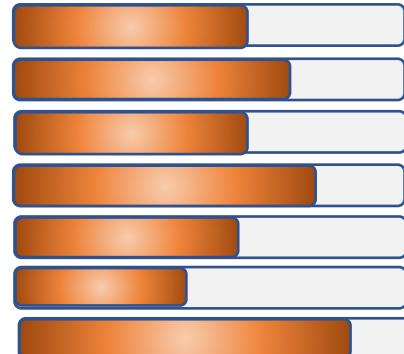




Me apasiona la tecnología y me encanta sumergirme en el análisis de datos utilizando herramientas como Excel, SQL, R, Python y RapidMiner. Además, disfruto creando visualizaciones impactantes utilizando Power BI. Mi pasión por la programación se extiende tanto al desarrollo web front-end como al backend. Me encanta enseñar y compartir mis conocimientos, y me emociona la idea de transmitir lo que aprendo cada día.



- ✓ C#, SQL, Crystal Report.
- ✓ HTML, CSS, Javascript.
- ✓ PHP, Python, SQL.
- ✓ Power BI, Excel.
- ✓ RapidMiner, R, Python.
- ✓ XAMARIN, C#, XAML
- ✓ Macola, Sap., MSeller



Santo Domingo, R.D.



Juancito.pena@gmail.com



809-767-9290



Ingeniero en Sistemas y Computación
Post-Grado en Ingeniería de Software
Maestría en Sistemas Mención Gerencial

Universidad Dominicana O&M



**Juancito
Peña V.**



Instructor de Tecnología (2019-Hoy)

Universidad Dominicana O&M



Enc. Soporte Tecnológico (2011-Hoy)

Cerveceria Vegana S.R.L.



Consultor de Tecnologias (2019-Hoy)



Independiente Freelancer.

Bloguer Tecnológico (2021)

<https://advisertecnology.com>



<https://www.youtube.com/@JuancitoPenaV>

Introducción:

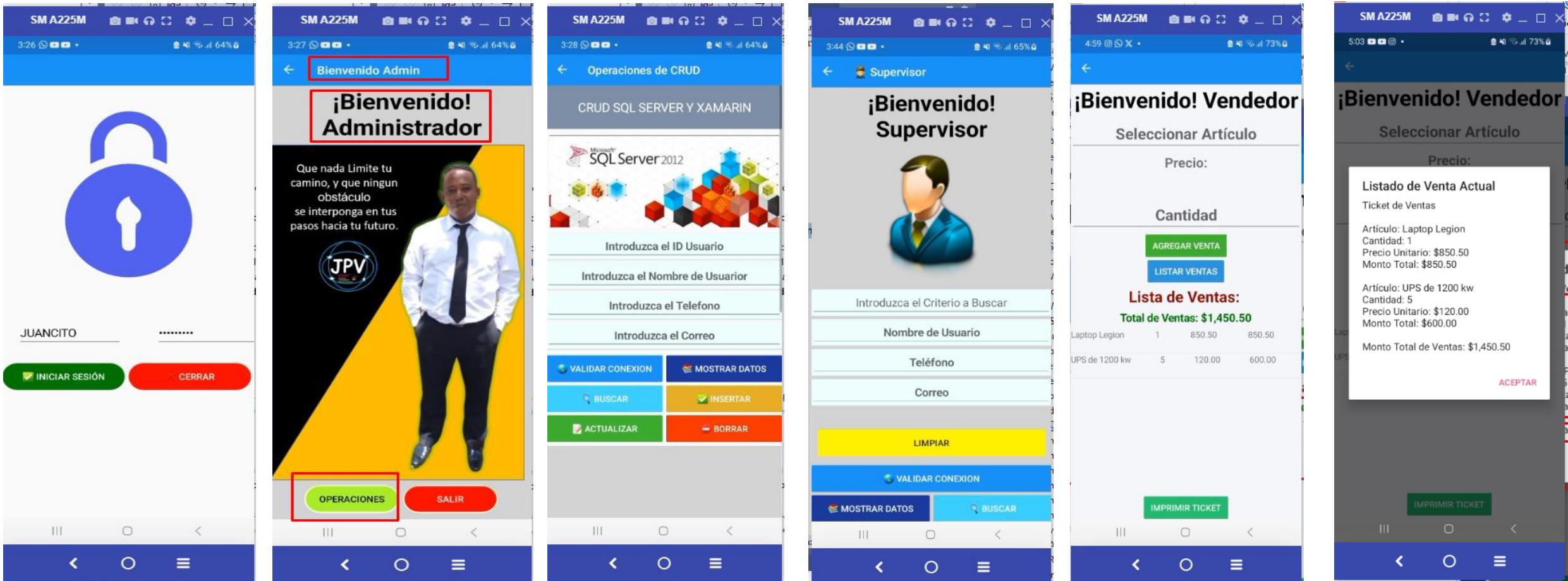
Introducción al desarrollado con Xamarin, C# y SQL Server. En este proyecto, exploraremos cómo construir una aplicación móvil funcional que permitirá a los usuarios registrar usuarios, roles de usuarios, artículos, y ventas, generar tickets, todo ello a través de una interfaz de usuario intuitiva y amigable.

Objetivos Clave:

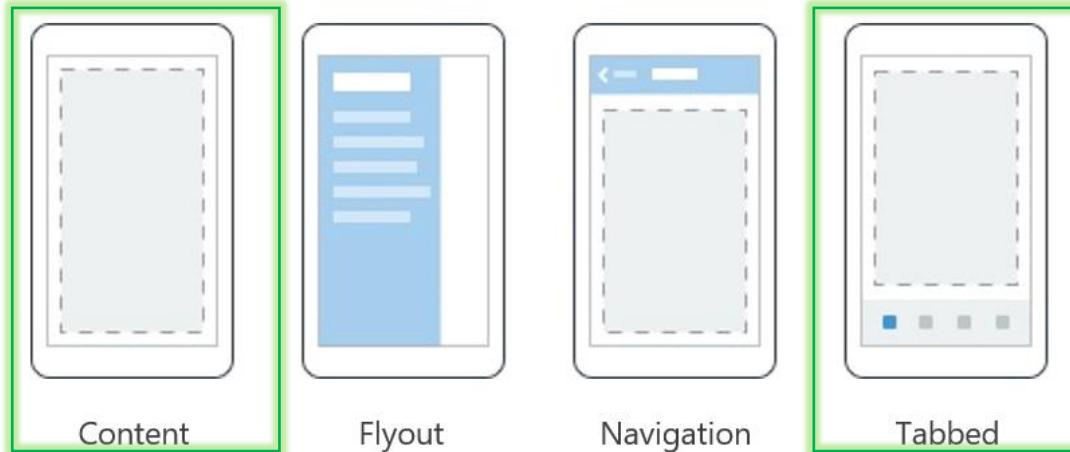
- a) Crear una interfaz de usuario atractiva y fácil de usar que permita a los usuarios navegar y realizar acciones de manera intuitiva.
- b) Implementar la lógica de negocio para registrar ventas y calcular montos totales automáticamente.
- c) Conectar la aplicación a una base de datos SQL Server para almacenar información sobre artículos y ventas.
- d) Desarrollar la funcionalidad de listar ventas y mostrar detalles relevantes.
- e) Generar tickets de venta con información detallada y presentarla al usuario.
- f) Proporcionar una experiencia de aprendizaje práctica en programación orientada a objetos, desarrollo móvil y manejo de bases de datos.

A medida que avancemos en este emocionante proyecto, exploraremos cómo los conceptos de programación orientada a objetos y el uso de herramientas como Xamarin y SQL Server se combinan para crear una aplicación móvil efectiva y funcional. Prepárate para sumergirte en la programación móvil y desarrollar tus habilidades en un entorno práctico y divertido. ¡Vamos a comenzar a construir nuestra aplicación de ventas y a aprender juntos en este emocionante viaje de desarrollo!

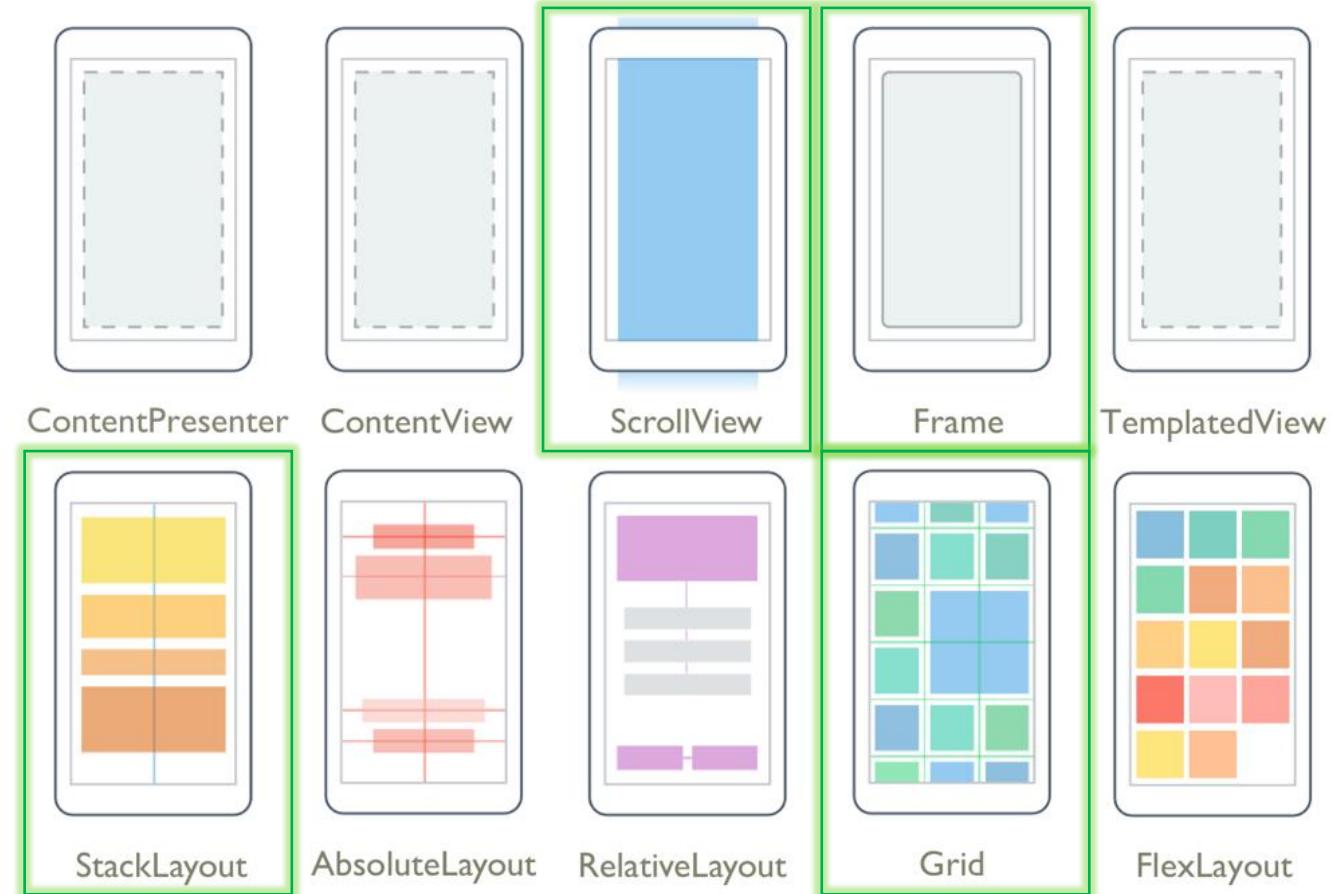
LA APLICACIÓN CONSISTIRÁ EN ESTO.



Páginas



Layouts



Controles

CarouselView

CollectionView

ListView

Picker

TableView

IndicatorView

ProgressBar

ActivityIndicator

Editor

Entry

CheckBox

DatePicker

Slider

Stepper

Switch

TimePicker

Button

ImageButton

RadioButton

RefreshView

SearchBar

SwipeView

BlazorWebview

Border

BoxView

Frame

GraphicsView

Image

Label

ScrollView

Lo primero que necesitas es descargar e instalar el SQL Server Express, en cualquiera de estas versiones, te recomiendo que lo hagas con la versión developer, que si más adelante tienes otros proyectos o decides trabajar con esto, pues aquí lo tendrás instalado.

<https://www.microsoft.com/en-us/sql-server/sql-server-downloads>



Developer

SQL Server 2022 Developer is a full-featured free edition, licensed for use as a development and test database in a non-production environment.

[Download now](#)



Express

SQL Server 2022 Express is a free edition of SQL Server, ideal for development and production for desktop, web, and small server applications.

[Download now](#)

Para poder administrar la base de datos de forma grafica tendrás que instalar el administrador de base de datos de SQL Server, el cual es el SQL Management Studio.

<https://learn.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver16>

Deploy, monitor, and upgrade the data-tier components used by your applications and build queries and scripts.

Use SSMS to query, design, and manage your databases and data warehouses, wherever they are - on your local computer or in the cloud.

Download SSMS

[Free Download for SQL Server Management Studio \(SSMS\) 19.0.2](#)

SSMS 19.0.2 is the latest general availability (GA) version. If you have a *preview* version of SSMS 19 installed, you should uninstall it before installing SSMS 19.0.2. If you have SSMS 19.x installed, installing SSMS 19.0.2 upgrades it to 19.0.2.

- Release number: 19.0.2
- Build number: 19.0.20209.0
- Release date: March 13, 2023

By using SQL Server Management Studio, you agree to its [license terms](#) and [privacy statement](#). If you have comments or suggestions or want to report issues, the best way to contact the SSMS team is at [SQL Server user feedback](#).

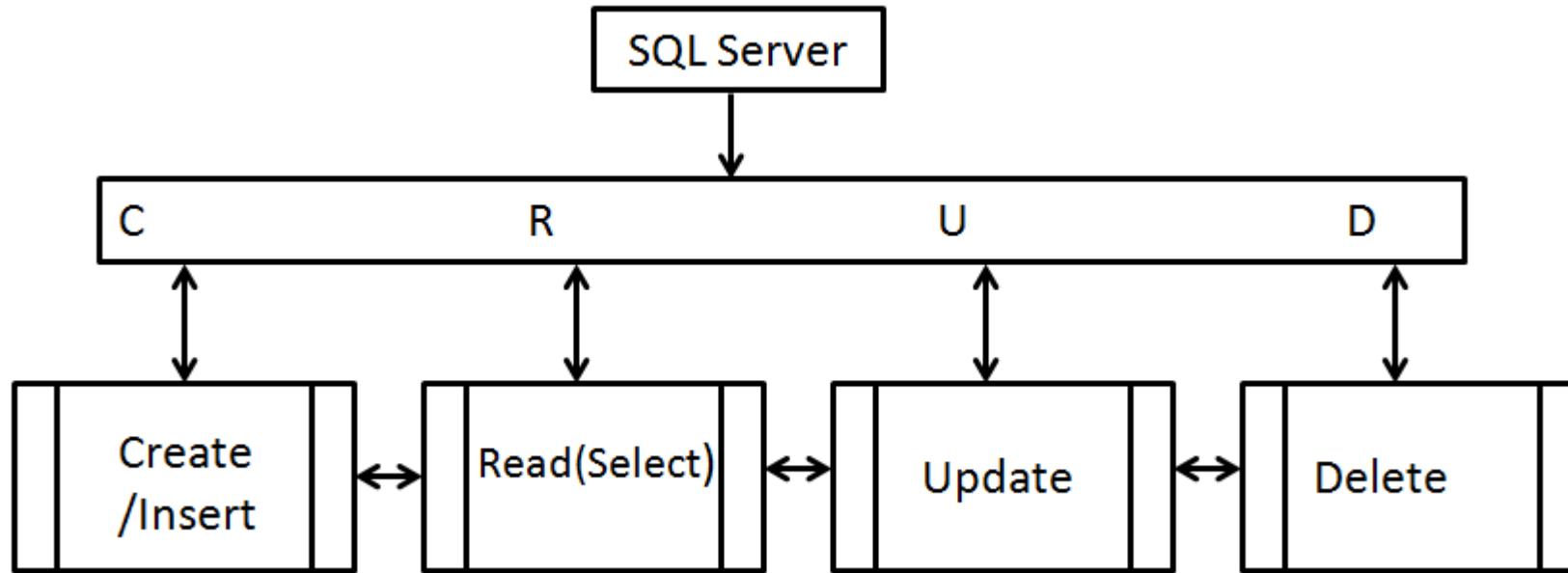
The SSMS 19.x installation doesn't upgrade or replace SSMS versions 18.x or earlier. SSMS 19.x installs alongside previous versions, so both versions are available for use. However,

Aquí en este video puedes ver Cómo se instala y configuran ambas version de SQL SERVER paso a paso

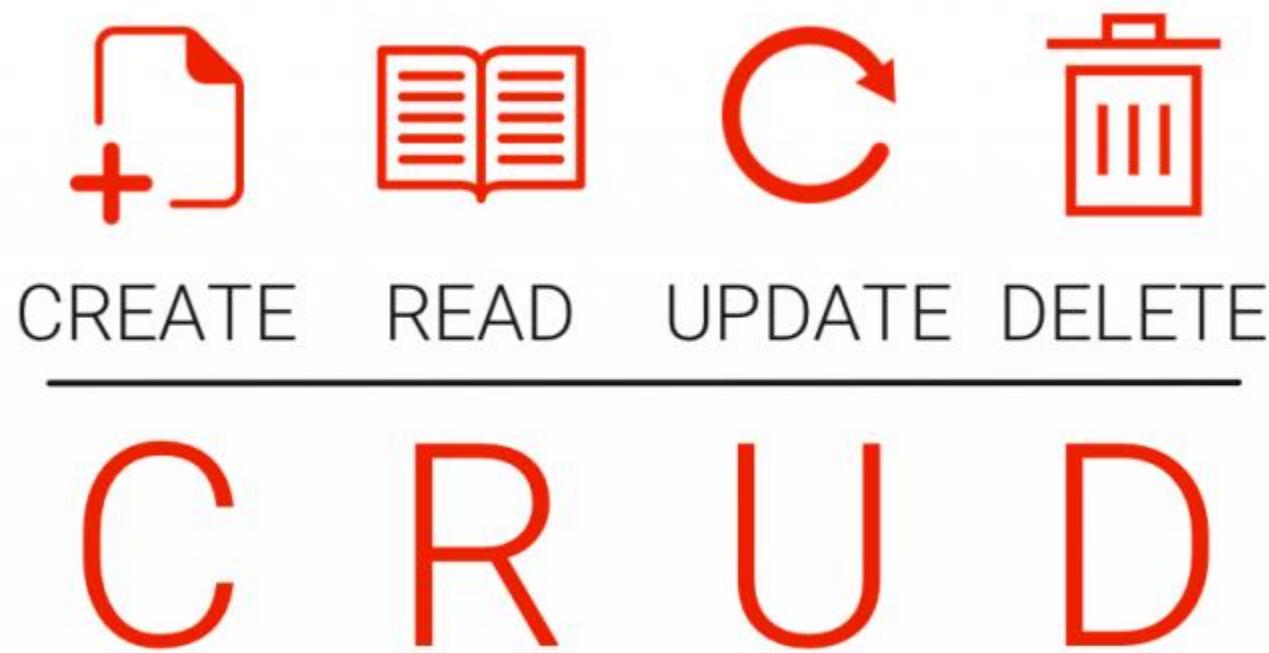


https://www.youtube.com/watch?v=mA1qoWdNCOE&ab_channel=SergioAlejandroCampos-EXCELeINFO

Xamarin.Forms - Operaciones CRUD de base de datos SQL Server



En programación de computadoras, crear, leer, actualizar y eliminar se le llama (CRUD), estas son las cuatro funciones básicas del almacenamiento persistente. A veces se utilizan palabras alternativas al definir las cuatro funciones básicas de CRUD, como recuperar en lugar de leer, modificar en lugar de actualizar o destruir en lugar de eliminar. CRUD también se usa a veces para describir las convenciones de la interfaz de usuario que facilitan la visualización, la búsqueda y el cambio de información; a menudo utilizando formularios e informes basados en computadora.



El término probablemente fue popularizado por primera vez por James Martin en su libro de 1983 sobre la gestión del entorno de base de datos.

El acrónimo puede extenderse a CRUDL para cubrir la lista de grandes conjuntos de datos que aportan complejidad adicional, como la paginación, cuando los conjuntos de datos son demasiado grandes para guardarlos fácilmente en la memoria".

En nuestro caso vamos
a crear un CRUD
utilizando el gestor de
Base de Datos SQL
Server, Xamarin, C# y
XAML.



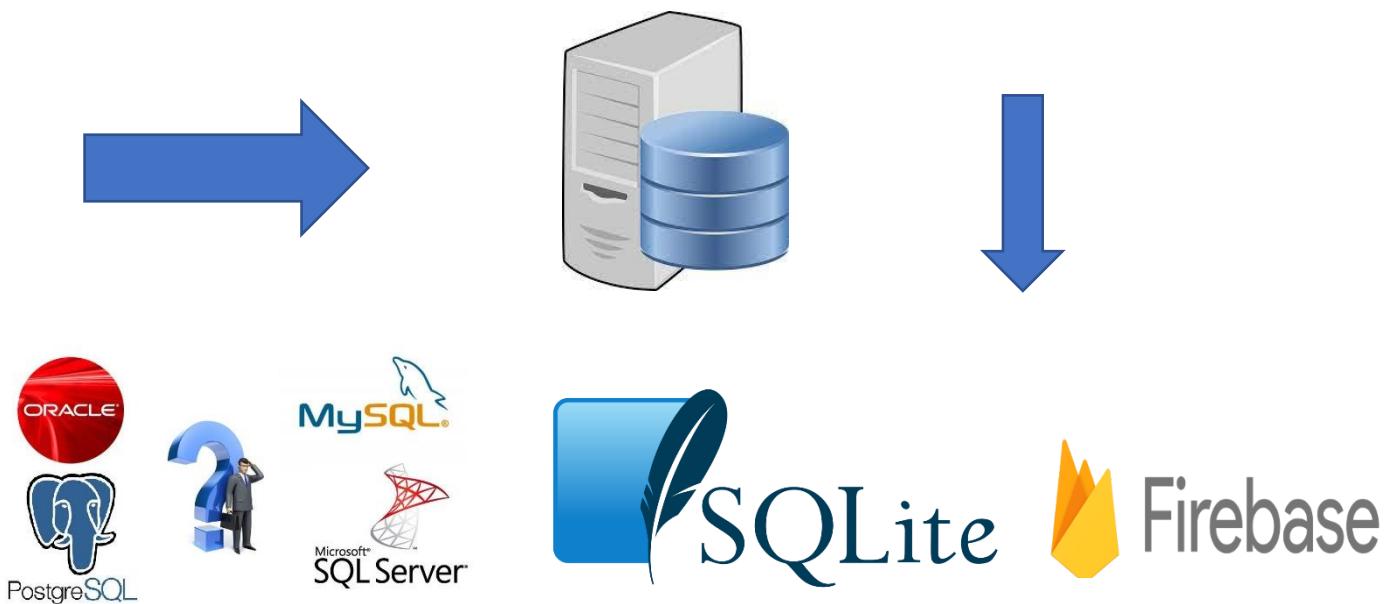
CRUD

C= CREATE- > CREAR- GUARDAR-AGREGAR

R- READ- > LEER- CONSULTAR- EXTRAER

U= UPDATE- > ACTUALIZAR - MODIFICAR

D= DELETE- > ELIMINAR - BORRAR



CRUD CON XAMARIN

Microsoft SQL Server

Formulario de Estudiantes

Codigo

Nombre

Apellido

Telefono

Direccion

Crear **Leer**

Actualizar **Eliminar**

CRUD



Base de Datos = Estudiantes



Tabla= Estudiantes

id	Nombre	Apellido	Telefono	Direccion
1	Arlete	De Lima	809-xxx-XXXX	VSVSVSVSVSS
2	Wilmoris	De Lima	809-xxx-XXXX	VSVSVSVSVSS
3	Arlete	De Lima	809-xxx-XXXX	VSVSVSVSVSS
4	Arlete	De Lima	809-xxx-XXXX	VSVSVSVSVSS

Crear un proyecto

Plantillas de proyecto recientes

Aplicación móvil (Xamarin.Forms)

C#

Aplicación de Windows Forms (.NET Framework)

Visual Basic

Aplicación de Windows Forms (.NET Framework)

C#

Aplicación de Windows Forms

C#

Integration Services Project

Aplicación de Crystal Reports

Visual Basic

Buscar plantillas (Alt+S)  Borrar todo

C#

Todas las plataformas

Móvil



Aplicación móvil (Xamarin.Forms)

Plantilla de varios proyectos para compilar aplicaciones para iOS y Android con Xamarin y Xamarin.Forms.

C#

Android

iOS

Windows

Móvil



Aplicación de Android (Xamarin)

Plantillas de proyecto para crear aplicaciones para teléfono y tableta de Android con Xamarin.

C#

Android

Móvil



Aplicación de iOS (Xamarin)

Plantillas de proyecto para crear aplicaciones de iOS para iPhone y iPad con Xamarin.

C#

iOS

Móvil



Aplicación Android Wear (Xamarin)

Proyecto para crear una aplicación Android Wear con Xamarin.

C#

Android

Móvil



Aplicación watchOS (Xamarin)

Proyecto para crear una aplicación de watchOS con Xamarin.

C#

iOS

Móvil



Biblioteca de clases de Android (Xamarin)

Proyecto de biblioteca de clases de Xamarin.Android.

C#

Android

Móvil

Siguiente

Configure su nuevo proyecto

Aplicación móvil (Xamarin.Forms)

C#

Android

iOS

Windows

Móvil

Nombre del proyecto

CRUD_SQL

[Sin título]

Ubicación

C:\Users\juanc\source\repos

Solución

Crear nueva solución

Nombre de la solución 

CRUD_SQL

Colocar la solución y el proyecto en el mismo directorio

Atrás

Crear

Nueva aplicación móvil

Seleccionar una plantilla para la aplicación

Control flotante

Una aplicación con un menú lateral que se puede contraer en pequeñas pantallas.

Con pestañas

Una aplicación que usa pestañas para navegar entre las secciones.

En blanco

Una aplicación vacía con una sola pantalla inicial.

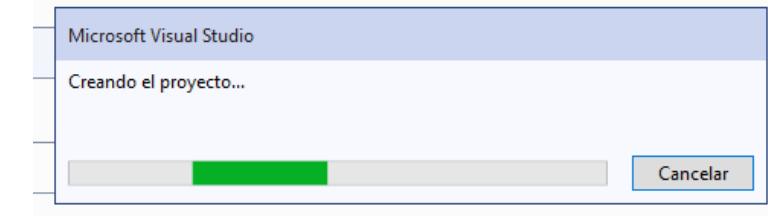


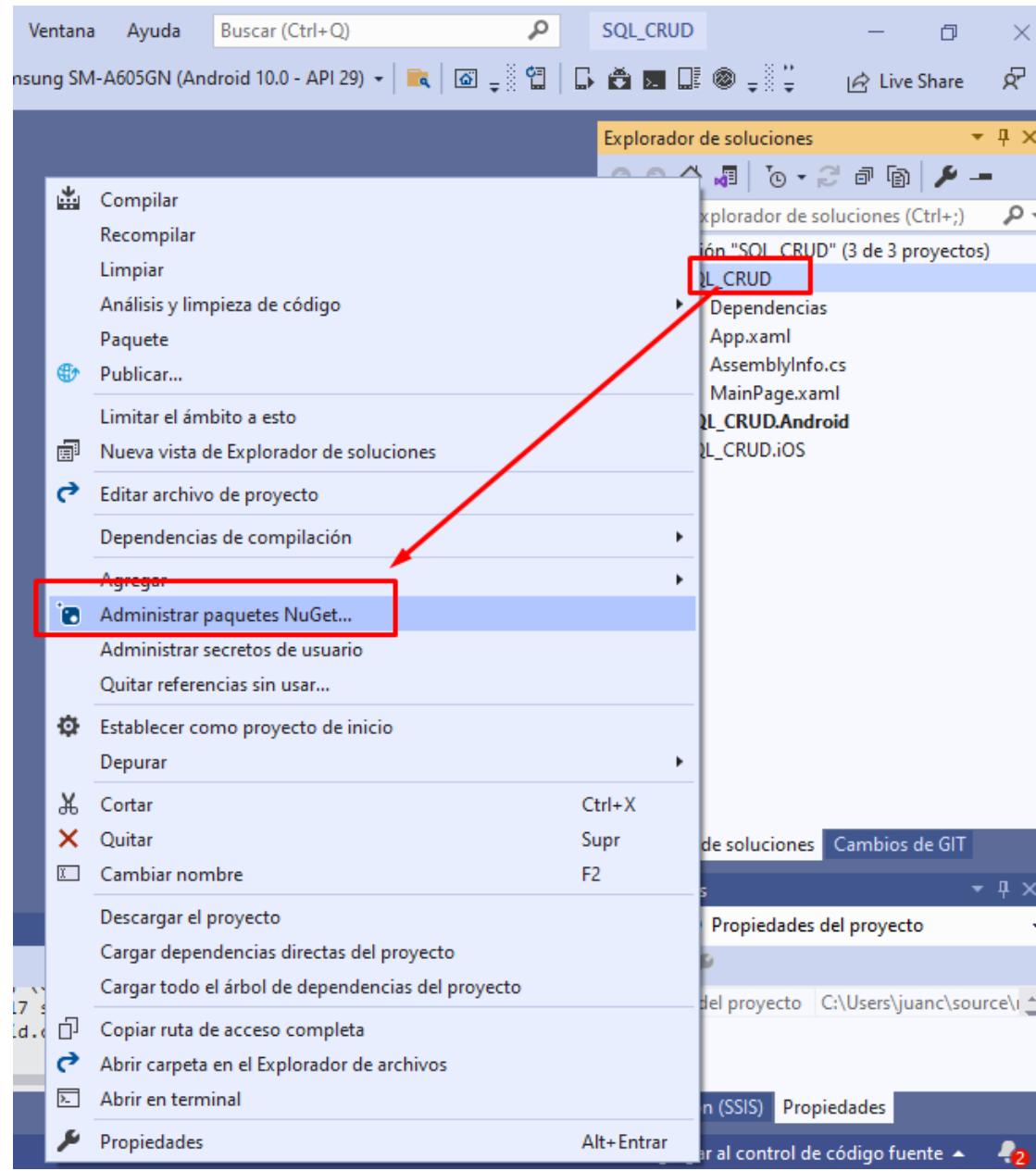
Tengo previsto desarrollar:

- Android
- iOS
- Windows (UWP) i

Atrás

Crear





Administrador de paquetes NuGet: SQL_CRUD

Origen del paquete: nuget.org

Examinar Instalado Actualizaciones 2

system.data

Incluir versión preliminar

System.Data.SqlClient por Microsoft, 251M descargas 4.8.3

Provides the data provider for SQL Server. These classes provide access to versions of SQL Server and encapsulate database-specific protocols, including tabular data stream (TDS)

System.Data.Common por Microsoft, 116M descargas 4.3.0

Provides the base abstract classes, including System.Data.DbConnection and System.Data.DbCommand, for all data providers.

System.Runtime por Microsoft, 509M descargas 4.3.1

Provides the fundamental primitives, classes and base classes that define commonly-used value and reference data types, events and event handlers, interfaces, attributes, and exceptions. This...

System.IO por Microsoft, 468M descargas 4.3.0

Provides base input and output (I/O) types, including System.IO.Stream, System.IO.StreamReader

La licencia de cada paquete la concede su propietario. NuGet no se hace responsable de los paquetes de terceros ni concede ninguna licencia para ellos.

No volver a mostrar

Ver más

System.Data.SqlClient

Versión: Versión estable más reciente

Instalar

Opciones

Descripción

Provides the data provider for SQL Server. These classes provide access to versions of SQL Server and encapsulate database-specific protocols, including tabular data stream (TDS)

Commonly Used Types:

- System.Data.SqlClient.SqlConnection
- System.Data.SqlClient.SqlException
- System.Data.SqlClient.SqlParameter
- System.Data.SqlClient.SqlDbType
- System.Data.SqlClient.SqlDataReader
- System.Data.SqlClient.SqlCommand
- System.Data.SqlClient.SqlTransaction

Explorador de soluciones

Solución "SQL_CRUD" (3 de 3 proyectos)

- SQL_CRUD
 - Dependencias
 - App.xaml
 - AssemblyInfo.cs
 - MainPage.xaml
- SQL_CRUD.Android
- SQL_CRUD.iOS

Explorador de soluciones Cambios de GIT

Propiedades

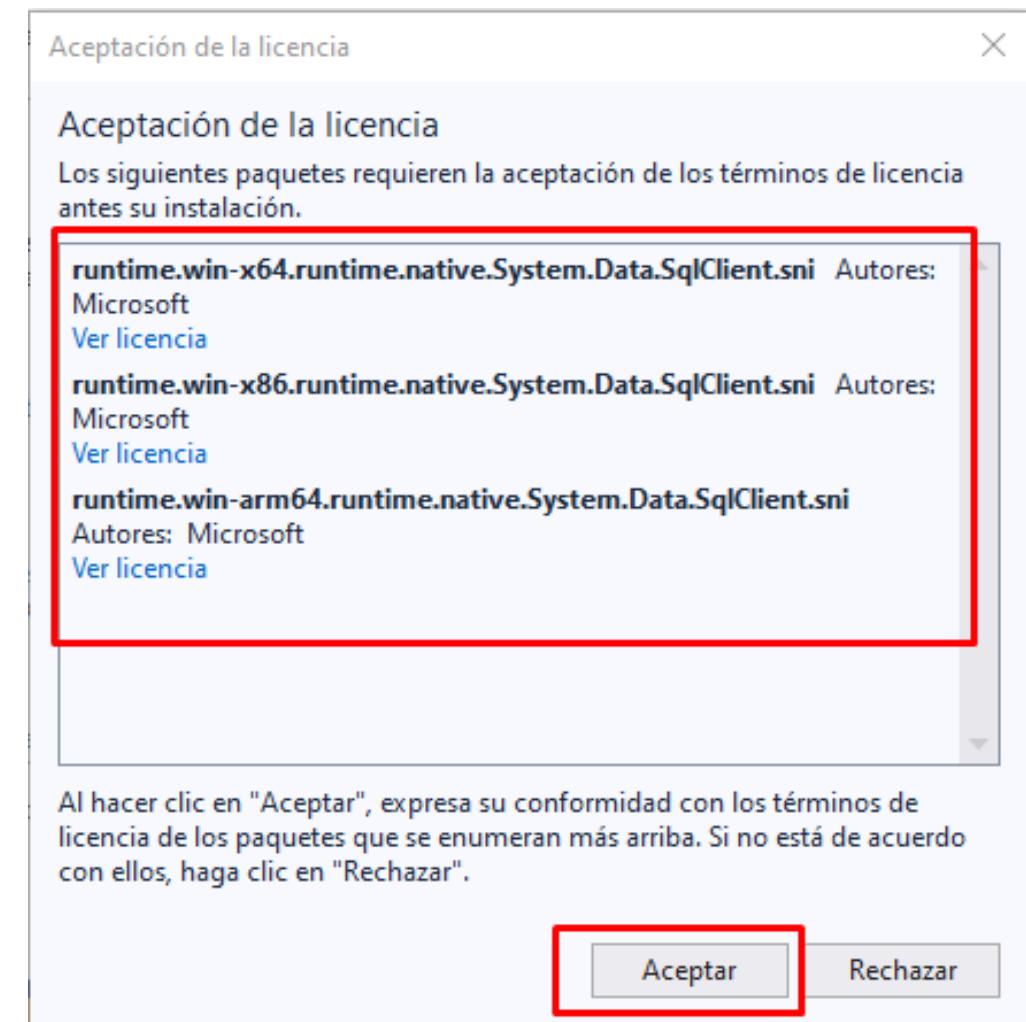
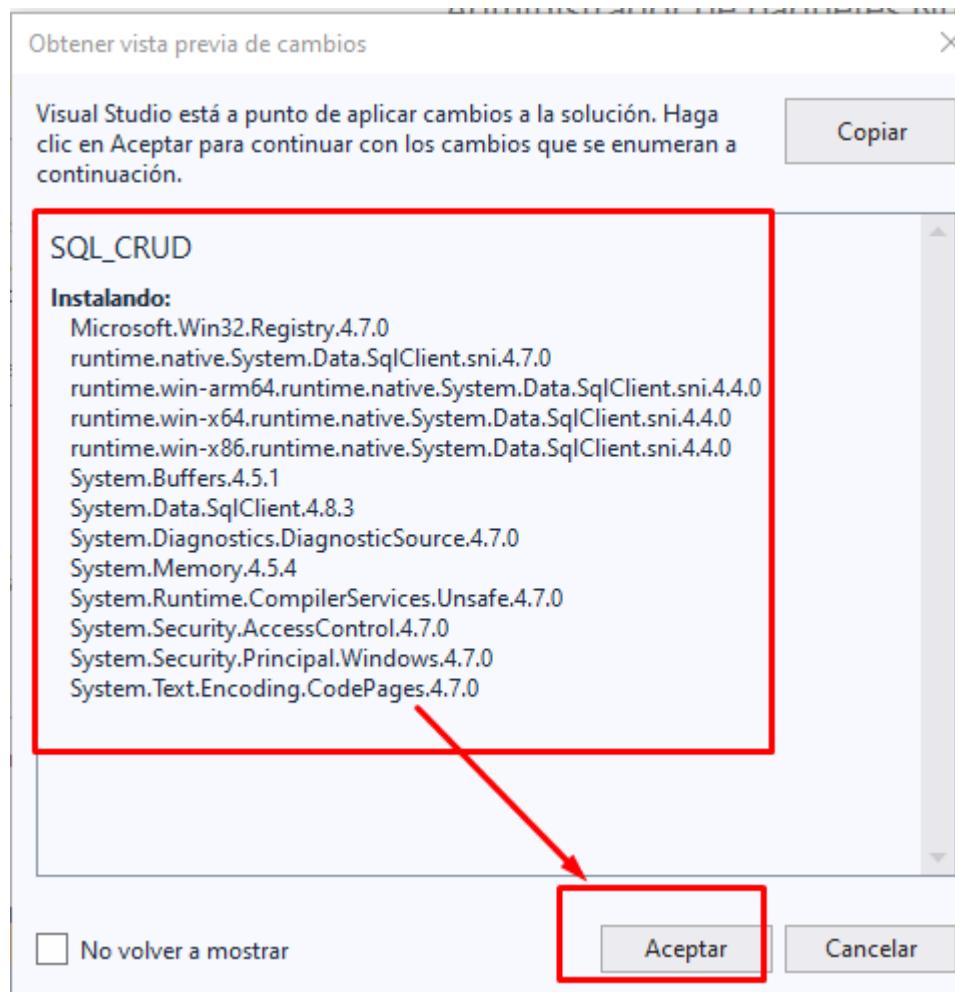
Introducción (SSIS) Propiedades

Lista Agregar al control de código fuente

23:19 28/09/2021

27°C Parc. nub... ENG

Live Share



Administrador de paquetes NuGet: SQL_CRUD

Origen del paquete: nuget.org

Examinar Instalado Actualizaciones 2

system.data

Incluir versión preliminar

System.Data.SqlClient por Microsoft, 251M descargas

Provides the data provider for SQL Server. These classes provide access to versions of SQL Server and encapsulate database-specific protocols, including tabular data stream (TDS)

4.8.3

System.Data.Common por Microsoft, 116M descargas

Provides the base abstract classes, including System.Data.DbConnection and System.Data.DbCommand, for all data providers.

4.3.0

System.Runtime por Microsoft, 509M descargas

Provides the fundamental primitives, classes and base classes that define commonly-used value and reference data types, events and event handlers, interfaces, attributes, and exceptions. This...

4.3.1

System.IO por Microsoft, 468M descargas

Provides base input and output (I/O) types, including System.IO.Stream, System.IO.StreamReader

4.3.0

La licencia de cada paquete la concede su propietario. NuGet no se hace responsable de los paquetes de terceros ni concede ninguna licencia para ellos.

No volver a mostrar

System.Data.SqlClient nuget.org

Instalado: 4.8.3 Desinstalar

Versión: 4.8.3 Actualizar

Opciones

Descripción

Provides the data provider for SQL Server. These classes provide access to versions of SQL Server and encapsulate database-specific protocols, including tabular data stream (TDS)

Commonly Used Types:

System.Data.SqlClient.SqlConnection
System.Data.SqlClient.SqlException
System.Data.SqlClient.SqlParameter
System.Data.SqlClient.SqlDbType
System.Data.SqlClient.SqlDataReader
System.Data.SqlClient.SqlCommand

Explorador de soluciones

Solución "SQL_CRUD" (3 de 3 proyectos)

SQL_CRUD

Dependencias

App.xaml

AssemblyInfo.cs

MainPage.xaml

SQL_CRUD.Android

SQL_CRUD.iOS

Explorador de soluciones Cambios de GIT

Propiedades

Introducción (SSIS) Propiedades

Lista de errores

Toda la solución 0 Errores 0 Advertencias 0 Mensajes Compilación + IntelliSense

Descripción

Proyecto Archivo

Sugerencias de IntelliCode Lista de errores Salida

Lista de errores de búsqueda

Agregar al control de código fuente

En la clase del **LOGIN**, usando los roles de la base de datos según el usuario, que está disponible en **YouTube**, vamos a hacer algunas modificaciones ahora ese proyecto con los siguientes datos, tambien en el archivo y manual que les compartí les dejé el **script** de la base de datos en este de sql server, con todas las tablas, de ellas no utilizamos la de Articulos ni de ventas, ahora vamos a hacer ese proyecto.

Una vez has instalado tu programa de SQL Server, ahora abres una nueva consulta, en la cual vas a copiar y pegar este código, el cual es la base de datos, la tabla, la inserción de datos, y algunas consultas que te ayudaran a ver esos registros.

```
--crear una base de datos SECCION-PDM:
```

```
create database LOGIN_PDM  
GO
```

```
--USAR ESA BASE DE DATOS:
```

```
USE LOGIN_PDM  
GO
```

```
-- Creación de tabla de roles
```

```
CREATE TABLE roles (  
    id_rol INT PRIMARY KEY,  
    nombre_rol VARCHAR(50) NOT NULL  
);
```

```
select * from roles
```

```
-- Creación de tabla de usuarios
CREATE TABLE usuarios (
    id_usuario INT PRIMARY KEY,
    nombre_usuario VARCHAR(50) NOT NULL,
    password VARCHAR(50) NOT NULL,
    id_rol INT NOT NULL,
    FOREIGN KEY (id_rol) REFERENCES roles(id_rol)
);

select * from usuarios

-- Insertando registros en tabla de roles
INSERT INTO roles (id_rol, nombre_rol) VALUES (1, 'administrador');
INSERT INTO roles (id_rol, nombre_rol) VALUES (2, 'supervisor');
INSERT INTO roles (id_rol, nombre_rol) VALUES (3, 'vendedor');

-- Insertando registros en tabla de usuarios
INSERT INTO usuarios (id_usuario, nombre_usuario, password, id_rol) VALUES (1, 'JUANCITO', 'ADMIN@123', 1);
INSERT INTO usuarios (id_usuario, nombre_usuario, password, id_rol) VALUES (2, 'DARIEL', 'DARIEL@123', 2);
INSERT INTO usuarios (id_usuario, nombre_usuario, password, id_rol) VALUES (3, 'DANIELA', 'DANIELA@123', 3);
INSERT INTO usuarios (id_usuario, nombre_usuario, password, id_rol) VALUES (4, 'MARIA', 'MARIA@123', 1);
INSERT INTO usuarios (id_usuario, nombre_usuario, password, id_rol) VALUES (5, 'YENNEFER ', 'YENNEFER@123', 2);
```

-- Consultas básicas

-- Seleccionar todos los registros de la tabla de roles
SELECT * FROM roles;

-- Seleccionar todos los registros de la tabla de usuarios
SELECT * FROM usuarios;

-- Seleccionar un registro específico de la tabla de roles
SELECT * FROM roles WHERE id_rol = 2;

-- Seleccionar un registro específico de la tabla de usuarios
SELECT * FROM usuarios WHERE id_usuario = 4;

-- Consultas avanzadas

-- Seleccionar todos los usuarios y sus roles
SELECT usuarios.*, roles.nombre_rol
FROM usuarios
INNER JOIN roles ON usuarios.id_rol = roles.id_rol;

-- Seleccionar los usuarios que tengan un rol específico
SELECT * FROM usuarios WHERE id_rol = 3;

```
-- Contar la cantidad de usuarios por cada rol
SELECT roles.nombre_rol, COUNT(usuarios.id_usuario) AS cantidad_usuarios
FROM usuarios
INNER JOIN roles ON usuarios.id_rol = roles.id_rol
GROUP BY roles.nombre_rol;
```

```
-- Actualizar un registro en la tabla de roles
UPDATE roles SET nombre_rol = 'administrador principal' WHERE id_rol = 1;
```

```
-- Eliminar un registro en la tabla de usuarios
DELETE FROM usuarios WHERE id_usuario = 8;
```

```
SELECT * FROM usuarios;
SELECT * FROM roles;
```

```
-- Creación de la tabla usuario
```

```
CREATE TABLE usuario (
    id_usuario INT NOT NULL,
    nombre_user VARCHAR(20) NOT NULL,
    telefono VARCHAR(30) NOT NULL,
    email VARCHAR(30) NOT NULL
);
```

```
SELECT * FROM usuario
```

```
-- Insertar datos en la tabla usuario
```

```
INSERT INTO usuario VALUES (1, 'JUANCITO', '809-555-8789', 'JUANCITO@GMAIL.COM');
```

```
-- Creación de la tabla Articulos
```

```
CREATE TABLE Articulos (
    IDArticulo INT PRIMARY KEY,
    Nombre NVARCHAR(100),
    Precio DECIMAL(10, 2)
);
```

```
-- Insertar datos en la tabla Articulos
```

```
INSERT INTO Articulos (IDArticulo, Nombre, Precio)  
VALUES  
    (1, 'Laptop Legion', 850.50),  
    (2, 'Mouse Gamer', 30.25),  
    (3, 'Monitor Gamer 32', 250.75),  
    (4, 'Bocinas', 12.90),  
    (5, 'UPS de 1200 kw', 120.00);
```

```
-- Creación de la tabla Ventas
```

```
CREATE TABLE Ventas (  
    IDVenta INT IDENTITY(1, 1) PRIMARY KEY,  
    IDArticulo INT,  
    Cantidad INT,  
    Precio DECIMAL(18, 2)  
);
```

```
-- Consulta de la tabla Ventas
```

```
SELECT * FROM Ventas;
```

-- Insertar 10 artículos tecnológicos en la tabla Articulos

```
INSERT INTO Articulos (IDArticulo, Nombre, Precio)
```

```
VALUES
```

```
(6, 'Smartphone', 500.00),  
(7, 'Laptop Legion', 1000.00),  
(8, 'Tablet', 300.00),  
(9, 'Smartwatch', 150.00),  
(10, 'Gafas de Realidad Virtual', 200.00),  
(11, 'Auriculares Inalámbricos', 100.00),  
(12, 'Altavoz Inteligente', 80.00),  
(13, 'Cámara de Acción', 250.00),  
(14, 'Monitor Curvo', 300.00),  
(15, 'Consola de Videojuegos', 400.00);
```

-- Consulta de la tabla Articulos

```
SELECT * FROM Articulos;
```

LOGIN SECCION_PD...3 (JUANCITO (60))*

```
113  --Ver tabla y registro roles:  
114  select * from roles;  
115  --Ver tabla y registros usuarios:  
116  select * from usuarios  
117  --Ver tabla y registros usuarios:  
118  select * from usuario  
119  -- Consulta de la tabla Articulos  
120  SELECT * FROM Articulos;  
121  -- Consulta de la tabla Ventas  
122  SELECT * FROM Ventas.
```

129 %

Results Messages

	id_rol	nombre_rol
1	1	administrador
2	2	supervisor
3	3	vendedor

	id_usuario	nombre_usuario	password	id_rol
1	1	JUANCITO	ADMIN@123	1
2	2	DARIEL	DARIEL@123	2
3	3	DANIELA	DANIELA@123	3
4	4	MARIA	MARIA@123	1
5	5	YENNEFER	YENNEFER@123	2

	id_usuario	nombre_user	telefono	email
1	1	PEDRO JOSE	809-789-0000	pedro@EMAIL.COM
2	2	Dariel Adolfo	809-507-4322	DARIEL@mmsil.com
3	3	María Vizcaino	809-777-8000	maria@EMAIL.COM
4	7	PEDRO JOSE	809-789-0000	pedro@EMAIL.COM
5	5	MIGUEL GUERA	809-789-7896	MIGUEL@EMAIL.COM
6	9	HOLA NUEVO	555-88888	NUEVOEMAIL.COM

	IDArticulo	Nombre	Precio
1	1	Laptop Legion	850.50
2	2	Mouse Gamer	30.25
3	3	Monitor Gamer 32	250.75
4	4	Bocinas	12.90
5	5	UPS de 1200 kw	120.00
6	6	Smartphone	500.00
7	7	Laptop	1000...
8	8	Tablet	300.00

	IDVenta	IDArticulo	Cantidad	Precio
1	20	1	1	850.50
2	21	5	5	120.00

Query executed successfully.

Ubicamos nuestra IP de nuestro que servira como un Servidor para nuestra base de datos.

```
Selección C:\WINDOWS\system32\cmd.exe

Adaptador de LAN inalámbrica Conexión de área local* 10:
  Estado de los medios. . . . . : medios desconectados
  Sufijo DNS específico para la conexión. . . :

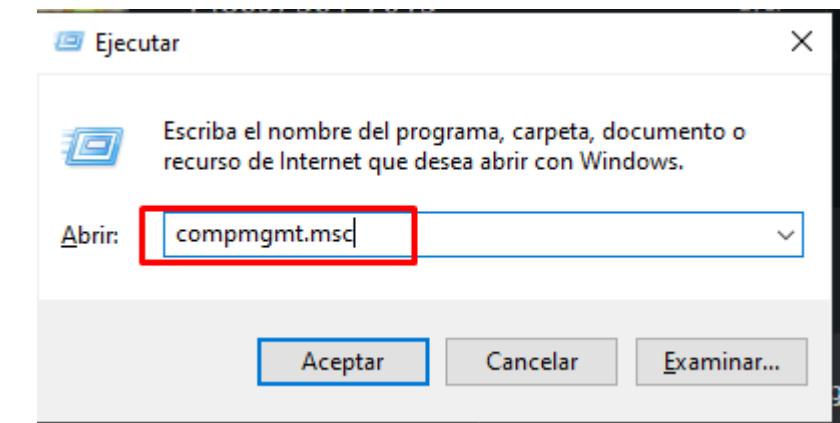
Adaptador de Ethernet Ethernet:
  Sufijo DNS específico para la conexión. . .
  Dirección IPv6 . . . . . : fd00:9acd:1947:9900:48de:8a37:c6a4:4c2b
  Dirección IPv6 temporal. . . . . : fd00:9acd:1947:9900:9539:95fa:7d41:9a1f
  Vínculo: dirección IPv6 local. . . . : fe80::48de:8a37:c6a4:4c2b%23
  Dirección IPv4. . . . . : 10.0.0.4
  Máscara de subred . . . . . : 255.255.255.0
  Puerta de enlace predeterminada . . . . . : 10.0.0.1

Adaptador de Ethernet Conexión de red Bluetooth:
  Estado de los medios. . . . . : medios desconectados
  Sufijo DNS específico para la conexión. . . :

Adaptador de Ethernet vEthernet (Default Switch):
  Sufijo DNS específico para la conexión. . .
  Vínculo: dirección IPv6 local. . . . : fe80::684e:ad8e:8ee0:30f1%52
  Dirección IPv4. . . . . : 172.29.144.1
  Máscara de subred . . . . . : 255.255.240.0
  Puerta de enlace predeterminada . . . . . :

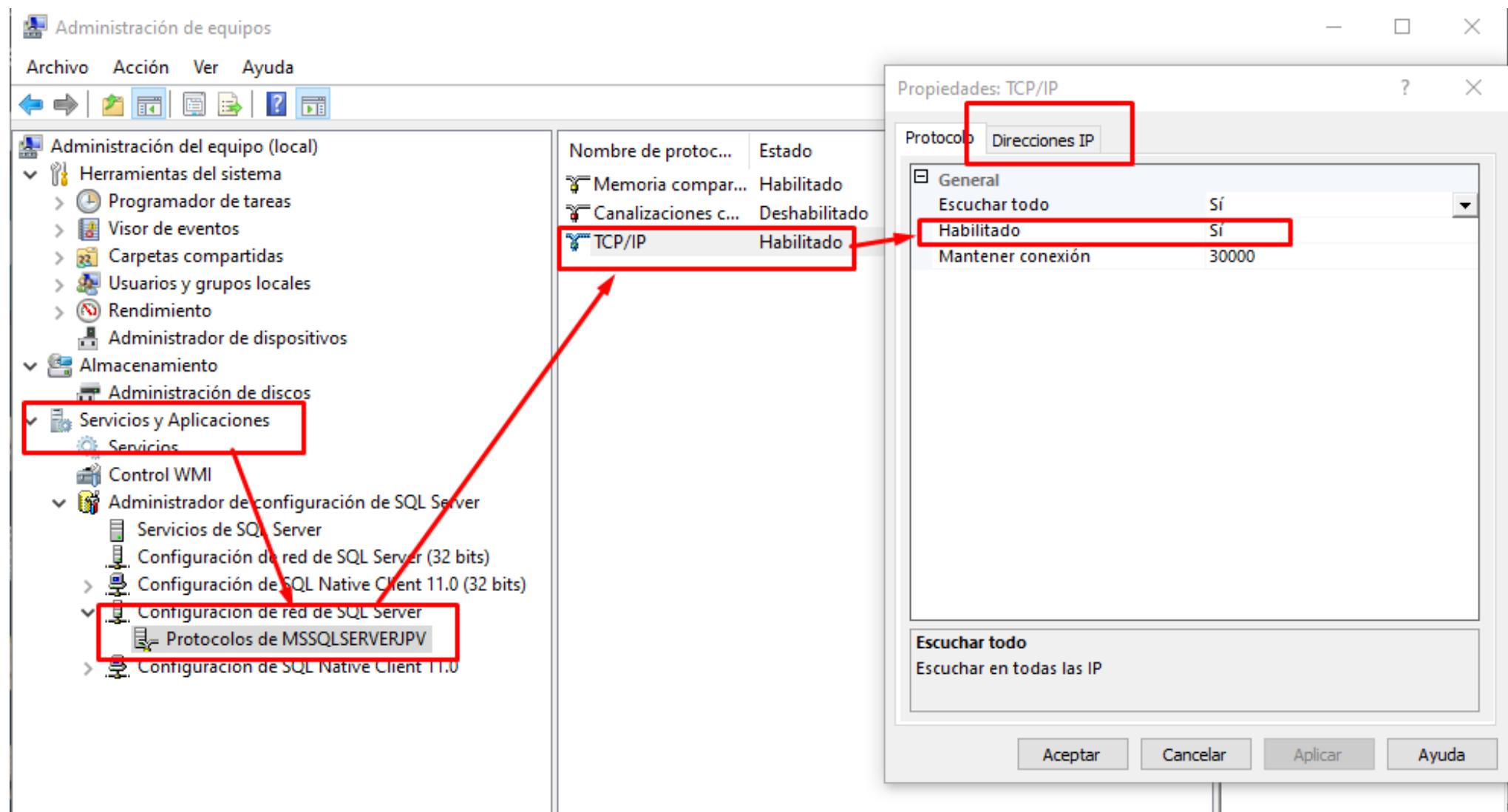
C:\Users\juanc>
```

compmgmt.msc

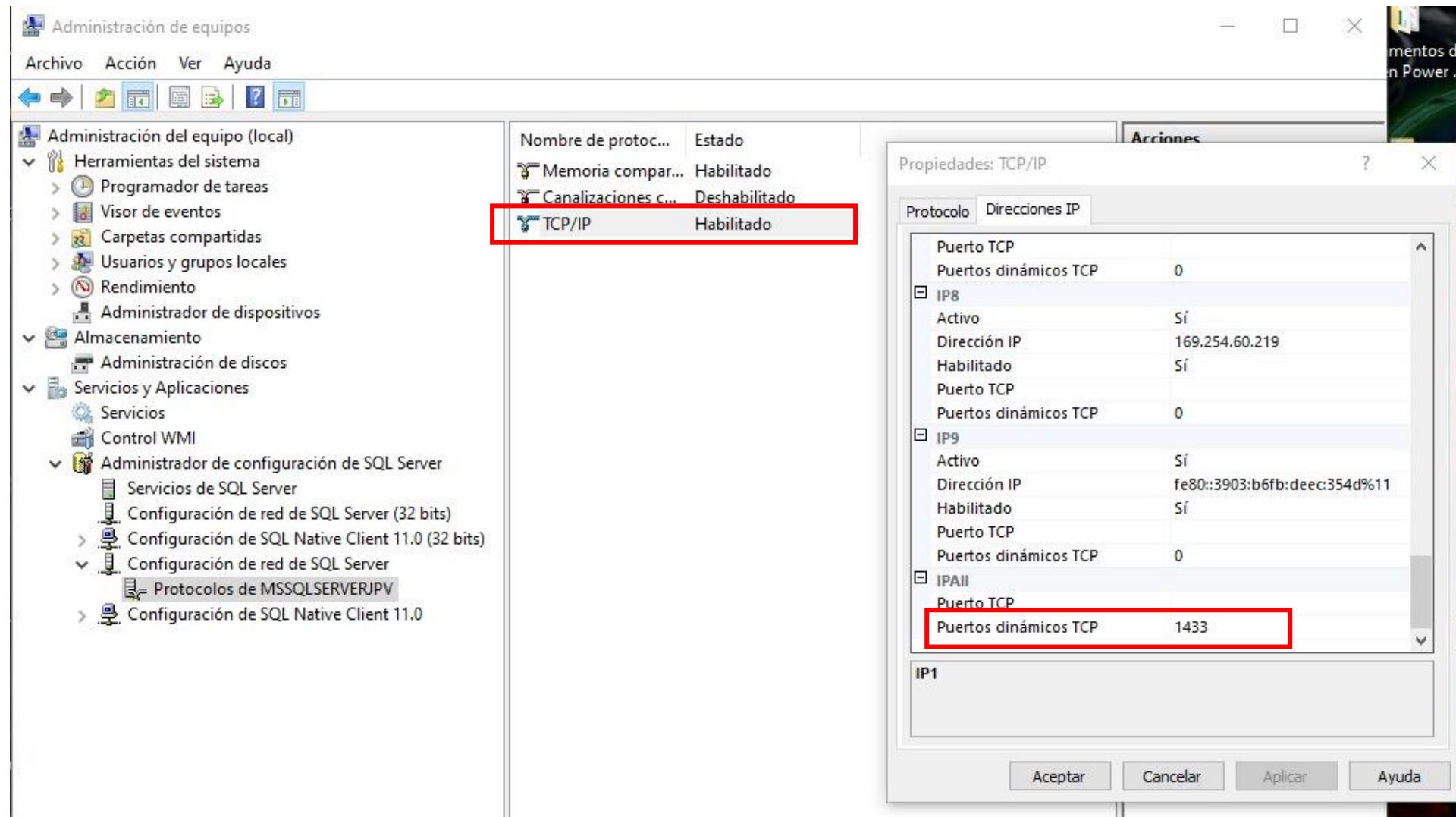


Habilitar el protocolo TCP/IP sobre la dirección IP del server

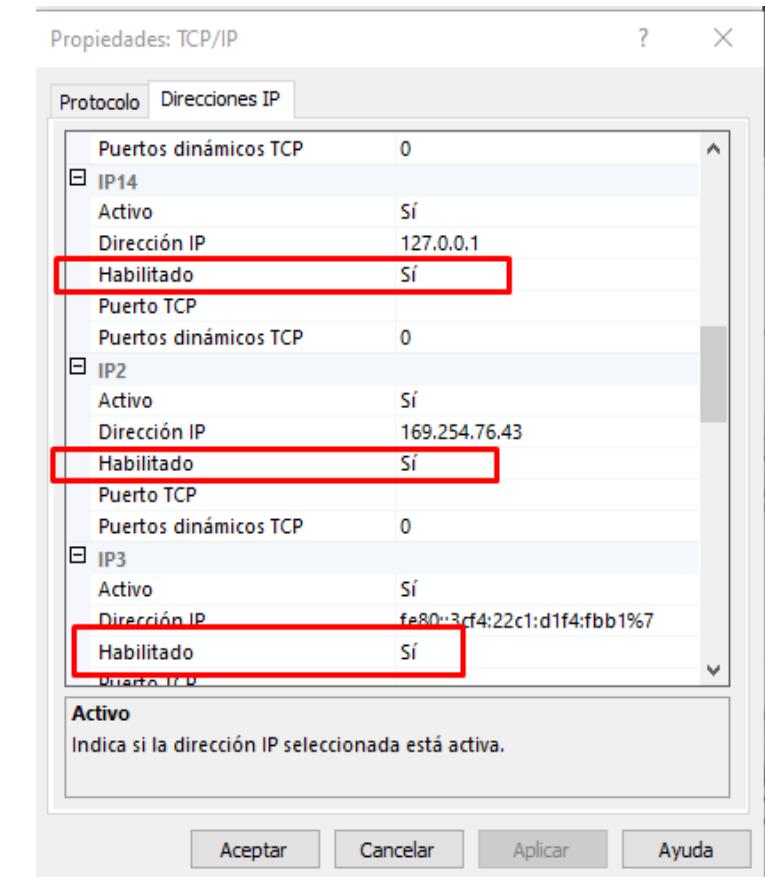
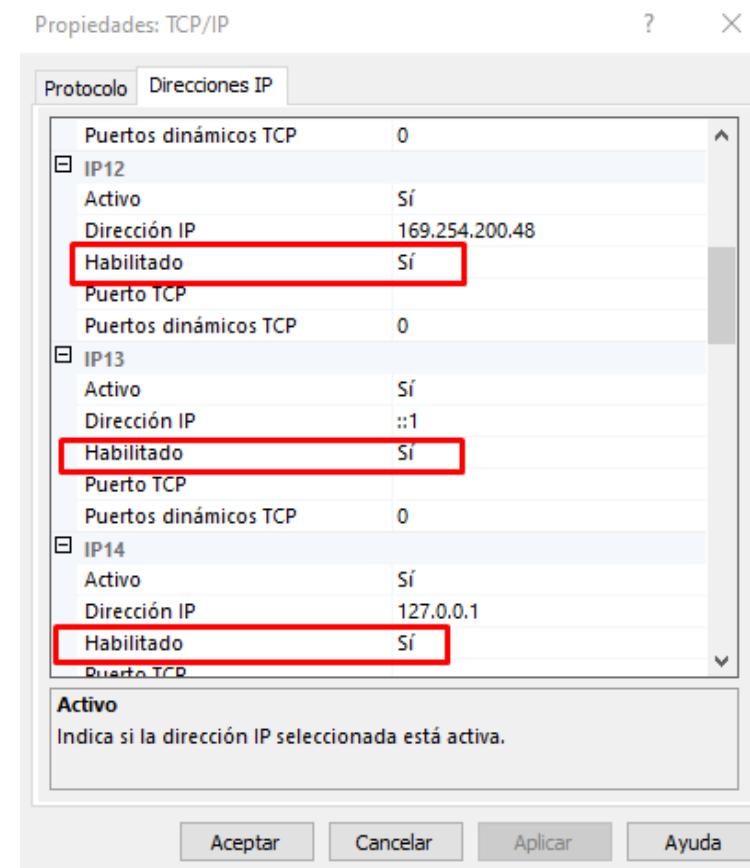
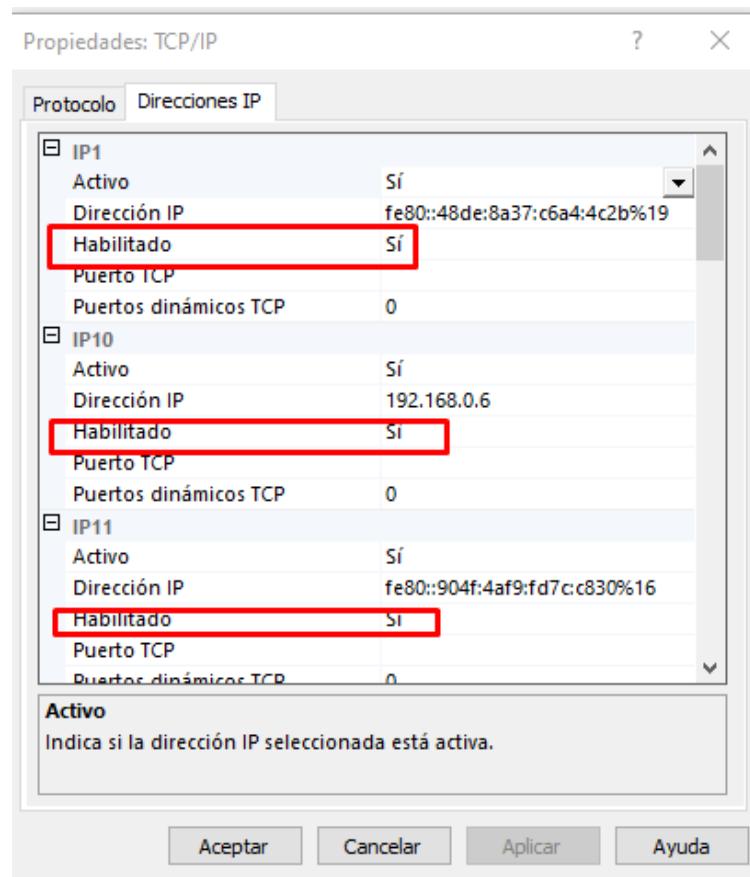
Abrir en el servidor el programa **Administrador de configuración de SQL Server** (Sql Server Configuration Manager), desde el menú de aplicaciones de SQL Server, o buscando el programa en el buscador del server.



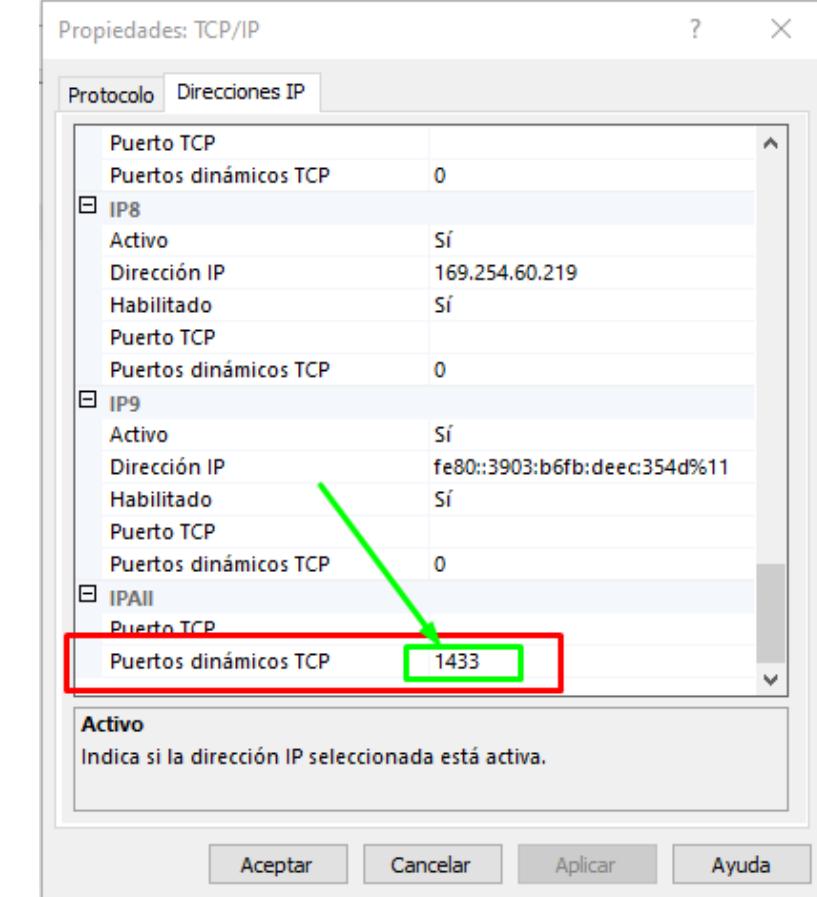
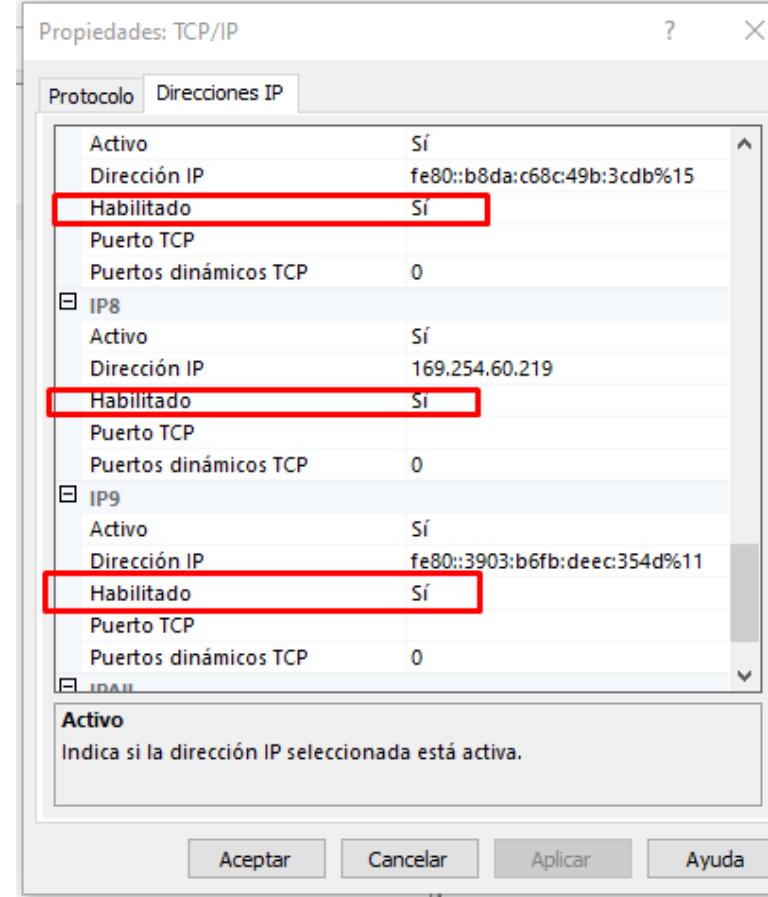
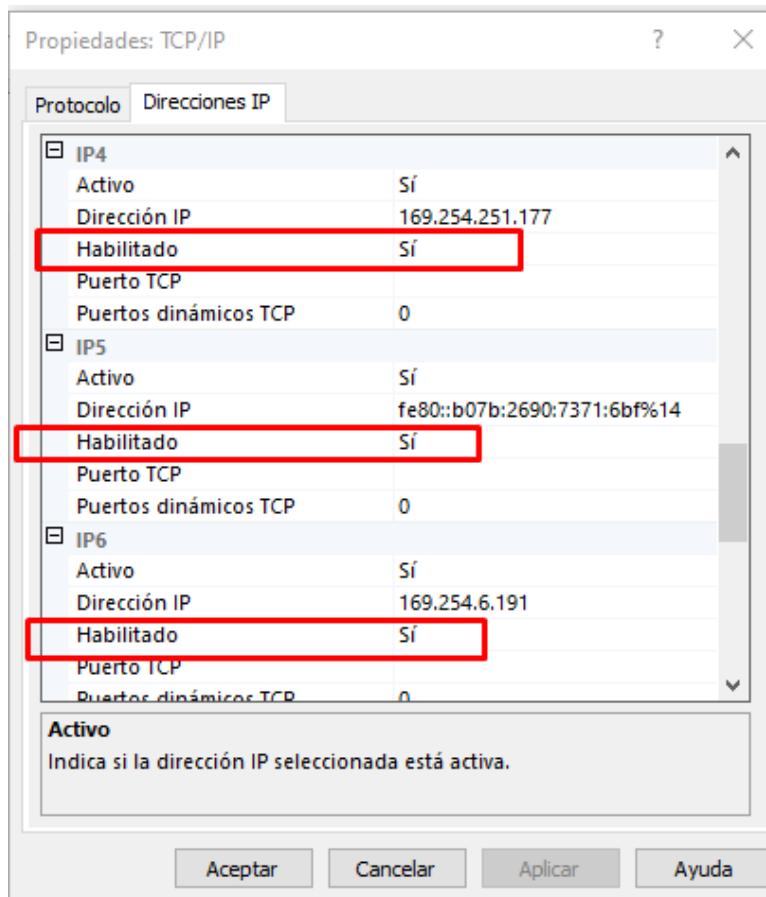
Una vez abierto, en el menú de la izquierda, en 'Configuración de red de SQL Server', Seleccionar 'Protocolos de [Nombre de la instancia]', y hacer doble click sobre el Nombre de protocolo TCP/IP para acceder a sus propiedades y seleccionar 'Sí' en 'Habilitado', y en la pestaña de Direcciones IP, seleccionar también 'Sí' en 'Habilitado' de la IP del server, este caso 192.168.1.116, y asegurarse de que el puerto es correcto (en este caso 1433 ya estaba bien porque es el puerto por defecto, pero si no hay que modificarlo)



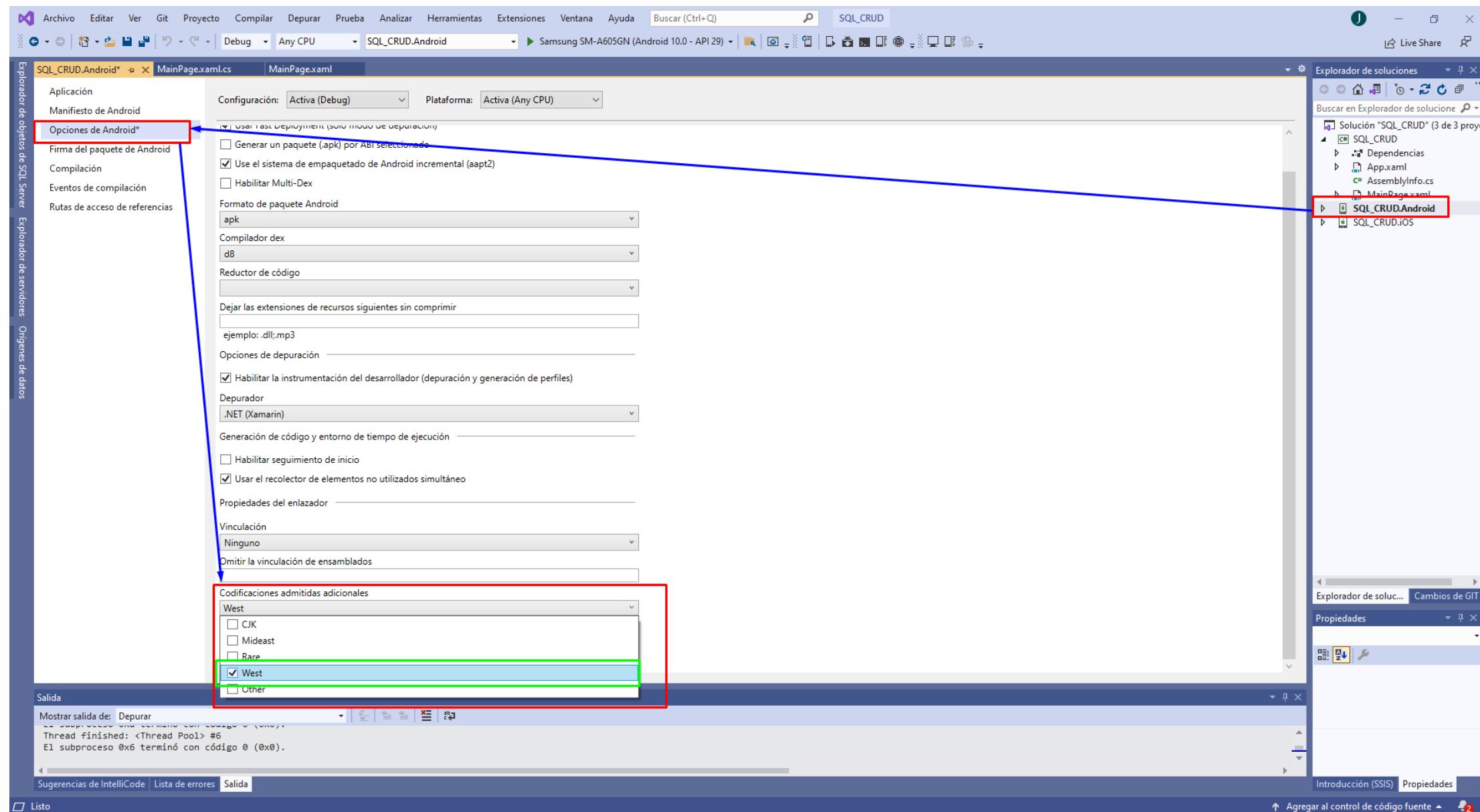
Verifico que todos los puertos ip esten habilitados, sino los habilito.



y asegurarse de que el puerto es correcto (en este caso 1433 ya estaba bien porque es el puerto por defecto, pero si no hay que modificarlo)



El siguiente Paso es ir al proyecto de Android, clic derecho, propiedades, ir a opciones de android, ir a codificaciones admitidas adicionales, marcamos West.

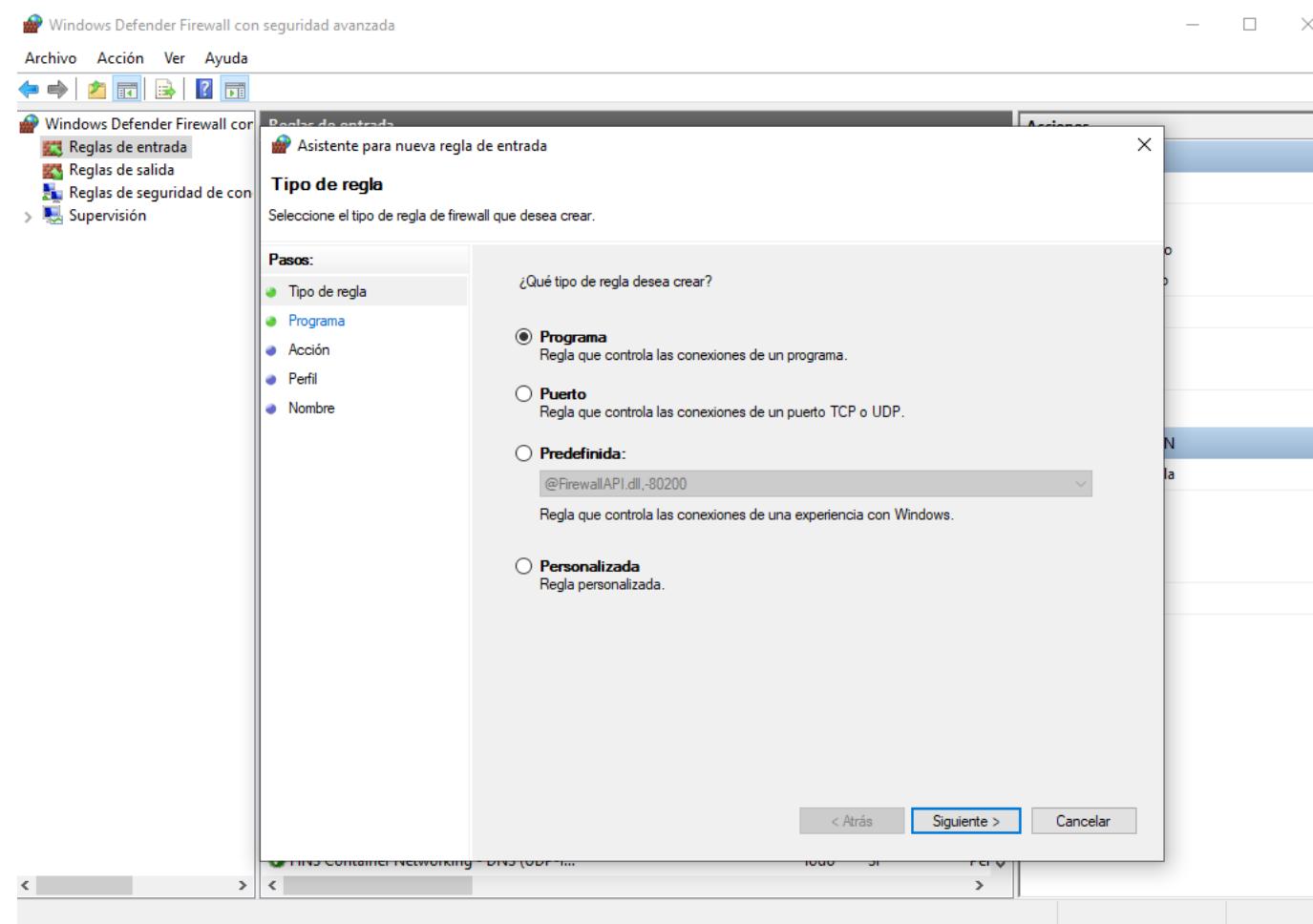


En el contexto de las opciones de codificación admitidas adicionales en **Android Studio**, "**West**" se refiere a la codificación occidental (también conocida como codificación **ISO-8859-1**), que es un estándar para representar caracteres en los idiomas de Europa occidental. Esta codificación cubre la mayoría de los idiomas europeos, incluyendo el **inglés, francés, español, alemán, italiano, portugués**, entre otros.

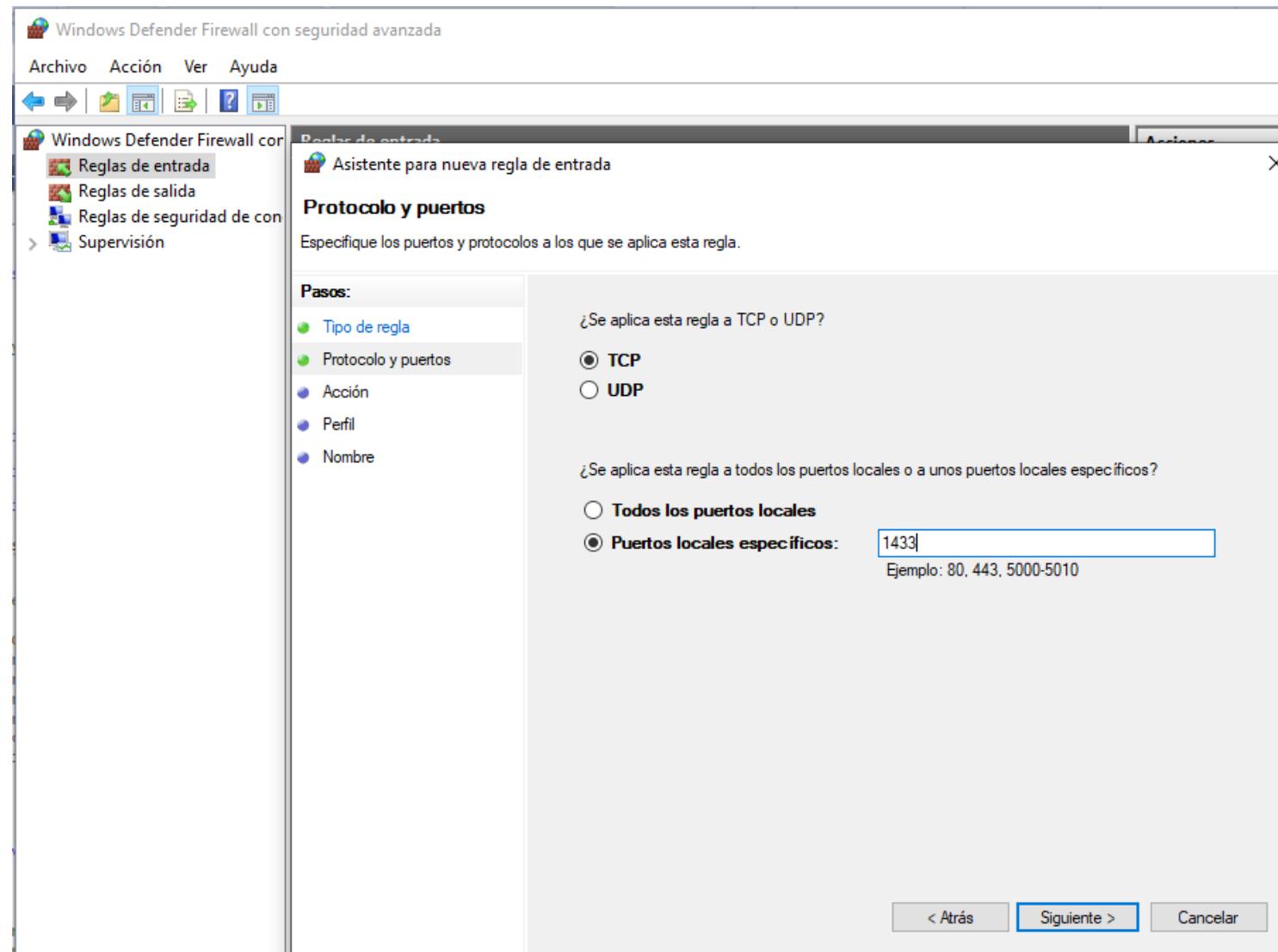
Abrir el puerto TCP desde el Firewall de Windows Server

El siguiente paso es abrir la aplicación Firewall de Windows, desde el panel de control (Panel de control\Sistema y seguridad\Firewall de Windows), o buscando Firewall en el buscador de aplicaciones.

Una vez abierto, seleccionar '*Configuración avanzada*' en el menú de la izquierda, y después, en el menú hacer click con el botón derecho sobre '*Reglas de entrada*', y seleccionar '*Nueva Regla*' (También se puede hacer desde la opción de menú '*Acción*')

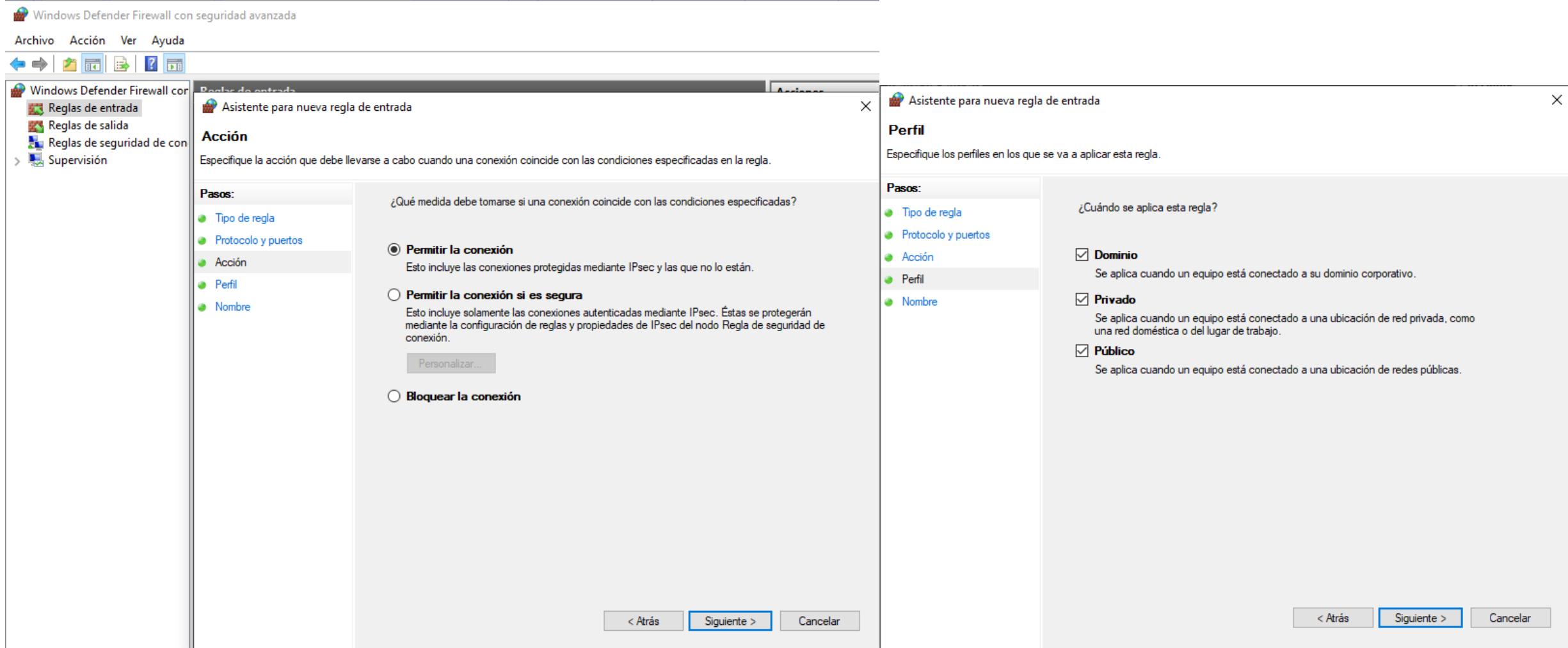


Seleccionar regla tipo '*Puerto*', y después '**TCP**', y escribir **1433** en la opción '*Puerto específico local*'. Esta es la configuración más sencilla, y sirve si esta es la única instancia de SQL Server instalada en el Servidor Windows.



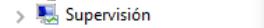
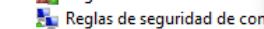
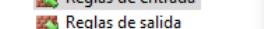
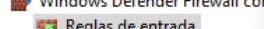
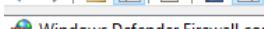
Si hubiera más instancias sería necesario abrir más puertos, activar el servicio **SQL Server Browser**, y abrir otro puerto UDP, y también crear una nueva regla personalizada en el firewall para el servicio.

Seleccionando las siguientes opciones (normalmente las que vienen por defecto ya van bien) se finaliza la creación de la regla del firewall, y el telnet por el puerto **1433** desde la máquina cliente ya debería responder:



Windows Defender Firewall con seguridad avanzada

Archivo Acción Ver Ayuda



Reglas de entrada

Reglas de salida

Reglas de seguridad de conexión

Supervisión

Reglas de entrada

Asistente para nueva regla de entrada

Nombre

Especifique el nombre y la descripción de esta regla.

Pasos:

- 1 Tipo de regla
- 2 Protocolo y puertos
- 3 Acción
- 4 Perfil
- 5 Nombre

Nombre:

CRUD_XAMARIN_SQL

Descripción (opcional):

CRUD_XAMARIN_SQL

< Atrás

Finalizar

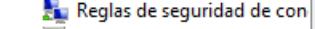
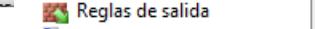
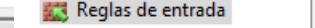
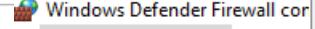
Cancelar

<

>

Windows Defender Firewall con seguridad avanzada

Archivo Acción Ver Ayuda



Reglas de entrada

Reglas de salida

Reglas de seguridad de conexión

Supervisión

Reglas de entrada

Nombre

CRUD_XAMARIN_SQL

CRUD_SQL_XAMARIN_SQL

AnyDesk

AnyDesk

AnyDesk

AnyDesk

AnyDesk

AnyDesk

anydesk.exe

anydesk.exe

AutoKMS

broker.exe

broker.exe

code.exe

code.exe

coppeliasim

coppeliasim

eyebeam.exe

eyebeam.exe

HNS Container

Reglas de salida

Reglas de seguridad de conexión

Supervisión

Propiedades: CRUD_XAMARIN_SQL

Opciones avanzadas

Entidades de seguridad locales

Usuarios remotos

General

Programas y servicios

Equipos remotos

Protocolos y puertos

Ámbito

General

Nombre:

CRUD_XAMARIN_SQL

Descripción:

CRUD_XAMARIN_SQL

Habilitado

Acción

Permitir la conexión

Permitir la conexión si es segura

Personalizar

Bloquear la conexión

Aceptar

Cancelar

Aplicar

1000

Si

Todo

Sí

X

Acción
Permitir la conexión
Permitir la conexión si es segura
Personalizar
Bloquear la conexión

Firewall de Windows Defender

Panel de control > Todos los elementos de Panel de control > Firewall de Windows Defender

Archivo Edición Ver Herramientas

Ayudar a proteger el equipo con Firewall de Windows Defender

Firewall de Windows Defender puede ayudar a impedir que piratas informáticos o software malintencionado obtengan acceso al equipo a través de Internet o una red.

Actualizar configuración de firewall

Firewall de Windows Defender no está usando la configuración recomendada para proteger el equipo.

Usar la configuración recomendada

¿Cuál es la configuración recomendada?

Redes privadas Conectado

Redes domésticas o del trabajo en cuyos usuarios y dispositivos confíe

Estado de Firewall de Windows Defender: Desactivado

Conexiones entrantes: Bloquear todas las conexiones a aplicaciones que no estén en la lista de aplicaciones permitidas

Redes privadas activas: NEO-JPV

Estado de notificación: Notificarme cuando Firewall de Windows Defender bloquee una nueva aplicación

Redes públicas o invitadas Conectado

Redes en lugares públicos como aeropuertos o cafeterías

Estado de Firewall de Windows Defender: Desactivado

Conexiones entrantes: Bloquear todas las conexiones a aplicaciones que no estén en la lista de aplicaciones permitidas

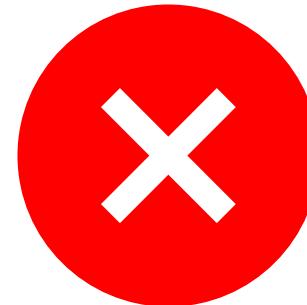
Redes públicas activas: Ninguno

Estado de notificación: Notificarme cuando Firewall de Windows Defender bloquee una nueva aplicación

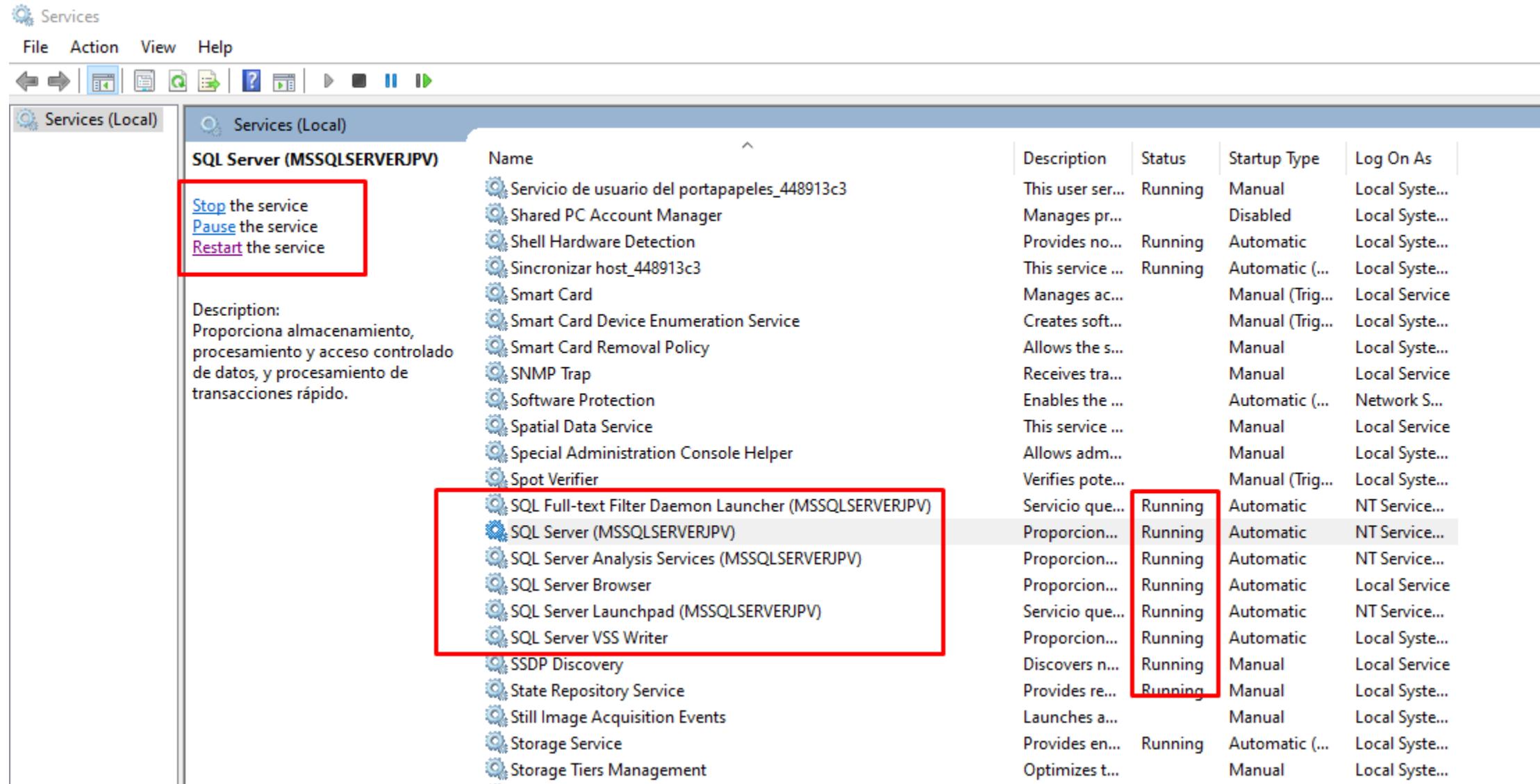
SI TE DA ERROR EN LA EJECUCION DEL PROGRAMA CUANDO ESTE LISTO CON EL CODIGO C#. SIGUE ESTOS.

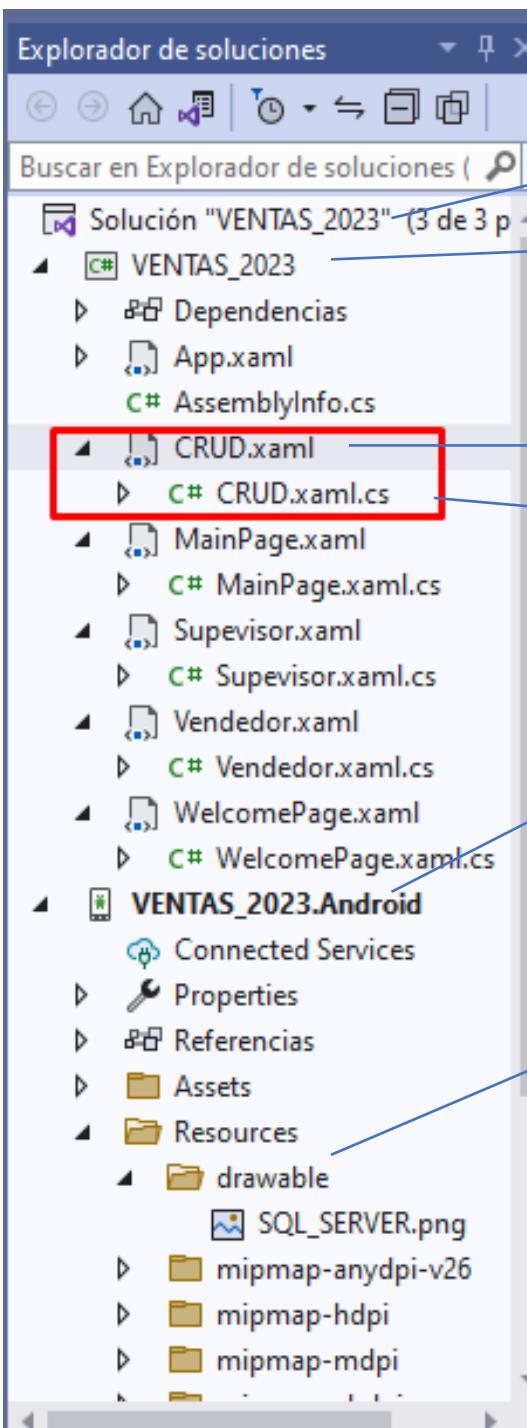
- 1) Abre el Administrador de servicios:** Puedes buscar "services.msc" en el menú Inicio o en la barra de búsqueda de Windows y abrir el "Administrador de servicios".
- 2) Encuentra el servicio de SQL Server:** En la lista de servicios, busca los servicios relacionados con SQL Server. Por lo general, verás servicios con nombres como "SQL Server (Nombre de la instancia)".
- 3) Reinicia el servicio de SQL Server:** Haz clic derecho en el servicio de SQL Server que deseas configurar y selecciona "Reiniciar". Esto reiniciará el servicio y aplicará cualquier cambio en la configuración.
- 4) Configura el Firewall:** Además de reiniciar el servicio, asegúrate de configurar el firewall para permitir conexiones entrantes en el puerto utilizado por SQL Server (generalmente el puerto 1433). Puedes crear una regla de entrada en el Firewall de Windows para permitir el tráfico en ese puerto.

Ten en cuenta que estos pasos pueden variar dependiendo de la versión de SQL Server que estés utilizando y de la configuración específica de tu entorno. Si sigues teniendo problemas para habilitar conexiones remotas, te recomiendo consultar la documentación de SQL Server específica para tu versión o buscar ayuda en foros de soporte técnico relacionados con SQL Server.



Como mencione anteriormente, muchas veces necesitamos reiniciar el servicio de SQL Server, ya que a veces se para según las acciones que hemos realizado.





Solucion o Proyecto Principal

Proyecto Compartido con C#

Pagina o Funcion Principal con XAML

Pagina principal con C# logica del Programa

Proyecto Android y Ejecutable por defecto.

Recurso o Imagen para el Banner o Encabezado

En el proyecto anterior modificamos y creamos algunos archivos en este vamos a crear uno que se llame CRUD, y lo vamos a agregar a estos.

```
using System;
using Xamarin.Forms;
using Xamarin.Forms.Xaml;

namespace VENTAS_2023
{
    public partial class App : Application
    {
        public App()
        {
            InitializeComponent();

            MainPage = new NavigationPage( new MainPage());
        }

        protected override void OnStart()
        {
        }

        protected override void OnSleep()
        {
        }

        protected override void OnResume()
        {
        }
    }
}
```

Modificamos el
App.xaml.cs

```

<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
    xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
    x:Class="VENTAS_2023.CRUD"
    Title="Operaciones de CRUD">
    <!-- Título de la página -->

    <ContentPage.Content>
        <!-- StackLayout que contiene el contenido de la página -->
        <StackLayout>
            <!-- ScrollView para permitir desplazamiento en caso de contenido grande -->
            <ScrollView>
                <!-- Grid para organizar los elementos en filas y columnas -->
                <Grid RowDefinitions="Auto,Auto,Auto">
                    <!-- Definición de tres filas -->

                    <!-- Sección de entrada de datos y título -->
                    <StackLayout Orientation="Vertical" Grid.Row="0">
                        <!-- StackLayout para organizar verticalmente -->
                        <!-- Marco con título -->
                        <Frame BackgroundColor="#7f8fa6" Padding="24" CornerRadius="0">
                            <Label Text="CRUD SQL SERVER Y XAMARIN" HorizontalTextAlignment="Center" TextColor="White" FontSize="20" Margin="0" />
                        </Frame>
                        <!-- Imagen O LOGO-->
                        <Image Source="SQL_SERVER.png" Margin="5"/>
                        <!-- Campos de entrada -->
                        <Entry Placeholder="Introduzca el ID Usuario" x:Name="id_usuario" HorizontalTextAlignment="Center" FontAttributes="Bold" TextColor="Black" BackgroundColor="Azure"/>
                        <Entry Placeholder="Introduzca el Nombre de Usuario" x:Name="nombre_user" HorizontalTextAlignment="Center" FontAttributes="Bold" TextColor="Black" BackgroundColor="Azure"/>
                        <Entry Placeholder="Introduzca el Telefono" x:Name="telefono" HorizontalTextAlignment="Center" FontAttributes="Bold" TextColor="Black" BackgroundColor="Azure"/>
                        <Entry Placeholder="Introduzca el Correo" x:Name="email" HorizontalTextAlignment="Center" FontAttributes="Bold" TextColor="Black" BackgroundColor="Azure"/>
                    </StackLayout>

                    <!-- Sección de botones -->
                    <Grid Grid.Row="2" RowDefinitions="Auto,Auto" ColumnDefinitions="*,*">
                        <!-- Dos filas, dos columnas -->
                        <!-- Botones con diferentes iconos y acciones -->
                        <Button Grid.Row="0" Grid.Column="0" Text="⌚ Validar Conexion" BackgroundColor="#2196F3" TextColor="White" Clicked="Button_Clicked"/>
                        <Button Grid.Row="0" Grid.Column="1" Text="📋 Mostrar Datos" BackgroundColor="#1e3799" TextColor="White" x:Name="getbutton" Clicked="Getbutton_Clicked" />
                        <Button Grid.Row="1" Grid.Column="0" Text="🔍 Buscar" BackgroundColor="#48dbfb" TextColor="White" x:Name="btnBuscar" Clicked="BtnBuscar_Clicked"/>
                        <Button Grid.Row="1" Grid.Column="1" Text="➕ Insertar" BackgroundColor="#e1b12c" TextColor="White" x:Name="postbutton" Clicked="Postbutton_Clicked"/>
                        <Button Grid.Row="2" Grid.Column="0" Text="📝 Actualizar" BackgroundColor="#44bd32" TextColor="White" x:Name="updatebutton" Clicked="Updatebutton_Clicked"/>
                        <Button Grid.Row="2" Grid.Column="1" Text="⌫ Borrar" BackgroundColor="#eb2f06" TextColor="White" x:Name="deletebutton" Clicked="Deletebutton_Clicked"/>
                    </Grid>

                    <!-- Lista de elementos mostrados en ListView -->
                    <ListView Grid.Row="3" x:Name="MyListView" HasUnevenRows="True" Margin="1" BackgroundColor="LightGray">
                        <ListView.ItemTemplate>
                            <DataTemplate>
                                <ViewCell>
                                    <Grid RowDefinitions="Auto" ColumnDefinitions="1*, 3*, 4*, 6*" Padding="1" Margin="2" BackgroundColor="Bisque">
                                        <Frame Grid.Row="0" Grid.Column="0" BorderColor="Black" Padding="5">
                                            <Label Text="{Binding Id_usuario}" TextColor="Black" HorizontalTextAlignment="Center" VerticalTextAlignment="Center"></Label>
                                        </Frame>
                                        <Frame Grid.Row="0" Grid.Column="1" BorderColor="Black" Padding="5">
                                            <Label Text="{Binding Nombre_user}" TextColor="Black" HorizontalTextAlignment="Center" VerticalTextAlignment="Center"></Label>
                                        </Frame>
                                        <Frame Grid.Row="0" Grid.Column="2" BorderColor="Black" Padding="5">
                                            <Label Text="{Binding Telefono}" TextColor="Black" HorizontalTextAlignment="Center" VerticalTextAlignment="Center"></Label>
                                        </Frame>
                                        <Frame Grid.Row="0" Grid.Column="3" BorderColor="Black" Padding="5">
                                            <Label Text="{Binding Email}" TextColor="Black" HorizontalTextAlignment="Center" VerticalTextAlignment="Center"></Label>
                                        </Frame>
                                    </Grid>
                                </ViewCell>
                            </DataTemplate>
                        </ListView.ItemTemplate>
                    </ListView>
                </Grid>
            </ScrollView>
        </StackLayout>
    </ContentPage.Content>
</ContentPage>

```

Código XAML en el CRUD para Copiar

// Inicio del Código CRUD.xaml.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data.SqlClient; // Librerias y espacio de Nombre Necesarias para-Nuestro Proyecto.
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Xamarin.Forms;

namespace Coneccion_SQL
{
    public partial class MainPage : ContentPage
    {

        public class MyTableList
        {
            public int Id_usuario { get; set; } // MyTableList =Clase Principal de Mi
            public string Nombre_user { get; set; } proyecto con los Metodos y campos de mi Tabla
            public string Telefono { get; set; }
            public string Email { get; set; }
        }
        SqlConnection sqlConnection; // Declara un objeto de conexión
    }
}
```

Código XAML.CS en el CRUD para Copiar



Codigo XAML.CS en el CRUD para Copiar

// Metodo creado para la conexion con el SQL Server, con la Base de Datos y las Tablas con las credenciales del Servidor y la cadena de Conexion.

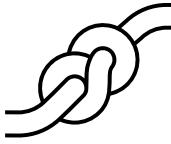


```
public CRUD()
{
    InitializeComponent();

    // Inicializa los parámetros de conexión
    string srvrdbname = "LOGIN_2023";
    string srvrname = "192.168.2.209";
    string srvrusername = "JUANCITO";
    string srvrpassword = "123456";
    string sqlconn = $"Data Source={srvrname};Initial Catalog={srvrdbname};User ID={srvrusername};Password={srvrpassword}";
    sqlConnection = new SqlConnection(sqlconn); // Crea una nueva instancia de conexión
}
```

// Fin Metodo creado para la conexion con el SQL Server, con la Base de Datos y las Tablas con las credenciales del Servidor y la cadena de Conexion.





Cadena de conexión **válida para nuestro caso, existen otras.**

El código establece una cadena de conexión para conectarse a una base de datos SQL Server. La cadena de conexión se utiliza para que la aplicación pueda acceder y gestionar datos en la base de datos. Veamos cada una de las partes de la cadena de conexión:

Data Source=10.0.0.4: Esto especifica la dirección IP o nombre del servidor de base de datos al que la aplicación se conectará. En este caso, la dirección IP es "**10.0.0.4**".

Initial Catalog=JPVBD: Indica el nombre de la base de datos a la que se desea conectarse. En este caso, la base de datos se llama "JPVBD".

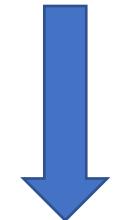
User ID=JUANCITO: Establece el nombre de usuario que se utilizará para autenticarse en el servidor de base de datos. En este caso, el nombre de usuario es "**JUANCITO**".

Password=123456: Proporciona la contraseña correspondiente al nombre de usuario para la autenticación. En este caso, la contraseña es "**123456**".

En resumen, esta cadena

// Código del Botón Confirmar Conexión al SQL Server evento clicked creado

Código XAML.CS en el CRUD
para Copiar



```
// Método para establecer una conexión a la base de datos
private async void Button_Clicked(object sender, EventArgs e)
{
    try
    {
        sqlConnection.Open(); // Abre la conexión a la base de datos
        await App.Current.MainPage.DisplayAlert("Alerta", "Conexión establecida correctamente.✓", "Ok");
        sqlConnection.Close(); // Cierra la conexión
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex.Message);
        throw;
    }
}
```

// Fin del Código del Botón Confirmar Conexión al SQL Server evento clicked creado



// Código del Botón Leer GetButton evento clicked creado

```
// Método para obtener datos de la base de datos y mostrarlos en la ListView
private async void Getbutton_Clicked(object sender, EventArgs e)
{
    try
    {
        List<MyTableList> myTableLists = new List<MyTableList>(); // Lista para almacenar los datos obtenidos
        sqlConnection.Open(); // Abre la conexión

        string queryString = "Select * from dbo.usuario"; // Consulta SQL
        SqlCommand command = new SqlCommand(queryString, sqlConnection);
        SqlDataReader reader = command.ExecuteReader(); // Ejecuta la consulta

        while (reader.Read()) // Recorre las filas obtenidas
        {
            // Crea un objeto MyTableList y asigna sus propiedades
            myTableLists.Add(new MyTableList
            {
                Id_usuario = Convert.ToInt32(reader["id_usuario"]),
                Nombre_user = reader["nombre_user"].ToString(),
                Telefono = reader["telefono"].ToString(),
                Email = reader["email"].ToString(),
            });
        }

        reader.Close();
        sqlConnection.Close(); // Cierra la conexión

        MyListView.ItemsSource = myTableLists; // Asigna los datos a la ListView
    }
    catch (Exception ex)
    {
        await App.Current.MainPage.DisplayAlert("Alerta", ex.Message, "Ok");
        throw;
    }
}
```

Código XAML.CS en el CRUD
para Copiar



// Fin del Código del Botón Leer GetButton evento clicked creado

// Código del Botón Guardar PostButton evento clicked creado

Código XAML.CS en el CRUD
para Copiar

```
// Método para insertar datos en la base de datos
private async void Postbutton_Clicked(object sender, EventArgs e)
{
    try
    {
        sqlConnection.Open(); // Abre la conexión

        // Prepara y ejecuta la consulta INSERT con parámetros
        using (SqlCommand command = new SqlCommand("INSERT INTO dbo.usuario VALUES(@id_usuario, @nombre_user, @telefono, @email)", sqlConnection))
        {
            command.Parameters.Add(new SqlParameter("id_usuario", id_usuario.Text));
            command.Parameters.Add(new SqlParameter("nombre_user", nombre_user.Text));
            command.Parameters.Add(new SqlParameter("telefono", telefono.Text));
            command.Parameters.Add(new SqlParameter("email", email.Text));
            command.ExecuteNonQuery(); // Ejecuta la consulta
        }

        sqlConnection.Close(); // Cierra la conexión
        await App.Current.MainPage.DisplayAlert("Alerta", ";Felicitaciones, los datos se han insertado correctamente!", "Ok");
    }
    catch (Exception ex)
    {
        await App.Current.MainPage.DisplayAlert("Alerta", ex.Message, "Ok");
        throw;
    }
}
```

// Fin del Código del Botón Guardar PostButton evento clicked creado

// Código del Botón Actualizar evento clicked creado

Código XAML.CS en el CRUD
para Copiar

```
// Método para actualizar datos en la base de datos
private async void Updatebutton_Clicked(object sender, EventArgs e)
{
    try
    {
        sqlConnection.Open(); // Abre la conexión

        // Obtiene los valores para la actualización
        int IdtoBeUpdated = Convert.ToInt32(id_usuario.Text);
        string NombreTobeUpdated = nombre_user.Text;
        string telefonoTobeUpdated = telefono.Text;
        string EmailTobeUpdated = email.Text;

        // Prepara y ejecuta la consulta UPDATE
        string qerystr = $"UPDATE dbo.usuario SET id_usuario='{IdtoBeUpdated}', nombre_user='{NombreTobeUpdated}', telefono='{telefonoTobeUpdated}', email='{EmailTobeUpdated}' WHERE id_usuario='{IdtoBeUpdated}'";
        using (SqlCommand command = new SqlCommand(qerystr, sqlConnection))
        {
            command.ExecuteNonQuery(); // Ejecuta la consulta
        }

        sqlConnection.Close(); // Cierra la conexión
        await App.Current.MainPage.DisplayAlert("Alerta", "¡Felicitaciones, el registro se ha actualizado correctamente!", "Ok");
    }
    catch (Exception ex)
    {
        await App.Current.MainPage.DisplayAlert("Alerta", ex.Message, "Ok");
        throw;
    }
}
```

// Fin Código del Botón Update evento clicked creado

// Código del Botón Borrar evento clicked creado

```
// Método para eliminar datos de la base de datos
private async void Deletebutton_Clicked(object sender, EventArgs e)
{
    try
    {
        sqlConnection.Open(); // Abre la conexión

        int idtodelete = Convert.ToInt32(id_usuario.Text);

        // Prepara y ejecuta la consulta DELETE
        using (SqlCommand command = new SqlCommand($"Delete FROM dbo.usuario WHERE id_usuario = {idtodelete}", sqlConnection))
        {
            command.ExecuteNonQuery(); // Ejecuta la consulta
        }

        sqlConnection.Close(); // Cierra la conexión
        await App.Current.MainPage.DisplayAlert("Alerta", "¡Felicidades, el registro se ha eliminado correctamente!", "Ok");
    }
    catch (Exception ex)
    {
        await App.Current.MainPage.DisplayAlert("Alerta", ex.Message, "Ok");
        throw;
    }
}
```

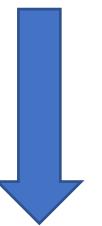
// Fin del Botón Eliminar

Código XAML.CS en el CRUD
para Copiar



// Código del Botón Borrar evento clicked creado

Código XAML.CS en el CRUD
para Copiar



```
// Método para buscar un registro específico en la base de datos
private void BtnBuscar_Clicked(object sender, EventArgs e)
{
    SqlCommand comando = new SqlCommand("Select * from dbo.usuario WHERE id_usuario=@id_usuario", sqlConnection);
    comando.Parameters.AddWithValue("@id_usuario", id_usuario.Text);
    sqlConnection.Open();

    SqlDataReader registro = comando.ExecuteReader();

    if (registro.Read()) // Si se encuentra un registro
    {
        // Llena los campos de texto con los valores obtenidos
        nombre_user.Text = registro["nombre_user"].ToString();
        telefono.Text = registro["telefono"].ToString();
        email.Text = registro["email"].ToString();
        sqlConnection.Close();
    }

    sqlConnection.Close();
}
}
```

// Fin del Botón Eliminar



```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
    xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
    x:Class="VENTAS_2023.WelcomePage"
    Title="Bienvenido Admin">
<ContentPage.Content>
    <ScrollView>
        <StackLayout BackgroundColor="LightGray">
            <Label Text="¡Bienvenido! Administrador"
                FontAttributes="Bold"
                FontSize="36"
                TextColor="Black"
                VerticalTextAlignment="Center"
                HorizontalTextAlignment="Center" />
            <StackLayout BackgroundColor="LightGray">
                <Image Source="https://advisertecnology.com/wp-content/uploads/2021/08/portadajpv.png" Aspect="Fill"/>
            </StackLayout>
            <StackLayout Orientation="Horizontal" HorizontalOptions="CenterAndExpand">
                <Button Text="Operaciones" TextColor="Black"
                    HorizontalOptions="Center"
                    BackgroundColor="GreenYellow"
                    WidthRequest="150"
                    HeightRequest="50"
                    BorderWidth="2"
                    BorderColor="LightYellow"
                    CornerRadius="25"
                    x:Name="btncrud"
                    Clicked="btncrud_Clicked"/>
                <Button Text="Salir" TextColor="White"
                    HorizontalOptions="Center"
                    BackgroundColor="Red"
                    WidthRequest="150"
                    HeightRequest="50"
                    BorderWidth="2"
                    BorderColor="LightYellow"
                    CornerRadius="25"
                    x:Name="Salir"
                    Clicked="Salir_Clicked"
                    />
            </StackLayout>
        </StackLayout>
    </ScrollView>
</ContentPage.Content>
</ContentPage>
```

Vamos a modificar el código de
WelcomePage.xaml

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

using Xamarin.Forms;
using Xamarin.Forms.Xaml;

namespace VENTAS_2023
{
    [XamlCompilation(XamlCompilationOptions.Compile)]
    public partial class WelcomePage : ContentPage
    {
        public WelcomePage()
        {
            InitializeComponent();
        }

        // Este método se ejecuta cuando se hace clic en un botón con el nombre "btncrud"
        private async void btncrud_Clicked(object sender, EventArgs e)
        {
            // Navega a una nueva página llamada "CRUD"
            await Navigation.PushAsync(new CRUD());
        }

        // Este método se ejecuta cuando se hace clic en un botón con el nombre "Salir"
        private void Salir_Clicked(object sender, EventArgs e)
        {
            // Salimos de la aplicación
            // Cerrar la aplicación
            System.Diagnostics.Process.GetCurrentProcess().Kill();
        }
    }
}
```

Vamos a modificar el código de
WelcomePage.xaml.cs

```

<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
    xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
    x:Class="VENTAS_2023.Supevisor"
    Title="Supervisor">
<ContentPage.Content>
    <!-- Se utiliza un ScrollView para que el contenido sea desplazable en caso de que no quepa en la pantalla -->
    <ScrollView>
        <!-- Contenedor principal de la página con un fondo gris claro -->
        <StackLayout BackgroundColor="LightGray">
            <!-- Título grande y centrado -->
            <Label Text="Bienvenido! Supervisor"
                FontAttributes="Bold"
                FontSize="36"
                TextColor="Black"
                VerticalTextAlignment="Center"
                HorizontalTextAlignment="Center" />
            <!-- Contenedor para la imagen del usuario -->
            <StackLayout BackgroundColor="LightGray">
                <Image Source="https://cdn.icon-icons.com/icons2/966/PNG/128/Users_icon-icons.com_74706.png" HeightRequest="250" Aspect="AspectFit"/>
            </StackLayout>
            <!-- Entradas de texto para el criterio de búsqueda, nombre de usuario, teléfono y correo -->
            <Entry Placeholder="Introduzca el Criterio a Buscar" x:Name="id_usuario" HorizontalTextAlignment="Center" TextColor="Black" BackgroundColor="Azure"/>
            <Entry Placeholder="Nombre de Usuario" x:Name="nombre_user" IsEnabled="False" HorizontalTextAlignment="Center" FontAttributes="Bold" TextColor="Black" BackgroundColor="Azure"/>
            <Entry Placeholder="Teléfono" x:Name="telefono" IsEnabled="False" HorizontalTextAlignment="Center" FontAttributes="Bold" TextColor="Black" BackgroundColor="Azure"/>
            <Entry Placeholder="Correo" x:Name="email" IsEnabled="False" HorizontalTextAlignment="Center" FontAttributes="Bold" TextColor="Black" BackgroundColor="Azure"/>
            <!-- Etiqueta para mostrar resultados de búsqueda -->
            <Label x:Name="resultadoBusqueda" Text="" />
            <!-- Botones para limpiar, validar conexión, mostrar datos y buscar -->
            <Button Text="Limpiar" Clicked="LimpiarButton_Clicked" BackgroundColor="Yellow" Margin="10"/>
            <Button Text="Validar Conexion" BackgroundColor="#2196F3" TextColor="White" Clicked="Button_Clicked"/>
            <!-- Grid para alinear los botones de "Mostrar Datos" y "Buscar" -->
            <Grid Grid.Row="2" RowDefinitions="Auto,Auto" ColumnDefinitions="*,*">
                <Button Grid.Row="0" Grid.Column="0" Text="Mostrar Datos" BackgroundColor="#1e3799" TextColor="White" x:Name="getbutton" Clicked="Getbutton_Clicked" />
                <Button Grid.Row="0" Grid.Column="1" Text="Buscar" BackgroundColor="#48dbfb" TextColor="White" x:Name="Buscar" Clicked="Buscar_Clicked" />
            </Grid>
            <!-- Botón para salir de la aplicación -->
            <StackLayout>
                <Button Text="Salir" TextColor="White"
                    HorizontalOptions="Center"
                    BackgroundColor="Red"
                    WidthRequest="150"
                    HeightRequest="50"
                    BorderWidth="2"
                    BorderColor="LightYellow"
                    CornerRadius="25"
                    x:Name="Salir"
                    Clicked="Salir_Clicked"/>
            </StackLayout>
            <!-- Lista de elementos mostrados en ListView -->
            <ListView Grid.Row="3" x:Name="MyListView" HasUnevenRows="True" Margin="1" BackgroundColor="LightGray">
                <ListView.ItemTemplate>
                    <DataTemplate>
                        <ViewCell>
                            <!-- Grid para mostrar los detalles de cada elemento en la lista -->
                            <Grid RowDefinitions="Auto" ColumnDefinitions="1, 3, 4, 6" Padding="1" Margin="2" BackgroundColor="Bisque">
                                <Frame Grid.Row="0" Grid.Column="0" BorderColor="Black" Padding="5">
                                    <Label Text="{Binding Id_usuario}" TextColor="Black" HorizontalTextAlignment="Center" VerticalTextAlignment="Center"></Label>
                                </Frame>
                                <Frame Grid.Row="0" Grid.Column="1" BorderColor="Black" Padding="5">
                                    <Label Text="{Binding Nombre_user}" TextColor="Black" HorizontalTextAlignment="Center" VerticalTextAlignment="Center"></Label>
                                </Frame>
                                <Frame Grid.Row="0" Grid.Column="2" BorderColor="Black" Padding="5">
                                    <Label Text="{Binding Telefono}" TextColor="Black" HorizontalTextAlignment="Center" VerticalTextAlignment="Center"></Label>
                                </Frame>
                                <Frame Grid.Row="0" Grid.Column="3" BorderColor="Black" Padding="5">
                                    <Label Text="{Binding Email}" TextColor="Black" HorizontalTextAlignment="Center" VerticalTextAlignment="Center"></Label>
                                </Frame>
                            </Grid>
                        </ViewCell>
                    </DataTemplate>
                </ListView.ItemTemplate>
            </ListView>
        </StackLayout>
    </ContentPage.Content>
</ContentPage>

```

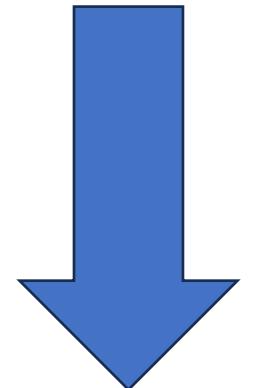
Vamos a modificar el código de supervisor.xaml

Vamos a modificar el código de
supervisor.xaml.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Data.SqlClient;
using Xamarin.Forms;
using Xamarin.Forms.Xaml;
using Xamarin.Essentials;

namespace VENTAS_2023
{
    [XamlCompilation(XamlCompilationOptions.Compile)]
    public partial class Supevisor : ContentPage
    {
        // Variable para controlar el estado del inicio de sesión
        bool isLoggedIn = false;

        // Clase que define la estructura de los elementos en la lista
        public class MyTableList
        {
            public int Id_usuario { get; set; }
            public string Nombre_user { get; set; }
            public string Telefono { get; set; }
            public string Email { get; set; }
        }
        // Objeto para manejar la conexión a la base de datos
        SqlConnection sqlConnection;
```



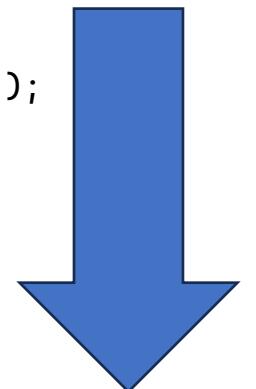
Vamos a modificar el código de
supervisor.xaml.cs

```
public Supervisor()
{
    InitializeComponent();

    // Configuración de la cadena de conexión a la base de datos
    string srvrdbname = "LOGIN_2023";
    string srvrname = "192.168.2.209";
    string srvrusername = "JUANCITO";
    string srvrpassword = "123456";
    string sqlconn = $"Data Source={srvrname};Initial Catalog={srvrdbname};User ID={srvrusername};Password={srvrpassword}";

    // Inicialización de la conexión a la base de datos
    sqlConnection = new SqlConnection(sqlconn);
}

// Método para validar la conexión a la base de datos
private async void Button_Clicked(object sender, EventArgs e)
{
    try
    {
        sqlConnection.Open(); // Abre la conexión
        await App.Current.MainPage.DisplayAlert("Alert", "Conexión Establecida Correctamente!✓", "Ok");
        sqlConnection.Close(); // Cierra la conexión
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex.Message);
        throw;
    }
}
```



```
// Método para obtener datos de la base de datos y mostrarlos en la ListView
private async void Getbutton_Clicked(object sender, EventArgs e)
{
    try
    {
        List<MyTableList> myTableLists = new List<MyTableList>(); // Lista para almacenar los datos obtenidos
        sqlConnection.Open(); // Abre la conexión

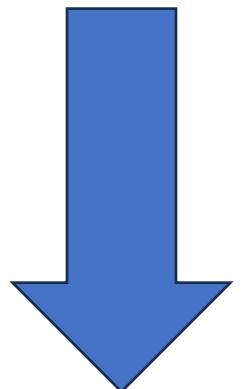
        string queryString = "Select * from dbo.usuario"; // Consulta SQL
        SqlCommand command = new SqlCommand(queryString, sqlConnection);
        SqlDataReader reader = command.ExecuteReader(); // Ejecuta la consulta

        while (reader.Read()) // Recorre las filas obtenidas
        {
            // Crea un objeto MyTableList y asigna sus propiedades
            myTableLists.Add(new MyTableList
            {
                Id_usuario = Convert.ToInt32(reader["id_usuario"]),
                Nombre_user = reader["nombre_user"].ToString(),
                Telefono = reader["telefono"].ToString(),
                Email = reader["email"].ToString(),
            });
        }

        reader.Close();
        sqlConnection.Close(); // Cierra la conexión

        MyListView.ItemsSource = myTableLists; // Asigna los datos a la ListView
    }
    catch (Exception ex)
    {
        await App.Current.MainPage.DisplayAlert("Alerta", ex.Message, "Ok");
        throw;
    }
}
```

Código de
supervisor.xaml.cs



Código de supervisor.xaml.cs

```
// Método para realizar una búsqueda en la base de datos
private void Buscar_Clicked(object sender, EventArgs e)
{
    // ... (implementación para buscar en la base de datos)
    if (string.IsNullOrWhiteSpace(id_usuario.Text))
    {
        resultadoBusqueda.Text = "Por favor, introduzca un criterio de búsqueda.";
        resultadoBusqueda.TextColor = Color.Red;
        resultadoBusqueda.HorizontalTextAlignment = TextAlignment.Center;
        resultadoBusqueda.FontAttributes = FontAttributes.Bold;
        resultadoBusqueda.FontSize = 22;
        return;
    }

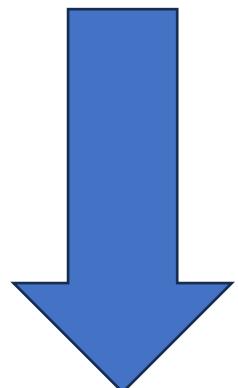
    string consultaSQL = "Select * from dbo.usuario WHERE 1 = 1";

    if (int.TryParse(id_usuario.Text, out int idUsuarioValue))
    {
        consultaSQL += " AND id_usuario = @valorBusqueda";
    }
    else
    {
        consultaSQL += " AND (nombre_user LIKE @valorBusqueda OR telefono = @valorBusqueda OR email = @valorBusqueda)";
    }

    SqlCommand comando = new SqlCommand(consultaSQL, sqlConnection);

    if (!int.TryParse(id_usuario.Text, out _))
    {
        comando.Parameters.AddWithValue("@valorBusqueda", "%" + id_usuario.Text + "%");
    }
    else
    {
        comando.Parameters.AddWithValue("@valorBusqueda", id_usuario.Text);
    }

    sqlConnection.Open();
```



Código de supervisor.xaml.cs

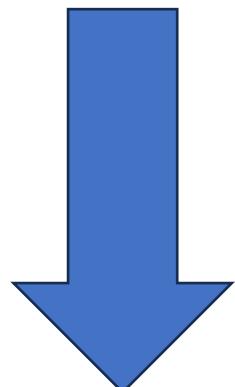
```
SqlDataReader registro = comando.ExecuteReader();

    if (registro.Read())
    {
        nombre_user.Text = registro["nombre_user"].ToString();
        telefono.Text = registro["telefono"].ToString();
        email.Text = registro["email"].ToString();
        resultadoBusqueda.Text = "Resultado de la búsqueda:";
    }
    else
    {
        resultadoBusqueda.Text = "No se encontraron resultados.";
        resultadoBusqueda.TextColor = Color.Red;
        resultadoBusqueda.HorizontalTextAlignment = TextAlignment.Center;
        resultadoBusqueda.FontAttributes = FontAttributes.Bold;
        resultadoBusqueda.FontSize = 22;
    }

    sqlConnection.Close();

}

// Método para limpiar los campos de búsqueda y resultados
private void LimpiarButton_Clicked(object sender, EventArgs e)
{
    // ... (implementación para limpiar campos)
    id_usuario.Text = string.Empty;
    nombre_user.Text = string.Empty;
    telefono.Text = string.Empty;
    email.Text = string.Empty;
    resultadoBusqueda.Text = string.Empty;
}
```



Código de supervisor.xaml.cs

```
// Método para manejar el botón de "Salir"
    private async void Salir_Clicked(object sender, EventArgs e)
    {

        if (!isUserLoggedIn)
        {
            var respuesta = await DisplayAlert("Confirmación", "¿Seguro que deseas cerrar sesión?", "Sí", "No");

            if (respuesta)
            {
                isUserLoggedIn = false;

                System.Diagnostics.Process.GetCurrentProcess().CloseMainWindow();
                await Navigation.PushAsync(new MainPage());
            }
        }
    }
}
```



Fin del Código de supervisor.xaml.cs

Vamos a Modificar el Código de vendedor.xaml

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
    xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
    x:Class="VENTAS_2023.Vendedor">
<ContentPage.Content>
    <!-- StackLayout que contiene la estructura principal de la página -->
    <StackLayout BackgroundColor="#f5f6fa">
        <!-- Etiqueta de bienvenida para el vendedor -->
        <Label Text="¡Bienvenido! Vendedor"
            FontAttributes="Bold"
            FontSize="36"
            TextColor="Black"
            VerticalTextAlignment="Center"
            HorizontalTextAlignment="Center" />
        <!-- Selector de artículos utilizando un Picker -->
        <Picker x:Name="ArticuloPicker" Title="Seleccionar Artículo" SelectedIndexChanged="ArticuloPicker_SelectedIndexChanged"
            FontSize="25"
            FontAttributes="Bold"
            HorizontalTextAlignment="Center"
            TextColor="Blue">
            <!-- Configuración de cómo se muestra cada elemento del Picker -->
            <Picker.ItemDisplayBinding>
                <Binding Path="Nombre" />
            </Picker.ItemDisplayBinding>
        </Picker>
        <!-- Etiqueta para mostrar el precio del artículo seleccionado -->
        <Label Text="Precio:" HorizontalOptions="Center" VerticalOptions="CenterAndExpand" FontAttributes="Bold" FontSize="22" />
        <!-- Etiqueta para mostrar el precio del artículo -->
        <Label x:Name="Preciolabel" HorizontalOptions="CenterAndExpand" VerticalOptions="CenterAndExpand" FontAttributes="Bold" FontSize="25" TextColor="#44bd32"/>
        <!-- Entrada de texto para la cantidad de artículos -->
        <Entry Placeholder="Cantidad" Keyboard="Numeric" x:Name="CantidadEntry" HorizontalTextAlignment="Center" FontAttributes="Bold" FontSize="25" TextColor="#44bd32"/>
        <!-- Botón para agregar una venta -->
        <Button Text="Agregar Venta" Clicked="AgregarVenta_Clicked"
            BackgroundColor="#44bd32" TextColor="White"
            BorderRadius="5" HeightRequest="40"
            HorizontalOptions="CenterAndExpand"
            VerticalOptions="CenterAndExpand" />
        <!-- Botón para listar las ventas -->
        <Button Text="Listar Ventas" Clicked="ListarVentas_Clicked"
            BackgroundColor="#3498db" TextColor="White"
            BorderRadius="5" HeightRequest="40"
            HorizontalOptions="CenterAndExpand"
            VerticalOptions="CenterAndExpand" />
        <!-- Etiqueta para encabezado de la lista de ventas -->
        <Label Text="Lista de Ventas: " FontSize="25" TextColor="DarkRed" FontAttributes="Bold" HorizontalOptions="CenterAndExpand" VerticalOptions="CenterAndExpand" />
        <!-- Etiqueta para mostrar el total de ventas -->
        <Label x:Name="TotalVentasLabel" FontAttributes="Bold" HorizontalOptions="CenterAndExpand" VerticalOptions="CenterAndExpand" />
        <!-- ListView para mostrar la lista de ventas -->
        <ListView x:Name="VentasListView">
            <ListView.ItemTemplate>
                <DataTemplate>
                    <ViewCell>
                        <!-- Layout horizontal para cada elemento de la lista -->
                        <StackLayout Orientation="Horizontal">
                            <Label Text="{Binding NombreArticulo}" HorizontalOptions="FillAndExpand" />
                            <Label Text="{Binding Cantidad}" HorizontalOptions="FillAndExpand" />
                            <Label Text="{Binding Precio}" HorizontalOptions="FillAndExpand" />
                            <Label Text="{Binding Monto}" HorizontalOptions="FillAndExpand" />
                        </StackLayout>
                    </ViewCell>
                </DataTemplate>
            </ListView.ItemTemplate>
        </ListView>
        <!-- Botón para imprimir un ticket -->
        <Button Text="Imprimir Ticket" Clicked="ImprimirTicket_Clicked"
            BackgroundColor="#2ecc71" TextColor="White"
            BorderRadius="5" HeightRequest="40"
            HorizontalOptions="CenterAndExpand"
            VerticalOptions="CenterAndExpand" />
    </StackLayout>
</ContentPage.Content>
</ContentPage>
```

```

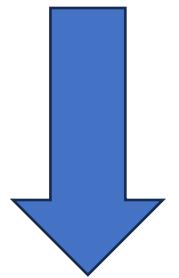
using System;
using System.Collections.ObjectModel;
using System.Data.SqlClient;
using System.Linq;
using Xamarin.Forms;
using Xamarin.Forms.Xaml;
using Xamarin.Essentials;
using System.Threading.Tasks;

namespace VENTAS_2023
{
    // Definición de la clase "Articulo"
    public class Articulo
    {
        public int IDArticulo { get; set; }          // Propiedad para el ID del artículo
        public string Nombre { get; set; }           // Propiedad para el nombre del artículo
        public decimal Precio { get; set; }          // Propiedad para el precio del artículo
    }

    // Definición de la clase "Venta"
    public class Venta
    {
        public int IDArticulo { get; set; }          // Propiedad para el ID del artículo
        public string NombreArticulo { get; set; }    // Propiedad para el nombre del artículo
        public int Cantidad { get; set; }             // Propiedad para la cantidad de artículos vendidos
        public decimal Precio { get; set; }           // Propiedad para el precio unitario del artículo
        public decimal Monto => Cantidad * Precio;   // Propiedad calculada para el monto total de la venta
    }
}

```

Vamos a Modificar el Código de
vendedor.xaml.cs



Vamos a Modificar el Código de vendedor.xaml.cs

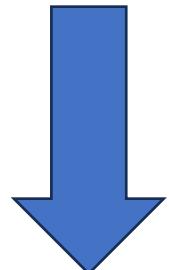
```
[XamlCompilation(XamlCompilationOptions.Compile)] // Atributo que indica la compilación de XAML
public partial class Vendedor : ContentPage
{
    private ObservableCollection<Articulo> articulos; // Colección para almacenar los artículos
    private ObservableCollection<Venta> ventas; // Colección para almacenar las ventas
    SqlConnection sqlConnection; // Objeto para manejar la conexión a la base de datos

    public Vendedor()
    {
        InitializeComponent(); // Inicializa la interfaz de usuario

        // Configuración de la cadena de conexión a la base de datos
        string srvrdbname = "LOGIN_2023";
        string srvrname = "192.168.2.209";
        string srvrusername = "JUANCITO";
        string srvrpassword = "123456";
        string sqlconn = $"Data Source={srvrname};Initial Catalog={srvrdbname};User ID={srvrusername};Password={srvrpassword}";
        sqlConnection = new SqlConnection(sqlconn); // Inicializa la conexión a la base de datos

        LoadArticulos(); // Cargar los artículos desde la base de datos

        LoadVentas(); // Cargar las ventas en la inicialización
        VentasListView.ItemsSource = ventas; // Cargar las ventas en el ListView
    }
}
```



```
// Método para cargar los artículos desde la base de datos y configurar el Picker de artículos
private void LoadArticulos()
{
    articulos = new ObservableCollection<Articulo>(); // Inicializa la colección de artículos

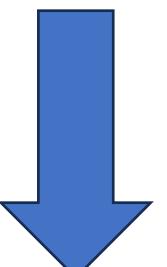
    try
    {
        sqlConnection.Open(); // Abre la conexión a la base de datos
        string queryString = "SELECT IDArticulo, Nombre, Precio FROM Articulos"; // Consulta SQL para obtener los artículos
        SqlCommand command = new SqlCommand(queryString, sqlConnection);
        SqlDataReader reader = command.ExecuteReader();

        while (reader.Read()) // Recorre los resultados de la consulta
        {
            // Crea objetos Articulo y agrega a la colección
            articulos.Add(new Articulo
            {
                IDArticulo = Convert.ToInt32(reader["IDArticulo"]),
                Nombre = reader["Nombre"].ToString(),
                Precio = Convert.ToDecimal(reader["Precio"])
            });
        }

        reader.Close(); // Cierra el lector
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex.Message); // Muestra cualquier error en la consola
        throw;
    }
    finally
    {
        sqlConnection.Close(); // Cierra la conexión a la base de datos en cualquier caso
    }
}

ArticuloPicker.ItemsSource = articulos; // Asigna los artículos al Picker para su selección
}
```

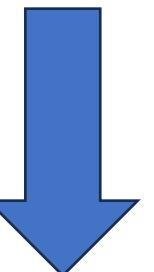
Vamos a Modificar el
Código de
vendedor.xaml.cs



Vamos a Modificar el
Código de
vendedor.xaml.cs

```
// Método que se ejecuta cuando se selecciona un artículo en el Picker
private void ArticuloPicker_SelectedIndexChanged(object sender, EventArgs e)
{
    var selectedArticulo = (Articulo)ArticuloPicker.SelectedItem; // Obtiene el artículo
    seleccionado del Picker

    if (selectedArticulo != null)
    {
        PrecioLabel.Text = selectedArticulo.Precio.ToString("C"); // Muestra el precio del
        artículo seleccionado en formato monetario
    }
    else
    {
        PrecioLabel.Text = string.Empty; // Si no se selecciona ningún artículo, se muestra un
        campo vacío
    }
}
```



```

// Método que se ejecuta cuando se hace clic en el botón de "Aregar Venta"
private async void AgregarVenta_Clicked(object sender, EventArgs e)
{
    if (ArticuloPicker.SelectedItem == null || string.IsNullOrWhiteSpace(CantidadEntry.Text))
        return; // Si no se selecciona un artículo o no se ingresa una cantidad, no hace nada

    var selectedArticulo = (Articulo)ArticuloPicker.SelectedItem; // Obtiene el artículo seleccionado
    int cantidad = Convert.ToInt32(CantidadEntry.Text); // Convierte la cantidad ingresada a un valor entero

    // Crea un nuevo objeto Venta con los detalles de la venta
    var nuevaVenta = new Venta
    {
        IDArticulo = selectedArticulo.IDArticulo,
        NombreArticulo = selectedArticulo.Nombre,
        Cantidad = cantidad,
        Precio = selectedArticulo.Precio
    };

    try
    {
        sqlConnection.Open(); // Abre la conexión a la base de datos

        // Consulta SQL para insertar la nueva venta en la tabla Ventas
        string insertQuery = "INSERT INTO Ventas (IDArticulo, Cantidad, Precio) VALUES (@IDArticulo, @Cantidad, @Precio)";
        SqlCommand insertCommand = new SqlCommand(insertQuery, sqlConnection);
        insertCommand.Parameters.AddWithValue("@IDArticulo", nuevaVenta.IDArticulo);
        insertCommand.Parameters.AddWithValue("@Cantidad", nuevaVenta.Cantidad);
        insertCommand.Parameters.AddWithValue("@Precio", nuevaVenta.Precio);
        insertCommand.ExecuteNonQuery(); // Ejecuta la consulta para insertar la venta

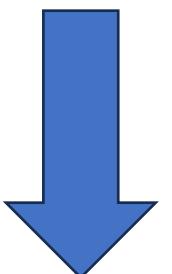
        sqlConnection.Close(); // Cierra la conexión a la base de datos
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex.Message); // Muestra cualquier error en la consola
        throw;
    }

    LoadVentas(); // Vuelve a cargar las ventas desde la base de datos
    ResetVentaInputs(); // Limpia los campos de entrada de la venta

    await DisplayAlert("Venta Agregada", "La venta ha sido agregada exitosamente.", "OK"); // Muestra una alerta de éxito
}

```

Vamos a Modificar el
Código de
vendedor.xaml.cs



```

// Método para cargar las ventas desde la base de datos y actualizar la lista y el total de ventas
private void LoadVentas()
{
    ventas = new ObservableCollection<Venta>(); // Inicializa la colección de ventas

    try
    {
        sqlConnection.Open(); // Abre la conexión a la base de datos
        string queryString = "SELECT IDArticulo, Cantidad, Precio FROM Ventas"; // Consulta SQL para obtener las ventas
        SqlCommand command = new SqlCommand(queryString, sqlConnection);
        SqlDataReader reader = command.ExecuteReader();

        decimal totalVentas = 0; // Variable para almacenar el total de ventas

        while (reader.Read()) // Recorre los resultados de la consulta
        {
            int idArticulo = Convert.ToInt32(reader["IDArticulo"]);
            Articulo articulo = articulos.FirstOrDefault(a => a.IDArticulo == idArticulo); // Busca el artículo correspondiente

            // Agrega una nueva venta a la colección
            ventas.Add(new Venta
            {
                IDArticulo = idArticulo,
                NombreArticulo = articulo != null ? articulo.Nombre : "Artículo Desconocido",
                Cantidad = Convert.ToInt32(reader["Cantidad"]),
                Precio = Convert.ToDecimal(reader["Precio"])
            });

            // Suma el monto de la venta al total de ventas.
            totalVentas += ventas.Last().Monto;
        }

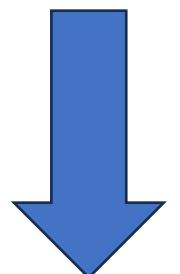
        // Configura el Label para mostrar el total de ventas
        TotalVentasLabel.Text = $"Total de Ventas: {totalVentas.ToString("C")}";
        TotalVentasLabel.FontSize = 18;
        TotalVentasLabel.FontAttributes = FontAttributes.Bold;
        TotalVentasLabel.TextColor = Color.Green;
        TotalVentasLabel.HorizontalTextAlignment = TextAlignment.Center;
        TotalVentasLabel.VerticalTextAlignment = TextAlignment.Center;

        reader.Close(); // Cierra el lector
    }
    catch (Exception ex)
    {
        Console.WriteLine(ex.Message); // Muestra cualquier error en la consola
        throw;
    }
    finally
    {
        sqlConnection.Close(); // Cierra la conexión a la base de datos en cualquier caso
    }
}

VentasListView.ItemsSource = ventas; // Actualiza la ListView con las ventas cargadas
}

```

Vamos a Modificar el
Código de
vendedor.xaml.cs



```

// Método para restablecer los campos de entrada de venta
private void ResetVentaInputs()
{
    ArticuloPicker.SelectedItem = null; // Limpia la selección de artículo en el Picker
    PrecioLabel.Text = string.Empty; // Limpia el contenido del Label de precio
    CantidadEntry.Text = string.Empty; // Limpia el contenido del Entry de cantidad
}

// Método que se ejecuta cuando se hace clic en el botón de "Listar Ventas"
private void ListarVentas_Clicked(object sender, EventArgs e)
{
    LoadVentas(); // Carga nuevamente las ventas desde la base de datos
    VentasListView.ItemsSource = ventas; // Actualiza la ListView con las ventas cargadas
}

// Método que se ejecuta cuando se hace clic en el botón de "Imprimir Ticket"
private async void ImprimirTicket_Clicked(object sender, EventArgs e)
{
    string contenidoTicket = "Ticket de Ventas\n\n"; // Encabezado del ticket
    decimal montoTotal = 0; // Variable para almacenar el monto total de ventas

    // Recorre las ventas y agrega sus detalles al contenido del ticket
    foreach (Venta venta in ventas)
    {
        contenidoTicket += $"Artículo: {venta.NombreArticulo}\n" +
                           $"Cantidad: {venta.Cantidad}\n" +
                           $"Precio Unitario: {venta.Precio:C}\n" +
                           $"Monto Total: {venta.Monto:C}\n\n";

        montoTotal += venta.Monto; // Suma el monto de cada venta al monto total
    }

    contenidoTicket += $"Monto Total de Ventas: {montoTotal:C}\n"; // Agrega el monto total al ticket

    // Muestra una ventana emergente con el contenido del ticket
    await DisplayAlert("Listado de Venta Actual", contenidoTicket, "Aceptar");
}

// Cierre de la clase y del namespace
}

```



Vamos a Modificar el
Código de
vendedor.xaml.cs

The screenshot shows a developer's environment with several windows open:

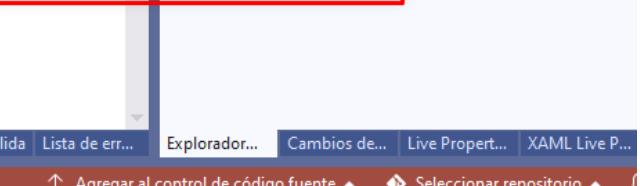
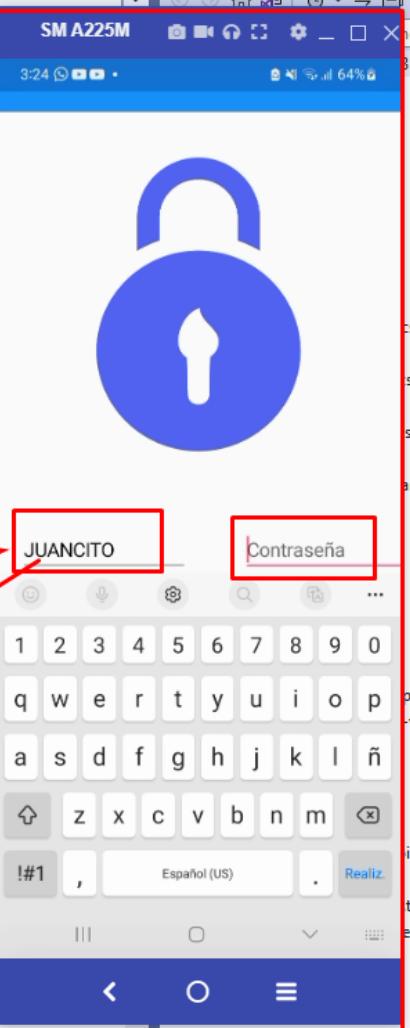
- Code Editor:** Displays C# code for a `MainPage.xaml.cs` file. A red box highlights the connection string line:

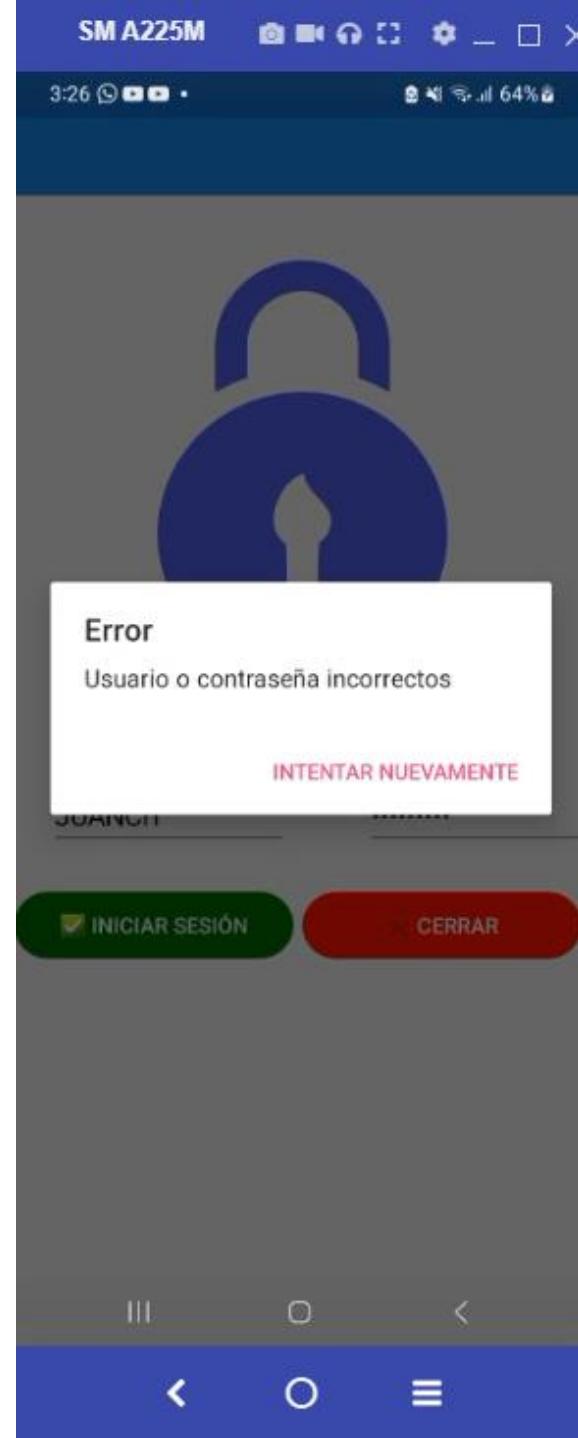
```
string connectionString = "Server=192.168.2.209,1433;Database=LOGIN_2023;User Id=JUANCITO;Password=123456;";
```
- Command Prompt:** Shows network interface information. A red box highlights the IP address line: `IPv4 Address : 192.168.2.209`.
- SQL Server Management Studio (SSMS):** An open database connection to `LOGIN_2023`. The **Object Explorer** shows the database structure. The **Results** pane displays data from the `rol` table:

	<code>id_rol</code>	<code>nombre_rol</code>
1	1	administrador
2	2	supervisor
3	3	vendedor

Results pane displays data from the `usuario` table:

	<code>id_usuario</code>	<code>nombre_usuario</code>	<code>password</code>	<code>id_rol</code>
1	1	JUANCITO	ADMIN@123	1
2	2	DARIEL	DARIEL@123	2
3	3	DANIELA	DANIELA@123	3
4	4	MARIA	MARIA@123	1
5	5	YENNEFER	YENNEFER@123	2





VENTAS_2023.MainPage

txtUsuario

File Edit View Query Project Tools Window Help

LOGIN SECCION_PDM_XAMARIN.sql - JUANCITO\MSSQLSERVERJPV.LOGIN_2023 (JUANCITO (60))* - Microsoft...

Quick Launch (Ctrl+Q)

Object Explorer

Connect JUANCITO\MSSQLSERVERJPV

- Databases
- Security
- Server Objects
- Replication
- PolyBase
- Always On High Availability
- Management
- Integration Services Catalog
- SQL Server Agent
- XEvent Profiler

LOGIN_2023

Execute

Properties

Results Messages

	id_rol	nombre_rol
1	1	administrador
2	2	supervisor
3	3	vendedor

	id_usuario	nombre_usuario	password	id_ml
1	1	JUANCITO	ADMIN@123	1
2	2	DARIEL	DARIEL@123	2
3	3	DANIELA	DANIELA@123	3
4	4	MARIA	MARIA@123	1
5	5	YENNEFER	YENNEFER@123	2

Query executed successfully. | JUANCITO\MSSQLSERVERJPV (15... | JUANCITO (60) | LOGIN_2023 | 00:00:00 | 8 rows

Ready Ln 48 Col 27 Ch 27 INS

Connection-specific DNS Suffix . : example.org

SM A225M

3:27 64%

JUANCITO

INICIAR SESIÓN CERRAR





```

string password = txtPassword.Text;

```

LOGIN SECCION_PDM_XAMARIN.sql - JUANCITO\MSSQLSERVERJPV.LOGIN_2023 (JUANCITO (60))* - Microsoft... Quick Launch (Ctrl+Q)

File Edit View Query Project Tools Window Help

New Query MDX DMX XMLA DAX

LOGIN_2023 Execute

Object Explorer Connect JUANCITO\MSSQLSERVERJPV

Databases Security Server Objects Replication PolyBase Always On High Availability Management Integration Services Catalog SQL Server Agent XEvent Profiler

LOGIN SECCION_PDM_XAMARIN.sql (JUANCITO (60))*

46 select * from usuario

47

129 %

Results Messages

	id_usuario	nombre_user	telefono	email
1	1	PEDRO JOSE	809-789-0000	pedro@EMAIL.COM
2	2	Daniel Adolfo	809-507-4322	DARIEL@rmsil.com
3	3	Maria Vizcaino	809-777-8000	maria@EMAIL.COM
4	7	PEDRO JOSE	809-789-0000	pedro@EMAIL.COM
5	5	MIGUEL GUERA	809-789-7896	MIGUEL@EMAIL.COM
6	9	HOLA NUEVO REGISTRO	555-88888	NUEVOEMAIL.COM

Query executed successfully. JUANCITO\MSSQLSERVERJPV (15... JUANCITO (60) LOGIN_2023 00:00:00 6 rows

Ready Ln 46 Col 1 Ch 1 INS Valor Connection-specific DNS Suffix . : example.org



VENIAS_2023.CRUD

await App.Current.MainPage.DisplayAlert("Alerta", "¡Felicitaciones, el registro se ha eliminado correctamente!")

LOGIN SECCION_PDM_XAMARIN.sql - JUANCITO\MSSQLSERVERJPV.LOGIN_2023 (JUANCITO (60)) - Microsoft...

File Edit View Query Project Tools Window Help

New Query MDX DAX XMLA DAX

LOGIN_2023 Execute

Object Explorer

Connect JUANCITO\MSSQLSERVERJPV

- Databases
- Security
- Server Objects
- Replication
- PolyBase
- Always On High Availability
- Management
- Integration Services Catalog
- SQL Server Agent
- XEvent Profiler

LOGIN SECCION_PDM_XAMARIN (JUANCITO (60))*

46 select * from usuario

47

129 %

Results Messages

	id_usuario	nombre_user	telefono	email
1	1	PEDRO JOSE	809-789-0000	pedro@EMAIL.COM
2	2	Daniel Adolfo	809-507-4322	DARIEL@msil.com
3	3	Maria Vizcaino	809-777-8000	maria@EMAIL.COM
4	7	PEDRO JOSE	809-789-0000	pedro@EMAIL.COM
5	5	MIGUEL GUERA	809-789-7896	MIGUEL@EMAIL.COM
6	9	HOLA NUEVO REGISTRO	555-88888	NUEVOEMAIL.COM

Query executed successfully. | JUANCITO\MSSQLSERVERJPV (15...) | JUANCITO (60) | LOGIN_2023 | 00:00:00 | 6 rows

Ready Ln 6 Col 2 INS Valor Tipo

SM A225M 3:33 64%

Operaciones de CRUD

CRUD SQL SERVER Y XAMARIN

Microsoft SQL Server 2012

1

PEDRO JOSE

809-789-0000

pedro@EMAIL.COM

VALIDAR CONEXION

Mostrar Datos

BUSCAR

INSERTAR

ACTUALIZAR

BORRAR

```
await App.Current.MainPage.DisplayAlert("Alerta", "¡Felicitaciones, el registro se ha eliminado correctamente!",
```

LOGIN SECCION_PDM_XAMARIN.sql - JUANCITO\MSSQLSERVERJPV.LOGIN_2023 (JUANCITO (60))* - Microsoft...

File Edit View Query Project Tools Window Help

LOGIN_2023 New Query Execute

Object Explorer

Connect JUANCITO\MSSQLSERVERJPV

- Databases
- Security
- Server Objects
- Replication
- PolyBase
- Always On High Availability
- Management
- Integration Services Catalog
- SQL Server Agent
- XEvent Profiler

LOGIN SECCION_PDM_3 (JUANCITO (60))*

46 select * from usuario

47

129 %

Results Messages

	id_usuario	nombre_user	telefono	email
1	1	PEDRO JOSE	809-789-0000	pedro@EMAIL.COM
2	2	Daniel Adolfo	809-507-4322	DARIEL@mmail.com
3	3	Maria Vizcaino	809-777-8000	maria@EMAIL.COM
4	7	PEDRO JOSE	809-789-0000	pedro@EMAIL.COM
5	5	MIGUEL GUERA	809-789-7896	MIGUEL@EMAIL.COM
6	9	HOLA NUEVO REGISTRO	555-88888	NUEVOEMAIL.COM

Query executed successfully. JUANCITO\MSSQLSERVERJPV (15... | JUANCITO (60) | LOGIN_2023 | 00:00:00 | 6 rows

Ready Ln 6 Col 2 INS Valor Tipo

SM A225M 3:34 64% ← Operaciones de CRUD

CRUD SQL SERVER Y XAMARIN

Microsoft SQL Server 2012

10 ADVISERTECNOLOGY 849-000-0001 ADVISERTETELOGY@EMAIL.COM

VALIDAR CONEXION	Mostrar Datos
BUSCAR	<input checked="" type="checkbox"/> INSERTAR
ACTUALIZAR	BORRAR

```
await App.Current.MainPage.DisplayAlert("Alerta", "¡Felicitaciones, el registro se ha eliminado correctamente!")
```

LOGIN SECCION_PDM_XAMARIN.sql - JUANCITO\MSSQLSERVERJPV.LOGIN_2023 (JUANCITO (60)) - Microsoft... Quick Launch (Ctrl+Q)

File Edit View Query Project Tools Window Help

LOGIN_2023 Execute

Object Explorer

Connect JUANCITO\MSSQLSERVERJPV

- Databases
- Security
- Server Objects
- Replication
- PolyBase
- Always On High Availability
- Management
- Integration Services Catalog
- SQL Server Agent
- XEvent Profiler

LOGIN SECCION_PDM_3 (JUANCITO (60))
select * from usuario

Results Messages

	id_usuario	nombre_user	telefono	email
1	1	PEDRO JOSE	809-789-0000	pedro@EMAIL.COM
2	2	Daniel Adolfo	809-507-4322	DARIEL@rmsil.com
3	3	Maria Vizcaino	809-777-8000	maria@EMAIL.COM
4	7	PEDRO JOSE	809-789-0000	pedro@EMAIL.COM
5	5	MIGUEL GUERA	809-789-7896	MIGUEL@EMAIL.COM
6	10	ADVISERTECNOLOGY	849-000-0001	ADVISERTETELOGY@EMAIL.COM
7	9	HOLA NUEVO REGISTRO	555-88888	NUEVOEMAIL.COM

Query executed successfully. JUANCITO\MSSQLSERVERJPV (15... JUANCITO (60) LOGIN_2023 00:00:00 7 rows

Ready Ln 6 Col 1 INS Valor

SM A225M 3:35 64%

Operaciones de CRUD

10

ADVISERTECNOLOGY 849-000-0001 ADVISERTETELOGY@EMAIL.COM

VALIDAR CONEXION MOSTRAR DATOS

INSERTAR

ACTUALIZAR BORRAR

1	PEDRO JOSE	809-789-0000	pedro@EMAIL.COM
2	Daniel Adolfo	809-507-4322	DARIEL@rmsil.com
3	Maria Vizcaino	809-777-8000	maria@EMAIL.COM
7	PEDRO JOSE	809-789-0000	pedro@EMAIL.COM
5	MIGUEL GUERA	809-789-7896	MIGUEL@EMAIL.COM
10	ADVISERTECNOLOGY	849-000-0001	ADVISERTETELOGY@EMAIL.COM

await App.Current.MainPage.DisplayAlert("Alerta", "¡Felicitaciones, el registro se ha eliminado correctamente!")

LOGIN SECCION_PDM_XAMARIN.sql - JUANCITO\MSSQLSERVERJPV.LOGIN_2023 (JUANCITO (60)) - Microsoft...

File Edit View Query Project Tools Window Help

LOGIN_2023

Object Explorer

Connect JUANCITO\MSSQLSERVERJPV

- Databases
- Security
- Server Objects
- Replication
- PolyBase
- Always On High Availability
- Management
- Integration Services Catalog
- SQL Server Agent
- XEvent Profiler

LOGIN SECCION_PDM_XAMARIN.sql (JUANCITO (60))

46 select * from usuario

47

129 %

Results Messages

	id_usuario	nombre_user	telefono	email
1	1	PEDRO JOSE	809-789-0000	pedro@EMAIL.COM
2	2	Daniel Adolfo	809-507-4322	DARIEL@rmsil.com
3	3	Maria Vizcaino	809-777-8000	maria@EMAIL.COM
4	7	PEDRO JOSE	809-789-0000	pedro@EMAIL.COM
5	5	MIGUEL GUERA	809-789-7896	MIGUEL@EMAIL.COM
6	10	ADVISERTECNOLOGY	849-000-0001	ADVISERTETELOGY@EMAIL.COM
7	9	HOLA NUEVO REGISTRO	555-888888	NUEVOEMAIL.COM

Query executed successfully. JUANCITO\MSSQLSERVERJPV (15... JUANCITO (60) LOGIN_2023 00:00:00 7 rows

Ready Ln 6 Col 1 INS Valor Tipo

SM A225M 3:36 64% Operaciones de CRUD 10 ADVISERTECNOLOGY 849-000-0001 ADVISERTETELOGY@EMAIL.COM Alerta ¡Felicitaciones, el registro se ha eliminado correctamente! OK

JOSE 809-789-0000 pedro@EMAIL.COM

Daniel Adolfo 809-507-4322 DARIEL@rmsil.com

Maria Vizcaino 809-777-8000 maria@EMAIL.COM

PEDRO JOSE 809-789-0000 pedro@EMAIL.COM

MIGUEL GUERA 809-789-7896 MIGUEL@EMAIL.COM

ADVISERTECNOLOGY 849-000-0001 ADVISERTETELOGY@EMAIL.COM

```
await App.Current.MainPage.DisplayAlert("Alerta", "¡Felicitaciones, el registro se ha eliminado correctamente!", null)
```

LOGIN SECCION_PDM_XAMARIN.sql - JUANCITO\MSSQLSERVERJPV.LOGIN_2023 (JUANCITO (60)) - Microsoft... Quick Launch (Ctrl+Q)

File Edit View Query Project Tools Window Help

LOGIN_2023

New Query MDX DAX XMLA DAX

Object Explorer

Connect JUANCITO\MSSQLSERVERJPV

- Databases
- Security
- Server Objects
- Replication
- PolyBase
- Always On High Availability
- Management
- Integration Services Catalog
- SQL Server Agent
- XEvent Profiler

LOGIN SECCION_PDM...3 (JUANCITO (60))

46 select * from usuario

47

Results Messages

	id_usuario	nombre_user	telefono	email
1	1	PEDRO JOSE	809-789-0000	pedro@EMAIL.COM
2	2	Daniel Adolfo	809-507-4322	DARIEL@mmsil.com
3	3	Maria Vizcaino	809-777-8000	maria@EMAIL.COM
4	7	PEDRO JOSE	809-789-0000	pedro@EMAIL.COM
5	5	MIGUEL GUERA	809-789-7896	MIGUEL@EMAIL.COM
6	9	HOLA NUEVO REGISTRO	555-88888	NUEVOEMAIL.COM

Query executed successfully. JUANCITO\MSSQLSERVERJPV (15... | JUANCITO (60) | LOGIN_2023 | 00:00:00 | 6 rows

Ready Ln 45 Col 1 Ch 1 INS

SM A225M 3:37 64%

Operaciones de CRUD

10

ADVISERTECNOLOGY

849-000-0001

ADVISERTECNOLOGY@EMAIL.COM

VALIDAR CONEXION MOSTRAR DATOS

BUSCAR INSERTAR

ACTUALIZAR BORRAR

1	PEDRO JOSE	809-789-0000	pedro@EMAIL.COM
2	Daniel Adolfo	809-507-4322	DARIEL@mmsil.com
3	Maria Vizcaino	809-777-8000	maria@EMAIL.COM
7	PEDRO JOSE	809-789-0000	pedro@EMAIL.COM
5	MIGUEL GUERA	809-789-7896	MIGUEL@EMAIL.COM
9	HOLA NUEVO REGISTRO	555-88888	NUEVOEMAIL.COM

VENTAS 2022 CRUD

Quick Launch (Ctrl+Q)

File Edit View Query Project Tools Window Help

New Query MDX DMX XMLA DAX

LOGIN_SECCION_PDM_XAMARIN.sql - JUANCITO\MSSQLSERVERJPV.LOGIN_2023 (JUANCITO (60))* - Microsoft...

LOGIN_2023

Execute

Object Explorer

Connect JUANCITO\MSSQLSERVERJPV

- Databases
- Security
- Server Objects
- Replication
- PolyBase
- Always On High Availability
- Management
- Integration Services Catalog
- SQL Server Agent
- XEvent Profiler

LOGIN_SECCION_PDM_XAMARIN (JUANCITO (60))*

46 select * from usuario

47

129 %

Results Messages

	id_usuario	nombre_user	telefono	email
1	1	PEDRO JOSE	809-789-0000	pedro@EMAIL.COM
2	2	Dariel Adolfo	809-507-4322	DARIEL@msil.com
3	3	Maria Vizcaino	809-777-8000	maria@EMAIL.COM
4	7	PEDRO JOSE	809-789-0000	pedro@EMAIL.COM
5	5	MIGUEL GUERRA	809-789-7896	MIGUEL@EMAIL.COM
6	9	HOLA NUEVO REGISTRO	555-88888	NUEVOEMAIL.COM

Properties

Query executed successfully.

JUANCITO\MSSQLSERVERJPV (15... | JUANCITO (60) | LOGIN_2023 | 00:00:00 | 6 rows

Ln 45 Col 1 Ch 1 INS

Profundidad de búsqueda: Nombre

Valor Tipo

Operaciones de CRUD

CRUD SQL SERVER Y XAMARIN

Microsoft SQL Server 2012

9

HOLA NUEVO

555-88888

NUEVOEMAIL.COM

VALIDAR CONEXION

BUSCAR

ACTUALIZAR

MOSTRAR DATOS

INSERTAR

BORRAR

Rectamente!

SM A225M

3:41 65%

← Operaciones de CRUD

CRUD SQL SERVER Y XAMARIN

Microsoft SQL Server 2012

9

HOLA NUEVO

555-88888

NUEVOEMAIL.COM

VALIDAR CONEXION

BUSCAR

ACTUALIZAR

MOSTRAR DATOS

INSERTAR

BORRAR

VENTAS_2022 CRUD

Quick Launch (Ctrl+Q)

File Edit View Query Project Tools Window Help

New Query MDX DAX XHLA DAY

LOGIN_2023 Execute

LOGIN SECCION_PDM_XAMARIN.sql - JUANCITO\MSSQLSERVERJPV.LOGIN_2023 (JUANCITO (60)) - Microsoft...

Object Explorer

Connect

JUANCITO\MSSQLSERVERJPV

- Databases
- Security
- Server Objects
- Replication
- PolyBase
- Always On High Availability
- Management
- Integration Services Catalog
- SQL Server Agent
- XEvent Profiler

LOGIN SECCION_PDM_XAMARIN.sql (JUANCITO (60))

46 select * from usuario

47

129 %

Results Messages

	id_usuario	nombre_user	telefono	email
1	1	PEDRO JOSE	809-789-0000	pedro@EMAIL.COM
2	2	Daniel Adolfo	809-507-4322	DARIEL@msil.com
3	3	Maria Vizcaino	809-777-8000	maria@EMAIL.COM
4	7	PEDRO JOSE	809-789-0000	pedro@EMAIL.COM
5	5	MIGUEL GUERA	809-789-7896	MIGUEL@EMAIL.COM
6	9	HOLA NUEVO	555-88888	NUEVOEMAIL.COM

Query executed successfully. | JUANCITO\MSSQLSERVERJPV (15... | JUANCITO (60) | LOGIN_2023 | 00:00:00 | 6 rows

Ready Ln 45 Col 1 Ch 1 INS

Profundidad de búsqueda: Nombre

Valor Tipo

Properties

Rectamente!

SM A225M

3:42 65%

Operaciones de CRUD

CRUD SQL SERVER Y XAMARIN

Microsoft SQL Server 2012

Alerta

¡Felicitaciones, el registro se ha actualizado correctamente!

OK

VALIDAR CONEXION

MOSTRAR DATOS

BUSCAR

ACTUALIZAR

INSERTAR

BORRAR

3 c

CRUD.xaml.cs X WelcomePage.xaml.cs Supervisor.xaml MainPage.xaml.cs WelcomePage.xaml Supervisor.xaml.cs Vendedor.xaml

VENTAS 2022 CRUD

LOGIN SECCION_PDM_XAMARIN.sql - JUANCITO\MSSQLSERVERJPV.LOGIN_2023 (JUANCITO (60)) - Microsoft...

Quick Launch (Ctrl+Q)

File Edit View Query Project Tools Window Help

New Query MDX DAX XMLA DAX

LOGIN_2023 Execute

Object Explorer

Connect JUANCITO\MSSQLSERVERJPV

- Databases
- Security
- Server Objects
- Replication
- PolyBase
- Always On High Availability
- Management
- Integration Services Catalog
- SQL Server Agent
- XEvent Profiler

LOGIN SECCION_PDM_XAMARIN.sql (JUANCITO (60))

```
40 --Ver tabla y registro roles:  
41  
42 select * from roles;  
43  
44 --Ver tabla y registros usuarios:  
45  
46 select * from usuarios
```

129 %

Results Messages

	id_rol	nombre_rol
1	1	administrador
2	2	supervisor
3	3	vendedor

	id_usuario	nombre_usuario	password	id_rol
1	1	JUANCITO	ADMIN@123	1
2	2	DARIEL	DARIEL@123	2
3	3	DANIELA	DANIELA@123	3
4	4	MARIA	MARIA@123	1
5	5	YENNEFER	YENNEFER@123	2

Query executed successfully. JUANCITO\MSSQLSERVERJPV (15... JUANCITO (60) LOGIN_2023 00:00:00 8 rows

Ready Ln 41 Col 1 Ch 1 INS

Profundidad de búsqueda: Nombre

SM A225M

3:43 65%

DARIEL

INICIAR SESIÓN





VENTAS_2023 CRUD

LOGIN SECCION_PDM_XAMARIN.sql - JUANCITO\MSSQLSERVERJPV.LOGIN_2023 (JUANCITO (60))* - Microsoft...

File Edit View Query Project Tools Window Help

New Query MDX DML XMLA DAX

LOGIN_2023 Execute

Object Explorer

Connect JUANCITO\MSSQLSERVERJPV

- Databases
- Security
- Server Objects
- Replication
- PolyBase
- Always On High Availability
- Management
- Integration Services Catalog
- SQL Server Agent
- XEvent Profiler

LOGIN SECCION_PDM_XAMARIN.sql (JUANCITO (60))*

```
--Ver tabla y registro roles:  
40  
41  
42 select * from roles;  
43  
44 --Ver tabla y registros usuarios:  
45  
46 select * from usuarios
```

129 %

Results Messages

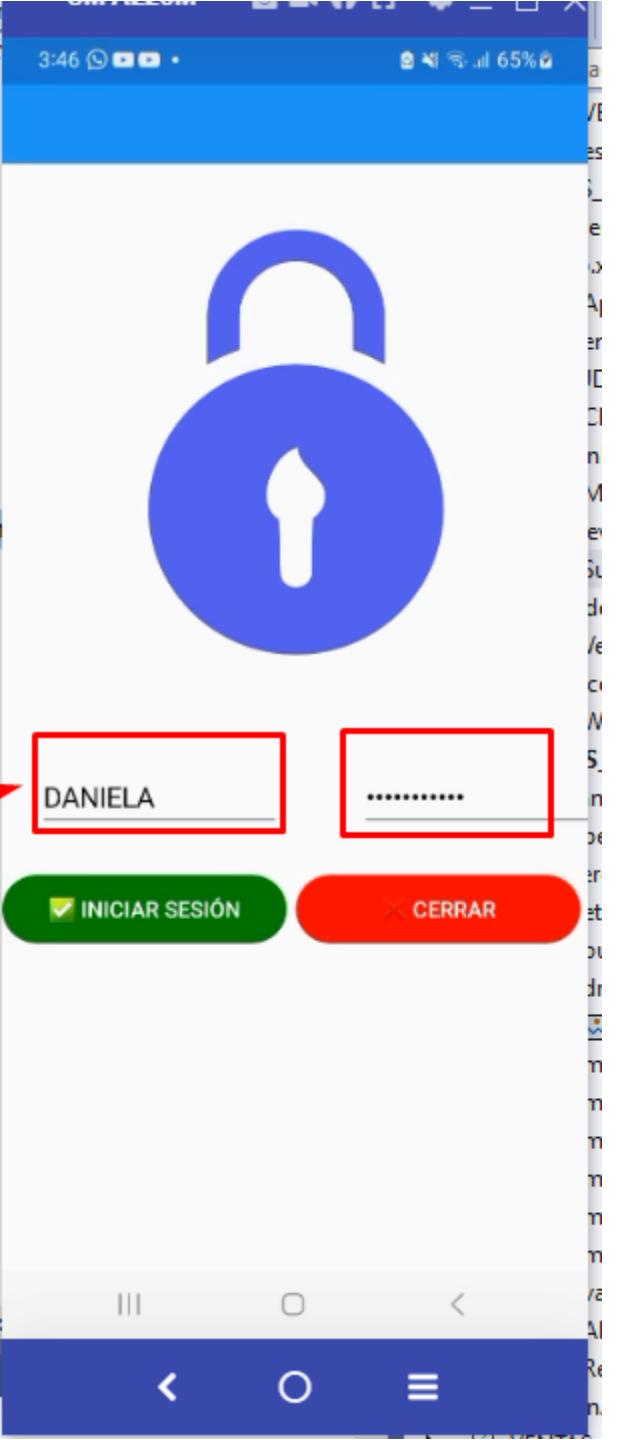
	id_rol	nombre_rol
1	1	administrador
2	2	supervisor
3	3	vendedor

	id_usuario	nombre_usuario	password	id_rol
1	1	JUANCITO	ADMIN@123	1
2	2	DARIEL	DARIEL@123	2
3	3	DANIELA	DANIELA@123	3
4	4	MARIA	MARIA@123	1
5	5	YENNEFER	YENNEFER@123	2

Query executed successfully. JUANCITO\MSSQLSERVERJPV (15... JUANCITO (60) LOGIN_2023 00:00:00 8 rows

Ready Ln 41 Col 1 Ch 1 INS

Profundidad de búsqueda: Nombre



File Edit View Query Project Tools Window Help

LOGIN SECCION_PDM_XAMARIN.sql - JUANCITO\MSSQLSERVERUPV.LOGIN_2023 (JUANCITO (60))* - Microsoft...

Quick Launch (Ctrl+Q)

LOGIN SECCION_PDM_XAMARIN.sql - JUANCITO\MSSQLSERVERUPV.LOGIN_2023 (JUANCITO (60))* - Microsoft...

Object Explorer

Connect JUANCITO\MSSQLSERVERUPV

- Databases
- Security
- Server Objects
- Replication
- PolyBase
- Always On High Availability
- Management
- Integration Services Catalog
- SQL Server Agent
- XEvent Profiler

112 -- Consulta de la tabla Articulos

113

114 SELECT * FROM Articulos;

115

116 -- Consulta de la tabla Ventas

117

118 SELECT * FROM Ventas;

129 %

Results

IDArticulo	Nombre	Precio
1	Laptop Legion	850.50
2	Mouse Gamer	30.25
3	Monitor Gamer 32	250.75
4	Bocinas	12.90
5	UPS de 1200 kw	120.00
6	Smartphone	500.00
7	Laptop	1000.00
8	Tablet	300.00
9	Smartwatch	150.00
10	Gafas de Realidad Virtual	200.00

Messages

IDVenta	IDArticulo	Cantidad	Precio

Query executed successfully. JUANCITO\MSSQLSERVERUPV (15... JUANCITO (60) LOGIN_2023 00:00:00 15 rows

Ready Ln 113 Col 1 Ch 1 INS

Nombre

¡Bienvenido! Vendedor

Seleccionar Artículo

Precio:

Cantidad

AGREGAR VENTA

LISTAR VENTAS

Lista de Ventas:

Total de Ventas: \$0.00

IMPRIMIR TICKET

CRUD.xaml.cs X WelcomePage.xaml.cs Supervisor.xaml MainPage.xaml.cs WelcomePage.xaml Supervisor.xaml.cs Vendedor.xaml

VENTAS_2023_CRUD

LOGIN SECCION_PDM_XAMARIN.sql - JUANCITO\MSSQLSERVERJPV.LOGIN_2023 (JUANCITO (60))* - Microsoft... Quick Launch (Ctrl+Q)

File Edit View Query Project Tools Window Help

LOGIN_2023

New Query MDX DAX XMLA DAX Execute

Object Explorer Connect JUANCITO\MSSQLSERVERJPV Databases Security Server Objects Replication PolyBase Always On High Availability Management Integration Services Catalog SQL Server Agent XEvent Profiler

LOGIN SECCION_PDM_XAMARIN.sql (JUANCITO (60))*

```
112 -- Consulta de la tabla Articulos
113
114 SELECT * FROM Articulos;
115
116 -- Consulta de la tabla Ventas
117
118 SELECT * FROM Ventas;
```

129 %

Results Messages

IDArticulo	Nombre	Precio
1	Laptop Legion	850.50
2	Mouse Gamer	30.25
3	Monitor Gamer 32	250.75
4	Bocinas	12.90
5	UPS de 1200 kw	120.00
6	Smartphone	500.00
7	Laptop	1000.00
8	Tablet	300.00
9	Smartwatch	150.00
10	Gafas de Realidad Virtual	200.00

Query executed successfully. JUANCITO\MSSQLSERVERJPV (15... JUANCITO (60) LOGIN_2023 00:00:00 15 rows

Ready Ln 113 Col 1 Ch 1 INS

Nombre

SM A225M

3:49 65%

Seleccionar Artículo

Laptop Legion
Mouse Gamer
Monitor Gamer 32
Bocinas
UPS de 1200 kw
Smartphone
Laptop
Tablet
Smartwatch
Gafas de Realidad Virtual
Auriculares Inalámbricos
Altavoz Inteligente
Cámara de Acción
Monitor Curvo

CANCELAR

CRUD.xaml.cs X WelcomePage.xaml.cs Supervisor.xaml MainPage.xaml.cs WelcomePage.xaml Supervisor.xaml Vendedor.xaml

VENTAS_2023_CRUD

LOGIN SECCION_PDM_XAMARIN.sql - JUANCITO\MSSQLSERVERJPV.LOGIN_2023 (JUANCITO (60))* - Microsoft... Quick Launch (Ctrl+Q)

File Edit View Query Project Tools Window Help

LOGIN_2023

Object Explorer

Connect JUANCITO\MSSQLSERVERJPV Databases Security Server Objects Replication PolyBase Always On High Availability Management Integration Services Catalog SQL Server Agent XEvent Profiler

112 -- Consulta de la tabla Articulos
113
114 SELECT * FROM Articulos;
115
116 -- Consulta de la tabla Ventas
117
118 SELECT * FROM Ventas;

129 %

Results Messages

IDArticulo	Nombre	Precio
1	Laptop Legion	850.50
2	Mouse Gamer	30.25
3	Monitor Gamer 32	250.75
4	Bocinas	12.90
5	UPS de 1200 kw	120.00
6	Smartphone	500.00
7	Laptop	1000.00
8	Tablet	300.00
9	Smartwatch	150.00
10	Gafas de Realidad Virtual	200.00

Query executed successfully. JUANCITO\MSSQLSERVERJPV (15... | JUANCITO (60) | LOGIN_2023 | 00:00:00 | 15 rows

Ready Ln 113 Col 1 Ch 1 INS

Profundidad de búsqueda: Nombre

SM A225M 3:49 65%

¡Bienvenido! Vendedor

Laptop Legion

Precio: \$850.50

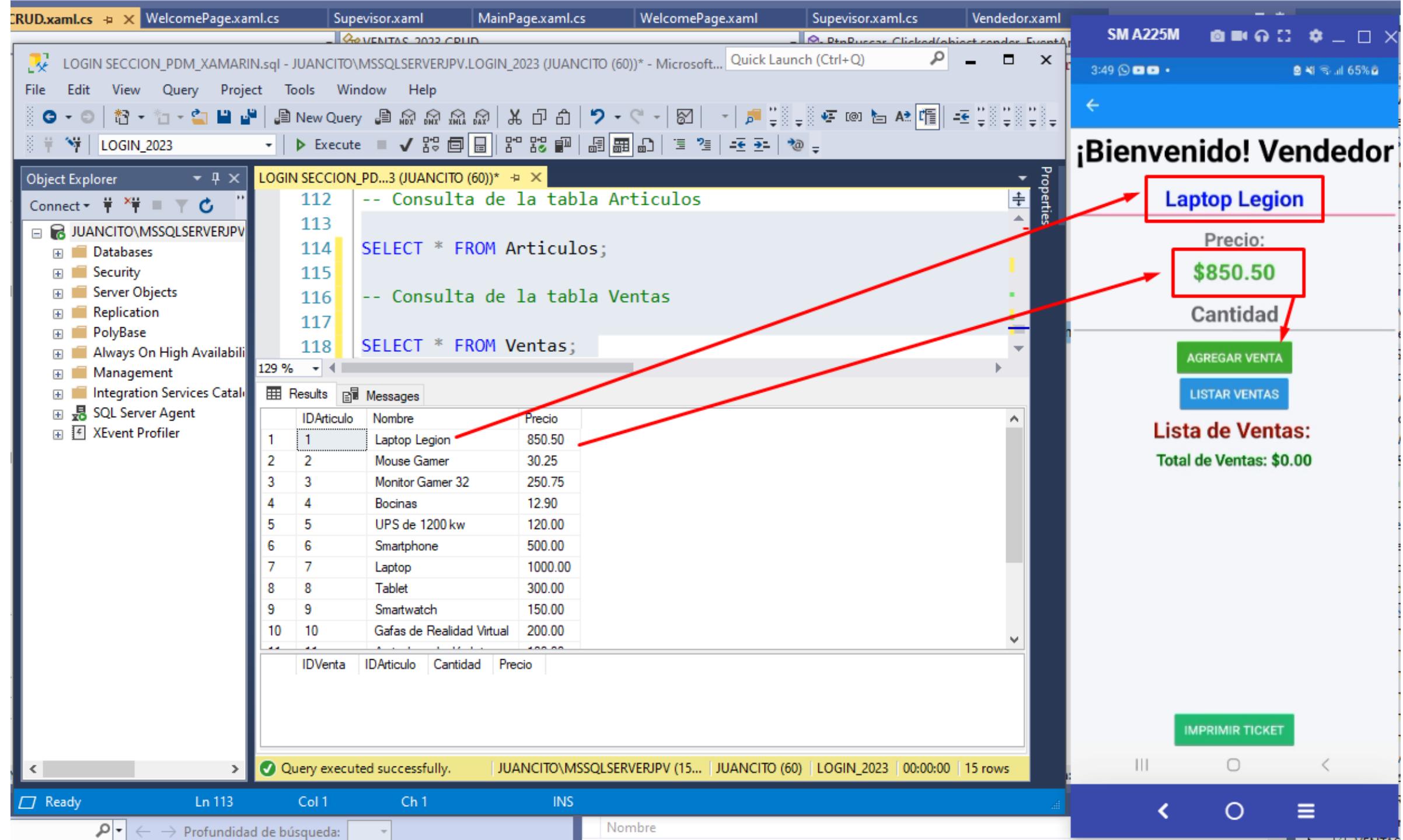
Cantidad

AGREGAR VENTA

LISTAR VENTAS

Lista de Ventas:
Total de Ventas: \$0.00

IMPRIMIR TICKET



VENTAS_2022 CRUD

LOGIN SECCION_PDM_XAMARIN.sql - JUANCITO\MSSQLSERVERJPV.LOGIN_2023 (JUANCITO (60))* - Microsoft...

Quick Launch (Ctrl+Q)

File Edit View Query Project Tools Window Help

New Query MDX DMX XMLA DAX

LOGIN_2023 Execute

Object Explorer

Connect JUANCITO\MSSQLSERVERJPV

- Databases
- Security
- Server Objects
- Replication
- PolyBase
- Always On High Availability
- Management
- Integration Services Catalog
- SQL Server Agent
- XEvent Profiler

LOGIN SECCION_PDM_XAMARIN.sql (JUANCITO (60))*

```
112 -- Consulta de la tabla Articulos
113
114 SELECT * FROM Articulos;
115
116 -- Consulta de la tabla Ventas
117
118 SELECT * FROM Ventas;
```

Results

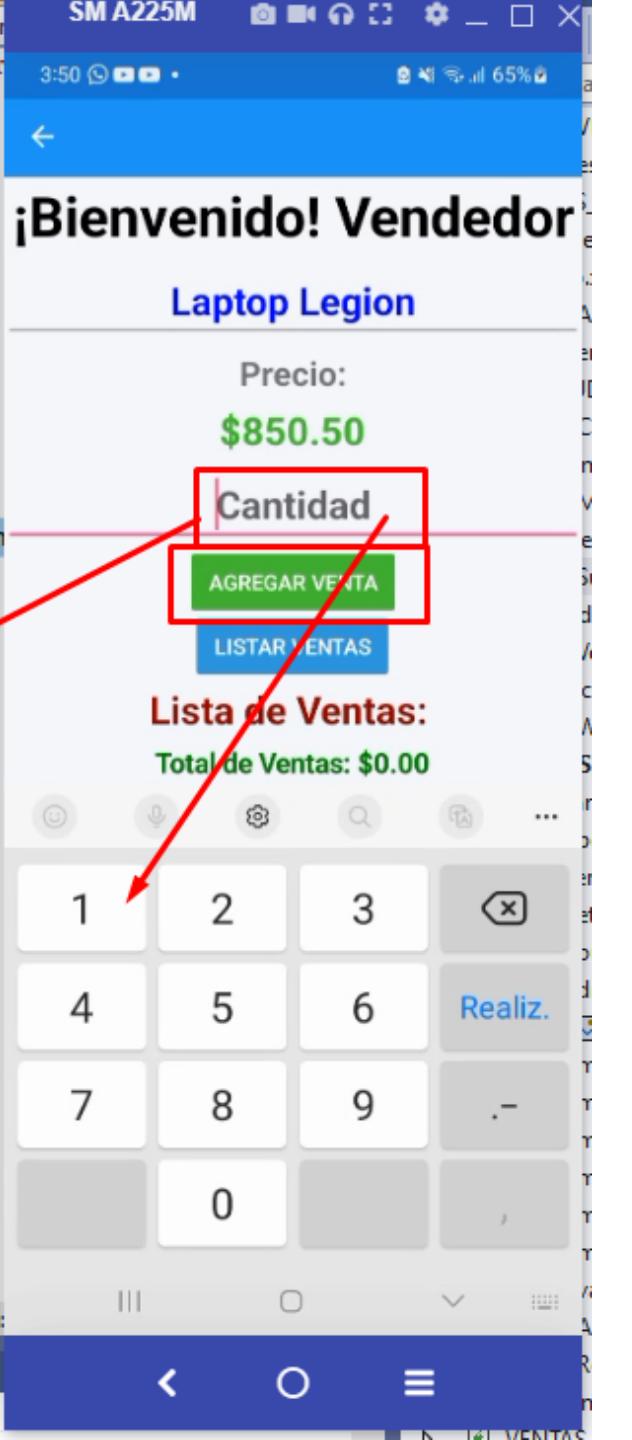
IDArticulo	Nombre	Precio
1	Laptop Legion	850.50
2	Mouse Gamer	30.25
3	Monitor Gamer 32	250.75
4	Bocinas	12.90
5	UPS de 1200 kw	120.00
6	Smartphone	500.00
7	Laptop	1000.00
8	Tablet	300.00
9	Smartwatch	150.00
10	Gafas de Realidad Virtual	200.00

IDVenta	IDArticulo	Cantidad	Precio

Query executed successfully. | JUANCITO\MSSQLSERVERJPV (15... | JUANCITO (60) | LOGIN_2023 | 00:00:00 | 15 rows

Ready Ln 113 Col 1 Ch 1 INS

Profundidad de búsqueda: Nombre



VENTAS_2023 CRUD

Quick Launch (Ctrl+Q)

File Edit View Query Project Tools Window Help

New Query MDX DMX XMLA DAX

LOGIN_2023 Execute

Object Explorer

Connect JUANCITO\MSSQLSERVERJPV

- Databases
- Security
- Server Objects
- Replication
- PolyBase
- Always On High Availability
- Management
- Integration Services Catalog
- SQL Server Agent
- XEvent Profiler

LOGIN SECCION_PDM_XAMARIN.sql - JUANCITO\MSSQLSERVERJPV.LOGIN_2023 (JUANCITO (60))* - Microsoft...

112 -- Consulta de la tabla Articulos

113

114 SELECT * FROM Articulos;

115

116 -- Consulta de la tabla Ventas

117

118 SELECT * FROM Ventas;

129 %

Results Messages

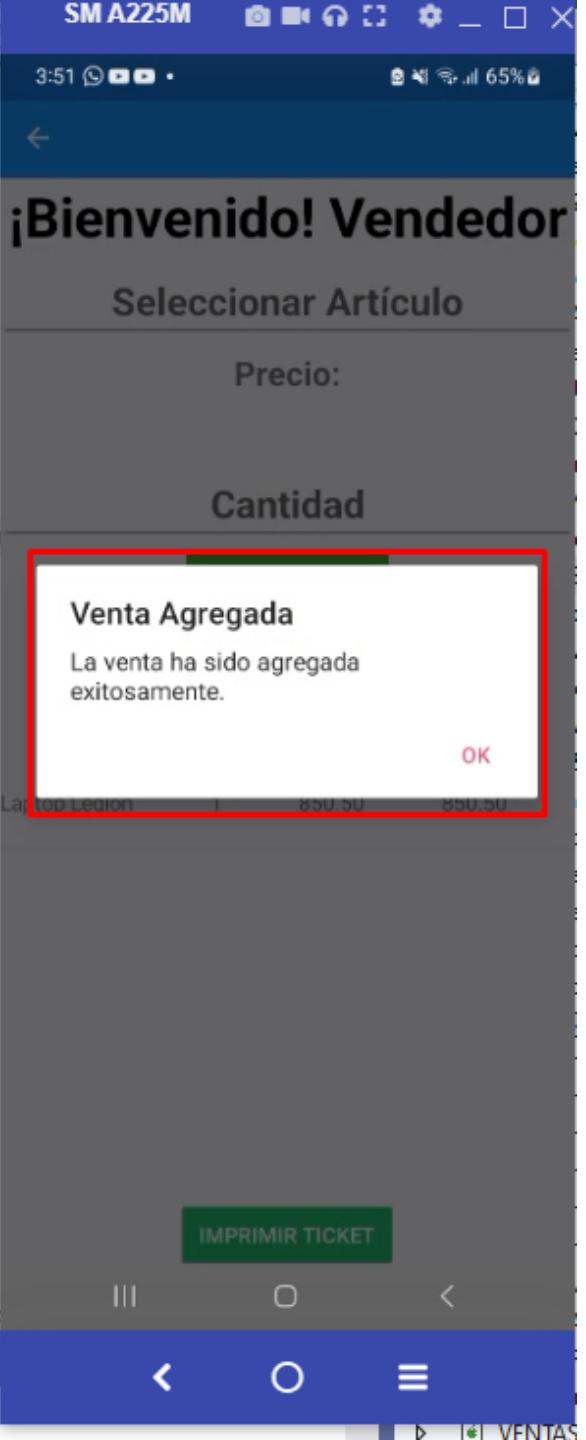
IDArticulo	Nombre	Precio
1	Laptop Legion	850.50
2	Mouse Gamer	30.25
3	Monitor Gamer 32	250.75
4	Bocinas	12.90
5	UPS de 1200 kw	120.00
6	Smartphone	500.00
7	Laptop	1000.00
8	Tablet	300.00
9	Smartwatch	150.00
10	Gafas de Realidad Virtual	200.00
		1000.00

IDVenta	IDArticulo	Cantidad	Precio

Query executed successfully. | JUANCITO\MSSQLSERVERJPV (15... | JUANCITO (60) | LOGIN_2023 | 00:00:00 | 15 rows

Ready Ln 113 Col 1 Ch 1 INS

Profundidad de búsqueda: Nombre



CRUD.xaml.cs WelcomePage.xaml.cs Supervisor.xaml MainPage.xaml.cs WelcomePage.xaml Supervisor.xaml Vendedor.xaml

VENTAS 2023 CRUD

LOGIN SECCION_PDM_XAMARIN.sql - JUANCITO\MSSQLSERVERJPV.LOGIN_2023 (JUANCITO (60))* - Microsoft...

File Edit View Query Project Tools Window Help

New Query MDX DMX XMLA DAX

LOGIN_2023 Execute

Object Explorer Connect JUANCITO\MSSQLSERVERJPV Databases Security Server Objects Replication PolyBase Always On High Availability Management Integration Services Catalog SQL Server Agent XEvent Profiler

LOGIN SECCION_PDM_3 (JUANCITO (60))*

```
112 -- Consulta de la tabla Articulos
113
114 SELECT * FROM Articulos;
115
116 -- Consulta de la tabla Ventas
117
118 SELECT * FROM Ventas;
```

129 %

Results Messages

	IDArticulo	Nombre	Precio
1	1	Laptop Legion	850.50
2	2	Mouse Gamer	30.25
3	3	Monitor Gamer 32	250.75
4	4	Bocinas	12.90
5	5	UPS de 1200 kw	120.00
6	6	Smartphone	500.00
7	7	Laptop	1000.00
8	8	Tablet	300.00
9	9	Smartwatch	150.00
10	10	Gafas de Realidad Virtual	200.00

	IDVenta	IDArticulo	Cantidad	Precio
1	20	1	1	850.50

Query executed successfully. JUANCITO\MSSQLSERVERJPV (15... JUANCITO (60) LOGIN_2023 00:00:00 16 rows

Ready Ln 114 Col 1 Ch 1 INS

Profundidad de búsqueda: Nombre

SM A225M 3:52 65%

CRUD.xaml.cs X WelcomePage.xaml.cs Supervisor.xaml MainPage.xaml.cs WelcomePage.xaml Supervisor.xaml.cs Vendedor.xaml

VENTAS_2023_CRUD

LOGIN SECCION_PDM_XAMARIN.sql - JUANCITO\MSSQLSERVERJPV.LOGIN_2023 (JUANCITO (60))* - Microsoft... Quick Launch (Ctrl+Q)

File Edit View Query Project Tools Window Help

New Query MDX DNX XMLA DAX

LOGIN_2023 Execute

Object Explorer

Connect JUANCITO\MSSQLSERVERJPV

- Databases
- Security
- Server Objects
- Replication
- PolyBase
- Always On High Availability
- Management
- Integration Services Catalog
- SQL Server Agent
- XEvent Profiler

112 -- Consulta de la tabla Articulos

113

114 SELECT * FROM Articulos;

115

116 -- Consulta de la tabla Ventas

117

118 SELECT * FROM Ventas;

129 %

Results Messages

	IDArticulo	Nombre	Precio
1	1	Laptop Legion	850.50
2	2	Mouse Gamer	30.25
3	3	Monitor Gamer 32	250.75
4	4	Bocinas	12.90
5	5	UPS de 1200 kw	120.00
6	6	Smartphone	500.00
7	7	Laptop	1000.00
8	8	Tablet	300.00
9	9	Smartwatch	150.00
10	10	Gafas de Realidad Virtual	200.00

	IDVenta	IDArticulo	Cantidad	Precio
1	20	1	1	850.50
2	21	5	5	120.00

Query executed successfully. JUANCITO\MSSQLSERVERJPV (15... | JUANCITO (60) | LOGIN_2023 | 00:00:00 | 17 rows

Ready Ln 114 Col 1 Ch 1 INS

RTbPasar_Clicked/objeto sender EventA

SM A225M 4:18 68%

¡Bienvenido! Vendedor

Seleccionar Artículo

Precio:

Cantidad

AGREGAR VENTA

LISTAR VENTAS

Total de Ventas: \$1,450.50

IMPRIMIR TICKET

Nombre

VENTAS_2022 CRUD

LOGIN SECCION_PDM_XAMARIN.sql - JUANCITO\MSSQLSERVERJPV.LOGIN_2023 (JUANCITO (60))* - Microsoft...

File Edit View Query Project Tools Window Help

New Query NDX DMX XMLA DAX

LOGIN_2023 Execute

Object Explorer

Connect JUANCITO\MSSQLSERVERJPV

- Databases
- Security
- Server Objects
- Replication
- PolyBase
- Always On High Availability
- Management
- Integration Services Catalog
- SQL Server Agent
- XEvent Profiler

LOGIN SECCION_PDM_XAMARIN.sql (JUANCITO (60))*

```

112 -- Consulta de la tabla Articulos
113
114 SELECT * FROM Articulos;
115
116 -- Consulta de la tabla Ventas
117
118 SELECT * FROM Ventas;

```

129 %

Results

IDArticulo	Nombre	Precio
1	Laptop Legion	850.50
2	Mouse Gamer	30.25
3	Monitor Gamer 32	250.75
4	Bocinas	12.90
5	UPS de 1200 kw	120.00
6	Smartphone	500.00
7	Laptop	1000.00
8	Tablet	300.00
9	Smartwatch	150.00
10	Gafas de Realidad Virtual	200.00

IDVenta	IDArticulo	Cantidad	Precio
1	1	1	850.50
2	5	5	120.00

Query executed successfully. | JUANCITO\MSSQLSERVERJPV (15... | JUANCITO (60) | LOGIN_2023 | 00:00:00 | 17 rows

Ready Ln 114 Col 1 Ch 1 INS

Profundidad de búsqueda: Nombre



Conclusión:

Este proyecto nos ha sumergido en la creación de una aplicación de ventas utilizando Xamarin, C# y SQL Server. A través de estas herramientas, hemos aprovechado una serie de beneficios:

Xamarin, la Versatilidad Móvil: Hemos desarrollado una única aplicación para iOS y Android, ahorrando tiempo y esfuerzo.

C#, Elegante y Potente: C# ha demostrado ser un lenguaje hábil para crear estructuras coherentes, como clases y métodos, impulsando la lógica de la aplicación.

SQL Server, Gestión de Datos Ágil: La base de datos SQL Server nos ha permitido almacenar y acceder a datos esenciales de manera ordenada y eficaz con 5 tablas como Roles, Usuario Administrativos, Usuarios Normales, Articulos y Ventas.

Métodos, Clases y Eventos, Fundamentos Lógicos: Hemos usado métodos y clases para diseñar una lógica sólida, mientras que los eventos han mejorado la experiencia del usuario.

Variables y Tablas, Organización y Almacenamiento: Variables temporales y tablas de SQL Server han sido esenciales para manipular y acceder a datos de manera efectiva con 5 tablas como Roles, Usuario Administrativos, Usuarios Normales, Articulos y Ventas.

Este proyecto nos ha equipado con habilidades clave en el desarrollo de aplicaciones móviles. Desde la creación de interfaces amigables hasta la gestión de datos y la implementación de lógica, hemos avanzado en nuestra travesía hacia el dominio del desarrollo móvil. ¡Sigamos explorando y creando en este emocionante viaje!

**Hacer esta
aplicacion, y con lo
que han aprendido
a nivel de la Interfaz
vamos Aplicarlo
aqui nuevamente y
con codigo xaml y
C# darle
funcionalidad.**

Te toca a ti.



**Puedes seguir agregando
mas tablas, por ejemplo
hacer un CRUD Para los
articulos, que sea solo el
administrador quien pueda
hacer, la tabla de clientes,
etc, y hacer que al hacer una
factura puedes se agregue
seleccionar al cliente, y a la
hora de hacer un reporte
puedas ver estos datos.**



GRACIAS

Bethbella