

**UNIVERSIDAD NACIONAL DE INGENIERÍA  
(UNI)**



**Alicia Jazmín Delgado Mairena**

**Yaiza Daniela Castellano Matamoros**

**Anthony Derek Ramirez Andino**

**Genesis Tatiana García Morales**

**María Elena García Casaya**

**DACTIC**

**CARRERA DE INGENIERÍA DE SISTEMAS**

**Programación II**

**ING. Abel Edmundo Marin Reyes**

**Mejora del Proyecto BancoSimpres2T1**

Creamos el repositorio en GitHub llamado “MejoraBancoSimple2T1”, además añadimos las ramas para trabajar en colaboración en los cambios a realizar.

github.com/AlidElgado21/MejoraBancoSimple2T1/settings/access?guidance\_task=

pestaña Microsoft Word Ejerc... El aviso Escolar - AB... T E Libro BLACKPINK

Select all

Type ▾

Find a collaborator...

Castellano-Dan

Awaiting Castellano-Dan's response

Pending Invite

derek569-hub

Awaiting derek569-hub's response

Pending Invite

ElenaGC17

Collaborator

Genesis-GM

Collaborator

Get team access controls and discussions for your contributors in an organization.

NEW

 Private repos and unlimited members are free.

Create an organization

master

9 minutes ago

Default

...

Your branches

Branch	Updated	Check status	Behind	Ahead	Pull request	
Rama-Cambios-Alicia	5 minutes ago		0	0		...

Active branches

Branch	Updated	Check status	Behind	Ahead	Pull request	
Rama-Cambios-Anthony	now		0	0		...
Rama-cambios-Yaiza	4 minutes ago		0	0		...
Rama-Cambios-Alicia	5 minutes ago		0	0		...
Rama-Cambios-Maria	5 minutes ago		0	0		...
Rama-Cambios-Genesis	7 minutes ago		0	0		...

Subimos la carpeta “BancoSimple2T1” al repositorio local “MejoraBancoSimple2T1”

```
her.runtimeconfig.json
create mode 100644 BancoSimple2T1/obj/Debug/net8.0-windows/BancoSimple2T1.dll
create mode 100644 BancoSimple2T1/obj/Debug/net8.0-windows/BancoSimple2T1.genru
ntimeconfig.cache
create mode 100644 BancoSimple2T1/obj/Debug/net8.0-windows/BancoSimple2T1.pdb
create mode 100644 BancoSimple2T1/obj/Debug/net8.0-windows/apphost.exe
create mode 100644 BancoSimple2T1/obj/Debug/net8.0-windows/ref/BancoSimple2T1.d
ll
create mode 100644 BancoSimple2T1/obj/Debug/net8.0-windows/refint/BancoSimple2T
L.dll
create mode 100644 BancoSimple2T1/obj/project.assets.json
create mode 100644 BancoSimple2T1/obj/project.nuget.cache
create mode 100644 Sql.txt

alici@AliciaD MINGW64 ~/Downloads/BancoSimple2T1/BancoSimple2T1 (master)
$ git remote add origin https://github.com/Alidelgado21/MejoraBancoSimple2T1.git

alici@AliciaD MINGW64 ~/Downloads/BancoSimple2T1/BancoSimple2T1 (master)
$ git push -u origin master
Enumerating objects: 231, done.
Counting objects: 100% (231/231), done.
Delta compression using up to 4 threads
Compressing objects: 100% (214/214), done.
Writing objects: 27% (63/231), 4.67 MiB | 779.00 KiB/s
```

Clonamos el repositorio

```
MINGW64:/c/Users/alici/Downloads/BancoSimple2T1/BancoSimple2T1
alici@AliciaD MINGW64 ~/Downloads/BancoSimple2T1/BancoSimple2T1 (master)
$ git clone https://github.com/Alidelgado21/MejoraBancoSimple2T1.git
Cloning into 'MejoraBancoSimple2T1'...
remote: Enumerating objects: 231, done.
remote: Counting objects: 100% (231/231), done.
remote: Compressing objects: 100% (137/137), done.
remote: Total 231 (delta 77), reused 231 (delta 77), pack-reused 0 (from 0)
Receiving objects: 100% (231/231), 13.09 MiB | 642.00 KiB/s, done.
Resolving deltas: 100% (77/77), done.
Updating files: 100% (195/195), done.
```

## Realización de Cambios

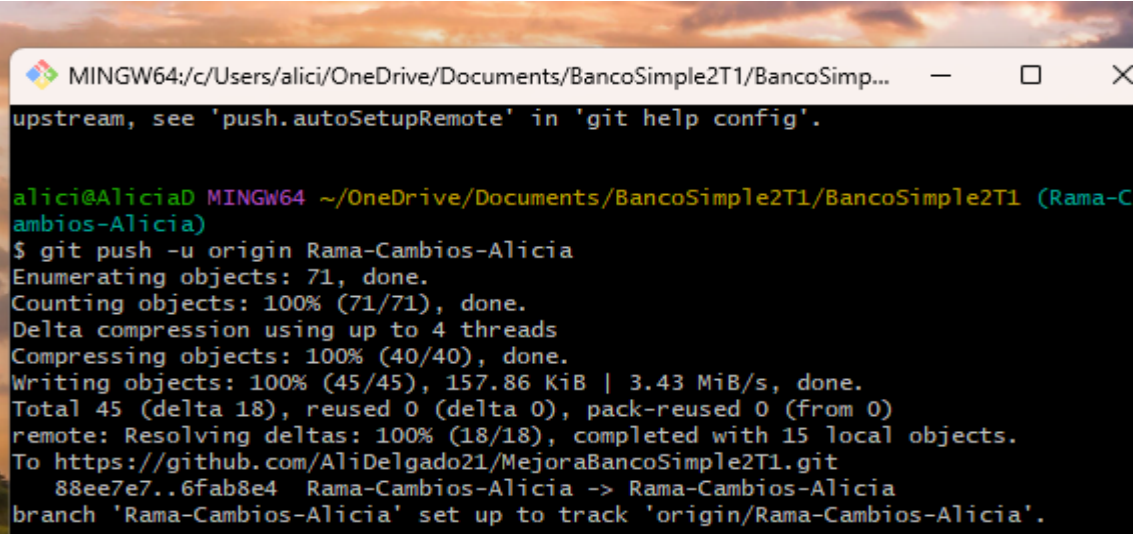
### 1. Clase [AgregarClienteForms.cs](#)

Se validan los campos txtNombre y txtIdentificacion además que agrega un nuevo cliente si son válidos

```
- referencias
private void btnAceptar_Click(object sender, EventArgs e)
{
    // Valida los datos ingresados y crea un nuevo cliente si son válidos.
    if (string.IsNullOrEmpty(txtNombre.Text) || string.IsNullOrEmpty(txtIdentificacion.Text))
    {
        MessageBox.Show("Todos los campos son obligatorios.", "Validación", MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return;
    }

    NuevoCliente = new Cliente
    {
        Nombre = txtNombre.Text.Trim(),
        Identificacion = txtIdentificacion.Text.Trim()
    };

    DialogResult = DialogResult.OK;
    Close();
}
```



The screenshot shows a terminal window with the title bar "MINGW64:/c:/Users/alici/OneDrive/Documents/BancoSimple2T1/BancoSimple2T1...". The terminal output shows a successful git push to a remote repository. The user is 'alici@AliciaD' and the branch is 'Rama-Cambios-Alicia'. The output includes details about enumerating objects, counting, compressing, and writing objects, as well as the remote repository URL and the branch name.

```
upstream, see 'push.autoSetupRemote' in 'git help config'.

alici@AliciaD MINGW64 ~/OneDrive/Documents/BancoSimple2T1/BancoSimple2T1 (Rama-Cambios-Alicia)
$ git push -u origin Rama-Cambios-Alicia
Enumerating objects: 71, done.
Counting objects: 100% (71/71), done.
Delta compression using up to 4 threads
Compressing objects: 100% (40/40), done.
Writing objects: 100% (45/45), 157.86 KiB | 3.43 MiB/s, done.
Total 45 (delta 18), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (18/18), completed with 15 local objects.
To https://github.com/AlidDelgado21/MejoraBancoSimple2T1.git
 88ee7e7..6fab8e4 Rama-Cambios-Alicia -> Rama-Cambios-Alicia
branch 'Rama-Cambios-Alicia' set up to track 'origin/Rama-Cambios-Alicia'.
```

Agregué un comentario sobre la función del botón Cancelar de la clase

[AgregarClienteForms.cs](#)

```
//Cancela la creación del cliente y cierra el formulario.
private void btnCancelar_Click(object sender, EventArgs e)
{
    DialogResult = DialogResult.Cancel;
    Close();
}
```

## 2. Clase BancoSimpleContext.cs

Agregué un comentario sobre la gestión y el contexto de la base de datos

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Microsoft.EntityFrameworkCore;
using BancoSimple2T1.Models;

namespace BancoSimple2T1.Data
{
    public class BancoSimpleContext : DbContext
    {
        // Contexto de base de datos para el sistema bancario.
        // Gestiona entidades como Clientes, Cuentas y Transacciones.
        public DbSet <Cliente> Cliente { get; set; }
        public DbSet <Cuenta> Cuenta { get; set; }
        public DbSet <Transaccion> Transacciones { get; set; }
    }
}
```

Agregué comentarios sobre el funcionamiento de los métodos OnConfiguring y OnModelCreating

```
// Configura la conexión a la base de datos.
protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
{
    optionsBuilder.UseSqlServer(@"Server = DESKTOP-B1JNRCE\SQLSERVER2019; data

// Configura reglas adicionales para el modelo, como filtros globales.
protected override void OnModelCreating(ModelBuilder modelBuilder)
{
    modelBuilder.Entity<Cuenta>().HasQueryFilter ( c => c.Activa);
}
```

Subí los cambios realizados al repositorio “MejoraBancoSimple2T1”

```
MINGW64:/c/Users/alici/OneDrive/Documents/BancoSimple2T1/BancoSimp...
emote: Compressing objects: 100% (9/9), done.
emote: Total 9 (delta 5), reused 0 (delta 0), pack-reused 0 (from 0)
npacking objects: 100% (9/9), 3.99 KiB | 70.00 KiB/s, done.
rom https://github.com/Alidelgado21/MejoraBancoSimple2T1
* branch      master      -> FETCH_HEAD
   3823868..20f2c8c master  -> origin/master
uccessfully rebased and updated refs/heads/master.

alici@AliciaD MINGW64 ~/OneDrive/Documents/BancoSimple2T1/BancoSimple2T1/Mejo
ncoSimple2T1 (master)
$ git push origin master
Enumerating objects: 25, done.
Counting objects: 100% (25/25), done.
Delta compression using up to 4 threads
Compressing objects: 100% (12/12), done.
Writing objects: 100% (16/16), 1.91 KiB | 979.00 KiB/s, done.
Total 16 (delta 8), reused 0 (delta 0), pack-reused 0 (from 0)
emote: Resolving deltas: 100% (8/8), completed with 6 local objects.
o https://github.com/Alidelgado21/MejoraBancoSimple2T1.git
   20f2c8c..e67625b master -> master

alici@AliciaD MINGW64 ~/OneDrive/Documents/BancoSimple2T1/BancoSimple2T1/Mejo
ncoSimple2T1 (master)
```

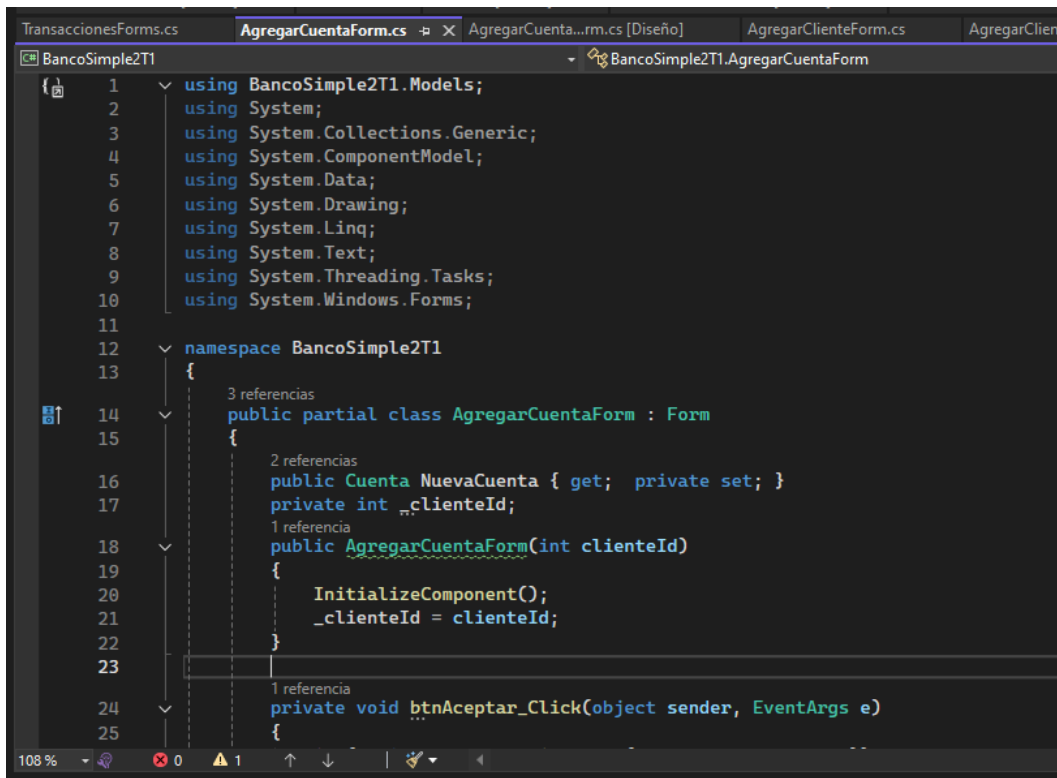
### 3. Clase AgregarCuentaForm.cs

Primero, cloné el repositorio para poder guardar y subir mis cambios.

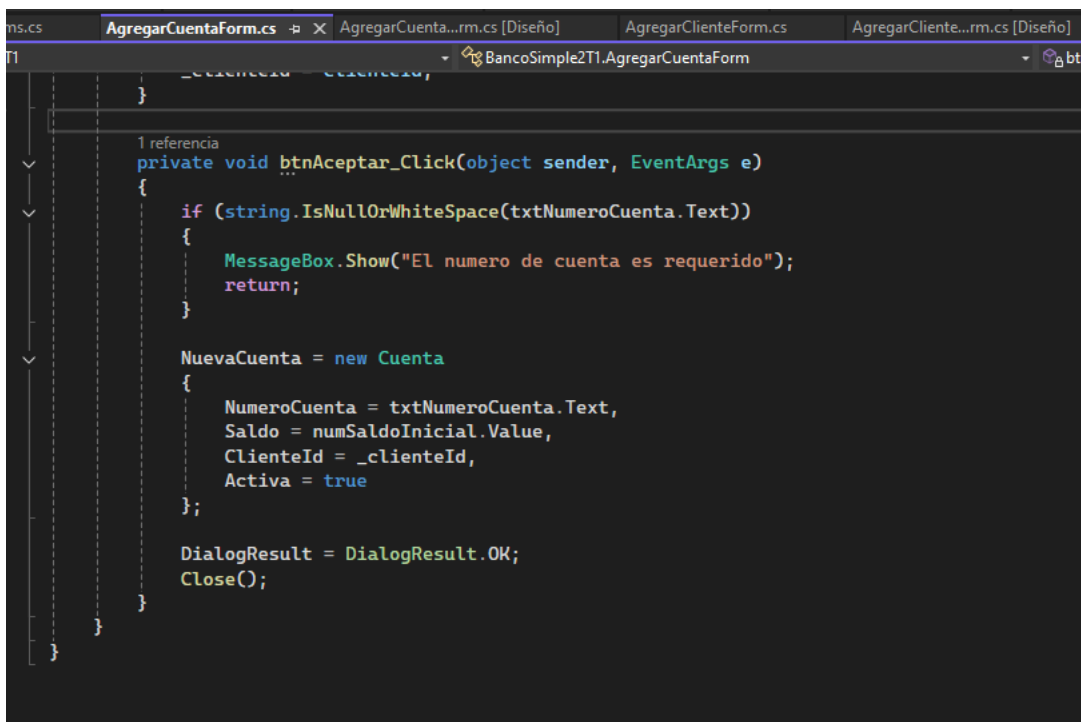
```
MINGW64:/c/Users/DELL/OneDrive/Desktop/Gene/MejoraBancoSimple2T1-...
DELL@DESKTOP-ELTTF1D MINGW64 ~/OneDrive/Desktop/Gene/MejoraBancoSimple2T1-master
$ git clone https://github.com/Alidelgado21/MejoraBancoSimple2T1.git
Cloning into 'MejoraBancoSimple2T1'...
remote: Enumerating objects: 279, done.
remote: Counting objects: 100% (279/279), done.
remote: Compressing objects: 100% (161/161), done.
remote: Total 279 (delta 101), reused 274 (delta 96), pack-reused 0 (from 0)
Receiving objects: 100% (279/279), 13.24 MiB | 9.94 MiB/s, done.
Resolving deltas: 100% (101/101), done.

DELL@DESKTOP-ELTTF1D MINGW64 ~/OneDrive/Desktop/Gene/MejoraBancoSimple2T1-master
$
```

Luego, entre al proyecto BancoSimple2T1 y luego a la clase [AgregarCuentaForms.cs](#)



```
1 using BancoSimple2T1.Models;
2 using System;
3 using System.Collections.Generic;
4 using System.ComponentModel;
5 using System.Data;
6 using System.Drawing;
7 using System.Linq;
8 using System.Text;
9 using System.Threading.Tasks;
10 using System.Windows.Forms;
11
12 namespace BancoSimple2T1
13 {
14     public partial class AgregarCuentaForm : Form
15     {
16         public Cuenta NuevaCuenta { get; private set; }
17         private int _clienteId;
18         public AgregarCuentaForm(int clienteId)
19         {
20             InitializeComponent();
21             _clienteId = clienteId;
22         }
23
24         private void btnAceptar_Click(object sender, EventArgs e)
25         {
```



```
        {
26             if (string.IsNullOrWhiteSpace(txtNumeroCuenta.Text))
27             {
28                 MessageBox.Show("El numero de cuenta es requerido");
29                 return;
30             }
31
32             NuevaCuenta = new Cuenta
33             {
34                 NumeroCuenta = txtNumeroCuenta.Text,
35                 Saldo = numSaldoInicial.Value,
36                 ClienteId = _clienteId,
37                 Activa = true
38             };
39
40             DialogResult = DialogResult.OK;
41             Close();
42         }
43     }
44 }
```

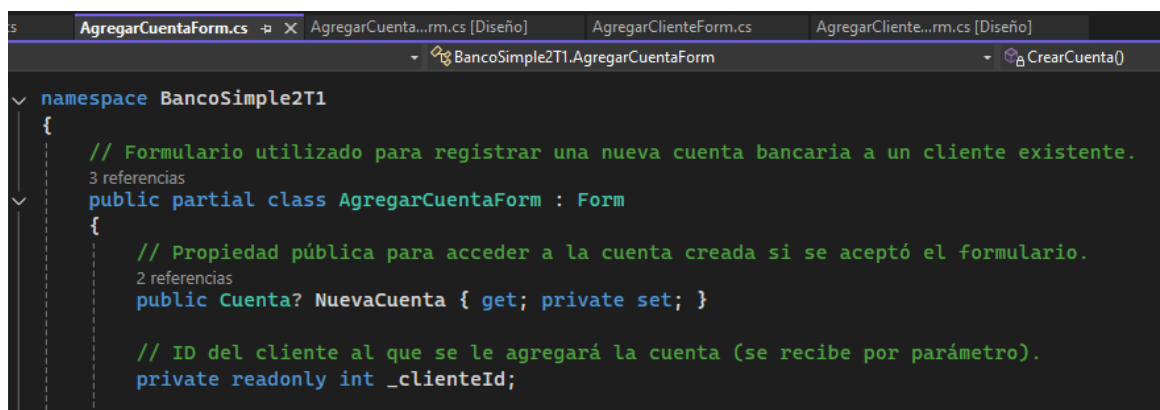
Una vez abrí la clase AgregarCuentaForm, me di cuenta de que tenía varios detalles que podían mejorarse:

1. No había manejo de errores en el botón Aceptar, lo que puede causar cierres inesperados.
2. La validación estaba integrada en el evento y no en un método separado, dificultando mantenimiento.
3. No existía opción para cancelar la operación; por eso agregué un botón Cancelar con su evento.
4. El campo `_clienteId` no era readonly, aunque no cambia después de asignarse.
5. La propiedad `NuevaCuenta` no permitía valores nulos, lo que podía generar confusión.
6. No se limpiaban espacios en el número de cuenta, lo que puede causar inconsistencias.
7. Faltaban comentarios que expliquen el código, dificultando su comprensión y mantenimiento.

Inicio de los cambios:

La clase `AgregarCuentaForm` no tenía comentarios que expliquen su función, también agregamos un comentario al inicio para describir el propósito del formulario. Luego se marcó como nullable (`Cuenta?`) porque antes de crear la cuenta la propiedad puede ser null. Esto evita confusión y posibles errores al acceder a ella.

Se añadió `readonly` para garantizar que el ID no se modifique accidentalmente después de la inicialización en el constructor. Además, agregamos comentario descriptivo.



```
namespace BancoSimple2T1
{
    // Formulario utilizado para registrar una nueva cuenta bancaria a un cliente existente.
    3 referencias
    public partial class AgregarCuentaForm : Form
    {
        // Propiedad pública para acceder a la cuenta creada si se aceptó el formulario.
        2 referencias
        public Cuenta? NuevaCuenta { get; private set; }

        // ID del cliente al que se le agregará la cuenta (se recibe por parámetro).
        private readonly int _clienteId;
    }
}
```

Se separó la validación en un método propio (`ValidarFormulario()`), mejorando legibilidad y reutilización. Hicimos la creación de la cuenta se movió a `CrearCuenta()`, separando responsabilidades.



```
AgregarCuentaForm.cs  X AgregarCuenta...rm.cs [Diseño] AgregarClienteForm.cs AgregarCliente...rm...
BancoSimple2T1.AgregarCuentaForm

// Constructor que recibe el ID del cliente y prepara el formulario.
1 referencia
public AgregarCuentaForm(int clienteId)
{
    InitializeComponent();
    _clienteId = clienteId;
}

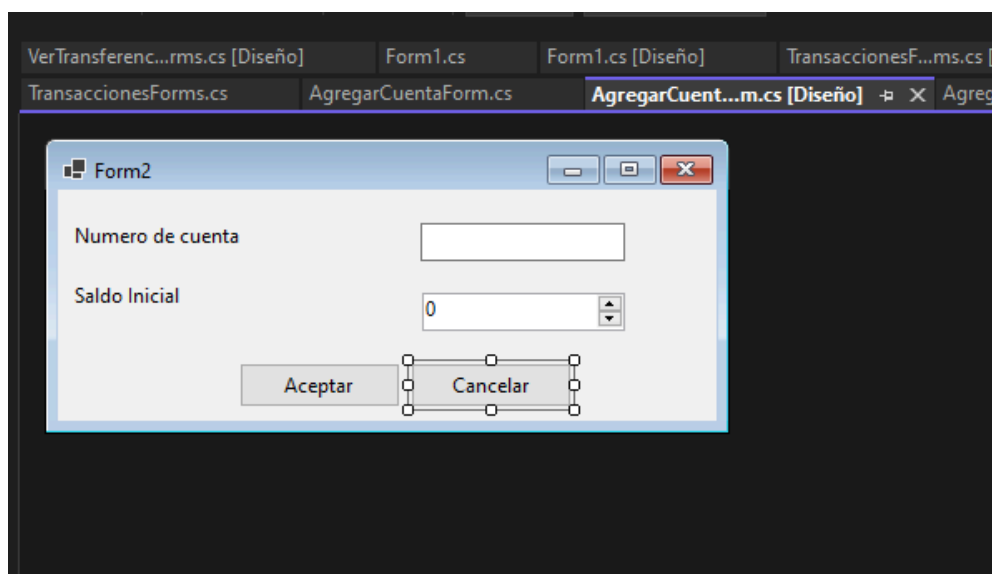
// Evento que se ejecuta al hacer clic en el botón "Aceptar".
// Valida el formulario, crea la cuenta y cierra con resultado OK.
1 referencia
private void btnAceptar_Click(object sender, EventArgs e)
{
    try
    {
        // Validación de campos obligatorios.
        if (!ValidarFormulario())
            return;

        // Se crea la cuenta y se cierra el formulario exitosamente.
        CrearCuenta();
        DialogResult = DialogResult.OK;
        Close();
    }
}
```

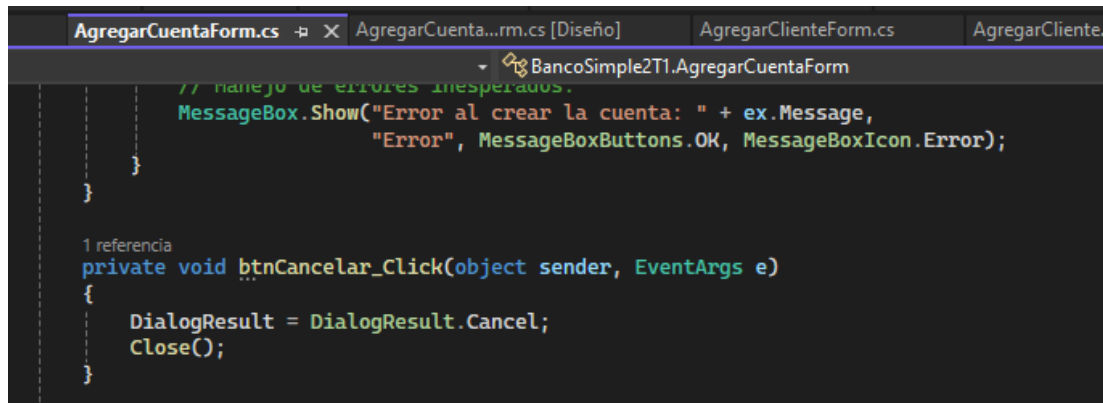
```
}
catch (Exception ex)
{
    // Manejo de errores inesperados.
    MessageBox.Show("Error al crear la cuenta: " + ex.Message,
        "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
}
}
```

Se añadió un bloque try-catch para manejar excepciones inesperadas y mostrar mensajes amigables al usuario. Esto evita cierres abruptos y mejora la robustez y se agregaron comentarios explicativos para mayor claridad.

Luego, se añadió un nuevo evento “Cancelar” ya que no existía código para cancelar.



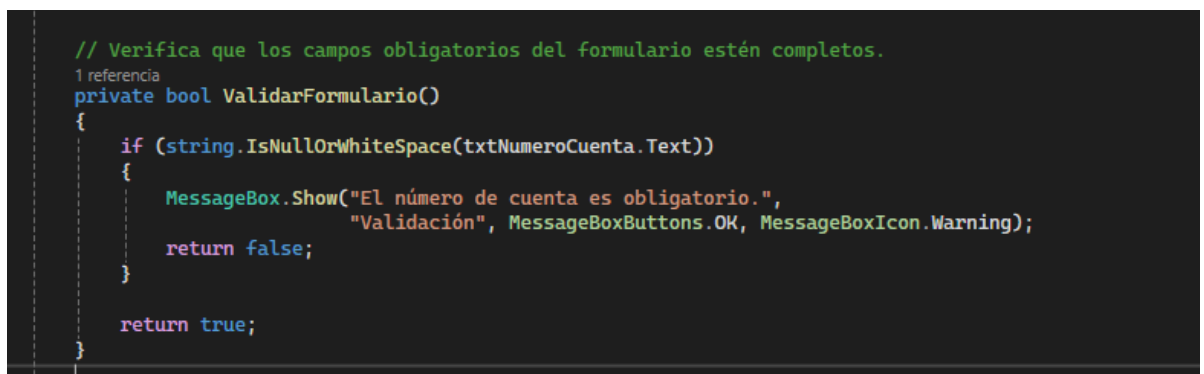
Se agregó este método para manejar el botón Cancelar, permitiendo que el usuario cierre el formulario sin crear una cuenta, mejorando la experiencia y control de flujo.



```
AgregarCuentaForm.cs  AgregarCuenta...rm.cs [Diseño]  AgregarClienteForm.cs  AgregarCliente...
BancoSimple2T1.AgregarCuentaForm
// Manejo de errores inesperados.
MessageBox.Show("Error al crear la cuenta: " + ex.Message,
               "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
}
}

1 referencia
private void btnCancelar_Click(object sender, EventArgs e)
{
    DialogResult = DialogResult.Cancel;
    Close();
}
```

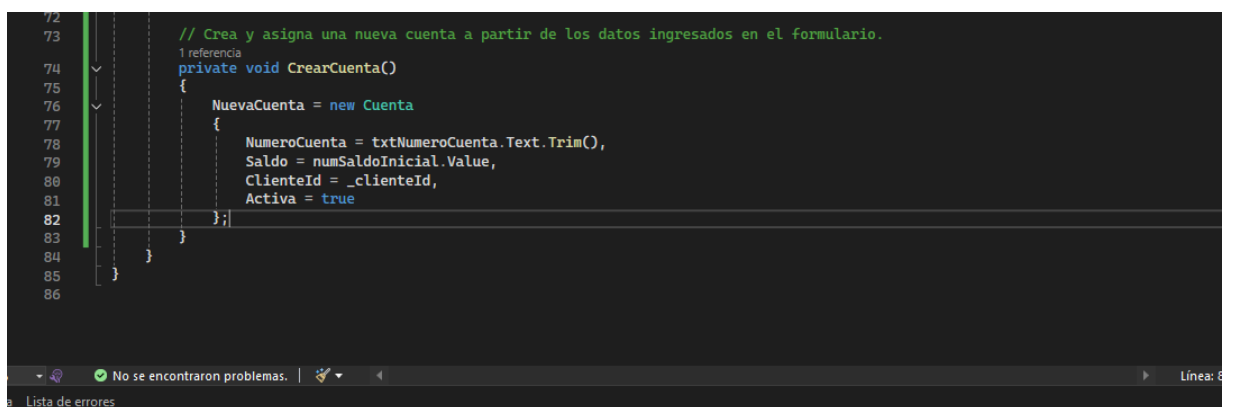
La validación en el código original estaba dentro de btnAceptar\_Click. Hicimos una extracción de la validación a un método aparte para mejor organización y posible reutilización.



```
// Verifica que los campos obligatorios del formulario estén completos.
1 referencia
private bool ValidarFormulario()
{
    if (string.IsNullOrWhiteSpace(txtNumeroCuenta.Text))
    {
        MessageBox.Show("El número de cuenta es obligatorio.",
                        "Validación", MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return false;
    }

    return true;
}
```

Hicimos la creación directa de la cuenta en el evento. Se movió la creación del objeto Cuenta a un método propio para desacoplar la lógica de creación del flujo del evento y se añadió Trim() para limpiar espacios en blanco en el número de cuenta, previniendo errores de formato.



```
72 // Crea y asigna una nueva cuenta a partir de los datos ingresados en el formulario.
73
74 1 referencia
75 private void CrearCuenta()
76 {
77     NuevaCuenta = new Cuenta
78     {
79         NumeroCuenta = txtNumeroCuenta.Text.Trim(),
80         Saldo = numSaldoInicial.Value,
81         ClienteId = _clienteId,
82         Activa = true
83     };
84 }
85
86
```

Lista de errores

#### 4. Clase [Transacciones.cs](#)

```
using BancoSimple2T1.Data;  
using Microsoft.EntityFrameworkCore;  
using System;  
using System.Windows.Forms;
```

using: importa bibliotecas necesarias para que el código funcione correctamente.

BancoSimple2T1.Data: contiene el BancoSimpleContext que permite acceder a la base de datos.

Microsoft.EntityFrameworkCore: permite usar métodos de EF Core como Include(), FirstOrDefault().

System: funciones básicas como decimal, Exception, etc.

System.Windows.Forms: clases para construir interfaces gráficas.

```
namespace BancoSimple2T1  
{  
    public partial class TransaccionesForms : Form  
    {
```

Define el formulario TransaccionesForms dentro del namespace del proyecto.

**partial** significa que esta clase está dividida: la otra parte está en el archivo generado por el diseñador de Visual Studio ([TransaccionesForms.Designer.cs](#)). Hereda de Form, lo que permite que sea una ventana gráfica.

```
public decimal Monto { get; private set; }
```

Propiedad pública que almacena el monto a transferir.

Solo se puede modificar desde esta clase (private set), pero puede ser leída desde otra clase (por ejemplo, la que abre este formulario).

```
private readonly int _cuentaOrigenId;  
private readonly int _cuentaDestinoId;
```

Guarda los IDs de las cuentas participantes.

**readonly**: garantiza que no cambian después de ser asignados en el constructor (más seguro y predecible).

```
private readonly BancoSimpleContext _dbContext;
```

BancoSimpleContext es la clase que permite acceder a las tablas de la base de datos. Se inicializa solo una vez en el constructor para usar durante toda la vida del formulario.

```
public TransaccionesForms(int cuentaOrigenId, int cuentaDestinoId)
```

Constructor que recibe dos IDs: el de la cuenta de origen y el de destino. Son necesarios para saber entre qué cuentas se moverá dinero.

```
_cuentaOrigenId = cuentaOrigenId;  
_cuentaDestinoId = cuentaDestinoId;
```

Asigna los valores recibidos a los campos privados para usarlos después.

```
_dbContext = new BancoSimpleContext();
```

Crea una instancia del contexto de Entity Framework. Permite hacer consultas a las tablas del modelo de datos.

```
CargarInformacionCuentas();
```

Llama a un método que recupera los datos de las cuentas y los muestra en pantalla.

```
private void CargarInformacionCuentas()  
{  
    try
```

Es un método privado llamado en el constructor. Se usa try para manejar cualquier error durante el acceso a la base de datos.

```
var cuentaOrigen = _dbContext.Cuenta  
    .Include(c => c.cliente)  
    .FirstOrDefault(c => c.CuentaId == _cuentaOrigenId);
```

Recupera una cuenta usando su ID desde la base de datos.

.Include(c => c.cliente): hace que también se cargue el objeto relacionado "cliente" (relación de EF).

.FirstOrDefault(...): retorna el primer resultado o null si no encuentra ninguno (evita errores).

```
var cuentaDestino = _dbContext.Cuenta
    .Include(c => c.cliente)
    .FirstOrDefault(c => c.CuentaId == _cuentaDestinoId);
```

Repite la lógica para la cuenta de destino.

```
if (cuentaOrigen == null || cuentaDestino == null)
```

Valida que ambas cuentas hayan sido encontradas. Si alguna es null, se muestra un error.

```
MessageBox.Show("Error al cargar las cuentas.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
Close(); // Cierra el formulario si hay un problema
return;
```

Muestra mensaje de error y cierra el formulario si hay un problema.

```
lblOrigen.Text = $"Nombre: {cuentaOrigen.cliente.Nombre} - Cuenta: {cuentaOrigen.NumeroCuenta}";
lblDestino.Text = $"Nombre: {cuentaDestino.cliente.Nombre} - Cuenta: {cuentaDestino.NumeroCuenta}";
lblDisponible.Text = $"Saldo Disponible: {cuentaOrigen.Saldo:C}";
```

Muestra los datos en las etiquetas del formulario.

:C es un formato para mostrar números como moneda

```
catch (Exception ex)
{
    MessageBox.Show($"Error al cargar información: {ex.Message}", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
    Close(); // Cerramos el formulario para evitar errores posteriores
}
```

Captura cualquier error inesperado (por ejemplo, error de conexión a la base de datos).  
Muestra un mensaje de error detallado y cierra el formulario.

```
private void btnAceptar_Click(object sender, EventArgs e)
{
```

Este método se ejecuta cuando se hace clic en el botón "Aceptar".

```
if (decimal.TryParse(txtSaldo.Text, out decimal monto))
```

Intenta convertir el texto ingresado en la caja txtSaldo a decimal. Si falla (por ejemplo, si escribieron letras), no lanza excepción, solo entra al else.

```
if (monto <= 0)
```

Verifica que el monto ingresado sea mayor a cero.

```
var cuentaOrigen = _dbContext.Cuenta.Find(_cuentaOrigenId);
```

Busca la cuenta por su ID de forma directa. No usa Include() porque no se necesita al cliente aquí, solo el saldo.

```
if (cuentaOrigen == null || cuentaOrigen.Saldo < monto)
```

Verifica que exista la cuenta y que el saldo sea suficiente.

```
MessageBox.Show("Saldo insuficiente en la cuenta de origen.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Warning);  
return;
```

Si no hay suficiente saldo, se avisa y se cancela el intento.

```
Monto = monto;  
DialogResult = DialogResult.OK;  
Close(); // Cierra el formulario y devuelve resultado
```

Si todo es correcto, se guarda el monto y se cierra el formulario con resultado OK.

```
else  
{  
    MessageBox.Show("Ingrese un monto válido", "Error", MessageBoxButtons.OK, MessageBoxIcon.Warning);  
}
```

Si el texto no es un número válido, se muestra un mensaje de advertencia.

```
private void btnCancelar_Click(object sender, EventArgs e)  
{  
    DialogResult = DialogResult.Cancel;  
    Close(); // Cancela y cierra  
}
```

Cierra el formulario sin hacer nada.

Devuelve el resultado Cancel para que el formulario que lo llama pueda detectarlo.

## Cambios en Transacción.cs

Como primer paso clonamos el repositorio con nuestra rama hecha del repositorio compartido

```
Maria Garcia@WINDOWS-UMBIRO6 MINGW64 ~/Documents
$ cd ~/Documents

Maria Garcia@WINDOWS-UMBIRO6 MINGW64 ~/Documents
$ git clone https://github.com/AlidElgado21/MejoraBancoSimple2T1.git
Cloning into 'MejoraBancoSimple2T1'...
remote: Enumerating objects: 288, done.
remote: Counting objects: 100% (288/288), done.
remote: Compressing objects: 100% (170/170), done.
remote: Total 288 (delta 105), reused 274 (delta 96), pack-reused 0 (from 0)
Receiving objects: 100% (288/288), 13.24 MiB | 7.27 MiB/s, done.
Resolving deltas: 100% (105/105), done.

Maria Garcia@WINDOWS-UMBIRO6 MINGW64 ~/Documents
$ cd MejoraBancoSimple2T1

Maria Garcia@WINDOWS-UMBIRO6 MINGW64 ~/Documents/MejoraBancoSimple2T1 (master)
$ git checkout Rama-Cambios-Maria
branch 'Rama-Cambios-Maria' set up to track 'origin/Rama-Cambios-Maria'.
Switched to a new branch 'Rama-Cambios-Maria'

Maria Garcia@WINDOWS-UMBIRO6 MINGW64 ~/Documents/MejoraBancoSimple2T1 (Rama-Cambios-Maria)
$ git branch
* Rama-Cambios-Maria
  master
```

A partir de ahí trabajamos en nuestros cambios dentro de Visual Studio 2022

1. El archivo Transacción.cs contenía una clase con propiedades básicas pero sin comentarios ni métodos que validaron la información. El código era funcional, pero poco documentado y sin validaciones.

Se añadieron comentarios XML en cada propiedad y la clase para mejorar la documentación del código y facilitar su entendimiento y mantenimiento.

```
2 Referencias
public class Transaccion
{
    /// <summary>
    /// Identificador único de la transacción.
    /// </summary>
    0 referencias
    public int TransaccionId { get; set; }
```

2. Inclusión del método Validar()

Se creó un método llamado validar() que verifica:

- Que el monto sea mayor que cero.

- Que al menos una de las cuentas (origen o destino) esté definida.
- Que las cuentas de origen y destino no sean iguales.

```
0 referencias
public bool Validar()
{
    if (Monto <= 0)
        return false;

    if (CuentaOrigenId == null && CuentaDestinoId == null)
        return false;

    if (CuentaOrigenId != null && CuentaDestinoId != null && CuentaOrigenId == CuentaDestinoId)
        return false;

    return true;
}
```

### 3. Código final mejorado

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace BancoSimple2T1.Models
{
    /// <summary>
    /// Representa una transacción bancaria entre dos cuentas.
    /// </summary>
    2 referencias
    public class Transaccion
    {
        /// <summary>
        /// Identificador único de la transacción.
        /// </summary>
        0 referencias
        public int TransaccionId { get; set; }

        /// <summary>
        /// Monto transferido en la transacción.
        /// </summary>
        2 referencias
        public decimal Monto { get; set; }

        /// <summary>
        /// Fecha y hora en que se realizó la transacción.
        /// </summary>
        1 referencia
        public DateTime Fecha { get; set; } = DateTime.Now;
    }
}
```



```

/// <summary>
/// Descripción opcional de la transacción.
/// </summary>
1 referencia
public string Descripcion { get; set; }

/// <summary>
/// ID de la cuenta de origen (puede ser null si no aplica).
/// </summary>
4 referencias
public int? CuentaOrigenId { get; set; }

/// <summary>
/// ID de la cuenta de destino (puede ser null si no aplica).
/// </summary>
4 referencias
public int? CuentaDestinoId { get; set; }

/// <summary>
/// Valida que la transacción tenga datos correctos y consistentes.
/// </summary>
/// <returns>True si la transacción es válida, de lo contrario False.</returns>
0 referencias
public bool Validar()
{
    if (Monto <= 0)
        return false;

    if (CuentaOrigenId == null && CuentaDestinoId == null)
        return false;

    if (CuentaOrigenId != null && CuentaDestinoId != null && CuentaOrigenId == CuentaDestinoId)
        return false;

    return true;
}

```

- Y como último paso, guardamos todos nuestros cambios en el repositorio por medio de git bash here

```

Maria Garcia@WINDOWS-UMBIRQ6 MINGW64 ~/Documents
$ cd ~/Documents

Maria Garcia@WINDOWS-UMBIRQ6 MINGW64 ~/Documents
$ git clone https://github.com/AlidElgado21/MejoraBancoSimple2T1.git
Cloning into 'MejoraBancoSimple2T1'...
remote: Enumerating objects: 288, done.
remote: Counting objects: 100% (288/288), done.
remote: Compressing objects: 100% (170/170), done.
remote: Total 288 (delta 105), reused 274 (delta 96), pack-reused 0 (from 0)
Receiving objects: 100% (288/288), 13.24 MiB | 7.27 MiB/s, done.
Resolving deltas: 100% (105/105), done.

Maria Garcia@WINDOWS-UMBIRQ6 MINGW64 ~/Documents
$ cd MejoraBancoSimple2T1

Maria Garcia@WINDOWS-UMBIRQ6 MINGW64 ~/Documents/MejoraBancoSimple2T1 (master)
$ git checkout Rama-Cambios-Maria
branch 'Rama-Cambios-Maria' set up to track 'origin/Rama-Cambios-Maria'.
Switched to a new branch 'Rama-Cambios-Maria'

Maria Garcia@WINDOWS-UMBIRQ6 MINGW64 ~/Documents/MejoraBancoSimple2T1 (Rama-Cambios-Maria)
$ git branch
* Rama-Cambios-Maria
  master

Maria Garcia@WINDOWS-UMBIRQ6 MINGW64 ~/Documents/MejoraBancoSimple2T1 (Rama-Cambios-Maria)
$ git add BancoSimple2T1/Models/Transaccion.cs

Maria Garcia@WINDOWS-UMBIRQ6 MINGW64 ~/Documents/MejoraBancoSimple2T1 (Rama-Cambios-Maria)
$ git commit -m "Cambios por Maria: Agregue metodo de validacion y comentarios XML en Transaccion.cs"
[Rama-Cambios-Maria 9e3fa4b] Cambios por Maria: Agregue metodo de validacion y comentarios XML en Transaccion.cs
1 file changed, 43 insertions(+), 2 deletions(-)

Maria Garcia@WINDOWS-UMBIRQ6 MINGW64 ~/Documents/MejoraBancoSimple2T1 (Rama-Cambios-Maria)
$ git push origin Rama-Cambios-Maria
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 8 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (5/5), 987 bytes | 493.00 KiB/s, done.
Total 5 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
to https://github.com/AlidElgado21/MejoraBancoSimple2T1.git
88ee7e7..9e3fa4b Rama-Cambios-Maria -> Rama-Cambios-Maria

Maria Garcia@WINDOWS-UMBIRQ6 MINGW64 ~/Documents/MejoraBancoSimple2T1 (Rama-Cambios-Maria)
$ |

```

## 6. CLASE FORMS1

```
private bool SeleccionarFila(DataGridView dgv, out DataGridViewRow fila)
```

Valida que haya una fila seleccionada exactamente, para evitar repetir esa lógica en varios botones.

```
private void MostrarMensaje(string mensaje)
```

Evita escribir `MessageBox.Show(...)` cada vez. Mejora la legibilidad y consistencia.

```
if (cuentaOrigen is null || cuentaDestino is null)
    throw new InvalidOperationException("Una o ambas cuentas no existen.");

if (!cuentaOrigen.Activa || !cuentaDestino.Activa)
    throw new InvalidOperationException("Una o ambas cuentas están inactivas.");
```

Usar `InvalidOperationException` ayuda a indicar un mal uso lógico del sistema.

```
transaccion.Commit();
```

```
transaccion.Rollback();
```

Para asegurar que la operación sea **atómica**.

## 7. Clase: Ver transferencia

```
MINGW64:/c/Users/brand/Downloads/MejoraBancoSimple2T1-master

Dan CM@MATDAN MINGW64 ~/Downloads/MejoraBancoSimple2T1-master
$ git init
Initialized empty Git repository in C:/Users/brand/Downloads/MejoraBancoSimple2T1-master/.git/

Dan CM@MATDAN MINGW64 ~/Downloads/MejoraBancoSimple2T1-master (master)
$ git remote add origin https://github.com/Alidelgado21/MejoraBancoSimple2T1.git

Dan CM@MATDAN MINGW64 ~/Downloads/MejoraBancoSimple2T1-master (master)
$ git fetch
remote: Enumerating objects: 231, done.
remote: Counting objects: 100% (231/231), done.
remote: Compressing objects: 100% (137/137), done.
remote: Total 231 (delta 77), reused 231 (delta 77), pack-reused 0 (from 0)
Receiving objects: 100% (231/231), 13.09 MiB | 3.74 MiB/s, done.
Resolving deltas: 100% (77/77), done.
From https://github.com/Alidelgado21/MejoraBancoSimple2T1
* [new branch]      Rama-Cambios-Alicia -> origin/Rama-Cambios-Alicia
* [new branch]      Rama-Cambios-Anthony -> origin/Rama-Cambios-Anthony
* [new branch]      Rama-Cambios-Genesis -> origin/Rama-Cambios-Genesis
* [new branch]      Rama-Cambios-Maria -> origin/Rama-Cambios-Maria
* [new branch]      Rama-Cambios-yaiza -> origin/Rama-Cambios-yaiza
* [new branch]      master -> origin/master

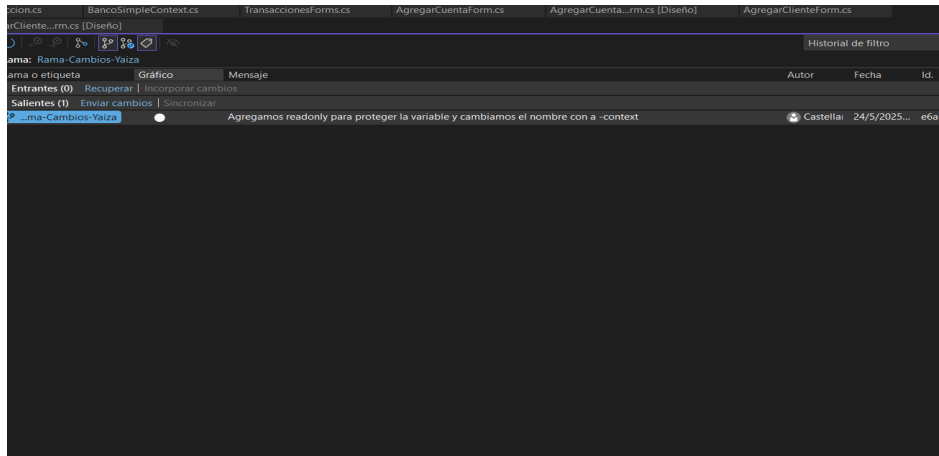
Dan CM@MATDAN MINGW64 ~/Downloads/MejoraBancoSimple2T1-master (master)
$ git checkout -b Rama-Cambios-yaiza origin/Rama-Cambios-yaiza
branch 'Rama-Cambios-yaiza' set up to track 'origin/Rama-Cambios-yaiza'.
Switched to a new branch 'Rama-Cambios-yaiza'

Dan CM@MATDAN MINGW64 ~/Downloads/MejoraBancoSimple2T1-master (Rama-Cambios-yaiza)
$ |
```

Clonamos el repositorio y empezamos a trabajar con los cambios de la clase.

Cambio 1: Se agrega **readonly** para proteger la variable contra reasignación.

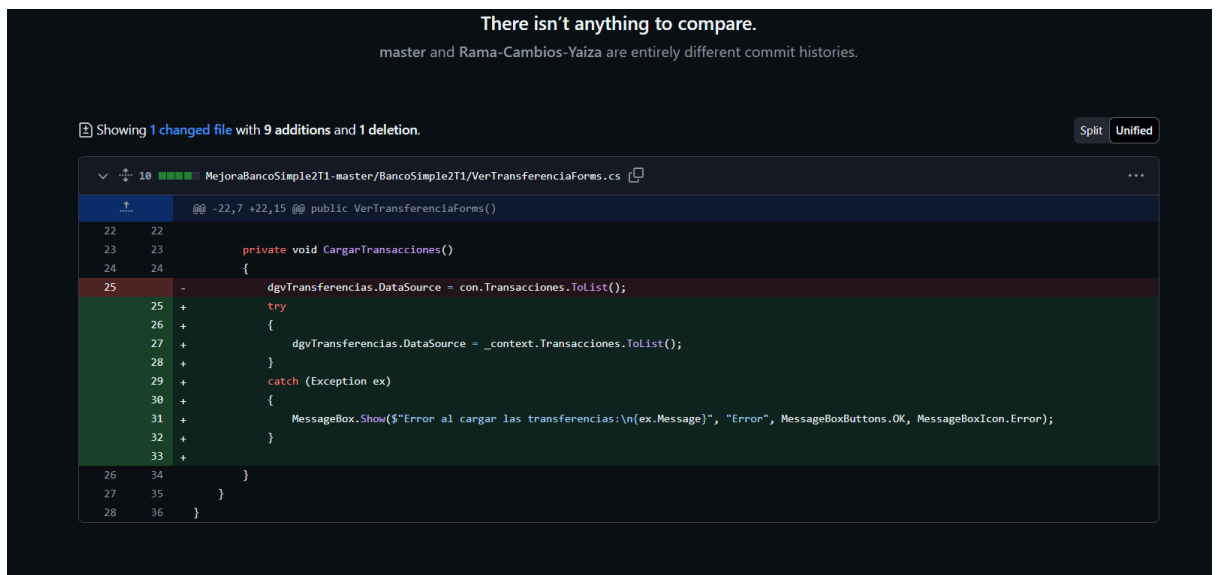
Se cambia el nombre de **con** a **\_context**, siguiendo las convenciones de nomenclatura en C# para campos privados.



Cambio 2: Se renombra el método para reflejar mejor que se están cargando transacciones, no solo transferencias (si este es el caso lógico).

Antes: CargarTransferencias();

Después: CargarTransacciones();



Cambio 3: Se maneja cualquier excepción que pueda ocurrir al consultar.

```
1 file changed +9 -1 lines changed
MejoraBancoSimple2T1-master/BancoSimple2T1/VerTransferenciaForms.cs
@@ -22,7 +22,15 @@ public VerTransferenciaForms()
22 22
23 23     private void CargarTransacciones()
24 24     {
25 -     dgvTransferencias.DataSource = con.Transacciones.ToList();
25 +     try
26 +     {
27 +     dgvTransferencias.DataSource = _context.Transacciones.ToList();
28 +     }
29 +     catch (Exception ex)
30 +     {
31 +     MessageBox.Show($"Error al cargar las transferencias:\n{ex.Message}", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
32 +     }
33 +     }
26 34     }
27 35 }
28 36 }
```