🚀 QUICK START: IMPLEMENTAR INTERFACES MULTI-VERTICAL

Guía Rápida para Desarrollar las Páginas Frontend

Los modelos de datos ya están creados y migrados a la base de datos. Esta guía te muestra cómo crear las interfaces para cada vertical.

1️⃣ STR - ALQUILERES TURÍSTICOS

Paso 1: Crear página de Listings

```
mkdir -p app/str/listings
```

```tsx
// app/str/listings/page.tsx
'use client';

import { useEffect, useState } from 'react';
import { useSession } from 'next-auth/react';
import { Card, CardContent, CardHeader, CardTitle } from '@/components/ui/card';
import { Button } from '@/components/ui/button';
import { Badge } from '@/components/ui/badge';

export default function STRListingsPage() {
  const { data: session } = useSession();
  const [listings, setListings] = useState([]);

  useEffect(() => {
    fetch('/api/str/listings')
      .then(res => res.json())
      .then(data => setListings(data));
  }, []);

  return (
    <div className="p-6">
      <h1 className="text-3xl font-bold mb-6">Anuncios Turísticos</h1>

      {/* KPIs */}
      <div className="grid grid-cols-4 gap-4 mb-6">
        <Card>
          <CardHeader>
            <CardTitle>Total Listings</CardTitle>
          </CardHeader>
          <CardContent>
            <div className="text-3xl font-bold">{listings.length}</div>
          </CardContent>
        </Card>
        {/* Más KPIs... */}
      </div>

      {/* Lista de listings */}
      <div className="grid grid-cols-3 gap-4">
        {listings.map(listing => (
          <Card key={listing.id}>
            <CardHeader>
              <CardTitle>{listing.titulo}</CardTitle>
              <Badge>{listing.tipoPropiedad}</Badge>
            </CardHeader>
            <CardContent>
              <p>Precio: €{listing.precioPorNoche}/noche</p>
              <p>Ocupación: {listing.tasaOcupacion}%</p>
              <p>Rating: {listing.ratingPromedio}/5</p>
              <Button className="mt-4">Ver Detalles</Button>
            </CardContent>
          </Card>
        ))}
      </div>
    </div>
  );
}
```

Paso 2: Crear API Route

```
// app/api/str/listings/route.ts
import { NextRequest, NextResponse } from 'next/server';
import { getServerSession } from 'next-auth';
import { authOptions } from '@/lib/auth-options';
import { prisma } from '@/lib/db';

export async function GET(request: NextRequest) {
  const session = await getServerSession(authOptions);
  if (!session?.user?.companyId) {
    return NextResponse.json({ error: 'Unauthorized' }, { status: 401 });
  }

  const listings = await prisma.sTRListing.findMany({
    where: {
      companyId: session.user.companyId
    },
    include: {
      unit: { include: { building: true } },
      bookings: true,
      channels: true
    }
  });

  return NextResponse.json(listings);
}

export async function POST(request: NextRequest) {
  const session = await getServerSession(authOptions);
  if (!session?.user?.companyId) {
    return NextResponse.json({ error: 'Unauthorized' }, { status: 401 });
  }

  const data = await request.json();

  const listing = await prisma.sTRListing.create({
    data: {
      ...data,
      companyId: session.user.companyId
    }
  });

  return NextResponse.json(listing, { status: 201 });
}
```

Paso 3: Crear página de Bookings

```
// app/str/bookings/page.tsx
'use client';

import { useState, useEffect } from 'react';
import { Calendar } from '@/components/ui/calendar';

export default function STRBookingsPage() {
  const [bookings, setBookings] = useState([]);
  const [selectedDate, setSelectedDate] = useState(new Date());

  useEffect(() => {
    fetch('/api/str/bookings')
      .then(res => res.json())
      .then(data => setBookings(data));
  }, []);

  return (
    <div className="p-6">
      <h1 className="text-3xl font-bold mb-6">Calendario de Reservas</h1>

      <div className="grid grid-cols-2 gap-6">
        {/* Calendario */}
        <div>
          <Calendar
            mode="single"
            selected={selectedDate}
            onSelect={setSelectedDate}
          />
        </div>

        {/* Lista de bookings del día seleccionado */}
        <div>
          <h2 className="text-xl font-bold mb-4">Reservas del día</h2>
          {bookings
            .filter(b => isSameDay(new Date(b.checkInDate), selectedDate))
            .map(booking => (
              <Card key={booking.id} className="mb-4">
                <CardHeader>
                  <CardTitle>{booking.guestNombre}</CardTitle>
                  <Badge>{booking.canal}</Badge>
                </CardHeader>
                <CardContent>
                  <p>Check-in: {formatDate(booking.checkInDate)}</p>
                  <p>Check-out: {formatDate(booking.checkOutDate)}</p>
                  <p>Huéspedes: {booking.numHuespedes}</p>
                  <p>Total: €{booking.precioTotal}</p>
                </CardContent>
              </Card>
            ))}
        </div>
      </div>
    </div>
  );
}
```

2️⃣ HOUSE FLIPPING

Paso 1: Crear página de Proyectos

```
// app/flipping/projects/page.tsx
'use client';

import { useState, useEffect } from 'react';
import { Card } from '@/components/ui/card';
import { Badge } from '@/components/ui/badge';
import { Progress } from '@/components/ui/progress';

export default function FlippingProjectsPage() {
  const [projects, setProjects] = useState([]);

  useEffect(() => {
    fetch('/api/flipping/projects')
      .then(res => res.json())
      .then(data => setProjects(data));
  }, []);

  const getStatusColor = (status) => {
    const colors = {
      PROSPECTO: 'bg-gray-500',
      ANALISIS: 'bg-blue-500',
      ADQUISICION: 'bg-yellow-500',
      RENOVACION: 'bg-orange-500',
      COMERCIALIZACION: 'bg-green-500',
      VENDIDO: 'bg-green-700',
      CANCELADO: 'bg-red-500'
    };
    return colors[status] || 'bg-gray-500';
  };

  return (
    <div className="p-6">
      <h1 className="text-3xl font-bold mb-6">Proyectos de Inversión</h1>

      {/* Pipeline Kanban */}
      <div className="grid grid-cols-7 gap-4">
        {['PROSPECTO', 'ANALISIS', 'ADQUISICION', 'RENOVACION', 'COMERCIALIZACION', 'V
ENDIDO', 'CANCELADO'].map(status => (
          <div key={status}>
            <h3 className="font-bold mb-4">{status}</h3>
            {projects
              .filter(p => p.estado === status)
              .map(project => (
                <Card key={project.id} className="mb-4 p-4">
                  <h4 className="font-semibold">{project.nombre}</h4>
                  <p className="text-sm text-muted-foreground">{project.direccion}</p>

                  <div className="mt-2">
                    <p className="text-xs">Inversión: €{project.inversionTotal?.toLoc-
aleString()}</p>
                    {project.roiPorcentaje && (
                      <Badge className={project.roiPorcentaje > 10 ? 'bg-green-500' :
'bg-yellow-500'}>
                        ROI: {project.roiPorcentaje}%
                      </Badge>
                    )}
                  </div>

                  {/* Progress bar para renovación */}
                  {status === 'RENOVACION' && (
                    <Progress value={calculateProgress(project)} className="mt-2" />
                  )}
```

```
                </Card>
              ))}
            </div>
          ))}
        </div>
      </div>
    );
}
```

Paso 2: Detalle de Proyecto

```
// app/flipping/projects/[id]/page.tsx
'use client';

import { useState, useEffect } from 'react';
import { useParams } from 'next/navigation';
import { Tabs, TabsContent, TabsList, TabsTrigger } from '@/components/ui/tabs';

export default function FlippingProjectDetailPage() {
  const params = useParams();
  const [project, setProject] = useState(null);

  useEffect(() => {
    fetch(`/api/flipping/projects/${params.id}`)
      .then(res => res.json())
      .then(data => setProject(data));
  }, [params.id]);

  if (!project) return <div>Cargando...</div>;

  return (
    <div className="p-6">
      <h1 className="text-3xl font-bold mb-6">{project.nombre}</h1>

      {/* ROI Summary */}
      <div className="grid grid-cols-4 gap-4 mb-6">
        <Card>
          <CardHeader><CardTitle>Inversión Total</CardTitle></CardHeader>
          <CardContent>€{project.inversionTotal?.toLocaleString()}</CardContent>
        </Card>
        <Card>
          <CardHeader><CardTitle>Beneficio Neto</CardTitle></CardHeader>
          <CardContent>€{project.beneficioNeto?.toLocaleString()}</CardContent>
        </Card>
        <Card>
          <CardHeader><CardTitle>ROI</CardTitle></CardHeader>
          <CardContent className="text-3xl font-bold text-green-600">
            {project.roiPorcentaje}%
          </CardContent>
        </Card>
        <Card>
          <CardHeader><CardTitle>Estado</CardTitle></CardHeader>
          <CardContent><Badge>{project.estado}</Badge></CardContent>
        </Card>
      </div>

      {/* Tabs */}
      <Tabs defaultValue="renovations">
        <TabsList>
          <TabsTrigger value="renovations">Renovaciones</TabsTrigger>
          <TabsTrigger value="expenses">Gastos</TabsTrigger>
          <TabsTrigger value="timeline">Timeline</TabsTrigger>
          <TabsTrigger value="gallery">Galería</TabsTrigger>
        </TabsList>

        <TabsContent value="renovations">
          {/* Lista de renovaciones */}
          {project.renovations.map(reno => (
            <Card key={reno.id} className="mb-4">
              <CardHeader>
                <CardTitle>{reno.categoria}</CardTitle>
                <p>{reno.descripcion}</p>
              </CardHeader>
```

```
              <CardContent>
                <div className="flex justify-between">
                  <span>Presupuestado: €{reno.presupuestado}</span>
                  <span>Real: €{reno.costoReal || 'Pendiente'}</span>
                </div>
                <Progress value={reno.porcentajeAvance} className="mt-2" />
              </CardContent>
            </Card>
          ))}
        </TabsContent>

        <TabsContent value="expenses">
          {/* Lista de gastos */}
          {project.expenses.map(expense => (
            <div key={expense.id} className="flex justify-between border-b py-2">
              <div>
                <p className="font-semibold">{expense.concepto}</p>
                <p className="text-sm text-muted-foreground">{expense.categoria}</p>
              </div>
              <p className="font-bold">€{expense.monto}</p>
            </div>
          ))}
        </TabsContent>

        <TabsContent value="timeline">
          {/* Timeline de hitos */}
          {project.milestones.map(milestone => (
            <div key={milestone.id} className="flex items-center gap-4 mb-4">
              <div className={`w-4 h-4 rounded-full ${milestone.completado ? 'bg-
green-500' : 'bg-gray-300'}`} />
              <div>
                <p className="font-semibold">{milestone.titulo}</p>
                <p className="text-sm">Fecha: {formatDate(milestone.fechaPrevista)}</
p>
              </div>
            </div>
          ))}
        </TabsContent>

        <TabsContent value="gallery">
          {/* Before/After Gallery */}
          <div className="grid grid-cols-2 gap-4">
            <div>
              <h3 className="font-bold mb-2">Antes</h3>
              {project.fotosAntes.map((foto, idx) => (
                <img key={idx} src={foto} className="mb-2 rounded" />
              ))}
            </div>
            <div>
              <h3 className="font-bold mb-2">Después</h3>
              {project.fotosDespues.map((foto, idx) => (
                <img key={idx} src={foto} className="mb-2 rounded" />
              ))}
            </div>
          </div>
        </TabsContent>
      </Tabs>
    </div>
  );
}
```

3️⃣ CONSTRUCCIÓN

Crear página de Proyectos de Construcción

```
// app/construction/projects/page.tsx
'use client';

import { useState, useEffect } from 'react';
import { Badge } from '@/components/ui/badge';

const CONSTRUCTION_PHASES = [
  'PLANIFICACION', 'PERMISOS', 'CIMENTACION', 'ESTRUCTURA',
  'CERRAMIENTOS', 'INSTALACIONES', 'ACABADOS', 'ENTREGA', 'GARANTIA'
];

export default function ConstructionProjectsPage() {
  const [projects, setProjects] = useState([]);

  useEffect(() => {
    fetch('/api/construction/projects')
      .then(res => res.json())
      .then(data => setProjects(data));
  }, []);

  return (
    <div className="p-6">
      <h1 className="text-3xl font-bold mb-6">Proyectos de Construcción</h1>

      <div className="space-y-4">
        {projects.map(project => (
          <Card key={project.id}>
            <CardHeader>
              <div className="flex justify-between">
                <div>
                  <CardTitle>{project.nombre}</CardTitle>
                  <p className="text-sm text-muted-foreground">{project.tipoProyecto}
</p>
                </div>
                <Badge>{project.faseActual}</Badge>
              </div>
            </CardHeader>
            <CardContent>
              {/* Progress Bar de Fases */}
              <div className="flex gap-2 mb-4">
                {CONSTRUCTION_PHASES.map(phase => (
                  <div
                    key={phase}
                    className={`h-2 flex-1 rounded ${
                      CONSTRUCTION_PHASES.indexOf(phase) <= CONSTRUC-
TION_PHASES.indexOf(project.faseActual)
                        ? 'bg-green-500'
                        : 'bg-gray-200'
                    }`}
                  />
                ))}
              </div>

              <div className="grid grid-cols-3 gap-4">
                <div>
                  <p className="text-sm text-muted-foreground">Presupuesto</p>
                  <p className="font-bold">€{pro-
ject.presupuestoTotal.toLocaleString()}</p>
                </div>
                <div>
                  <p className="text-sm text-muted-foreground">Gastado</p>
                  <p className="font-bold">€{project.gastosReales.toLocaleString()}</
```

```
p>
                </div>
                <div>
                  <p className="text-sm text-muted-foreground">Avance</p>
                  <p className="font-bold">{project.porcentajeAvance}%</p>
                </div>
              </div>

              <div className="mt-4 flex gap-2">
                {project.licenciaObra && <Badge>Licencia ✓</Badge>}
                {project.certificadoFinal && <Badge>Certificado Final ✓</Badge>}
                {project.habitabilidad && <Badge>Habitabilidad ✓</Badge>}
              </div>
            </CardContent>
          </Card>
        ))}
      </div>
    </div>
  );
}
```

4️⃣ SERVICIOS PROFESIONALES

Crear página de Proyectos Profesionales

```
// app/professional/projects/page.tsx
'use client';

import { useState, useEffect } from 'react';

const PROJECT_TYPES = {
  PROYECTO_BASICO: 'Proyecto Básico',
  PROYECTO_EJECUCION: 'Proyecto de Ejecución',
  DIRECCION_OBRA: 'Dirección de Obra',
  CERTIFICACION_ENERGETICA: 'Certificación Energética',
  INSPECCION_TECNICA: 'Inspección Técnica',
  TASACION: 'Tasación',
  CONSULTORIA: 'Consultoría'
};

export default function ProfessionalProjectsPage() {
  const [projects, setProjects] = useState([]);

  useEffect(() => {
    fetch('/api/professional/projects')
      .then(res => res.json())
      .then(data => setProjects(data));
  }, []);

  return (
    <div className="p-6">
      <h1 className="text-3xl font-bold mb-6">Proyectos Profesionales</h1>

      <div className="grid grid-cols-3 gap-4">
        {projects.map(project => (
          <Card key={project.id}>
            <CardHeader>
              <CardTitle>{project.titulo}</CardTitle>
              <Badge>{PROJECT_TYPES[project.tipo]}</Badge>
            </CardHeader>
            <CardContent>
              <div className="space-y-2">
                <div>
                  <p className="text-sm text-muted-foreground">Cliente</p>
                  <p className="font-semibold">{project.clienteNombre}</p>
                </div>

                <div>
                  <p className="text-sm text-muted-foreground">Honorarios</p>
                  <p className="font-bold text-lg">€{project.total.toLocaleString()}</p
>
                </div>

                <div>
                  <p className="text-sm text-muted-foreground">Estado</p>
                  <Badge variant={project.estado === 'ENTREGADO' ? 'success' : 'de-
fault'}>
                    {project.estado}
                  </Badge>
                </div>

                <div>
                  <p className="text-sm text-muted-foreground">Avance</p>
                  <Progress value={project.porcentajeAvance} />
                </div>

                <div className="flex gap-2 mt-4">
```

```
                  <Button size="sm">Ver Detalles</Button>
                  <Button size="sm" variant="outline">Entregables</Button>
                </div>
              </div>
            </CardContent>
          </Card>
        ))}
      </div>
    </div>
  );
}
```

🔧 APIS COMUNES

Todas las APIs siguen el mismo patrón:

```
// app/api/[vertical]/[resource]/route.ts
import { NextRequest, NextResponse } from 'next/server';
import { getServerSession } from 'next-auth';
import { authOptions } from '@/lib/auth-options';
import { prisma } from '@/lib/db';

export async function GET(request: NextRequest) {
  const session = await getServerSession(authOptions);
  if (!session?.user?.companyId) {
    return NextResponse.json({ error: 'Unauthorized' }, { status: 401 });
  }

  const data = await prisma.[MODEL].findMany({
    where: {
      companyId: session.user.companyId
    },
    include: {
      // Relations...
    }
  });

  return NextResponse.json(data);
}

export async function POST(request: NextRequest) {
  const session = await getServerSession(authOptions);
  if (!session?.user?.companyId) {
    return NextResponse.json({ error: 'Unauthorized' }, { status: 401 });
  }

  const body = await request.json();

  const record = await prisma.[MODEL].create({
    data: {
      ...body,
      companyId: session.user.companyId
    }
  });

  return NextResponse.json(record, { status: 201 });
}
```

📱 AGREGAR AL SIDEBAR

```
// components/layout/sidebar.tsx

// Agregar al array de navigation items:
const navigationItems = [
  // ... items existentes

  // STR
  { name: 'Listings STR', href: '/str/listings', icon: Hotel, moduloCodigo: 'str_listi
ngs' },
  { name: 'Bookings', href: '/str/bookings', icon: Calendar, moduloCodigo: 'str_bookin
gs' },
  { name: 'Channel Manager', href: '/str/channels', icon: Cloud, moduloCodigo: 'str_ch
annels' },

  // House Flipping
  { name: 'Proyectos Flipping', href: '/flipping/projects', icon: Hammer,
moduloCodigo: 'flipping_projects' },
  { name: 'ROI Calculator', href: '/flipping/roi', icon: Calculator, moduloCodigo: 'fl
ipping_roi' },

  // Construction
  { name: 'Construcción', href: '/construction/projects', icon: Building,
moduloCodigo: 'construction' },
  { name: 'Órdenes de Trabajo', href: '/construction/work-orders', icon: HardHat, mod-
uloCodigo: 'construction_work' },

  // Professional
  { name: 'Proyectos Pro', href: '/professional/projects', icon: Briefcase, modulo-
Codigo: 'professional' },
  { name: 'Clientes', href: '/professional/clients', icon: Users, moduloCodigo: 'pro-
fessional_clients' },
];
```

✅ CHECKLIST DE IMPLEMENTACIÓN

STR Module

- [ ] `/str/listings` - Lista y creación de listings
- [ ] `/str/listings/[id]` - Detalle de listing
- [ ] `/str/bookings` - Calendario de reservas
- [ ] `/str/channels` - Gestión de canales
- [ ] `/str/revenue` - Revenue management
- [ ] `/api/str/listings/route.ts`
- [ ] `/api/str/bookings/route.ts`
- [ ] `/api/str/channels/route.ts`

House Flipping Module

- [ ] `/flipping/projects` - Lista de proyectos
- [ ] `/flipping/projects/[id]` - Detalle de proyecto
- [ ] `/flipping/roi` - Calculadora ROI

- [ ] `/api/flipping/projects/route.ts`
- [ ] `/api/flipping/projects/[id]/route.ts`

Construction Module

- [ ] `/construction/projects` - Lista de proyectos
- [ ] `/construction/projects/[id]` - Detalle
- [ ] `/construction/work-orders` - Órdenes de trabajo
- [ ] `/construction/inspections` - Inspecciones
- [ ] `/api/construction/projects/route.ts`

Professional Module

- [ ] `/professional/projects` - Proyectos
- [ ] `/professional/projects/[id]` - Detalle
- [ ] `/professional/deliverables` - Entregables
- [ ] `/api/professional/projects/route.ts`

🎯 PRIORIDADES

Fase 1 (1-2 semanas):
1. ✅ STR Listings + Bookings (más demandado)
2. ✅ House Flipping Projects + ROI

Fase 2 (1-2 semanas):
3. ✅ Construction Projects
4. ✅ Professional Projects

Fase 3 (2-3 semanas):
5. Integraciones Airbnb/Booking
6. Advanced features

¡Comienza con STR Listings! Es el vertical con mayor demanda.