**DOCUMENTACION TECNICA DE PRUEBA TECNICA DESARROLLADOR DE SERVICIOS WEB SENIOR – DENIS ARANA**

# SISTEMA GENERADOR DE ESTADOS DE CUENTA DE TARJETAS DE CREDITO

**CONTENIDO**

REQUERIMIENTOS

DIAGRAMAS

TECNOLOGIAS UTILIZADAS

PROYECTOS DE LA SOLUCION

EJECUCION DE LA SOLUCION

**REQUERIMIENTOS**

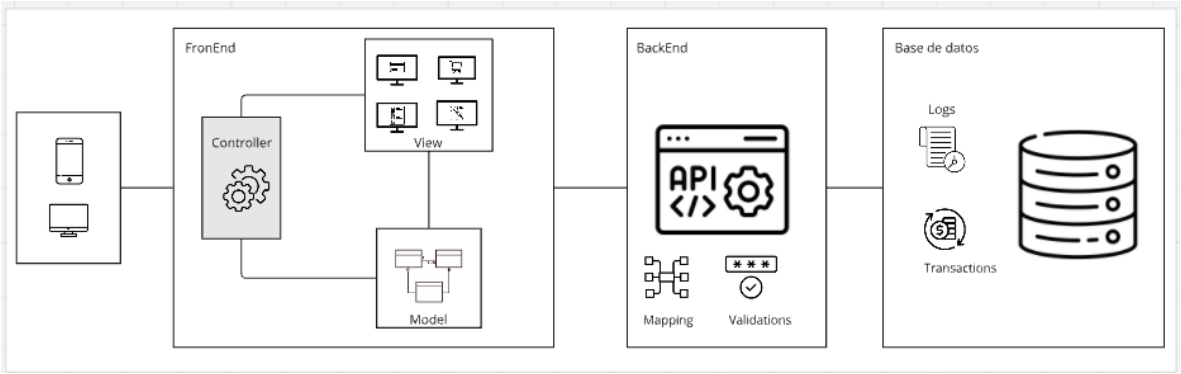
1. **Desarrollar una aplicación para generar el Estado de cuenta de una tarjeta de crédito**

La aplicación debe permitir visualizar el estado de cuenta de una tarjeta de crédito, incluyendo el detalle de movimientos, cálculo de cuota mínima, cálculo de contado e interés bonificable, y mostrar el saldo utilizado y disponible de la tarjeta de crédito.

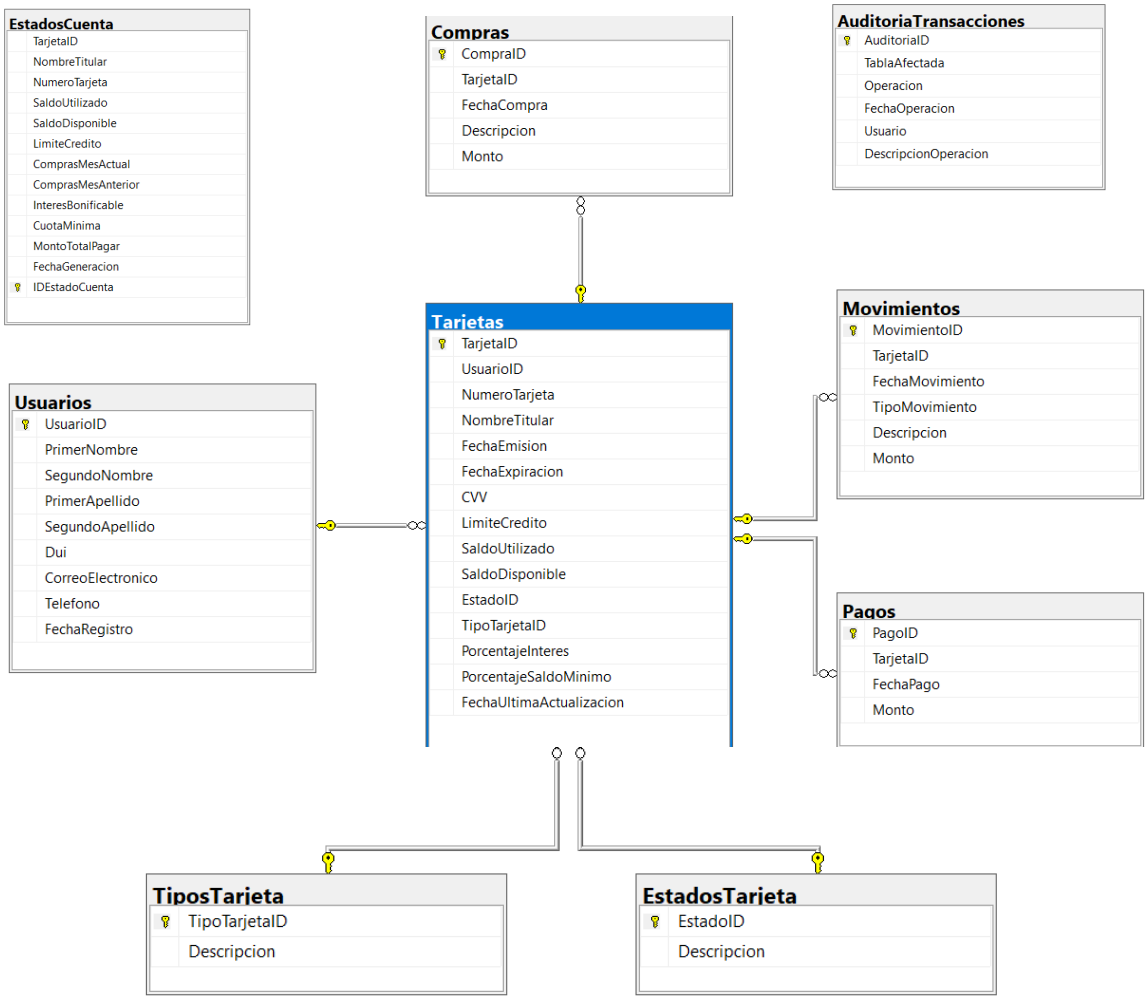
2. **La aplicación debe cumplir con los siguientes requisitos funcionales**
    a. **Pantalla de Estado de Cuenta que muestre:**
        i. **Campos:** nombre del titular, número de tarjeta, Mostrar el saldo actual, límite de crédito y el saldo disponible, Mostrar una lista de las compras mes actual, monto total de compras en el mes actual y anterior, Mostrar el Interés Bonificable
        ii. **Cuota Mínima a Pagar**
        iii. **Monto total a Pagar**
        iv. **Monto total de Contado con Intereses**
    b. **Pantalla de Compras**
    c. **Pantalla de Pagos**
    d. **Pantalla de Historial de Transacciones del mes**

# DIAGRAMAS

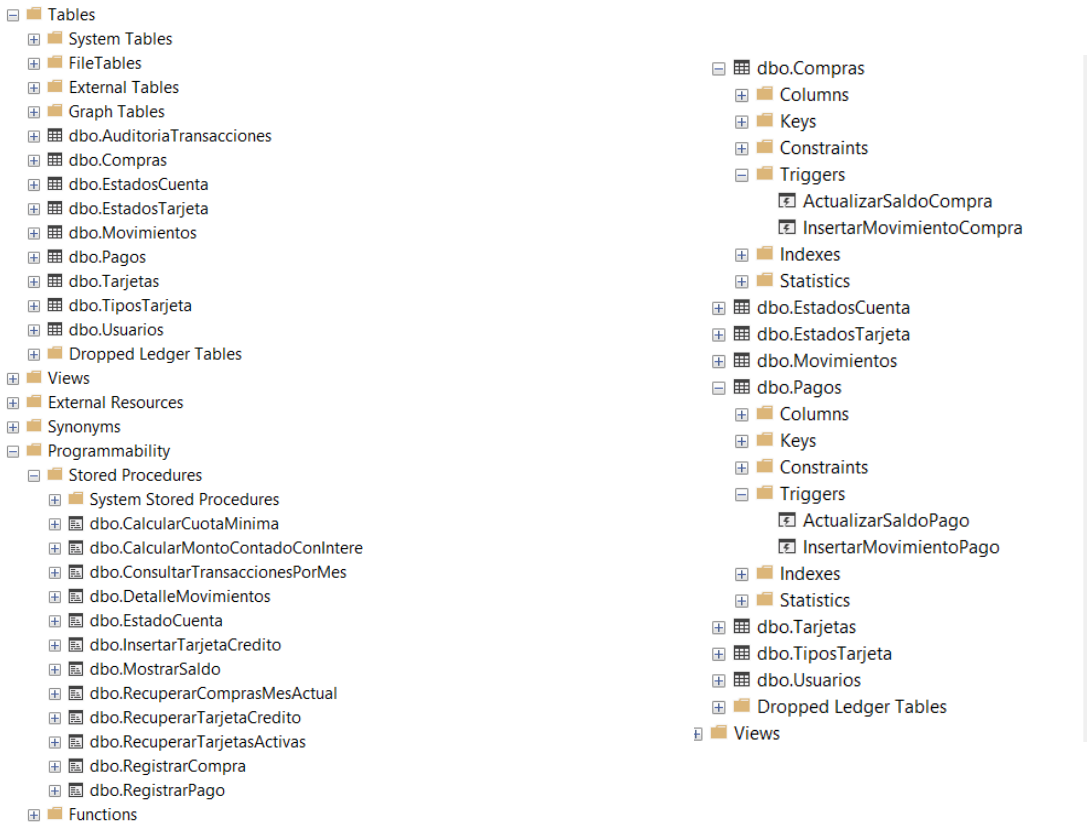
## 1. Diagrama de arquitectura




## 2. Diagrama de base de datos



## Procedimientos almacenados y Triggers



**TECNOLOGIAS UTILIZADAS**

1. **Backend**
    a. .Net Core Web API
2. **FrontEnd**
    a. Aplicación web Mvc .Net
3. **Versión**
    a. .Net: 6.0
4. **Librerías**
    a. Automapper
    b. FluentValidation
5. **Base de datos**
    a. SqlServer Developer 2022
6. **IDE**
    a. Microsoft Visual Studio Community 2022 (64 bits) - Current
    b. Versión 17.11.5
    c. SQLServer Management Studio
7. **Software de pruebas API**
    a. Postman
8. **Documentacion API**
    a. Swagger

**PROYECTOS DE LA SOLUCION**

- **API**: endpoints que usa el frontend para acceder a la información de tarjetas
- **Data**: Acceso a la base de datos, patron repository, mapeo tablas-entidades
- **FrontEndTarjetaCredito**: Pantallas de tarjetas, estado de cuenta, compras y pagos
- **Model**: Entidades y DTOs
- **Utils**: Automapper, FluentValidation.

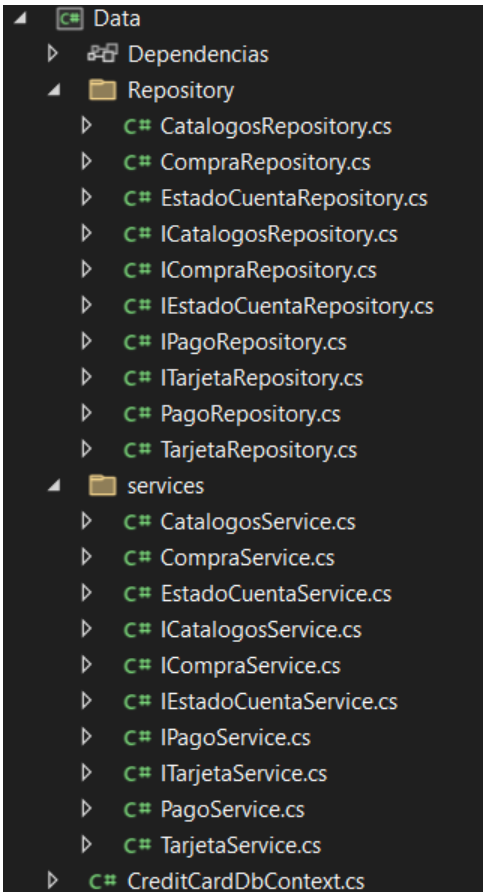
Para el backend se implementó **REST API** para la interacción con la base de datos utilizando **ASP.NET Web API** y para el FrontEnd Se creo una **aplicación web MVC .Net**.

**Proyectos en la solución**

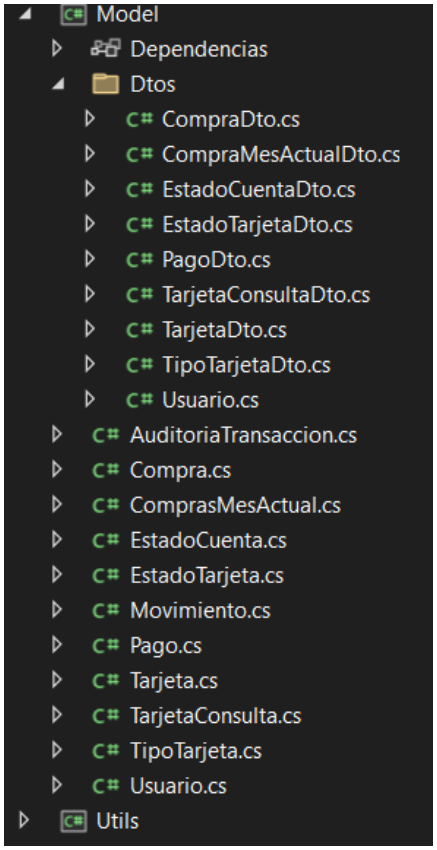
- **BackEnd**

1. Proyecto API: contiene los endpoint que acceden a la base de datos, a continuación se muestra la documentación usando Swagger.

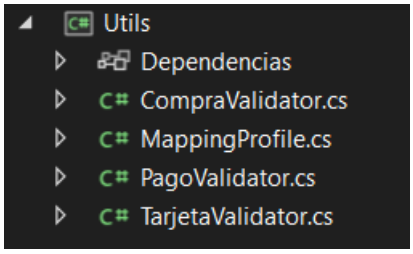
2. Proyecto Data: implementa el patrón repository para el acceso a datos




3. Proyecto Model: contiene las entidades y DTOs

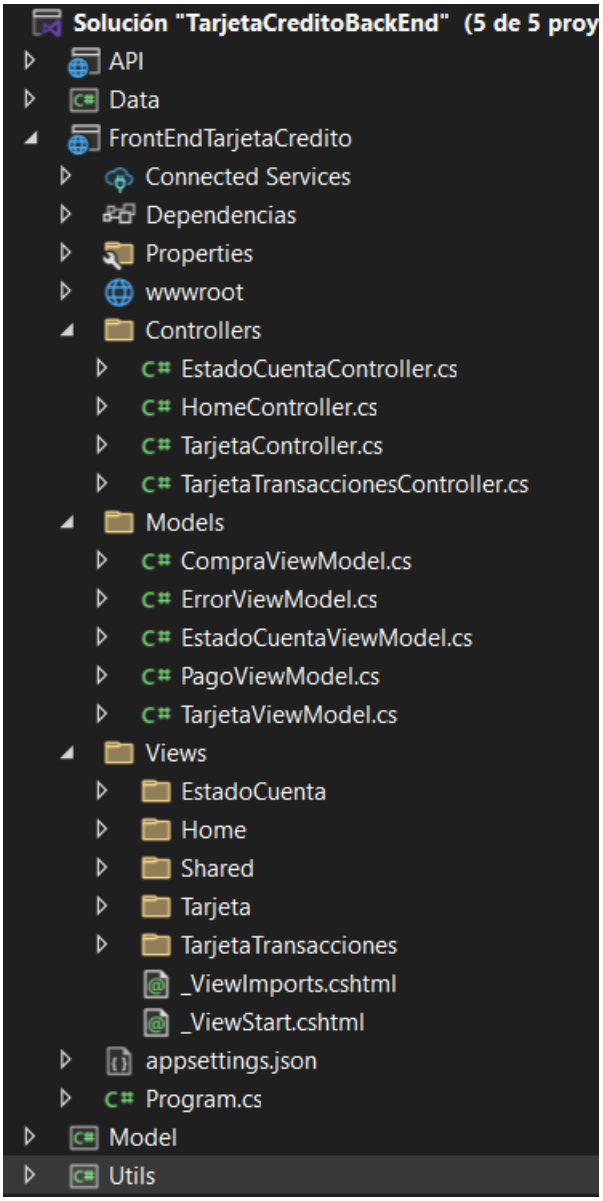



4. Proyecto Utils: contiene la implementación de Automapper y FluentValidation




- **FrontEnd**

1. Proyecto FrontEndTarjetaCredito: contiene las vistas, ViewModel y Controllers

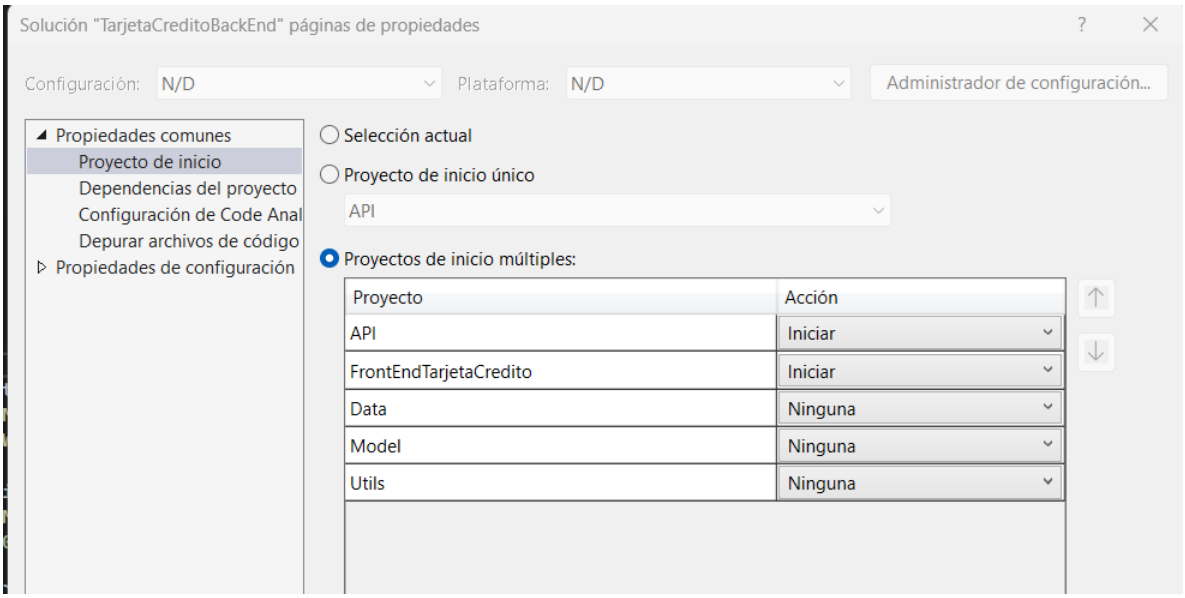

Solución "TarjetaCreditoBackEnd" (5 de 5 proy
API
Data
FrontEndTarjetaCredito
Connected Services
Dependencias
Properties
wwwroot
Controllers
EstadoCuentaController.cs
HomeController.cs
TarjetaController.cs
TarjetaTransaccionesController.cs
Models
CompraViewModel.cs
ErrorViewModel.cs
EstadoCuentaViewModel.cs
PagoViewModel.cs
TarjetaViewModel.cs
Views
EstadoCuenta
Home
Shared
Tarjeta
TarjetaTransacciones
_ViewImports.cshtml
_ViewStart.cshtml
appsettings.json
Program.cs
Model
Utils

Prueba Técnica Desarrollador de Servicios Web Senior – Denis Arana

**EJECUTAR EL PROYECTO**

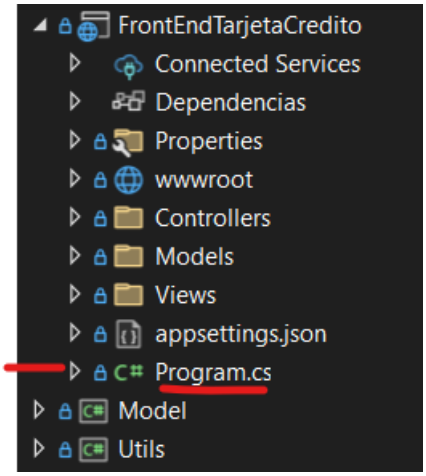
Para iniciar el proyecto debe estar configurado el inicio de la **API** y el **FrontEndTarjetaCredito**

1. **Clic derecho sobre el nombre de la solución y seleccionar la opción propiedades**



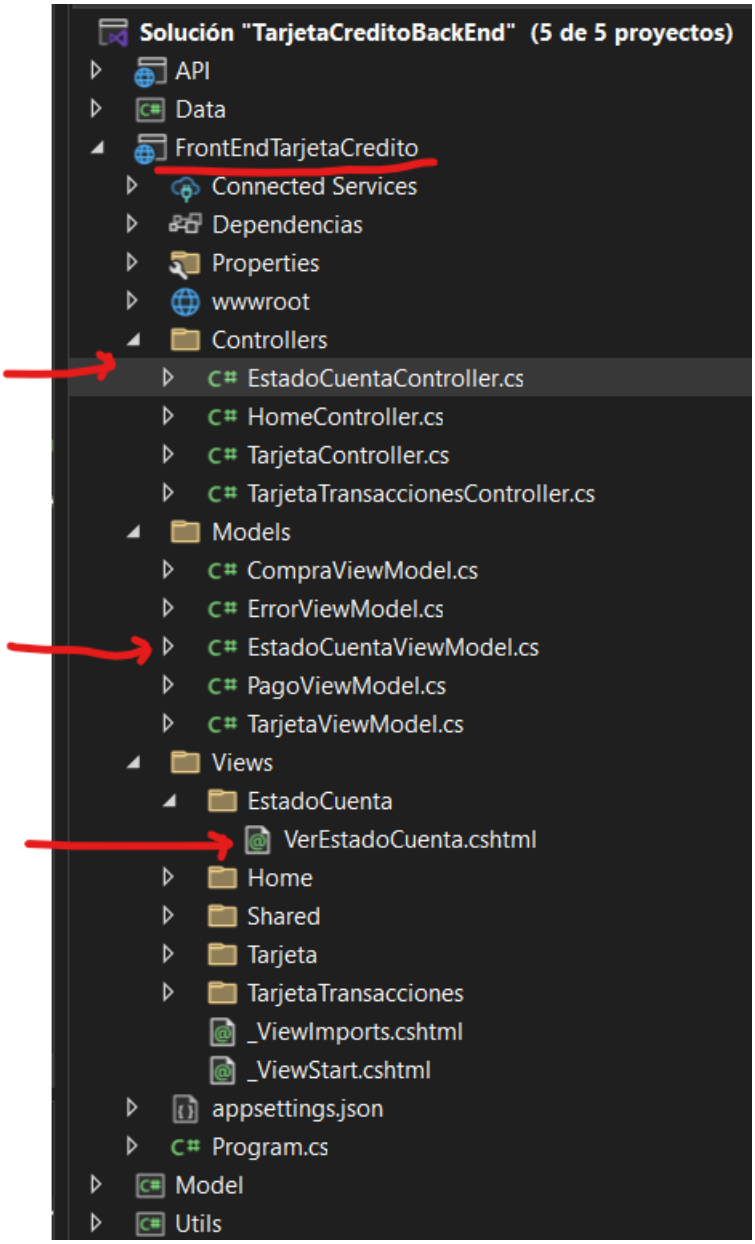

2. **Descargar e instalar script SQL para generar la base de datos:**
   **proyecto:** Utils/sqlscript/TarjetaCreditoDb.sql

3. **Establecer la ruta base servidor y puerto (Ejemplo: http://localhost:5128/ ) en el archivo** Program.cs **del proyecto** FrontEndTarjetaCredito.
   **Linea 11: client.BaseAddress = new Uri("http://localhost:5128/");**

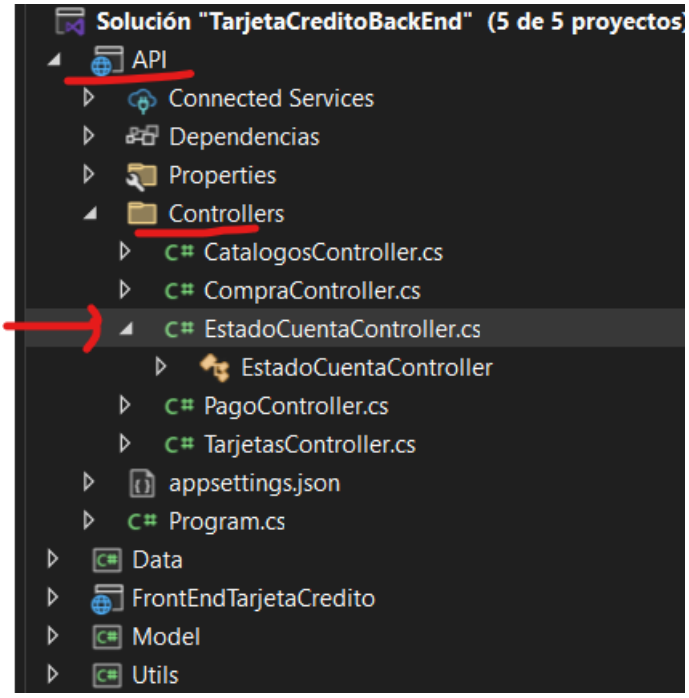


## Para navegar por la funcionalidad de generación de estado de cuenta seguir los pasos que se detallan:

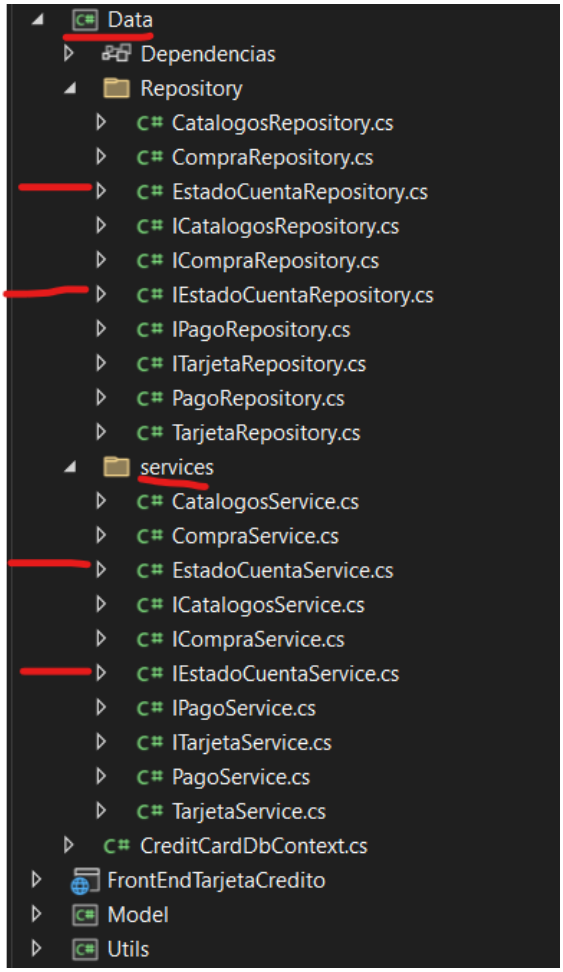
1. Desplegar el proyecto **FrontEndTarjetaCredito**
2. Desplegar la carpeta **Views/EstadoCuenta** y abrir el archivo **VerEstadoCuenta.cshtml**
3. En la carpeta **Models** abrir el archivo **EstadoCuentaViewModel.cs**
4. En la carpeta **Controllers** abrir el archivo **EstadoCuentaController.cs**




5. En el proyecto **API** verificar en la carpeta **Controllers** el controlador **EstadoCuentaController.cs**

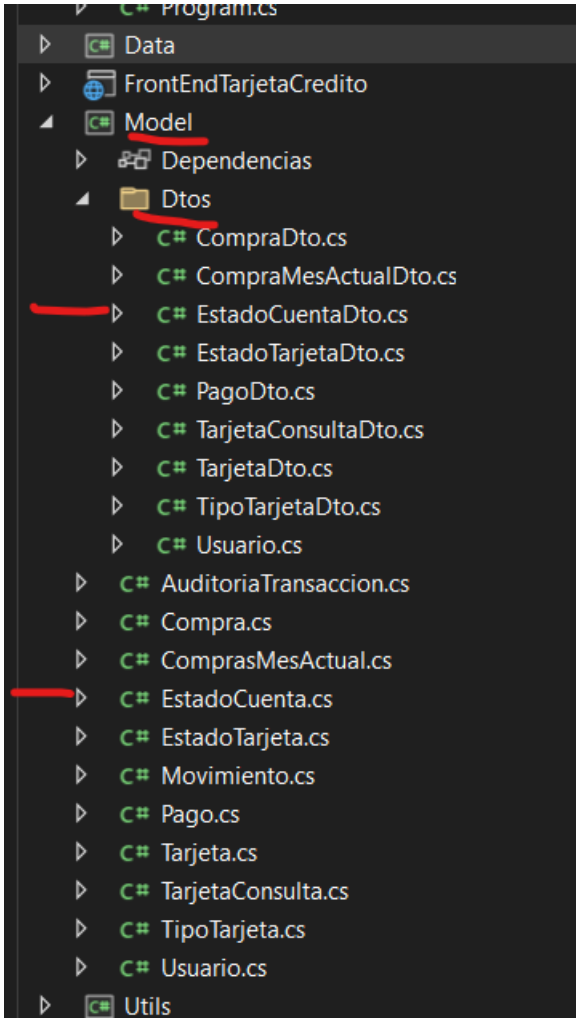



6. En el proyecto **Data** verificar en la carpeta **Service** los archivos **IEstadocuentaService** y **EstadocuentaService**



7. Tambien verificar en la carpeta **Repository** los archivos **IEstadocuentaRepository** y **EstadocuentaRepository**

8. En el proyecto **Model,** carpeta **Dtos** verificar el archivo **EstadoCuentaDto.cs y EstadoCuenta.cs**




9. Procedimiento almacenado y tabla en base de datos: **[dbo].[EstadoCuenta], [dbo].[EstadosCuenta]**