

# Sistema de venta

## Ciencias de la Computación V

Carlos Kevin Gamarra Cajavilca - Frank Charly Arias Ingaruca

Universidad Nacional Mayor de San Marcos  
Facultad de Ciencias Matemáticas  
Escuela Profesional de Computación Científica

29 de Octubre de 2022

# Contenido

- 1 Introducción
- 2 Uso de C#
- 3 Uso de SQL
- 4 Diagrama del sistema
- 5 Uso de clases
  - Uso de CapaDatos
  - Uso de CapaEntidad
  - Uso de CapaNegocio
  - Uso de CapaPresentación
- 6 Implementación del proyecto

- 1 Introducción
- 2 Uso de C#
- 3 Uso de SQL
- 4 Diagrama del sistema
- 5 Uso de clases
  - Uso de CapaDatos
  - Uso de CapaEntidad
  - Uso de CapaNegocio
  - Uso de CapaPresentación
- 6 Implementación del proyecto

En el Perú, la informalidad cuenta con un alto porcentaje con respecto a los negocios pequeños y de mayor demanda, como lo son las bodegas que se encuentran casi en cada calle.

Un aporte a estos negocios, es un sistema de registro de sus productos. Este proyecto se crea para solucionar esa necesidad de miles de personas que buscan una herramienta fácil y sencilla de utilizar.

# Contenido

- 1 Introducción
- 2 **Uso de C#**
- 3 Uso de SQL
- 4 Diagrama del sistema
- 5 Uso de clases
  - Uso de CapaDatos
  - Uso de CapaEntidad
  - Uso de CapaNegocio
  - Uso de CapaPresentación
- 6 Implementación del proyecto

Para la codificación de este sistema se usó C# porque nos ayuda en :

- Creación de clases y subclases.
- Permite relacionar los arreglos de las clases
- Podemos importarla a una base de datos y así guardar los cambios realizados.
- Permite crear una interfaz vistosa a manera de facilitar las operaciones del cliente.

# Contenido

- 1 Introducción
- 2 Uso de C#
- 3 Uso de SQL**
- 4 Diagrama del sistema
- 5 Uso de clases
  - Uso de CapaDatos
  - Uso de CapaEntidad
  - Uso de CapaNegocio
  - Uso de CapaPresentación
- 6 Implementación del proyecto

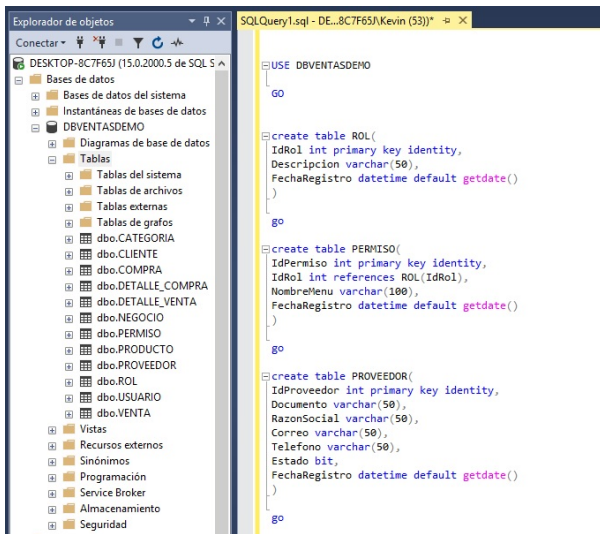
Usaremos SQL para crear una base de datos de nuestra tienda, en ella registraremos:

- Usuarios
- Cuentas
- Ventas
- Compras
- Creación de procedimientos
- Procesos de venta y compra
- Procesos de registros



# Uso de SQL

## Creación de las tablas ROL, PERMISO y PROVEEDOR.



## Creación de las tablas CLIENTE, USUARIO y CATEGORÍA.

```
SQLQuery1.sql - DE...8C7F65F\Kevin (53))* - X
create table CLIENTE(
    IdCliente int primary key identity,
    Documento varchar(50),
    NombreCompleto varchar(50),
    Correo varchar(50),
    Telefono varchar(50),
    Estado bit,
    FechaRegistro datetime default getdate()
)
go

create table USUARIO(
    IdUsuario int primary key identity,
    Documento varchar(50),
    NombreCompleto varchar(50),
    Correo varchar(50),
    Clave varchar(50),
    IdRol int references ROL(IdRol),
    Estado bit,
    FechaRegistro datetime default getdate()
)
go

create table CATEGORIA(
    IdCategoria int primary key identity,
    Descripcion varchar(100),
    Estado bit,
    FechaRegistro datetime default getdate()
)
go

create table PRODUCTO(
    IdProducto int primary key identity,
   Codigo varchar(50),
   Nombre varchar(50).
```

## Creación de las tablas PRODUCTO, COMPRA y DETALLE\_COMPRA

```
SQLQuery1.sql - DE...8C7F65\Kevin (53)]*  X
create table PRODUCTO(
    IdProducto int primary key identity,
    Codigo varchar(50),
    Nombre varchar(50),
    Descripcion varchar(50),
    IdCategoria int references CATEGORIA(IdCategoria),
    Stock int not null default 0,
    PrecioCompra decimal(10,2) default 0,
    PrecioVenta decimal(10,2) default 0,
    Estado bit,
    FechaRegistro datetime default getdate()
)
go

create table COMPRA(
    IdCompra int primary key identity,
    IdUsuario int references USUARIO(IdUsuario),
    IdProveedor int references PROVEEDOR(IdProveedor),
    TipoDocumento varchar(50),
    NumeroDocumento varchar(50),
    MontoTotal decimal(10,2),
    FechaRegistro datetime default getdate()
)
go

create table DETALLE_COMPRA(
    IdDetalleCompra int primary key identity,
    IdCompra int references COMPRA(IdCompra),
    IdProducto int references PRODUCTO(IdProducto),
    PrecioCompra decimal(10,2) default 0,
    PrecioVenta decimal(10,2) default 0,
    Cantidad int,
    MontoTotal decimal(10,2),
    FechaRegistro datetime default getdate()
)
```

## Creación de las tablas VENTA, DETALLE\_VENTA y NEGOCIO.

```
SQLQuery1.sql - DE...8C7F65\Kevin (53))* X
create table VENTA(
    IdVenta int primary key identity,
    IdUsuario int references USUARIO(IdUsuario),
    TipoDocumento varchar(50),
    NumeroDocumento varchar(50),
    DocumentoCliente varchar(50),
    NombreCliente varchar(100),
    MontoPago decimal(10,2),
    MontoCambio decimal(10,2),
    MontoTotal decimal(10,2),
    FechaRegistro datetime default getdate()
)
go

create table DETALLE_VENTA(
    IdDetalleVenta int primary key identity,
    IdVenta int references VENTA(IdVenta),
    IdProducto int references PRODUCTO(IdProducto),
    PrecioVenta decimal(10,2),
    Cantidad int,
    SubTotal decimal(10,2),
    FechaRegistro datetime default getdate()
)
go

create table NEGOCIO(
    IdNegocio int primary key,
    Nombre varchar(60),
    RUC varchar(60),
    Direccion varchar(60),
    Logo varbinary(max) NULL
)
go
```

## Creación de los procedimientos almacenados.

```
SQLQuery1.sql - DE...8C7F65\Kevin (53)) * X
/****** CREACION DE PROCEDIMIENTOS ALMACENADOS *****/
/*-----*/

go

create PROC SP_REGISTRARUSUARIO(
@Documento varchar(50),
@NombreCompleto varchar(100),
@Correo varchar(100),
@Clave varchar(100),
@IdRol int,
@Estado bit,
@IdUsuarioResultado int output,
@Mensaje varchar(500) output
)
as
begin
    set @IdUsuarioResultado = 0
    set @Mensaje = ''

    if not exists(select * from USUARIO where Documento = @Documento)
    begin
        insert into usuario(Documento,NombreCompleto,Correo,Clave,IdRol,Estado) values
        (@Documento,@NombreCompleto,@Correo,@Clave,@IdRol,@Estado)

        set @IdUsuarioResultado = SCOPE_IDENTITY()
    end
    else
        set @Mensaje = 'No se puede repetir el documento para más de un usuario'
    end
end
go
```

## Creación de los procedimientos para categorías.

```
/* ----- PROCEDIMIENTOS PARA CATEGORIA ----- */

create PROC SP_RegistrarCategoria(
    @Descripcion varchar(50),
    @Estado bit,
    @Resultado int output,
    @Mensaje varchar(500) output
)as
begin
    SET @Resultado = 0
    IF NOT EXISTS (SELECT * FROM CATEGORIA WHERE Descripcion = @Descripcion)
    begin
        insert into CATEGORIA(Descripcion,Estado) values (@Descripcion,@Estado)
        set @Resultado = SCOPE_IDENTITY()
    end
    ELSE
        set @Mensaje = 'No se puede repetir la descripcion de una categoria'
end

go

Create procedure sp_EditarCategoria(
    @IdCategoria int,
    @Descripcion varchar(50),
    @Estado bit,
    @Resultado bit output,
    @Mensaje varchar(500) output
)
as
begin
    SET @Resultado = 1
    IF NOT EXISTS (SELECT * FROM CATEGORIA WHERE Descripcion =@Descripcion and IdCategoria != @IdCategoria)
        update CATEGORIA set
            Descripcion = @Descripcion,
```

## Creación de los procedimientos para los productos.

```
/* ----- PROCEDIMIENTOS PARA PRODUCTO ----- */  
  
create PROC sp_RegistrarProducto(  
    @Codigo varchar(20),  
    @Nombre varchar(30),  
    @Descripcion varchar(30),  
    @IdCategoria int,  
    @Estado bit,  
    @Resultado int output,  
    @Mensaje varchar(500) output  
    ) as  
begin  
    SET @Resultado = 0  
    IF NOT EXISTS (SELECT * FROM producto WHERE Codigo = @Codigo)  
    begin  
        insert into producto(Codigo,Nombre,Descripcion,IdCategoria,Estado) values (@Codigo,@Nombre,@Descripcion,@IdCategoria,@Estado)  
        set @Resultado = SCOPE_IDENTITY()  
    end  
    ELSE  
        SET @Mensaje = 'Ya existe un producto con el mismo codigo'  
  
end  
GO  
  
create procedure sp_ModificarProducto(  
    @IdProducto int,  
    @Codigo varchar(20),  
    @Nombre varchar(30),  
    @Descripcion varchar(30),  
    @IdCategoria int,  
    @Estado bit,  
    @Resultado bit output,  
    @Mensaje varchar(500) output  
    )  
as  
begin  
    SET @Resultado = 1
```

## Creación de los procedimientos para el cliente.

```
/* ----- PROCEDIMIENTOS PARA CLIENTE ----- */  
  
create PROC sp_RegistrarCliente(  
    @Documento varchar(50),  
    @NombreCompleto varchar(50),  
    @Correo varchar(50),  
    @Telefono varchar(50),  
    @Estado bit,  
    @Resultado int output,  
    @Mensaje varchar(500) output  
    )as  
begin  
    SET @Resultado = 0  
    DECLARE @IDPERSONA INT  
    IF NOT EXISTS (SELECT * FROM CLIENTE WHERE Documento = @Documento)  
    begin  
        insert into CLIENTE(Documento,NombreCompleto,Correo,Telefono,Estado) values (  
            @Documento,@NombreCompleto,@Correo,@Telefono,@Estado)  
  
        set @Resultado = SCOPE_IDENTITY()  
    end  
    else  
        set @Mensaje = 'El numero de documento ya existe'  
end  
go  
  
create PROC sp_ModificarCliente(  
    @IdCliente int,  
    @Documento varchar(50),  
    @NombreCompleto varchar(50),  
    @Correo varchar(50),  
    @Telefono varchar(50),  
    @Estado bit,  
    @Resultado bit output,  
    @Mensaje varchar(500) output  
    )as
```



## Creación de los procedimientos para los proveedores.

```
/* ----- PROCEDIMIENTOS PARA PROVEEDOR ----- */  
  
create PROC sp_RegistrarProveedor(  
    @Documento varchar(50),  
    @RazonSocial varchar(50),  
    @Correo varchar(50),  
    @Telefono varchar(50),  
    @Estado bit,  
    @Resultado int output,  
    @Mensaje varchar(500) output  
    )as  
begin  
    SET @Resultado = 0  
    DECLARE @IDPERSONA INT  
    IF NOT EXISTS (SELECT * FROM PROVEEDOR WHERE Documento = @Documento)  
    begin  
        insert into PROVEEDOR(Documento,RazonSocial,Correo,Telefono,Estado) values (  
            @Documento,@RazonSocial,@Correo,@Telefono,@Estado)  
  
        set @Resultado = SCOPE_IDENTITY()  
    end  
    else  
        set @Mensaje = 'El numero de documento ya existe'  
    end  
GO  
  
create PROC sp_ModificarProveedor(  
    @IdProveedor int,  
    @Documento varchar(50),  
    @RazonSocial varchar(50),  
    @Correo varchar(50),  
    @Telefono varchar(50),  
    @Estado bit,  
    @Resultado bit output,  
    @Mensaje varchar(500) output  
    )as  
begin
```

## Creación de los procedimientos para registrar una compra.

```
/* PROCESOS PARA REGISTRAR UNA COMPRA */

CREATE TYPE [dbo].[EDetalle_Compra] AS TABLE(
    [IdProducto] int NULL,
    [PrecioCompra] decimal(18,2) NULL,
    [PrecioVenta] decimal(18,2) NULL,
    [Cantidad] int NULL,
    [MontoTotal] decimal(18,2) NULL
)
GO

CREATE PROCEDURE sp_RegistrarCompra(
    @IdUsuario int,
    @IdProveedor int,
    @TipoDocumento varchar(500),
    @NumeroDocumento varchar(500),
    @MontoTotal decimal(18,2),
    @DetalleCompra [EDetalle_Compra] READONLY,
    @Resultado bit output,
    @Mensaje varchar(500) output
)
as
begin
    begin try

        declare @idcompra int = 0
        set @Resultado = 1
        set @Mensaje = ''

        begin transaction registro

            insert into COMPRA(IdUsuario,IdProveedor,TipoDocumento,NumeroDocumento,MontoTotal)
            values(@IdUsuario,@IdProveedor,@TipoDocumento,@NumeroDocumento,@MontoTotal)
```

## Creación de los procedimientos para registrar una venta.

```
/* PROCESOS PARA REGISTRAR UNA VENTA */

CREATE TYPE [dbo].[EDetalle_Venta] AS TABLE(
    [IdProducto] int NULL,
    [PrecioVenta] decimal(18,2) NULL,
    [Cantidad] int NULL,
    [SubTotal] decimal(18,2) NULL
)
GO

create procedure usp_RegistrarVenta(
    @IdUsuario int,
    @TipoDocumento varchar(500),
    @NumeroDocumento varchar(500),
    @DocumentoCliente varchar(500),
    @NombreCliente varchar(500),
    @MontoPago decimal(18,2),
    @MontoCambio decimal(18,2),
    @MontoTotal decimal(18,2),
    @DetalleVenta [EDetalle_Venta] READONLY,
    @Resultado bit output,
    @Mensaje varchar(500) output
)
as
begin
    begin try
        declare @idventa int = 0
        set @Resultado = 1
        set @Mensaje = ''
```

## Proceso de insertación de registros a las tablas.

```

--/***** INSERTAMOS REGISTROS A LAS TABLAS *****/
--/*-----*/
GO

insert into rol (Descripcion)
values('ADMINISTRADOR')

GO

insert into rol (Descripcion)
values('EMPLEADO')

GO

insert into USUARIO(Documento,NombreCompleto,Correo,Clave,IdRol,Estado)
values
('101010','ADMIN','@GMAIL.COM','123',1,1)

GO

insert into USUARIO(Documento,NombreCompleto,Correo,Clave,IdRol,Estado)
values
('20','EMPLEADO','@GMAIL.COM','456',2,1)

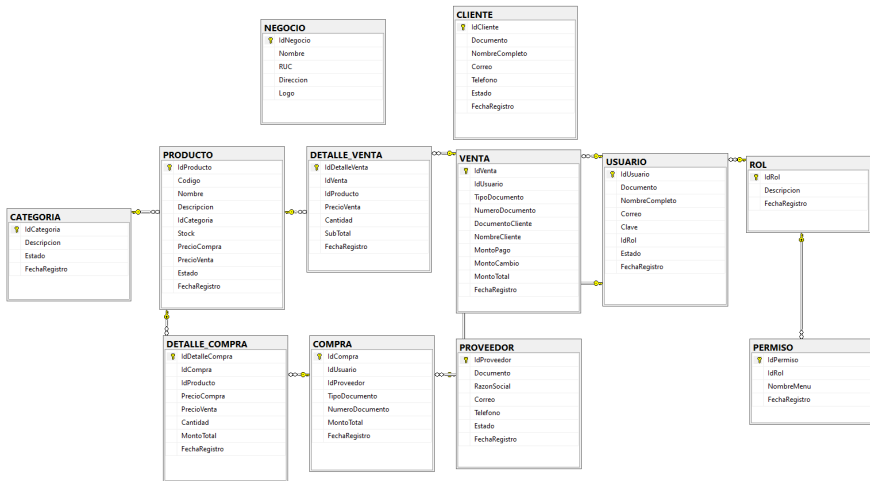
GO

insert into PERMISO(IdRol,NombreMenu) values
(1,'menuusuarios'),
(1,'menumantenedor'),
(1,'menuventas'),
(1,'menucompras'),
(1,'menuclientes'),
(1,'menuproveedores'),
(1,'menureportes'),
(1,'menuacercade')
```

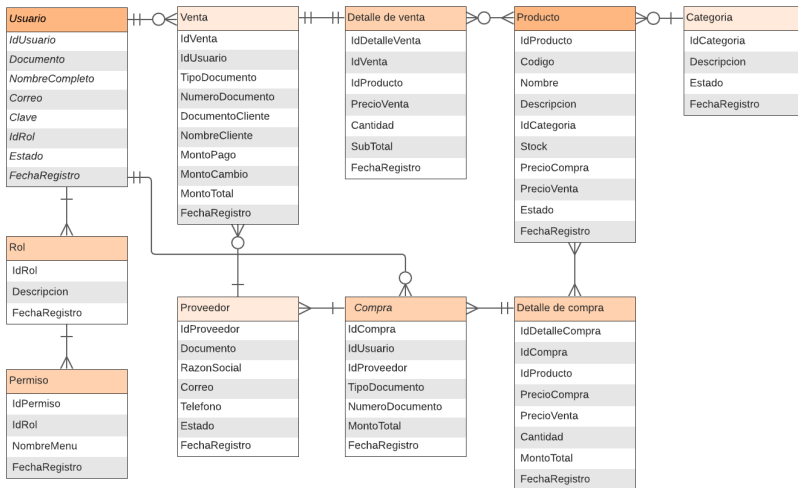
# Contenido

- 1 Introducción
- 2 Uso de C#
- 3 Uso de SQL
- 4 Diagrama del sistema**
- 5 Uso de clases
  - Uso de CapaDatos
  - Uso de CapaEntidad
  - Uso de CapaNegocio
  - Uso de CapaPresentación
- 6 Implementación del proyecto

# Diagrama del sistema



# Diagrama de sistema



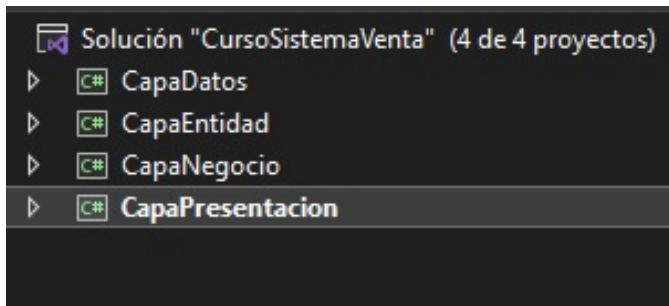
# Contenido

- 1 Introducción
- 2 Uso de C#
- 3 Uso de SQL
- 4 Diagrama del sistema
- 5 Uso de clases**
  - Uso de CapaDatos
  - Uso de CapaEntidad
  - Uso de CapaNegocio
  - Uso de CapaPresentación
- 6 Implementación del proyecto

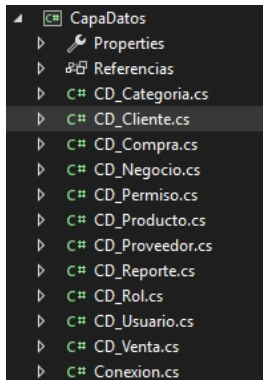


# Uso de Clases

Nuestro código se subdivide en 4 principales clases (capas) donde cada una de ellas cumple un papel fundamental y a su vez están relacionadas entre sí para el correcto funcionamiento, estas son CapaDatos, CapaEntidad, CapaNegocio y CapaPresentación, evidentemente contienen subclases similares porque están relacionadas.

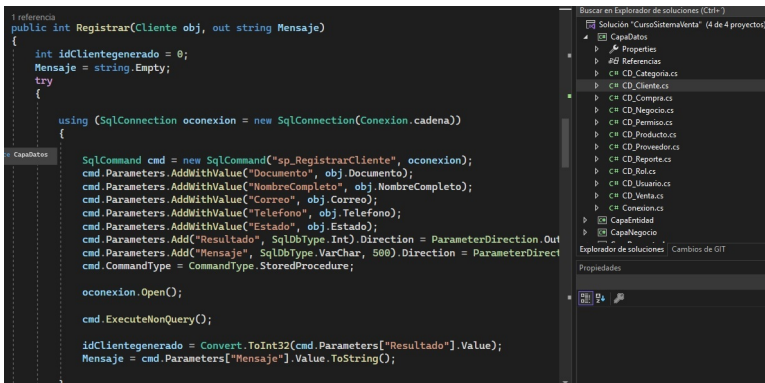


Esta clase es la más importante de todas porque contiene gran parte de la implementación del sistema.



# CapaDatos

Contiene subclases como Categoría, Cliente, Compra, etc, y a su vez comprende métodos declarados, entre ellos, Registrar, Editar, Eliminar, todas las clases funcionan bajo esta clase, aquí una captura de la subclase Cliente y mostrando el método Registrar.



```
1 referencia
public int Registrar(Cliente obj, out string Mensaje)
{
    int idClientegenerado = 0;
    Mensaje = string.Empty;
    try
    {
        using (SqlConnection oconexion = new SqlConnection(Conexion.cadena))
        {
            SqlCommand cmd = new SqlCommand("sp_RegistrarCliente", oconexion);
            cmd.Parameters.AddWithValue("Documento", obj.Documento);
            cmd.Parameters.AddWithValue("NombreCompleto", obj.NombreCompleto);
            cmd.Parameters.AddWithValue("Correo", obj.Correo);
            cmd.Parameters.AddWithValue("Telefono", obj.Telefono);
            cmd.Parameters.AddWithValue("Estado", obj.Estado);
            cmd.Parameters.Add("Resultado", SqlDbType.Int).Direction = ParameterDirection.Out;
            cmd.Parameters.Add("Mensaje", SqlDbType.VarChar, 500).Direction = ParameterDirection.Out;
            cmd.CommandType = CommandType.StoredProcedure;

            oconexion.Open();

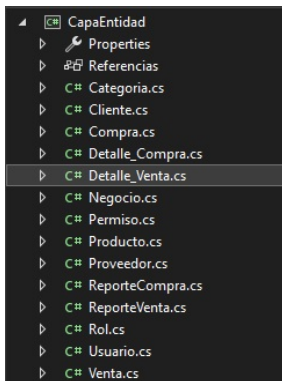
            cmd.ExecuteNonQuery();

            idClientegenerado = Convert.ToInt32(cmd.Parameters["Resultado"].Value);
            Mensaje = cmd.Parameters["Mensaje"].Value.ToString();
        }
    }
}
```

El explorador de soluciones a la derecha muestra la estructura del proyecto 'CursoSistemaVenta' con subcarpetas para CapaDatos, CapaEntidad y CapaNegocio, y una lista de archivos de código fuente como CD\_Categorias.cs, CD\_Cliente.cs, etc.

# CapaEntidad

Esta clase define los métodos que darán acceso al tipo de usuario que está ingresando, además referencia todos los métodos que va a utilizar cada subclase.



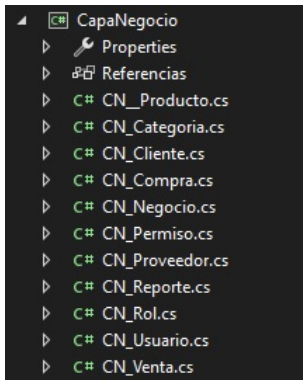
# CapaEntidad

Por ejemplo, aquí tenemos una subclase de la clase CapaEntidad, como podemos ver se definen métodos que van a interactuar con las otras clases y subclases del código.

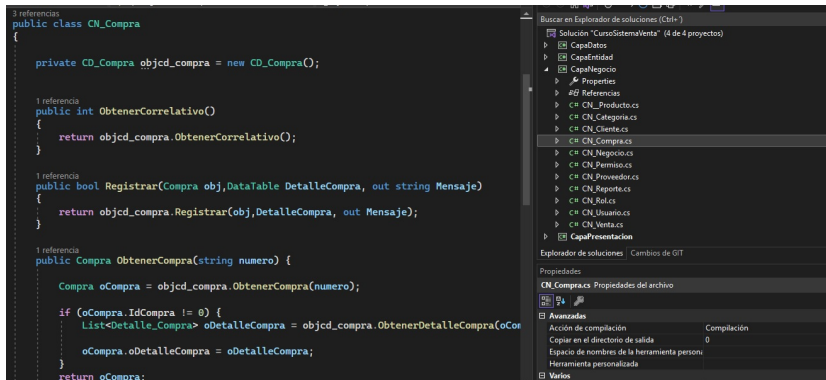
```
public class ReporteCompra
{
    public string FechaRegistro { get; set; }
    public string TipoDocumento { get; set; }
    public string NumeroDocumento { get; set; }
    public string MontoTotal { get; set; }
    public string UsuarioRegistro { get; set; }
    public string DocumentoProveedor { get; set; }
    public string RazonSocial { get; set; }
    public stringCodigoProducto { get; set; }
    public string NombreProducto { get; set; }
    public string Categoria { get; set; }
    public string PrecioCompra { get; set; }
    public string PrecioVenta { get; set; }
    public string Cantidad { get; set; }
    public string SubTotal { get; set; }
}
```

# CapaNegocio

Esta clase nos va a permitir crear las listas y a su vez definir los métodos que van a usar estas listas de cada subclase.

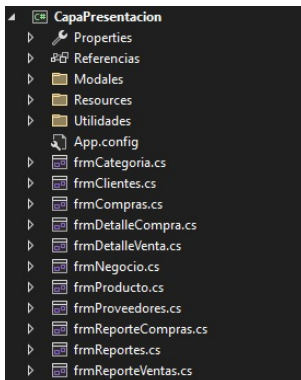


Por ejemplo de esta clase se muestra la subclase Compra, en ella se definen los métodos que va a tener el negocio como registrar la compra, efectuar la compra, etc.



# CapaPresentación

Esta clase sirve para preparar toda la interfaz del sistema, así como las ventanas que va a usar cada subclase, en otras palabras, es todo el menú que se muestra en pantalla.





# CapaPresentación

En este caso se muestra la ventana que genera ReporteCompras, en ella nos muestra los detalles de las compras realizadas por el cliente, así como también las fechas en las que el administrador quiere tener en su reporte.

The screenshot displays the 'frmReporteCompras' application window. The title bar reads 'frmReporteCompras'. The main content area is titled 'Reporte Compras'. Below the title, there are filter controls: 'Fecha Inicio:' with a date picker set to '28/10/2022', 'Fecha Fin:' with a date picker set to '28/10/2022', and 'Proveedor:' with a dropdown menu. A 'Buscar' button is located to the right of these filters. Below the filters, there is a 'Descargar Excel' button on the left and a 'Buscar por:' dropdown on the right. The main area contains a table with the following columns: Fecha Registro, Tipo Documento, Numero Documento, Monto Total, Usuario Registro, Documento Proveedor, Razon Social,Codigo Producto, and Nombre Producto. The table is currently empty. On the right side of the application, a Visual Studio solution explorer is visible, showing the project 'CursoSistemaVenta' with 4 sub-projects. The 'CapaPresentacion' project is expanded, showing a list of files including 'Properties', 'Referencias', 'Modulos', 'Resources', 'Utilidades', 'App.config', and several '.cs' files. The file 'frmReporteCompras.cs' is currently selected and highlighted.

Fecha Registro	Tipo Documento	Numero Documento	Monto Total	Usuario Registro	Documento Proveedor	Razon Social	Codigo Producto	Nombre Producto
----------------	----------------	------------------	-------------	------------------	---------------------	--------------	-----------------	-----------------

# Contenido

- 1 Introducción
- 2 Uso de C#
- 3 Uso de SQL
- 4 Diagrama del sistema
- 5 Uso de clases
  - Uso de CapaDatos
  - Uso de CapaEntidad
  - Uso de CapaNegocio
  - Uso de CapaPresentación
- 6 Implementación del proyecto



**SISTEMA DE VENTA**

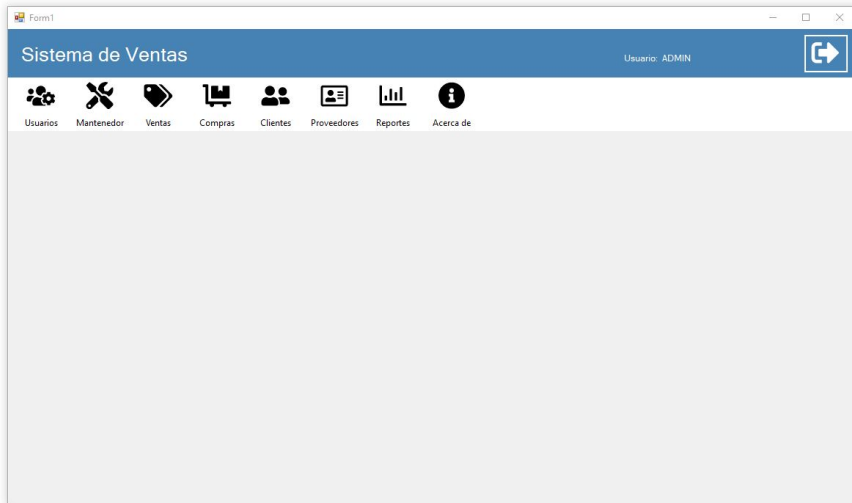
## INICIAR SESION



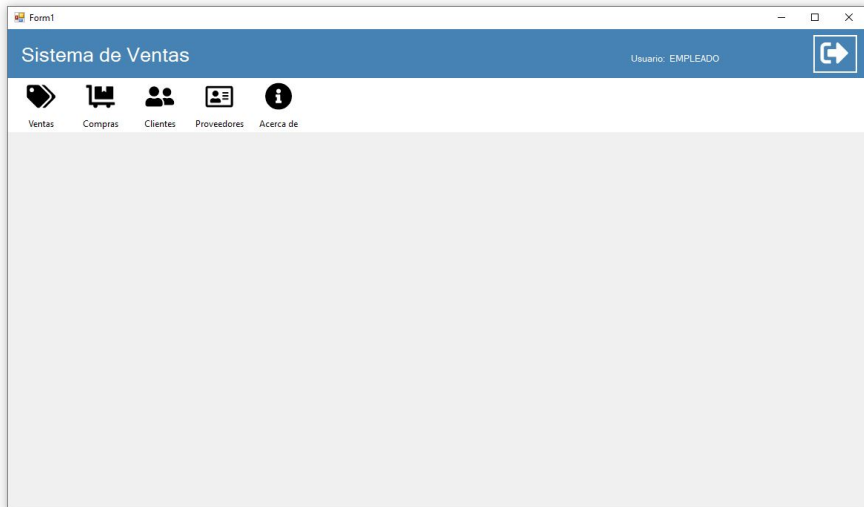
Nro Documento:

Contraseña:

# Ejecución



# Ejecución



Form1

Sistema de Ventas

Usuario: ADMIN

Usuarios

Mantenedor

Ventas

Compras

Clientes

Proveedores

Reportes

Acerca de

Detalle Usuario

Nro Documento:

Nombre Completo:

Correo:

Contraseña:

Confirmar Contraseña:

Rol:  
ADMINISTRADOR

Estado:  
Activo

Guardar

Limpiar

Eliminar

Lista de Usuarios:

Buscar por: Nro Documento

	Nro Documento	Nombre Completo	Correo	Rol	Estado
▶	101010	ADMIN	@GMAIL.COM	ADMINISTRADOR	Activo
	20	EMPLEADO	@GMAIL.COM	EMPLEADO	Activo

Exposición Final

Sistema de venta

29 de Octubre de 2022


38 / 48


## Ejecución


Form1


Sistema de Ventas


Usuario: ADMIN


 Usuarios


 Mantenedor


 Ventas

 Compras

 Clientes

 Proveedores

 Reportes

 Acerca de

Detalle Categoría

Descripción:

Estado:

Activo

Guardar

Limpiar

Eliminar

Lista de Categorías:

Buscar por:

Descripción

	Descripcion	Estado
▶	bebidas	Activo
	carnes	Activo
	licores	No Activo

Form1

Sistema de Ventas

Usuario: ADMIN

Usuarios

Mantenedor

Ventas

Compras

Clientes

Proveedores

Reportes

Acerca de

Detalle Producto

Codigo:

Nombre:

Descripcion:

Categoria:

bebidas

Estado:

Activo

Guardar

Limpiar

Eliminar

Lista de Productos:

Descargar Excel

Buscar por: Codigo

		Codigo	Nombre	Descripcion	Categoria	Stock	Precio Compra	Precio Venta	E
	✓	1	frugos del valle	500 ml	bebidas	0	0.00	0.00	Act
	✓	2	pollo entero fresco	2.2 kg aprox	carnes	99	14.50	22.50	Act

Exposición Final

Sistema de venta

29 de Octubre de 2022

40 / 48



Form1


## Sistema de Ventas

Usuario: ADMIN

Usuarios Mantenedor Ventas Compras Clientes Proveedores Reportes Acerca de

### Detalle Negocio

Logo:



Nombre Negocio:  
Sistema De Ventas

R.U.C.:  
10984116698

Dirección:  
av. Hermes 223

Subir Guardar Cambios

## Ejecución

Form1

Sistema de Ventas

Usuario: ADMIN

Usuarios

Mantenedor

Ventas

Compras

Cientes

Proveedores

Reportes

Acerca de

Registrar Venta

Información Venta

Fecha:  
29/10/2022

Tipo Documento:  
Boleta

Información Cliente

Número Documento:

Nombre Completo:

Información de Producto

Cod. Producto:

Producto:

Precio:

Stock:

Cantidad:  
1

Producto	Precio	Cantidad	Sub Total
----------	--------	----------	-----------

Total a Pagar:  
0

Paga con:

Cambio:

Crear Venta

Form1

Sistema de Ventas

Usuario: ADMIN

Usuarios Mantenedor Ventas Compras Clientes Proveedores Reportes Acerca de

### Detalle Venta

Numero Documento:

Información Venta

Fecha:  Tipo Documento:  Usuario:

Información Cliente

Documento Cliente:  Nombre Cliente:

Producto	Precio	Cantidad	Sub Total
----------	--------	----------	-----------

Monto Total:  Monto Pago:  Monto Cambio:

Form1

## Sistema de Ventas

Usuario: ADMIN

Usuarios Mantenedor Ventas **Compras** Clientes Proveedores Reportes Acerca de

### Registrar Compra

Información Compra

Fecha: 29/10/2022

Tipo Documento: Boleta

Información Proveedor

Número Documento:

Razón Social:

Información de Producto

Cod. Producto:   Producto:  Precio Compra:  Precio Venta:  Cantidad:

Producto	Precio Compra	Cantidad	Sub Total
----------	---------------	----------	-----------

Total a Pagar:

Form1

Sistema de Ventas

Usuario: ADMIN

Usuarios

Mantenedor

Ventas

Compras

Clientes

Proveedores

Reportes

Acerca de

Detalle Compra

Numero Documento:

Información Compra

Fecha:  Tipo Documento:  Usuario:

Información Proveedor

Número Documento:  Razón Social:

Producto	Precio Compra	Cantidad	Sub Total
----------	---------------	----------	-----------

Monto Total:

Form1

Sistema de Ventas

Usuario: ADMIN

Usuarios

Mantenedor

Ventas

Compras

Clientes

Proveedores

Reportes

Acerca de

Detalle Cliente

Nro Documento:

Nombre Completo:

Correo:

Telefono:

Estado:  
Activo

Guardar

Limpiar

Eliminar

Lista de Clientes:

Buscar por: Nro Documento

	Nro Documento	Nombre Completo	Correo	Telefono	Estado
<input checked="" type="checkbox"/>	2022	carlos	12345@gmail.com	984116698	Activo

Form1

Sistema de Ventas

Usuario: ADMIN

Usuarios

Mantenedor

Ventas

Compras

Clientes

Proveedores

Reportes

Acerca de

Detalle Proveedor

Nro Documento:

Razon Social:

Correo:

Telefono:

Estado:  
Activo

Guardar

Limpiar

Eliminar

Lista de Proveedores:

Buscar por: Nro Documento

	Nro Documento	Razon Social	Correo	Telefono	Estado
▶	48229016	Coca cola	123@gmail.com	3489305	Activo
	48239016	Gloria	1234@gmail.com	3489306	Activo
	42249016	san fernado	12345@gmail.com	3489307	Activo

Form1

Sistema de Ventas

Usuario: ADMIN

Usuarios

Mantenedor

Ventas

Compras

Clientes

Proveedores

Reportes

Acerca de

Reporte Compras

Fecha Inicio: 19/10/2022

Fecha Fin: 29/10/2022

Proveedor: TODOS

Buscar

Descargar Excel

Buscar por: Fecha Registro

Q

	Fecha Registro	Tipo Documento	Numero Documento	Monto Total	Usuario Registro	Documento Proveedor	Razon Social	Codigo Producto	Nombre Producto	Categoria	Precio Compra
▶	28/10/2022	Boleta	00001	1450.00	ADMIN	42249016	san fernado	2	pollo entero fresco	carnes	14.50