



MANUAL TÉCNICO APLICACIÓN DE VENTAS ALL-IN-ONE MART S.A

Aplicación para la compraventa de productos de miscelánea.

Versión 1.0.

Realizado por: Pelsar Adolfo Donis Donis

Carné: 7490-20-12222

Índice

Índice.....	2
Introducción.....	3
Requisitos del Sistema	4
Requisitos de Software.....	4
Arquitectura de la aplicación.....	4
Instalación y configuración	4
Apertura de la Aplicación	4
Instalación de Base de Datos	5
Tablas	7
Triggers	7
Inserts.....	9
Conexión Base de Datos con Programa en VS	10
Funcionalidades principales	13
Generación de informes	13
Soporte Técnico	13
Diagramas	14
Diagrama de Caso de Uso.....	14
Diagrama Entidad Relación	15
Diagrama de Clases	16

Introducción

El Manual Técnico de la Aplicación All-In-One Mart S.A proporciona información detallada sobre la arquitectura, configuración y funcionamiento interno de la aplicación. Este manual está dirigido a desarrolladores, administradores del sistema y otros profesionales técnicos involucrados en la implementación y mantenimiento de la aplicación.

Requisitos del Sistema

- Requisitos de Hardware
- Procesador: mínimo 2 GHz de velocidad
- Memoria RAM: mínimo 4 GB
- Espacio en Disco: mínimo 2 GB

Requisitos de Software

- Sistema Operativo: Windows 8 o superior
- Visual Studio Community 2022
- SQL Server


Arquitectura de la aplicación

Sigue una arquitectura de Windows Forms, el cual se instala como una aplicación ejecutable para ordenadores. Utiliza una Base de Datos relacional para almacenar la información.

Instalación y configuración

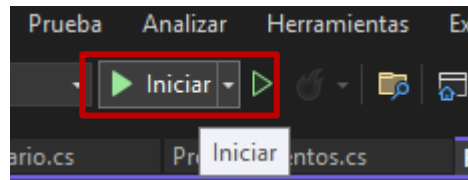
Apertura de la Aplicación

Al clonar nuestro proyecto, dentro de la carpeta “BBDD-Proyecto” nos encontraremos con los siguientes archivos:

<input type="checkbox"/> Nombre	Fecha de modificación	Tipo	Tamaño
.vs	10/05/2023 10:35	Carpeta de archivos	
BBDD-Proyecto	20/05/2023 12:09	Carpeta de archivos	
images	18/05/2023 7:46	Carpeta de archivos	
 BBDD-Proyecto.sln	10/05/2023 10:35	Visual Studio Solu...	2 KB






Donde, daremos doble click a “BBDD-Proyecto.sln” para que nos abra nuestro proyecto en Visual Studio.

Luego, cuando nos abra nuestro programa, daremos click al ícono “Iniciar” para que nos ejecute como tal, nuestro programa.

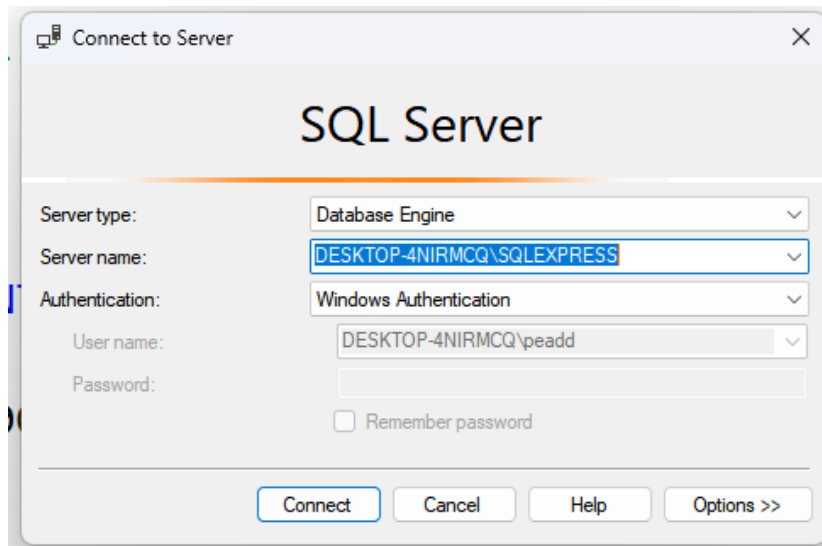


Instalación de Base de Datos

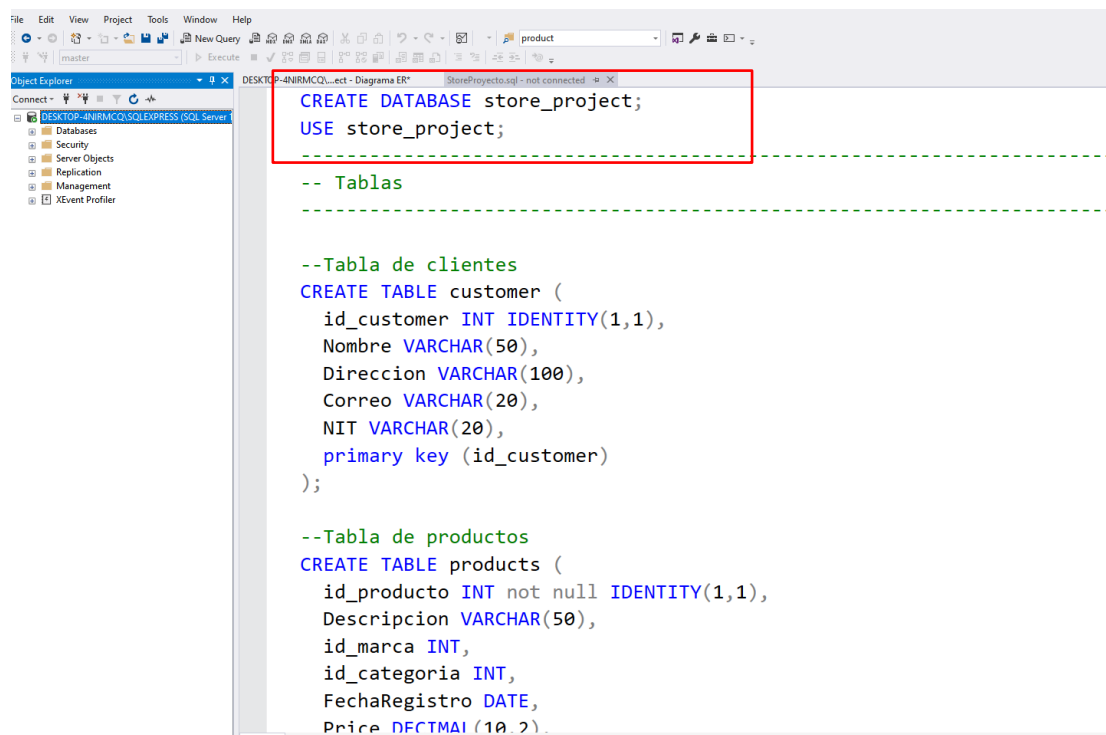
Ejecutamos el archivo “StoreProyecto”

	.vs	10/05/2023 10:35	Carpeta de archivos	
	BBDD-Proyecto	20/05/2023 12:48	Carpeta de archivos	
	images	18/05/2023 7:46	Carpeta de archivos	
	BBDD-Proyecto.sln	10/05/2023 10:35	Visual Studio Solu...	2 KB
	StoreProyecto	18/05/2023 9:01	Microsoft SQL Ser...	9 KB

Nos abrirá SQL Server y nos mostrará esta ventana de LOGIN, donde, vamos a copiar el valor que se encuentra en “Server Name” ya que nos servirá para enlazar nuestra Base de Datos con el programa más adelante.



Luego de logearnos, nos mostrará la siguiente ventana donde, tenemos que ejecutar primero el crear la base de datos “store_project” y luego usar esa base de datos para luego realizar la creación de las debidas tablas.



Luego de haber realizado dicho proceso, ejecutamos la creación de cada una de las tablas, triggers y después los debidos inserts que se encuentran en el query.

Tablas

```
--Tabla de clientes
CREATE TABLE customer (
  id_customer INT IDENTITY(1,1),
  Nombre VARCHAR(50),
  Direccion VARCHAR(100),
  Correo VARCHAR(20),
  NIT VARCHAR(20),
  primary key (id_customer)
);

--Tabla de productos
CREATE TABLE products (
  id_producto INT not null IDENTITY(1,1),
  Description VARCHAR(50),
  id_marca INT,
  id_categoria INT,
  FechaRegistro DATE,
  Price DECIMAL(10,2),
  primary key (id_producto)
);

--Tabla de empresas
CREATE TABLE company (
  NIT varchar(20),
  Nombre VARCHAR(50),
  Direccion VARCHAR(50),
  Telefono VARCHAR(20),
  Correo VARCHAR(20),
  Portal VARCHAR(20),
  primary key (NIT)
);

--Tabla de ventas, 1
create table Sales_d (
  id_sale int not null identity(1,1),
  cantidad int not null,
  product INT not null REFERENCES products(id_producto),
  factura int not null FOREIGN KEY REFERENCES Sales_h(Numero),
  serie varchar (10) not null,
);

--Tabla de ventas, 2
CREATE TABLE Sales_h (
  Numero int IDENTITY(1,1) PRIMARY KEY,
  Serie varchar(10) NOT NULL,
  Fecha date NOT NULL,
  codigo_cliente int NOT NULL REFERENCES customer(id_customer),
  nit_empresa varchar(20) NOT NULL REFERENCES company(NIT)
);

--Tabla de compras
CREATE TABLE compras (
  id INT PRIMARY KEY IDENTITY(1,1),
  descripcion VARCHAR(255),
  id_producto INT FOREIGN KEY REFERENCES products(id_producto),
  cantidad INT,
  fecha DATE
);

--Tabla de inventario
CREATE TABLE Inventory(
  id_inventory int not null identity(1,1),
  date_int date,
  date_out date,
  id_product int not null FOREIGN KEY REFERENCES products(id_producto),
  stock_in int not null,
  entries int not null,
  outlets int not null,
  primary key (id_inventory)
);
```

Triggers

```
-- Triggers

--TRIGGER QUE SE EJECUTARA A LA HORA DE REALIZAR UNA COMPRA PARA ACTUALIZAR LA TABLA DE INVENTARIO

Create trigger TrCompraInventrario
on compras for insert
as
Declare @Cantidad int
Declare @Stock_in int
Declare @entries int
Declare @outlets int
Declare @Date_int date
Declare @Date_out date
Declare @id_producto int

Select @id_producto = id_producto, @Cantidad = cantidad, @Date_int = fecha, @Date_out = fecha from inserted
Select @Stock_in =stock_in from Inventory where id_product = @id_producto;
Update Inventory set Inventory.stock_in = @Cantidad + @Stock_in, Inventory.date_int = @Date_int, Inventory.date_out = Inventory.date_out,
Inventory.id_product = @id_producto, Inventory.entries = @Cantidad + Inventory.entries, Inventory.outlets = Inventory.outlets
where Inventory.id_product = @id_producto
go
```

```

--Agrega al inventario el producto creado
Create trigger trAgregarProductoInventario
On products for insert
as
Declare @Stock_in int
Declare @entries int
Declare @outlets int
Declare @Date_int date
Declare @Date_out date
Declare @id_producto int

Select @id_producto = id_producto, @Date_int = FechaRegistro, @Date_out = FechaRegistro from inserted
insert into Inventory(id_product, outlets, stock_in, entries, date_int, date_out)
values(@id_producto, 0, 0, 0, @Date_int, @Date_out)
go

--Trigger cuando se haga una venta, actualizar el inventario
Create trigger TrVentaInventrario
on Sales_d for insert
as
Declare @Cantidad int
Declare @Stock_in int
Declare @entries int
Declare @outlets int
Declare @Date_int date
Declare @Date_out date
Declare @id_producto int
Declare @factura int
Declare @serie varchar(10)

Select @id_producto = product, @Cantidad = cantidad, @factura = factura, @serie = serie from inserted
Select @Stock_in = stock_in from Inventory where id_product = @id_producto;
Select @Date_out = Fecha from Sales_h where @factura = Numero

if(@Stock_in >= @Cantidad)
begin
    Update Inventory set Inventory.stock_in = @Stock_in - @Cantidad, Inventory.date_int = Inventory.date_int, Inventory.date_out = @Date_out,
    Inventory.id_product = @id_producto, Inventory.entries = Inventory.entries, Inventory.outlets = Inventory.outlets + @Cantidad
    where Inventory.id_product = @id_producto
end
else
    RAISERROR('El producto no tiene stock', 16, 1)
    RETURN
go

```


Inserts

```
-- Inserts
```

```
--Insercion de clientes
```

```
INSERT INTO customer ( Nombre, Direccion, Correo, NIT)
VALUES
( 'Juan Perez', 'Calle 123, Ciudad', 'juanz@outlook.com', '123456-7'),
( 'Maria Rodriguez', 'Avenida 456, Ciudad', 'mariaz@gmail.com', '987654-3'),
( 'Pedro Gomez', 'Carrera 789, Ciudad', 'pedrz@outlook.com', '543210-1'),
( 'Oscar Perez', 'Avenida Las Hostias, Ciudad', 'oPerez@gmail.com', '585896-4'),
( 'Kevin Osorio', 'Calle las Piscinas, Ciudad', 'osorioK@gmail.com', '256933-8');
```

```
--Insercion de productos
```

```
INSERT INTO products (Descripcion, id_marca, id_categoria, FechaRegistro, Price)
VALUES
('Sopa Laky Men Camaron', 1, 1, GETDATE(), 5.00);
```

```
INSERT INTO products (Descripcion, id_marca, id_categoria, FechaRegistro, Price)
VALUES
('Coca-Cola 2.5 Litros', 2, 2, GETDATE(), 22.00);
```

```
INSERT INTO products (Descripcion, id_marca, id_categoria, FechaRegistro, Price)
VALUES
('Pepsi 3 Litros', 3, 2, GETDATE(), 15.00);
```

```
INSERT INTO products (Descripcion, id_marca, id_categoria, FechaRegistro, Price)
VALUES
('Lift 3 Litros', 4, 2, GETDATE(), 15.00);
```

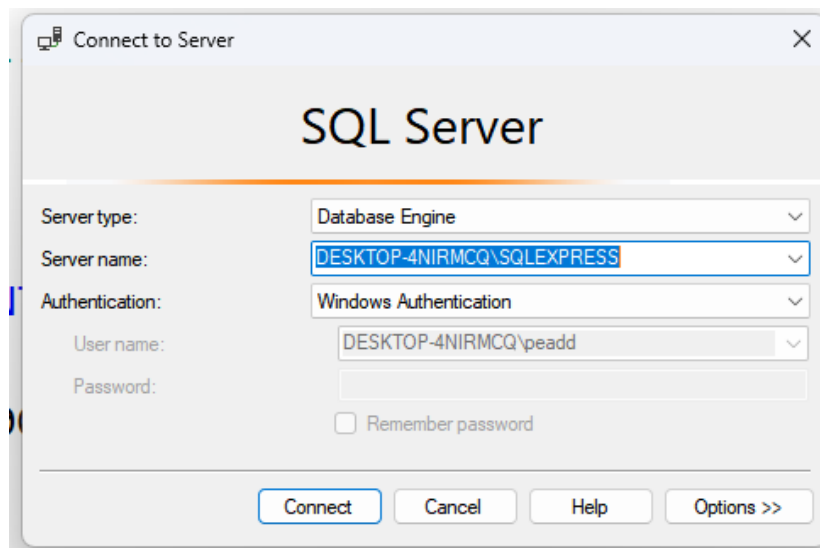
```
-- Insercion empresas
```

```
INSERT INTO company (NIT, Nombre, Direccion, Telefono, Correo, Portal)
VALUES
('123456-7', 'Los Pollos Hermanos S.A', 'Calle 1 #23-45', '555-1234', 'pollosher@gmail.com', 'www.pollosher.com'),
('987654-3', 'Las Margaritas S.A', 'Avenida 5 #12-34', '555-5678', 'margaritas@gmail.com', 'www.margaritas.com'),
('456789-0', 'La Esquinita', 'Carrera 8 #56-78', '555-9012', 'esquinita@gmail.com', 'www.esquinitas.com')
```

Conexión Base de Datos con Programa en VS

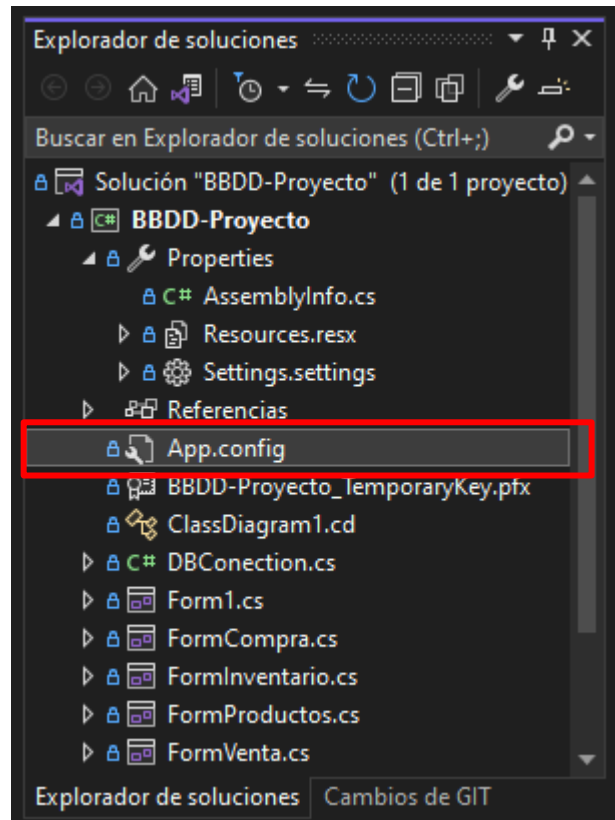
Después de haber creado la base de datos en nuestra computadora, con sus correspondientes tablas, triggers e inserts, debemos realizar la conexión con nuestro programa para que pueda usar y almacenar datos en la base de datos.

Entonces, debimos haber copiado el valor que se encuentra en “Server Name” al ejecutar el query en SQL Server.



Después de copiarla, nos vamos a nuestro Visual Studio donde se encuentra el programa como tal.

A la derecha, en el explorador de soluciones, buscamos el archivo que se llama "App.config" y lo abrimos.



En los dos apartados de Data Source, vamos a pegar la línea que copiamos.

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>

  <connectionStrings>
    <add name="ConnectionDDBB" connectionString="Data Source=DESKTOP-4NIRMCQ\SQLEXPRESS;Initial Catalog=store_project;Integrated Security=True" providerName="System.Data.SqlClient" />
    <add name="store_project.Properties.Settings.store_projectConnectionString" connectionString="Data Source=DESKTOP-4NIRMCQ\SQLEXPRESS;Initial Catalog=store_project;Integrated Security=True" providerName="System.Data.SqlClient" />
  </connectionStrings>

  <startup>
    <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.7.2" />
  </startup>
</configuration>
```

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>

  <connectionStrings>
    <add name="ConnectionDDBB" connectionString="Data Source=DESKTOP-4NIRMCQ\SQLEXPRESS;Initial Catalog=store_project;Integrated Security=True" providerName="System.Data.SqlClient" />
    <add name="store_project.Properties.Settings.store_projectConnectionString" connectionString="Data Source=DESKTOP-4NIRMCQ\SQLEXPRESS;Initial Catalog=store_project;Integrated Security=True" providerName="System.Data.SqlClient" />
  </connectionStrings>

  <startup>
    <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.7.2" />
  </startup>
</configuration>
```

Con estas configuraciones, nuestro programa está listo para funcionar de manera correcta.

Funcionalidades principales

El usuario puede crear compras y ventas, a su vez, agregar productos y verificar el inventario de estos.

Generación de informes

El usuario generará un comprobante luego de realizar una compra o venta.

El formato de cada comprobante es PDF.

Soporte Técnico

Si encuentra algún problema o tiene alguna pregunta técnica, póngase en contacto con el equipo de soporte técnico a través del correo electrónico support@allinonemartsa.com

Diagramas

Diagrama de Caso de Uso

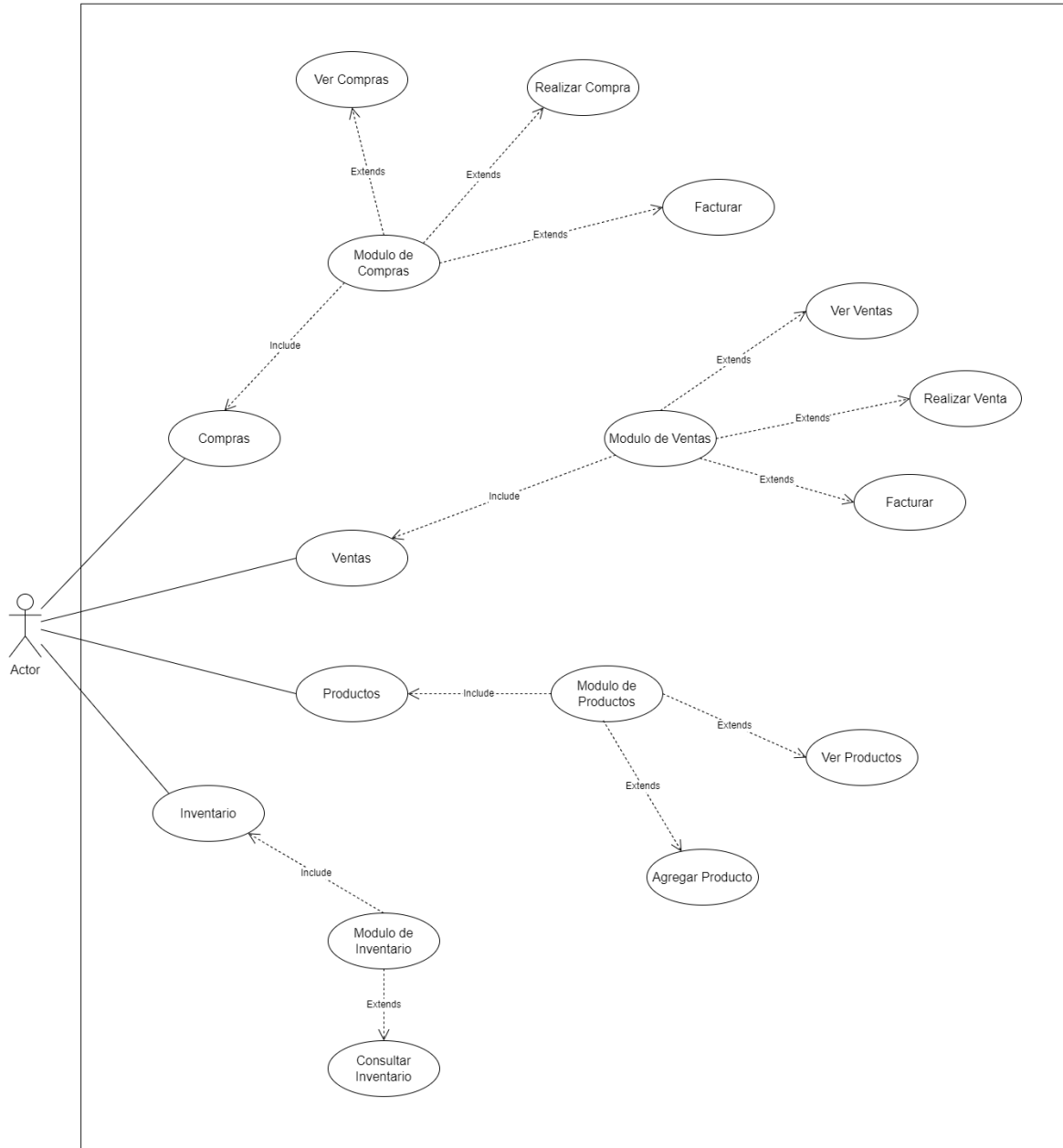


Diagrama Entidad Relación



Diagrama de Clases

