Universidad ORT Uruguay

Facultad de Ingeniería

Escuela de tecnología

Obligatorio 1 Programación 3

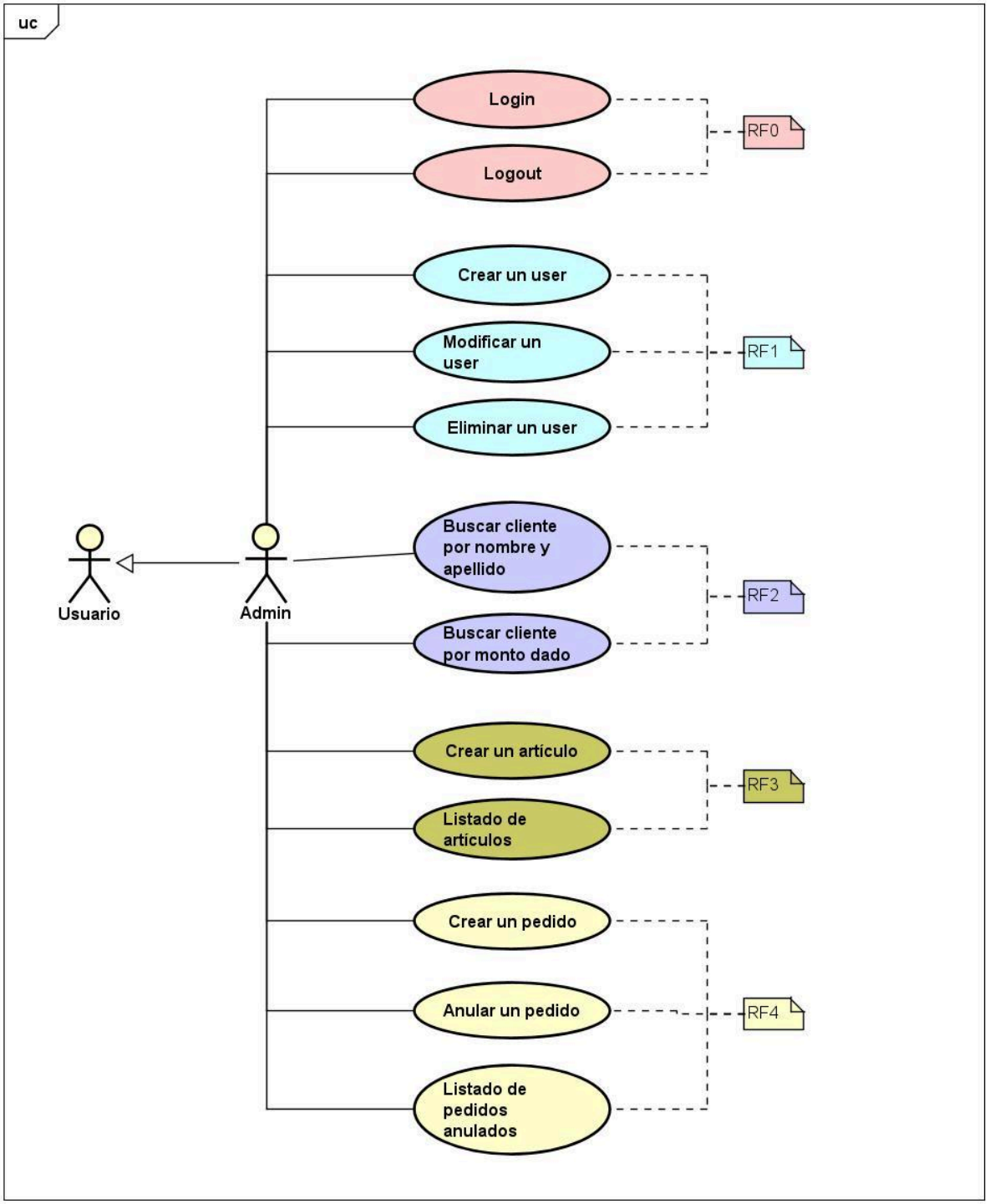
Agustina Sánchez Montoro - **200383**

Jonathan Lima - **318410**

Grupo: N3D
Docente: Lucas López
Mayo 2024
Analista Programador y Tecnologías de la Información

Indice

Casos de uso:


uc
Login
Logout
RF0
Crear un user
Modificar un user
Eliminar un user
RF1
Usuario
Admin
Buscar cliente por nombre y apellido
Buscar cliente por monto dado
RF2
Crear un artículo
Listado de artículos
RF3
Crear un pedido
Anular un pedido
Listado de pedidos anulados
RF4

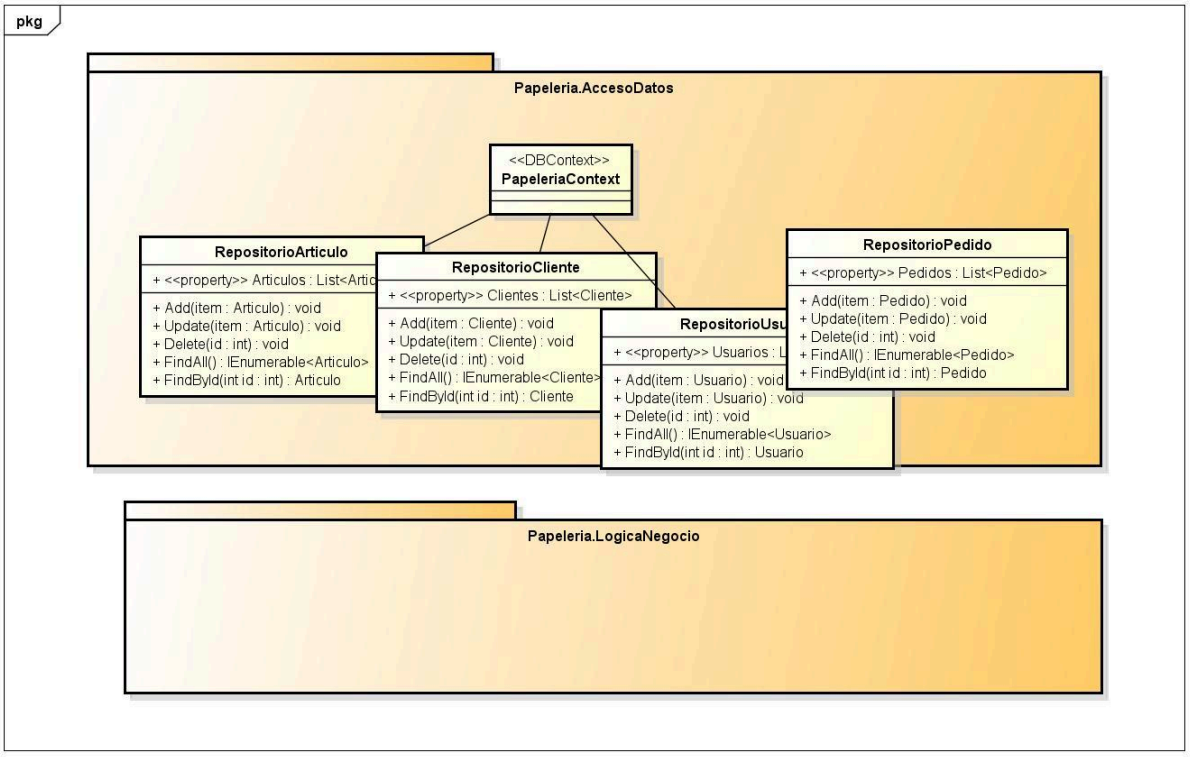
Descripción narrativa de Alta de pedido:

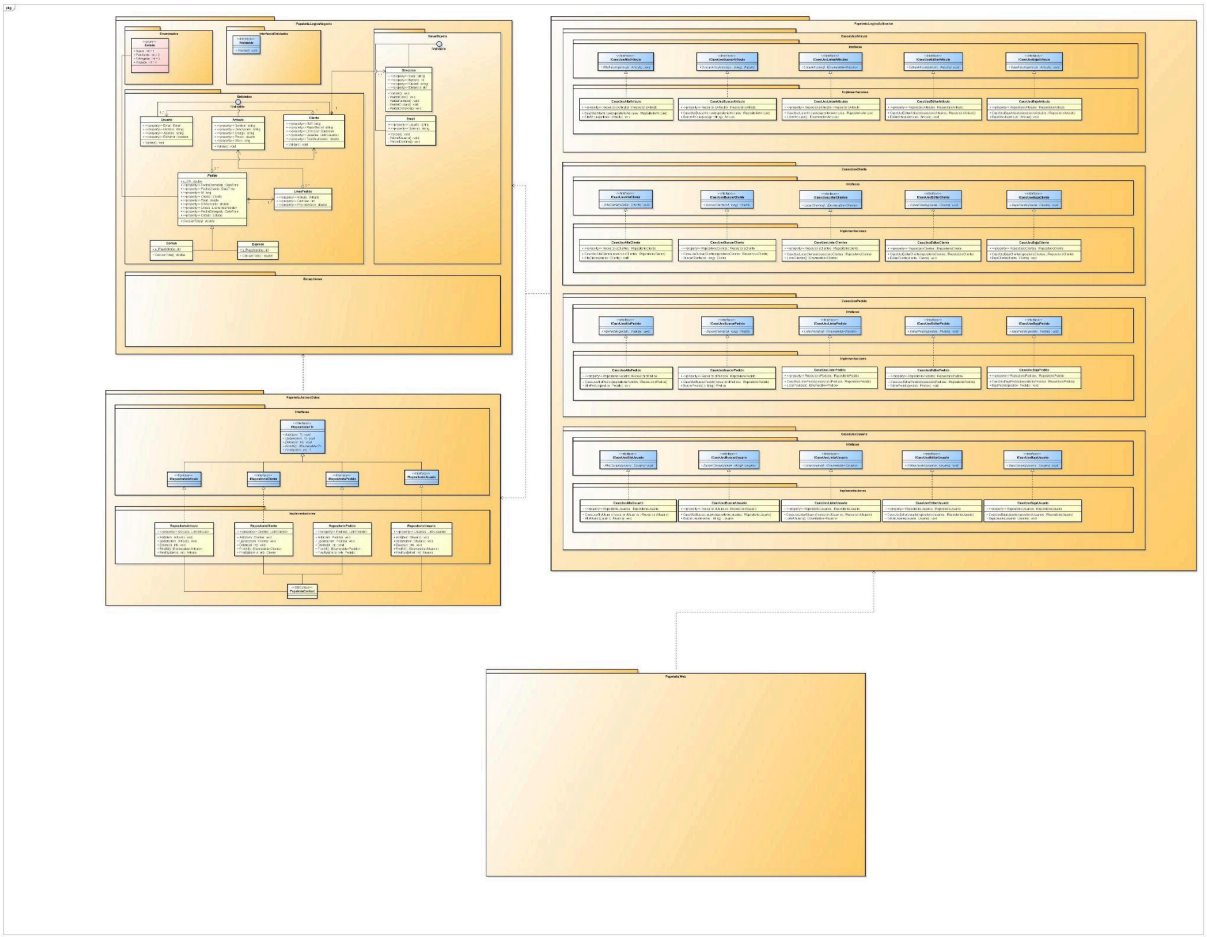
Caso Uso 1 - Alta de Pedido	Value
Descripción	Permite crear un nuevo pedido
Actor	Admin
Precondiciones	El actor está identificado con el usuario “administrador” y el pedido debe haber pasado por sus respectivas validaciones
Poscondiciones	El pedido una vez al pasar las validaciones/restricciones para dar el alta, queda registrado y se muestra en pantalla el monto total con impuestos correspondientes.
Flujo normal	1. El actor ingresa a dar de alta un pedido. 2. El sistema solicita: tipo de pedido, cliente, fecha de entrega, líneas: artículos y cantidad. 3. El actor introduce los datos solicitados y solicita continuar. 4. El sistema verifica que el cliente exista, la fecha del pedido que será la actual, stock suficiente. 5. El actor confirma el alta de pedido. 6. El sistema comprueba la validez de los datos, le asigna un ID y muestra en pantalla el monto total con los impuestos correspondientes.
Flujos alternativos	3a. El actor cancela el ingreso por ende el pedido no es guardado 4a. No se ingresa algún dato / fecha de entrega incorrecta / no hay stock suficiente / no existe el cliente por ende avisa que hay un error, marcando los campos erróneos con una descripción del error quedando a la espera de su corrección.
Flujos excepcionales	La comunicación con el servidor se corta durante el registro de alta de pedido y es descartado.

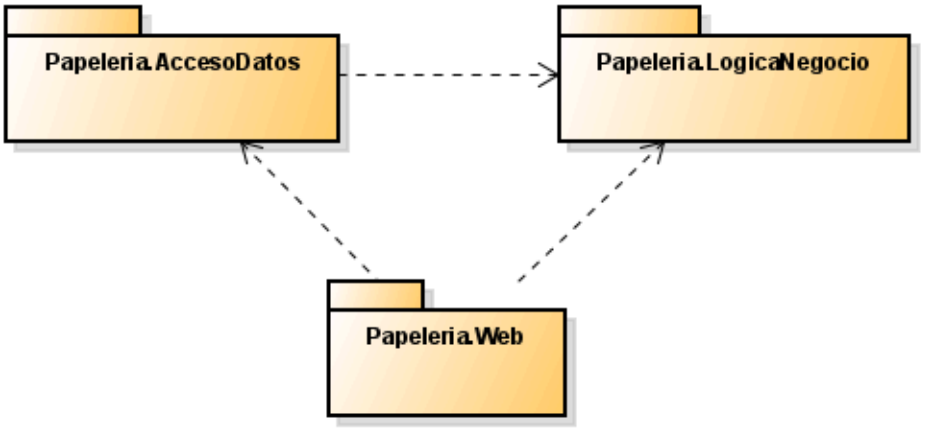
Descripción narrativa de Anulación de Pedido:

Caso Uso 2 - Anulación de Pedido	Value
Descripción	Se ingresa fecha de emisión del pedido y se muestra todos los pedidos de esa fecha no entregados
Actor	Admin
Precondiciones	El actor está identificado con el usuario “administrador” y el pedido debe estar dado de alta.
Poscondiciones	Se anula el pedido desplegando un listado con todos los pedidos anulados.
Flujo normal	1. El actor ingresa la fecha de emisión de un pedido 2. Se muestran todos los pedidos de esa fecha no entregados desplegando fecha prometida de entrega, cliente y total del pedido. 3. El actor selecciona un pedido para anularlo 4. Click en el botón de anular pedido 5. Pedido es anulado 6. Se despliega el listado de pedidos anulados ordenados descendentemente por fecha de pedido realizado con WebApi.
Flujos alternativos	1a. La fecha ingresada no coincide con ningún pedido, se muestra mensaje de error 2a. El cliente no tiene pedidos no entregados
Flujos excepcionales	La comunicación con el servidor se corta durante el registro de alta de pedido y es descartado.

UML:


pkg
Papeleria.AccesoDatos
<<DBContext>>
PapeleriaContext
RepositorioArticulo
+ <<property>> Articulos : List<Artic
+ Add(item : Articulo) : void
+ Update(item : Articulo) : void
+ Delete(id : int) : void
+ FindAll() : IEnumerable<Articulo>
+ FindById(int id : int) : Articulo
RepositorioCliente
+ <<property>> Clientes : List<Cliente>
+ Add(item : Cliente) : void
+ Update(item : Cliente) : void
+ Delete(id : int) : void
+ FindAll() : IEnumerable<Cliente>
+ FindById(int id : int) : Cliente
RepositorioUsu
+ <<property>> Usuarios : L
+ Add(item : Usuario) : void
+ Update(item : Usuario) : void
+ Delete(id : int) : void
+ FindAll() : IEnumerable<Usuario>
+ FindById(int id : int) : Usuario
RepositorioPedido
+ <<property>> Pedidos : List<Pedido>
+ Add(item : Pedido) : void
+ Update(item : Pedido) : void
+ Delete(id : int) : void
+ FindAll() : IEnumerable<Pedido>
+ FindById(int id : int) : Pedido
Papeleria.LogicaNegocio





Scripts SQL:

```
USE [ObligatorioPapeleria]
GO
/****** Object:  Table [dbo].[__EFMigrationsHistory]    Script Date: 16/5/2024 9:36:04 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[__EFMigrationsHistory](
	[MigrationId] [nvarchar](150) NOT NULL,
	[ProductVersion] [nvarchar](32) NOT NULL,
 CONSTRAINT [PK___EFMigrationsHistory] PRIMARY KEY CLUSTERED
(
	[MigrationId] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY =
OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON,
OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Articulos]    Script Date: 16/5/2024 9:36:04 ******/
SET ANSI_NULLS ON
GO
```

```
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Articulos](
	[Id] [int] IDENTITY(1,1) NOT NULL,
	[Nombre_NombreArticuloValor] [nvarchar](max) NOT NULL,
	[Descripcion_DescripcionArticuloValor] [nvarchar](max) NOT NULL,
	[Codigo_CodigoArticuloValor] [bigint] NOT NULL,
	[PrecioVenta] [decimal](18, 2) NOT NULL,
	[Stock] [int] NOT NULL,
 CONSTRAINT [PK_Articulos] PRIMARY KEY CLUSTERED
(
	[Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY =
OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON,
OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Clientes]    Script Date: 16/5/2024 9:36:04 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Clientes](
	[Id] [int] IDENTITY(1,1) NOT NULL,
	[RazonSocial] [nvarchar](max) NOT NULL,
	[Direccion_Calle] [nvarchar](max) NOT NULL,
	[Direccion_Ciudad] [nvarchar](max) NOT NULL,
	[Direccion_Distancia] [int] NOT NULL,
	[Direccion_Numero] [int] NOT NULL,
	[NombreCompleto_Apellido] [nvarchar](max) NOT NULL,
	[NombreCompleto_Nombre] [nvarchar](max) NOT NULL,
	[RUT_NroRut] [nvarchar](max) NOT NULL,
 CONSTRAINT [PK_Clientes] PRIMARY KEY CLUSTERED
(
	[Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY =
OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON,
OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO
/****** Object:  Table [dbo].[LineaPedido]    Script Date: 16/5/2024 9:36:04 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[LineaPedido](
	[ArticuloId] [int] NOT NULL,
	[PedidoId] [int] NOT NULL,
```

```
	[CantidadArticulo] [int] NOT NULL,
	[PrecioUnitario] [decimal](18, 2) NOT NULL,
 CONSTRAINT [PK_LineaPedido] PRIMARY KEY CLUSTERED
(
	[PedidoId] ASC,
	[ArticuloId] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY =
OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON,
OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Pedidos]    Script Date: 16/5/2024 9:36:04 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Pedidos](
	[Id] [int] IDENTITY(1,1) NOT NULL,
	[FechaPrometida] [datetime2](7) NOT NULL,
	[FechaCreado] [datetime2](7) NOT NULL,
	[ClienteId] [int] NOT NULL,
	[Total] [decimal](18, 2) NOT NULL,
	[IVAAplicado] [float] NOT NULL,
	[FechaEntregado] [datetime2](7) NULL,
	[Estado] [int] NOT NULL,
	[Express] [bit] NOT NULL,
 CONSTRAINT [PK_Pedidos] PRIMARY KEY CLUSTERED
(
	[Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY =
OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON,
OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Usuarios]    Script Date: 16/5/2024 9:36:04 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Usuarios](
	[Id] [int] IDENTITY(1,1) NOT NULL,
	[Rol_RolValor] [int] NOT NULL,
	[Contrasenia_Password] [nvarchar](max) NOT NULL,
	[ContraseniaEncriptada_ValorContrasenia] [nvarchar](max) NOT NULL,
	[Email_DireccionEmail] [nvarchar](max) NOT NULL,
	[NombreCompleto_Apellido] [nvarchar](max) NOT NULL,
	[NombreCompleto_Nombre] [nvarchar](max) NOT NULL,
 CONSTRAINT [PK_Usuarios] PRIMARY KEY CLUSTERED
```

```
(
        [Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY =
OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON,
OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO
INSERT [dbo].[__EFMigrationsHistory] ([MigrationId], [ProductVersion]) VALUES
(N'20240429002908_init', N'8.0.4')
GO
INSERT [dbo].[__EFMigrationsHistory] ([MigrationId], [ProductVersion]) VALUES
(N'20240429003152_initial', N'8.0.4')
GO
INSERT [dbo].[__EFMigrationsHistory] ([MigrationId], [ProductVersion]) VALUES
(N'20240430161211_nueva', N'8.0.4')
GO
INSERT [dbo].[__EFMigrationsHistory] ([MigrationId], [ProductVersion]) VALUES
(N'20240509232501_pedidosAdd', N'8.0.4')
GO
INSERT [dbo].[__EFMigrationsHistory] ([MigrationId], [ProductVersion]) VALUES
(N'20240514215027_fixes', N'8.0.4')
GO
INSERT [dbo].[__EFMigrationsHistory] ([MigrationId], [ProductVersion]) VALUES
(N'20240515154550_NuevosDatos', N'8.0.4')
GO
INSERT [dbo].[__EFMigrationsHistory] ([MigrationId], [ProductVersion]) VALUES
(N'20240516005422_aVerSIfunciona', N'8.0.4')
GO
INSERT [dbo].[__EFMigrationsHistory] ([MigrationId], [ProductVersion]) VALUES
(N'20240516105922_nosesifunciona', N'8.0.4')
GO
INSERT [dbo].[__EFMigrationsHistory] ([MigrationId], [ProductVersion]) VALUES
(N'20240516123431_terminamosElOblig', N'8.0.4')
GO
SET IDENTITY_INSERT [dbo].[Articulos] ON
GO
INSERT [dbo].[Articulos] ([Id], [Nombre_NombreArticuloValor],
[Descripcion_DescripcionArticuloValor], [Codigo_CodigoArticuloValor], [PrecioVenta],
[Stock]) VALUES (1, N'Hojas A4', N'Hojas de impresora A4', 8572019346852, CAST(250.00
AS Decimal(18, 2)), 150)
GO
INSERT [dbo].[Articulos] ([Id], [Nombre_NombreArticuloValor],
[Descripcion_DescripcionArticuloValor], [Codigo_CodigoArticuloValor], [PrecioVenta],
[Stock]) VALUES (2, N'Lápices Faber Castell', N'Lápices Faber Castell set 12 colores',
1234567891234, CAST(150.00 AS Decimal(18, 2)), 50)
GO
INSERT [dbo].[Articulos] ([Id], [Nombre_NombreArticuloValor],
[Descripcion_DescripcionArticuloValor], [Codigo_CodigoArticuloValor], [PrecioVenta],
```

```
[Stock]) VALUES (5, N'Lapicera BIC', N'Lapicera BIC azul', 7932481674023, CAST(99.00
AS Decimal(18, 2)), 100)
GO
INSERT [dbo].[Articulos] ([Id], [Nombre_NombreArticuloValor],
[Descripcion_DescripcionArticuloValor], [Codigo_CodigoArticuloValor], [PrecioVenta],
[Stock]) VALUES (6, N'Cuaderno rayado', N'Cuaderno rayado tapa dura', 5127398462058,
CAST(139.00 AS Decimal(18, 2)), 100)
GO
INSERT [dbo].[Articulos] ([Id], [Nombre_NombreArticuloValor],
[Descripcion_DescripcionArticuloValor], [Codigo_CodigoArticuloValor], [PrecioVenta],
[Stock]) VALUES (7, N'Sylvanian Family', N'Sylvanian Family blind box', 9273816405928,
CAST(459.00 AS Decimal(18, 2)), 30)
GO
INSERT [dbo].[Articulos] ([Id], [Nombre_NombreArticuloValor],
[Descripcion_DescripcionArticuloValor], [Codigo_CodigoArticuloValor], [PrecioVenta],
[Stock]) VALUES (8, N'Lápices HB', N'Lápices de grafito estándar', 4927610358274,
CAST(50.00 AS Decimal(18, 2)), 200)
GO
INSERT [dbo].[Articulos] ([Id], [Nombre_NombreArticuloValor],
[Descripcion_DescripcionArticuloValor], [Codigo_CodigoArticuloValor], [PrecioVenta],
[Stock]) VALUES (9, N'Marcadores de colores', N'Juego de marcadores de colores surtidos',
7518296403728, CAST(80.00 AS Decimal(18, 2)), 50)
GO
INSERT [dbo].[Articulos] ([Id], [Nombre_NombreArticuloValor],
[Descripcion_DescripcionArticuloValor], [Codigo_CodigoArticuloValor], [PrecioVenta],
[Stock]) VALUES (10, N'Cinta adhesiva', N'Rollo de cinta adhesiva transparente',
3654928107365, CAST(30.00 AS Decimal(18, 2)), 300)
GO
INSERT [dbo].[Articulos] ([Id], [Nombre_NombreArticuloValor],
[Descripcion_DescripcionArticuloValor], [Codigo_CodigoArticuloValor], [PrecioVenta],
[Stock]) VALUES (11, N'Calculadora básica', N'Calculadora de funciones básicas',
9246738102584, CAST(200.00 AS Decimal(18, 2)), 80)
GO
INSERT [dbo].[Articulos] ([Id], [Nombre_NombreArticuloValor],
[Descripcion_DescripcionArticuloValor], [Codigo_CodigoArticuloValor], [PrecioVenta],
[Stock]) VALUES (12, N'Resaltadores fluorescentes', N'Juego de resaltadores de colores',
8367291047523, CAST(120.00 AS Decimal(18, 2)), 75)
GO
SET IDENTITY_INSERT [dbo].[Articulos] OFF
GO
SET IDENTITY_INSERT [dbo].[Clientes] ON
GO
INSERT [dbo].[Clientes] ([Id], [RazonSocial], [Direccion_Calle], [Direccion_Ciudad],
[Direccion_Distancia], [Direccion_Numero], [NombreCompleto_Apellido],
[NombreCompleto_Nombre], [RUT_NroRut]) VALUES (5, N'Kawaii Gaming', N'Av Rivera',
N'Montevideo', 5, 1234, N'Kim', N'Jiwoo', N'195357789456')
GO
```

```
INSERT [dbo].[Clientes] ([Id], [RazonSocial], [Direccion_Calle], [Direccion_Ciudad], [Direccion_Distancia], [Direccion_Numero], [NombreCompleto_Apellido], [NombreCompleto_Nombre], [RUT_NroRut]) VALUES (6, N'Librería del Este', N'Av Italia', N'Montevideo', 10, 1567, N'Jo', N'Haseul', N'205479630812')
GO
INSERT [dbo].[Clientes] ([Id], [RazonSocial], [Direccion_Calle], [Direccion_Ciudad], [Direccion_Distancia], [Direccion_Numero], [NombreCompleto_Apellido], [NombreCompleto_Nombre], [RUT_NroRut]) VALUES (7, N'Cube Gaming', N'Mataojo', N'Montevideo', 15, 1890, N'Jung', N'Jinsoul', N'198765432109')
GO
INSERT [dbo].[Clientes] ([Id], [RazonSocial], [Direccion_Calle], [Direccion_Ciudad], [Direccion_Distancia], [Direccion_Numero], [NombreCompleto_Apellido], [NombreCompleto_Nombre], [RUT_NroRut]) VALUES (8, N'Imprenta Rápida', N'Comercio', N'Montevideo', 20, 1345, N'Jeon', N'Heejin', N'203040506070')
GO
INSERT [dbo].[Clientes] ([Id], [RazonSocial], [Direccion_Calle], [Direccion_Ciudad], [Direccion_Distancia], [Direccion_Numero], [NombreCompleto_Apellido], [NombreCompleto_Nombre], [RUT_NroRut]) VALUES (9, N'Papel y Tinta', N'Calle 70', N'Canelones', 104, 1678, N'Kim', N'Hyunjin', N'195678234567')
GO
INSERT [dbo].[Clientes] ([Id], [RazonSocial], [Direccion_Calle], [Direccion_Ciudad], [Direccion_Distancia], [Direccion_Numero], [NombreCompleto_Apellido], [NombreCompleto_Nombre], [RUT_NroRut]) VALUES (10, N'Tienda de Papel', N'Av Propios', N'Montevideo', 7, 1901, N'Im', N'Yeojin', N'198765432156')
GO
INSERT [dbo].[Clientes] ([Id], [RazonSocial], [Direccion_Calle], [Direccion_Ciudad], [Direccion_Distancia], [Direccion_Numero], [NombreCompleto_Apellido], [NombreCompleto_Nombre], [RUT_NroRut]) VALUES (11, N'Papel y Lápiz', N'Mar de Ajo', N'Canelones', 106, 1237, N'Park', N'Chaewon', N'205468732910')
GO
INSERT [dbo].[Clientes] ([Id], [RazonSocial], [Direccion_Calle], [Direccion_Ciudad], [Direccion_Distancia], [Direccion_Numero], [NombreCompleto_Apellido], [NombreCompleto_Nombre], [RUT_NroRut]) VALUES (12, N'La Papelería Creativa', N'Gestido', N'Canelones', 99, 1567, N'Wong', N'Gaahei', N'198765432101')
GO
INSERT [dbo].[Clientes] ([Id], [RazonSocial], [Direccion_Calle], [Direccion_Ciudad], [Direccion_Distancia], [Direccion_Numero], [NombreCompleto_Apellido], [NombreCompleto_Nombre], [RUT_NroRut]) VALUES (13, N'Escritura', N'Rambla Costanera', N'Montevideo', 85, 1890, N'Ha', N'Sooyoung', N'201980746352')
GO
INSERT [dbo].[Clientes] ([Id], [RazonSocial], [Direccion_Calle], [Direccion_Ciudad], [Direccion_Distancia], [Direccion_Numero], [NombreCompleto_Apellido], [NombreCompleto_Nombre], [RUT_NroRut]) VALUES (14, N'Arte en Papel', N'Av Loona', N'Montevideo', 120, 1123, N'Choi', N'Verim', N'198765432102')
GO
SET IDENTITY_INSERT [dbo].[Clientes] OFF
GO
```

```
INSERT [dbo].[LineaPedido] ([ArticuloId], [PedidoId], [CantidadArticulo], [PrecioUnitario])
VALUES (1, 12, 1, CAST(250.00 AS Decimal(18, 2)))
GO
INSERT [dbo].[LineaPedido] ([ArticuloId], [PedidoId], [CantidadArticulo], [PrecioUnitario])
VALUES (7, 13, 2, CAST(459.00 AS Decimal(18, 2)))
GO
INSERT [dbo].[LineaPedido] ([ArticuloId], [PedidoId], [CantidadArticulo], [PrecioUnitario])
VALUES (5, 14, 1, CAST(99.00 AS Decimal(18, 2)))
GO
INSERT [dbo].[LineaPedido] ([ArticuloId], [PedidoId], [CantidadArticulo], [PrecioUnitario])
VALUES (7, 15, 1, CAST(459.00 AS Decimal(18, 2)))
GO
INSERT [dbo].[LineaPedido] ([ArticuloId], [PedidoId], [CantidadArticulo], [PrecioUnitario])
VALUES (7, 16, 1, CAST(459.00 AS Decimal(18, 2)))
GO
INSERT [dbo].[LineaPedido] ([ArticuloId], [PedidoId], [CantidadArticulo], [PrecioUnitario])
VALUES (7, 17, 1, CAST(459.00 AS Decimal(18, 2)))
GO
INSERT [dbo].[LineaPedido] ([ArticuloId], [PedidoId], [CantidadArticulo], [PrecioUnitario])
VALUES (7, 18, 2, CAST(459.00 AS Decimal(18, 2)))
GO
INSERT [dbo].[LineaPedido] ([ArticuloId], [PedidoId], [CantidadArticulo], [PrecioUnitario])
VALUES (8, 18, 2, CAST(50.00 AS Decimal(18, 2)))
GO
INSERT [dbo].[LineaPedido] ([ArticuloId], [PedidoId], [CantidadArticulo], [PrecioUnitario])
VALUES (10, 18, 5, CAST(30.00 AS Decimal(18, 2)))
GO
INSERT [dbo].[LineaPedido] ([ArticuloId], [PedidoId], [CantidadArticulo], [PrecioUnitario])
VALUES (2, 19, 1, CAST(150.00 AS Decimal(18, 2)))
GO
INSERT [dbo].[LineaPedido] ([ArticuloId], [PedidoId], [CantidadArticulo], [PrecioUnitario])
VALUES (6, 19, 1, CAST(139.00 AS Decimal(18, 2)))
GO
INSERT [dbo].[LineaPedido] ([ArticuloId], [PedidoId], [CantidadArticulo], [PrecioUnitario])
VALUES (9, 19, 1, CAST(80.00 AS Decimal(18, 2)))
GO
INSERT [dbo].[LineaPedido] ([ArticuloId], [PedidoId], [CantidadArticulo], [PrecioUnitario])
VALUES (1, 20, 5, CAST(250.00 AS Decimal(18, 2)))
GO
INSERT [dbo].[LineaPedido] ([ArticuloId], [PedidoId], [CantidadArticulo], [PrecioUnitario])
VALUES (7, 21, 2, CAST(459.00 AS Decimal(18, 2)))
GO
SET IDENTITY_INSERT [dbo].[Pedidos] ON
GO
INSERT [dbo].[Pedidos] ([Id], [FechaPrometida], [FechaCreado], [ClienteId], [Total],
[IVAAplicado], [FechaEntregado], [Estado], [Express]) VALUES (12,
CAST(N'2024-05-20T22:07:14.8934370' AS DateTime2),
```

```
CAST(N'2024-05-15T22:07:14.8933361' AS DateTime2), 5, CAST(12.30 AS Decimal(18,
2)), 22, CAST(N'2024-05-17T22:07:00.0000000' AS DateTime2), 4, 0)
GO
INSERT [dbo].[Pedidos] ([Id], [FechaPrometida], [FechaCreado], [ClienteId], [Total],
[IVAAplicado], [FechaEntregado], [Estado], [Express]) VALUES (13,
CAST(N'2024-05-20T23:57:45.4045692' AS DateTime2),
CAST(N'2024-05-15T23:57:45.4045666' AS DateTime2), 5, CAST(12.30 AS Decimal(18,
2)), 22, CAST(N'2024-05-19T23:57:00.0000000' AS DateTime2), 1, 0)
GO
INSERT [dbo].[Pedidos] ([Id], [FechaPrometida], [FechaCreado], [ClienteId], [Total],
[IVAAplicado], [FechaEntregado], [Estado], [Express]) VALUES (14,
CAST(N'2024-05-21T00:07:31.5559320' AS DateTime2),
CAST(N'2024-05-16T00:07:31.5559292' AS DateTime2), 8, CAST(0.00 AS Decimal(18, 2)),
22, CAST(N'2024-05-17T00:07:00.0000000' AS DateTime2), 4, 0)
GO
INSERT [dbo].[Pedidos] ([Id], [FechaPrometida], [FechaCreado], [ClienteId], [Total],
[IVAAplicado], [FechaEntregado], [Estado], [Express]) VALUES (15,
CAST(N'2024-05-21T00:17:43.1819026' AS DateTime2),
CAST(N'2024-05-16T00:17:43.1818987' AS DateTime2), 6, CAST(504.90 AS Decimal(18,
2)), 22, CAST(N'2024-05-22T00:17:00.0000000' AS DateTime2), 1, 0)
GO
INSERT [dbo].[Pedidos] ([Id], [FechaPrometida], [FechaCreado], [ClienteId], [Total],
[IVAAplicado], [FechaEntregado], [Estado], [Express]) VALUES (16,
CAST(N'2024-05-21T08:00:29.4172722' AS DateTime2),
CAST(N'2024-05-16T08:00:29.4172704' AS DateTime2), 5, CAST(504.90 AS Decimal(18,
2)), 22, CAST(N'2024-05-16T00:00:00.0000000' AS DateTime2), 1, 1)
GO
INSERT [dbo].[Pedidos] ([Id], [FechaPrometida], [FechaCreado], [ClienteId], [Total],
[IVAAplicado], [FechaEntregado], [Estado], [Express]) VALUES (17,
CAST(N'2024-05-21T08:00:46.7095615' AS DateTime2),
CAST(N'2024-05-16T08:00:46.7095595' AS DateTime2), 6, CAST(459.00 AS Decimal(18,
2)), 22, CAST(N'2024-05-16T00:00:00.0000000' AS DateTime2), 1, 0)
GO
INSERT [dbo].[Pedidos] ([Id], [FechaPrometida], [FechaCreado], [ClienteId], [Total],
[IVAAplicado], [FechaEntregado], [Estado], [Express]) VALUES (18,
CAST(N'2024-05-21T09:29:12.7840415' AS DateTime2),
CAST(N'2024-05-16T09:29:12.7840396' AS DateTime2), 8, CAST(1284.80 AS Decimal(18,
2)), 22, CAST(N'2024-05-16T00:00:00.0000000' AS DateTime2), 1, 1)
GO
INSERT [dbo].[Pedidos] ([Id], [FechaPrometida], [FechaCreado], [ClienteId], [Total],
[IVAAplicado], [FechaEntregado], [Estado], [Express]) VALUES (19,
CAST(N'2024-05-21T09:30:16.0395457' AS DateTime2),
CAST(N'2024-05-16T09:30:16.0395440' AS DateTime2), 10, CAST(369.00 AS Decimal(18,
2)), 22, CAST(N'2024-05-22T00:00:00.0000000' AS DateTime2), 1, 0)
GO
INSERT [dbo].[Pedidos] ([Id], [FechaPrometida], [FechaCreado], [ClienteId], [Total],
[IVAAplicado], [FechaEntregado], [Estado], [Express]) VALUES (20,
CAST(N'2024-05-21T09:30:43.7260048' AS DateTime2),
```

```
CAST(N'2024-05-16T09:30:43.7260032' AS DateTime2), 11, CAST(1312.50 AS Decimal(18,
2)), 22, CAST(N'2024-05-16T00:00:00.0000000' AS DateTime2), 1, 0)
GO
INSERT [dbo].[Pedidos] ([Id], [FechaPrometida], [FechaCreado], [ClienteId], [Total],
[IVAAplicado], [FechaEntregado], [Estado], [Express]) VALUES (21,
CAST(N'2024-05-21T09:30:54.4197304' AS DateTime2),
CAST(N'2024-05-16T09:30:54.4197276' AS DateTime2), 5, CAST(918.00 AS Decimal(18,
2)), 22, CAST(N'2024-05-16T00:00:00.0000000' AS DateTime2), 1, 0)
GO
SET IDENTITY_INSERT [dbo].[Pedidos] OFF
GO
SET IDENTITY_INSERT [dbo].[Usuarios] ON
GO
INSERT [dbo].[Usuarios] ([Id], [Rol_RolValor], [Contrasenia_Password],
[ContraseniaEncriptada_ValorContrasenia], [Email_DireccionEmail],
[NombreCompleto_Apellido], [NombreCompleto_Nombre]) VALUES (1, 2, N'Ariana.1',
N'Ariana.1', N'arianagrande@gmail.com', N'Grande', N'Ariana')
GO
INSERT [dbo].[Usuarios] ([Id], [Rol_RolValor], [Contrasenia_Password],
[ContraseniaEncriptada_ValorContrasenia], [Email_DireccionEmail],
[NombreCompleto_Apellido], [NombreCompleto_Nombre]) VALUES (2, 1, N'Taylor.1',
N'Taylor.1', N'taylorswift@gmail.com', N'Swift', N'Taylor')
GO
INSERT [dbo].[Usuarios] ([Id], [Rol_RolValor], [Contrasenia_Password],
[ContraseniaEncriptada_ValorContrasenia], [Email_DireccionEmail],
[NombreCompleto_Apellido], [NombreCompleto_Nombre]) VALUES (5, 1, N'Lanita!1',
N'Lanita!1', N'lanadelrey@gmail.com', N'Del Rey', N'Lana')
GO
INSERT [dbo].[Usuarios] ([Id], [Rol_RolValor], [Contrasenia_Password],
[ContraseniaEncriptada_ValorContrasenia], [Email_DireccionEmail],
[NombreCompleto_Apellido], [NombreCompleto_Nombre]) VALUES (6, 1, N'XCXWorld1!',
N'XCXWorld1!', N'charlixcx@gmail.com', N'XCX', N'Charli')
GO
INSERT [dbo].[Usuarios] ([Id], [Rol_RolValor], [Contrasenia_Password],
[ContraseniaEncriptada_ValorContrasenia], [Email_DireccionEmail],
[NombreCompleto_Apellido], [NombreCompleto_Nombre]) VALUES (7, 1, N'underWater!1',
N'underWater!1', N'chuu@gmail.com', N'Kim', N'JiWoo')
GO
INSERT [dbo].[Usuarios] ([Id], [Rol_RolValor], [Contrasenia_Password],
[ContraseniaEncriptada_ValorContrasenia], [Email_DireccionEmail],
[NombreCompleto_Apellido], [NombreCompleto_Nombre]) VALUES (8, 1, N'Gowonnie!2',
N'Gowonnie!2', N'gowon@gmail.com', N'Park', N'Chaewon')
GO
INSERT [dbo].[Usuarios] ([Id], [Rol_RolValor], [Contrasenia_Password],
[ContraseniaEncriptada_ValorContrasenia], [Email_DireccionEmail],
[NombreCompleto_Apellido], [NombreCompleto_Nombre]) VALUES (9, 1, N'Twice2!',
N'Twice2!', N'nayeon@gmail.com', N'Im', N'Nayeon')
GO
```

```
INSERT [dbo].[Usuarios] ([Id], [Rol_RolValor], [Contrasenia_Password],
[ContraseniaEncriptada_ValorContrasenia], [Email_DireccionEmail],
[NombreCompleto_Apellido], [NombreCompleto_Nombre]) VALUES (10, 1, N'Momo.1',
N'Momo.1!', N'momo@gmail.com', N'Hirai', N'Momo')
GO
INSERT [dbo].[Usuarios] ([Id], [Rol_RolValor], [Contrasenia_Password],
[ContraseniaEncriptada_ValorContrasenia], [Email_DireccionEmail],
[NombreCompleto_Apellido], [NombreCompleto_Nombre]) VALUES (11, 1, N'Hyeju.1',
N'Hyeju.1', N'hyeju@gmail.com', N'Son', N'Hyeju')
GO
INSERT [dbo].[Usuarios] ([Id], [Rol_RolValor], [Contrasenia_Password],
[ContraseniaEncriptada_ValorContrasenia], [Email_DireccionEmail],
[NombreCompleto_Apellido], [NombreCompleto_Nombre]) VALUES (12, 1, N'Hyunjin.1',
N'Hyunjin.1', N'hyunjin@gmail.com', N'Kim', N'Hyunjin')
GO
INSERT [dbo].[Usuarios] ([Id], [Rol_RolValor], [Contrasenia_Password],
[ContraseniaEncriptada_ValorContrasenia], [Email_DireccionEmail],
[NombreCompleto_Apellido], [NombreCompleto_Nombre]) VALUES (13, 1, N'Charlotte.1',
N'Charlotte.1', N'charlotte@gmail.com', N'Tilbury', N'Charlotte')
GO
SET IDENTITY_INSERT [dbo].[Usuarios] OFF
GO
ALTER TABLE [dbo].[Pedidos] ADD  DEFAULT (CONVERT([bit],(0))) FOR [Express]
GO
ALTER TABLE [dbo].[LineaPedido]  WITH CHECK ADD  CONSTRAINT
[FK_LineaPedido_Articulos_ArticuloId] FOREIGN KEY([ArticuloId])
REFERENCES [dbo].[Articulos] ([Id])
ON DELETE CASCADE
GO
ALTER TABLE [dbo].[LineaPedido] CHECK CONSTRAINT
[FK_LineaPedido_Articulos_ArticuloId]
GO
ALTER TABLE [dbo].[LineaPedido]  WITH CHECK ADD  CONSTRAINT
[FK_LineaPedido_Pedidos_PedidoId] FOREIGN KEY([PedidoId])
REFERENCES [dbo].[Pedidos] ([Id])
ON DELETE CASCADE
GO
ALTER TABLE [dbo].[LineaPedido] CHECK CONSTRAINT
[FK_LineaPedido_Pedidos_PedidoId]
GO
ALTER TABLE [dbo].[Pedidos]  WITH CHECK ADD  CONSTRAINT
[FK_Pedidos_Clientes_ClienteId] FOREIGN KEY([ClienteId])
REFERENCES [dbo].[Clientes] ([Id])
ON DELETE CASCADE
GO
ALTER TABLE [dbo].[Pedidos] CHECK CONSTRAINT [FK_Pedidos_Clientes_ClienteId]
GO
```

Código fuente:

Papelería context

```
using LogicaNegocio.Entidades;
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace AccesoDatos.Implementaciones.EntityFramework
{
    public class PapeleriaContext : DbContext
    {
        public DbSet<Articulo> Articulos { get; set; }
        public DbSet<Cliente> Clientes { get; set; }
        public DbSet<Pedido> Pedidos { get; set; }
        public DbSet<Usuario> Usuarios { get; set; }

        protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
        {
            base.OnConfiguring(optionsBuilder);

optionsBuilder.UseSqlServer(@"SERVER=(localdb)\MsSqlLocalDb;DATABASE=Obligatorio
Papeleria2;Integrated Security=true;");
        }
    }
}
```

Repositorio Artículos

```
using AccesoDatos.Interfaces;
using LogicaNegocio.Entidades;
using LogicaNegocio.Excepciones.Articulos;
using LogicaNegocio.Excepciones.Generales;
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace AccesoDatos.Implementaciones.EntityFramework
{
    public class RepositorioArticulos : IRepositorioArticulos
    {
        private PapeleriaContext _papeleriaContext;

        public RepositorioArticulos()
        {
            _papeleriaContext = new PapeleriaContext();
        }

        #region CRUD Operations
        public void Add(Articulo articuloNuevo)
        {
            try
            {
                _papeleriaContext.Articulos.Add(articuloNuevo);
                _papeleriaContext.SaveChanges();
            }
            catch (ArticuloNoEncontradoException)
            {
                throw;
            }
            catch (DbUpdateException)
            {
                throw;
            }
            catch (Exception ex)
            {
                throw new Exception($"Error desconocido: {ex.Message} (Trace:
{ex.StackTrace})");
            }
```

```
    }

    public Articulo GetById(int id)
    {
        if (!_papeleriaContext.Articulos.Any())
            throw new DataBaseSetException("La tabla Articulos esta vacia");

        try
        {
            Articulo? articuloEncontrado =
_papeleriaContext.Articulos.AsNoTracking().FirstOrDefault(articulo => articulo.Id == id);
            return articuloEncontrado ?? throw new ArticuloNoEncontradoException($"No se
encontro el articulo de Id: {id}");
        }
        catch (ArticuloNoEncontradoException)
        {
            throw;
        }
        catch (Exception ex)
        {
            throw new Exception($"Error desconocido: {ex.Message} (Trace:
{ex.StackTrace})");
        }
    }

    public Articulo RetrieveById(int id)
    {
        if (!_papeleriaContext.Articulos.Any())
            throw new DataBaseSetException("La tabla Articulos esta vacia");

        try
        {
            Articulo? articuloEncontrado = _papeleriaContext.Articulos.FirstOrDefault(articulo
=> articulo.Id == id);
            return articuloEncontrado ?? throw new ArticuloNoEncontradoException($"No se
encontro el articulo de Id: {id}");
        }
        catch (ArticuloNoEncontradoException)
        {
            throw;
        }
        catch (Exception ex)
        {
            throw new Exception($"Error desconocido: {ex.Message} (Trace:
{ex.StackTrace})");
        }
    }
```

```
        public IEnumerable<Articulo> GetAll()
        {
            return _papeleriaContext.Articulos.ToList();
        }

        public void Update(int id, Articulo articuloEditado)
        {
            try
            {
                _papeleriaContext.Articulos.Update(articuloEditado);
                _papeleriaContext.SaveChanges();
            }
            catch (ArticuloNoValidoException)
            {
                throw;
            }
            catch (Exception ex)
            {
                throw new Exception($"Error desconocido: {ex.Message} (Trace:
{ex.StackTrace})");
            }
        }

        public void Remove(int id)
        {
            try
            {
                Articulo articuloParaBorrar = GetById(id);
                _papeleriaContext.Articulos.Remove(articuloParaBorrar);
                _papeleriaContext.SaveChanges();
            }
            catch (ArticuloNoEncontradoException)
            {
                throw;
            }
            catch (Exception ex)
            {
                throw new Exception($"Error desconocido: {ex.Message} (Trace:
{ex.StackTrace})");
            }
        }
        #endregion

        #region DML Methods
     public IEnumerable<Articulo> ListadoAscendente()
{
    return _papeleriaContext.Articulos.OrderBy(articulo =>
articulo.Nombre.NombreArticuloValor).ToList();
```

```
}
```

Repositorio Clientes

```
using AccesoDatos.Interfaces;
using LogicaNegocio.Entidades;
using LogicaNegocio.Excepciones.Clientes;
using LogicaNegocio.Excepciones.Generales;
using LogicaNegocio.ValueObjects;
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace AccesoDatos.Implementaciones.EntityFramework
{
    public class RepositorioClientes : IRepositorioClientes
    {
        private readonly PapeleriaContext _papeleriaContext;

        public RepositorioClientes()
        {
            //Inyeccion de dependencia
            _papeleriaContext = new PapeleriaContext();
        }

        public void Add(Cliente clienteNuevo)
        {
            try
            {
                _papeleriaContext.Clientes.Add(clienteNuevo);
                _papeleriaContext.SaveChanges();
            }
            catch (ClienteNoValidoException)
            {
                throw;
            }
            catch (DbUpdateException)
            {
                throw;
            }
            catch (Exception ex)
            {
```

```
            throw new Exception($"Error desconocido: {ex.Message} (Trace:
{ex.StackTrace})");
        }
    }

    public Cliente GetById(int id)
    {
        if (!_papeleriaContext.Clientes.Any())
            throw new DataBaseSetException("No hay clientes para mostrar, ingrese uno
primero");

        try
        {
            Cliente? clienteEncontrado =
_papeleriaContext.Clientes.AsNoTracking().FirstOrDefault(cliente => cliente.Id == id);
            return clienteEncontrado ?? throw new ClienteNoEncontradoException($"No se
encontro el cliente de ID: {id}");
        }
        catch (ClienteNoEncontradoException)
        {
            throw;
        }
        catch (Exception ex)
        {
            throw new Exception($"Error desconocido: {ex.Message} (Trace:
{ex.StackTrace})");
        }
    }

    public Cliente RetrieveById(int id)
    {
        if (!_papeleriaContext.Clientes.Any())
            throw new DataBaseSetException("No hay clientes para mostrar, ingrese uno
primero");

        try
        {
            Cliente? clienteEncontrado = _papeleriaContext.Clientes.FirstOrDefault(cliente =>
cliente.Id == id);
            return clienteEncontrado ?? throw new ClienteNoEncontradoException($"No se
encontro el cliente de ID: {id}");
        }
        catch (ClienteNoEncontradoException)
        {
            throw;
        }
        catch (Exception ex)
        {
```

```
            throw new Exception($"Error desconocido: {ex.Message} (Trace:
{ex.StackTrace})");
        }
    }

    public IEnumerable<Cliente> GetAll()
    {
        return _papeleriaContext.Clientes.ToList();
    }

    public void Update(int id, Cliente clienteEditado)
    {
        try
        {
            _papeleriaContext.Clientes.Update(clienteEditado);
            _papeleriaContext.SaveChanges();
        }
        catch (ClienteNoValidoException)
        {
            throw;
        }
        catch (DbUpdateException)
        {
            throw;
        }
        catch (Exception ex)
        {
            throw new Exception($"Error desconocido: {ex.Message} (Trace:
{ex.StackTrace})");
        }
    }

    public void Remove(int id)
    {
        try
        {
            Cliente clienteParaBorrar = GetById(id);
            _papeleriaContext.Clientes.Remove(clienteParaBorrar);
            _papeleriaContext.SaveChanges(true);
        }
        catch (ClienteNoEncontradoException)
        {
            throw;
        }
        catch (Exception ex)
        {
            throw new Exception($"Error desconocido: {ex.Message} (Trace:
{ex.StackTrace})");
```

```
        }
    }

    #region DML Methods

    public IEnumerable<Cliente> GetByText(string texto)
    {
        if (!_papeleriaContext.Clientes.Any())
            throw new DataBaseSetException("No hay clientes para mostrar, ingrese uno
primero");

        IEnumerable<Cliente> clientes = _papeleriaContext.Clientes
            .Where(cli => cli.NombreCompleto.Nombre.Contains(texto) ||
cli.NombreCompleto.Apellido.Contains(texto))
            .ToList();
        //TODO tira errorcillo
        return clientes.Any() ? clientes : throw new ClienteNoEncontradoException($"No se
pudo encontrar ningun cliente que contenga {texto} en su nombre o apellido");
    }

    public Cliente GetByRut(string rut)
    {
        if (!_papeleriaContext.Clientes.Any())
            throw new DataBaseSetException("No hay clientes para mostrar, ingrese uno
primero");

        try
        {
            Cliente? clienteEncontrado = _papeleriaContext.Clientes.FirstOrDefault(cliente =>
cliente.RUT.NroRut == rut);
            return clienteEncontrado ?? throw new ClienteNoEncontradoException($"No se
pudo encontrar al cliente de RUT {rut}");
        }
        catch (ClienteNoEncontradoException)
        {
            throw;
        }
        catch (Exception ex)
        {
            throw new Exception($"Error desconocido: {ex.Message} (Trace:
{ex.StackTrace})");
        }

    }

    public IEnumerable<Cliente> GetByMonto(decimal monto)
    {
        IEnumerable<Cliente> clientes = _papeleriaContext.Pedidos
```

```
            .Include(pedido => pedido.Cliente)
            .Where(pedido => pedido.Total > monto)
            .Select(pedido => pedido.Cliente)
            .Distinct()
            .ToList();

        return clientes.Any() ? clientes : throw new ClienteNoEncontradoException($"No se pudo encontrar ningun cliente que con un total de pedidos superior a ${monto}");

    }
    #endregion
  }
}
```

Repositorio Pedidos

```
using AccesoDatos.Interfaces;
using LogicaNegocio.Entidades;
using LogicaNegocio.Enumerados;
using LogicaNegocio.Excepciones.Generales;
using LogicaNegocio.Excepciones.Pedidos;
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace AccesoDatos.Implementaciones.EntityFramework
{
  public class RepositorioPedidos : IRepositorioPedidos
  {
    private readonly PapeleriaContext _papeleriaContext;

    public RepositorioPedidos()
    {
      _papeleriaContext = new PapeleriaContext();
    }

    #region CRUD Operations
    public void Add(Pedido pedidoNuevo)
    {
      try
      {
        //TODO: check this with same article twice
        _papeleriaContext.Entry(pedidoNuevo.Cliente).State = EntityState.Unchanged;
```

```
            foreach(var item in pedidoNuevo.Lineas)
            {
                _papeleriaContext.Entry(item.Articulo).State = EntityState.Unchanged;
            }
            _papeleriaContext.Pedidos.Add(pedidoNuevo);
            _papeleriaContext.SaveChanges();
        }
        catch (PedidoNoValidoException)
        {
            throw;
        }
        catch (Exception ex)
        {
            throw new Exception($"Error desconocido: {ex.Message}");
        }
    }

    public Pedido GetById(int id)
    {
        try
        {
            Pedido? pedidoEncontrado = _papeleriaContext.Pedidos
                .Include(p => p.Lineas)
                .Include(p => p.Cliente)
                .AsNoTracking()
                .FirstOrDefault(pedido => pedido.Id == id);
            return pedidoEncontrado ?? throw new PedidoNoEncontradoException($"No se
encontro el pedido de ID: {id}");
        }
        catch (Exception ex)
        {
            throw new Exception($"Error desconocido: {ex.Message}");
        }
    }

    public Pedido RetrieveById(int id)
    {
        try
        {
            Pedido? pedidoEncontrado = _papeleriaContext.Pedidos
                .Include(p => p.Lineas)
                .Include(p => p.Cliente)
                .FirstOrDefault(pedido => pedido.Id == id);
            return pedidoEncontrado ?? throw new PedidoNoEncontradoException($"No se
encontro el pedido de ID: {id}");
        }
        catch (Exception ex)
        {
```

```
            throw new Exception($"Error desconocido: {ex.Message}");
        }
    }

    public IEnumerable<Pedido> GetAll()
    {
        return _papeleriaContext.Pedidos.Include(p=>p.Lineas).Include(p =>
p.Cliente).ToList();
    }

    public void Update(int id, Pedido pedidoEditado)
    {
        /*try
        {
            _papeleriaContext.Pedidos.Update(pedidoEditado);
            _papeleriaContext.SaveChanges();
        }
        catch (PedidoNoValidoException)
        {
            throw;
        }
        catch (Exception ex)
        {
            throw new Exception($"Error desconocido: {ex.Message} (Trace:
{ex.StackTrace})");
        }*/

    }

    public void Remove(int id)
    {
        throw new NotImplementedException();
    }

    public void AnularPedido(int id)
    {
        Pedido? pedidoEncontrado = GetById(id);

        try
        {
            pedidoEncontrado.Estado = EEstado.ANULADO;
            _papeleriaContext.Pedidos.Update(pedidoEncontrado);
            _papeleriaContext.SaveChanges();
        }
        catch (PedidoNoEncontradoException)
        {
            throw;
        }
```

```
            catch (Exception ex)
            {
                throw new Exception($"Error desconocido: {ex.Message}");
            }
        }

        public IEnumerable<Pedido> ListadoDescendente()
{
    public IEnumerable<Pedido> ListadoDescendente()
{
    return _papeleriaContext.Pedidos.Where(p => p.Estado == EEstado.ANULADO)
        .Include(p => p.Cliente)
        .Include(p => p.Lineas)
        .OrderByDescending(p => p.FechaCreado)
        .ToList();
}

}
}
```

Repositorio Usuarios

```
using AccesoDatos.Interfaces;
using LogicaNegocio.Entidades;
using LogicaNegocio.Excepciones.Generales;
using LogicaNegocio.Excepciones.Usuarios;
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace AccesoDatos.Implementaciones.EntityFramework
{
    public class RepositorioUsuarios : IRepositorioUsuarios
    {
        private PapeleriaContext _papeleriaContext;

        public RepositorioUsuarios()
        {
            //Inyeccion de dependencia
            _papeleriaContext = new PapeleriaContext();
        }
```

```
        #region CRUD Operations
        public Usuario GetById(int id)
        {
            Usuario? usuarioEncontrado =
_papeleriaContext.Usuarios.AsNoTracking().FirstOrDefault(usuario => usuario.Id == id);

            return usuarioEncontrado ?? throw new UsuarioNoEncontradoException($"No se
pudo encontrar el usuario de ID: {id}");
        }

        public Usuario RetrieveById(int id)
        {

            Usuario? usuarioEncontrado =
_papeleriaContext.Usuarios.AsNoTracking().FirstOrDefault(usuario => usuario.Id == id);

            return usuarioEncontrado ?? throw new UsuarioNoEncontradoException($"No se
pudo encontrar el usuario de ID: {id}");
        }

        public IEnumerable<Usuario> GetAll()
        {
            return _papeleriaContext.Usuarios.ToList();
        }

        public void Add(Usuario usuarioNuevo)
        {
            try
            {
                _papeleriaContext.Usuarios.Add(usuarioNuevo);
                _papeleriaContext.SaveChanges();
            }
            catch (DbUpdateException)
            {
                throw;
            }
            catch (Exception ex)
            {
                throw new Exception($"Error desconocido: {ex.Message} (Trace:
{ex.StackTrace})");
            }
        }

        public void Update(int id, Usuario usuarioEditado)
        {
            try
            {
```

```
            _papeleriaContext.Usuarios.Update(usuarioEditado);
            _papeleriaContext.SaveChanges();
        }
        catch (UsuarioNoValidoException)
        {
            throw;
        }
        catch (Exception ex)
        {
            throw new Exception($"Error desconocido: {ex.Message} (Trace:
{ex.StackTrace})");
        }
    }

    public void Remove(int id)
    {
        try
        {
            Usuario aBorrar = GetById(id);
            _papeleriaContext.Usuarios.Remove(aBorrar);
            _papeleriaContext.SaveChanges();
        }
        catch (UsuarioNoEncontradoException)
        {
            throw;
        }
        catch (Exception)
        {
            throw;
        }
    }

    #endregion

    #region DML Methods
    public Usuario ObtenerUsuarioPorEmail(string email)
    {
        Usuario? usuarioEncontrado = _papeleriaContext.Usuarios.FirstOrDefault(usuario =>
usuario.Email.DireccionEmail == email);
        return usuarioEncontrado ?? throw new
UsuarioNoEncontradoException("Credenciales incorrectas. Revise el mail y/o la
contraseña");
    }
    #endregion
  }
}
```

IRepositorio Articulos

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

namespace AccesoDatos.Interfaces
{
    public interface IRepositorioArticulos : IRepositorioCRUD<Articulo>
    {
        public IEnumerable<Articulo> ListadoAscendente();
    }
}
```

IRepositorio Clientes

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

namespace AccesoDatos.Interfaces
{
    public interface IRepositorioClientes : IRepositorioCRUD<Cliente>
    {
        public IEnumerable<Cliente> GetByText(string texto);
        public IEnumerable<Cliente> GetByMonto(decimal monto);
        public Cliente GetByRut(string monto);
    }
}
```

IRepositorio CRUD

```
using System;
using System.Collections.Generic;
using System.Linq;
```

```
using System.Text;
using System.Threading.Tasks;

namespace AccesoDatos.Interfaces
{
    public interface IRepositorioCRUD<T> where T : class
    {
        void Add(T obj);
        T GetById(int id);
        T RetrieveById(int id);
        IEnumerable<T> GetAll();
        void Update(int id, T obj);
        void Remove(int id);
    }
}
```

IRepositorio Pedidos

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

namespace AccesoDatos.Interfaces
{
    public interface IRepositorioPedidos : IRepositorioCRUD<Pedido>
    {
        public void AnularPedido(int id);
        public IEnumerable<Pedido> ListadoDescendente();
    }
}
```

IRepositorio Usuarios

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

```
namespace AccesoDatos.Interfaces
{
    public interface IRepositorioUsuarios : IRepositorioCRUD<Usuario>
    {
        public Usuario ObtenerUsuarioPorEmail(string email);
    }
}
```

Casos Uso Artículos Implementaciones e interfaces

```
using AccesoDatos.Implementaciones.EntityFramework;
using AccesoDatos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoArticulos.Interfaces;
using LogicaAplicacion.DataTransferObjects.Mappers;
using LogicaAplicacion.DataTransferObjects.Models.Articulos;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoArticulos.Implementaciones
{
    public class CasoUsoAltaArticulo : ICasoUsoAltaArticulo
    {
        public IRepositorioArticulos RepositorioArticulos { get; init; }

        public CasoUsoAltaArticulo(IRepositorioArticulos repositorioArticulos)
        {
            //Inyeccion de dependencia
            RepositorioArticulos = repositorioArticulos;
        }

        public void AltaArticulo(ArticulosDTO articuloNuevoDTO)
```

```
        {
            if (articuloNuevoDTO == null)
            {
                throw new ArgumentNullException(nameof(articuloNuevoDTO));
            }
            Articulo srticuloNuevo = MapperArticulo.FromDTO(articuloNuevoDTO);
            RepositorioArticulos.Add(srticuloNuevo);
        }
    }
}

using AccesoDatos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoArticulos.Interfaces;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoArticulos.Implementaciones
{
    public class CasoUsoBajaArticulo(IRepositorioArticulos repositorioArticulos) :
ICasoUsoBajaArticulo
    {
        public IRepositorioArticulos RepositorioArticulos { get; set; } = repositorioArticulos;

        public void BajaArticulo(int id)
        {
            this.RepositorioArticulos.Remove(id);
        }
    }
}

using AccesoDatos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoArticulos.Interfaces;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoArticulos.Implementaciones
{
    public class CasoUsoBuscarArticulo : ICasoUsoBuscarArticulo
    {
        public IRepositorioArticulos RepositorioArticulos { get; set; }
```

```
        public CasoUsoBuscarArticulo(IRepositorioArticulos repositorioArticulos)
        {
            RepositorioArticulos = repositorioArticulos;
        }

        public Articulo BuscarArticulo(int id)
        {
            return this.RepositorioArticulos.GetById(id);
        }
    }
}

using AccesoDatos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoArticulos.Interfaces;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoArticulos.Implementaciones
{
    public class CasoUsoEditarArticulo(IRepositorioArticulos repositorioArticulos) :
ICasoUsoEditarArticulo
    {
        public IRepositorioArticulos RepositorioArticulos { get; set; } = repositorioArticulos;

        public void EditarArticulo(int idArticulo,Articulo articulo)
        {
            this.RepositorioArticulos.Update(idArticulo,articulo);
        }
    }
}

using AccesoDatos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoArticulos.Interfaces;
using LogicaAplicacion.DataTransferObjects.Mappers;
using LogicaAplicacion.DataTransferObjects.Models.Articulos;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoArticulos.Implementaciones
{
```

```
    public class CasoUsoListarArticulos : ICasoUsoListarArticulos
    {
        public IRepositorioArticulos RepositorioArticulos { get; set; }

        public CasoUsoListarArticulos(IRepositorioArticulos repositorioArticulos)
        {
            RepositorioArticulos = repositorioArticulos;
        }

        public IEnumerable<ArticulosListadoDTO> ListarArticulos()
        {
            return MapperArticulo.FromList(RepositorioArticulos.GetAll());
        }
    }
}
using AccesoDatos.Implementaciones.EntityFramework;
using AccesoDatos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoArticulos.Interfaces;
using LogicaAplicacion.DataTransferObjects.Mappers;
using LogicaAplicacion.DataTransferObjects.Models.Articulos;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoArticulos.Implementaciones
{
    public class CasoUsoListarOrdenadoAlfabeticamenteAscendenteArticulo :
ICasoUsoListarOrdenadoAlfabeticamenteAscendenteArticulo
    {
        public IRepositorioArticulos _repositorioArticulos;

        public
CasoUsoListarOrdenadoAlfabeticamenteAscendenteArticulo(IRepositorioArticulos
repositorioArticulos)
        {
            _repositorioArticulos = repositorioArticulos;
        }

        public IEnumerable<ArticulosListadoDTO>
ListarArticulosOrdenadoAlfabeticamenteAscendente()
        {
            return MapperArticulo.FromList(_repositorioArticulos.ListadoAscendente());
        }
    }
}
```

```
using LogicaAplicacion.DataTransferObjects.Models.Articulos;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoArticulos.Interfaces
{
    public interface ICasoUsoAltaArticulo
    {
        public void AltaArticulo(ArticulosDTO articuloNuevo);
    }
}
```

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

namespace LogicaAplicacion.CasosUso.CasosUsoArticulos.Interfaces
{
    public interface ICasoUsoBajaArticulo
    {
        public void BajaArticulo(int id);
    }
}
```

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

namespace LogicaAplicacion.CasosUso.CasosUsoArticulos.Interfaces
{
    public interface ICasoUsoBuscarArticulo
    {
        public Articulo BuscarArticulo(int id);
    }
}
```

```
using LogicaNegocio.Entidades;
using System;
```

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

namespace LogicaAplicacion.CasosUso.CasosUsoArticulos.Interfaces
{
    public interface ICasoUsoEditarArticulo
    {
        public void EditarArticulo(int idArticulo,Articulo articulo);
    }
}

using LogicaAplicacion.DataTransferObjects.Models.Articulos;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoArticulos.Interfaces
{
    public interface ICasoUsoListarArticulos
    {
        public IEnumerable<ArticulosListadoDTO> ListarArticulos();
    }
}

using LogicaAplicacion.DataTransferObjects.Models.Articulos;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoArticulos.Interfaces
{
    public interface ICasoUsoListarOrdenadoAlfabeticamenteAscendenteArticulo
    {
        public IEnumerable<ArticulosListadoDTO>
ListarArticulosOrdenadoAlfabeticamenteAscendente();
    }
}
```

Casos Uso Clientes Implementaciones e interfaces

```
using AccesoDatos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoClientes.Interfaces;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoClientes.Implementaciones
{
    public class CasoUsoAltaCliente : ICasoUsoAltaCliente
    {
        public IRepositorioClientes RepositorioClientes { get; init; }

        public CasoUsoAltaCliente(IRepositorioClientes repositorioClientes)
        {
            // Inyeccion de dependencia
            RepositorioClientes = repositorioClientes;
        }

        public void AltaCliente(Cliente cliente)
        {
            this.RepositorioClientes.Add(cliente);
        }

    }
}
using AccesoDatos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoClientes.Interfaces;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoClientes.Implementaciones
{
    public class CasoUsoBajaCliente : ICasoUsoBajaCliente
```

```
    {
        public IRepositorioClientes RepositorioClientes { get; set; }

        public CasoUsoBajaCliente(IRepositorioClientes repositorioClientes)
        {
            // Inyeccion de dependencia
            RepositorioClientes = repositorioClientes;

        }

        public void BajaCliente(int id)
        {
            this.RepositorioClientes.Remove(id);
        }

    }
}
using AccesoDatos.Implementaciones.EntityFramework;
using AccesoDatos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoClientes.Interfaces;
using LogicaAplicacion.DataTransferObjects.Mappers;
using LogicaAplicacion.DataTransferObjects.Models.Clientes;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using static System.Net.Mime.MediaTypeNames;

namespace LogicaAplicacion.CasosUso.CasosUsoClientes.Implementaciones
{
    public class CasoUsoBuscarCliente : ICasoUsoBuscarCliente
    {
        public IRepositorioClientes RepositorioClientes { get; set; }

        public CasoUsoBuscarCliente(IRepositorioClientes repositorioClientes)
        {
            // Inyeccion de dependencia
            RepositorioClientes = repositorioClientes;
        }

        public Cliente BuscarClientePorId(int id)
        {
            return RepositorioClientes.GetById(id);
        }

        public Cliente BuscarClientePorRut(string rut)
```

```
        {
            return RepositorioClientes.GetByRut(rut);
        }


        public IEnumerable<ClienteDTO> BuscarClientePorTexto(string texto)
        {
            return MapperCliente.FromList(RepositorioClientes.GetByText(texto));
        }

        public IEnumerable<ClienteDTO> BuscarClientePorMonto(decimal monto)
        {
            return MapperCliente.FromList(RepositorioClientes.GetByMonto(monto));
        }
    }
}
```

```
using AccesoDatos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoClientes.Interfaces;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoClientes.Implementaciones
{
    public class CasoUsoEditarCliente : ICasoUsoEditarCliente
    {
        public IRepositorioClientes RepositorioClientes { get; init; }

        public CasoUsoEditarCliente(IRepositorioClientes repositorioClientes)
        {
            // Inyeccion de dependencia
            RepositorioClientes = repositorioClientes;
        }

        public void EditarCliente(int idCliente,Cliente cliente)
        {
            RepositorioClientes.Update(idCliente,cliente);
        }

    }
}
```

```
using AccesoDatos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoClientes.Interfaces;
```

```
using LogicaAplicacion.DataTransferObjects.Mappers;
using LogicaAplicacion.DataTransferObjects.Models.Clientes;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoClientes.Implementaciones
{
    public class CasoUsoListarClientes : ICasoUsoListarClientes
    {
        public IRepositorioClientes RepositorioClientes { get; init; }

        public CasoUsoListarClientes(IRepositorioClientes repositorioClientes)
        {
            // Inyeccion de dependencia
            RepositorioClientes = repositorioClientes;

        }

        public IEnumerable<ClienteDTO> ListarClientes()
        {
            return MapperCliente.FromList(RepositorioClientes.GetAll());
        }
    }
}

using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoClientes.Interfaces
{
    public interface ICasoUsoAltaCliente
    {
        public void AltaCliente(Cliente cliente);

    }
}

using System;
using System.Collections.Generic;
using System.Linq;
```

```
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoClientes.Interfaces
{
    public interface ICasoUsoBajaCliente
    {
        public void BajaCliente(int id);

    }
}

using LogicaAplicacion.DataTransferObjects.Models.Clientes;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoClientes.Interfaces
{
    public interface ICasoUsoBuscarCliente
    {
        public Cliente BuscarClientePorId(int id);
        public Cliente BuscarClientePorRut(string rut);
        public IEnumerable<ClienteDTO> BuscarClientePorTexto(string texto);
        public IEnumerable<ClienteDTO> BuscarClientePorMonto(decimal monto);

    }
}

using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoClientes.Interfaces
{
    public interface ICasoUsoEditarCliente
    {
        public void EditarCliente(int idCliente, Cliente clienteEditado);

    }
}
```

```
using LogicaAplicacion.DataTransferObjects.Models.Clientes;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoClientes.Interfaces
{
    public interface ICasoUsoListarClientes
    {
        public IEnumerable<ClienteDTO> ListarClientes();

    }
}
```

Caso Uso Login Implementacion e interfaz

```
using AccesoDatos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoLogin.Interfaces;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoLogin.Implementaciones
{
    public class CasoUsoLoginUsuario : ICasoUsoLoginUsuario
    {
        public IRepositorioUsuarios RepositorioUsuarios { get; init; }

        public CasoUsoLoginUsuario(IRepositorioUsuarios repositorioUsuario)
        {
            RepositorioUsuarios = repositorioUsuario;
        }
        public Usuario Ejecutar(string email, string contraseña)
        {
            Usuario buscado = RepositorioUsuarios.ObtenerUsuarioPorEmail(email);

            return buscado.Contrasenia.Password == contraseña ? buscado : null;
        }
```

```
    }
}
```

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

namespace LogicaAplicacion.CasosUso.CasosUsoLogin.Interfaces
{
    public interface ICasoUsoLoginUsuario
    {
        public Usuario Ejecutar(string email, string contraseña);
    }
}
```

Casos Uso Pedidos Implementaciones e interfaces

```
using AccesoDatos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoPedidos.Interfaces;
using LogicaAplicacion.DataTransferObjects.Mappers;
using LogicaAplicacion.DataTransferObjects.Models.Pedidos;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoPedidos.Implementaciones
{
    public class CasoUsoAltaPedido : ICasoUsoAltaPedido
    {
        public IRepositorioPedidos RepositorioPedidos { get; init; }

        public CasoUsoAltaPedido(IRepositorioPedidos repositorioPedidos)
        {
            // Inyeccion de dependencia
            this.RepositorioPedidos = repositorioPedidos;
        }
```

```
        public void AltaPedido(PedidoDTO pedido)
        {
            this.RepositorioPedidos.Add(MapperPedido.FromDTO(pedido));
        }

    }
}
```

```
using AccesoDatos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoPedidos.Interfaces;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoPedidos.Implementaciones
{
    public class CasoUsoBajaPedido : ICasoUsoBajaPedido
    {
        public IRepositorioPedidos RepositorioPedidos { get; set; }

        public CasoUsoBajaPedido(IRepositorioPedidos repositorioPedidos)
        {
            // Inyeccion de dependencia
            this.RepositorioPedidos = repositorioPedidos;
        }

        public void BajaPedido(int id)
        {
            this.RepositorioPedidos.AnularPedido(id);
        }

    }
}
```

```
using AccesoDatos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoPedidos.Interfaces;
using LogicaAplicacion.DataTransferObjects.Mappers;
using LogicaAplicacion.DataTransferObjects.Models.Pedidos;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
```

```
namespace LogicaAplicacion.CasosUso.CasosUsoPedidos.Implementaciones
{
    public class CasoUsoBuscarPedido : ICasoUsoBuscarPedido
    {
        public IRepositorioPedidos RepositorioPedidos { get; init; }

        public CasoUsoBuscarPedido(IRepositorioPedidos repositorioPedidos)
        {
            // Inyeccion de dependencia
            RepositorioPedidos = repositorioPedidos;
        }

        public PedidoDTO BuscarPedido(int id)
        {
            return MapperPedido.ToDTO(RepositorioPedidos.GetById(id));
        }

    }
}

using AccesoDatos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoPedidos.Interfaces;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoPedidos.Implementaciones
{
    public class CasoUsoEditarPedido : ICasoUsoEditarPedido
    {
        public IRepositorioPedidos RepositorioPedidos { get; init; }

        public CasoUsoEditarPedido(IRepositorioPedidos repositorioPedidos)
        {
            // Inyeccion de dependencia
            this.RepositorioPedidos = repositorioPedidos;
        }

        public void EditarPedido(int idPedido,Pedido pedido)
        {
            this.RepositorioPedidos.Update(idPedido,pedido);
        }

    }
}
```

```
using AccesoDatos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoPedidos.Interfaces;
using LogicaAplicacion.DataTransferObjects.Mappers;
using LogicaAplicacion.DataTransferObjects.Models.Pedidos;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoPedidos.Implementaciones
{
    public class CasoUsoListarOrdenadoDescendentementePorFechaPedido :
ICasoUsoListarOrdenadoDescendentementePorFechaPedido
    {
        public IRepositorioPedidos _repositorioPedidos;

        public
CasoUsoListarOrdenadoDescendentementePorFechaPedido(IRepositorioPedidos
repositorioPedidos)
        {
            _repositorioPedidos = repositorioPedidos;
        }
        public IEnumerable<PedidoDTO>
ListarPedidoOrdenadoDescendentementePorFecha()
        {
            return MapperPedido.ToListAll(_repositorioPedidos.ListadoDescendente());
        }
    }
}
using AccesoDatos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoPedidos.Interfaces;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoPedidos.Implementaciones
{
    public class CasoUsoListarPedido : ICasoUsoListarPedido
    {
        public IRepositorioPedidos RepositorioPedidos { get; init; }

        public CasoUsoListarPedido(IRepositorioPedidos repositorioPedidos)
        {
            /// Inyeccion de dependencia
```

```
            RepositorioPedidos = repositorioPedidos;
        }

        public IEnumerable<Pedido> ListarPedidos()
        {
            return this.RepositorioPedidos.GetAll();
        }

    }
}
using LogicaAplicacion.DataTransferObjects.Models.Pedidos;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoPedidos.Interfaces
{
    public interface ICasoUsoAltaPedido
    {
        public void AltaPedido(PedidoDTO pedido);

    }
}

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

namespace LogicaAplicacion.CasosUso.CasosUsoPedidos.Interfaces
{
    public interface ICasoUsoBajaPedido
    {
        public void BajaPedido(int id);

    }
}

using LogicaAplicacion.DataTransferObjects.Models.Pedidos;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
```

```
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoPedidos.Interfaces
{
    public interface ICasoUsoBuscarPedido
    {
        public PedidoDTO BuscarPedido(int id);

    }
}

using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoPedidos.Interfaces
{
    public interface ICasoUsoEditarPedido
    {
        public void EditarPedido(int idPedido,Pedido pedido);

    }
}

using LogicaAplicacion.DataTransferObjects.Models.Pedidos;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoPedidos.Interfaces
{
    public interface ICasoUsoListarOrdenadoDescendentementePorFechaPedido
    {
        public IEnumerable<PedidoDTO>
ListarPedidoOrdenadoDescendentementePorFecha();
    }
}

using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
```

```
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoPedidos.Interfaces
{
    public interface ICasoUsoListarPedido
    {
        public IEnumerable<Pedido> ListarPedidos();

    }
}
```

Casos Uso Usuarios Implementaciones e interfaces

```
using AccesoDatos.Implementaciones.EntityFramework;
using AccesoDatos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoUsuarios.Interfaces;
using LogicaAplicacion.DataTransferObjects.Mappers;
using LogicaAplicacion.DataTransferObjects.Models.Usuarios;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoUsuarios.Implementaciones
{
    public class CasoUsoAltaUsuario : ICasoUsoAltaUsuario
    {
        public IRepositorioUsuarios RepositorioUsuarios { get; init; }

        public CasoUsoAltaUsuario(IRepositorioUsuarios repositorioUsuarios)
        {
            // Inyeccion de dependencia
            RepositorioUsuarios = repositorioUsuarios;
        }

        public void AltaUsuario(UsuarioAltaDTO usuarioNuevo)
        {
            if (usuarioNuevo == null)
            {
                throw new ArgumentNullException(nameof(usuarioNuevo));
            }
```

```
            Usuario user = MapperUsuario.FromDTO(usuarioNuevo);
            RepositorioUsuarios.Add(user);
        }

    }
}

using AccesoDatos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoUsuarios.Interfaces;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoUsuarios.Implementaciones
{
    public class CasoUsoBajaUsuario : ICasoUsoBajaUsuario
    {
        public IRepositorioUsuarios RepositorioUsuarios { get; init; }

        public CasoUsoBajaUsuario(IRepositorioUsuarios repositorioUsuarios)
        {
            // Inyeccion de dependencia
            RepositorioUsuarios = repositorioUsuarios;
        }

        public void BajaUsuario(int id)
        {
            RepositorioUsuarios.Remove(id);
        }

    }
}
using AccesoDatos.Implementaciones.EntityFramework;
using AccesoDatos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoUsuarios.Interfaces;
using LogicaAplicacion.DataTransferObjects.Mappers;
using LogicaAplicacion.DataTransferObjects.Models.Usuarios;
using LogicaNegocio.Entidades;
using LogicaNegocio.Excepciones.Usuarios;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoUsuarios.Implementaciones
```

```
{
    public class CasoUsoBuscarUsuario : ICasoUsoBuscarUsuario
    {
        public IRepositorioUsuarios RepositorioUsuarios { get; init; }

        public CasoUsoBuscarUsuario(IRepositorioUsuarios repositorioUsuarios)
        {
            // Inyeccion de dependencia
            RepositorioUsuarios = repositorioUsuarios;
        }

        public UsuarioListadoDTO BuscarUsuario(int id)
        {
            Usuario user = RepositorioUsuarios.GetById(id) ?? throw new
UsuarioNoEncontradoException("No hay usuarios con ese id");

            UsuarioListadoDTO userDTO = MapperUsuario.ToDTO(user);

            return userDTO;
        }

    }
}

using AccesoDatos.Implementaciones.EntityFramework;
using AccesoDatos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoUsuarios.Interfaces;
using LogicaAplicacion.DataTransferObjects.Mappers;
using LogicaAplicacion.DataTransferObjects.Models.Usuarios;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoUsuarios.Implementaciones
{
    public class CasoUsoEditarUsuario : ICasoUsoEditarUsuario
    {
        public IRepositorioUsuarios RepositorioUsuarios { get; init; }

        public CasoUsoEditarUsuario(IRepositorioUsuarios repositorioUsuarios)
        {
            // Inyeccion de dependencia
            RepositorioUsuarios = repositorioUsuarios;
        }
```

```
        public void EditarUsuario(int idUsuario,UsuarioModificacionDTO
usuarioModificadoDTO)
        {
            Usuario buscar = RepositorioUsuarios.GetById(idUsuario);

            //No se puede modificar ni el rol ni el email
            usuarioModificadoDTO.Rol = buscar.Rol.RolValor;
            usuarioModificadoDTO.Email = buscar.Email.DireccionEmail;


            Usuario usuarioModificado = MapperUsuario.FromDTO(usuarioModificadoDTO);
            RepositorioUsuarios.Update(idUsuario, usuarioModificado);
        }

    }
}

using AccesoDatos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoUsuarios.Interfaces;
using LogicaAplicacion.DataTransferObjects.Mappers;
using LogicaAplicacion.DataTransferObjects.Models.Usuarios;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoUsuarios.Implementaciones
{
    public class CasoUsoListarUsuario : ICasoUsoListarUsuario
    {
        public IRepositorioUsuarios RepositorioUsuarios { get; init; }

        public CasoUsoListarUsuario(IRepositorioUsuarios repositorioUsuarios)
        {
            // Inyeccion de dependencia
            RepositorioUsuarios = repositorioUsuarios;
        }

        public IEnumerable<UsuarioListadoDTO> ListarUsuarios()
        {
            return MapperUsuario.FromList(RepositorioUsuarios.GetAll());
        }

    }
}
```

```
using LogicaAplicacion.DataTransferObjects.Models.Usuarios;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoUsuarios.Interfaces
{
    public interface ICasoUsoAltaUsuario
    {
        public void AltaUsuario(UsuarioAltaDTO usuario);
    }
}

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

namespace LogicaAplicacion.CasosUso.CasosUsoUsuarios.Interfaces
{
    public interface ICasoUsoBajaUsuario
    {
        public void BajaUsuario(int id);
    }
}

using LogicaAplicacion.DataTransferObjects.Models.Usuarios;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoUsuarios.Interfaces
{
    public interface ICasoUsoBuscarUsuario
    {
        public UsuarioListadoDTO BuscarUsuario(int id);
    }
}

using LogicaAplicacion.DataTransferObjects.Models.Usuarios;
using LogicaNegocio.Entidades;
```

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

namespace LogicaAplicacion.CasosUso.CasosUsoUsuarios.Interfaces
{
    public interface ICasoUsoEditarUsuario
    {
        public void EditarUsuario(int idUsuario,UsuarioModificacionDTO usuarioDTO);
    }
}

using LogicaAplicacion.DataTransferObjects.Models.Usuarios;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.CasosUsoUsuarios.Interfaces
{
    public interface ICasoUsoListarUsuario
    {
        public IEnumerable<UsuarioListadoDTO> ListarUsuarios();

    }
}
```

Mapper Articulo

```
using LogicaAplicacion.DataTransferObjects.Models.Articulos;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.DataTransferObjects.Mappers
{
    public class MapperArticulo
    {
        public static Articulo FromDTO(ArticulosDTO dto)
```

```
        {
            ArgumentNullException.ThrowIfNull(dto);
            return new Articulo(dto.Nombre, dto.Descripcion, dto.Codigo, dto.PrecioVenta,
dto.Stock);
        }

        public static ArticulosListadoDTO ToDTO(Articulo art)
        {
            return new ArticulosListadoDTO()
            {
                Id = art.Id,
                Nombre = art.Nombre.NombreArticuloValor,
                Descripcion = art.Descripcion.DescripcionArticuloValor,
                Codigo = art.Codigo.CodigoArticuloValor,
                PrecioVenta = art.PrecioVenta,
                Stock = art.Stock
            };
        }

        public static IEnumerable<ArticulosListadoDTO> FromList(IEnumerable<Articulo> arts)
        {
            return arts.Select(x => ToDTO(x));
        }
    }
}
```

Mapper Cliente

```
using LogicaAplicacion.DataTransferObjects.Models.Clientes;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.DataTransferObjects.Mappers
{
    public class MapperCliente
    {
        public static ClienteDTO ToDTO(Cliente client)
        {
            return new ClienteDTO()
            {
                Id = client.Id,
                Nombre = client.NombreCompleto.Nombre,
```

```
            Apellido = client.NombreCompleto.Apellido,
            RazonSocial = client.RazonSocial,
            Rut = client.RUT.NroRut,
            Calle = client.Direccion.Calle,
            Numero = client.Direccion.Numero,
            Ciudad = client.Direccion.Ciudad,
            Distancia = client.Direccion.Distancia
        };
    }
    public static IEnumerable<ClienteDTO> FromList(IEnumerable<Cliente> clients)
    {
        return clients.Select(x => ToDTO(x));
    }
  }
}
```

Mapper LineaPedido

```
using LogicaAplicacion.DataTransferObjects.Models.Clientes;
using LogicaAplicacion.DataTransferObjects.Models.Pedidos;
using LogicaAplicacion.DataTransferObjects.Models.Usuarios;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.DataTransferObjects.Mappers
{
  public class MapperLineaPedido
  {
    public static LineaPedidoDTO ToDTO(LineaPedido lineaPedido)
    {
      return new LineaPedidoDTO
      {
        ArticuloId = lineaPedido.ArticuloId,
        PedidoId = lineaPedido.PedidoId,
        CantidadArticulo = lineaPedido.CantidadArticulo,
        PrecioUnitario = lineaPedido.PrecioUnitario,
        Articulo = lineaPedido.Articulo
      };
    }
    public static LineaPedido FromDTO(LineaPedidoDTO lineaPedidoDTO)
    {
```

```
            return new LineaPedido
            {
                ArticuloId = lineaPedidoDTO.ArticuloId,
                PedidoId = lineaPedidoDTO.PedidoId,
                CantidadArticulo = lineaPedidoDTO.CantidadArticulo,
                PrecioUnitario = lineaPedidoDTO.PrecioUnitario,
                Articulo = lineaPedidoDTO.Articulo
            };
        }
        public static List<LineaPedido> ToList(IEnumerable<LineaPedidoDTO> lineas)
        {
            return lineas.Select(x => FromDTO(x)).ToList();
        }

        public static List<LineaPedidoDTO> FromList(IEnumerable<LineaPedido> lineas)
        {
            return lineas.Select(x => ToDTO(x)).ToList();
        }

    }
}
```

Mapper Pedido

```
using LogicaAplicacion.DataTransferObjects.Models.Pedidos;
using LogicaNegocio.Entidades;
using System.Linq;

namespace LogicaAplicacion.DataTransferObjects.Mappers
{
    public class MapperPedido
    {
        public static Pedido FromDTO(PedidoDTO pedidoDTO)
        {
            Pedido pedido = new Pedido
            {
                FechaPrometida = pedidoDTO.FechaPrometida,
                FechaCreado = pedidoDTO.FechaCreado,
                Cliente = pedidoDTO.Cliente,
                IVAAplicado = pedidoDTO.IVAAplicado,
                FechaEntregado = pedidoDTO.FechaEntregado,
                Estado = pedidoDTO.Estado,
                Lineas = MapperLineaPedido.ToList(pedidoDTO.Lineas),
                Express = pedidoDTO.Express
            };
```

```
            if (pedido.Cliente == null)
                pedido.Total = pedido.CalcularCostoBase();
            else
                pedido.Total = pedido.CalcularTotal();

            return pedido;
        }

        public static PedidoDTO ToDTO(Pedido pedido)
        {
            if (pedido == null)
                return null;

            PedidoDTO pedidoDTO = new PedidoDTO
            {
                Id = pedido.Id,
                FechaPrometida = pedido.FechaPrometida,
                FechaCreado = pedido.FechaCreado,
                Cliente = pedido.Cliente,
                IVAAplicado = pedido.IVAAplicado,
                FechaEntregado = pedido.FechaEntregado,
                Estado = pedido.Estado,
                Lineas = MapperLineaPedido.FromList(pedido.Lineas),
                Express = pedido.Express
            };
            if (pedido.Cliente == null)
                pedidoDTO.Total = pedido.CalcularCostoBase();
            else
                pedidoDTO.Total = pedido.CalcularTotal();
            return pedidoDTO;
        }

        public static IEnumerable<PedidoDTO> ToListAll(IEnumerable<Pedido> pedidos)
        {
            return pedidos.Select(p => ToDTO(p)).ToList();
        }

    }
}
```

Mapper Usuario

```
using LogicaAplicacion.DataTransferObjects.Models.Usuarios;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
```

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

namespace LogicaAplicacion.DataTransferObjects.Mappers
{
   public class MapperUsuario
   {
      public static Usuario FromDTO(UsuarioAltaDTO dto)
      {
         ArgumentNullException.ThrowIfNull(dto);
         return new Usuario(dto.Email, dto.Nombre, dto.Apellido, dto.Password, dto.Rol);
      }
      public static Usuario FromDTO(UsuarioModificacionDTO dto)
      {
         ArgumentNullException.ThrowIfNull(dto);
         var usuario = new Usuario(dto.Email, dto.Nombre, dto.Apellido, dto.Password,
dto.Rol)
         {
            Id = dto.Id
         };
         return usuario;
      }

      public static UsuarioListadoDTO ToDTO(Usuario user)
      {
         return new UsuarioListadoDTO()
         {
            Email = user.Email.DireccionEmail,
            Password = user.Contrasenia.Password,
            Rol = (int)user.Rol.RolValor,
            Nombre = user.NombreCompleto.Nombre,
            Apellido = user.NombreCompleto.Apellido,
            Id = user.Id
         };
      }

      public static IEnumerable<UsuarioListadoDTO> FromList(IEnumerable<Usuario>
users)
      {
         return users.Select(x => ToDTO(x));
      }
   }
}
```

DTOs

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

namespace LogicaAplicacion.DataTransferObjects.Models.Articulos
{
    public class ArticulosDTO
    {
        public int Id { get; set; }

        [Required(ErrorMessage = "Debe ingresar un nombre para el Articulo")]
        public string Nombre { get; set; }

        [Required(ErrorMessage = "Debe ingresar una descripcion para el Articulo")]
        public string Descripcion { get; set; }

        [Required(ErrorMessage = "Debe ingresar un codigo para el Articulo")]
        public long Codigo { get; set; }

        [Required(ErrorMessage = "Debe ingresar un precio de venta para el Articulo")]
        public decimal PrecioVenta { get; set; }

        [Required(ErrorMessage = "Debe ingresar un precio de venta para el Articulo")]
        public int Stock { get; set; }

        public ArticulosDTO() { }
    }
}

using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.DataTransferObjects.Models.Articulos
{
    public class ArticulosListadoDTO
```

```
    {
        public int Id { get; set; }
        [Required]
        public string Nombre { get; set; }
        public string Descripcion { get; set; }
        public long Codigo { get; set; }
        public decimal PrecioVenta { get; set; }
        public int Stock { get; set; }

        public ArticulosListadoDTO() { }
    }
}

using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.DataTransferObjects.Models.Clientes
{
    public class ClienteDTO
    {
        public int Id { get; set; }

        public string Nombre { get; set; }
        public string Apellido { get; set; }

        [Display(Name = "Razon Social")]
        public string RazonSocial { get; set; }

        [Display(Name = "RUT")]
        [Range(12, int.MinValue, ErrorMessage = "El RUT debe tener 12 digitos")]
        public string Rut { get; set; }
        public string Calle { get; set; }
        public int Numero { get; set; }
        public string Ciudad { get; set; }
        public int Distancia { get; set; }

        public ClienteDTO() { }
    }
}

using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.ComponentModel;
```

```
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.DataTransferObjects.Models.Pedidos
{
    public class LineaPedidoDTO
    {
        [Required]
        public int ArticuloId { get; set; }

        [Required]
        public int PedidoId { get; set; }

        [DisplayName("Cantidad del articulo")]
        [Required(ErrorMessage = "Debe ingresar una cantidad de articulos para el articulo
seleccionado")]
        public int CantidadArticulo { get; set; }

        [DisplayName("Precio del articulo")]
        public decimal PrecioUnitario { get; set; }

        public required Articulo Articulo { get; set; }
    }
}

using LogicaNegocio.Entidades;
using LogicaNegocio.Enumerados;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.DataTransferObjects.Models.Pedidos
{
    public class PedidoDTO
    {
        public int Id { get; set; }

        [Display(Name = "Fecha prometida")]
        //[DisplayFormat(DataFormatString = "{dd/MM/yyyy}", ApplyFormatInEditMode = true)]
        public DateTime FechaPrometida { get; set; }

        [Display(Name = "Fecha creado")]
```

```
        //[DisplayFormat(DataFormatString = "{dd/MM/yyyy}", ApplyFormatInEditMode = true)]
        public DateTime FechaCreado { get; set; }

        [Required(ErrorMessage = "Debe ingresar un cliente para el Pedido")]
        [DisplayName("Cliente")]
        public Cliente Cliente { get; set; }

        [Required(ErrorMessage = "Debe ingresar un total para el Pedido")]
        public decimal Total { get; set; }

        [Required(ErrorMessage = "Debe ingresar un IVA para el Pedido")]
        [DisplayName("Tasa de impuestos aplicados")]
        public double IVAAplicado { get; set; }

        [Required(ErrorMessage = "Debe ingresar una fecha de entrega para el Pedido")]
        [DisplayName("Fecha de entrega")]
        public DateTime? FechaEntregado { get; set; }

        public EEstado Estado { get; set; }

        public int ClienteId { get; set; }

        public List<LineaPedidoDTO>? Lineas { get; set; }

        public bool Express { get; set; }
    }
}

using LogicaNegocio.Enumerados;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.DataTransferObjects.Models.Usuarios
{
    public class UsuarioAltaDTO
    {
        public string Id { get; set; }
        public string Nombre { get; set; }
        public string Apellido { get; set; }
        public string Email { get; set; }
        public string Password { get; set; }
        public ERol Rol { get; set; }

        public UsuarioAltaDTO() { }
    }
```

```
}

using LogicaNegocio.Enumerados;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.DataTransferObjects.Models.Usuarios
{
    public class UsuarioListadoDTO
    {
        public int Id { get; set; }
        public string Nombre { get; set; }
        public string Apellido { get; set; }
        public string Email { get; set; }
        public string Password { get; set; }
        public int Rol { get; set; }

        public UsuarioListadoDTO() { }
    }
}

using LogicaNegocio.Enumerados;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.DataTransferObjects.Models.Usuarios
{
    public class UsuarioModificacionDTO
    {
        public int Id { get; set; }
        public string Nombre { get; set; }
        public string Apellido { get; set; }
        public string Email { get; set; }
        public string Password { get; set; }
        public ERol Rol { get; set; }

        public UsuarioModificacionDTO() { }
    }
}
```

Entidades

```
using LogicaNegocio.Excepciones.Articulos;
using LogicaNegocio.Interfaces;
using LogicaNegocio.ValueObjects;
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaNegocio.Entidades
{
    public class Articulo
    {
        #region Properties
        public int Id { get; set; }
        public NombreArticulo Nombre { get; set; }
        [Range(5, int.MinValue, ErrorMessage = "La descripción debe tener minimo 5
caracteres")]
        public DescripcionArticulo Descripcion { get; set; }
        [Range(13, int.MinValue, ErrorMessage = "El código debe tener 13 dígitos")]
        public CodigoArticulo Codigo { get; set; }
        [Display(Name = "Precio")]
        public decimal PrecioVenta { get; set; }
        public int Stock { get; set; }
        #endregion

        #region Constructor
        public Articulo(string nombre, string descripcion, long codigo, decimal precioVenta, int
stock)
        {
            Nombre = new NombreArticulo(nombre);
            Descripcion = new DescripcionArticulo(descripcion);
            Codigo = new CodigoArticulo(codigo);
            PrecioVenta = precioVenta;
            Stock = stock;
        }

        public Articulo(int id, string nombre, string descripcion, long codigo, decimal
precioVenta, int stock)
        {
            Id = id;
            Nombre = new NombreArticulo(nombre);
            Descripcion = new DescripcionArticulo(descripcion);
            Codigo = new CodigoArticulo(codigo);
```

```
            PrecioVenta = precioVenta;
            Stock = stock;
        }

        public Articulo() { }
        #endregion

    }
}

using LogicaNegocio.ValueObjects;
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaNegocio.Entidades
{
    public class Cliente
    {
        #region Properties
        public int Id { get; set; }

        [Display(Name = "Nombre Completo")]
        public NombreCompleto NombreCompleto { get; set; }

        [Display(Name = "Razon Social")]
        public string RazonSocial { get; set; }

        [Range(12, int.MinValue, ErrorMessage = "El RUT debe tener 12 digitos")]
        public RUT RUT { get; set; }

        public Direccion Direccion { get; set; }
        #endregion

        #region Constructor
        public Cliente(string razonSocial, string nombre, string apellido, string nroRut, string
calle, int numero, string ciudad, int distancia)
        {
            RazonSocial = razonSocial;
            NombreCompleto = new NombreCompleto(nombre, apellido);
            RUT = new RUT(nroRut);
            Direccion = new Direccion(calle, numero, ciudad, distancia);
        }
        public Cliente(int id, string razonSocial, string nombre, string apellido, string nroRut,
string calle, int numero, string ciudad, int distancia)
```

```
        {
            Id = id;
            RazonSocial = razonSocial;
            NombreCompleto = new NombreCompleto(nombre, apellido);
            RUT = new RUT(nroRut);
            Direccion = new Direccion(calle, numero, ciudad, distancia);
        }
        //Used for EntityFramework
        public Cliente() { }
        #endregion
    }
}
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaNegocio.Entidades
{
    [PrimaryKey(nameof(PedidoId),nameof(ArticuloId))]
    public class LineaPedido
    {

        #region Properties
        [Display(Name = "Articulo Id")]
        public int ArticuloId { get; set; }
        [Display(Name = "Pedido Id")]
        public int PedidoId { get; set; }
        [Display(Name = "Cantidad de articulos")]
        public int CantidadArticulo { get; set; }
        [Display(Name = "Precio")]
        public decimal PrecioUnitario { get; set; }
        public Articulo Articulo { get; set; }
        #endregion

        #region Constructors
        public LineaPedido(int cantidadArticulo, decimal precioUnitario, Articulo articulo)
        {
            CantidadArticulo = cantidadArticulo;
            PrecioUnitario = precioUnitario;
            Articulo = articulo;
        }
        public LineaPedido() { }
        #endregion
```

```
    }
}

using LogicaNegocio.Enumerados;
using LogicaNegocio.Excepciones.Clientes;
using LogicaNegocio.Interfaces;
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaNegocio.Entidades
{
    public class Pedido : IValidable<Pedido>
    {
        #region Properties
        //TODO: Pasar esto a archivo de configuracion
        public static double s_IVA = 22;

        public int Id { get; set; }

        [Display(Name = "Fecha prometida")]
        [DisplayFormat(DataFormatString = "{0:dd/MM/yyyy}", ApplyFormatInEditMode = true)]
        public DateTime FechaPrometida { get; set; }

        [Display(Name = "Fecha creado")]
        [DisplayFormat(DataFormatString = "{0:dd/MM/yyyy}", ApplyFormatInEditMode = true)]
        public DateTime FechaCreado { get; set; }

        public Cliente Cliente { get; set; }

        public decimal Total { get; set; }
        [Display(Name = "IVA aplicado")]
        [DisplayFormat(DataFormatString = "{0:N2}")]
        public double IVAAplicado { get; set; }

        public List<LineaPedido> Lineas { get; set; }
        [Display(Name = "Fecha de entrega")]
        [DisplayFormat(DataFormatString = "{0:dd/MM/yyyy}", ApplyFormatInEditMode = true)]
        public DateTime? FechaEntregado { get; set; }

        public EEstado Estado { get; set; }

        public bool Express { get; set; }

        #endregion
```

```
protected Pedido(DateTime fechaPrometida, Cliente cliente, bool express = false)
{
    FechaPrometida = fechaPrometida;
    FechaCreado = DateTime.Now;
    Cliente = cliente;
    Total = 0;
    IVAAplicado = s_IVA;
    Lineas = [];
    FechaEntregado = null;
    Estado = EEstado.NUEVO;
    EsValido();
    Express = express;
}

public Pedido()
{

}

public decimal CalcularCostoBase()
{
    decimal total = 0;

    foreach (LineaPedido linea in Lineas)
    {
        total += linea.CantidadArticulo * linea.PrecioUnitario;
    }

    return total;
}

#region Methods definitions
public decimal CalcularTotal()
{
    decimal total = CalcularCostoBase();

    if (Express)
    {
        TimeSpan diferenciaFechas = FechaPrometida - FechaCreado;
        int diferenciaDias = diferenciaFechas.Days;

        if (diferenciaDias < 1)
            total *= 1.15M;
        else
            total *= 1.1M;
    }
    else
```

```
            {
                if (Cliente.Direccion.Distancia > 100)
                    total *= 1.05M;
            }

            return total;
        }


        #endregion
        public void EsValido()
        {
            ValidarCliente();
        }

        public void ValidarCliente()
        {
            if (Cliente == null) throw new ClienteNoValidoException("Cliente no valido");

        }
    }
}
using LogicaNegocio.Enumerados;
using LogicaNegocio.Excepciones.Usuarios;
using LogicaNegocio.Interfaces;
using LogicaNegocio.ValueObjects;
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaNegocio.Entidades
{
    public class Usuario
    {
        #region Properties
        public int Id { get; set; }
        public Email Email { get; set; }
        public NombreCompleto NombreCompleto { get; set; }
        [Range(6, int.MinValue, ErrorMessage = "La contraseña debe tener minimo 6
caracteres")]
        public Contrasenia Contrasenia { get; set; }
        public ContraseniaEncriptada ContraseniaEncriptada { get; set; }
        public Rol Rol { get; set; }
        #endregion
```

```
        #region Constructor
        public Usuario(string email, string nombre, string apellido, string password, ERol rol)
        {
            Email = new Email(email);
            NombreCompleto = new NombreCompleto(nombre, apellido);
            Contrasenia = new Contrasenia(password);
            ContraseniaEncriptada = new ContraseniaEncriptada(password);
            Rol = new Rol(rol);
        }

        public Usuario(int id, string email, string nombre, string apellido, string password, ERol
rol)
        {
            Id = id;
            Email = new Email(email);
            NombreCompleto = new NombreCompleto(nombre, apellido);
            Contrasenia = new Contrasenia(password);
            ContraseniaEncriptada = new ContraseniaEncriptada(password);
            Rol = new Rol(rol);
        }
        public Usuario() { }
        #endregion

        #region Methods
        public void ModificarUsuario(Usuario obj)
        {
            Email = obj.Email;
            Contrasenia = obj.Contrasenia;
            Rol = obj.Rol;
            NombreCompleto = obj.NombreCompleto;
        }
        #endregion
    }
}
```

Enumerados

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

namespace LogicaNegocio.Enumerados
{
    public enum EEstado
    {
        NUEVO = 1,
        PENDIENTE = 2,
        ENTREGADO = 3,
        ANULADO = 4,
    }
}
```

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

namespace LogicaNegocio.Enumerados
{
    public enum ERol
    {
        USUARIO = 1,
        ADMINISTRADOR = 2
    }
}
```

Excepciones

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

namespace LogicaNegocio.Excepciones.Articulos
{
    public class ArticuloNoEncontradoException : Exception
    {
```

```
        public ArticuloNoEncontradoException() { }

        public ArticuloNoEncontradoException(string? message) : base(message) { }

        public ArticuloNoEncontradoException(string? message, Exception? innerException) :
base(message, innerException) { }
    }
}

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

namespace LogicaNegocio.Excepciones.Articulos
{
    public class ArticuloNoValidoException : Exception
    {
        public ArticuloNoValidoException() { }

        public ArticuloNoValidoException(string? message) : base(message) { }

        public ArticuloNoValidoException(string? message, Exception? innerException) :
base(message, innerException) { }
    }
}

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

namespace LogicaNegocio.Excepciones.Clientes
{
    public class ClienteNoEncontradoException : Exception
    {
        public ClienteNoEncontradoException() { }

        public ClienteNoEncontradoException(string? message) : base(message) { }

        public ClienteNoEncontradoException(string? message, Exception? innerException) :
base(message, innerException) { }
    }
}

using System;
```

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

namespace LogicaNegocio.Excepciones.Clientes
{
    public class ClienteNoValidoException : Exception
    {
        public ClienteNoValidoException() { }

        public ClienteNoValidoException(string? message) : base(message) { }

        public ClienteNoValidoException(string? message, Exception? innerException) :
base(message, innerException) { }
    }
}

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

namespace LogicaNegocio.Excepciones.Generales
{
    public class DataBaseSetException : Exception
    {
        public DataBaseSetException() { }

        public DataBaseSetException(string? message) : base(message) { }

        public DataBaseSetException(string? message, Exception? innerException) :
base(message, innerException) { }
    }
}

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

namespace LogicaNegocio.Excepciones.LineasPedidos
{
    public class LineaPedidoNoEncontradaException : Exception
    {
        public LineaPedidoNoEncontradaException() { }
```

```
        public LineaPedidoNoEncontradaException(string? message) : base(message) { }

        public LineaPedidoNoEncontradaException(string? message, Exception?
innerException) : base(message, innerException) { }
    }
}

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

namespace LogicaNegocio.Excepciones.LineasPedidos
{
    public class LineaPedidoNoValidaException : Exception
    {
        public LineaPedidoNoValidaException() { }

        public LineaPedidoNoValidaException(string? message) : base(message) { }

        public LineaPedidoNoValidaException(string? message, Exception? innerException) :
base(message, innerException) { }
    }
}

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

namespace LogicaNegocio.Excepciones.Pedidos
{
    public class PedidoNoEncontradoException : Exception
    {
        public PedidoNoEncontradoException() { }

        public PedidoNoEncontradoException(string? message) : base(message) { }

        public PedidoNoEncontradoException(string? message, Exception? innerException) :
base(message, innerException) { }
    }
}

using System;
using System.Collections.Generic;
```

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

namespace LogicaNegocio.Excepciones.Pedidos
{
    public class PedidoNoValidoException : Exception
    {
        public PedidoNoValidoException() { }

        public PedidoNoValidoException(string? message) : base(message) { }

        public PedidoNoValidoException(string? message, Exception? innerException) :
base(message, innerException) { }
    }
}

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

namespace LogicaNegocio.Excepciones.Usuarios
{
    public class ContraseñaNoValidaException : Exception
    {
        public ContraseñaNoValidaException() { }

        public ContraseñaNoValidaException(string? message) : base(message) { }

        public ContraseñaNoValidaException(string? message, Exception? innerException) :
base(message, innerException) { }
    }
}

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

namespace LogicaNegocio.Excepciones.Usuarios
{
    public class MailNoValidoException : Exception
    {
        public MailNoValidoException() { }
```

```
        public MailNoValidoException(string? message) : base(message) { }

        public MailNoValidoException(string? message, Exception? innerException) :
base(message, innerException) { }
    }
}

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

namespace LogicaNegocio.Excepciones.Usuarios
{
    public class NombreNoValidoException : Exception
    {
        public NombreNoValidoException() { }

        public NombreNoValidoException(string? message) : base(message) { }

        public NombreNoValidoException(string? message, Exception? innerException) :
base(message, innerException) { }
    }
}

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

namespace LogicaNegocio.Excepciones.Usuarios
{
    public class RolNoValidoException : Exception
    {
        public RolNoValidoException() { }

        public RolNoValidoException(string? message) : base(message) { }

        public RolNoValidoException(string? message, Exception? innerException) :
base(message, innerException) { }
    }
}

using System;
using System.Collections.Generic;
using System.Linq;
```

```
using System.Text;
using System.Threading.Tasks;

namespace LogicaNegocio.Excepciones.Usuarios
{
    public class UsuarioNoEncontradoException : Exception
    {
        public UsuarioNoEncontradoException() { }

        public UsuarioNoEncontradoException(string? message) : base(message) { }

        public UsuarioNoEncontradoException(string? message, Exception? innerException) :
base(message, innerException) { }
    }
}

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

namespace LogicaNegocio.Excepciones.Usuarios
{
    public class UsuarioNoValidoException : Exception
    {
        public UsuarioNoValidoException() { }

        public UsuarioNoValidoException(string? message) : base(message) { }

        public UsuarioNoValidoException(string? message, Exception? innerException) :
base(message, innerException) { }
    }
}
```

Interfaz IValidable

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

namespace LogicaNegocio.Interfaces
{
    public interface IValidable<T> where T : class
```

```
    {
        public void EsValido();
    }
}
```

Value Objects

```
using LogicaNegocio.Excepciones.Articulos;
using LogicaNegocio.Interfaces;
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations.Schema;
using System.Linq;
using System.Text;
using System.Text.RegularExpressions;
using System.Threading.Tasks;

namespace LogicaNegocio.ValueObjects
{
    [ComplexType]
    public class CodigoArticulo : IValidable<CodigoArticulo>
    {
        #region Properties
        public long CodigoArticuloValor { get; init; }
        #endregion

        #region Constructors
        public CodigoArticulo(long codigoArticuloValor)
        {
            CodigoArticuloValor = codigoArticuloValor;
            EsValido();
        }

        //Used for EntityFramework
        private CodigoArticulo(){}
        #endregion

        #region Validations
        public void EsValido()
        {
            ValidarCodigo();
        }

        private void ValidarCodigo()
        {
```

```
            string patronValido = @"\d{13}";

            if (string.IsNullOrEmpty(CodigoArticuloValor.ToString()))
                throw new ArticuloNoValidoException("El código del artículo no puede estar
vacio");
            else if (!Regex.IsMatch(CodigoArticuloValor.ToString(), patronValido))
                throw new ArticuloNoValidoException("El código del artículo debe tener 13
digitos");
        }
        #endregion
    }
}

using LogicaNegocio.Excepciones.Usuarios;
using LogicaNegocio.Interfaces;
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations.Schema;
using System.Linq;
using System.Text;
using System.Text.RegularExpressions;
using System.Threading.Tasks;

namespace LogicaNegocio.ValueObjects
{
    [ComplexType]
    public class Contrasenia : IValidable<Contrasenia>
    {
        #region Properties
        public string Password { get; init; }
        #endregion

        #region Constructors
        public Contrasenia(string password)
        {
            Password = password;
            EsValido();
        }

        //Used for EntityFramework
        private Contrasenia() {}
        #endregion

        #region Validations
        public void EsValido()
        {
            ValidarContrasenia();
            EsValido2();
```

```
    }

    private void ValidarContrasenia()
    {
        string patronValido = @"^(?=.*[A-Z])(?=.*[a-z])(?=.*\d)(?=.*[.,;!]).{6,}$";
        if (!Regex.IsMatch(Password, patronValido))
            throw new ContraseñaNoValidaException("La contraseña debe tener al menos
largo 6. Largo mínimo de 6 caracteres, al menos una letra mayúscula, una minúscula, un
dígito y un carácter de puntuación (.,;!)");
    }

    //Deprecated
    public void EsValido2()
    {
        if (string.IsNullOrEmpty(Password))
        {
            throw new ContraseñaNoValidaException("La contraseña no puede ser nula");
        }
        //La contraseña debe tener un largo minimo de 6 caracteres
        if (Password.Length < 6)
        {
            throw new ContraseñaNoValidaException("La contraseña debe tener mas de 6
caracteres");
        }

        //La contraseña debe tener al menos una mayuscula
        if (!Regex.IsMatch(Password, "[A-Z]"))
        {
            throw new ContraseñaNoValidaException("La contraseña debe contener al menos
una letra mayúscula.");
        }

        //La contraseña debe tener al menos una minuscula
        if (!Regex.IsMatch(Password, "[a-z]"))
        {
            throw new ContraseñaNoValidaException("La contraseña debe contener al menos
una letra minúscula.");
        }

        //La contraseña debe tener al menos un digito
        if (!Regex.IsMatch(Password, @"\d"))
        {
            throw new ContraseñaNoValidaException("La contraseña debe contener al menos
un dígito.");
        }

        //La contrasñea debe tener al menos un caracter de puntuacion
        if (!Regex.IsMatch(Password, "[.,;:!]"))
```

```
            {
                throw new ContraseñaNoValidaException("La contraseña debe contener al menos
uno de los siguientes caracteres: . , ; : !");
            }
        }
        #endregion
    }
}
using LogicaNegocio.Excepciones.Usuarios;
using LogicaNegocio.Interfaces;
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations.Schema;
using System.Linq;
using System.Text;
using System.Text.RegularExpressions;
using System.Threading.Tasks;

namespace LogicaNegocio.ValueObjects
{
    [ComplexType]
    public class ContraseniaEncriptada : IValidable<ContraseniaEncriptada>
    {
        #region Properties
        public string ValorContrasenia { get; init; }
        #endregion

        #region Constructors
        public ContraseniaEncriptada(string contrasenia)
        {
            ValorContrasenia = contrasenia;
            EsValido();
            ValorContrasenia = Encriptar(contrasenia);
        }

        //Used for EntityFramework
        private ContraseniaEncriptada() { }
        #endregion

        #region Validations
        public void EsValido()
        {
            ValidarContrasenia();
        }

        private void ValidarContrasenia()
        {
            string patronValido = @"^(?=.*[A-Z])(?=.*[a-z])(?=.*\d)(?=.*[.,;!]).{6,}$";
```

```
            if (!Regex.IsMatch(ValorContrasenia, patronValido))
                throw new ContraseñaNoValidaException("La contraseña debe tener al menos largo 6. Largo mínimo de 6 caracteres, al menos una letra mayúscula, una minúscula, un dígito y un carácter de puntuación (.,;!)");
        }
        #endregion

        private string Encriptar(string contrasenia)
        {
            using (SHA256 sha256Hash = SHA256.Create())
            {
                // Convertimos la contraseña en una secuencia de bytes
                byte[] bytes = sha256Hash.ComputeHash(Encoding.UTF8.GetBytes(contrasenia));

                // Convertimos los bytes a una cadena hexadecimal
                StringBuilder builder = new StringBuilder();
                for (int i = 0; i < bytes.Length; i++)
                {
                    builder.Append(bytes[i].ToString("x2"));
                }
                return builder.ToString();
            }

    }
}
using LogicaNegocio.Excepciones.Articulos;
using LogicaNegocio.Interfaces;
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations.Schema;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaNegocio.ValueObjects
{
    [ComplexType]
    public class DescripcionArticulo : IValidable<DescripcionArticulo>
    {
        #region Properties
        public string DescripcionArticuloValor { get; init; }
        #endregion

        #region Constructors
        public DescripcionArticulo(string descripcionArticulovalor)
        {
            DescripcionArticuloValor = descripcionArticulovalor;
```

```
            EsValido();
        }

        //Used for EntityFramework
        private DescripcionArticulo() { }
        #endregion

        #region Validations
        public void EsValido()
        {
            ValidarDescripcion();
        }

        private void ValidarDescripcion()
        {
            if (DescripcionArticuloValor.Length < 5)
                throw new ArticuloNoValidoException("La descripción del artículo debe contener al
menso 5 caracteres");
        }
        #endregion
    }
}


using LogicaNegocio.Excepciones.Clientes;
using LogicaNegocio.Interfaces;
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations.Schema;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaNegocio.ValueObjects
{
    [ComplexType]
    public class Direccion : IValidable<Direccion>
    {
        #region Properties
        public string Calle { get; init; }
        public int Numero { get; init; }
        public string Ciudad { get; init; }
        public int Distancia { get; init; }
        #endregion

        #region Constructors
        public Direccion(string calle, int numero, string ciudad, int distancia)
        {
```

```
            Calle = calle;
            Numero = numero;
            Ciudad = ciudad;
            Distancia = distancia;
            EsValido();
        }

        //Used for EntityFramework
        private Direccion() { }
        #endregion

        #region Validations
        public void EsValido()
        {
            ValidarDistancia();
            ValidarCiudad();
            ValidarCalle();
            ValidarNumero();
        }

        private void ValidarDistancia()
        {
            if (Distancia == 0)
                throw new ClienteNoValidoException("La distancia debe ser mayor a 0");
        }

        private void ValidarCiudad()
        {
            if (string.IsNullOrEmpty(Ciudad))
                throw new ClienteNoValidoException("Debe ingresar una ciudad");
        }
        private void ValidarCalle()
        {
            if (string.IsNullOrEmpty(Calle))
                throw new ClienteNoValidoException("Debe ingresar una calle");
        }

        private void ValidarNumero()
        {
            if (Numero <= 0)
                throw new ClienteNoValidoException("Debe ingresar un numero de puerta
valido");
        }

        #endregion
    }
}
```

```
using LogicaNegocio.Excepciones.Usuarios;
using LogicaNegocio.Interfaces;
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations.Schema;
using System.Linq;
using System.Text;
using System.Text.RegularExpressions;
using System.Threading.Tasks;

namespace LogicaNegocio.ValueObjects
{
    [ComplexType]
    public class Email : IValidable<Email>
    {
        #region Properties
        public string DireccionEmail { get; init; }
        #endregion

        #region Constructors
        public Email(string direccionEmail)
        {
            DireccionEmail = direccionEmail;
            EsValido();
        }

        //Used for EntityFramework
        private Email(){}
        #endregion

        #region Validations
        public void EsValido()
        {
            ValidarDireccionEmail();
        }

        private void ValidarDireccionEmail()
        {
            string patronValido = @"^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$";

            if (string.IsNullOrEmpty(DireccionEmail))
                throw new MailNoValidoException("El email no puede ser vacio.");
            else if (!Regex.IsMatch(DireccionEmail, patronValido))
                throw new MailNoValidoException("Ingrese un email valido");
        }
        #endregion
    }
}
```

```
using LogicaNegocio.Excepciones.Articulos;
using LogicaNegocio.Interfaces;
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations.Schema;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaNegocio.ValueObjects
{
    [ComplexType]
    public class NombreArticulo : IValidable<NombreArticulo>
    {
        #region Properties
        public string NombreArticuloValor { get; init; }
        #endregion

        #region Constructors
        public NombreArticulo(string nombreArticulo)
        {
            NombreArticuloValor = nombreArticulo;
            EsValido();
        }

        //Used for EntityFramework
        private NombreArticulo(){}
        #endregion

        #region Validations
        public void EsValido()
        {
            ValidarNombre();
        }

        private void ValidarNombre()
        {
            if (string.IsNullOrEmpty(NombreArticuloValor))
                throw new ArticuloNoValidoException("Debe ingresar un nombre para el articulo");
            else if (NombreArticuloValor.Length < 10 || NombreArticuloValor.Length > 200)
                throw new ArticuloNoValidoException("El nombre del artículo debetener entre 10 y
200 carácteres");
        }
        #endregion
    }
}
using LogicaNegocio.Excepciones.Usuarios;
```

```
using LogicaNegocio.Interfaces;
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations.Schema;
using System.Linq;
using System.Text;
using System.Text.RegularExpressions;
using System.Threading.Tasks;

namespace LogicaNegocio.ValueObjects
{
    [ComplexType]
    public class NombreCompleto : IValidable<NombreCompleto>
    {
        #region Properties
        public string Nombre { get; init; }
        public string Apellido { get; init; }
        #endregion

        #region Constructors
        public NombreCompleto(string nombre, string apellido)
        {
            Nombre = nombre;
            Apellido = apellido;
            EsValido();
        }

        //Used for EntityFramework
        private NombreCompleto(){}
        #endregion

        #region Validations
        public void EsValido()
        {
            ValidarNombre();
            ValidarApellido();
        }

        public void ValidarNombre()
        {
            string patronValido = @"^[a-zA-ZáéíóúÁÉÍÓÚüÜñÑ]+(?:['
-][a-zA-ZáéíóúÁÉÍÓÚüÜñÑ]+)*$";

            if (string.IsNullOrEmpty(Nombre))
                throw new NombreNoValidoException("Nombre no puede ser vacío");
            else if (!Regex.IsMatch(Nombre, patronValido))
```

```
            throw new NombreNoValidoException("El nombre solo puede contener letras, espacios, apostrofes o guiones. Los caracteres especiales no pueden estar al principio ni al final.");
        }

        public void ValidarApellido()
        {
            string patronValido = @"^[a-zA-ZáéíóúÁÉÍÓÚüÜñÑ]+(?:['-][a-zA-ZáéíóúÁÉÍÓÚüÜñÑ]+)*$";

            if (string.IsNullOrEmpty(Apellido))
                throw new NombreNoValidoException("Apellido no puede ser vacío");
            else if (!Regex.IsMatch(Apellido, patronValido))
                throw new NombreNoValidoException("El apellido solo puede contener letras, espacios, apostrofes o guiones. Los caracteres especiales no pueden estar al principio ni al final.");
        }
        #endregion

    }
}
using LogicaNegocio.Enumerados;
using LogicaNegocio.Excepciones.Usuarios;
using LogicaNegocio.Interfaces;
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations.Schema;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaNegocio.ValueObjects
{
    [ComplexType]
    public class Rol : IValidable<Rol>
    {
        #region Properties
        public ERol RolValor {  get; init; }
        #endregion

        #region Constructor
        public Rol(ERol rol)
        {
            RolValor = rol;
            EsValido();
        }

        public Rol() { }
```

```
        #endregion


        public void EsValido()
        {
            ValidarRol();
        }

        private void ValidarRol()
        {
            if (RolValor != ERol.ADMINISTRADOR && RolValor != ERol.USUARIO)
                throw new UsuarioNoValidoException("Ingrese un rol valido");
        }
    }
}

using LogicaNegocio.Excepciones.Clientes;
using LogicaNegocio.Interfaces;
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations.Schema;
using System.Linq;
using System.Text;
using System.Text.RegularExpressions;
using System.Threading.Tasks;

namespace LogicaNegocio.ValueObjects
{
    [ComplexType]
    public class RUT : IValidable<RUT>
    {
        public string NroRut { get; init; }

        #region Constructors
        public RUT(string nroRut)
        {
            NroRut = nroRut;
            EsValido();
        }

        //Used for EntityFramework
        private RUT(){}
        #endregion

        #region Validations
        public void EsValido()
        {
            ValidarRut();
```

```
        }

        public void ValidarRut()
        {
            string patronValido = @"\d{12}";

            if (string.IsNullOrEmpty(NroRut) || !Regex.IsMatch(NroRut, patronValido))
                throw new ClienteNoValidoException($"El RUT ingresado {NroRut} no es valido.");
        }
        #endregion
    }
}
```

Articulos Controller

```
using AccesoDatos.Implementaciones.EntityFramework;
using AccesoDatos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoArticulos.Implementaciones;
using LogicaAplicacion.CasosUso.CasosUsoArticulos.Interfaces;
using LogicaAplicacion.DataTransferObjects.Models.Articulos;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;

namespace MVC_Papeleria.Controllers
{
    public class ArticulosController : Controller
    {
        //casos de uso
        private ICasoUsoAltaArticulo _altaArticulos;
        private ICasoUsoListarArticulos _getAllArticulos;

        public ArticulosController(ICasoUsoAltaArticulo altaArticulo, ICasoUsoListarArticulos
listarArticulos)
        {
            _altaArticulos = altaArticulo;
            _getAllArticulos = listarArticulos;
        }


        // GET: ArticulosController
        public ActionResult Index()
        {
            if (HttpContext.Session.GetString("email") != null)
            {
                return View(_getAllArticulos.ListarArticulos());
```

```
    }
    return RedirectToAction("Login");
}

// GET: ArticulosController/Details/5
public ActionResult Details(int id)
{
    return View();
}

// GET: ArticulosController/Create
public ActionResult Create()
{
    return View();
}

// POST: ArticulosController/Create
[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult Create(ArticulosDTO art)
{
    try
    {
        _altaArticulos.AltaArticulo(art);
        return RedirectToAction(nameof(Index));
    }
    catch (Exception ex)
    {
        ViewBag.ErrorMessage = ex.Message;
        return View();
    }
}

// GET: ArticulosController/Edit/5
public ActionResult Edit(int id)
{
    return View();
}

// POST: ArticulosController/Edit/5
[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult Edit(int id, IFormCollection collection)
{
    try
    {
        return RedirectToAction(nameof(Index));
    }
```

```
            catch
            {
                return View();
            }
        }

        // GET: ArticulosController/Delete/5
        public ActionResult Delete(int id)
        {
            return View();
        }

        // POST: ArticulosController/Delete/5
        [HttpPost]
        [ValidateAntiForgeryToken]
        public ActionResult Delete(int id, IFormCollection collection)
        {
            try
            {
                return RedirectToAction(nameof(Index));
            }
            catch
            {
                return View();
            }
        }
    }
}
```

Clientes Controller

```
using AccesoDatos.Implementaciones.EntityFramework;
using AccesoDatos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoClientes.Implementaciones;
using LogicaAplicacion.CasosUso.CasosUsoClientes.Interfaces;
using LogicaAplicacion.DataTransferObjects.Models.Clientes;
using Microsoft.AspNetCore.Mvc;

namespace MVC_Papeleria.Controllers
{
    public class ClientesController : Controller
    {
        //casos de uso
        private ICasoUsoBuscarCliente _filtrarTexto;
        private ICasoUsoBuscarCliente _filtrarMonto;
        private ICasoUsoListarClientes _getAllClientes;
```

```
    public ClientesController(ICasoUsoBuscarCliente buscarPorTexto,
ICasoUsoBuscarCliente buscarPorMonto, ICasoUsoListarClientes listarClientes)
    {
        _filtrarTexto = buscarPorTexto;
        _filtrarMonto = buscarPorMonto;
        _getAllClientes = listarClientes;

    }

    // GET: ClientesController
    public ActionResult Index()
    {
        if (HttpContext.Session.GetString("email") != null)
        {
            return View(_getAllClientes.ListarClientes());
        }
        return RedirectToAction("Login");
    }

    [HttpPost]
    public ActionResult Index(string textoAFiltrar, string montoAFiltrar)
    {
        IEnumerable<ClienteDTO> filtrados = null;
        if (textoAFiltrar != null && montoAFiltrar != null)
        {
            ViewBag.ErrorMessage = "Solo filtrar por un campo";
            return View(_getAllClientes.ListarClientes());
        }
        if (textoAFiltrar == null && montoAFiltrar == null)
        {
            ViewBag.ErrorMessage = "Ingrese valores en al menos un campo para filtrar";
            return View(_getAllClientes.ListarClientes());
        }
        if (textoAFiltrar != null)
        {
            filtrados = _filtrarTexto.BuscarClientePorTexto(textoAFiltrar);
        }
        if (montoAFiltrar != null)
        {
            decimal monto = decimal.Parse(montoAFiltrar);
            filtrados = _filtrarMonto.BuscarClientePorMonto(monto);
        }
        if (filtrados == null || filtrados.Count() == 0)
        {
            ViewBag.Message = "Lista vacia";
            return View();
        }
```

```
    return View(filtrados);

}


// GET: ClientesController/Details/5
public ActionResult Details(int id)
{
    return View();
}

// GET: ClientesController/Create
public ActionResult Create()
{
    return View();
}

// POST: ClientesController/Create
[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult Create(IFormCollection collection)
{
    try
    {
        return RedirectToAction(nameof(Index));
    }
    catch
    {
        return View();
    }
}

// GET: ClientesController/Edit/5
public ActionResult Edit(int id)
{
    return View();
}

// POST: ClientesController/Edit/5
[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult Edit(int id, IFormCollection collection)
{
    try
    {
        return RedirectToAction(nameof(Index));
    }
    catch
```

```
            {
                return View();
            }
        }

        // GET: ClientesController/Delete/5
        public ActionResult Delete(int id)
        {
            return View();
        }

        // POST: ClientesController/Delete/5
        [HttpPost]
        [ValidateAntiForgeryToken]
        public ActionResult Delete(int id, IFormCollection collection)
        {
            try
            {
                return RedirectToAction(nameof(Index));
            }
            catch
            {
                return View();
            }
        }
    }
}
```

Home Controller

```
using Microsoft.AspNetCore.Mvc;
using MVC_Papeleria.Models;
using System.Diagnostics;

namespace MVC_Papeleria.Controllers
{
    public class HomeController : Controller
    {
        private readonly ILogger<HomeController> _logger;

        public HomeController(ILogger<HomeController> logger)
        {
            _logger = logger;
        }

        public IActionResult Index()
```

```
        {
            return View();
        }

        public IActionResult Privacy()
        {
            return View();
        }

        [ResponseCache(Duration = 0, Location = ResponseCacheLocation.None, NoStore =
true)]
        public IActionResult Error()
        {
            return View(new ErrorViewModel { RequestId = Activity.Current?.Id ??
HttpContext.TraceIdentifier });
        }
    }
}
```

Pedidos Controller

```
using LogicaAplicacion.CasosUso.CasosUsoArticulos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoClientes.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoPedidos.Interfaces;
using LogicaAplicacion.DataTransferObjects.Mappers;
using LogicaAplicacion.DataTransferObjects.Models.Articulos;
using LogicaAplicacion.DataTransferObjects.Models.Pedidos;
using LogicaNegocio.Entidades;
using LogicaNegocio.Enumerados;
using Microsoft.AspNetCore.Mvc;

namespace MVC_Papeleria.Controllers
{
    public class PedidosController : Controller
    {
        //Casos de uso
        private ICasoUsoAltaPedido _altaPedido;
        private ICasoUsoBajaPedido _bajaPedido;
        private ICasoUsoBuscarArticulo _buscarArticulo;
        private ICasoUsoBuscarCliente _buscarCliente;
        private ICasoUsoBuscarPedido _buscarPedido;
        private ICasoUsoListarPedido _listarPedidos;
        private ICasoUsoListarClientes _listarClientes;
        private ICasoUsoListarArticulos _listarArticulos;

        private static List<LineaPedidoDTO>? tempLineas;
```

```
public PedidosController(ICasoUsoAltaPedido altaPedido,
    ICasoUsoBajaPedido bajaPedido,
    ICasoUsoBuscarArticulo buscarArticulo,
    ICasoUsoBuscarCliente buscarCliente,
    ICasoUsoBuscarPedido buscarPedido,
    ICasoUsoListarPedido listarPedidos,
    ICasoUsoListarClientes listarClientes,
    ICasoUsoListarArticulos listarArticulos)
{
    _altaPedido = altaPedido;
    _bajaPedido = bajaPedido;
    _buscarArticulo = buscarArticulo;
    _buscarCliente = buscarCliente;
    _buscarPedido = buscarPedido;
    _listarPedidos = listarPedidos;
    _listarClientes = listarClientes;
    _listarArticulos = listarArticulos;
}


// GET: PedidosController
public ActionResult Index()
{
    return View(MapperPedido.ToListAll(_listarPedidos.ListarPedidos()));
}


// GET: PedidosController/AddLines
public ActionResult AddLines()
{
    try
    {
        IEnumerable<ArticulosListadoDTO> articulos = _listarArticulos.ListarArticulos();
        ViewBag.Articulos = articulos;

        if (tempLineas != null)
            ViewBag.LineasAgregadas = tempLineas;

        return View();
    }
    catch (Exception ex)
    {
        TempData["ErrorMessage"] = ex.Message;
        return RedirectToAction(nameof(Index));
    }

}
```

```
// POST: PedidosController/AddLines
[HttpPost]
public ActionResult AddLines(int articuloId, int cantidadArticulo)
{
    try
    {
        Articulo articulo = _buscarArticulo.BuscarArticulo(articuloId);

        LineaPedidoDTO lineaPedido = new LineaPedidoDTO
        {
            Articulo = articulo,
            CantidadArticulo = cantidadArticulo,
            PrecioUnitario = articulo.PrecioVenta
        };

        if (tempLineas == null)
            tempLineas = new List<LineaPedidoDTO>();

        tempLineas.Add(lineaPedido);

        return RedirectToAction(nameof(AddLines));
    }
    catch (Exception ex)
    {
        TempData["ErrorMessage"] = ex.Message;
        return RedirectToAction(nameof(Index));
    }

}

// GET: PedidosController/BorrarLineas
public ActionResult BorrarLineas()
{
    if (tempLineas != null)
    {
        tempLineas = null;
        ViewBag.LineasAgregadas = null;
    }

    return RedirectToAction(nameof(AddLines));
}

// GET: PedidosController/Create
public ActionResult Create()
{
    ViewBag.Clientes = _listarClientes.ListarClientes();
```

```
            if (tempLineas != null)
                ViewBag.Lineas = tempLineas;

            PedidoDTO pedidoDTO = new PedidoDTO();
            pedidoDTO.Lineas = tempLineas;
            pedidoDTO.FechaCreado = DateTime.Now.Date;
            pedidoDTO.FechaPrometida = DateTime.Now.AddDays(5).Date;
            pedidoDTO.FechaEntregado = DateTime.Now.Date;
            pedidoDTO.IVAAplicado = 22;
            pedidoDTO.Estado = EEstado.NUEVO;
            pedidoDTO.Total = MapperPedido.FromDTO(pedidoDTO).CalcularCostoBase();

            return View(pedidoDTO);
        }

        // POST: PedidosController/Create
        [HttpPost]
        [ValidateAntiForgeryToken]
        public ActionResult Create(PedidoDTO nuevoPedido)
        {
            try
            {
                if (tempLineas != null && tempLineas.Count > 0)
                    nuevoPedido.Lineas = tempLineas;
                nuevoPedido.Id = nuevoPedido.Id;
                nuevoPedido.FechaCreado = DateTime.Now;
                //TODO: Agregar el parametro de settings
                nuevoPedido.FechaPrometida = DateTime.Now.AddDays(5);
                //TODO: Agregar el parametro de settings
                nuevoPedido.IVAAplicado = 22;
                nuevoPedido.Estado = EEstado.NUEVO;
                nuevoPedido.Total = nuevoPedido.Total;
                //nuevoPedido.Cliente = new Cliente { Id = nuevoPedido.ClienteId };
                nuevoPedido.Cliente =
_buscarCliente.BuscarClientePorId(nuevoPedido.ClienteId);

                _altaPedido.AltaPedido(nuevoPedido);

                tempLineas = null;
                return RedirectToAction(nameof(Index));
            }
            //TODO Hacer los catch
            catch (Exception ex)
            {
                TempData["ErrorMessage"] = ex.Message;
                return RedirectToAction(nameof(Create));
            }
        }
```

```
        public ActionResult AnularPedido(int id)
        {
            if (id == null)
            {
                ViewBag.ErrorMessage = "Se requiere el id para el pedido";
                return View();
            }
            try
            {
                PedidoDTO pedidoDTO = _buscarPedido.BuscarPedido(id);

                if(pedidoDTO != null)
                    return View(pedidoDTO);
                else
                    return View();
            }
            catch (Exception ex)
            {
                TempData["ErrorMessage"] = ex.Message;
                return View();
            }
        }

        [HttpPost]
        [ValidateAntiForgeryToken]
        public ActionResult AnularPedido(int id, PedidoDTO pedidoDTO)
        {
            try
            {
                _bajaPedido.BajaPedido(pedidoDTO.Id);
                return RedirectToAction(nameof(Index));
            }
            catch (Exception ex)
            {
                TempData["ErrorMessage"] = ex.Message;
                return View();
            }
        }

    }
}
```

Usuarios Controller

```
using AccesoDatos.Implementaciones.EntityFramework;
```

```
using AccesoDatos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoLogin.Implementaciones;
using LogicaAplicacion.CasosUso.CasosUsoLogin.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoUsuarios.Implementaciones;
using LogicaAplicacion.CasosUso.CasosUsoUsuarios.Interfaces;
using LogicaAplicacion.DataTransferObjects.Models.Usuarios;
using LogicaNegocio.Entidades;
using LogicaNegocio.Enumerados;
using Microsoft.AspNetCore.Mvc;

namespace MVC_Papeleria.Controllers
{
    public class UsuariosController : Controller
    {
        //Casos de Uso
        private ICasoUsoLoginUsuario _loginUsuario;
        private ICasoUsoAltaUsuario _altaUsuario;
        private ICasoUsoListarUsuario _getAllUsuarios;
        private ICasoUsoEditarUsuario _modificarUsuario;
        private ICasoUsoBajaUsuario _borrarUsuario;
        private ICasoUsoBuscarUsuario _getUsuario;

        public UsuariosController(ICasoUsoLoginUsuario loginUsuario, ICasoUsoAltaUsuario
altaUsuario, ICasoUsoListarUsuario getAllUsuarios, ICasoUsoEditarUsuario
modificarUsuario, ICasoUsoBajaUsuario borrarUsuario, ICasoUsoBuscarUsuario
getUsuario)
        {
            _loginUsuario = loginUsuario;
            _altaUsuario = altaUsuario;
            _getAllUsuarios = getAllUsuarios;
            _modificarUsuario = modificarUsuario;
            _borrarUsuario = borrarUsuario;
            _getUsuario = getUsuario;
        }



        // GET: UsuariosController
        public ActionResult Index()
        {
            if (HttpContext.Session.GetString("email") != null)
            {
                return View(_getAllUsuarios.ListarUsuarios());
            }
            return RedirectToAction("Login");
        }

        // GET: UsuariosController/Details/5
```

```
public ActionResult Details(int id)
{
    return View();
}

// GET: UsuariosController/Create
public ActionResult Create()
{
    return View();
}

// POST: UsuariosController/Create
[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult Create(UsuarioAltaDTO user)
{
    try
    {
        _altaUsuario.AltaUsuario(user);

        return RedirectToAction(nameof(Index));
    }
    catch (Exception ex)
    {
        ViewBag.ErrorMessage = ex.Message;
        return View();
    }
}

// GET: UsuariosController/Edit/5
public ActionResult Edit(int? id)
{
    if (id == null)
    {
        ViewBag.ErrorMessage = "Se requiere el id del usuario";
        return View();
    }
    try
    {
        UsuarioListadoDTO dto = _getUsuario.BuscarUsuario(id.GetValueOrDefault());
        UsuarioModificacionDTO mod = new UsuarioModificacionDTO()
        {
            Id = dto.Id,
            Nombre = dto.Nombre,
            Apellido = dto.Apellido,
            Email = dto.Email,
            Password = dto.Password,
            Rol = (LogicaNegocio.Enumerados.ERol)dto.Rol
```

```
        };

        if (dto != null)
        {
            return View(mod);

        }
        return View();

    }
    catch (Exception ex)
    {
        ViewBag.ErrorMessage = ex.Message;
        return View();
    }
}

// POST: UsuariosController/Edit/5
[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult Edit(int id, UsuarioModificacionDTO usuarioDTO)
{
    try
    {
        _modificarUsuario.EditarUsuario(id, usuarioDTO);
        return RedirectToAction(nameof(Index));
    }
    catch (Exception ex)
    {
        ViewBag.ErrorMessage = ex.Message;
        return View();
    }
}

// GET: UsuariosController/Delete/5
public ActionResult Delete(int id)
{
    try
    {
        UsuarioListadoDTO user = _getUsuario.BuscarUsuario(id);
        return View(user);
    }
    catch (Exception ex)
    {
        ViewBag.ErrorMessage = ex.Message;
        return View();
    }
```

```
}

// POST: UsuariosController/Delete/5
[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult Delete(int id, UsuarioListadoDTO dto)
{
    try
    {
        _borrarUsuario.BajaUsuario(id);
        return RedirectToAction(nameof(Index));
    }
    catch (Exception ex)
    {
        ViewBag.ErrorMessage = ex.Message;
        return View();
    }
}

public ActionResult Login()
{
    return View();
}
[HttpPost]
public ActionResult Login(string email, string contraseña)
{
    try
    {
        Usuario? logueado = _loginUsuario.Ejecutar(email, contraseña);
        if (logueado != null)
        {
            HttpContext.Session.SetString("email", email);
            HttpContext.Session.SetString("rol", logueado.Rol.RolValor.ToString());
        }
        if (logueado.Rol.RolValor == ERol.ADMINISTRADOR)
            return RedirectToAction("Index", "Usuarios");
        else
            return RedirectToAction("Index", "Articulos");
    }
    catch (Exception ex)
    {
        TempData["ErrorMessage"] = ex.Message;
        return RedirectToAction("Login");
    }

}
public ActionResult Logout()
{
```

```
            HttpContext.Session.Remove("email");
            return RedirectToAction("Login");
        }
    }
}
```

Program

```
using AccesoDatos.Implementaciones.EntityFramework;
using AccesoDatos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoArticulos.Implementaciones;
using LogicaAplicacion.CasosUso.CasosUsoArticulos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoClientes.Implementaciones;
using LogicaAplicacion.CasosUso.CasosUsoClientes.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoLogin.Implementaciones;
using LogicaAplicacion.CasosUso.CasosUsoLogin.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoPedidos.Implementaciones;
using LogicaAplicacion.CasosUso.CasosUsoPedidos.Interfaces;
using LogicaAplicacion.CasosUso.CasosUsoUsuarios.Implementaciones;
using LogicaAplicacion.CasosUso.CasosUsoUsuarios.Interfaces;

namespace MVC_Papeleria
{
    public class Program
    {
        public static void Main(string[] args)
        {
            var builder = WebApplication.CreateBuilder(args);

            // Add services to the container.
            builder.Services.AddControllersWithViews();

            #region Repositorios
            builder.Services.AddScoped<IRepositorioArticulos, RepositorioArticulos>();
            builder.Services.AddScoped<IRepositorioClientes, RepositorioClientes>();
            builder.Services.AddScoped<IRepositorioPedidos, RepositorioPedidos>();
            builder.Services.AddScoped<IRepositorioUsuarios, RepositorioUsuarios>();
            #endregion

            #region CasosUso
            //Usuario
            builder.Services.AddScoped<ICasoUsoLoginUsuario, CasoUsoLoginUsuario>();
            builder.Services.AddScoped<ICasoUsoAltaUsuario, CasoUsoAltaUsuario>();
            builder.Services.AddScoped<ICasoUsoListarUsuario, CasoUsoListarUsuario>();
            builder.Services.AddScoped<ICasoUsoEditarUsuario, CasoUsoEditarUsuario>();
            builder.Services.AddScoped<ICasoUsoBajaUsuario, CasoUsoBajaUsuario>();
```

```
builder.Services.AddScoped<ICasoUsoBuscarUsuario, CasoUsoBuscarUsuario>();
//Articulos
builder.Services.AddScoped<ICasoUsoAltaArticulo, CasoUsoAltaArticulo>();
builder.Services.AddScoped<ICasoUsoListarArticulos, CasoUsoListarArticulos>();
builder.Services.AddScoped<ICasoUsoBuscarArticulo, CasoUsoBuscarArticulo>();
//Clientes
builder.Services.AddScoped<ICasoUsoBuscarCliente, CasoUsoBuscarCliente>();
builder.Services.AddScoped<ICasoUsoListarClientes, CasoUsoListarClientes>();
//Pedidos
builder.Services.AddScoped<ICasoUsoAltaPedido, CasoUsoAltaPedido>();
builder.Services.AddScoped<ICasoUsoListarPedido, CasoUsoListarPedido>();
builder.Services.AddScoped<ICasoUsoEditarPedido, CasoUsoEditarPedido>();
builder.Services.AddScoped<ICasoUsoBajaPedido, CasoUsoBajaPedido>();
builder.Services.AddScoped<ICasoUsoBuscarPedido, CasoUsoBuscarPedido>();
#endregion

builder.Services.AddDistributedMemoryCache();
//Sesion

builder.Services.AddSession(options =>
{
    //15 minutos por sesion
    options.IdleTimeout = TimeSpan.FromSeconds(900);
    options.Cookie.HttpOnly = true;
    options.Cookie.IsEssential = true;
});

var app = builder.Build();

// Configure the HTTP request pipeline.
if (!app.Environment.IsDevelopment())
{
    app.UseExceptionHandler("/Home/Error");
    // The default HSTS value is 30 days. You may want to change this for production
scenarios, see https://aka.ms/aspnetcore-hsts.
    app.UseHsts();
}

app.UseHttpsRedirection();
app.UseStaticFiles();

app.UseRouting();

app.UseAuthorization();

app.UseSession(); //NECESARIO

app.MapControllerRoute(
```

```
            name: "default",
            pattern: "{controller=Home}/{action=Index}/{id?}");

        app.Run();
    }
  }
}
```