

INSTITUTO TECNICO RICALDONE

MANUAL TECNICO

Credaris

Sistema de gestión de préstamos



CRE-DARIS

INTEGRANTES:

Franklin Eduardo Villanueva Vasquez

CODIGO:

20250827

Jonathan Samuel Rivera Rojas

20250236

Rodríguez López Josué Nahum

20250764

Leonardo Daniel Romero Valladares

20240848

INDICE

Contenido

1. Introducción	3
2. Tecnologías y herramientas	4
3. Estructura de la base de datos	6
Modelo relacional.....	6
4. Diccionario de datos	7
5. Arquitectura de Software	8
6. Estructura del proyecto	10
Estructura del backend.....	11
Estructura del Frontend.....	12
Poppers.....	14
Resources.....	15
7. Diseño de la aplicación	16
Paleta de colores.....	16
Aplicación en los diferentes elementos.....	16
Tipos y tamaños de fuentes.....	18
Tipos y dimensión de imágenes utilizadas.....	18
Tamaños de pantalla soportados.....	19
9. Requerimientos de hardware y software.....	20
Localmente.....	20
Servidor.....	21
10. Instalación y configuración.....	22

1. INTRODUCCIÓN

El presente manual técnico está dirigido a desarrolladores, administradores de sistemas. Su objetivo es proporcionar una guía clara y detallada sobre la arquitectura, tecnologías, herramientas y buenas prácticas implementadas en el desarrollo de la plataforma, asegurando que su despliegue y administración se realicen de manera eficiente y segura.

El objetivo del sistema es optimizar el proceso de otorgar, control y seguimiento de préstamos dentro de la empresa, eliminando el uso de métodos manuales o poco prácticos que pueden generar errores y pérdida de información.

Este documento tiene como propósito proporcionar una guía clara y detallada sobre la arquitectura del software, las tecnologías utilizadas, la estructura de la base de datos y las buenas prácticas implementadas en el desarrollo. De esta forma se garantiza que el sistema pueda desplegarse, administrarse y mantenerse de manera eficiente, segura y escalable.

Este manual técnico busca ser una guía completa y precisa para comprender todos los aspectos técnicos del proyecto Credaris, facilitando su desarrollo, mantenimiento y futuras actualizaciones.

2. TECNOLOGIAS Y HERRAMIENTAS.

Tecnologías y herramientas que usamos:

- IDE

Entorno de Desarrollo Integrado

VISUAL STUDIO 2022

Uso: Usado para desarrollar la aplicación principal en C#, manejar formularios y conectar con SQL Server.

Sitio web: <https://visualstudio.microsoft.com/es/>

- Frontend

Framework

C# Windows Forms

Uso: Se usa para crear la interfaz gráfica de la aplicación.

Sitio Web: <https://learn.microsoft.com/es-es/dotnet/desktop/winforms/overview>

- Backend

Lenguaje de programación:

C#

Uso: Utilizado para el desarrollo de la aplicación de escritorio, creando la lógica del sistema, la interfaz de usuario y la conexión con la base de datos.

Sitio web: <https://learn.microsoft.com/es-es/dotnet/csharp/>

Base de Datos

SQL Server Management Studio 20 (SSMS)

Uso: Usado para administrar la base de datos, crear tablas, definir relaciones, ejecutar consultas SQL y verificar la integridad de los datos.

Sitio Web: <https://learn.microsoft.com/es-es/ssms/release-notes-20>

- Apoyo

Herramienta De Diagramas

Lucidchart

Uso: Utilizada para diseñar el modelo relacional de la base de datos y representar gráficamente las relaciones entre las tablas, facilitando la comprensión del diseño.

Sitio Web: <https://www.lucidchart.com/>

Control de versiones

GitHub

Uso: Implementado para almacenar el proyecto en la nube, mantener un historial de cambios y facilitar el trabajo colaborativo en caso de que otros desarrolladores participen.

Sitio Web: <https://github.com/>

Herramienta de Diseño

Figma

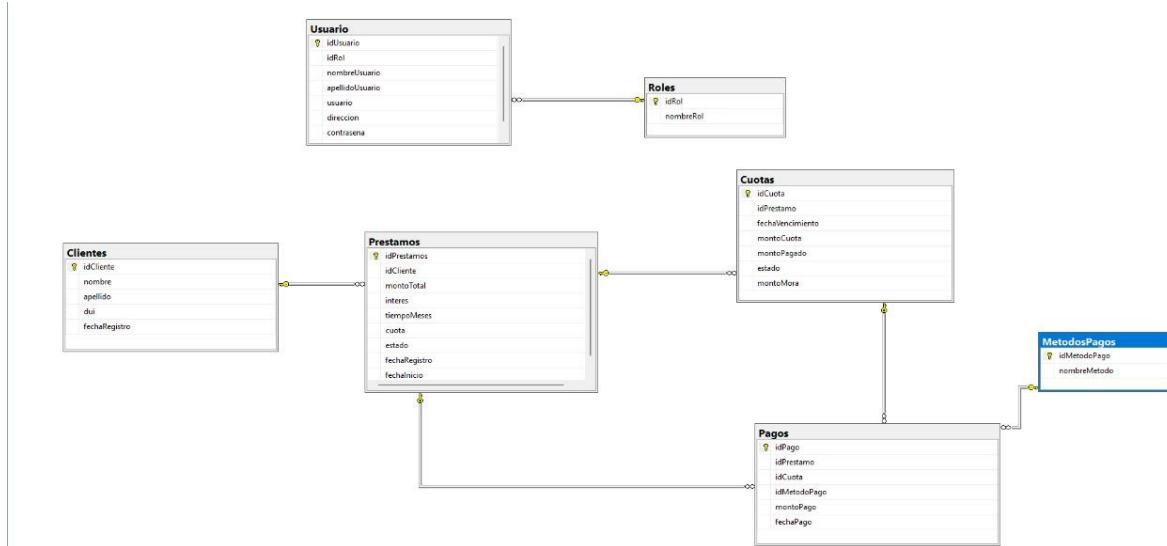
Uso: Usado para diseñar la aplicación antes de empezar a desarrollarla.

Sitio Web: <https://help.figma.com/hc/es-419/articles/14563969806359--Qué-es-Figma>

3. ESTRUCTURA DE LA BASE DE DATOS.

Modelo relacional

Diagrama de la Base de Datos:



Nota: En el siguiente enlace se encuentra la imagen con mejor calidad:

https://drive.google.com/file/d/1ga7jWImVNz6s4b3iGxjznT_HzNNHwzp8/view?usp=sharing

- Script de la Base de Datos:

https://drive.google.com/file/d/1Od7Ip7WS2_4P_OWRFDLSMXY4AbKo4klY/view?usp=sharing

Nota: es posible la foto no cargue en vista previa ya que es algo pesada por lo que se tiene que descargar.

4. DICCIONARIO DE DATOS.

TABLA 1				
Nombre del campo	Tipo de dato	Tamaño	Restriccion	Descripcion
idRol	INT	---	PRIMARY KEY IDENTITY (1,1)	En este campo se genera el id del Rol
nombreRol	VARCHAR		50 NOT NULL	En este campo se ingresa el nombre del rol
TABLA 2				
Nombre del campo	Tipo de dato	Tamaño	Restriccion	Descripcion
idMetodoPago	INT	---	PRIMARY KEY IDENTITY (1,1)	En este campo se genera el id del Metodo de pago
nombreMetodo	VARCHAR		50 NOT NULL, UNIQUE	En este campo se ingresa el nombre del metodo de pago
TABLA 3				
Nombre del campo	Tipo de dato	Tamaño	Restriccion	Descripcion
idUser	INT	---	PRIMARY KEY IDENTITY (1,1)	En este campo se genera el id del Usuario
idRol	INT		NOT NULL, FOREIGN KEY	En este campo se genera el id del Rol
nombreUsuario	NVARCHAR		50 NOT NULL	En este campo se ingresa el nombre del usuario
apellidoUsuario	NVARCHAR		50 NOT NULL	En este campo se ingresa el apellido del usuario
usuario	NVARCHAR		15 NOT NULL, UNIQUE	En este campo se ingresa el nombre del usuario para el login
direccion	NVARCHAR		200 NOT NULL	En este campo se ingresa la dirección del usuario
contraseña	NVARCHAR		250 NOT NULL	En este campo se ingresa la contraseña del usuario.
fechaRegistroUsuario	DATETIME	-----	DEFAULT GETDATE()	En este campo se ingresa la fecha de registro del usuario.
TABLA 4				
Nombre del campo	Tipo de dato	Tamaño	Restriccion	Descripcion
idCliente	INT	---	PRIMARY KEY IDENTITY (1,1)	En este campo se genera el id del Cliente
nombre	NVARCHAR		50 NOT NULL	En este campo se ingresa el nombre del cliente
apellido	NVARCHAR		50 NOT NULL	En este campo se ingresa el apellido del cliente
dui	VARCHAR		10 NOT NULL, UNIQUE	En este campo se ingresa el carnet del usuario
fechaRegistro	DATE	-----	DEFAULT GETDATE()	En este campo se ingresa la fecha de registro del cliente.
TABLA 5				
Nombre del campo	Tipo de dato	Tamaño	Restriccion	Descripcion
idPrestamo	INT	---	PRIMARY KEY IDENTITY (1,1)	En este campo se genera el id del Prestamo
idCliente	INT	---	NOT NULL, FOREIGN KEY ON DELETE CASCADE	En este campo se genera el id del cliente
montoTotal	DECIMAL	12,2	NOT NULL	En este campo se ingresa el monto total del prestamo
interes	DECIMAL	5,2	NOT NULL	En este campo se ingresa el interes del prestamo
tiempoMeses	INT	---	NOT NULL	En este campo se ingresa el plazo a meses del prestamo
cuota	DECIMAL	12,2	NOT NULL	En este campo se ingresa la cuota del prestamo
estado	INT		NOT NULL	En este campo se ingresa el estado del prestamo
fechainicio	DATETIME	-----	NOT NULL, DEFAULT GETDATE()	En este campo se ingresa la fecha de inicio del prestamo
fechaRegistro	DATE	-----	NOT NULL	En este campo se ingresa la fecha de registro del prestamo
TABLA 6				
Nombre del campo	Tipo de dato	Tamaño	Restriccion	Descripcion
idCuota	INT	---	PRIMARY KEY IDENTITY (1,1)	En este campo se genera el id de la cuota
idPrestamo	INT	---	NOT NULL, FOREIGN KEY ON DELETE CASCADE	En este campo se genera el id del prestamo
fechaVencimiento	DATE	---	NOT NULL	En este campo se ingresa la fecha de vencimiento
montoCuota	DECIMAL	12,2	NOT NULL	En este campo se ingresa el monto de la cuota del prestamo
montoPagado	DECIMAL	12,2	NOT NULL	En este campo se ingresa el monto pagado del prestamo
estado	INT		NOT NULL	En este campo se ingresa el estado de la cuota
montoMora	DECIMAL	12,2	NOT NULL	En este campo se ingresa el monto en mora del prestamo
TABLA 7				
Nombre del campo	Tipo de dato	Tamaño	Restriccion	Descripcion
idPago	INT	---	PRIMARY KEY IDENTITY (1,1)	En este campo se genera el id del pago
idPrestamo	INT	---	NOT NULL, FOREIGN KEY	En este campo se genera el id del Prestamo
idCuenta	INT	---	NOT NULL, FOREIGN KEY ON DELETE CASCADE	En este campo se genera el id de la cuenta
idMetodoPago	INT	---	NOT NULL, FOREIGN KEY	En este campo se genera el id del Metodo de pago
montoPago	DECIMAL	12,2	NOT NULL	En este campo se ingresa el monto a pago realizado
fechaPago	DATETIME	---	NOT NULL	En este campo se ingresa la fecha en que se realizo el pago

En este enlace encontrara El Diccionario de Datos con mayor calidad

TABLA 1-4

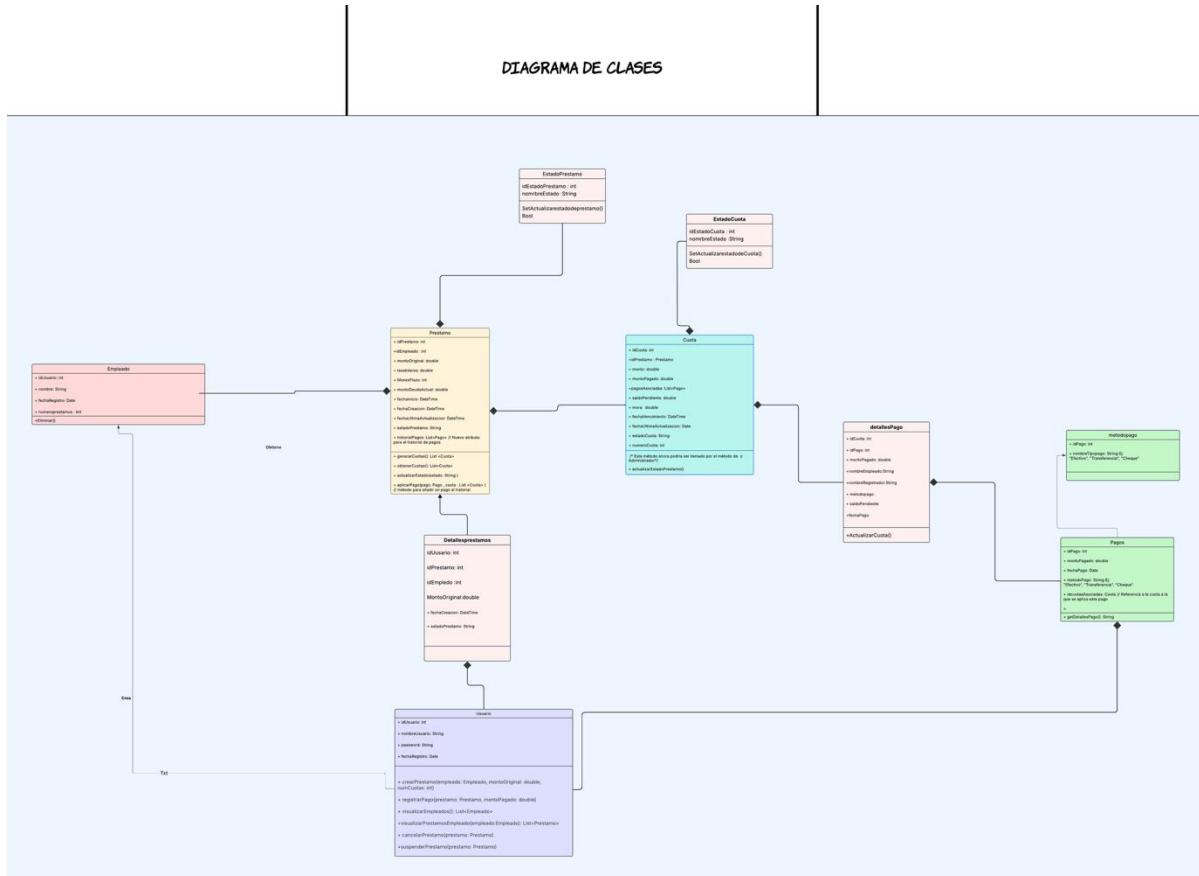
https://drive.google.com/file/d/1tIfyu4ZLtapDb_q-DtUhbkvhw5xd3ftC/view?usp=sharing

TABLA 5-7

<https://drive.google.com/file/d/1Yv4jjD-bBPTOhnKDK3hWJWkMygmHtg5X/view?usp=sharing>

5. ARQUITECTURA DE SOFTWARE.

Diagrama de Clases

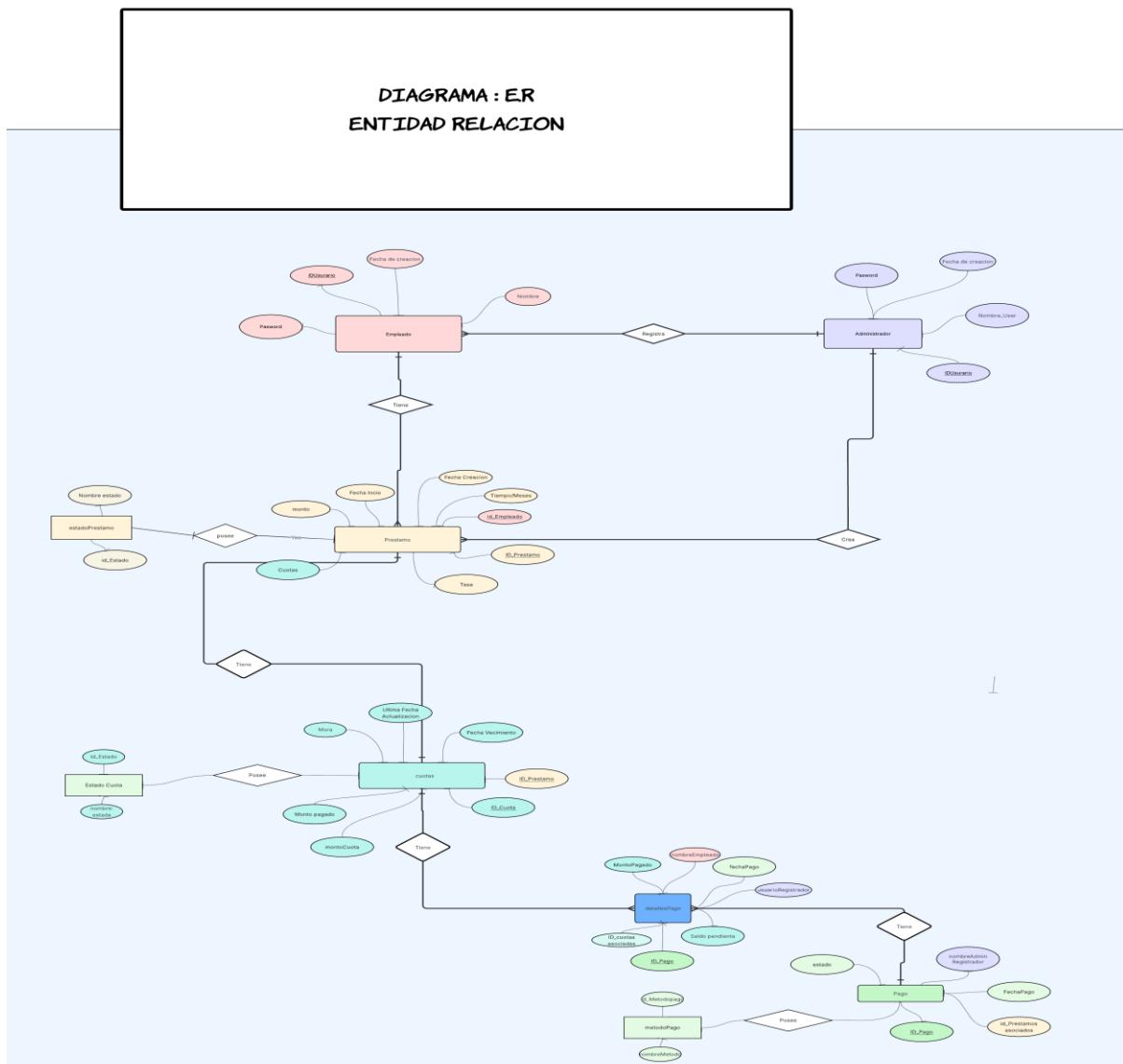


El diagrama de clases muestra las principales clases del sistema , cada una con sus respectivos datos, métodos, información y tipos de datos involucrados relacionados entre respetando la normalización con tablas intermedias

Nota: En el siguiente enlace encontrara la imagen con mayor calidad:

<https://drive.google.com/file/d/1iAfmHWJFNwxGFEfNvvc-CcDCAe4BzJ71/view?usp=sharing>

- Diagrama de Entidad Relación

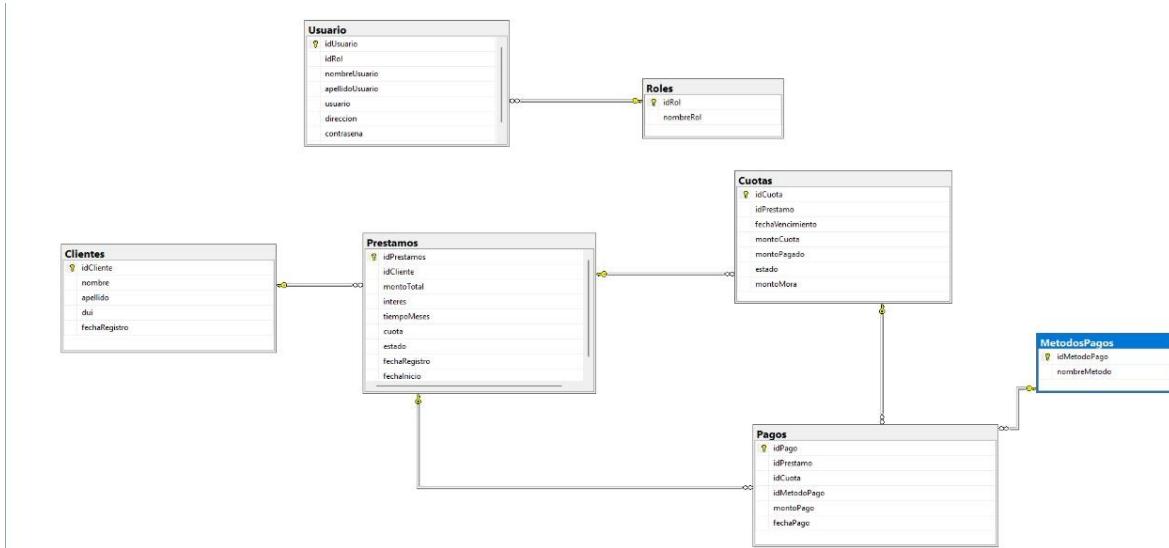


El diagrama Entidad relación con el cual se definen las relaciones sus atributos y sus relaciones entre sirve para diseñar la estructura de una base de datos. mostrar cómo los diferentes elementos de tu sistema se conectan entre sí y qué información contienen.

Nota: En el siguiente enlace encontrara la imagen con mayor calidad:

https://drive.google.com/file/d/1SuN8X6EnJh_-JZ5RRwZG4O9iyCvNXMIU/view?usp=sharing

- Diagrama de Modelo de Dominio



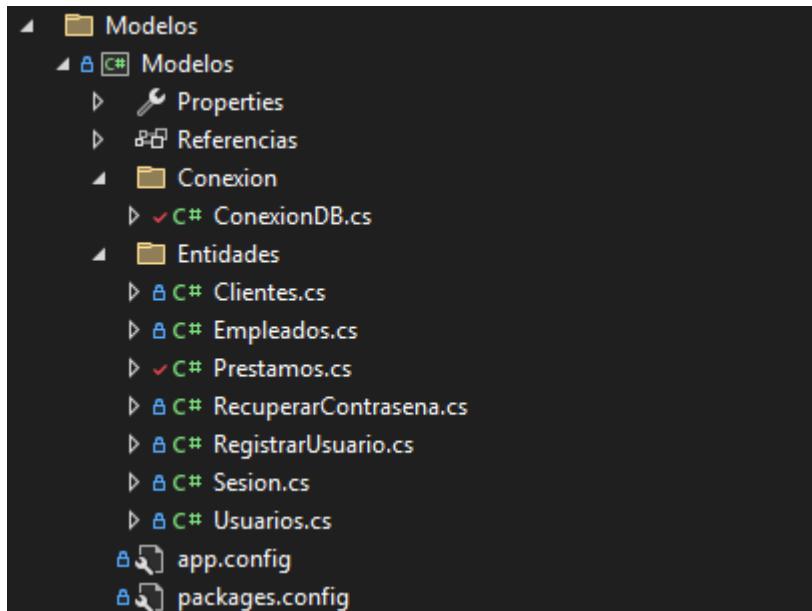
Nota: En el siguiente enlace encontrara la imagen con mayor calidad:

https://drive.google.com/file/d/1ga7jWImVNz6s4b3iGxjznT_HzNNHwzp8/view?usp=sharing

6. ESTRUCTURA DEL PROYECTO.

El proyecto Credaris está organizado en una estructura que sigue el patrón de diseño Modelo-Vista (MV), adaptado para una aplicación de escritorio con Windows Forms. El objetivo de esta organización es separar las responsabilidades de la aplicación en tres capas distintas: la lógica de negocio, la interfaz de usuario y la gestión de datos. Esta separación facilita el desarrollo, el mantenimiento y la escalabilidad del sistema.

Estructura del backend:



El backend del proyecto, ubicado en el directorio Modelos, es la capa encargada de la lógica de negocio y el acceso a los datos. Es el "motor" de la aplicación que procesa la información y se comunica con la base de datos de SQL Server.

La organización interna de esta capa es la siguiente:

- Conexión:

ConexionDB.cs: es la responsable de establecer la conexión entre la aplicación y la base de datos de SQL Server. Su única función es proporcionar un método estático (conectar()) que crea y abre una conexión, utilizando credenciales y una cadena de conexión predefinidas. Esta centralización del código de conexión permite que el resto de la aplicación obtenga una conexión de manera sencilla y consistente.

- Entidades: Contiene las clases que representan la estructura de los datos del sistema. Por ejemplo:

Cuentas.cs: Modela la información de los clientes, como su nombre y dirección.

Empleados.cs: Modela los datos de los empleados, como su salario o cargo.

Prestamos.cs: Representa la información de los préstamos, como el monto y las fechas.

Usuarios.cs: Representa los datos de los usuarios del sistema, como sus credenciales y roles.

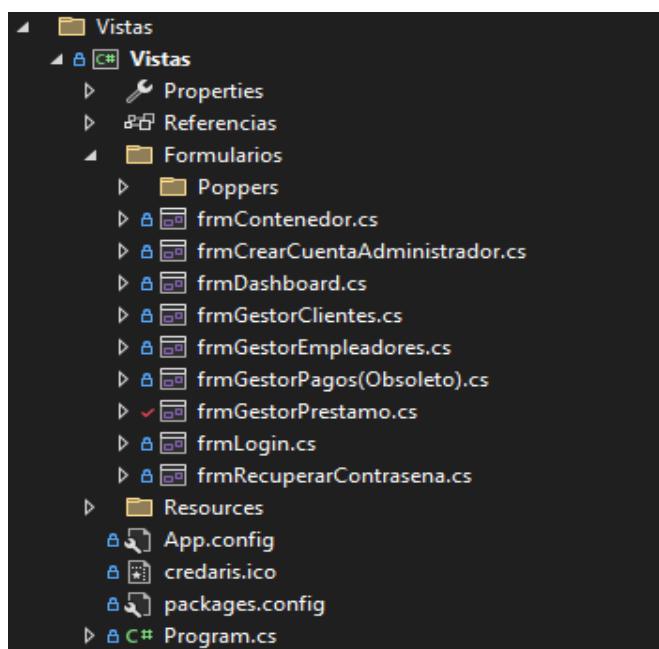
Sesion.cs: Gestiona el estado de la sesión del usuario para mantener la información de autenticación mientras el programa está en uso.

- Clases de Lógica de Negocio y Control de Flujo:

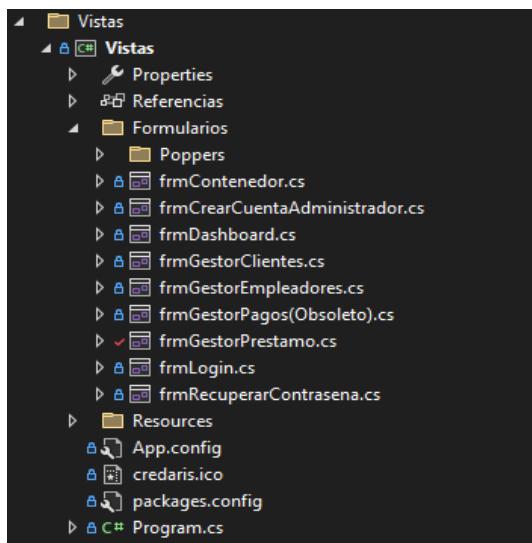
RecuperarContrasena.cs: Contiene la lógica para gestionar el proceso de recuperación de contraseñas de los usuarios.

RegistrarUsuario.cs: Maneja el proceso de registro de nuevos usuarios en la base de datos, incluyendo la validación de los datos.

Estructura del Frontend:



El frontend del proyecto, ubicado en el directorio Vistas, es la capa encargada de la interfaz de usuario y la experiencia del usuario. Es la "cara" de la aplicación que los usuarios ven e interactúan, y se comunica con el backend para solicitar y mostrar la información.



En el directorio de Vistas se encuentran 3 carpetas que son Formularios, Poppers y Resources.

Formularios:

frmContenedor.cs: Este es el formulario principal o padre de la aplicación. Actúa como un contenedor que aloja otros formularios (hijos) dentro de él.

frmCrearCuentaAdministrador.cs: Este formulario está diseñado para que un usuario con privilegios de administrador pueda crear una nueva cuenta de administrador.

frmDashboard.cs: El Dashboard es un panel de control que te da una bienvenida y te muestra la hora en tiempo real.

frmGestorClientes.cs: Este formulario permite gestionar a los clientes. Las funciones típicas de un gestor de clientes son ver, agregar, editar y eliminar información de clientes, como nombres, datos de contacto.

frmGestorEmpleados.cs: Similar al gestor de clientes, este formulario se usa para administrar la información de los empleados de la empresa. Permite a los usuarios autorizados ver y modificar los datos de los empleados.

frmGestorPago.cs: Este formulario es para la gestión de los pagos, para poder agregar los pagos realizados por los clientes.

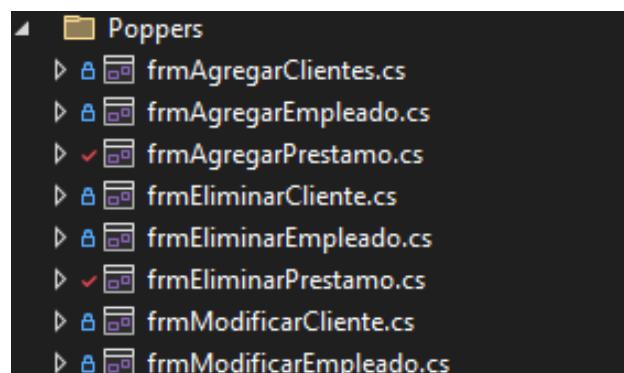
frmGestorPrestamo.cs: Este formulario es para la gestión de préstamos. permite a los usuarios registrar nuevos préstamos, consultar el estado de los préstamos existentes, ver el historial de pagos y realizar otras operaciones relacionadas con los créditos.

frmLogin.cs: Este es el formulario de inicio de sesión. Es la primera pantalla que se muestra al usuario al abrir la aplicación. Su función es autenticar al usuario pidiendo su nombre de usuario y contraseña para verificar su identidad antes de darle acceso a las demás funcionalidades de la aplicación.

frmRecuperarContrasena.cs: Este formulario proporciona la funcionalidad para que un usuario pueda recuperar su contraseña en caso de que la haya olvidado. Suele requerir información de verificación, como una dirección de correo electrónico o una pregunta de seguridad, para permitir al usuario restablecer su contraseña de forma segura.

Poppers:

los poppers son formularios o ventanas secundarias que se abren de manera temporal y sobre la ventana principal. Su propósito principal es realizar una tarea específica de forma rápida y concisa, sin ocupar toda la pantalla.



frmAgregarClientes.cs: Este formulario se utiliza para agregar un nuevo cliente al sistema. Probablemente se abre desde los formularios padres y contiene campos para introducir la información del nuevo cliente, como nombre, dirección y datos de contacto.

frmAgregarEmpleado.cs: Similar al anterior, este formulario es para añadir un nuevo empleado a la base de datos. Se abriría desde el frmGestorEmpleados y contendría los campos necesarios para capturar la información del empleado.

frmAgregarPrestamo.cs: Este formulario permite registrar un nuevo préstamo. Se abriría desde el frmGestorPrestamo y tendría campos para capturar detalles como el monto del préstamo, la fecha y el cliente o empleado asociado.

frmEliminarCliente.cs: Su función es eliminar un cliente existente. se abre como una ventana de confirmación en frmGestorEmpleados que solicita al usuario verificar que realmente desea eliminar el registro del cliente seleccionado.

frmEliminarEmpleado.cs: Al igual que el de clientes, este formulario se encarga de eliminar un registro de empleado. También funciona como una ventana de confirmación para evitar eliminaciones accidentales.

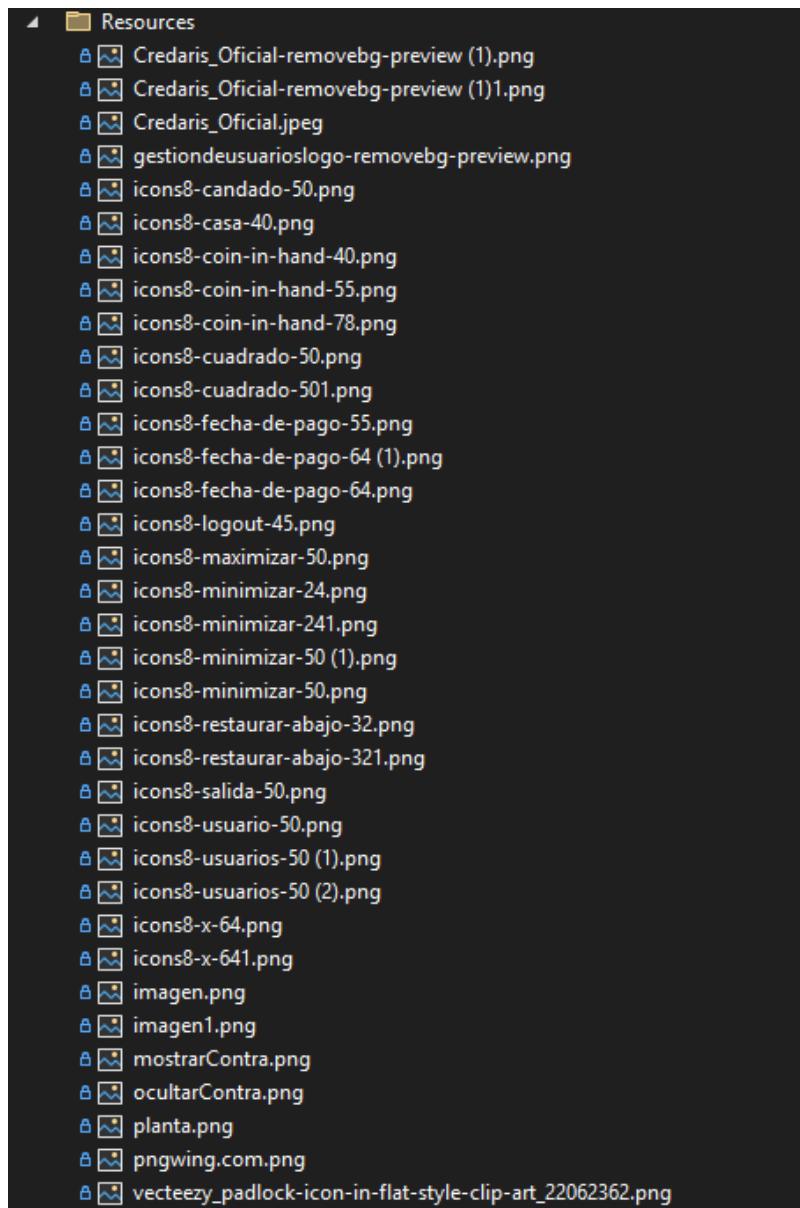
frmEliminarPrestamo.cs: Este formulario se usa para eliminar un registro de préstamo. Se abre para confirmar la eliminación de un préstamo del sistema.

frmModificarCliente.cs: Este formulario se utiliza para modificar la información de un cliente que ya existe en el sistema. Al abrirse, probablemente precargará los datos actuales del cliente para que el usuario pueda editarlos y luego guardarlos.

frmModificarEmpleado.cs: Similar al formulario de clientes, este se encarga de editar los datos de un empleado. Abre una ventana con los campos de información del empleado para que puedan ser actualizados.

RESOURCES:

El directorio Resources es donde se guardan los recursos del proyecto, como imágenes, íconos y otros archivos que no son código.



7. DISEÑO DE LA APLICACION.

Paleta de colores:

En el proyecto Credaris se utilizó esta paleta de colores ya que transmite confianza y claridad, esta combinación de colores busca un equilibrio entre profesional y algo amigable a la vista, para facilitar la comprensión de las interacciones del usuario con la aplicación:

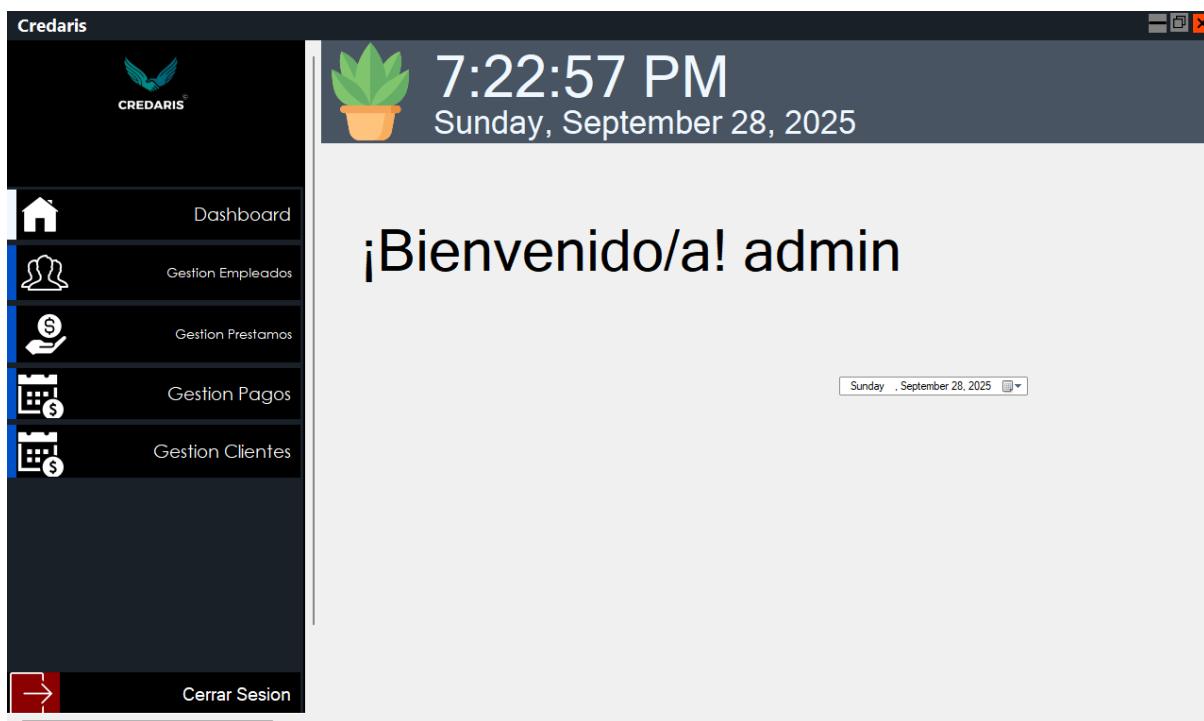
Paleta:

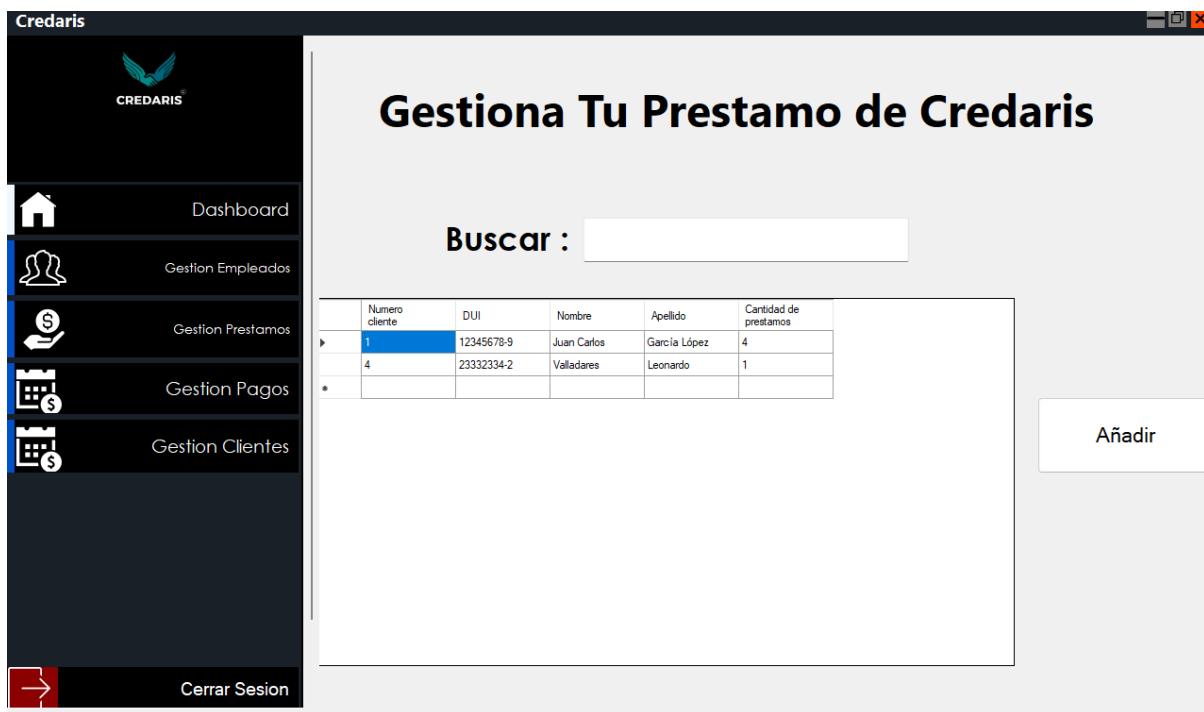


Aplicación en diferentes elementos:

En la aplicación Credaris, la paleta de colores está basada en tonos azules y tonos morados los cuales se aplican de manera bien distribuida y equilibrada con las interfaces, para garantizar un diseño agradable y moderno a la vista. A continuación, se muestran algunas imágenes para demostrar cómo se emplean estas paletas de colores en los diferentes componentes:







Tipos y tamaños de fuentes

El tipo de fuente empleado en el proyecto es:

Microsoft Yahei

Microsoft Yahei

A b c d e f g h i j k l m n ñ o p q r s t u w x y z

0 1 2 3 4 5 6 7 8 9

El cual se emplea con un tamaño estandar de 9.25

Tipos y dimensiones de imágenes utilizadas

Todas las imágenes utilizadas en el sistema son PNG y estas imágenes varían de tamaños.

En esta versión de la aplicación no contamos con muchas imágenes, por los momentos contamos con estas que van desde 21px a 360px



Tamaños de pantalla soportados

La interfaz de Credaris está optimizada para dos estados de visualización principales, asegurando una experiencia de usuario clara y funcional en ambos escenarios:

- Pantalla de Inicio de Sesión (**1280 x 760**): Esta es la resolución de la ventana de login al iniciar la aplicación. Se ajusta para ser funcional y clara, enfocándose únicamente en el proceso de autenticación.
- Tamaño Normal de Ventana (**1005 x 624**): Después de iniciar sesión, la ventana principal se redimensiona a este tamaño. Es la vista predeterminada para el trabajo diario, equilibrando la visibilidad y la capacidad de realizar otras tareas en el escritorio.
- Modo de Pantalla Completa (**1920 x 1080**): Al maximizar la ventana, la interfaz se expande a esta resolución (o a la resolución máxima del monitor). Este modo se utiliza para una visualización inmersiva, ideal para analizar datos en detalle sin distracciones.

9. REQUERIMIENTOS DE HARDWARE Y SOFTWARE

Localmente

Cuando se habla de requerimientos de hardware y software en entorno local se refiere a los requerimientos optimos para una ejecución satisfactoria del sistema.

Hardware mínimo:

Sistema operativo

- Windows 10 o superior.
- Linux Ubuntu 20.04 o superior.

Memoria RAM:

- 4 GB de RAM

Procesador:

- Intel Core I3 (de 7^a Generation o superior) o equivalente.
- AMD: AMD Ryzen 3 (de lá serie 1000 o superior) o equivalente.

Espacio HDD:

- 200 MB libres.

Hardware recomendado:

Sistema operativo

- Windows 11
- Linux Ubuntu 24.04

Memoria RAM:

- 8 GB de RAM

Procesador:

- Intel Core I3 (de 7^a Generation o superior) o equivalente.
- AMD: AMD Ryzen 3 (de lá serie 1000 o superior) o equivalente.

Espacio SSD:

- 500 MB libres.

Servidor

El backend de la aplicación Credaris requiere de un servidor con hardware específico para asegurar un rendimiento óptimo, alta disponibilidad y un manejo eficiente de la lógica de negocio y el acceso a la base de datos de SQL Server.

- CPU (Procesador) Se recomienda un procesador con 4 núcleos a 2.5 GHz o superior, preferentemente de la familia Intel Xeon o AMD EPYC para servidores. Se requiere una arquitectura de 64 bits para el manejo eficiente de la memoria y los procesos.
- Memoria RAM Se necesita un mínimo de 8 GB de RAM, aunque se recomienda 16 GB para garantizar un mejor rendimiento bajo una carga de trabajo moderada a alta, especialmente durante el procesamiento de consultas complejas.
- Almacenamiento (Disco Duro) Es crucial contar con una unidad SSD de al menos 100 GB para el sistema operativo y la base de datos, ya que mejora significativamente la velocidad de lectura y escritura. Adicionalmente, se recomienda un HDD de 500 GB para almacenamiento secundario de copias de seguridad, registros de la aplicación y archivos de usuario. También es vital el soporte para RAID 1 o RAID 5 para la redundancia y seguridad de los datos.
- Ancho de Banda de Red Se requiere una conexión mínima de 1 Gbps para la red interna del servidor. Para el tráfico de internet, un ancho de banda mínimo de 100 Mbps es suficiente, aunque para un uso más intensivo se recomienda una conexión de 500 Mbps.

10. INSTALACION Y CONFIGURACION

Proceso de Instalación

Para instalar y ejecutar la aplicación, debes obtener el código y configurar la conexión a la base de datos de tu computadora.

1. Obtención del Código

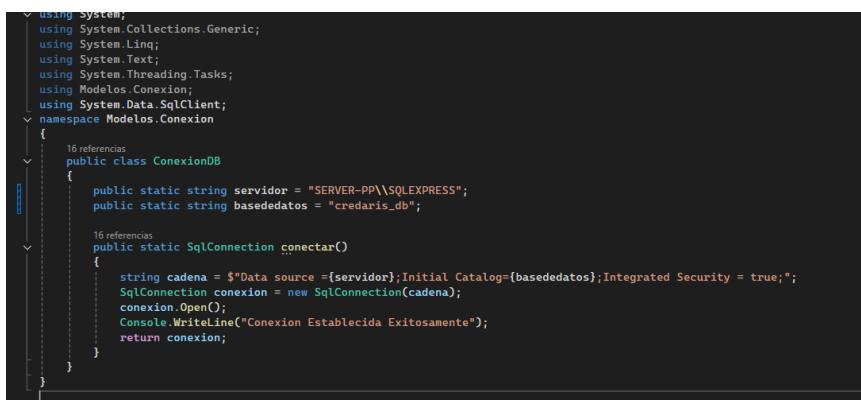
Puedes obtener el código de la aplicación de dos maneras:

- Clonar desde GitHub: La opción más recomendada es clonar el repositorio de GitHub. Para ello, necesitas tener Git instalado en tu computadora.
- Descargar el archivo .RAR: Si eliges este método, es posible que Windows bloquee algunos archivos por motivos de seguridad. Para evitar errores, asegúrate de desbloquear el archivo .rar antes de descomprimirlo. Solo tienes que hacer clic derecho sobre el archivo, ir a Propiedades y marcar la casilla Desbloquear en la pestaña General.

2. Conexión a la Base de Datos

Después de obtener el código, el siguiente paso es conectar la base de datos localmente.

1. Ejecutar la base de datos: Restaura el archivo de base de datos proporcionado SQL Server para que la base de datos esté disponible.
2. Abrir el proyecto en Visual Studio: Abre el archivo de la solución (.sln) en Visual Studio.
3. Cambiar el nombre del servidor: En el archivo de configuración del proyecto (conexionDB.cs), busca la cadena de conexión. Deberás cambiar el nombre del servidor por el de tu computadora para establecer la conexión correcta.



```
using system;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Modelos.Conexion;
using System.Data.SqlClient;
namespace Modelos.Conexion
{
    16 referencias
    public class ConexionDB
    {
        public static string servidor = "SERVER-PP\\SQLEXPRESS";
        public static string basededatos = "creدارis_db";

        16 referencias
        public static SqlConnection conectar()
        {
            string cadena = $"Data source ={servidor};Initial Catalog={basededatos};Integrated Security = true;";
            SqlConnection conexion = new SqlConnection(cadena);
            conexion.Open();
            Console.WriteLine("Conexion Establecida Exitosamente");
            return conexion;
        }
    }
}
```

4. Código a modificar: public static string servidor = "SERVER-PP\\SQLEXPRESS";

Ejemplo: Si el nombre de tu servidor es "PC-USUARIO\SQLEXPRESS", la línea quedaría así: public static string servidor = "PC-USUARIO\\SQLEXPRESS"; Al hacer este cambio y compilar la aplicación, esta se conectará a la base de datos en su computadora.

