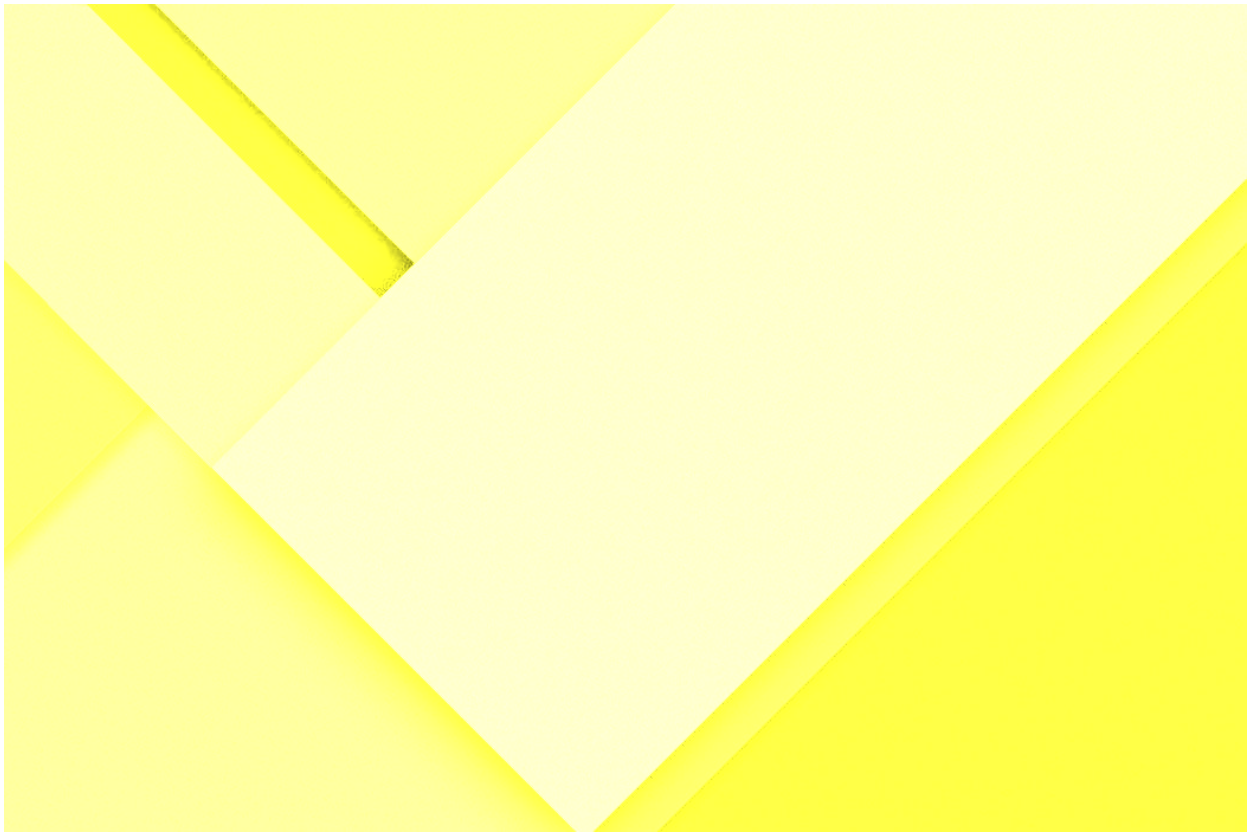


SPA - Risk Control - Backend

03/01/2026

Christian Edgardo Carrillo Aburto

Prueba Técnica (Ejercicio 2- Backend)

Introducción

Propósito de la API

API REST para la gestión de proveedores y evaluación de riesgos mediante integración con fuentes externas (OFAC, Offshore Leaks, World Bank).

Versión

- Versión de la API: 1.0
- Fecha: 03/01/2026

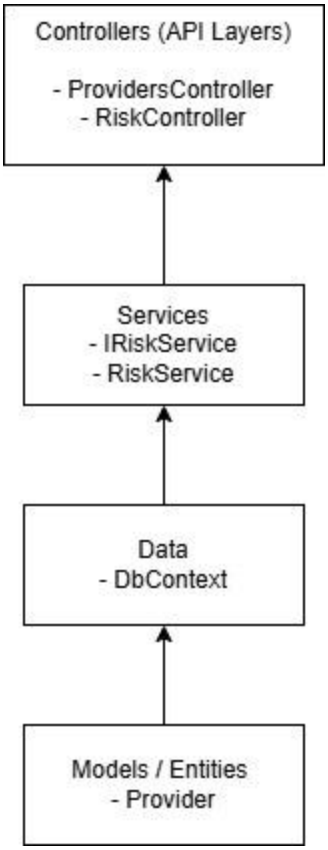
Descripción

Sistema backend que permite registrar proveedores y evaluar su nivel de riesgo consultando bases de datos internacionales de sanciones y filtros.

Arquitectura del Backend

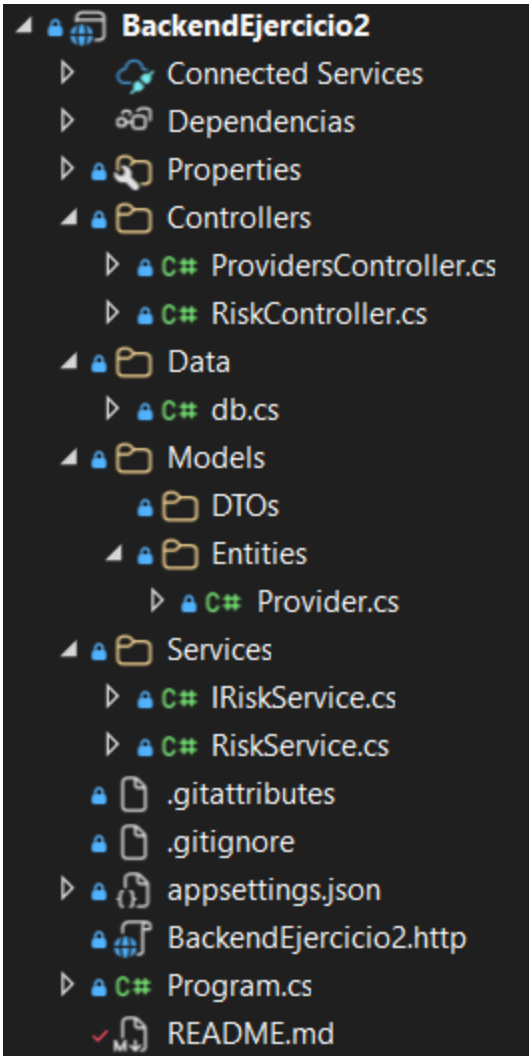
Patrón Arquitectónico

El proyecto utiliza una arquitectura en capas (N-Layer Architecture) con separación de responsabilidades:



Estructura de Carpetas

En la siguiente imagen se muestra la estructura de carpetas propuesta para este ejercicio.



Tecnologías y Dependencias

Stack Tecnológico

- .NET 10

- Entity Framework Core - ORM para acceso a datos

- SQL Server - Base de datos relacional

- HttpClient - Para consumir APIs externas de Python

Paquetes de NuGet Principales

- Microsoft.EntityFrameworkCore.SqlServer

- Swashbuckle.AspNetCore (Swagger)

Integraciones Externas

- API de Python (localhost:8000) - Para el servicio de screening contra OFAC , Offshore Leaks y The World Bank

Modelo de Datos

Entidad: Provider

Se pondrá una imagen para poder simplificar la entidad.

```
namespace BackendEjercicio2.Models.Entities
{
    [Table("PROVIDERS")]
    3 referencias
    public class Provider
    {
        [Key]
        4 referencias
        public int Id { get; set; }
        [Required]
        [MaxLength(200)]
        1 referencia
        public string LegalName { get; set; }
        [MaxLength(200)]
        0 referencias
        public string TradeName { get; set; }

        [Required]
        [StringLength(11)]
        0 referencias
        public string TaxId { get; set; }
```

Estructura de Base de Datos

```
CREATE TABLE PROVIDERS(
        ID INT IDENTITY PRIMARY KEY,
        LEGALNAME NVARCHAR(255) NOT NULL, -- Razón Social
        TRADENAME NVARCHAR(255),         -- Nombre Comercial
        TAXID CHAR(11) NOT NULL,         -- Identificación tributaria (11 dígitos)
        PHONE NVARCHAR(20),              -- Teléfono
        EMAIL NVARCHAR(255),                        -- Correo
        WEBSITE NVARCHAR(255),                      -- Sitio Web
        ADDRESSS NVARCHAR(500),                     -- Dirección Física
        COUNTRY NVARCHAR(100),                      -- País
        ANNUALREVENUE DECIMAL(18,2),    -- Facturación Anual
        LASTUPDATED DATETIME2 NOT NULL DEFAULT SYSDATETIME()
);
```

Endpoints de la API

Providers - Listar Todos

Endpoint: GET /api/providers

Descripción: Obtiene todos los proveedores ordenados por última actualización


```
[ { "id": 1, "legalName": "Empresa ABC S.A.", "tradeName": "ABC", "taxId": "12345678901",
"phone": "5551234567", "email": "contacto@abc.com", "website": "https://abc.com",
"addresss": "Calle 123", "country": "México", "annualRevenue": 1000000.00,
"lastUpdated": "2026-01-03T10:30:00Z" } ]
```


Providers - Obtener por id

Endpoint: GET /api/providers/{id`

Descripción: Obtiene un proveedor específico por su ID.

Parámetros:

- id (int, ruta): ID del proveedor


```
{ "id": 1, "legalName": "Empresa ABC S.A.", "tradeName": "ABC", "taxId": "12345678901",
```

"phone": "5551234567", "email": "contacto@abc.com", "website": "https://abc.com", "addresss": "Calle 123", "country": "México", "annualRevenue": 1000000.00, "lastUpdated": "2026-01-03T10:30:00Z" }

Providers - Crear

Endpoint: POST /api/providers

Descripción: Crea un nuevo proveedor.

Request Body

{ "legalName": "Empresa XYZ S.A.", "tradeName": "XYZ", "taxId": "98765432109", "phone": "5559876543", "email": "info@xyz.com", "website": "https://xyz.com", "addresss": "Avenida 456", "country": "Colombia", "annualRevenue": 500000.00 }

Providers - Actualizar

Endpoint: PUT /api/providers/{id}

Descripción: Actualiza un proveedor existente

Mismo parámetro que el de obtener por ID

Providers -Eliminar

Endpoint: DELETE /api/providers/{id}

Descripción: Elimina un proveedor.

Parámetros:

- id (int, ruta): ID del proveedor

Risk - Evaluar todas las fuentes

Endpoint POST /api/risk/{providerId}/all

Descripción: Evalúa el riesgo de un proveedor consultando OFAC, Offshore Leaks y World Bank.

Parámetros:

- providerId (int, ruta): ID del proveedor

Risk - Evaluar con Fuente Específica.

Endpoint: POST /api/risk/{providerId}/source/{sourceName}

Descripción: Evalúa el riesgo de un proveedor con una fuente específica.

Parámetros:

- providerId (int, ruta): ID del proveedor

- sourceName (string, ruta): Nombre de la fuente (`ofac`, `offshore-leaks`, `world-bank`)

Lógica de Negocio

RiskService

Responsabilidades:

- Consultar proveedores en la base de datos

- Comunicarse con la API externa de Python

- Transformar requests/responses

- Validar fuentes de datos disponibles

Reglas de Negocio:

1. El proveedor debe existir antes de evaluar riesgo

2. Solo se permiten fuentes: “ofac”, “offshore-leaks”, “world-bank”, “all”

3. Se utiliza el “LegalName” del proveedor para la búsqueda

4. Cada actualización de proveedor actualiza automáticamente “LastUpdated”

Integración con Fuentes Externas

API de Python - Screening Service

URL BASE, la cual es configurable en appsettings.json (por defecto: http://localhost:8000)

Autenticación: API KEY en Header ‘x-api-key’

Endpoints Consumidos

- POST /api/v1/search/all - Búsqueda en todas las fuentes

- POST /api/v1/search/ofac - Búsqueda solo en OFAC

- POST /api/v1/search/offshore-leaks - Búsqueda solo en Offshore Leaks

- POST /api/v1/search/world-bank - Búsqueda solo en World Bank

Request Format:

```
{
    "entity_name": “Miami " // Por ejemlo
}
```

Base de Datos

Conexión

Motor: SQL Server

Base de datos: EyTest2