarjetas
- ☐ Aplicar el patrón de entrada stagger en cualquier lista (rifas, tickets, etc.)
- ☐ En móvil: usar bottom sheet en lugar de modales centrados para acciones táctiles
- ☐ Añadir `safe-bottom` y `safe-top` en elementos fijos que toquen los bordes de la pantalla

- ☐ Usar `no-scrollbar` en cualquier scroll horizontal
- ☐ Asegurarse de que todos los `focus:` en inputs usen `brand-accent`