

- [Base de datos "FacturadorDB"](#)
  - [Crear Tablas](#)
    - [Cliente](#)
    - [Factura\\_Cabecera](#)
    - [Factura\\_Detalle](#)
    - [Articulo](#)
- [Proyecto API "FacturadorAPI" \(BackEnd\)](#)
  - [Modelos](#)
  - [Controladores](#)
    - [Aclaraciones](#)
- [Proyecto ASP.net mvc o Blazor "FacturadorMVC" \(FrontEnd\)](#)
  - [Menu Bar](#)
    - [Clientes:](#)
    - [Factura\\_Cabecera:](#)
    - [Factura\\_Detalle:](#)
    - [Articulo:](#)
  - [Consultas Linq](#)
  - [Vistas \(SQL\)](#)
  - [SP](#)
- [Programación](#)
  - [Herramientas a Evaluar:](#)
  - [Resumen](#)
  - [Menu](#)
  - [Validaciones adicionales:](#)

---

## Base de datos "FacturadorDB"

Crear proyecto de base de datos en Devcontainer con .dotnet 6.0 y sqlserver.

### Crear Tablas

Se deberá generar el script sql para la creación de las siguientes tablas.

#### Cliente

ID	Tipo Dato	Descripción
Cli_ID	entero	ID Pk de la tabla partiendo desde 1000 incrementando de uno en uno
Razon Social	cadena 255	Descripción de la rason social
CUIT	cadena 50	CUIT del cliente

ID	Tipo Dato	Descripción
Direccion	cadena 255	Dirección cliente
Deshabilitado	boolean (0/1)	1 verdadero 0 falso

### Factura\_Cabecera

ID	Tipo Dato	Descripción
FC_ID	entero	ID Pk de la tabla partiendo desde 2000 incrementando de uno en uno
Fecha Alta	fecha	fecha de alta de registro
Cli_ID	cadena(50)	ID cliente
Estado	cadena(50)	fecha de alta de registro

### Factura\_Detalle

ID	Tipo Dato	Descripción
Fact_ID	entero	ID factura cabecera
FC_DTL_ID	entero	ID linea factura partiendo sde 1 incrementando de uno en uno
Fecha Alta	fecha	fecha de alta de registro
ART_ID	cadena(50)	id de articulo
Cant	decimal	cantidad de articulo
Precio	decimal	precio de articulo
Moto	decimal	monto de la linea

### Articulo

1. Crear la tabla articulo con valores de columnas relevantes.

---

## Proyecto API "FacturadorAPI" (BackEnd)

En el mismo devcontainer anterior, crear un proyecto de tipo API con dotnet 6.0.

### Modelos

Generar clases de modelos de las tablas creadas, en lo posible usar herramientas de scaffolding.

### Controladores

Generar controladores de los modelos creados con los siguientes metodos

1. Listar datos de entidad.
2. Insertar datos de entidad.
3. Actualizar datos de entidad.
4. Eliminar datos de entidad.

### Aclaraciones

1. Los métodos y clases pueden llamarse como desee.
2. De lo posible agregar capa de repositorio (Interfaz y Clase de controlador).
3. Usar Entity Framework con su respectiva clase de DbContext para el manejo de la base de datos.

## Proyecto ASP.net mvc o Blazor "FacturadorMVC" (FrontEnd)

Generar un proyecto ASP.net MVC que consuma la API generada anteriormente

### Menu Bar

#### Clientes:

1. Al presionar la opción de menu se deben listar todos los registros.
2. Debe existir un botón que llamado **Nuevo** que al presionarlo permita capturar los datos la entidad y generar el alta en la base de datos.
3. Los registros listados deben tener un botón o link adjunto que permita eliminar o editar el registro.
4. Al presionar editar o eliminar se debe abrir una pantalla con el detalle del registro y debe mostrar los botones eliminar, actualizar o cancelar.
5. se deben programar las acciones de los botones.

#### Factura\_Cabecera:

1. Al presionar la opción de menu se deben listar todos los registros.
2. Debe existir un botón que llamado **Nuevo** que al presionarlo permita capturar los datos la entidad y generar el alta en la base de datos.
3. Los registros listados deben tener un botón o link adjunto que permita eliminar o editar el registro.
4. Al presionar editar o eliminar se debe abrir una pantalla con el detalle del registro y debe mostrar los botones eliminar, actualizar o cancelar.
5. se deben programar las acciones de los botones.
6. De ser posible mostrar grilla con detalle de factura.

#### Factura\_Detalle:

1. Al presionar la opción de menu se deben listar todos los registros.

2. Debe existir un botón que llamado **Nuevo** que al presionarlo permita capturar los datos la entidad y generar el alta en la base de datos.
3. Los registros listados deben tener un botón o link adjunto que permita eliminar o editar el registro.
4. Al presionar editar o eliminar se debe abrir una pantalla con el detalle del registro y debe mostrar los botones eliminar, actualizar o cancelar.
5. se deben programar las acciones de los botones.

### Articulo:

1. Al presionar la opción de menu se deben listar todos los registros.
2. Debe existir un botón que llamado **Nuevo** que al presionarlo permita capturar los datos la entidad y generar el alta en la base de datos.
3. Los registros listados deben tener un botón o link adjunto que permita eliminar o editar el registro.
4. Al presionar editar o eliminar se debe abrir una pantalla con el detalle del registro y debe mostrar los botones eliminar, actualizar o cancelar.
5. se deben programar las acciones de los botones.

## Consultas Linq

1. Query1: Realizar la suma del monto de las facturas realizadas el mes anterior donde los montos facturados superen los 10000.
2. Query2: Listar las facturas de los clientes cuyo cuit termine en 8.
3. Query3: Listar las facturas que contengan el producto con id igual a **AGD\_123**.

## Vistas (SQL)

1. View1: Crear una vista que muestre los siguientes campos de la tabla Factura\_Cabecera: [FC\_ID, Estado] y los siguientes campos de la tabla Factura\_Detalle: [sum(Moto)].
2. View2: Crear una vista que muestre de la tabla Factura\_Cabecera: FC\_ID, Fecha Alta y de la tabla Cliente: Razon Social, CUIT.

## SP


1. SP1: Crear un Sp llamado listado de **FacturasPorClienteProductoMasVendido** que reciba por parametro Fecha desde y Fecha Hasta y el ID de cliente y liste las facturas creadas en ese intervalo de tiempo para el cliente indicado, la salida debe mostrar todos los datos de la factura cabecera con el ID y Nombre del producto mas vendido para cada cliente.
  2. SP2: Crear un SP igual al anterior pero que muestre todos los datos de la cabecera de las facturas y el detalle de las mismas.
-

# Programación

## Herramientas a Evaluar:

1. **Requerido:** Lenguaje C#, Proyecto APi, [ASP.net](#) mvc, SP, Exportación archivo .txt, uso de git, uso de azure devops.
2. **Deseable:** Se valoraran uso de librerías conocidas EJ: Serilog, OData, Polly, Fluent validation, etc.

## Resumen

-  Realizar un programa que muestre el siguiente menu y permita ejecutar las operaciones indicadas.

## Menu

1. Clientes
  - a. CRUD Cliente.  
Funciones y Validaciones:
  - b. Validar cuit unico.
  - c. Buscar por cuit.
  - d. Deshabilitar clientes para los que la razón social empiece con la palabra **Distribuidora**.
  - e. Listar Clientes (listar todos los clientes)
2. Facturas
  - a. CRUD (primero cabecera luego detalle, validar cliente valido y valores mayor a cero en cantidad y precio, calcular monto).  
Funciones y Validaciones:
  - c. Listar Facturas (listar todas las facturas)
3. Consultas
  - a. Query1: Ejecutar Query, mostrar resultado Query OK o Error.
4. Vistas:
  - a. View1: Ejecutar Vista y mostrar salida.
5. SP
  - a. SP1: Solicitar parametros par SP y ejecutar, mostrar resultado OK o Error.
  - b. SP2: Solicitar parametros par SP y ejecutar, mostrar salida.

## Validaciones adicionales:

1. SP: Al ejecutar ya sea SP1 o SP2, se debe guardar un log en un archivo .txt que indica la fecha y hora, el nombre del sql que se ejecutó y el nro de registros devueltos.