Proyecto CRM inteligente basado en documentos.

1. Primera impresión de la estructura general del proyecto

```
CRM-DocumentAI/
├── src/
│   ├── Backend/
│   │   ├── CRM.API/                    # Web API principal
│   │   ├── CRM.Application/            # Lógica de aplicación
│   │   ├── CRM.Domain/                 # Modelos de dominio
│   │   └── CRM.Infrastructure/         # Acceso a datos, APIs externas
│   ├── Frontend/
│   │   ├── crm-web/                    # Next.js 14 application
│   │   ├── components/                 # Componentes reutilizables
│   │   ├── pages/                      # Páginas de la aplicación
│   │   └── lib/                        # Utilidades del frontend
│   ├── Core/
│   │   ├── CRM.Core.Security/          # Autenticación, autorización
│   │   ├── CRM.Core.Common/            # Utilidades compartidas
│   │   └── CRM.Core.Middleware/        # Middlewares personalizados
│   ├── Share/
│   │   ├── CRM.Shared.DTOs/            # Data Transfer Objects
│   │   ├── CRM.Shared.Models/          # Modelos compartidos
│   │   └── CRM.Shared.Constants/       # Constantes del sistema
│   ├── Data/
│   │   ├── CRM.Data.Context/           # Entity Framework Context
│   │   ├── CRM.Data.Repositories/      # Repositorios
│   │   └── CRM.Data.Migrations/        # Migraciones de BD
│   └── Services/
│       ├── CRM.Services.AI/            # Servicios de IA (OpenAI, etc.)
│       ├── CRM.Services.Document/      # Procesamiento de documentos
│       ├── CRM.Services.N8N/           # Integración con N8N
│       ├── CRM.Services.Email/         # Servicios de email
│       └── CRM.Services.Notification/  # Notificaciones
├── tests/
├── docs/
└── docker-compose.yml
```

2. Desarrollo por capas

    a. Capa de seguridad

        Componentes principales:

        - JWT Authentication: Para tokens de acceso
        - OAuth2 con 2FA: Integración con proveedores (Google, Microsoft)
        - Authorization Policies: Basados en roles y claims
        - Rate Limiting: Protección contra ataques

        Para OAuth2 + 2FA necesitamos:

        - Registro en Google/Microsoft Developer Console
        - Configurar redirect URIs

- Implementar TOTP (Time-based One-Time Password)
- Librería recomendada: Microsoft.AspNetCore.Authentication.Google

b. Capa de Internacionalización

Estructura de archivos .resx:

```
Resources/
├── Strings.es-ES.resx
├── Strings.en-US.resx
├── Strings.pt-BR.resx
└── Validation.es-ES.resx
```

Implementación:

```
public class LocalizationService : IStringLocalizer
{
    // Gestión de recursos multiidioma
    // Detección automática de cultura
    // Fallback a idioma por defecto
}
```

Implementación de la Capa de Internacionalización

Ubicación en la Estructura:

```
CRM-DocumentAI/
├── src/
│   ├── Core/
│   │   ├── CRM.Core.Localization/          # 🆕 CAPA DE INTERNACIONALIZACIÓN
│   │   │   ├── Services/
│   │   │   │   ├── ILocalizationService.cs
│   │   │   │   ├── LocalizationService.cs
│   │   │   │   └── CultureService.cs
│   │   │   ├── Resources/                  # Archivos .resx
│   │   │   │   ├── Strings.es-ES.resx
│   │   │   │   ├── Strings.en-US.resx
│   │   │   │   ├── Strings.pt-BR.resx
│   │   │   │   ├── Validation.es-ES.resx
│   │   │   │   ├── Validation.en-US.resx
│   │   │   │   └── Validation.pt-BR.resx
│   │   │   ├── Extensions/
│   │   │   │   └── LocalizationExtensions.cs
│   │   │   └── Models/
│   │   │       ├── LocalizedString.cs
```

c. Capa de ayuda

Funcionalidades:

- Tooltips contextuales
- Tours guiados (bibliotecas como Shepherd.js)
- Centro de ayuda integrado

- Documentación interactiva

3. Backend - Base de Datos

   Tecnologías recomendadas:

   - ORM: Entity Framework Core 8
   - BD: SQL Server + Azure Blob Storage (documentos)
   - Cache: Redis para resultados de IA
   - Vector DB: Pinecone para RAG

   Esquema sugerido para SQL Server

```
-- Tablas principales
Documents (Id, Name, Type, Content, UploadDate, UserId)
Entities (Id, DocumentId, Type, Value, Confidence)
Insights (Id, DocumentId, Category, Summary, Score)
Recommendations (Id, DocumentId, Action, Priority)
Users (Id, Email, Name, Role, Preferences)
DocumentAnalysis (Id, DocumentId, SentimentScore, KeyTopics)
```

4. Frontend - Dashboard

   Bibliotecas clave:

   - react-pdf para visualización de PDFs
   - react-dropzone para upload de archivos
   - recharts para gráficos y analytics
   - monaco-editor para edición de templates

   Frontend sugerido para el dashboard

```
Dashboard/
├── DocumentUpload/       # Drag & drop zona
├── DocumentViewer/       # PDF viewer integrado
├── AnalyticsCharts/      # Recharts dashboards
├── RecommendationPanel/  # Sugerencias IA
├── SearchInterface/      # RAG search
└── NotificationCenter/   # Alertas inteligentes
```