

# Software Inmobiliario Full Stack

## Objetivo General

Desarrollar un **software inmobiliario completo** con backend en .NET y frontend en React.

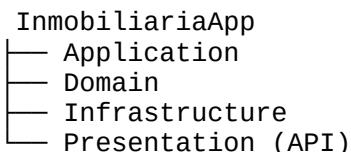
El sistema debe permitir gestionar propiedades, manejar roles de usuario (Administrador y Cliente), proteger los recursos con JWT, y almacenar imágenes en la nube utilizando **Cloudinary** y desplegar ambos proyectos en entornos accesibles públicamente.

---

## Parte 1: Backend (.NET 8 or 9 + EF Core + DDD + JWT)

### Arquitectura requerida

El backend debe estar organizado bajo el enfoque **DDD (Domain Driven Design)**, con al menos las siguientes capas:



### Requerimientos funcionales

#### 1. Autenticación y autorización (JWT):

- Registro e inicio de sesión de usuarios.
- Roles: **Administrador** y **Cliente**.
- Protección de endpoints según rol.

#### 2. Gestión de propiedades (solo para Administrador):

- Crear, actualizar, eliminar y listar propiedades.
- Cada propiedad debe tener:
  - Id
  - Título
  - Descripción
  - Precio
  - Ubicación
  - URL(s) de imagen almacenadas en **Cloudinary**

#### 3. Gestión de clientes:

- El cliente podrá:
  - Registrarse.
  - Iniciar sesión.

- Ver el listado de propiedades disponibles.
- Ver detalles de una propiedad.
- Contactar al propietario o agencia por medio de un envío de correo.

#### **4. Almacenamiento de imágenes:**

- Las imágenes deben subirse a **Cloudinary** mediante una integración desde el backend.
- Guardar en base de datos solo la URL devuelta por Cloudinary.

#### **Tecnologías y librerías sugeridas**

- .NET 8 / ASP.NET Core Web API
- Entity Framework Core (Code First o Migrations)
- SQL Server o SQLite
- JWT (para autenticación)
- Cloudinary .NET SDK
- AutoMapper (para mapear entidades y DTOs)
- FluentValidation (opcional para validaciones)

#### **Requerimientos funcionales del frontend**

##### **1. Login y registro de usuarios.**

- El registro se comunica con `/api/auth/register`.
- El login consume `/api/auth/login` y guarda el token JWT en `localStorage`.
- Según el rol, se redirige:
  - **Administrador** → Dashboard de administración.
  - **Cliente** → Home de cliente.

##### **2. Dashboard del administrador:**

- CRUD de propiedades.
- Formulario para subir imágenes (una o varias) y enviarlas al backend, que las sube a Cloudinary.

##### **3. Vista del cliente:**

- Ver propiedades disponibles.
- Ver detalles de cada propiedad (imagen, precio, descripción, ubicación).
- Simular botón de “Contactar”.

##### **4. Seguridad en el frontend:**

- Rutas protegidas con React Router.
  - Validación de token JWT.
  - Logout que limpia el `localStorage` y redirige al login.
- 

## Cloudinary

Cada equipo deberá:

1. Crear una cuenta gratuita en <https://cloudinary.com>.
2. Obtener sus credenciales (`CLOUD_NAME`, `API_KEY`, `API_SECRET`).
3. Configurar en el backend la conexión usando variables de entorno.
4. Enviar las imágenes desde el frontend al backend, y desde allí subirlas a Cloudinary.