Universidad ORT Uruguay
Facultad de Ingeniería
Escuela de Tecnología

PROGRAMACIÓN 3

Gianfranco Bonanni – 274029

Gabriel Moreno – 222706

Docente: Adriana Cabella

Analista en Tecnologías de la Información

16-05-2024

Contenido

DIAGRAMA CASOS DE USO

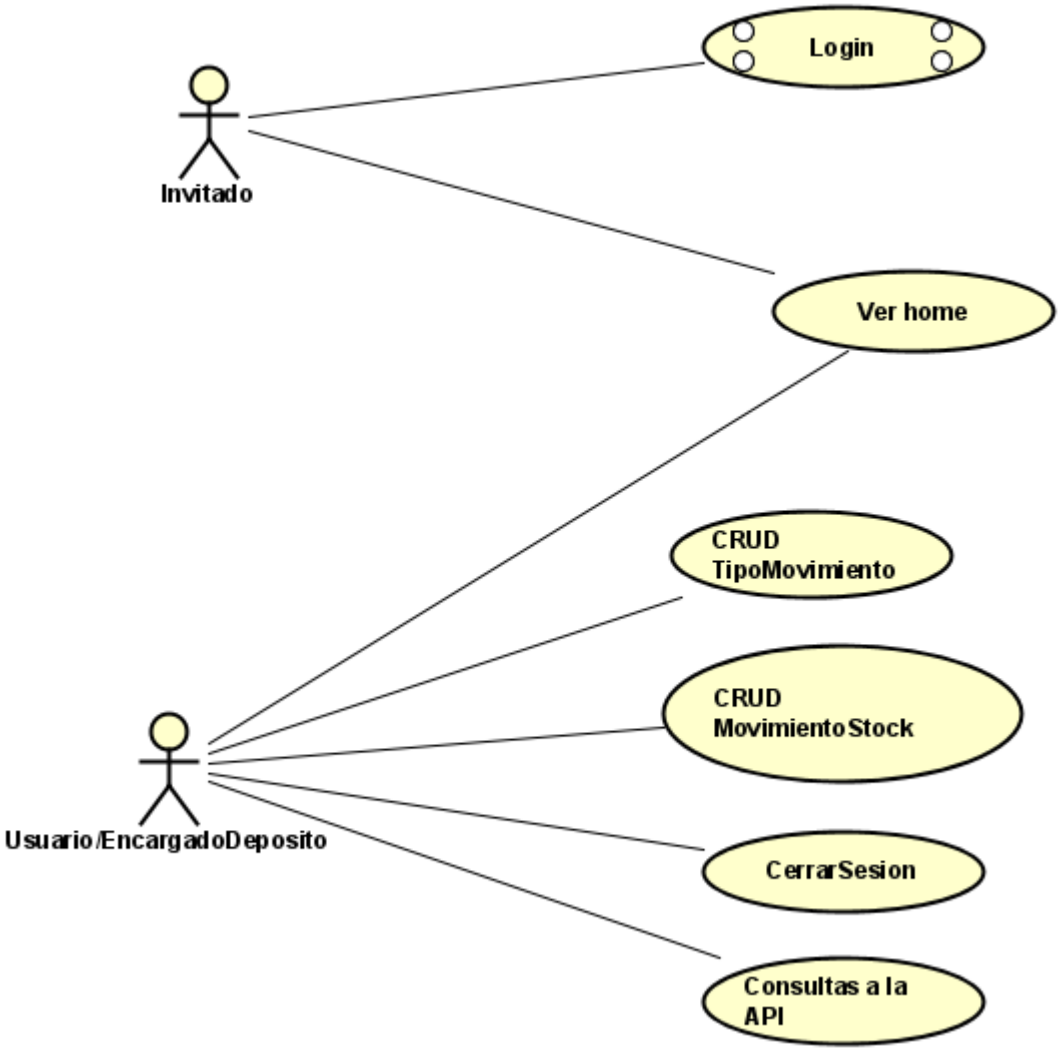


DIAGRAMA DE CLASES

Diagrama clases

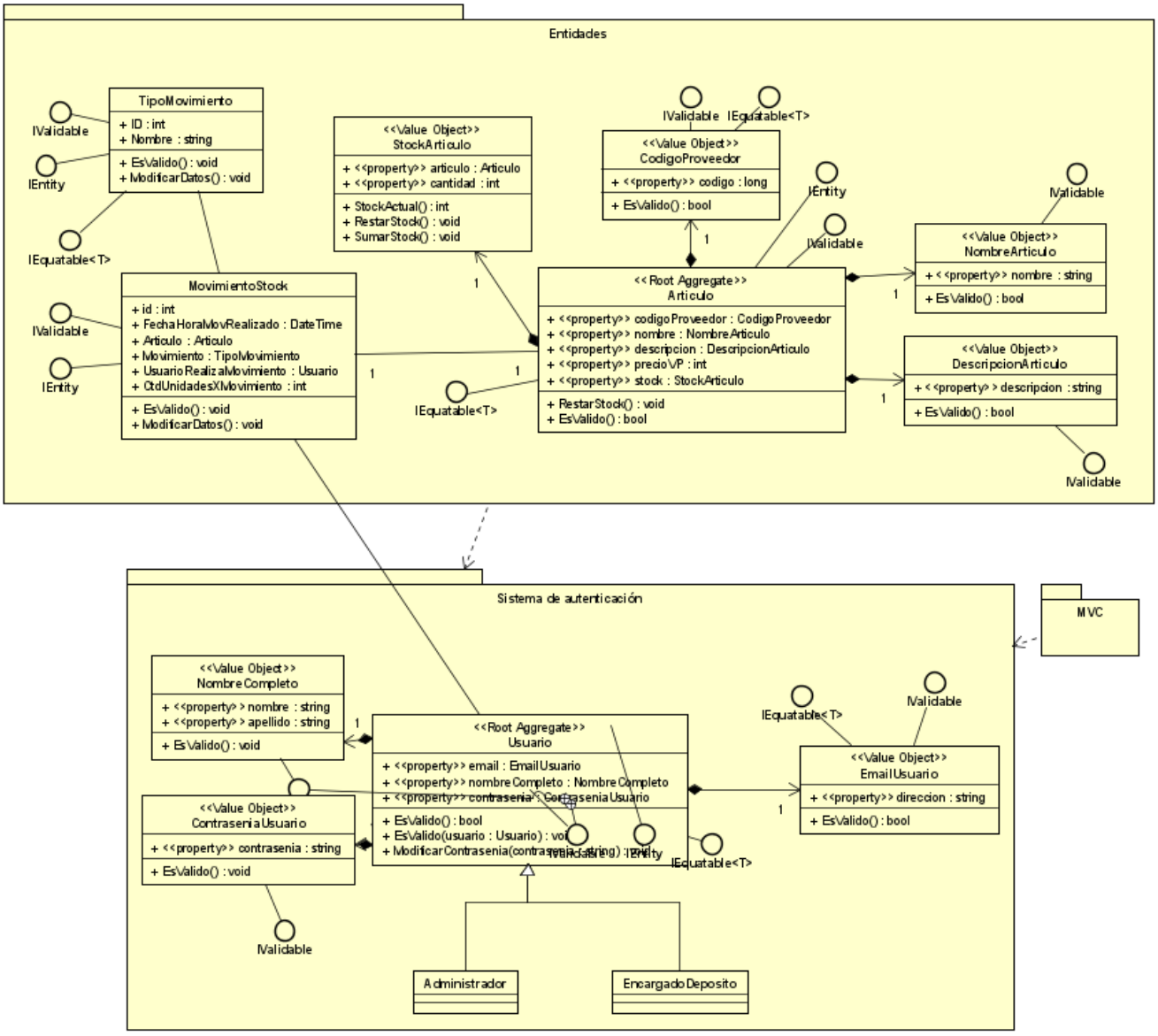
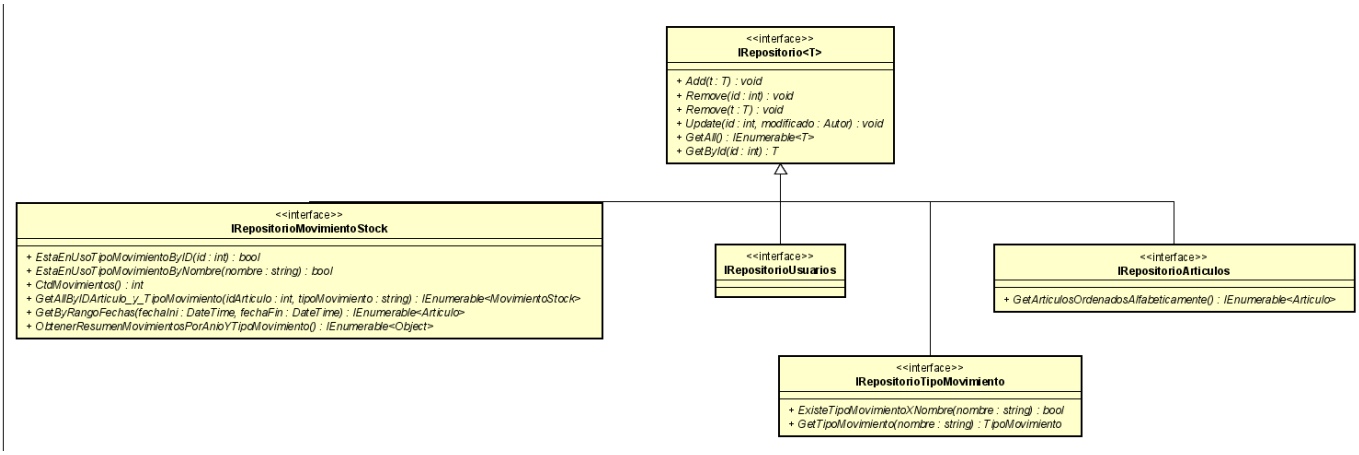
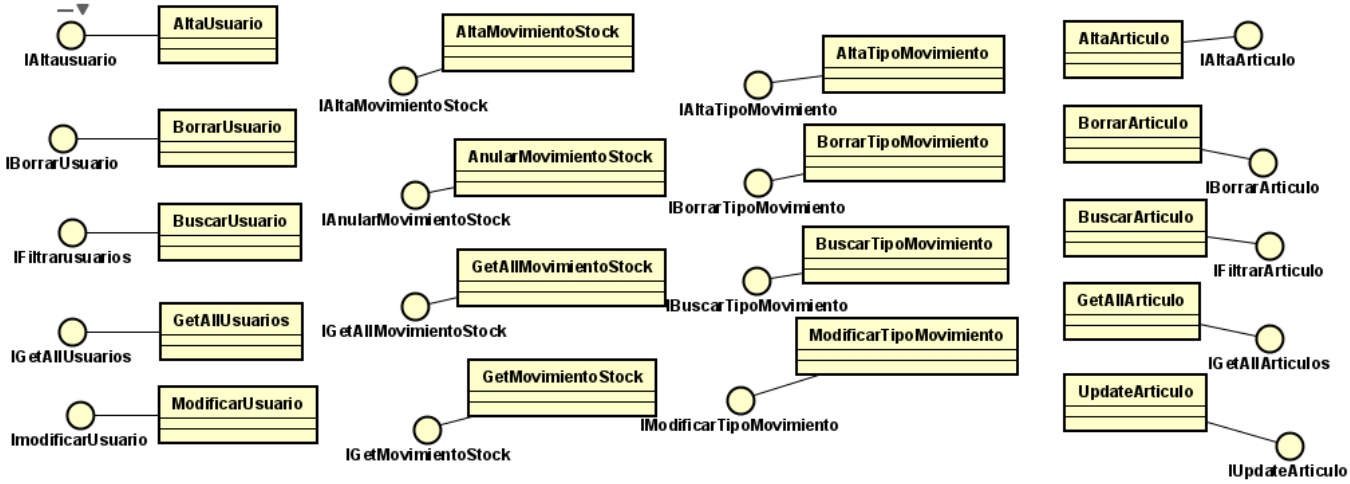


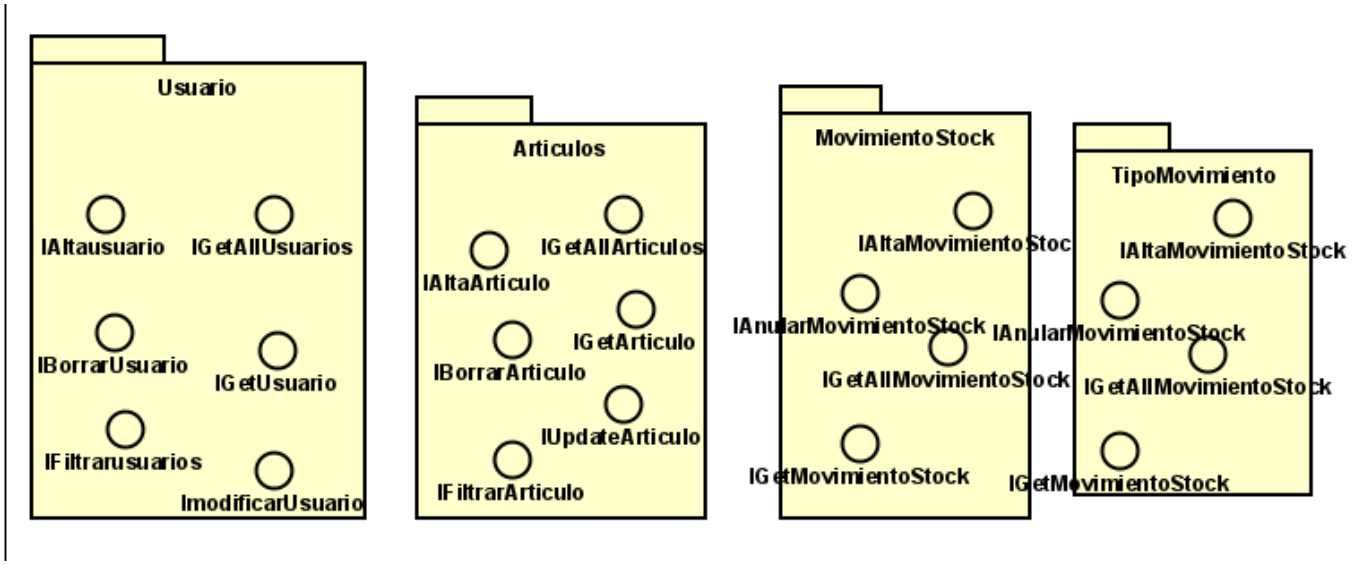
Diagrama Interfaces



ImplementacionCasoUso



Interfaces de Caso de uso



CODIGO FUENTE DE LA APLICACIÓN.

WEBAPI

****

Archivo: PapeleriaContext.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.AccesoDatos\EF\PapeleriaContext.cs

****

```
using Microsoft.EntityFrameworkCore;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Empresa.LogicaDeNegocio.Entidades;

using Empresa.LogicaDeNegocio.Sistema;

using Papeleria.LogicaNegocio.Entidades;

using Microsoft.EntityFrameworkCore.Storage.ValueConversion;



namespace Papeleria.AccesoDatos.EF

{

    public class PapeleriaContext:DbContext

    {

        public DbSet<Usuario> Usuarios{ get; set; }
```

```
        public DbSet<Articulo> Articulos { get; set; }
        public DbSet<EncargadoDeposito> EncargadosDepositos { get; set; }
        public DbSet<TipoMovimiento> TiposMovimientos { get; set; }
        public DbSet<MovimientoStock> MovimientoStocks { get; set; }

        public PapeleriaContext(DbContextOptions options) : base(options) { }
        public PapeleriaContext():base()
        {

        }
         protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
          {
            if (!optionsBuilder.IsConfigured) throw new Exception("OptionsBuilder not configured");
          }

        protected override void OnModelCreating(ModelBuilder modelBuilder)
        {

modelBuilder.Entity<Usuario>().HasDiscriminator<string>("Tipo").HasValue<Administrador>("Administrador").HasValue<EncargadoDeposito>("EncargadoDeposito");
            base.OnModelCreating(modelBuilder);
        }
    }
}
```

****

Archivo: RepositorioArticuloEF.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.AccesoDatos\EF\RepositorioArticuloEF.cs

****

```
using Empresa.LogicaDeNegocio.Entidades;
using Papeleria.LogicaNegocio.Entidades;
using Papeleria.LogicaNegocio.Entidades.ValueObjects.Articulos;
using Papeleria.LogicaNegocio.Excepciones.Articulo;
using Papeleria.LogicaNegocio.Excepciones.Usuario;
using Papeleria.LogicaNegocio.InterfacesRepositorio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Papeleria.AccesoDatos.EF
{
    public class RepositorioArticuloEF : IRepositorioArticulo
    {
        private PapeleriaContext _db;
```

```
public RepositorioArticuloEF(PapeleriaContext context) { _db = context; }

public void Add(Articulo obj)

{

   try

   {

      _db.Articulos.Add(obj);

      _db.SaveChanges();

   }

   catch (Exception ex)

   {

      throw new ArticuloNoValidoException(ex.Message);

   }

}



public IEnumerable<Articulo> GetAll()

{

   return _db.Articulos.ToList();

}



public Articulo GetArticuloByCodigo(CodigoProveedorArticulos codigo)

{

   Articulo? articulo = _db.Articulos.FirstOrDefault(art => art.CodigoProveedor.codigo == codigo.codigo);
```

```
    return articulo;
}

public IEnumerable<Articulo> GetArticulosOrdenadosAlfabeticamente()
{
    throw new NotImplementedException();
}

public Articulo GetById(int id)
{
    Articulo? articulo = _db.Articulos.FirstOrDefault(art => art.ID == id);
    return articulo;
}

public IEnumerable<Articulo> GetObjectsByID(List<int> ids)
{
    throw new NotImplementedException();
}

public void Remove(int id)
{
    var articulo = _db.Articulos.FirstOrDefault(u => u.ID == id);
    if (articulo != null)
```

```
    {
        _db.Articulos.Remove(articulo);
        _db.SaveChanges();
    }
}

public void Remove(Articulo obj)
{
    _db.Articulos.Remove(obj);
    _db.SaveChanges();
}

public void Update(int id, Articulo obj)
{
    var articulo = _db.Articulos.FirstOrDefault(u => u.ID == id);

    if (articulo != null)
    {
        try
        {
            articulo.ModificarDatos(obj);
            _db.SaveChanges();
        }
```

```
            catch (Exception ex)
            {


                throw new ArticuloNoValidoException(ex.Message);
            }
        }
        else
        {
            throw new ArticuloNuloException("El articulo no existe");
        }
    }


    public bool ExisteArticuloConNombre(string nombre)
    {
        Articulo? articulo = _db.Articulos.FirstOrDefault(art => art.NombreArticulo.Nombre == nombre);
        if (articulo != null) {
            return true;
        }
        return false;
    }
  }
}
```

****

Archivo: RepositorioMovimientoStockEF.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.AccesoDatos\EF\RepositorioMovimientoStockEF.cs

****

```
using Empresa.LogicaDeNegocio.Entidades;
using Microsoft.EntityFrameworkCore;
using Microsoft.Identity.Client;
using Papeleria.LogicaNegocio.Entidades;
using Papeleria.LogicaNegocio.Excepciones.Articulo;
using Papeleria.LogicaNegocio.Excepciones.MovimientoStock;
using Papeleria.LogicaNegocio.InterfacesRepositorio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using static System.Runtime.InteropServices.JavaScript.JSType;

namespace Papeleria.AccesoDatos.EF
{
    public class RepositorioMovimientoStockEF : IRepositorioMovimientoStock
    {
```

```
    private PapeleriaContext _db;
    private IRepositorioTipoMovimiento _repoTM;
    private IRepositorioArticulo _repoArt;
    private IRepositorioUsuario _repoUsr;
    public RepositorioMovimientoStockEF(PapeleriaContext context, IRepositorioTipoMovimiento repoTM, IRepositorioArticulo repoArt, IRepositorioUsuario repoUsr)
    {
        _db = context; _repoTM = repoTM;
        _repoArt = repoArt;
        _repoUsr = repoUsr;
    }
    public void Add(MovimientoStock obj)
    {
        if (obj == null)
            throw new MovimientoStockNuloException("No puede ser nulo el objeto a agregar.");
        try
        {
            _db.MovimientoStocks.Add(obj);
            _db.SaveChanges();
        }
        catch (Exception ex)
        {
            throw new MovimientoStockNoValidoException(ex.Message);
        }
```

```
}

public int CtdMovimientos()
{
    return _db.MovimientoStocks.Count();
}

public bool EstaEnUsoTipoMovimientoByID(int id)
{
    if (id == null)
        throw new MovimientoStockNuloException("No puede ser nulo el ID del TipoMovimiento a buscar.");
    try
    {
        var tipMovs = _db.MovimientoStocks.Where(tipMov => tipMov.Movimiento.ID == id);
        return tipMovs != null;
    }
    catch (Exception ex)
    {
        throw new MovimientoStockNoValidoException(ex.Message);
    }
}

public bool EstaEnUsoTipoMovimientoByNombre(string nombre)
```

```
{
    if (nombre == null)
        throw new MovimientoStockNuloException("No puede ser nulo el nombre del TipoMovimiento a buscar.");
    try
    {
        var tipMovs = _db.MovimientoStocks.Where(tipMov => tipMov.Movimiento.Nombre.Contains(nombre));
        return tipMovs != null;
    }
    catch (Exception ex)
    {
        throw new MovimientoStockNoValidoException(ex.Message);
    }
}

public IEnumerable<MovimientoStock> GetAll()
{
    var movimientos = _db.MovimientoStocks.Include(a => a.Articulo).Include(b => b.Movimiento).Include(c => c.UsuarioRealizaMovimiento).ToList();
    return movimientos;
}

//A) Dados un id de artículo y un tipo de movimiento, todos los movimientos de ese tipo
```

```
//realizados sobre ese artículo.Deberán estar ordenados descendentemente por fecha y en

//forma ascendente por la cantidad de unidades involucradas en el movimiento Se deberá

//mostrar todos los datos del movimiento, incluyendo todos los datos de su tipo.

public IEnumerable<MovimientoStock> GetAllByIDArticulo_y_TipoMovimiento(int idArticulo, string tipoMovimiento)

{

    if (idArticulo == null)

        throw new MovimientoStockNoValidoException("El ID del articulo por el cual quiere filtrar el movimiento de stock no puede ser nulo.");

    if (tipoMovimiento == null)

        throw new MovimientoStockNoValidoException("El tipo de movimiento por el cual quiere filtrar el movimiento de stock no puede ser nulo.");

    try

    {

        var movimientos = _db.MovimientoStocks.Where(mov => mov.Articulo.ID == idArticulo && mov.Movimiento.Nombre.ToLower().Equals(tipoMovimiento.ToLower()))

            .Include(mov => mov.Articulo)

            .Include(mov => mov.UsuarioRealizaMovimiento)

            .Include(mov => mov.Movimiento)

            .OrderByDescending(mov => mov.FecHorMovRealizado)

            .ThenBy(mov => mov.CtdUnidadesXMovimiento)

            .ToList();

        return movimientos;

    }

    catch (Exception ex)
```

```
    {
        throw new MovimientoStockNoValidoException(ex.Message);
    }
}

public MovimientoStock GetById(int id)
{
    if (id == null)
        throw new MovimientoStockNuloException("No puede ser nulo el ID del objeto a buscar.");
    try
    {
        MovimientoStock? movStock = _db.MovimientoStocks.Include(a => a.Articulo).Include(b => b.Movimiento).Include(c => c.UsuarioRealizaMovimiento).FirstOrDefault(art => art.ID == id);
        return movStock;
    }
    catch (Exception ex)
    {
        throw new MovimientoStockNoValidoException(ex.Message);
    }

}

public IEnumerable<Articulo> GetByRangoFechas(DateTime fechaIni, DateTime fechaFin)
{
```

```
        if (fechaIni == null)
            throw new MovimientoStockNoValidoException("La fecha de inicio no puede ser nula.");
        if (fechaFin == null)
            throw new MovimientoStockNoValidoException("La fecha final no puede ser nula.");
        if (fechaIni > fechaFin)
            throw new MovimientoStockNoValidoException("La fecha inicial no puede ser mayor a la fecha final.");
        try
        {
            var articulos = _db.MovimientoStocks
                    .Where(mov => mov.FecHorMovRealizado >= fechaIni && mov.FecHorMovRealizado <= fechaFin)
                    .Select(mov => mov.Articulo)
                    .ToList();
            return articulos;
        }
        catch (Exception ex)
        {
            throw new MovimientoStockNoValidoException(ex.Message);
        }
    }

    public IEnumerable<MovimientoStock> GetObjectsByID(List<int> ids)
    {
```

```
        throw new NotImplementedException();
    }

    /* public IEnumerable<object> ObtenerResumenMovimientosPorAnioYTipoMovimiento()
     {
        try
        {
            var resumen = _db.MovimientoStocks
                .GroupBy(mov => new { Year = mov.FecHorMovRealizado.Year, TipoMovimiento = mov.Movimiento.Nombre })
                .Select(group => new
                {
                    Anio = group.Key.Year,
                    Detalles = group.Select(g => new
                    {
                        Tipo = g.Movimiento.Nombre,
                        Cantidad = group.Count() // Contar la cantidad de filas por tipo de movimiento
                    }),
                    TotalAnio = group.Count() // Contar el total de filas por año
                })
                .OrderBy(res => res.Anio)
                .ThenBy(res => res.Detalles.Select(det => det.Tipo))
                .ToList();
```

```
            return resumen.Select(r => new
            {
                Año = r.Anio,
                Detalles = r.Detalles.Select(det => $"Tipo: \"{det.Tipo}\" - Cantidad: {det.Cantidad}"),
                TotalAnio = r.TotalAnio
            });
        }
        catch (Exception ex)
        {
            throw new MovimientoStockNoValidoException("Error al obtener el resumen de movimientos por año y tipo de movimiento");
        }
    }*/
    public IEnumerable<object> ObtenerResumenMovimientosPorAnioYTipoMovimiento()
    {
        try
        {
            var resumen = _db.MovimientoStocks
                .GroupBy(mov => mov.FecHorMovRealizado.Year)
                .Select(group => new
                {
                    Anio = group.Key,
                    Movimientos = group.GroupBy(mov => mov.Movimiento.Nombre).Select(tipoGroup => new {TipoMovimiento=tipoGroup.Key,
```

```
CantidadMovida=tipoGroup.Sum(mov=>mov.CtdUnidadesXMovimiento)}).OrderBy(tipo => tipo.TipoMovimiento).ToList()

                })

                .OrderBy(res => res.Anio)

                .ToList();



            return resumen;

        }

        catch (Exception ex)

        {

            throw new MovimientoStockNoValidoException("Error al obtener el resumen de movimientos por año y tipo de movimiento", ex);

        }

    }



    public void Remove(int id)

    {

        if (id == null)

            throw new MovimientoStockNuloException("No puede ser nulo el ID del objeto a remover.");

        try

        {

            var movStock = _db.MovimientoStocks.FirstOrDefault(u => u.ID == id);

            if (movStock != null)

            {
```

```
            _db.MovimientoStocks.Remove(movStock);
            _db.SaveChanges();
        }
    }
    catch (Exception ex)
    {
        throw new MovimientoStockNoValidoException(ex.Message);
    }

}

public void Remove(MovimientoStock obj)
{
    if (obj == null)
        throw new MovimientoStockNuloException("No puede ser nulo el OBJETO a remover.");
    try
    {
        _db.MovimientoStocks.Remove(obj);
        _db.SaveChanges();
    }
    catch (Exception ex)
    {
        throw new MovimientoStockNoValidoException(ex.Message);
```

```
    }


}


public void Update(int id, MovimientoStock obj)
{
    var movStock = _db.MovimientoStocks.FirstOrDefault(u => u.ID == id);


    if (movStock != null)
    {
        try
        {
            movStock.ModificarDatos(obj);
            _db.SaveChanges();
        }
        catch (Exception ex)
        {
            throw new MovimientoStockNoValidoException(ex.Message);
        }
    }
    else
    {
        throw new MovimientoStockNuloException("El movimiento de stock no existe.");
```

```
        }
      }
   }
}
```

****

Archivo: RepositorioTipoMovimientoEF.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.AccesoDatos\EF\RepositorioTipoMovimientoEF.cs

****

```
using Empresa.LogicaDeNegocio.Sistema;
using Papeleria.LogicaNegocio.Entidades;
using Papeleria.LogicaNegocio.Excepciones.MovimientoStock;
using Papeleria.LogicaNegocio.Excepciones.TipoMovimiento;
using Papeleria.LogicaNegocio.InterfacesRepositorio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Papeleria.AccesoDatos.EF
{
```

```
public class RepositorioTipoMovimientoEF : IRepositorioTipoMovimiento
{
    private PapeleriaContext _db;
    public RepositorioTipoMovimientoEF(PapeleriaContext context)
    {
        _db = context;
    }

    public void Add(TipoMovimiento obj)
    {
        if (obj == null)
            throw new TipoMovimientoNuloException("No puede ser nulo el objeto a agregar.");
        try
        {
            _db.TiposMovimientos.Add(obj);
            _db.SaveChanges();
        }
        catch (Exception ex)
        {
            throw new TipoMovimientoNoValidoException(ex.Message);
        }
    }
```

```
public bool ExisteTipoMovimientoXNombre(string nombre)
{
    TipoMovimiento tipMov = GetTipoMovimientoXNombre(nombre);
    return tipMov != null;
}

public IEnumerable<TipoMovimiento> GetAll()
{
    return _db.TiposMovimientos.ToList();
}

public TipoMovimiento GetById(int id)
{
    if (id == null)
        throw new TipoMovimientoNuloException("No puede ser nulo el ID del objeto a buscar.");
    try
    {
        TipoMovimiento? tipMov = _db.TiposMovimientos.FirstOrDefault(art => art.ID == id);
        return tipMov;
    }
    catch (Exception ex)
    {
        throw new TipoMovimientoNoValidoException(ex.Message);
```

```
    }
}

public IEnumerable<TipoMovimiento> GetObjectsByID(List<int> ids)
{
    throw new NotImplementedException();
}

public TipoMovimiento GetTipoMovimientoXNombre(string nombre)
{
    if (nombre == null)
        throw new TipoMovimientoNuloException("El nombre ingresado para GetTipoMovimientoXNombre no puede ser nulo.");
    return _db.TiposMovimientos.FirstOrDefault(u => u.Nombre == nombre);
}

public void Remove(int id)
{
    if (id == null)
        throw new TipoMovimientoNuloException("No puede ser nulo el ID del objeto a remover.");
    try
    {
        var tipMov = _db.TiposMovimientos.FirstOrDefault(u => u.ID == id);
        if (tipMov != null)
```

```
        {
            _db.TiposMovimientos.Remove(tipMov);
            _db.SaveChanges();
        }
    }
    catch (Exception ex)
    {
        throw new TipoMovimientoNoValidoException(ex.Message);
    }
}

public void Remove(TipoMovimiento obj)
{
    if (obj == null)
        throw new TipoMovimientoNuloException("No puede ser nulo el OBJETO a remover.");
    try
    {
        _db.TiposMovimientos.Remove(obj);
        _db.SaveChanges();
    }
    catch (Exception ex)
    {
        throw new TipoMovimientoNoValidoException(ex.Message);
```

```
        }
    }

    public void Update(int id, TipoMovimiento obj)
    {
        var movStock = _db.TiposMovimientos.FirstOrDefault(u => u.ID == id);
        if (movStock != null)
        {
            try
            {
                movStock.ModificarDatos(obj);
                _db.SaveChanges();
            }
            catch (Exception ex)
            {
                throw new TipoMovimientoNoValidoException(ex.Message);
            }
        }
        else
        {
            throw new TipoMovimientoNuloException("El ID ingresado no pertenece a ningun TipoMovimiento");
        }
    }
```

```
    }
}
```

****

Archivo: RepositorioUsuarioEF.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.AccesoDatos\EF\RepositorioUsuarioEF.cs

****

```
using Empresa.LogicaDeNegocio.Entidades;
using Empresa.LogicaDeNegocio.Sistema;
using Microsoft.EntityFrameworkCore;
using Papeleria.LogicaNegocio.Entidades;
using Papeleria.LogicaNegocio.Entidades.ValueObjects.Usuario;
using Papeleria.LogicaNegocio.Excepciones.EncargadoDeposito;
using Papeleria.LogicaNegocio.Excepciones.Usuario;
using Papeleria.LogicaNegocio.Excepciones.Usuario.UsuarioExcepcions.Constrasenia;
using Papeleria.LogicaNegocio.InterfacesRepositorio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
```

```
namespace Papeleria.AccesoDatos.EF
{
    public class RepositorioUsuarioEF : IRepositorioUsuario
    {
        private PapeleriaContext _db;
        public RepositorioUsuarioEF(PapeleriaContext context) { _db = context; }

        public Usuario GetUsuarioPorEmail(string email)
        {
            return _db.Usuarios.FirstOrDefault(u => u.Email.Direccion == email);
        }

        public void ModificarContrasenia(int id, ContraseniaUsuario contraseniaNueva)
        {
            var usuario = _db.Usuarios.FirstOrDefault(u => u.ID == id);

            if (usuario != null)
            {
                try
                {
                    usuario.Contrasenia = new ContraseniaUsuario(contraseniaNueva.Valor);
                    _db.SaveChanges();
                }
```

```
        catch (Exception ex)
        {
            throw new ContraseniaNoValidoException(ex.Message);
        }
    }
    else
    {
        throw new UsuarioNuloExcepcion("El usuario no existe");
    }
}

public Usuario GetById(int id)
{
    if (id == null)
    {
        throw new UsuarioNuloExcepcion("La ID a buscar no puede ser nula.");
    }
    Usuario? usuario = _db.Usuarios.FirstOrDefault(usr => usr.ID == id);
    return usuario;
}

public void Add(Usuario obj)
{
```

```
    if (obj == null) throw new UsuarioNuloExcepcion("El usuario es nulo");
    try
    {
        _db.Usuarios.Add(obj);
        _db.SaveChanges();
    }
    catch (Exception ex)
    {
        throw new UsuarioNoValidoExcepcion(ex.Message);
    }
}

public void Update(int id, Usuario obj)
{
    var usuario = _db.Usuarios.FirstOrDefault(u => u.ID == id);

    if (usuario != null)
    {
        try
        {
            usuario.ModificarDatos(obj);
            _db.SaveChanges();
        }
```

```
        catch (Exception ex)
        {

            throw new UsuarioNoValidoExcepcion(ex.Message);
        }
    }
    else
    {
        throw new UsuarioNuloExcepcion("El usuario no existe");
    }
}

public void Remove(int id)
{
    var usuario = _db.Usuarios.FirstOrDefault(u => u.ID == id);
    if (usuario != null)
    {
        _db.Usuarios.Remove(usuario);
        _db.SaveChanges();
    }
}

public void Remove(Usuario obj)
```

```
{
    _db.Usuarios.Remove(obj);
    _db.SaveChanges();
}

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

public IEnumerable<Usuario> GetObjectsByID(List<int> ids)
{
    throw new NotImplementedException();
}

public Usuario Login(string email, string contrasenia)
{
    try
    {
        var usuarios = _db.Usuarios.ToList();
        var usr = _db.Usuarios.AsEnumerable()
            .Where(u => u.Email.Direccion.Equals(email) && u.Contrasenia.Valor.Equals(contrasenia))
            .SingleOrDefault();
```

```
        if (usr == null)
        {
            throw new UsuarioNoValidoExcepcion("Usuario o contraseña incorrectos");
        }
        return usr;
    }
    catch (UsuarioNoValidoExcepcion ex)
    {
        throw new UsuarioNoValidoExcepcion(ex.Message);
    }
}

public bool ExisteUsuarioConEmail(string email)
{
    Usuario usuario = GetUsuarioPorEmail(email);
    return usuario != null;
}

public EncargadoDeposito GetEncargadoByID(int id)
{
    if (id == null)
    {
        throw new EncargadoNoValidoException("La ID a buscar no puede ser nula.");
```

```
        }

        EncargadoDeposito? usuario = (EncargadoDeposito?)_db.Usuarios.FirstOrDefault(usr => usr.ID == id);

        return usuario;

      }

   }

}
```

****

Archivo: ArgumentNullRepositorioException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.AccesoDatos\Excepciones\ArgumentNullRepositorioException.cs

****

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace Papeleria.AccesoDatos.Excepciones

{

   public class ArgumentNullRepositorioException : RepositorioException

   {
```

```
        public ArgumentNullRepositorioException() { }

        public ArgumentNullRepositorioException(string message) : base(message) { }

    }

}
```

****

Archivo: InfraException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.AccesoDatos\Excepciones\InfraException.cs

****

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace Papeleria.AccesoDatos.Excepciones

{

    public class InfraException : RepositorioException

    {

        public InfraException() { }

        public InfraException(string message) : base(message) { }

    }
```

```
}
```

****

Archivo: RepositorioException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.AccesoDatos\Excepciones\RepositorioException.cs

****

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace Papeleria.AccesoDatos.Excepciones

{

    public class RepositorioException : Exception

    {

        public RepositorioException() { }

        public RepositorioException(string message) : base(message) { }

    }

}
```

****

Archivo: 20240619200433_inicial.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.AccesoDatos\Migrations\20240619200433_inicial.cs

****

```
using System;
using Microsoft.EntityFrameworkCore.Migrations;


#nullable disable


namespace Papeleria.AccesoDatos.Migrations
{
    /// <inheritdoc />
    public partial class inicial : Migration
    {
        /// <inheritdoc />
        protected override void Up(MigrationBuilder migrationBuilder)
        {
            migrationBuilder.CreateTable(
                name: "Articulos",
                columns: table => new
                {
                    ID = table.Column<int>(type: "int", nullable: false)
```

```
            .Annotation("SqlServer:Identity", "1, 1"),
        PrecioVP = table.Column<double>(type: "float", nullable: false),
        CodigoProveedor_codigo = table.Column<long>(type: "bigint", nullable: false),
        Descripcion_Descripcion = table.Column<string>(type: "nvarchar(max)", nullable: false),
        NombreArticulo_Nombre = table.Column<string>(type: "nvarchar(max)", nullable: false)
    },
    constraints: table =>
    {
        table.PrimaryKey("PK_Articulos", x => x.ID);
    });

migrationBuilder.CreateTable(
    name: "TiposMovimientos",
    columns: table => new
    {
        ID = table.Column<int>(type: "int", nullable: false)
            .Annotation("SqlServer:Identity", "1, 1"),
        Nombre = table.Column<string>(type: "nvarchar(max)", nullable: false)
    },
    constraints: table =>
    {
        table.PrimaryKey("PK_TiposMovimientos", x => x.ID);
    });
```

```
migrationBuilder.CreateTable(
    name: "Usuarios",
    columns: table => new
    {
        ID = table.Column<int>(type: "int", nullable: false)
            .Annotation("SqlServer:Identity", "1, 1"),
        Discriminator = table.Column<string>(type: "nvarchar(21)", maxLength: 21, nullable: false),
        Contrasenia_Valor = table.Column<string>(type: "nvarchar(max)", nullable: false),
        Email_Direccion = table.Column<string>(type: "nvarchar(max)", nullable: false),
        NombreCompleto_Apellido = table.Column<string>(type: "nvarchar(max)", nullable: false),
        NombreCompleto_Nombre = table.Column<string>(type: "nvarchar(max)", nullable: false)
    },
    constraints: table =>
    {
        table.PrimaryKey("PK_Usuarios", x => x.ID);
    });

migrationBuilder.CreateTable(
    name: "MovimientoStocks",
    columns: table => new
    {
        ID = table.Column<int>(type: "int", nullable: false)
```

```
            .Annotation("SqlServer:Identity", "1, 1"),
        FecHorMovRealizado = table.Column<DateTime>(type: "datetime2", nullable: false),
        ArticuloID = table.Column<int>(type: "int", nullable: false),
        MovimientoID = table.Column<int>(type: "int", nullable: false),
        UsuarioRealizaMovimientoID = table.Column<int>(type: "int", nullable: false),
        CtdUnidadesXMovimiento = table.Column<int>(type: "int", nullable: false)
    },
    constraints: table =>
    {
        table.PrimaryKey("PK_MovimientoStocks", x => x.ID);
        table.ForeignKey(
            name: "FK_MovimientoStocks_Articulos_ArticuloID",
            column: x => x.ArticuloID,
            principalTable: "Articulos",
            principalColumn: "ID",
            onDelete: ReferentialAction.Cascade);
        table.ForeignKey(
            name: "FK_MovimientoStocks_TiposMovimientos_MovimientoID",
            column: x => x.MovimientoID,
            principalTable: "TiposMovimientos",
            principalColumn: "ID",
            onDelete: ReferentialAction.Cascade);
        table.ForeignKey(
```

```
            name: "FK_MovimientoStocks_Usuarios_UsuarioRealizaMovimientoID",
            column: x => x.UsuarioRealizaMovimientoID,
            principalTable: "Usuarios",
            principalColumn: "ID",
            onDelete: ReferentialAction.Cascade);
        });

    migrationBuilder.CreateIndex(
        name: "IX_MovimientoStocks_ArticuloID",
        table: "MovimientoStocks",
        column: "ArticuloID");

    migrationBuilder.CreateIndex(
        name: "IX_MovimientoStocks_MovimientoID",
        table: "MovimientoStocks",
        column: "MovimientoID");

    migrationBuilder.CreateIndex(
        name: "IX_MovimientoStocks_UsuarioRealizaMovimientoID",
        table: "MovimientoStocks",
        column: "UsuarioRealizaMovimientoID");
}
```

```
        /// <inheritdoc />
        protected override void Down(MigrationBuilder migrationBuilder)
        {
            migrationBuilder.DropTable(
                name: "MovimientoStocks");

            migrationBuilder.DropTable(
                name: "Articulos");

            migrationBuilder.DropTable(
                name: "TiposMovimientos");

            migrationBuilder.DropTable(
                name: "Usuarios");
        }
    }
}
```

****

Archivo: 20240619200433_inicial.Designer.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.AccesoDatos\Migrations\20240619200433_inicial.Designer.cs

****

```
// <auto-generated />
using System;
using System.Collections.Generic;
using Microsoft.EntityFrameworkCore;
using Microsoft.EntityFrameworkCore.Infrastructure;
using Microsoft.EntityFrameworkCore.Metadata;
using Microsoft.EntityFrameworkCore.Migrations;
using Microsoft.EntityFrameworkCore.Storage.ValueConversion;
using Papeleria.AccesoDatos.EF;

#nullable disable

namespace Papeleria.AccesoDatos.Migrations
{
    [DbContext(typeof(PapeleriaContext))]
    [Migration("20240619200433_inicial")]
    partial class inicial
    {
        /// <inheritdoc />
        protected override void BuildTargetModel(ModelBuilder modelBuilder)
        {
#pragma warning disable 612, 618
            modelBuilder
```

```
            .HasAnnotation("ProductVersion", "8.0.6")
            .HasAnnotation("Relational:MaxIdentifierLength", 128);

        SqlServerModelBuilderExtensions.UseIdentityColumns(modelBuilder);

        modelBuilder.Entity("Empresa.LogicaDeNegocio.Entidades.Articulo", b =>
            {
                b.Property<int>("ID")
                    .ValueGeneratedOnAdd()
                    .HasColumnType("int");

                SqlServerPropertyBuilderExtensions.UseIdentityColumn(b.Property<int>("ID"));

                b.Property<double>("PrecioVP")
                    .HasColumnType("float");

                b.ComplexProperty<Dictionary<string, object>>("CodigoProveedor",
"Empresa.LogicaDeNegocio.Entidades.Articulo.CodigoProveedor#CodigoProveedorArticulos", b1 =>
                    {
                        b1.IsRequired();

                        b1.Property<long>("codigo")
                            .HasColumnType("bigint");
                    });
```

```
            b.ComplexProperty<Dictionary<string, object>>("Descripcion",
"Empresa.LogicaDeNegocio.Entidades.Articulo.Descripcion#DescripcionArticulo", b1 =>
                {
                    b1.IsRequired();

                    b1.Property<string>("Descripcion")
                        .IsRequired()
                        .HasColumnType("nvarchar(max)");
                });

            b.ComplexProperty<Dictionary<string, object>>("NombreArticulo",
"Empresa.LogicaDeNegocio.Entidades.Articulo.NombreArticulo#NombreArticulo", b1 =>
                {
                    b1.IsRequired();

                    b1.Property<string>("Nombre")
                        .IsRequired()
                        .HasColumnType("nvarchar(max)");
                });

            b.HasKey("ID");

            b.ToTable("Articulos");
```

```
        });

    modelBuilder.Entity("Empresa.LogicaDeNegocio.Sistema.Usuario", b =>
        {
            b.Property<int>("ID")
                .ValueGeneratedOnAdd()
                .HasColumnType("int");

            SqlServerPropertyBuilderExtensions.UseIdentityColumn(b.Property<int>("ID"));

            b.Property<string>("Discriminator")
                .IsRequired()
                .HasMaxLength(21)
                .HasColumnType("nvarchar(21)");

            b.ComplexProperty<Dictionary<string, object>>("Contrasenia",
"Empresa.LogicaDeNegocio.Sistema.Usuario.Contrasenia#ContraseniaUsuario", b1 =>
                {
                    b1.IsRequired();

                    b1.Property<string>("Valor")
                        .IsRequired()
                        .HasColumnType("nvarchar(max)");
                });
```

```
            b.ComplexProperty<Dictionary<string, object>>("Email",
"Empresa.LogicaDeNegocio.Sistema.Usuario.Email#EmailUsuario", b1 =>
                {
                    b1.IsRequired();

                    b1.Property<string>("Direccion")
                        .IsRequired()
                        .HasColumnType("nvarchar(max)");
                });

            b.ComplexProperty<Dictionary<string, object>>("NombreCompleto",
"Empresa.LogicaDeNegocio.Sistema.Usuario.NombreCompleto#NombreCompleto", b1 =>
                {
                    b1.IsRequired();

                    b1.Property<string>("Apellido")
                        .IsRequired()
                        .HasColumnType("nvarchar(max)");

                    b1.Property<string>("Nombre")
                        .IsRequired()
                        .HasColumnType("nvarchar(max)");
                });
```

```
        b.HasKey("ID");

        b.ToTable("Usuarios");

        b.HasDiscriminator<string>("Discriminator").HasValue("Usuario");

        b.UseTphMappingStrategy();
    });

modelBuilder.Entity("Papeleria.LogicaNegocio.Entidades.MovimientoStock", b =>
    {
        b.Property<int>("ID")
            .ValueGeneratedOnAdd()
            .HasColumnType("int");

        SqlServerPropertyBuilderExtensions.UseIdentityColumn(b.Property<int>("ID"));

        b.Property<int>("ArticuloID")
            .HasColumnType("int");

        b.Property<int>("CtdUnidadesXMovimiento")
            .HasColumnType("int");
```

```
            b.Property<DateTime>("FecHorMovRealizado")
                .HasColumnType("datetime2");

            b.Property<int>("MovimientoID")
                .HasColumnType("int");

            b.Property<int>("UsuarioRealizaMovimientoID")
                .HasColumnType("int");

            b.HasKey("ID");

            b.HasIndex("ArticuloID");

            b.HasIndex("MovimientoID");

            b.HasIndex("UsuarioRealizaMovimientoID");

            b.ToTable("MovimientoStocks");
        });

    modelBuilder.Entity("Papeleria.LogicaNegocio.Entidades.TipoMovimiento", b =>
        {
```

```
        b.Property<int>("ID")
            .ValueGeneratedOnAdd()
            .HasColumnType("int");

        SqlServerPropertyBuilderExtensions.UseIdentityColumn(b.Property<int>("ID"));

        b.Property<string>("Nombre")
            .IsRequired()
            .HasColumnType("nvarchar(max)");

        b.HasKey("ID");

        b.ToTable("TiposMovimientos");
    });

modelBuilder.Entity("Papeleria.LogicaNegocio.Entidades.EncargadoDeposito", b =>
    {
        b.HasBaseType("Empresa.LogicaDeNegocio.Sistema.Usuario");

        b.HasDiscriminator().HasValue("EncargadoDeposito");
    });

modelBuilder.Entity("Papeleria.LogicaNegocio.Entidades.MovimientoStock", b =>
```

```
        {
            b.HasOne("Empresa.LogicaDeNegocio.Entidades.Articulo", "Articulo")
                .WithMany()
                .HasForeignKey("ArticuloID")
                .OnDelete(DeleteBehavior.Cascade)
                .IsRequired();

            b.HasOne("Papeleria.LogicaNegocio.Entidades.TipoMovimiento", "Movimiento")
                .WithMany()
                .HasForeignKey("MovimientoID")
                .OnDelete(DeleteBehavior.Cascade)
                .IsRequired();

            b.HasOne("Papeleria.LogicaNegocio.Entidades.EncargadoDeposito",
"UsuarioRealizaMovimiento")
                .WithMany()
                .HasForeignKey("UsuarioRealizaMovimientoID")
                .OnDelete(DeleteBehavior.Cascade)
                .IsRequired();

            b.Navigation("Articulo");

            b.Navigation("Movimiento");
```

```
                b.Navigation("UsuarioRealizaMovimiento");
            });
#pragma warning restore 612, 618
        }
    }
}
```

****

Archivo: 20240620035748_discriminator.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.AccesoDatos\Migrations\20240620035748_discriminator.cs

****

```
using Microsoft.EntityFrameworkCore.Migrations;

#nullable disable

namespace Papeleria.AccesoDatos.Migrations
{
    /// <inheritdoc />
    public partial class discriminator : Migration
    {
        /// <inheritdoc />
        protected override void Up(MigrationBuilder migrationBuilder)
```

```
        {
            migrationBuilder.RenameColumn(
                name: "Discriminator",
                table: "Usuarios",
                newName: "Tipo");
        }

        /// <inheritdoc />
        protected override void Down(MigrationBuilder migrationBuilder)
        {
            migrationBuilder.RenameColumn(
                name: "Tipo",
                table: "Usuarios",
                newName: "Discriminator");
        }
    }
}
```

****

Archivo: 20240620035748_discriminator.Designer.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.AccesoDatos\Migrations\20240620035748_discriminator.Designer.cs

****

```
// <auto-generated />
using System;
using System.Collections.Generic;
using Microsoft.EntityFrameworkCore;
using Microsoft.EntityFrameworkCore.Infrastructure;
using Microsoft.EntityFrameworkCore.Metadata;
using Microsoft.EntityFrameworkCore.Migrations;
using Microsoft.EntityFrameworkCore.Storage.ValueConversion;
using Papeleria.AccesoDatos.EF;

#nullable disable

namespace Papeleria.AccesoDatos.Migrations
{
    [DbContext(typeof(PapeleriaContext))]
    [Migration("20240620035748_discriminator")]
    partial class discriminator
    {
        /// <inheritdoc />
        protected override void BuildTargetModel(ModelBuilder modelBuilder)
        {
#pragma warning disable 612, 618
            modelBuilder
```

```
            .HasAnnotation("ProductVersion", "8.0.6")
            .HasAnnotation("Relational:MaxIdentifierLength", 128);

        SqlServerModelBuilderExtensions.UseIdentityColumns(modelBuilder);

        modelBuilder.Entity("Empresa.LogicaDeNegocio.Entidades.Articulo", b =>
            {
                b.Property<int>("ID")
                    .ValueGeneratedOnAdd()
                    .HasColumnType("int");

                SqlServerPropertyBuilderExtensions.UseIdentityColumn(b.Property<int>("ID"));

                b.Property<double>("PrecioVP")
                    .HasColumnType("float");

                b.ComplexProperty<Dictionary<string, object>>("CodigoProveedor",
"Empresa.LogicaDeNegocio.Entidades.Articulo.CodigoProveedor#CodigoProveedorArticulos", b1 =>
                    {
                        b1.IsRequired();

                        b1.Property<long>("codigo")
                            .HasColumnType("bigint");
                    });
```

```
            b.ComplexProperty<Dictionary<string, object>>("Descripcion",
"Empresa.LogicaDeNegocio.Entidades.Articulo.Descripcion#DescripcionArticulo", b1 =>
                {
                    b1.IsRequired();

                    b1.Property<string>("Descripcion")
                        .IsRequired()
                        .HasColumnType("nvarchar(max)");
                });

            b.ComplexProperty<Dictionary<string, object>>("NombreArticulo",
"Empresa.LogicaDeNegocio.Entidades.Articulo.NombreArticulo#NombreArticulo", b1 =>
                {
                    b1.IsRequired();

                    b1.Property<string>("Nombre")
                        .IsRequired()
                        .HasColumnType("nvarchar(max)");
                });

            b.HasKey("ID");

            b.ToTable("Articulos");
```

```
        });

    modelBuilder.Entity("Empresa.LogicaDeNegocio.Sistema.Usuario", b =>
        {
            b.Property<int>("ID")
                .ValueGeneratedOnAdd()
                .HasColumnType("int");

            SqlServerPropertyBuilderExtensions.UseIdentityColumn(b.Property<int>("ID"));

            b.Property<string>("Tipo")
                .IsRequired()
                .HasMaxLength(21)
                .HasColumnType("nvarchar(21)");

            b.ComplexProperty<Dictionary<string, object>>("Contrasenia",
"Empresa.LogicaDeNegocio.Sistema.Usuario.Contrasenia#ContraseniaUsuario", b1 =>
                {
                    b1.IsRequired();

                    b1.Property<string>("Valor")
                        .IsRequired()
                        .HasColumnType("nvarchar(max)");
                });
```

```
            b.ComplexProperty<Dictionary<string, object>>("Email",
"Empresa.LogicaDeNegocio.Sistema.Usuario.Email#EmailUsuario", b1 =>
                {
                    b1.IsRequired();

                    b1.Property<string>("Direccion")
                        .IsRequired()
                        .HasColumnType("nvarchar(max)");
                });

            b.ComplexProperty<Dictionary<string, object>>("NombreCompleto",
"Empresa.LogicaDeNegocio.Sistema.Usuario.NombreCompleto#NombreCompleto", b1 =>
                {
                    b1.IsRequired();

                    b1.Property<string>("Apellido")
                        .IsRequired()
                        .HasColumnType("nvarchar(max)");

                    b1.Property<string>("Nombre")
                        .IsRequired()
                        .HasColumnType("nvarchar(max)");
                });
```

```
        b.HasKey("ID");

        b.ToTable("Usuarios");

        b.HasDiscriminator<string>("Tipo").HasValue("Usuario");

        b.UseTphMappingStrategy();
    });

modelBuilder.Entity("Papeleria.LogicaNegocio.Entidades.MovimientoStock", b =>
    {
        b.Property<int>("ID")
            .ValueGeneratedOnAdd()
            .HasColumnType("int");

        SqlServerPropertyBuilderExtensions.UseIdentityColumn(b.Property<int>("ID"));

        b.Property<int>("ArticuloID")
            .HasColumnType("int");

        b.Property<int>("CtdUnidadesXMovimiento")
            .HasColumnType("int");
```

```
            b.Property<DateTime>("FecHorMovRealizado")
                .HasColumnType("datetime2");

            b.Property<int>("MovimientoID")
                .HasColumnType("int");

            b.Property<int>("UsuarioRealizaMovimientoID")
                .HasColumnType("int");

            b.HasKey("ID");

            b.HasIndex("ArticuloID");

            b.HasIndex("MovimientoID");

            b.HasIndex("UsuarioRealizaMovimientoID");

            b.ToTable("MovimientoStocks");
        });

    modelBuilder.Entity("Papeleria.LogicaNegocio.Entidades.TipoMovimiento", b =>
        {
```

```
            b.Property<int>("ID")
                .ValueGeneratedOnAdd()
                .HasColumnType("int");

            SqlServerPropertyBuilderExtensions.UseIdentityColumn(b.Property<int>("ID"));

            b.Property<string>("Nombre")
                .IsRequired()
                .HasColumnType("nvarchar(max)");

            b.HasKey("ID");

            b.ToTable("TiposMovimientos");
        });

    modelBuilder.Entity("Papeleria.LogicaNegocio.Entidades.Administrador", b =>
        {
            b.HasBaseType("Empresa.LogicaDeNegocio.Sistema.Usuario");

            b.HasDiscriminator().HasValue("Administrador");
        });

    modelBuilder.Entity("Papeleria.LogicaNegocio.Entidades.EncargadoDeposito", b =>
```

```
    {
        b.HasBaseType("Empresa.LogicaDeNegocio.Sistema.Usuario");

        b.HasDiscriminator().HasValue("EncargadoDeposito");
    });

modelBuilder.Entity("Papeleria.LogicaNegocio.Entidades.MovimientoStock", b =>
    {
        b.HasOne("Empresa.LogicaDeNegocio.Entidades.Articulo", "Articulo")
            .WithMany()
            .HasForeignKey("ArticuloID")
            .OnDelete(DeleteBehavior.Cascade)
            .IsRequired();

        b.HasOne("Papeleria.LogicaNegocio.Entidades.TipoMovimiento", "Movimiento")
            .WithMany()
            .HasForeignKey("MovimientoID")
            .OnDelete(DeleteBehavior.Cascade)
            .IsRequired();

        b.HasOne("Empresa.LogicaDeNegocio.Sistema.Usuario", "UsuarioRealizaMovimiento")
            .WithMany()
            .HasForeignKey("UsuarioRealizaMovimientoID")
```

```
                    .OnDelete(DeleteBehavior.Cascade)

                    .IsRequired();


                b.Navigation("Articulo");


                b.Navigation("Movimiento");


                b.Navigation("UsuarioRealizaMovimiento");
            });
#pragma warning restore 612, 618
        }
    }
}
```

****

Archivo: PapeleriaContextModelSnapshot.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.AccesoDatos\Migrations\PapeleriaContextModelSnapshot.cs

****

```
// <auto-generated />
using System;
using System.Collections.Generic;
using Microsoft.EntityFrameworkCore;
```

```
using Microsoft.EntityFrameworkCore.Infrastructure;
using Microsoft.EntityFrameworkCore.Metadata;
using Microsoft.EntityFrameworkCore.Storage.ValueConversion;
using Papeleria.AccesoDatos.EF;

#nullable disable

namespace Papeleria.AccesoDatos.Migrations
{
    [DbContext(typeof(PapeleriaContext))]
    partial class PapeleriaContextModelSnapshot : ModelSnapshot
    {
        protected override void BuildModel(ModelBuilder modelBuilder)
        {
#pragma warning disable 612, 618
            modelBuilder
                .HasAnnotation("ProductVersion", "8.0.6")
                .HasAnnotation("Relational:MaxIdentifierLength", 128);

            SqlServerModelBuilderExtensions.UseIdentityColumns(modelBuilder);

            modelBuilder.Entity("Empresa.LogicaDeNegocio.Entidades.Articulo", b =>
                {
```

```
            b.Property<int>("ID")
                .ValueGeneratedOnAdd()
                .HasColumnType("int");

            SqlServerPropertyBuilderExtensions.UseIdentityColumn(b.Property<int>("ID"));

            b.Property<double>("PrecioVP")
                .HasColumnType("float");

            b.ComplexProperty<Dictionary<string, object>>("CodigoProveedor",
"Empresa.LogicaDeNegocio.Entidades.Articulo.CodigoProveedor#CodigoProveedorArticulos", b1 =>
                {
                    b1.IsRequired();

                    b1.Property<long>("codigo")
                        .HasColumnType("bigint");
                });

            b.ComplexProperty<Dictionary<string, object>>("Descripcion",
"Empresa.LogicaDeNegocio.Entidades.Articulo.Descripcion#DescripcionArticulo", b1 =>
                {
                    b1.IsRequired();

                    b1.Property<string>("Descripcion")
```

```
                .IsRequired()
                .HasColumnType("nvarchar(max)");
            });

        b.ComplexProperty<Dictionary<string, object>>("NombreArticulo",
"Empresa.LogicaDeNegocio.Entidades.Articulo.NombreArticulo#NombreArticulo", b1 =>
            {
                b1.IsRequired();

                b1.Property<string>("Nombre")
                .IsRequired()
                .HasColumnType("nvarchar(max)");
            });

        b.HasKey("ID");

        b.ToTable("Articulos");
    });

modelBuilder.Entity("Empresa.LogicaDeNegocio.Sistema.Usuario", b =>
    {
        b.Property<int>("ID")
            .ValueGeneratedOnAdd()
            .HasColumnType("int");
```

```
            SqlServerPropertyBuilderExtensions.UseIdentityColumn(b.Property<int>("ID"));

            b.Property<string>("Tipo")
                .IsRequired()
                .HasMaxLength(21)
                .HasColumnType("nvarchar(21)");

            b.ComplexProperty<Dictionary<string, object>>("Contrasenia",
"Empresa.LogicaDeNegocio.Sistema.Usuario.Contrasenia#ContraseniaUsuario", b1 =>
                {
                    b1.IsRequired();

                    b1.Property<string>("Valor")
                        .IsRequired()
                        .HasColumnType("nvarchar(max)");
                });

            b.ComplexProperty<Dictionary<string, object>>("Email",
"Empresa.LogicaDeNegocio.Sistema.Usuario.Email#EmailUsuario", b1 =>
                {
                    b1.IsRequired();

                    b1.Property<string>("Direccion")
```

```
                .IsRequired()
                .HasColumnType("nvarchar(max)");
            });

        b.ComplexProperty<Dictionary<string, object>>("NombreCompleto",
"Empresa.LogicaDeNegocio.Sistema.Usuario.NombreCompleto#NombreCompleto", b1 =>
            {
                b1.IsRequired();

                b1.Property<string>("Apellido")
                    .IsRequired()
                    .HasColumnType("nvarchar(max)");

                b1.Property<string>("Nombre")
                    .IsRequired()
                    .HasColumnType("nvarchar(max)");
            });

        b.HasKey("ID");

        b.ToTable("Usuarios");

        b.HasDiscriminator<string>("Tipo").HasValue("Usuario");
```

```
        b.UseTphMappingStrategy();
    });

modelBuilder.Entity("Papeleria.LogicaNegocio.Entidades.MovimientoStock", b =>
    {
        b.Property<int>("ID")
            .ValueGeneratedOnAdd()
            .HasColumnType("int");

        SqlServerPropertyBuilderExtensions.UseIdentityColumn(b.Property<int>("ID"));

        b.Property<int>("ArticuloID")
            .HasColumnType("int");

        b.Property<int>("CtdUnidadesXMovimiento")
            .HasColumnType("int");

        b.Property<DateTime>("FecHorMovRealizado")
            .HasColumnType("datetime2");

        b.Property<int>("MovimientoID")
            .HasColumnType("int");
```

```
        b.Property<int>("UsuarioRealizaMovimientoID")
            .HasColumnType("int");

        b.HasKey("ID");

        b.HasIndex("ArticuloID");

        b.HasIndex("MovimientoID");

        b.HasIndex("UsuarioRealizaMovimientoID");

        b.ToTable("MovimientoStocks");
    });

modelBuilder.Entity("Papeleria.LogicaNegocio.Entidades.TipoMovimiento", b =>
    {
        b.Property<int>("ID")
            .ValueGeneratedOnAdd()
            .HasColumnType("int");

        SqlServerPropertyBuilderExtensions.UseIdentityColumn(b.Property<int>("ID"));

        b.Property<string>("Nombre")
```

```
            .IsRequired()

            .HasColumnType("nvarchar(max)");


        b.HasKey("ID");


        b.ToTable("TiposMovimientos");
    });

modelBuilder.Entity("Papeleria.LogicaNegocio.Entidades.Administrador", b =>
    {
        b.HasBaseType("Empresa.LogicaDeNegocio.Sistema.Usuario");


        b.HasDiscriminator().HasValue("Administrador");
    });

modelBuilder.Entity("Papeleria.LogicaNegocio.Entidades.EncargadoDeposito", b =>
    {
        b.HasBaseType("Empresa.LogicaDeNegocio.Sistema.Usuario");


        b.HasDiscriminator().HasValue("EncargadoDeposito");
    });

modelBuilder.Entity("Papeleria.LogicaNegocio.Entidades.MovimientoStock", b =>
```

```
{
    b.HasOne("Empresa.LogicaDeNegocio.Entidades.Articulo", "Articulo")
        .WithMany()
        .HasForeignKey("ArticuloID")
        .OnDelete(DeleteBehavior.Cascade)
        .IsRequired();

    b.HasOne("Papeleria.LogicaNegocio.Entidades.TipoMovimiento", "Movimiento")
        .WithMany()
        .HasForeignKey("MovimientoID")
        .OnDelete(DeleteBehavior.Cascade)
        .IsRequired();

    b.HasOne("Empresa.LogicaDeNegocio.Sistema.Usuario", "UsuarioRealizaMovimiento")
        .WithMany()
        .HasForeignKey("UsuarioRealizaMovimientoID")
        .OnDelete(DeleteBehavior.Cascade)
        .IsRequired();

    b.Navigation("Articulo");

    b.Navigation("Movimiento");
```

```
                b.Navigation("UsuarioRealizaMovimiento");
            });
#pragma warning restore 612, 618
        }
    }
}
```

****

Archivo: IAlta.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasoDeUsoGeneral\IAlta.cs

****

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

namespace Papeleria.LogicaAplicacion.Interaces
{
    public interface IAlta<T>
    {
        public void Crear(T obj);
```

```
    }
}
```

****

Archivo: IGet.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasoDeUsoGeneral\IGet.cs

****

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

namespace Papeleria.LogicaAplicacion.InterfacesCasoDeUsoGeneral
{
    public interface IGet<T>
    {
        public T Get(int id);
        public T Get(string id);
        public IEnumerable<T> GetMany(string id);
        public IEnumerable<T> GetAll(IRepositorio<T> repo);
```

```
    }
}
```

****

Archivo: IGetAll.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasoDeUsoGeneral\IGetAll.cs

****

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

namespace Papeleria.LogicaAplicacion.InterfacesCasoDeUsoGeneral
{
    public interface IGetAll<T>
    {
        public IEnumerable<T> GetAll(IRepositorio<T> repo);
    }
}
```

****

Archivo: IListar.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasoDeUsoGeneral\IListar.cs

****

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


namespace Papeleria.LogicaAplicacion.Interaces
{
    public interface IListar<T>
    {
        public IEnumerable<T> ListarTodo();
        public T ListarUno(int id);
        public IEnumerable<T> ListarPorNombre(string name);
        public T ListarUnoPorNombre(string nombre);
        public List<T> ListarSeleccionPorId(List<int> ids);
    }
}
```

****

Archivo: IRemove.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasoDeUsoGeneral\IRemove.cs

****

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


namespace Papeleria.LogicaAplicacion.Interaces
{
    public interface IRemove<T>
    {
        public void Remove(T obj);
    }
}
```

****

Archivo: IUpdate.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasoDeUsoGeneral\IUpdate.cs

****

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace Papeleria.LogicaAplicacion.Interaces

{

   public interface IUpdate<T>

   {

      public void Update(int id, T obj);

   }

}
```

****

Archivo: Administrador.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Entidades\Administrador.cs

****

```
using Empresa.LogicaDeNegocio.Sistema;

using Papeleria.LogicaNegocio.Entidades.ValueObjects.Usuario;

using System;

using System.Collections.Generic;
```

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


namespace Papeleria.LogicaNegocio.Entidades
{
    public class Administrador : Usuario
    {
        public Administrador()
        {
        }


        public Administrador(string email, string nombre, string apellido, string contrasenia) : base(email, nombre, apellido, contrasenia)
        {
            this.Email = new EmailUsuario(email);
            this.NombreCompleto = new NombreCompleto(nombre, apellido);
            this.Contrasenia = new ContraseniaUsuario(contrasenia);
            esValido();
        }


    }
}
```

****

Archivo: Articulo.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Entidades\Articulo.cs

****

```
using Papeleria.LogicaNegocio.Entidades.ValueObjects.Articulos;
using Papeleria.LogicaNegocio.Excepciones.Articulo;
using Papeleria.LogicaNegocio.Excepciones.Articulo.ArticulosValueObjects.CodigoProveedor;
using Papeleria.LogicaNegocio.Excepciones.Articulo.ArticulosValueObjects.DescripcionArticulo;
using Papeleria.LogicaNegocio.Excepciones.Usuario.UsuarioExcepcions.Nombre;
using Papeleria.LogicaNegocio.InterfacesEntidades;
using System.Collections.Generic;
using System.Timers;


namespace Empresa.LogicaDeNegocio.Entidades
{
   public class Articulo : IValidable<Articulo>, IEquatable<Articulo>, IEntity
        {
      public int ID { get; set; }
                public NombreArticulo NombreArticulo{ get; set; }
      public CodigoProveedorArticulos CodigoProveedor{ get; set; }
                public DescripcionArticulo Descripcion{ get; set; }
                public double PrecioVP{ get; set; }
```

```
public Articulo(long codigoProveedor, string nombre, string descripcion, double precioVP)
{
    this.CodigoProveedor = new CodigoProveedorArticulos(codigoProveedor);
    this.NombreArticulo = new NombreArticulo(nombre);
    this.Descripcion = new DescripcionArticulo(descripcion);
    this.PrecioVP = precioVP;
    esValido();
}

public Articulo()
{

}

public bool Equals(Articulo? other)
{
    if (other == null) return false;
    return this.NombreArticulo == other.NombreArticulo;
}

public void esValido()
{
```

```
        if (CodigoProveedor==null) {
            throw new CodigoProveedorNuloException("El codigo de proveedor no puede ser nulo.");
        }
        if (NombreArticulo==null) {
            throw new NombreNuloException("El nombre del articulo no puede ser nulo.");
        }
        if (Descripcion==null) {
            throw new DescripcionArticuloNuloException("La descripcion no puede ser nula.");
        }
    }
    public void CambiarPrecioVP(int nuevoPrecio){
        PrecioVP = nuevoPrecio;
    }

    public void ModificarDatos(Articulo obj)
    {
        this.CodigoProveedor = obj.CodigoProveedor;
        this.NombreArticulo = obj.NombreArticulo;
        this.Descripcion = obj.Descripcion;
        this.PrecioVP = obj.PrecioVP;
    }
  }
}
```

****

Archivo: EncargadoDeposito.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Entidades\EncargadoDeposito.cs

****

```
using Empresa.LogicaDeNegocio.Sistema;
using Papeleria.LogicaNegocio.Entidades.ValueObjects.Usuario;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;


namespace Papeleria.LogicaNegocio.Entidades
{
    public class EncargadoDeposito : Usuario
    {
        public EncargadoDeposito()
        {
        }
```

```
        public EncargadoDeposito(string email, string nombre, string apellido, string contrasenia) : base(email, nombre, apellido, contrasenia)
        {
            this.Email = new EmailUsuario(email);
            this.NombreCompleto = new NombreCompleto(nombre, apellido);
            this.Contrasenia = new ContraseniaUsuario(contrasenia);
            esValido();
        }
        public override void ModificarContraseña(string contrasenia)
        {
            base.ModificarContraseña(contrasenia);
        }
        public override void ModificarDatos(EncargadoDeposito usu)
        {
            base.ModificarDatos(usu);
        }

    }
}
```

****

Archivo: MovimientoStock.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Entidades\MovimientoStock.cs

****

```
using Empresa.LogicaDeNegocio.Entidades;
using Empresa.LogicaDeNegocio.Sistema;
using Papeleria.LogicaNegocio.Excepciones.MovimientoStock;
using Papeleria.LogicaNegocio.InterfacesEntidades;
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations.Schema;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Papeleria.LogicaNegocio.Entidades
{
    public class MovimientoStock : IEntity, IValidable<MovimientoStock>
    {
        public int ID { get; set; }
        public DateTime FecHorMovRealizado { get; set; }
        public Articulo Articulo { get; set; }
        public TipoMovimiento Movimiento { get; set; }
        public Usuario UsuarioRealizaMovimiento { get; set; }
```

```
public int CtdUnidadesXMovimiento { get; set; }

public MovimientoStock(Articulo articulo, TipoMovimiento movimiento, Usuario usuarioRealizaMovimiento, int ctdUnidadesXMovimiento)
{
    FecHorMovRealizado = DateTime.Now;
    Articulo = articulo;
    Movimiento = movimiento;
    UsuarioRealizaMovimiento = usuarioRealizaMovimiento;
    CtdUnidadesXMovimiento = ctdUnidadesXMovimiento;
}

public MovimientoStock()
{

}

public void esValido()
{
    if (Articulo == null)
    {
        throw new MovimientoStockNuloException("Articulo no puede ser nulo");
    }
    if (Movimiento == null)
```

```
        {
            throw new MovimientoStockNuloException("Movimiento no puede ser nulo");
        }
        if (UsuarioRealizaMovimiento == null)
        {
            throw new MovimientoStockNuloException("Usuario no puede ser nulo");
        }
        if (CtdUnidadesXMovimiento == null)
        {
            throw new MovimientoStockNuloException("Cantidad de unidades a mover no puede ser nulo");
        }
    }

    public void ModificarDatos(MovimientoStock obj)
    {
        FecHorMovRealizado = obj.FecHorMovRealizado;
        Articulo = obj.Articulo;
        Movimiento = obj.Movimiento;
        UsuarioRealizaMovimiento = obj.UsuarioRealizaMovimiento;
        CtdUnidadesXMovimiento = obj.CtdUnidadesXMovimiento;
    }

}
```

}

****

Archivo: TipoMovimiento.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Entidades\TipoMovimiento.cs

****

```
using Empresa.LogicaDeNegocio.Entidades;
using Papeleria.LogicaNegocio.Excepciones.TipoMovimiento;
using Papeleria.LogicaNegocio.InterfacesEntidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Papeleria.LogicaNegocio.Entidades
{
    public class TipoMovimiento : IEntity, IValidable<TipoMovimiento>, IEquatable<TipoMovimiento>
    {
        public int ID { get; set; }
        public string Nombre { get; set; }
```

```
public TipoMovimiento(string nombre)
{
    Nombre = nombre;
    esValido();
}

public TipoMovimiento()
{

}

public void esValido()
{
    if (Nombre == null) {
        throw new TipoMovimientoNoValidoException("Nombre no puede ser nulo.");
    }
}

public void ModificarDatos(TipoMovimiento obj)
{
    this.Nombre = obj.Nombre;
}
```

```
        public bool Equals(TipoMovimiento? other)
        {
            if (other == null) return false;
            return this.Nombre == other.Nombre;
        }
    }
}
```

****

Archivo: Usuario.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Entidades\Usuario.cs

****

```
using Empresa.LogicaDeNegocio.Entidades;
using Papeleria.LogicaNegocio.Entidades;
using Papeleria.LogicaNegocio.Entidades.ValueObjects.Usuario;
using Papeleria.LogicaNegocio.Excepciones.Usuario;
using Papeleria.LogicaNegocio.Excepciones.Usuario.UsuarioExcepcions.Constrasenia;
using Papeleria.LogicaNegocio.InterfacesEntidades;

namespace Empresa.LogicaDeNegocio.Sistema
```

```
{
    public abstract class Usuario : IValidable<Usuario>, IEquatable<Usuario>, IEntity
        {
        public int ID { get; set; }
        public EmailUsuario Email{ get; set; }

                public NombreCompleto NombreCompleto{ get; set; }
                public ContraseniaUsuario Contrasenia{ get; set; }

        public Usuario(string email, string nombre, string apellido, string contrasenia)
        {
            this.Email = new EmailUsuario(email);
            this.NombreCompleto = new NombreCompleto(nombre, apellido);
            this.Contrasenia = new ContraseniaUsuario(contrasenia);
            esValido();
        }
        public Usuario()
        {

        }
        public bool Equals(Usuario? other)
        {
            if (other == null)
```

```
            throw new UsuarioNoValidoExcepcion("Debe incluir el autor a comparar");


        return this.ID == other.ID || this.Email == other.Email;
    }


    public void esValido()
    {
        esValido(this);
    }


    public void esValido(Usuario usuario)
    {
        if(Email == null) {
            throw new UsuarioNoValidoExcepcion("El email no puede ser nulo para crear el usuario.");
        }
        if(NombreCompleto == null) {
            throw new UsuarioNoValidoExcepcion("El nombre y apellido no puede ser nulo para crear el usuario.");
        }
        if(Contrasenia == null) {
            throw new UsuarioNoValidoExcepcion("La contrase□a no puede ser nula para crear el usuario.");
        }
    }
```

```
public virtual void ModificarContrase□a(string contrasenia)
{
    if (contrasenia == null)
        throw new ContraseniaNuloException("La contrase□a no puede ser nula.");
    this.Contrasenia = new ContraseniaUsuario(contrasenia);
}

public virtual void ModificarDatos(Usuario usu) {
    if (usu.Contrasenia == null)
        throw new ContraseniaNuloException("La contrase□a no puede ser nula.");
    if (usu.NombreCompleto == null)
        throw new ContraseniaNuloException("La contrase□a no puede ser nula.");
    this.NombreCompleto = usu.NombreCompleto;
    this.Contrasenia = usu.Contrasenia;
}
public virtual void ModificarDatos(EncargadoDeposito usu)
{
    if (usu.Contrasenia == null)
        throw new ContraseniaNuloException("La contrase□a no puede ser nula.");
    if (usu.NombreCompleto == null)
        throw new ContraseniaNuloException("La contrase□a no puede ser nula.");
    this.NombreCompleto = usu.NombreCompleto;
    this.Contrasenia = usu.Contrasenia;
```

```
        }
    }


}
```

****

Archivo: IEntity.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\InterfacesEntidades\IEntity.cs

****

```
using Empresa.LogicaDeNegocio.Sistema;
using Papeleria.LogicaNegocio.Entidades.ValueObjects;


namespace Empresa.LogicaDeNegocio.Entidades
{
        public interface IEntity
        {
                public int ID { get; set; }
    }


}
```

****

Archivo: IValidable.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\InterfacesEntidades\IValidable.cs

****

```
using Empresa.LogicaDeNegocio.Entidades;
using Papeleria.LogicaNegocio.Entidades.ValueObjects;
using Papeleria.LogicaNegocio.Entidades.ValueObjects.Usuario;


namespace Papeleria.LogicaNegocio.InterfacesEntidades
{
    public interface IValidable<T> where T : class
    {
        void esValido();


    }


}
```

****

Archivo: IRepositorio.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\InterfacesRepositorio\IRepositorio.cs

****

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


namespace Papeleria.LogicaNegocio.InterfacesRepositorio
{
    public interface IRepositorio<T>
    {
        public T GetById(int id);
        public void Add(T obj);
        public void Update(int id, T obj);
        public void Remove(int id);
        public void Remove(T obj);
        public IEnumerable<T> GetAll();
        public IEnumerable<T> GetObjectsByID(List<int> ids);
    }
}
```

****

Archivo: IRepositorioArticulo.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\InterfacesRepositorio\IRepositorioArticulo.cs

****

```
using Empresa.LogicaDeNegocio.Entidades;
using Papeleria.LogicaNegocio.Entidades.ValueObjects.Articulos;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;


namespace Papeleria.LogicaNegocio.InterfacesRepositorio
{
    public interface IRepositorioArticulo:IRepositorio<Articulo>
    {
        public IEnumerable<Articulo> GetArticulosOrdenadosAlfabeticamente();
        public Articulo GetArticuloByCodigo(CodigoProveedorArticulos codigo);
        public bool ExisteArticuloConNombre(string nombre);
    }
}
```

****

Archivo: IRepositorioMovimientoStock.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\InterfacesRepositorio\IRepositorioMovimientoStock.cs

****

```
using Empresa.LogicaDeNegocio.Entidades;
using Papeleria.LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.InteropServices;
using System.Text;
using System.Threading.Tasks;


namespace Papeleria.LogicaNegocio.InterfacesRepositorio
{
    public interface IRepositorioMovimientoStock : IRepositorio<MovimientoStock>
    {
        public bool EstaEnUsoTipoMovimientoByID(int id);
        public bool EstaEnUsoTipoMovimientoByNombre(string nombre);
        public int CtdMovimientos();
        public IEnumerable<MovimientoStock> GetAllByIDArticulo_y_TipoMovimiento(int idArticulo, string tipoMovimiento);
```

```
        public IEnumerable<Articulo> GetByRangoFechas(DateTime fecha1, DateTime fecha2);

        public IEnumerable<object> ObtenerResumenMovimientosPorAnioYTipoMovimiento();

    }

}
```

****

Archivo: IRepositorioTipoMovimiento.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\InterfacesRepositorio\IRepositorioTipoMovimiento.cs

****

```
using Papeleria.LogicaNegocio.Entidades;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace Papeleria.LogicaNegocio.InterfacesRepositorio

{

    public interface IRepositorioTipoMovimiento:IRepositorio<TipoMovimiento>

    {

        public bool ExisteTipoMovimientoXNombre(string nombre);

        public TipoMovimiento GetTipoMovimientoXNombre(string nombre);
```

```
    }
}
```

****

Archivo: IRepositorioUsuario.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\InterfacesRepositorio\IRepositorioUsuario.cs

****

```
using Empresa.LogicaDeNegocio.Entidades;
using Empresa.LogicaDeNegocio.Sistema;
using Papeleria.LogicaNegocio.Entidades;
using Papeleria.LogicaNegocio.Entidades.ValueObjects.Usuario;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Papeleria.LogicaNegocio.InterfacesRepositorio
{
    public interface IRepositorioUsuario:IRepositorio<Usuario>
    {
        public Usuario GetUsuarioPorEmail(string email);
```

```
        public void ModificarContrasenia(int id, ContraseniaUsuario contraseniaNueva);

        public Usuario Login(string email, string contrasenia);

        bool ExisteUsuarioConEmail(string email);

        public EncargadoDeposito GetEncargadoByID(int id);

    }

}
```

****

Archivo: ArticulosController.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.WebApi\Controllers\ArticulosController.cs

****

```
using Empresa.LogicaDeNegocio.Entidades;

using Microsoft.AspNetCore.Mvc;

using Papeleria.AccesoDatos.EF;

using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Articulos;

using Papeleria.LogicaAplicacion.ImplementacionCasosUso.Articulos;

using Papeleria.LogicaAplicacion.InterfacesCasosUso.Articulos;

using Papeleria.LogicaNegocio.Excepciones.Articulo;

using Papeleria.LogicaNegocio.InterfacesRepositorio;


// For more information on enabling Web API for empty projects, visit https://go.microsoft.com/fwlink/?LinkID=397860
```

```
namespace Papeleria.WebApi.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class ArticulosController : ControllerBase
    {
        private PapeleriaContext _context;
        private IRepositorioArticulo _repoArticulos;
        private IGetAllArticulos _cuGetArticulos;
        private IGetArticulo _cuGetArticulo;
        private IAltaArticulo _cuAltaArticulo;
        private IBorrarArticulo _cuBorrarArticulo;
        private IUpdateArticulo _cuModificarArticulo;

        public ArticulosController(PapeleriaContext context, IRepositorioArticulo _repo, IAltaArticulo cuAltaArticulo, IGetAllArticulos cuGetArticulos,
            IGetArticulo cuGetArticulo, IBorrarArticulo cuBorrarArticulo, IUpdateArticulo cuModificarArticulo)
        {
            _context = context;
            _cuGetArticulos = cuGetArticulos;
            _cuGetArticulo = cuGetArticulo;
            _cuAltaArticulo = cuAltaArticulo;
```

```
    _cuBorrarArticulo = cuBorrarArticulo;
    _cuModificarArticulo = cuModificarArticulo;
    _repoArticulos = _repo;
}
// GET: api/<ArticulosController>
/// <summary>
/// Listar todos los articulos
/// </summary>
/// <returns>Lista de articulos ordenados alfabeticamente.</returns>
[HttpGet]
public ActionResult<IEnumerable<ArticuloDTO>> Get()
{
    try
    {
        var articulosDto = _cuGetArticulos.Ejecutar();
        var ordenada = articulosDto.OrderBy(articulo => articulo.NombreArticulo);
        return Ok(ordenada);
    }
    catch (ArticuloNoValidoException ex)
    {
        return BadRequest(ex.Message);
    }
    catch (Exception ex)
```

```
        {
            return StatusCode(500, ex.Message);
        }


    }


    // GET api/<ArticulosController>/5
    /// <summary>
    /// Listar articulo particuloar
    /// </summary>
    /// <param name="id">Número entero con el valor Id del articulo a buscar</param>
    /// <returns>Articulo correspondiente al ID - Code 200 | Error 400 (Bad Request) si parametro/articulo es invalido |  500 - Error con la DB / Excepcion particular</returns>
    [HttpGet("{id}", Name = "GetArticuloByID")]
    public ActionResult<ArticuloDTO> Get(int id)
    {
        try
        {
            var articuloDto = _cuGetArticulo.GetById(id);
            return Ok(articuloDto);
        }
        catch (ArticuloNoValidoException ex)
        {
            return BadRequest(ex.Message);
```

```
        }
        catch (Exception ex)
        {
            return StatusCode(500, ex.Message);
        }
    }
    // POST api/<ArticulosController>
    /// <summary>
    /// Agregar articulo
    /// </summary>
    /// <param name="articulo">Parametro que toma el articulo armado con sus respectivos atributos y lo pasa a la aplicacion para registrarlo</param>
    /// <returns>201 - Si el Articulo fue creado satisfactoriamente | 400 - Si el Articulo suministrado no es valido | 500 - Error con la DB / Excepcion particular</returns>
    [HttpPost]
    public ActionResult<ArticuloDTO> Post(ArticuloDTO articulo)
    {
        try
        {
            _cuAltaArticulo.Ejecutar(articulo);
            return CreatedAtRoute("GetArticuloByID", new { id = articulo.Id }, articulo);
        }

        catch (ArticuloNoValidoException ex)
```

```
        {
            return BadRequest(ex.Message);
        }
        catch (Exception ex)
        {
            return StatusCode(500, ex.Message);
        }
    }

    // PUT api/<ArticulosController>/5
    /// <summary>
    /// Modificar articulo
    /// </summary>
    /// <param name="id">Proporciona el ID del objeto a modificar</param>
    /// <param name="articulo">Proporciona el cuerpo del articulo que va a reemplazar al existente</param>
    /// <returns>200 - Articulo modificado correctamente | 400 - ID/Articulo nuevo invalido | 500 - Error en la DB / Excepcion particular</returns>
    [HttpPut("{id}")]
    public ActionResult<ArticuloDTO> Put(int id, ArticuloDTO articulo)
    {
        try
        {
            _cuModificarArticulo.Ejecutar(id, articulo);
```

```
            return Ok(articulo);
        }
        catch (ArticuloNoValidoException ex)
        {
            return BadRequest(ex.Message);
        }
        catch (Exception ex)
        {
            return StatusCode(500, ex.Message);
        }
    }

    // DELETE api/<ArticulosController>/5
    /// <summary>
    /// Borrar articulo
    /// </summary>
    /// <param name="id">Proporciona el ID del articulo a borrar</param>
    /// <returns>200 - Articulo borrado correctamente | 400 - ID Invalido o Articulo no valido | 500 - Error de la DB / Excepcion particular</returns>
    [HttpDelete("{id}")]
    public ActionResult<ArticuloDTO> Delete(int id)
    {
        try
        {
```

```
            _cuBorrarArticulo.Ejecutar(id);
            return Ok();
        }
        catch (ArticuloNoValidoException ex)
        {
            return BadRequest(ex.Message);
        }
        catch (Exception ex)
        {
            return StatusCode(500, ex.Message);
        }
    }
  }
}
```

****

Archivo: MovimientosController.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.WebApi\Controllers\MovimientosController.cs

****

```
using Empresa.LogicaDeNegocio.Entidades;
using Microsoft.AspNetCore.Authorization;
using Microsoft.AspNetCore.Mvc;
```

```
using Papeleria.AccesoDatos.EF;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Articulos;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.MovimientoStock;
using Papeleria.LogicaAplicacion.ImplementacionCasosUso.Movimiento;
using Papeleria.LogicaAplicacion.InterfacesCasosUso.Articulos;
using Papeleria.LogicaAplicacion.InterfacesCasosUso.Movimientos;
using Papeleria.LogicaAplicacion.InterfacesCasosUso.MovimientoStock;
using Papeleria.LogicaNegocio.Excepciones.Articulo;
using Papeleria.LogicaNegocio.InterfacesRepositorio;

// For more information on enabling Web API for empty projects, visit https://go.microsoft.com/fwlink/?LinkID=397860

namespace Papeleria.WebApi.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class MovimientosController : ControllerBase
    {
        private PapeleriaContext _context;
        private IRepositorioArticulo _repoArt;
        private IRepositorioUsuario _repoUsr;
        private IRepositorioMovimientoStock _repo;
        private IGetAllArticulos _cuGetArticulos;
```

```
private IGetArticulo _cuGetArticulo;

private IAltaArticulo _cuAltaArticulo;

private IBorrarArticulo _cuBorrarArticulo;

private IUpdateArticulo _cuModificarArticulo;

private IGetMovimiento _cuGetMovimiento;

private IAltaMovimiento _cuAltaMovimiento;


public MovimientosController(IRepositorioArticulo repoArt,IRepositorioUsuario repoUsr, IRepositorioMovimientoStock repo, IAltaArticulo cuAltaArticulo, IGetAllArticulos cuGetArticulos,

    IGetArticulo cuGetArticulo, IBorrarArticulo cuBorrarArticulo, IUpdateArticulo cuModificarArticulo, IAltaMovimiento altaMovimiento, IGetMovimiento getMovimiento)

{

    _repoArt = repoArt;

    _repoUsr = repoUsr;

    _repo = repo;

    _cuGetArticulos = cuGetArticulos;

    _cuGetArticulo = cuGetArticulo;

    _cuAltaArticulo = cuAltaArticulo;

    _cuBorrarArticulo = cuBorrarArticulo;

    _cuModificarArticulo = cuModificarArticulo;

    _cuAltaMovimiento = altaMovimiento;

    _cuGetMovimiento = getMovimiento;

}

// GET: api/<Movimientos>
```

```
[Authorize]
[HttpGet]
public ActionResult<IEnumerable<MovimientoStockDTO>> Get()
{
    try
    {
        var movimientosDto = _cuGetMovimiento.GetAll();
        var ordenada = movimientosDto.OrderByDescending(movimientos => movimientos.ID);
        return Ok(ordenada);
    }
    catch (ArticuloNoValidoException ex)
    {
        return BadRequest(ex.Message);
    }
    catch (Exception ex)
    {
        return StatusCode(500, ex.Message);
    }
}

// GET api/<Movimientos>/5
[Authorize]
[HttpGet("{id}", Name = "GetMov ByID")]
```

```
public ActionResult<MovimientoStockDTO> Get(int id)
{
    try
    {
        var movimientosDto = _cuGetMovimiento.GetByDTO(id);
        return Ok(movimientosDto);
    }
    catch (ArticuloNoValidoException ex)
    {
        return BadRequest(ex.Message);
    }
    catch (Exception ex)
    {
        return StatusCode(500, ex.Message);
    }
}

// POST api/<Movimientos>
[Authorize]
[HttpPost]
public ActionResult<MovimientoStockDTO> Post([FromBody] MovimientoStockDTO mov)
{
    try
```

```
    {
        _cuAltaMovimiento.Crear(mov);
        //return CreatedAtRoute("GetMovByID", new { id = mov.ID }, mov);
        return Ok(mov);
    }

    catch (ArticuloNoValidoException ex)
    {
        return BadRequest(ex.Message);
    }
    catch (Exception ex)
    {
        return StatusCode(500, ex.Message);
    }

}

/* // PUT api/<Movimientos>/5
 [Authorize]
 [HttpPut("{id}")]
 public ActionResult<MovimientoStockDTO> Put(int id, [FromBody] MovimientoStockDTO mov)
 {
    try
```

```
    {
        _cuUpdateMovimiento.Update(id, mov);
        return Ok(mov);
    }
    catch (ArticuloNoValidoException ex)
    {
        return BadRequest(ex.Message);
    }
    catch (Exception ex)
    {
        return StatusCode(500, ex.Message);
    }
}

// DELETE api/<Movimientos>/5
[Authorize]
[HttpDelete("{id}")]
public ActionResult<MovimientoStockDTO> Delete(int id)
{
    try
    {
        _cuBorrarMovimiento.Remove(id);
        return Ok();
```

```
    }
    catch (ArticuloNoValidoException ex)
    {
        return BadRequest(ex.Message);
    }
    catch (Exception ex)
    {
        return StatusCode(500, ex.Message);
    }
}*/

//TODO Verificar autorizacion en API
// GET api/<Movimientos>/5/entradas
[Authorize]
[HttpGet("{id}/{mov}")]
public ActionResult<MovimientoStockDTO> GetByIdYTipoMov(int id, string mov)
{
    if (id == null)
        return BadRequest("Debe indicar el ID del Aritculo a buscar en MovimientoStock.");
    if (mov == null)
        return BadRequest("Debe indicar el Tipo Movimiento a buscar en MovimientoStock.");
    try
    {
```

```
            var movimientosDto = _cuGetMovimiento.GetMovimientosByIDArticuloYTipoMov(id, mov);
            if (movimientosDto.Count() == 0)
                return NotFound("No se encontro movimiento con esa ID de Articulo y Tipo Movimiento.");
            return Ok(movimientosDto);
        }
        catch (ArticuloNoValidoException ex)
        {
            return BadRequest(ex.Message);
        }
        catch (Exception ex)
        {
            return StatusCode(500, ex.Message);
        }
    }

    // GET api/<Movimientos>/articulos-por-fechas
    [Authorize]
    [HttpGet("articulos-por-fechas")]
    public ActionResult<ArticuloDTO> GetArticuloByRangoFechas(DateTime fechaIni, DateTime fechaFin)
    {
        if (fechaIni == null)
            return BadRequest("Debe indicar la fecha de inicio para buscar en MovimientoStock.");
        if (fechaFin == null)
```

```
            return BadRequest("Debe indicar la fecha de fin para buscar en MovimientoStock.");
        if (fechaIni > fechaFin)
            return BadRequest("La fecha de inicio no puede ser mayor que la fecha de fin.");
        try
        {
            var articulosDTO = _cuGetMovimiento.GetArticulosByRangoFecha(fechaIni,fechaFin);
            if (articulosDTO.Count() == 0)
                return NotFound("No se encontraron articulos en ese periodo de fechas.");
            return Ok(articulosDTO);
        }
        catch (ArticuloNoValidoException ex)
        {
            return BadRequest(ex.Message);
        }
        catch (Exception ex)
        {
            return StatusCode(500, ex.Message);
        }
    }

    // GET api/<Movimientos>/resumen
    //[Authorize]
    [HttpGet("resumen")]
```

```
        public ActionResult<object> ObtenerResumenMovimientosPorAnioYTipoMovimiento()
        {
            try
            {
                var resumen = _cuGetMovimiento.ObtenerResumenMovimientosPorAnioYTipoMovimiento();
                if (resumen.Count() == 0)
                    return NotFound("No se articulo en ese periodo de fechas.");
                return Ok(resumen);
            }
            catch (ArticuloNoValidoException ex)
            {
                return BadRequest(ex.Message);
            }
            catch (Exception ex)
            {
                return StatusCode(500, ex.Message);
            }
        }
    }
}
```

****

Archivo: TipoMovimientosController.cs

Carpeta:
C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.WebApi\Controllers\TipoMovimientosController.cs

****

```
using Microsoft.AspNetCore.Authorization;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Papeleria.AccesoDatos.EF;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Articulos;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.TipoMovimientos;
using Papeleria.LogicaAplicacion.ImplementacionCasosUso.Articulos;
using Papeleria.LogicaAplicacion.ImplementacionCasosUso.TipoMovimientos;
using Papeleria.LogicaAplicacion.InterfacesCasosUso.Articulos;
using Papeleria.LogicaAplicacion.InterfacesCasosUso.Movimientos;
using Papeleria.LogicaAplicacion.InterfacesCasosUso.MovimientoStock;
using Papeleria.LogicaAplicacion.InterfacesCasosUso.TipoMovimientos;
using Papeleria.LogicaNegocio.Excepciones.Articulo;
using Papeleria.LogicaNegocio.Excepciones.MovimientoStock;
using Papeleria.LogicaNegocio.Excepciones.TipoMovimiento;
using Papeleria.LogicaNegocio.InterfacesRepositorio;

namespace Papeleria.WebApi.Controllers
{
    [Route("api/[controller]")]
```

```
[ApiController]
public class TipoMovimientosController : ControllerBase
{
    private IRepositorioTipoMovimiento _repoTipoMov;
    private IRepositorioMovimientoStock _repoStock;
    private IAltaTiposMovimientos _cuAltaTipoMov;
    private IBorrarTipoMovimiento _cuBorrarTipoMov;
    private IGetAllTipoMovimiento _cuGetAllTipoMov;
    private IGetTipoMovimiento _cuGetTipoMovimiento;
    private IUpdateTipoMovimiento _cuUpdateTipoMovimiento;
    private IFiltrarMovimiento _cuBuscarMovimiento;

    public TipoMovimientosController(IRepositorioTipoMovimiento repoTipoMov, IAltaTiposMovimientos cuAltaTipoMov, IBorrarTipoMovimiento cuBorrarTipoMov,
        IGetAllTipoMovimiento cuGetAllTipoMov, IGetTipoMovimiento cuGetTipoMovimiento, IUpdateTipoMovimiento cuUpdateTipoMovimiento, IFiltrarMovimiento cuBuscarMovimiento)
    {
        _repoTipoMov = repoTipoMov;
        _cuAltaTipoMov = cuAltaTipoMov;
        _cuBorrarTipoMov = cuBorrarTipoMov;
        _cuGetAllTipoMov = cuGetAllTipoMov;
        _cuGetTipoMovimiento = cuGetTipoMovimiento;
        _cuUpdateTipoMovimiento = cuUpdateTipoMovimiento;
        _cuBuscarMovimiento = cuBuscarMovimiento;
```

```
}


// GET: api/<TipoMovimientosController>
/// <summary>
/// Listar todos los Tipos Movimientos
/// </summary>
/// <returns>Tipos Movimientos ordenados por ID.</returns>
[HttpGet]
[AllowAnonymous]
public ActionResult<IEnumerable<TipoMovimientoDTO>> Get()
{
    try
    {
        var tipMovDTO = _cuGetAllTipoMov.Ejecutar();
        var ordenada = tipMovDTO.OrderBy(mov => mov.ID);
        return Ok(ordenada);
    }
    catch (TipoMovimientoNoValidoException ex)
    {
        return BadRequest(ex.Message);
    }
    catch (Exception ex)
    {
```

```
            return StatusCode(500, ex.Message);
        }


    }

    // GET api/<TipoMovimientosController>/5
    /// <summary>
    /// Listar TipoMovimiento particular por ID
    /// </summary>
    /// <param name="ID">Número entero con el valor ID del tipo movimiento a buscar.</param>
    /// <returns>Tipo movimiento correspondiente al ID - Code 200 | Error 400 (Bad Request) si parametro/articulo es invalido |  500 - Error con la DB / Excepcion particular</returns>
    [HttpGet("{id}", Name = "GetTipoMovimientoByID")]
    public ActionResult<TipoMovimientoDTO> Get(int id)
    {
        try
        {
            var tipMovDTO = _cuGetTipoMovimiento.GetById(id);
            return Ok(tipMovDTO);
        }
        catch (TipoMovimientoNoValidoException ex)
        {
            return BadRequest(ex.Message);
        }
```

```
        catch (Exception ex)
        {
            return StatusCode(500, ex.Message);
        }
    }

    // GET api/<TipoMovimientosController>/nombre
    /// <summary>
    /// Listar TipoMovimiento particular por nombre.
    /// </summary>
    /// <param name="nombre">Nombre del tipo movimiento a buscar.</param>
    /// <returns>Tipo movimiento correspondiente al ID - Code 200 | Error 400 (Bad Request) si parametro/articulo es invalido |  500 - Error con la DB / Excepcion particular</returns>
    /*
    [HttpGet("{Nombre}", Name = "GetByNombre")]
    [AllowAnonymous]
    public ActionResult<TipoMovimientoDTO> GetByNombre(string nombre)
    {
        try
        {
            var tipMovDTO = _cuGetTipoMovimiento.GetByNombre(nombre);
            return Ok(tipMovDTO);
        }
        catch (TipoMovimientoNoValidoException ex)
```

```
        {
            return BadRequest(ex.Message);
        }
        catch (Exception ex)
        {
            return StatusCode(500, ex.Message);
        }
    }
    */
    // POST api/<TipoMovimientosController>
    /// <summary>
    /// Agregar Tipo Movimiento
    /// </summary>
    /// <param name="tipMov">Parametro que toma el "TipoMovimiento" armado con sus respectivos atributos y lo pasa a la aplicacion para registrarlo</param>
    /// <returns>201 - Si el Articulo fue creado satisfactoriamente | 400 - Si el Articulo suministrado no es valido | 500 - Error con la DB / Excepcion particular</returns>
    [HttpPost]
    [AllowAnonymous]
    public ActionResult<TipoMovimientoDTO> Post(TipoMovimientoDTO tipMov)
    {
        try
        {
            _cuAltaTipoMov.Ejecutar(tipMov);
```

```
            return CreatedAtRoute("GetTipoMovimientoByID", new { id = tipMov.ID }, tipMov);
        }

        catch (TipoMovimientoNoValidoException ex)
        {
            return BadRequest(ex.Message);
        }
        catch (Exception ex)
        {
            return StatusCode(500, ex.Message);
        }
    }

    // PUT api/<TipoMovimientosController>/5
    /// <summary>
    /// Modificar TipoMovimiento
    /// </summary>
    /// <param name="id">Proporciona el ID del objeto a modificar</param>
    /// <param name="tipMov">Proporciona el cuerpo del articulo que va a reemplazar al existente</param>
    /// <returns>200 - Articulo modificado correctamente | 400 - ID/Articulo nuevo invalido | 500 - Error en la DB / Excepcion particular</returns>
    [HttpPut("{id}")]
    [AllowAnonymous]
```

```
public ActionResult<TipoMovimientoDTO> Put(int id, TipoMovimientoDTO tipMov)
{
    try
    {
        if (_cuBuscarMovimiento.ExisteTipoMovimientoEnMovimientoByID(id))
            return BadRequest("Existe un MovimientoStock que esta utilizando este TipoMovimiento.");
        else
            _cuUpdateTipoMovimiento.Ejecutar(id, tipMov);
            return Ok(tipMov);
    }
    catch (TipoMovimientoNoValidoException ex)
    {
        return BadRequest(ex.Message);
    }
    catch (Exception ex)
    {
        return StatusCode(500, ex.Message);
    }
}

// DELETE api/<TipoMovimientosController>/5
/// <summary>
/// Borrar Tipo Movimiento.
```

```
        /// </summary>
        /// <param name="id">Proporciona el ID del "Tipo Movimiento" a borrar</param>
        /// <returns>200 - Articulo borrado correctamente | 400 - ID Invalido o Articulo no valido | 500 - Error de la DB / Excepcion particular</returns>
        [HttpDelete("{id}")]
        [AllowAnonymous]
        public ActionResult<TipoMovimientoDTO> Delete(int id)
        {
            try
            {
                if (_cuBuscarMovimiento.ExisteTipoMovimientoEnMovimientoByID(id))
                    return BadRequest("Existe un MovimientoStock que esta utilizando este TipoMovimiento.");
                else
                    _cuBorrarTipoMov.Ejecutar(id);
                    return Ok();
            }
            catch (TipoMovimientoNoValidoException ex)
            {
                return BadRequest(ex.Message);
            }
            catch (Exception ex)
            {
                return StatusCode(500, ex.Message);
            }
```

```
        }
    }
}
```

****

Archivo: UsuariosController.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.WebApi\Controllers\UsuariosController.cs

****

```
using Empresa.LogicaDeNegocio.Sistema;
using Microsoft.AspNetCore.Authorization;
using Microsoft.AspNetCore.Mvc;
using Papeleria.AccesoDatos.EF;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Articulos;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Usuarios;
using Papeleria.LogicaAplicacion.DataTransferObjects.MapeosDatos;
using Papeleria.LogicaAplicacion.ImplementacionCasosUso.Usuarios;
using Papeleria.LogicaAplicacion.InterfacesCasosUso.Articulos;
using Papeleria.LogicaAplicacion.InterfacesCasosUso.Usuarios;
using Papeleria.LogicaNegocio.Entidades;
using Papeleria.LogicaNegocio.Excepciones.Articulo;
using Papeleria.LogicaNegocio.InterfacesRepositorio;
using SistemaDocentes.Api.UtilidadesJwt;
```

```
using System.Net.Http.Headers;

// For more information on enabling Web API for empty projects, visit https://go.microsoft.com/fwlink/?LinkID=397860

namespace Papeleria.WebApi.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class UsuariosController : ControllerBase
    {
        private ILogin _login;
        private PapeleriaContext _context;
        private IRepositorioUsuario _repo;
        private IAltaUsuario _altaUsuario;
        private IBorrarUsuario _borrarUsuario;
        private IGetAllUsuarios _getAllUsuarios;
        private IGetUsuario _getUsuario;
        private IModificarUsuario _modificarUsuario;
        public UsuariosController(ILogin login, PapeleriaContext context, IRepositorioUsuario repo, IAltaUsuario altaUsuario, IBorrarUsuario borrarUsuario, IGetAllUsuarios getAllUsuarios, IGetUsuario getUsuario, IModificarUsuario modificarUsuario)
        {
            _login = login;
```

```
    _context = context;
    _repo = repo;
    _altaUsuario = altaUsuario;
    _borrarUsuario = borrarUsuario;
    _getAllUsuarios = getAllUsuarios;
    _getUsuario = getUsuario;
    _modificarUsuario = modificarUsuario;
}



// GET: api/<UsuarioController>
[Authorize]
[HttpGet]
public ActionResult<IEnumerable<UsuarioDTO>> Get()
{
    try
    {
        var usuariosDto = _getAllUsuarios.Ejecutar();
        var ordenada = usuariosDto.OrderBy(usuario => usuario.Nombre);
        return Ok(ordenada);
    }
    catch (ArticuloNoValidoException ex)
    {
```

```
        return BadRequest(ex.Message);
    }
    catch (Exception ex)
    {
        return StatusCode(500, ex.Message);
    }
}
[HttpPost]
public ActionResult<UsuarioDTO> Post(UsuarioDTO usr)
{
    try
    {
        _altaUsuario.Ejecutar(usr);
        return CreatedAtRoute("GetUsrByID", new { id = usr.Id }, usr);
    }

    catch (ArticuloNoValidoException ex)
    {
        return BadRequest(ex.Message);
    }
    catch (Exception ex)
    {
        return StatusCode(500, ex.Message);
```

```
    }
}
[HttpGet("{id}", Name = "GetUsrByID")]
public ActionResult<UsuarioDTO> Get(int id)
{
    try
    {
        var usr = _getUsuario.GetById(id);
        return Ok(usr);
    }
    catch (ArticuloNoValidoException ex)
    {
        return BadRequest(ex.Message);
    }
    catch (Exception ex)
    {
        return StatusCode(500, ex.Message);
    }
}

// POST: api/<UsuarioController>/login
[AllowAnonymous]
[HttpPost("login")]
```

```
//xxxxx
public IActionResult Login(UsuarioDTO usr)
{
    try
    {
        var usuario = _login.Ejecutar(usr.Email, usr.Contrasenia);
        //var usuario = _repo.Login(usr.Email, usr.Contrasenia);
        var rol = usuario.GetType().Name;
        var dto = UsuariosMappers.ToDto(usuario);

        if (string.IsNullOrWhiteSpace(rol))
        {
            return Unauthorized("Credenciales incorrectas");
        }
        string token = ManejadorJwt.GenerarToken(usr.Email, rol);
        return Ok(new { Token = token, Rol = rol, Email = dto.Email, userId = dto.Id });
    }
    catch (Exception ex)
    {
        return Unauthorized(new { Error = "Credenciales incorrectas" });
    }

}
```

```
    }
}
```

****

Archivo: ManejadorJwt.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.WebApi\UtilidadesJWT\ManejadorJwt.cs

****

```
using Microsoft.Extensions.Configuration;
using Microsoft.IdentityModel.Tokens;
using System.IdentityModel.Tokens.Jwt;
using System.Security.Claims;
using System.Text;
using static Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Usuarios.UsuarioDTO;

namespace SistemaDocentes.Api.UtilidadesJwt
{
    public class ManejadorJwt
    {

        /// <summary>
        /// Método para generar el token JWT usando una función estática (no es necesario tener instancias)
        /// </summary>
```

```
/// <remarks> Creación del "payload" con tiene la información del usuario que se logueó (subject)
/// El usuario tiene "claims", que son pares nombre/valor que se utilizan para guardar
/// en el cliente. No pueden ser sensibles
/// Se le debe setear el periodo temporal de validez (Expires)
///Se utiliza un algoritmo de clave simétrica para generar el token</remarks>

public static string GenerarToken(string email, string rol)
{
    var claveDificil = "ClaveMuySecreta1_ClaveMuySecreta1_ClaveMuySecreta1_ClaveMuySecreta1_ClaveMuySecreta1_ClaveMuySecreta1";
    var claveDificilEncriptada = new SymmetricSecurityKey(Encoding.UTF8.GetBytes(claveDificil));
    List<Claim> claims = [
        new Claim(ClaimTypes.Email, email),
        new Claim(ClaimTypes.Role, rol)
        ];

    var credenciales = new SigningCredentials(claveDificilEncriptada, SecurityAlgorithms.HmacSha512Signature);

    var token = new JwtSecurityToken(claims: claims, expires: DateTime.Now.AddHours(1), signingCredentials: credenciales);

    var jwt = new JwtSecurityTokenHandler().WriteToken(token);
```

```
            return jwt;


        }




    }
}
```

****

Archivo: ArticulosMappers.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\DataTransferObjects\MapeosDatos\ArticulosMappers.cs

****

```
using Empresa.LogicaDeNegocio.Sistema;
using Empresa.LogicaDeNegocio.Entidades;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Usuarios;
using Papeleria.LogicaNegocio.Excepciones.Usuario;
using System;
using System.Collections.Generic;
using System.Linq;
```

```
using System.Text;
using System.Threading.Tasks;
using Papeleria.LogicaNegocio.Excepciones.Articulo;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Articulos;

namespace Papeleria.LogicaAplicacion.DataTransferObjects.MapeosDatos
{
    public class ArticulosMappers
    {
        public static Articulo FromDto(ArticuloDTO dto)
        {
            if (dto == null) throw new ArticuloNuloException(nameof(dto));
            return new Articulo(dto.CodigoProveedor, dto.NombreArticulo, dto.Descripcion, dto.PrecioVP);
        }
        public static Articulo FromDtoUpdate(ArticuloDTO dto)
        {
            if (dto == null) throw new ArticuloNuloException(nameof(dto));
            var articulo = new Articulo(dto.CodigoProveedor, dto.NombreArticulo, dto.Descripcion, dto.PrecioVP);
            articulo.ID = dto.Id;
            return articulo;
        }
        public static ArticuloDTO ToDto(Articulo articulo)
        {
```

```
        if (articulo == null) throw new ArticuloNuloException();
        return new ArticuloDTO()
        {
            Id = articulo.ID,
            CodigoProveedor = articulo.CodigoProveedor.codigo,
            NombreArticulo = articulo.NombreArticulo.Nombre,
            Descripcion = articulo.Descripcion.Descripcion,
            PrecioVP = articulo.PrecioVP,
        };
    }

    public static IEnumerable<ArticuloDTO> FromLista(IEnumerable<Articulo> articulos)
    {
        if (articulos == null)
        {
            throw new ArticuloNuloException("La lista de articulos no puede ser nula");
        }
        return articulos.Select(articulo => ToDto(articulo));
    }
  }
}
```

****

Archivo: MovimientoStockMappers.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\DataTransferObjects\MapeosDatos\MovimientoStockMappers.cs

****

```
using Empresa.LogicaDeNegocio.Entidades;

using Empresa.LogicaDeNegocio.Sistema;

using Papeleria.AccesoDatos.EF;

using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Articulos;

using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.MovimientoStock;

using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.TipoMovimientos;

using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Usuarios;

using Papeleria.LogicaAplicacion.ImplementacionCasosUso.Articulos;

using Papeleria.LogicaAplicacion.ImplementacionCasosUso.TipoMovimientos;

using Papeleria.LogicaAplicacion.ImplementacionCasosUso.Usuarios;

using Papeleria.LogicaAplicacion.InterfacesCasosUso.Articulos;

using Papeleria.LogicaAplicacion.InterfacesCasosUso.TipoMovimientos;

using Papeleria.LogicaAplicacion.InterfacesCasosUso.Usuarios;

using Papeleria.LogicaNegocio.Entidades;

using Papeleria.LogicaNegocio.Excepciones.MovimientoStock;

using Papeleria.LogicaNegocio.InterfacesRepositorio;


namespace Papeleria.LogicaAplicacion.DataTransferObjects.MapeosDatos

{
```

```
public class MovimientoStockMappers
{
    private static PapeleriaContext _context;
    private static IRepositorioArticulo _repoArt;
    private static IRepositorioUsuario _repoUsr;
    private static IRepositorioTipoMovimiento _repoTipMov;
    private static IGetArticulo _getArticulo;
    private static IGetTipoMovimiento _getTipoMovimiento;
    private static IGetUsuario _getUsuario;
    public static MovimientoStock FromDTO(MovimientoStockDTO dto, IRepositorioArticulo _repoArticulos, IRepositorioUsuario _repoUsr, IRepositorioTipoMovimiento _repoTipMov)
    {
        _getArticulo = new BuscarArticulo(_repoArticulos);
        _getTipoMovimiento = new BuscarTipoMovimiento(_repoTipMov);
        _getUsuario = new BuscarUsuario(_repoUsr);
        if (dto == null)
        {
            throw new MovimientoStockNuloException("Movimiento de stock nulo");
        }
        Articulo articulo = _getArticulo.GetById(dto.ArticuloID);
        TipoMovimiento tipoMovimiento = _getTipoMovimiento.GetById(dto.TipoMovimientoID);
        Usuario usuario = _getUsuario.GetEncargadoByID(dto.UsuarioID);
        return new MovimientoStock(articulo, tipoMovimiento, usuario, dto.CtdUnidadesXMovimiento);
    }
```

```
public static MovimientoStock FromDTOUpdate(MovimientoStockDTO dto, IRepositorioArticulo _repoArticulos, IRepositorioUsuario _repoUsr, IRepositorioTipoMovimiento _repoTipMov)
{
    _getArticulo = new BuscarArticulo(_repoArticulos);
    _getTipoMovimiento = new BuscarTipoMovimiento(_repoTipMov);
    _getUsuario = new BuscarUsuario(_repoUsr);
    if (dto == null)
    {
        throw new MovimientoStockNuloException("Movimiento de stock nulo");
    }
    Articulo articulo = _getArticulo.GetById(dto.ArticuloID);
    Usuario usuario = _getUsuario.GetEncargadoByID(dto.UsuarioID);
    TipoMovimiento tipoMovimiento = _getTipoMovimiento.GetById(dto.TipoMovimientoID);
    MovimientoStock mov = new MovimientoStock(articulo, tipoMovimiento, usuario, dto.CtdUnidadesXMovimiento);
    mov.ID = dto.ID;
    return mov;
}
public static MovimientoStockDTO ToDto(MovimientoStock mov)
{
    if (mov == null) throw new MovimientoStockNuloException("Movimiento de stock nulo"); ;
    ArticuloDTO articulo = ArticulosMappers.ToDto(mov.Articulo);
    UsuarioDTO usuario = UsuariosMappers.ToDto(mov.UsuarioRealizaMovimiento);
    TipoMovimientoDTO tipMov = TipoMovimientoMappers.ToDto(mov.Movimiento);
```

```
            return new MovimientoStockDTO()
            {
                ID = mov.ID,
                FecHorMovRealizado = mov.FecHorMovRealizado,
                ArticuloID = articulo.Id,
                CodigoProveedor = articulo.CodigoProveedor,
                NombreArticulo = articulo.NombreArticulo,
                Descripcion = articulo.Descripcion,
                PrecioVP = articulo.PrecioVP,
                TipoMovimientoID = tipMov.ID,
                TipoMovimientoNombre = tipMov.Nombre,
                UsuarioID = usuario.Id,
                Email = usuario.Email,
                Nombre = usuario.Nombre,
                Apellido = usuario.Apellido,
                Contrasenia = usuario.Contrasenia,
                CtdUnidadesXMovimiento = mov.CtdUnidadesXMovimiento
            };
        }

        public static IEnumerable<MovimientoStockDTO> FromLista(IEnumerable<MovimientoStock> movimientos)
        {
            if (movimientos == null)
```

```
            {
                throw new MovimientoStockNuloException("La lista de articulos no puede ser nula");
            }
            return movimientos.Select(mov => ToDto(mov));
        }
    }
}
```

****

Archivo: TipoMovimientoMappers.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\DataTransferObjects\MapeosDatos\TipoMovimientoMappers.cs

****

```
using Empresa.LogicaDeNegocio.Entidades;
using Empresa.LogicaDeNegocio.Sistema;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Articulos;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.TipoMovimientos;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Usuarios;
using Papeleria.LogicaNegocio.Entidades;
using Papeleria.LogicaNegocio.Excepciones.Articulo;
using Papeleria.LogicaNegocio.Excepciones.TipoMovimiento;
using Papeleria.LogicaNegocio.Excepciones.Usuario;
using System;
```

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

namespace Papeleria.LogicaAplicacion.DataTransferObjects.MapeosDatos
{
    public class TipoMovimientoMappers
    {
        public static TipoMovimiento FromDto(TipoMovimientoDTO dto)
        {
            if (dto == null) throw new TipoMovimientoNuloException(nameof(dto));
            return new TipoMovimiento(dto.Nombre);
        }
        public static TipoMovimiento FromDtoUpdate(TipoMovimientoDTO dto)
        {
            if (dto == null) throw new TipoMovimientoNuloException(nameof(dto));
            var tipoMovimiento = new TipoMovimiento(dto.Nombre);
            tipoMovimiento.ID = dto.ID;
            return tipoMovimiento;
        }
        public static TipoMovimientoDTO ToDto(TipoMovimiento tipMov)
        {
```

```
        if (tipMov == null) throw new TipoMovimientoNuloException();
        return new TipoMovimientoDTO()
        {
            ID = tipMov.ID,
            Nombre = tipMov.Nombre
        };
    }

    public static IEnumerable<TipoMovimientoDTO> FromLista(IEnumerable<TipoMovimiento> tipoMovimientos)
    {
        if (tipoMovimientos == null)
        {
            throw new TipoMovimientoNuloException("La lista de usuarios no puede ser nula");
        }
        return (IEnumerable<TipoMovimientoDTO>)tipoMovimientos.Select(tipoMovimiento => ToDto(tipoMovimiento));
    }
  }
}
```

****

Archivo: UsuariosMappers.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\DataTransferObjects\MapeosDatos\UsuariosMappers.cs

****

```
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Articulos;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Papeleria.LogicaNegocio.Entidades;
using Empresa.LogicaDeNegocio.Entidades;
using Empresa.LogicaDeNegocio.Sistema;
using Papeleria.LogicaNegocio.Excepciones.Usuario;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Usuarios;

namespace Papeleria.LogicaAplicacion.DataTransferObjects.MapeosDatos
{
    public class UsuariosMappers
    {
        public static Usuario FromDto(UsuarioDTO dto)
        {
            if (dto == null) throw new UsuarioNuloExcepcion(nameof(dto));
            if (dto.Admin == 1)
```

```
    {
        return new Administrador(dto.Email, dto.Nombre, dto.Apellido, dto.Contrasenia);


    }
    else
    {
        return new EncargadoDeposito(dto.Email, dto.Nombre, dto.Apellido, dto.Contrasenia);
    }
}
public static Usuario FromDtoUpdate(UsuarioDTO dto)
{
    if (dto == null) throw new UsuarioNuloExcepcion(nameof(dto));
    if (dto.Admin == 1)
    {
        var usuario = new Administrador(dto.Email, dto.Nombre, dto.Apellido, dto.Contrasenia);
        usuario.ID = dto.Id;
        return usuario;
    }
    else
    {
        var usuario = new EncargadoDeposito(dto.Email, dto.Nombre, dto.Apellido, dto.Contrasenia);
        usuario.ID = dto.Id;
        return usuario;
```

```
    }
}
public static UsuarioDTO ToDto(Usuario usuario)
{
    if (usuario == null) throw new UsuarioNuloExcepcion();
    if(usuario.GetType() == typeof(Administrador))
    {
        return new UsuarioDTO()
        {
            Id = usuario.ID,
            Admin = 1,
            Email = usuario.Email.Direccion,
            Nombre = usuario.NombreCompleto.Nombre,
            Apellido = usuario.NombreCompleto.Apellido,
            Contrasenia = usuario.Contrasenia.Valor
        };
    }
    else
    {
        return new UsuarioDTO()
        {
            Id = usuario.ID,
            Admin = 0,
```

```
                Email = usuario.Email.Direccion,
                Nombre = usuario.NombreCompleto.Nombre,
                Apellido = usuario.NombreCompleto.Apellido,
                Contrasenia = usuario.Contrasenia.Valor
            };
        }
    }

    public static IEnumerable<UsuarioDTO> FromLista(IEnumerable<Usuario> usuarios)
    {
        if (usuarios == null)
        {
            throw new UsuarioNuloExcepcion("La lista de usuarios no puede ser nula");
        }
        return usuarios.Select(usuario => ToDto(usuario));
    }
    //https://vimeopro.com/universidadortfi/fi-5212-programacion-3-cabella-69235-p3-m3a-remoto/video/929607409
    //1:36:07

  }
}
```

****

Archivo: AltaArticulo.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\ImplementacionCasosUso\Articulos\AltaArticulo.cs

****

```
using Empresa.LogicaDeNegocio.Sistema;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Usuarios;
using Papeleria.LogicaAplicacion.DataTransferObjects.MapeosDatos;
using Papeleria.LogicaNegocio.Excepciones.Articulo;
using Papeleria.LogicaNegocio.Excepciones.Usuario;
using Papeleria.LogicaNegocio.InterfacesRepositorio;
using Empresa.LogicaDeNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Articulos;
using Papeleria.LogicaAplicacion.InterfacesCasosUso.Articulos;
using Papeleria.LogicaNegocio.Excepciones.Usuario.UsuarioExcepcions.Email;

namespace Papeleria.LogicaAplicacion.ImplementacionCasosUso.Articulos
{
    public class AltaArticulo : IAltaArticulo
```

```
{
    private IRepositorioArticulo _repoArticulos;

    public AltaArticulo(IRepositorioArticulo repo)
    {
        _repoArticulos = repo;
    }

    public void Ejecutar(ArticuloDTO dto)
    {
        if (dto == null)
            throw new ArticuloNuloException("Nulo");

        bool nombreExiste = _repoArticulos.ExisteArticuloConNombre(dto.NombreArticulo);
        if (nombreExiste)
        {
            throw new ArticuloNoValidoException("El nombre del articulo ya está en uso.");
        }
        else
        {
            Articulo articulo = ArticulosMappers.FromDto(dto);
            _repoArticulos.Add(articulo);
        }
```

```
        }
    }
}
```

****

Archivo: BorrarArticulo.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\ImplementacionCasosUso\Articulos\BorrarArticulo.cs

****

```
using Empresa.LogicaDeNegocio.Entidades;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Articulos;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Usuarios;
using Papeleria.LogicaAplicacion.DataTransferObjects.MapeosDatos;
using Papeleria.LogicaAplicacion.InterfacesCasosUso.Articulos;
using Papeleria.LogicaNegocio.Excepciones.Articulo;
using Papeleria.LogicaNegocio.InterfacesRepositorio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Papeleria.LogicaAplicacion.ImplementacionCasosUso.Articulos
```

```
{
    public class BorrarArticulo : IBorrarArticulo
    {
        private IRepositorioArticulo _repoArticulos;

        public BorrarArticulo(IRepositorioArticulo repo)
        {
            _repoArticulos = repo;
        }
        public void Ejecutar(int id)
        {
            var articulo = _repoArticulos.GetById(id);
            if(articulo == null)
            {
                throw new ArticuloNuloException("Articulo no puede ser nulo");
            }
            try
            {
                _repoArticulos.Remove(articulo);
            } catch (Exception ex)
            {
                throw new ArticuloNoValidoException(ex.Message);
            }
```

```
        }
        public void Ejecutar(Articulo articulo)
        {
            if (articulo == null)
            {
                throw new ArticuloNuloException("Articulo no puede ser nulo");
            }
            try
            {
                _repoArticulos.Remove(articulo);
            }
            catch (Exception ex)
            {
                throw new ArticuloNoValidoException(ex.Message);
            }
        }
    }
}
```

****

Archivo: BuscarArticulo.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\ImplementacionCasosUso\Articulos\BuscarArticulo.cs

****

```
using Empresa.LogicaDeNegocio.Entidades;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Articulos;
using Papeleria.LogicaAplicacion.DataTransferObjects.MapeosDatos;
using Papeleria.LogicaAplicacion.InterfacesCasosUso.Articulos;
using Papeleria.LogicaNegocio.Entidades.ValueObjects.Articulos;
using Papeleria.LogicaNegocio.Excepciones.Articulo;
using Papeleria.LogicaNegocio.InterfacesRepositorio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Papeleria.LogicaAplicacion.ImplementacionCasosUso.Articulos
{
    public class BuscarArticulo : IGetArticulo
    {

        private IRepositorioArticulo _repoArticulos;

        public BuscarArticulo(IRepositorioArticulo repo)
        {
```

```
    _repoArticulos = repo;
}

public ArticuloDTO GetByIdDTO(int id)
{
    var articulo = _repoArticulos.GetById(id);
    if (articulo == null)
    {
        throw new ArticuloNuloException("Articulo no encontrado con el ID especificado");
    }
    var articuloReturn = ArticulosMappers.ToDto(articulo);
    return articuloReturn;
}
public Articulo GetById(int id)
{
    return _repoArticulos.GetById(id);
}

public ArticuloDTO GetArticuloPorCodigo(CodigoProveedorArticulos codigoProveedor)
{
    var articulo = _repoArticulos.GetArticuloByCodigo(codigoProveedor);
    if(articulo == null)
    {
```

```
            throw new ArticuloNuloException("Articulo no encontrado con el codigo especificado");
        }
        var articuloReturn = ArticulosMappers.ToDto(articulo);
        return articuloReturn;
    }

    public IEnumerable<ArticuloDTO> GetArticulosPorNombre(string nombre)
    {
        throw new NotImplementedException();
    }
  }
}
```

****

Archivo: GetAllArticulos.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\ImplementacionCasosUso\Articulos\GetAllArticulos.cs

****

```
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Articulos;
using Papeleria.LogicaAplicacion.DataTransferObjects.MapeosDatos;
using Papeleria.LogicaAplicacion.InterfacesCasosUso.Articulos;
using Papeleria.LogicaNegocio.Excepciones.Articulo;
using Papeleria.LogicaNegocio.InterfacesRepositorio;
```

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

namespace Papeleria.LogicaAplicacion.ImplementacionCasosUso.Articulos
{
    public class GetAllArticulos : IGetAllArticulos
    {
        private IRepositorioArticulo _repoArticulos;

        public GetAllArticulos(IRepositorioArticulo repo)
        {
            _repoArticulos = repo;
        }

        public IEnumerable<ArticuloDTO> Ejecutar()
        {
            var articulos = _repoArticulos.GetAll();
            if (articulos == null || articulos.Count() == 0)
            {
                throw new ArticuloNuloException("No se encontraron articulos en el sistema");
```

```
            }

            return ArticulosMappers.FromLista(articulos);

        }

    }

}
```

****

Archivo: UpdateArticulo.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\ImplementacionCasosUso\Articulos\UpdateArticulo.cs

****

```
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Articulos;

using Papeleria.LogicaAplicacion.DataTransferObjects.MapeosDatos;

using Papeleria.LogicaAplicacion.InterfacesCasosUso.Articulos;

using Papeleria.LogicaNegocio.Excepciones.Articulo;

using Papeleria.LogicaNegocio.InterfacesRepositorio;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace Papeleria.LogicaAplicacion.ImplementacionCasosUso.Articulos
```

```
{
    public class UpdateArticulo : IUpdateArticulo
    {

        private IRepositorioArticulo _repoArticulos;

        public UpdateArticulo(IRepositorioArticulo repo)
        {
            _repoArticulos = repo;
        }

        public void Ejecutar(int id, ArticuloDTO articuloMod)
        {
            if (articuloMod == null) { throw new ArticuloNuloException("Articulo modificado no puede ser nulo"); }
            try
            {
                var articulo = ArticulosMappers.FromDtoUpdate(articuloMod);
                _repoArticulos.Update(id, articulo);
            }
            catch (Exception ex) { throw new ArticuloNoValidoException(ex.Message); }
        }
    }
}
```

****

Archivo: AltaMovimiento.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\ImplementacionCasosUso\Movimiento\AltaMovimiento.cs

****

```
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.MovimientoStock;
using Papeleria.LogicaAplicacion.DataTransferObjects.MapeosDatos;
using Papeleria.LogicaAplicacion.Interaces;
using Papeleria.LogicaAplicacion.InterfacesCasosUso.Movimientos;
using Papeleria.LogicaNegocio.Entidades;
using Papeleria.LogicaNegocio.InterfacesRepositorio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Papeleria.LogicaAplicacion.ImplementacionCasosUso.Movimiento
{
    public class AltaMovimiento : IAltaMovimiento
    {
        private IRepositorioMovimientoStock _repoMov;
```

```
        private IRepositorioArticulo _repoArt;
        private IRepositorioUsuario _repoUsr;
        private IRepositorioTipoMovimiento _repoTipMov;
        public AltaMovimiento(IRepositorioMovimientoStock repo, IRepositorioArticulo repoArt, IRepositorioUsuario repoUsr, IRepositorioTipoMovimiento repoTipMov)
        {
            _repoMov = repo; _repoArt = repoArt; _repoUsr = repoUsr; _repoTipMov = repoTipMov ;
        }
        public void Crear(MovimientoStockDTO obj)
        {
            if(obj == null)
            {
                throw new Exception("No se puede ingresar algo nulo"); // TODO excepciones
            }
            if(obj.CtdUnidadesXMovimiento <= 0)
            {
                throw new Exception("No se puede realizar un movimiento de cantidades nulas o negativas");
            }
            MovimientoStock movimiento = MovimientoStockMappers.FromDTO(obj, _repoArt, _repoUsr, _repoTipMov);
            _repoMov.Add(movimiento);
        }
    }
}
```

****

Archivo: BuscarMovimiento.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\ImplementacionCasosUso\Movimiento\BuscarMovimiento.cs

****

```
using Empresa.LogicaDeNegocio.Entidades;

using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Articulos;

using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.MovimientoStock;

using Papeleria.LogicaAplicacion.DataTransferObjects.MapeosDatos;

using Papeleria.LogicaAplicacion.InterfacesCasoDeUsoGeneral;

using Papeleria.LogicaAplicacion.InterfacesCasosUso.Movimientos;

using Papeleria.LogicaNegocio.Entidades;

using Papeleria.LogicaNegocio.Excepciones.Articulo;

using Papeleria.LogicaNegocio.Excepciones.MovimientoStock;

using Papeleria.LogicaNegocio.Excepciones.Usuario;

using Papeleria.LogicaNegocio.InterfacesRepositorio;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;
```

```
namespace Papeleria.LogicaAplicacion.ImplementacionCasosUso.Movimiento
{
    public class BuscarMovimiento : IGetMovimiento
    {

        private IRepositorioMovimientoStock _repoMov;
        private IRepositorioArticulo _repoArt;
        private IRepositorioUsuario _repoUsuario;
        private IRepositorioTipoMovimiento _repoTipoMovimiento;
        public BuscarMovimiento(IRepositorioMovimientoStock repo, IRepositorioArticulo repoArt, IRepositorioUsuario repoUsuario, IRepositorioTipoMovimiento repoTipoMovimiento)
        {
            _repoMov = repo; _repoArt = repoArt;
            _repoUsuario = repoUsuario;
            _repoTipoMovimiento = repoTipoMovimiento;
        }
        public MovimientoStock Get(int id)
        {
            return _repoMov.GetById(id);
        }

        public IEnumerable<MovimientoStockDTO> GetAll()
        {
            var movimientos = _repoMov.GetAll();
```

```
        if (movimientos == null || movimientos.Count() == 0)
        {
            throw new MovimientoStockNuloException("No hay movimientos registrados");
        }
        return MovimientoStockMappers.FromLista(movimientos);
    }

    public IEnumerable<ArticuloDTO> GetArticulosByRangoFecha(DateTime fechaIni, DateTime fechaFin)
    {
        IEnumerable<Articulo> articulos = _repoMov.GetByRangoFechas(fechaIni,fechaFin);
        if (articulos.Count()==0)
        {
            throw new ArticuloNuloException("No se ha encontrado articulos que hayan tenido movimiento en ese rango de fechas."); // Handler de exception
        }
        var ret = ArticulosMappers.FromLista(articulos);
        return ret;
    }

    public MovimientoStockDTO GetByDTO(int id)
    {
        var movimiento = _repoMov.GetById(id);
        if (movimiento == null)
```

```
        {
            throw new MovimientoStockNuloException("Movimiento no encontrado con el ID especificado"); // Handler de exception
        }
        var ret = MovimientoStockMappers.ToDto(movimiento);
        return ret;
    }

    public IEnumerable<MovimientoStockDTO> GetMovimientosByIDArticuloYTipoMov(int idArticulo, string tipoMovimiento)
    {
        var movimientos = _repoMov.GetAllByIDArticulo_y_TipoMovimiento(idArticulo, tipoMovimiento);
        if (movimientos == null)
        {
            throw new MovimientoStockNuloException("No se ha encontrado movimientos con el ID de articulo y tipo de movimiento ingresado."); // Handler de exception
        }
        var ret = MovimientoStockMappers.FromLista(movimientos);
        return ret;
    }

    public IEnumerable<object> ObtenerResumenMovimientosPorAnioYTipoMovimiento()
    {
        var resumen = _repoMov.ObtenerResumenMovimientosPorAnioYTipoMovimiento();
```

```
            if (resumen == null)
            {
                throw new MovimientoStockNuloException("No se ha encontrado movimientos."); // Handler de exception
            }
            return resumen;
        }
    }
}
```

****

Archivo: FiltrarMovimiento.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\ImplementacionCasosUso\Movimientos\FiltrarMovimiento.cs

****

```
using Papeleria.LogicaAplicacion.DataTransferObjects.MapeosDatos;
using Papeleria.LogicaAplicacion.InterfacesCasosUso.Movimientos;
using Papeleria.LogicaNegocio.Excepciones.TipoMovimiento;
using Papeleria.LogicaNegocio.InterfacesRepositorio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
```

```
using System.Threading.Tasks;


namespace Papeleria.LogicaAplicacion.ImplementacionCasosUso.Movimientos
{
    public class FiltrarMovimiento : IFiltrarMovimiento
    {
        private IRepositorioMovimientoStock _repoMovimiento;


        public FiltrarMovimiento(IRepositorioMovimientoStock repo)
        {
            _repoMovimiento = repo;
        }
        public bool ExisteTipoMovimientoEnMovimientoByID(int id)
        {
            if (id == null)
            {
                throw new TipoMovimientoNuloException("Debe ingresar un ID para buscar el tipo movimiento");
            }
            try
            {
                var ctdMov = _repoMovimiento.CtdMovimientos();
                if(ctdMov==0)
                    return false;
```

```
            var mov = _repoMovimiento.EstaEnUsoTipoMovimientoByID(id);
            return mov;
        }
        catch (Exception ex)
        {
            throw new TipoMovimientoNuloException(ex.Message);
        }
    }

    public bool ExisteTipoMovimientoEnMovimientoByNombre(string nombre)
    {
        if (nombre == null)
        {
            throw new TipoMovimientoNuloException("Debe ingresar un ID para buscar el tipo movimiento");
        }
        try
        {
            var mov = _repoMovimiento.EstaEnUsoTipoMovimientoByNombre(nombre);
            return mov;
        }
        catch (Exception ex)
        {
            throw new TipoMovimientoNuloException(ex.Message);
```

```
        }
      }
    }
}
```

****

Archivo: AltaTipoMovimiento.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\ImplementacionCasosUso\TipoMovimientos\AltaTipoMovimiento.cs

****

```
using Empresa.LogicaDeNegocio.Sistema;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Usuarios;
using Papeleria.LogicaAplicacion.DataTransferObjects.MapeosDatos;
using Papeleria.LogicaNegocio.Excepciones.Usuario.UsuarioExcepcions.Email;
using Papeleria.LogicaNegocio.Excepciones.Usuario;
using Papeleria.LogicaNegocio.InterfacesRepositorio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Papeleria.LogicaNegocio.Excepciones.TipoMovimiento;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.TipoMovimientos;
```

```
using Papeleria.LogicaNegocio.Entidades;
using Papeleria.LogicaAplicacion.InterfacesCasosUso.TipoMovimientos;

namespace Papeleria.LogicaAplicacion.ImplementacionCasosUso.TipoMovimientos
{
    public class AltaTipoMovimiento : IAltaTiposMovimientos
    {
        private IRepositorioTipoMovimiento _repoTipoMovimiento;

        public AltaTipoMovimiento(IRepositorioTipoMovimiento repo)
        {
            _repoTipoMovimiento = repo;
        }

        public void Ejecutar(TipoMovimientoDTO dto)
        {
            if (dto == null)
                throw new TipoMovimientoNuloException("No han llegado datos.");

            bool emailExistente = _repoTipoMovimiento.ExisteTipoMovimientoXNombre(dto.Nombre);
            if (emailExistente)
            {
                throw new TipoMovimientoNoValidoException("El nombre del tipo de movimiento ya está en
uso.");
```

```
            }
            else
            {
                TipoMovimiento tipMov = TipoMovimientoMappers.FromDto(dto);
                _repoTipoMovimiento.Add(tipMov);
            }
        }
    }
}
```

****

Archivo: BorrarTipoMovimiento.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\ImplementacionCasosUso\TipoMovimientos\BorrarTipoMovimiento.cs

****

```
using Papeleria.LogicaAplicacion.InterfacesCasosUso.TipoMovimientos;
using Papeleria.LogicaNegocio.Entidades;
using Papeleria.LogicaNegocio.Excepciones.TipoMovimiento;
using Papeleria.LogicaNegocio.Excepciones.Usuario;
using Papeleria.LogicaNegocio.InterfacesRepositorio;
using System;
using System.Collections.Generic;
using System.Linq;
```

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


namespace Papeleria.LogicaAplicacion.ImplementacionCasosUso.TipoMovimientos
{
    public class BorrarTipoMovimiento : IBorrarTipoMovimiento
    {
        private IRepositorioTipoMovimiento _repoTipoMovimiento;
        public BorrarTipoMovimiento(IRepositorioTipoMovimiento repo)
        {
            _repoTipoMovimiento = repo;
        }
        public void Ejecutar(int id)
        {
            if (id == null)
                throw new TipoMovimientoNuloException("El ID que desea borrar no puede ser nulo.");
            try
            {
                var tipMov = _repoTipoMovimiento.GetById(id);
                if (tipMov == null)
                    throw new TipoMovimientoNuloException("No se encontro tipo de movimiento con ese ID.");
                else
                    _repoTipoMovimiento.Remove(tipMov);
```

```
            }
            catch (Exception ex)
            {
                throw new TipoMovimientoNoValidoException(ex.Message);
            }
        }

        public void Ejecutar(TipoMovimiento articulo)
        {
            try
            {
                if (articulo == null)
                    throw new TipoMovimientoNuloException("El Tipo Movimiento que desea borrar no puede ser nulo.");
                _repoTipoMovimiento.Remove(articulo);
            }
            catch (Exception ex)
            {
                throw new TipoMovimientoNoValidoException(ex.Message);
            }
        }
    }
}
```

****

Archivo: BuscarTipoMovimiento.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\ImplementacionCasosUso\TipoMovimientos\BuscarTipoMovimiento.cs

****

```
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.TipoMovimientos;
using Papeleria.LogicaAplicacion.DataTransferObjects.MapeosDatos;
using Papeleria.LogicaAplicacion.InterfacesCasosUso.TipoMovimientos;
using Papeleria.LogicaNegocio.Entidades;
using Papeleria.LogicaNegocio.Excepciones.TipoMovimiento;
using Papeleria.LogicaNegocio.Excepciones.Usuario;
using Papeleria.LogicaNegocio.InterfacesRepositorio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Papeleria.LogicaAplicacion.ImplementacionCasosUso.TipoMovimientos
{
    public class BuscarTipoMovimiento : IGetTipoMovimiento
    {
        private IRepositorioTipoMovimiento _repoTipoMovimiento;
```

```
public BuscarTipoMovimiento(IRepositorioTipoMovimiento repo)
{
    _repoTipoMovimiento = repo;
}
public TipoMovimiento GetById(int id)
{
    try
    {
        return _repoTipoMovimiento.GetById(id);
    }
    catch (Exception ex)
    {
        throw new TipoMovimientoNuloException(ex.Message);
    }
}


public TipoMovimientoDTO GetByIdDTO(int id)
{
    try
    {
        var tipMov = _repoTipoMovimiento.GetById(id);
        if (tipMov == null)
        {
```

```
            throw new TipoMovimientoNuloException("No hay tipo movimiento con ese id");
        }
        var tmDTO = TipoMovimientoMappers.ToDto(tipMov);
        return tmDTO;
    }
    catch (Exception ex)
    {
        throw new TipoMovimientoNuloException(ex.Message);
    }

}

public TipoMovimiento GetByNombre(string nombre)
{
    try
    {
        return _repoTipoMovimiento.GetTipoMovimientoXNombre(nombre);
    }
    catch (Exception ex)
    {
        throw new TipoMovimientoNuloException(ex.Message);
    }
}
```

```
    }
}
```

****

Archivo: GetAllTiposMovimientos.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\ImplementacionCasosUso\TipoMovimientos\GetAllTiposMovimientos.cs

****

```
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.TipoMovimientos;
using Papeleria.LogicaAplicacion.DataTransferObjects.MapeosDatos;
using Papeleria.LogicaAplicacion.InterfacesCasosUso.TipoMovimientos;
using Papeleria.LogicaNegocio.Excepciones.TipoMovimiento;
using Papeleria.LogicaNegocio.Excepciones.Usuario;
using Papeleria.LogicaNegocio.InterfacesRepositorio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Papeleria.LogicaAplicacion.ImplementacionCasosUso.TipoMovimientos
{
    public class GetAllTiposMovimientos : IGetAllTipoMovimiento
```

```
{
    private IRepositorioTipoMovimiento _repoTipoMovimiento;
    public GetAllTiposMovimientos(IRepositorioTipoMovimiento repo)
    {
        _repoTipoMovimiento = repo;
    }
    public IEnumerable<TipoMovimientoDTO> Ejecutar()
    {
        try
        {
            var tmOrigen = _repoTipoMovimiento.GetAll();
            if (tmOrigen == null || tmOrigen.Count() == 0)
            {
                throw new TipoMovimientoNuloException("No hay tipo movimientos registrados");
            }
            return TipoMovimientoMappers.FromLista(tmOrigen);
        }
        catch (Exception ex)
        {
            throw new TipoMovimientoNoValidoException(ex.Message);
        }

    }
```

```
    }
}
```

****

Archivo: ModificarTipoMovimiento.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\ImplementacionCasosUso\TipoMovimientos\ModificarTipoMovimiento.cs

****

```
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.TipoMovimientos;
using Papeleria.LogicaAplicacion.DataTransferObjects.MapeosDatos;
using Papeleria.LogicaAplicacion.InterfacesCasosUso.TipoMovimientos;
using Papeleria.LogicaNegocio.Excepciones.TipoMovimiento;
using Papeleria.LogicaNegocio.Excepciones.Usuario;
using Papeleria.LogicaNegocio.InterfacesRepositorio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Papeleria.LogicaAplicacion.ImplementacionCasosUso.TipoMovimientos
{
    public class ModificarTipoMovimiento : IUpdateTipoMovimiento
```

```
    {
        private IRepositorioTipoMovimiento _repoTipoMovimiento;
        public ModificarTipoMovimiento(IRepositorioTipoMovimiento repo)
        {
            _repoTipoMovimiento = repo;
        }

        public void Ejecutar(int id, TipoMovimientoDTO tipoMovModificado)
        {
            if (tipoMovModificado == null)
                throw new TipoMovimientoNuloException("Tipo de movimiento no puede ser nulo al modificar.");
            try
            {
                var tp = TipoMovimientoMappers.FromDtoUpdate(tipoMovModificado);
                _repoTipoMovimiento.Update(id, tp);
            }
            catch (Exception ex)
            {
                throw new TipoMovimientoNoValidoException(ex.Message);
            }
        }
    }
}
```

****

Archivo: AltaUsuarios.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\ImplementacionCasosUso\Usuarios\AltaUsuarios.cs

****

```
using Empresa.LogicaDeNegocio.Sistema;

using Papeleria.AccesoDatos.EF;

using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Usuarios;

using Papeleria.LogicaAplicacion.DataTransferObjects.MapeosDatos;

using Papeleria.LogicaAplicacion.Interaces;

using Papeleria.LogicaAplicacion.InterfacesCasosUso.Usuarios;

using Papeleria.LogicaNegocio.Excepciones.Usuario;

using Papeleria.LogicaNegocio.Excepciones.Usuario.UsuarioExcepcions.Email;

using Papeleria.LogicaNegocio.InterfacesRepositorio;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace Papeleria.LogicaAplicacion.ImplementacionCasosUso.Usuarios

{
```

```
public class AltaUsuarios : IAltaUsuario
{
    private IRepositorioUsuario _repoUsuarios;

    public AltaUsuarios(IRepositorioUsuario repo)
    {
        _repoUsuarios = repo;
    }

    public void Ejecutar(UsuarioDTO dto)
    {
        if (dto == null)
            throw new UsuarioNuloExcepcion("No han llegado datos.");

        bool emailExistente = _repoUsuarios.ExisteUsuarioConEmail(dto.Email);
        if (emailExistente)
        {
            throw new EmailNoValidoException("El email ya está en uso.");
        }
        else
        {
            var usuario = UsuariosMappers.FromDto(dto);
            _repoUsuarios.Add(usuario);
```

```
        }
      }
    }
}
```

****

Archivo: BorrarUsuario.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\ImplementacionCasosUso\Usuarios\BorrarUsuario.cs

****

```
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Usuarios;
using Papeleria.LogicaAplicacion.DataTransferObjects.MapeosDatos;
using Papeleria.LogicaAplicacion.InterfacesCasosUso.Usuarios;
using Papeleria.LogicaNegocio.Excepciones.Usuario;
using Papeleria.LogicaNegocio.InterfacesRepositorio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Papeleria.LogicaAplicacion.ImplementacionCasosUso.Usuarios
{
```

```
public class BorrarUsuario : IBorrarUsuario
{
    private IRepositorioUsuario _repoUsuarios;

    public BorrarUsuario(IRepositorioUsuario repo)
    {
        _repoUsuarios = repo;
    }

    public void Ejecutar(int id, UsuarioDTO usuarioBorrar)
    {
        if (usuarioBorrar == null)
            throw new UsuarioNuloExcepcion("Usuario no puede ser nulo.");
        try
        {
            var usuario = _repoUsuarios.GetById(usuarioBorrar.Id);
            _repoUsuarios.Remove(usuario);
        }
        catch (Exception ex)
        {
            throw new UsuarioNoValidoExcepcion(ex.Message);
        }
    }
```

```
    }
}
```

****

Archivo: BuscarUsuario.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\ImplementacionCasosUso\Usuarios\BuscarUsuario.cs

****

```
using Empresa.LogicaDeNegocio.Sistema;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Usuarios;
using Papeleria.LogicaAplicacion.DataTransferObjects.MapeosDatos;
using Papeleria.LogicaAplicacion.Interaces;
using Papeleria.LogicaAplicacion.InterfacesCasosUso.Usuarios;
using Papeleria.LogicaNegocio.Entidades;
using Papeleria.LogicaNegocio.Excepciones.Usuario;
using Papeleria.LogicaNegocio.InterfacesRepositorio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Papeleria.LogicaAplicacion.ImplementacionCasosUso.Usuarios
```

```
{
    public class BuscarUsuario : IGetUsuario
    {
        private IRepositorioUsuario _repoUsuarios;


        public BuscarUsuario(IRepositorioUsuario repo)
        {
            _repoUsuarios = repo;
        }
        public UsuarioDTO GetByIdDTO(int id)
        {
            var usu = _repoUsuarios.GetById(id);
            if (usu == null)
            {
                throw new UsuarioNuloExcepcion("No hay usuario con ese id");
            }
            var usuDto = UsuariosMappers.ToDto(usu);
            return usuDto;
        }
        public Usuario GetById(int id)
        {
            return _repoUsuarios.GetById(id);
        }
```

```
        public Usuario GetEncargadoByID(int id)
        {
            Usuario usr = _repoUsuarios.GetById(id);
            if (usr.GetType().Name == "EncargadoDeposito")
            {
                return usr;
            } else
            {
                throw new Exception("El ID especificado no es de un encargado del deposito"); // TODO Exception Handler
            }
        }
    }


}
```

****

Archivo: GetAllUsuarios.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\ImplementacionCasosUso\Usuarios\GetAllUsuarios.cs

```
**********
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Usuarios;
using Papeleria.LogicaAplicacion.DataTransferObjects.MapeosDatos;
using Papeleria.LogicaAplicacion.InterfacesCasosUso.Usuarios;
using Papeleria.LogicaNegocio.Excepciones.Usuario;
using Papeleria.LogicaNegocio.InterfacesRepositorio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Papeleria.LogicaAplicacion.ImplementacionCasosUso.Usuarios
{
    public class GetAllUsuarios : IGetAllUsuarios
    {
        private IRepositorioUsuario _repositorioUsuarios;
        public GetAllUsuarios(IRepositorioUsuario repo)
        {
            _repositorioUsuarios = repo;
        }
        public IEnumerable<UsuarioDTO> Ejecutar()
        {
```

```
            var usuariosOrigen = _repositorioUsuarios.GetAll();

            if (usuariosOrigen == null || usuariosOrigen.Count() == 0)

            {

                throw new UsuarioNuloExcepcion("No hay autores registrados");

            }

            return UsuariosMappers.FromLista(usuariosOrigen);

        }

    }

}
```

****

Archivo: Login.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\ImplementacionCasosUso\Usuarios\Login.cs

****

```
using Empresa.LogicaDeNegocio.Sistema;

using Papeleria.LogicaAplicacion.InterfacesCasosUso.Usuarios;

using Papeleria.LogicaNegocio.InterfacesRepositorio;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;
```

```
namespace Papeleria.LogicaAplicacion.ImplementacionCasosUso.Usuarios
{
    public class Login : ILogin
    {
        private IRepositorioUsuario _repoUsr;
        public Login(IRepositorioUsuario repoUsr)
        {
            _repoUsr = repoUsr;
        }
        public Usuario Ejecutar(string email, string pwd)
        {
            return _repoUsr.Login(email, pwd);
            /*var usr = _repoUsr.Login(email, pwd);
            if (usr == null)
                return string.Empty;
            return usr.GetType().Name;*/
        }
    }
}
```

****

Archivo: ModificarUsuario.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\ImplementacionCasosUso\Usuarios\ModificarUsuario.cs

****

```
using Empresa.LogicaDeNegocio.Sistema;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Usuarios;
using Papeleria.LogicaAplicacion.DataTransferObjects.MapeosDatos;
using Papeleria.LogicaAplicacion.Interaces;
using Papeleria.LogicaAplicacion.InterfacesCasosUso.Usuarios;
using Papeleria.LogicaNegocio.Excepciones.Usuario;
using Papeleria.LogicaNegocio.InterfacesRepositorio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Papeleria.LogicaAplicacion.ImplementacionCasosUso.Usuarios
{
    public class ModificarUsuario:IModificarUsuario
    {
        private IRepositorioUsuario _repoUsuarios;

        public ModificarUsuario(IRepositorioUsuario repo)
```

```
        {
            _repoUsuarios = repo;
        }

        public void Ejecutar(int id, UsuarioDTO usuarioModificado)
        {
            if (usuarioModificado == null)
                throw new UsuarioNuloExcepcion("Usuario no puede ser nulo.");
            try
            {
                var usuario = UsuariosMappers.FromDtoUpdate(usuarioModificado);
                _repoUsuarios.Update(id, usuario);
            }
            catch (Exception ex)
            {
                throw new UsuarioNoValidoExcepcion(ex.Message);
            }

        }
    }
}
```

****

Archivo: IAltaArticulo.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasosUso\Articulos\IAltaArticulo.cs

****

```
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Articulos;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Usuarios;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Papeleria.LogicaAplicacion.InterfacesCasosUso.Articulos
{
    public interface IAltaArticulo
    {
        void Ejecutar(ArticuloDTO dto);
    }
}
```

****

Archivo: IBorrarArticulo.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasosUso\Articulos\IBorrarArticulo.cs

****

```
using Empresa.LogicaDeNegocio.Entidades;

using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Articulos;

using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Usuarios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace Papeleria.LogicaAplicacion.InterfacesCasosUso.Articulos

{

    public interface IBorrarArticulo

    {

        void Ejecutar(int id);

        void Ejecutar(Articulo articulo);

    }

}
```

****

Archivo: IFiltrarArticulo.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasosUso\Articulos\IFiltrarArticulo.cs

****

```
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Articulos;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Usuarios;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Papeleria.LogicaAplicacion.InterfacesCasosUso.Articulos
{
    public interface IFiltrarArticulo
    {
        IEnumerable<ArticuloDTO> GetArticuloPorCodigo(long codigoProveedor);
        IEnumerable<ArticuloDTO> GetArticulosPorNombre(string nombre);

    }
}
```

****

Archivo: IGetAllArticulos.cs

Carpeta:
C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasosUso\Articulos\IGetAllArticulos.cs

****

```
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Articulos;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Usuarios;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Papeleria.LogicaAplicacion.InterfacesCasosUso.Articulos
{
    public interface IGetAllArticulos
    {
        public IEnumerable<ArticuloDTO> Ejecutar();
    }
}
```

****

Archivo: IGetArticulo.cs

Carpeta:
C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasosUso\Articulos\IGetArticulo.cs

****

```
using Empresa.LogicaDeNegocio.Entidades;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Articulos;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Usuarios;
using Papeleria.LogicaNegocio.Entidades.ValueObjects.Articulos;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Papeleria.LogicaAplicacion.InterfacesCasosUso.Articulos
{
    public interface IGetArticulo
    {
        ArticuloDTO GetByIdDTO(int id);
        Articulo GetById(int id);
        ArticuloDTO GetArticuloPorCodigo(CodigoProveedorArticulos codigoProveedor);
        IEnumerable<ArticuloDTO> GetArticulosPorNombre(string nombre);
    }
}
```

****

Archivo: IUpdateArticulo.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasosUso\Articulos\IUpdateArticulo.cs

****

```
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Articulos;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Usuarios;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Papeleria.LogicaAplicacion.InterfacesCasosUso.Articulos
{
    public interface IUpdateArticulo
    {
        void Ejecutar(int id, ArticuloDTO articuloModificado);
    }
}
```

****

Archivo: IAltaMovimiento.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasosUso\MovimientoStock\IAltaMovimiento.cs

****

```
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.MovimientoStock;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Papeleria.LogicaAplicacion.InterfacesCasosUso.Movimientos
{
    public interface IAltaMovimiento
    {
        public void Crear(MovimientoStockDTO obj);
    }
}
```

****

Archivo: IBorrarMovimiento.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasosUso\MovimientoStock\IBorrarMovimiento.cs

****

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


namespace Papeleria.LogicaAplicacion.InterfacesCasosUso.Movimientos
{
    public interface IBorrarMovimiento
    {
        public void Remove(int id);
        public void Remove(Papeleria.LogicaNegocio.Entidades.MovimientoStock obj);
    }
}
```

****

Archivo: IFiltrarMovimiento.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasosUso\MovimientoStock\IFiltrarMovimiento.cs

****

```
using System;
```

```
using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace Papeleria.LogicaAplicacion.InterfacesCasosUso.Movimientos

{

    public interface IFiltrarMovimiento

    {

        bool ExisteTipoMovimientoEnMovimientoByNombre(string nombre);

        bool ExisteTipoMovimientoEnMovimientoByID(int id);

    }

}
```

****

Archivo: IGetMovimiento.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasosUso\MovimientoStock\IGetMovimiento.cs

****

```
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Articulos;

using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.MovimientoStock;

using Papeleria.LogicaNegocio.Entidades;

using Papeleria.LogicaNegocio.InterfacesRepositorio;
```

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

namespace Papeleria.LogicaAplicacion.InterfacesCasosUso.Movimientos
{
    public interface IGetMovimiento
    {
        public Papeleria.LogicaNegocio.Entidades.MovimientoStock Get(int id);
        public MovimientoStockDTO GetByDTO(int id);
        public IEnumerable<MovimientoStockDTO> GetAll();
        public IEnumerable<MovimientoStockDTO> GetMovimientosByIDArticuloYTipoMov(int idArticulo, string tipoMovimiento);
        public IEnumerable<ArticuloDTO> GetArticulosByRangoFecha(DateTime fechaIni, DateTime fechaFin);
        public IEnumerable<object> ObtenerResumenMovimientosPorAnioYTipoMovimiento();
    }
}
```

****

Archivo: IUpdateMovimiento.cs

Carpeta:
C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasosUso\MovimientoStock\IUpdateMovimiento.cs

****

```
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.MovimientoStock;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace Papeleria.LogicaAplicacion.InterfacesCasosUso.MovimientoStock

{

    public interface IUpdateMovimiento

    {

        public void Update(int id, MovimientoStockDTO obj);

    }

}
```

****

Archivo: IAltaTiposMovimientos.cs

Carpeta:
C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasosUso\TipoMovimientos\IAltaTiposMovimientos.cs

****

```
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.TipoMovimientos;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace Papeleria.LogicaAplicacion.InterfacesCasosUso.TipoMovimientos

{

   public interface IAltaTiposMovimientos

   {

      void Ejecutar(TipoMovimientoDTO dto);

   }

}
```

****

Archivo: IBorrarTipoMovimiento.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasosUso\TipoMovimientos\IBorrarTipoMovimiento.cs

****

```
using Empresa.LogicaDeNegocio.Entidades;

using Papeleria.LogicaNegocio.Entidades;

using System;
```

```
using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace Papeleria.LogicaAplicacion.InterfacesCasosUso.TipoMovimientos

{

   public interface IBorrarTipoMovimiento

   {

      void Ejecutar(int id);

      void Ejecutar(TipoMovimiento articulo);

   }

}
```

****

Archivo: IFiltrarTipoMovimiento.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasosUso\TipoMovimientos\IFiltrarTipoMovimiento.cs

****

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;
```

```
using System.Threading.Tasks;


namespace Papeleria.LogicaAplicacion.InterfacesCasosUso.TipoMovimientos
{
    public interface IFiltrarTipoMovimiento
    {


    }
}
```

****

Archivo: IGetAllTipoMovimiento.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasosUso\TipoMovimientos\IGetAllTipoMovimiento.cs

****

```
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.TipoMovimientos;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;


namespace Papeleria.LogicaAplicacion.InterfacesCasosUso.TipoMovimientos
```

```
{
    public interface IGetAllTipoMovimiento
    {
        public IEnumerable<TipoMovimientoDTO> Ejecutar();
    }
}
```

****

Archivo: IGetTipoMovimiento.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasosUso\TipoMovimientos\IGetTipoMovimiento.cs

****

```
using Empresa.LogicaDeNegocio.Entidades;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Articulos;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.TipoMovimientos;
using Papeleria.LogicaNegocio.Entidades;
using Papeleria.LogicaNegocio.Entidades.ValueObjects.Articulos;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
```

```
namespace Papeleria.LogicaAplicacion.InterfacesCasosUso.TipoMovimientos
{
    public interface IGetTipoMovimiento
    {
        TipoMovimientoDTO GetByIdDTO(int id);
        TipoMovimiento GetById(int id);

        TipoMovimiento GetByNombre(string nombre);
    }
}
```

****

Archivo: IUpdateTipoMovimiento.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasosUso\TipoMovimientos\IUpdateTipoMovimiento.cs

****

```
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Articulos;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.TipoMovimientos;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
```

```
namespace Papeleria.LogicaAplicacion.InterfacesCasosUso.TipoMovimientos
{
    public interface IUpdateTipoMovimiento
    {
        void Ejecutar(int id, TipoMovimientoDTO tipoMovModificado);
    }
}
```

****

Archivo: IAltaUsuario.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasosUso\Usuarios\IAltaUsuario.cs

****

```
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Articulos;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Usuarios;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Papeleria.LogicaAplicacion.InterfacesCasosUso.Usuarios
```

```
{
    public interface IAltaUsuario
    {
        void Ejecutar(UsuarioDTO dto);
    }
}
```

****

Archivo: IBorrarUsuario.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasosUso\Usuarios\IBorrarUsuario.cs

****

```
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Usuarios;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Papeleria.LogicaAplicacion.InterfacesCasosUso.Usuarios
{
    public interface IBorrarUsuario
    {
```

```
        void Ejecutar(int id, UsuarioDTO usu);
    }
}
```

****

Archivo: IFiltrarUsuarios.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasosUso\Usuarios\IFiltrarUsuarios.cs

****

```
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Usuarios;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Papeleria.LogicaAplicacion.InterfacesCasosUso.Usuarios
{
    public interface IFiltrarUsuarios
    {
        IEnumerable<UsuarioDTO> GetUsuarioPorEmail(string email);
    }
}
```

****

Archivo: IGetAllUsuarios.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasosUso\Usuarios\IGetAllUsuarios.cs

****

```
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Usuarios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace Papeleria.LogicaAplicacion.InterfacesCasosUso.Usuarios

{

   public interface IGetAllUsuarios

   {

      public IEnumerable<UsuarioDTO> Ejecutar();

   }

}
```

****

Archivo: IGetUsuario.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasosUso\Usuarios\IGetUsuario.cs

****

```
using Empresa.LogicaDeNegocio.Sistema;
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Usuarios;
using Papeleria.LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Papeleria.LogicaAplicacion.InterfacesCasosUso.Usuarios
{
    public interface IGetUsuario
    {
        UsuarioDTO GetByIdDTO(int id);
        Usuario GetById(int id);
        Usuario GetEncargadoByID(int id);
    }
}
```

****

Archivo: ILogin.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasosUso\Usuarios\ILogin.cs

****

```
using Empresa.LogicaDeNegocio.Sistema;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;


namespace Papeleria.LogicaAplicacion.InterfacesCasosUso.Usuarios
{
    public interface ILogin
    {
        public Usuario Ejecutar(string email, string pwd);
    }
}
```

****

Archivo: IModificarUsuario.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\InterfacesCasosUso\Usuarios\IModificarUsuario.cs

****

```
using Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Usuarios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace Papeleria.LogicaAplicacion.InterfacesCasosUso.Usuarios

{

    public interface IModificarUsuario

    {

        void Ejecutar(int id, UsuarioDTO usuarioModificado);

    }

}
```

****

Archivo: AdminstradorNoValidoException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Administrador\AdminstradorNoValidoException.cs

****

```
using System;

using System.Collections.Generic;
```

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

namespace Papeleria.LogicaNegocio.Excepciones.Administrador
{
    public class AdminstradorNoValidoException : Exception
    {
        public AdminstradorNoValidoException()
        {
        }

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

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

        protected AdminstradorNoValidoException(SerializationInfo info, StreamingContext context) : base(info, context)
```

```
        {

        }

    }

}
```

****

Archivo: ArticuloDuplicadoException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Articulo\ArticuloDuplicadoException.cs

****

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace Papeleria.LogicaNegocio.Excepciones.Articulo

{

    internal class ArticuloDuplicadoException

    {

    }

}
```

****

Archivo: ArticuloNoValidoException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Articulo\ArticuloNoValidoException.cs

****

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.Text;

using System.Threading.Tasks;


namespace Papeleria.LogicaNegocio.Excepciones.Articulo

{

    public class ArticuloNoValidoException : Exception

    {

        public ArticuloNoValidoException()

        {

        }


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

        {

        }
```

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

    protected ArticuloNoValidoException(SerializationInfo info, StreamingContext context) : base(info, context)
    {
    }
  }
}
```

****

Archivo: ArticuloNuloException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Articulo\ArticuloNuloException.cs

****

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

```
namespace Papeleria.LogicaNegocio.Excepciones.Articulo
{
    public class ArticuloNuloException : Exception
    {
        public ArticuloNuloException()
        {
        }

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

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

        protected ArticuloNuloException(SerializationInfo info, StreamingContext context) : base(info, context)
        {
        }
    }
}
```

****

Archivo: EncargadoNoValidoException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\EncargadoDeposito\EncargadoNoValidoException.cs

****

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.Text;

using System.Threading.Tasks;


namespace Papeleria.LogicaNegocio.Excepciones.EncargadoDeposito

{

    public class EncargadoNoValidoException : Exception

    {

        public EncargadoNoValidoException()

        {

        }


        public EncargadoNoValidoException(string? message) : base(message)

        {
```

```
        }

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

        protected EncargadoNoValidoException(SerializationInfo info, StreamingContext context) : base(info, context)
        {
        }
    }
}
```

****

Archivo: MovimientoStockDuplicadoException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\MovimientoStock\MovimientoStockDuplicadoException.cs

****

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.Text;
```

```
using System.Threading.Tasks;

namespace Papeleria.LogicaNegocio.Excepciones.MovimientoStock
{
    public class MovimientoStockDuplicadoException : Exception
    {
        public MovimientoStockDuplicadoException()
        {
        }

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

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

        protected MovimientoStockDuplicadoException(SerializationInfo info, StreamingContext context) : base(info, context)
        {
        }
    }
```

}

****

Archivo: MovimientoStockNoValidoException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\MovimientoStock\MovimientoStockNoValidoException.cs

****

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.Text;

using System.Threading.Tasks;


namespace Papeleria.LogicaNegocio.Excepciones.MovimientoStock

{

    public class MovimientoStockNoValidoException : Exception

    {

        public MovimientoStockNoValidoException()

        {

        }


        public MovimientoStockNoValidoException(string? message) : base(message)
```

```
        {

        }


        public MovimientoStockNoValidoException(string? message, Exception? innerException) : base(message, innerException)

        {

        }


        protected MovimientoStockNoValidoException(SerializationInfo info, StreamingContext context) : base(info, context)

        {

        }

    }

}
```

****

Archivo: MovimientoStockNuloException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\MovimientoStock\MovimientoStockNuloException.cs

****

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;
```

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

namespace Papeleria.LogicaNegocio.Excepciones.MovimientoStock
{
    public class MovimientoStockNuloException : Exception
    {
        public MovimientoStockNuloException()
        {
        }

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

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

        protected MovimientoStockNuloException(SerializationInfo info, StreamingContext context) : base(info, context)
        {
        }
```

```
    }
}
```

****

Archivo: TipoMovimientoDuplicadoException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\TipoMovimiento\TipoMovimientoDuplicadoException.cs

****

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

namespace Papeleria.LogicaNegocio.Excepciones.TipoMovimiento
{
    public class TipoMovimientoDuplicadoException : Exception
    {
        public TipoMovimientoDuplicadoException()
        {
        }
```

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

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

        protected TipoMovimientoDuplicadoException(SerializationInfo info, StreamingContext context) : base(info, context)
        {
        }
    }
}
```

****

Archivo: TipoMovimientoNoValidoException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\TipoMovimiento\TipoMovimientoNoValidoException.cs

****

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

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

namespace Papeleria.LogicaNegocio.Excepciones.TipoMovimiento
{
    public class TipoMovimientoNoValidoException : Exception
    {
        public TipoMovimientoNoValidoException()
        {
        }

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

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

        protected TipoMovimientoNoValidoException(SerializationInfo info, StreamingContext context) : base(info, context)
        {
```

```
        }
    }
}
```

****

Archivo: TipoMovimientoNuloException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\TipoMovimiento\TipoMovimientoNuloException.cs

****

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

namespace Papeleria.LogicaNegocio.Excepciones.TipoMovimiento
{
    public class TipoMovimientoNuloException : Exception
    {
        public TipoMovimientoNuloException()
        {
        }
```

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

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

        protected TipoMovimientoNuloException(SerializationInfo info, StreamingContext context) : base(info, context)
        {
        }
    }
}
```

****

Archivo: UsuarioDuplicadoExcepcion.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Usuario\UsuarioDuplicadoExcepcion.cs

****

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

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

namespace Papeleria.LogicaNegocio.Excepciones.Usuario
{
    public class UsuarioDuplicadoExcepcion : Exception
    {
        public UsuarioDuplicadoExcepcion()
        {
        }

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

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

        protected UsuarioDuplicadoExcepcion(SerializationInfo info, StreamingContext context) : base(info, context)
```

```
        {

        }

    }

}
```

****

Archivo: UsuarioNoValidoExcepcion.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Usuario\UsuarioNoValidoExcepcion.cs

****

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.Text;

using System.Threading.Tasks;


namespace Papeleria.LogicaNegocio.Excepciones.Usuario

{

    public class UsuarioNoValidoExcepcion : Exception

    {

        public UsuarioNoValidoExcepcion()

        {
```

```
        }

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

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

        protected UsuarioNoValidoExcepcion(SerializationInfo info, StreamingContext context) : base(info, context)
        {
        }
    }
}
```

****

Archivo: UsuarioNuloExcepcion.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Usuario\UsuarioNuloExcepcion.cs

****

```
using System;
```

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

namespace Papeleria.LogicaNegocio.Excepciones.Usuario
{
    public class UsuarioNuloExcepcion : Exception
    {
        public UsuarioNuloExcepcion()
        {
        }

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

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

```
        protected UsuarioNuloExcepcion(SerializationInfo info, StreamingContext context) : base(info, context)
        {
        }
    }
}
```

****

Archivo: ArticuloDTO.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\DataTransferObjects\Dtos\Articulos\ArticuloDTO.cs

****

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

namespace Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Articulos
{
    public class ArticuloDTO
    {
```

```
        public int Id { get; set; }

        public long CodigoProveedor { get; set; }

        public string NombreArticulo { get; set; }

        public string Descripcion { get; set; }

        public double PrecioVP { get; set; }

    }

}
```

****

Archivo: MovimientoStockDTO.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\DataTransferObjects\Dtos\MovimientoStock\MovimientoStockDTO.cs

****

```
using Empresa.LogicaDeNegocio.Entidades;

using Papeleria.LogicaNegocio.Entidades;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.MovimientoStock

{
```

```
    public class MovimientoStockDTO
    {
        public int ID { get; set; }
        public DateTime FecHorMovRealizado { get; set; }
        public int ArticuloID { get; set; }
        public long CodigoProveedor { get; set; }
        public string NombreArticulo { get; set; }
        public string Descripcion { get; set; }
        public double PrecioVP { get; set; }
        public int TipoMovimientoID { get; set; }
        public string TipoMovimientoNombre { get; set; }
        public int UsuarioID { get; set; }
        public string Email { get; set; }
        public string Nombre { get; set; }
        public string Apellido { get; set; }
        public string Contrasenia { get; set; }
        public int CtdUnidadesXMovimiento { get; set; }
    }
}
```

****

Archivo: TipoMovimientoDTO.cs

Carpeta:
C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\DataTransferObjects\Dtos\TipoMovimientos\TipoMovimientoDTO.cs

****

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.TipoMovimientos

{

   public class TipoMovimientoDTO

   {

      public int ID { get; set; }

      public string Nombre { get; set; }

   }

}
```

****

Archivo: UsuarioDTO.cs

Carpeta:
C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaAplicacion\DataTransferObjects\Dtos\Usuarios\UsuarioDTO.cs

****

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace Papeleria.LogicaAplicacion.DataTransferObjects.Dtos.Usuarios

{

   public class UsuarioDTO

   {

      public int Id { get; set; }

      public int Admin {  get; set; }

      public string Email { get; set; }

      public string Nombre { get; set; }

      public string Apellido { get; set; }

      public string Contrasenia { get; set; }

   }

}
```

****

Archivo: CodigoProveedorArticulos.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Entidades\ValueObjects\Articulos\CodigoProveedorArticulos.cs

```
**********
using Empresa.LogicaDeNegocio.Entidades;
using Papeleria.LogicaNegocio.Excepciones.Articulo.ArticulosValueObjects.CodigoProveedor;
using Papeleria.LogicaNegocio.InterfacesEntidades;
using System.ComponentModel.DataAnnotations.Schema;

namespace Papeleria.LogicaNegocio.Entidades.ValueObjects.Articulos
{
    [ComplexType]
    public record CodigoProveedorArticulos : IValidable<CodigoProveedorArticulos>
    {
        public long codigo { get; init; }
        public CodigoProveedorArticulos()
        {

        }

        public CodigoProveedorArticulos(long codigo)
        {
            if (codigo == null) {
                throw new CodigoProveedorNuloException("El codigo no puede ser nulo");
            }
            this.codigo = codigo;
```

```
            esValido();
        }

        public void esValido()
        {
            if (codigo.ToString().Length != 13) {
                throw new CodigoProveedorNoValidoException("No puede tener un largo distinto de 13 digitos.");
            }
        }
    }

}
```

****

Archivo: DescripcionArticulo.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Entidades\ValueObjects\Articulos\DescripcionArticulo.cs

****

```
using Empresa.LogicaDeNegocio.Entidades;
using Papeleria.LogicaNegocio.Excepciones.Articulo.ArticulosValueObjects.DescripcionArticulo;
using Papeleria.LogicaNegocio.InterfacesEntidades;
```

```
using System.ComponentModel.DataAnnotations.Schema;
using System.Drawing;

namespace Papeleria.LogicaNegocio.Entidades.ValueObjects.Articulos
{
    [ComplexType]
    public record DescripcionArticulo : IValidable<DescripcionArticulo>
    {
        public string Descripcion { get; init; }
        public DescripcionArticulo()
        {

        }

        public DescripcionArticulo(string descripcion)
        {

            Descripcion = descripcion;
            esValido();
        }

        public void esValido()
        {
```

```
            if (Descripcion.Length < 5) {

                throw new DescripcionArticuloNoValidoException("La descripcion no puede ser menor a 5 caracteres.");

            }

        }

    }


}
```

****

Archivo: NombreArticulo.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Entidades\ValueObjects\Articulos\NombreArticulo.cs

****

```
using Empresa.LogicaDeNegocio.Entidades;

using Papeleria.LogicaNegocio.Excepciones.Articulo;

using Papeleria.LogicaNegocio.Excepciones.Articulo.ArticulosValueObjects.NombreArticulo;

using Papeleria.LogicaNegocio.InterfacesEntidades;

using System.ComponentModel.DataAnnotations.Schema;


namespace Papeleria.LogicaNegocio.Entidades.ValueObjects.Articulos

{
```

```
[ComplexType]
public record NombreArticulo : IValidable<NombreArticulo>, IEquatable<NombreArticulo>
{
    public string Nombre { get; init; }

    public NombreArticulo(string nombre)
    {
        Nombre = nombre;
        esValido();
    }

    public NombreArticulo()
    {

    }
    public void esValido()
    {
        if (Nombre == null || Nombre.Length<1) {
            throw new NombreArticuloNuloException("El nombre del articulo no puede ser nulo o vacio.");
        }

    }
}
```

}

****

Archivo: ContraseniaUsuario.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Entidades\ValueObjects\Usuarios\ContraseniaUsuario.cs

****

```
using Empresa.LogicaDeNegocio.Entidades;
using Papeleria.LogicaNegocio.Excepciones.Usuario.UsuarioExcepcions.Constrasenia;
using Papeleria.LogicaNegocio.Excepciones.Usuario.UsuarioExcepcions.Email;
using Papeleria.LogicaNegocio.InterfacesEntidades;
using System.ComponentModel.DataAnnotations.Schema;
using System.Security.Cryptography;
using System.Text;
using System.Text.RegularExpressions;

namespace Papeleria.LogicaNegocio.Entidades.ValueObjects.Usuario
{
    [ComplexType]
    public record ContraseniaUsuario : IValidable<ContraseniaUsuario>
    {
```

```
public string Valor { get; set; }

public ContraseniaUsuario(string contrasenia)

{

   Valor = contrasenia;

   esValido();

}

public ContraseniaUsuario()

{



}

//public void esValido(string contrasenia) {

//    if (Valor == null)

//    {

//        throw new ContraseniaNuloException("La contrasenia no puede ser nula.");

//    }

//    if (Valor.Length < 6)

//    {

//        throw new ContraseniaNoValidoException("La contrase□a debe contener un largo minimo de 6 caracteres.");

//    }

//    if (!Regex.IsMatch(Valor, @"^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)(?=.*[.,;!])[A-Za-z\d.,;!]+$"))

//    {
```

```
//          throw new ContraseniaNoValidoException("La contrase□a debe contener al menos una letra may□scula, una min□scula, un d□gito y un car□cter de puntuaci□n: punto, punto y coma, coma, signo de admiraci□n de cierre.");
//    }
//}
public void esValido()
{
   if (Valor == null)
   {
      throw new ContraseniaNuloException("La contrasenia no puede ser nula.");
   }
   if (Valor.Length < 6)
   {
      throw new ContraseniaNoValidoException("La contrase□a debe contener un largo minimo de 6 caracteres.");
   }
   if (!Regex.IsMatch(Valor, @"^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)(?=.*[.,;!])[A-Za-z\d.,;!]+$"))
   {
      throw new ContraseniaNoValidoException("La contrase□a debe contener al menos una letra may□scula, una min□scula, un d□gito y un car□cter de puntuaci□n: punto, punto y coma, coma, signo de admiraci□n de cierre.");
   }
}
public string Encriptar(string contrasenia)
{
```

```
        using (SHA256 sha256Hash = SHA256.Create())
        {
            byte[] bytes = sha256Hash.ComputeHash(Encoding.UTF8.GetBytes(contrasenia));

            StringBuilder builder = new StringBuilder();
            for (int i = 0; i < bytes.Length; i++)
            {
                builder.Append(bytes[i].ToString("x2"));
            }
            return builder.ToString();
        }
    }
}
```

****

Archivo: EmailUsuario.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Entidades\ValueObjects\Usuarios\EmailUsuario.cs

****

```
using Empresa.LogicaDeNegocio.Entidades;
using Papeleria.LogicaNegocio.Excepciones.Usuario.UsuarioExcepcions.Email;
```

```
using Papeleria.LogicaNegocio.InterfacesEntidades;
using System.ComponentModel.DataAnnotations.Schema;
using System.Drawing;

namespace Papeleria.LogicaNegocio.Entidades.ValueObjects.Usuario
{
    [ComplexType]
    public record EmailUsuario :IValidable<EmailUsuario>, IEquatable<EmailUsuario>
    {
        public string Direccion { get; init; }
        public EmailUsuario(string direccion) {
            if (direccion == null)
            {
                throw new ArgumentNullException(nameof(direccion), "No puede ser nulo");
            }
            Direccion = direccion;
            esValido();
        }
        public void esValido(EmailUsuario emailUsuario)
        {
            if (emailUsuario == null) {
                throw new EmailNuloException("El email no puede ser nulo.");
            }
```

```
        if (emailUsuario.Direccion.Length < 6) {
            throw new EmailNoValidoException("Email no v□lido. Largo m□nimo 6.");
        }
        if (!emailUsuario.Direccion.Contains("@")) {
            throw new EmailNoValidoException("Email no v□lido. Debe incluir un arroba.");
        }
        if (emailUsuario.Direccion.IndexOf("@") == 0 || emailUsuario.Direccion.IndexOf("@") == emailUsuario.Direccion.Length-1) {
            throw new EmailNoValidoException("Email no v□lido. No puede contener @ en el principio o en el final.");
        }
    }

    public void esValido()
    {
        esValido(this);
    }
  }
}
```

****

Archivo: NombreCompleto.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Entidades\ValueObjects\Usuarios\NombreCompleto.cs

```
**********************************

using Papeleria.LogicaNegocio.Excepciones.Usuario.UsuarioExcepcions.Nombre;

using Papeleria.LogicaNegocio.InterfacesEntidades;

using System.Text.RegularExpressions;

using System.ComponentModel.DataAnnotations.Schema;


namespace Papeleria.LogicaNegocio.Entidades.ValueObjects.Usuario

{

    [ComplexType]

    public record NombreCompleto : IValidable<NombreCompleto>

    {

        public string Nombre { get; init; }

        public string Apellido { get; init; }


        public NombreCompleto(string nombre, string apellido)

        {

            Nombre = FormatearInicialesMayuscula(nombre);

            Apellido = FormatearInicialesMayuscula(apellido);

            esValido();

        }

        //public void esValido(string nombre, string apellido) {

        //    if (nombre == null || apellido == null)

        //    {
```

```
//        throw new NombreNuloException("El nombre o apellido no pueden ser nulos.");
//    }
//    if (nombre.Length <= 2 && !nombre.Any(c => char.IsDigit(c)))
//    {
//        throw new NombreNoValidoException($"{nombre}: no es un nombre valido.");
//    }
//    if (apellido.Length <= 2 && !apellido.Any(c => char.IsDigit(c)))
//    {
//        throw new NombreNoValidoException($"{apellido}: no es un apellido valido.");
//    }
//    if (!Regex.IsMatch(nombre, @"^[a-zA-Z]+([' -]?[a-zA-Z]+)*$"))
//    {
//        throw new NombreNoValidoException($"{nombre}: no es un nombre valido.");
//    }
//    if (!Regex.IsMatch(apellido, @"^[a-zA-Z]+([' -]?[a-zA-Z]+)*$"))
//    {
//        throw new NombreNoValidoException($"{apellido}: no es un apellido valido.");
//    }
//}
public static string FormatearInicialesMayuscula(string texto)
{
    string[] palabras = texto.Split(' ');
    for (int i = 0; i < palabras.Length; i++)
```

```
    {
        if (palabras[i].Length > 0)
        {
            palabras[i] = char.ToUpper(palabras[i][0]) + palabras[i].Substring(1);
        }
    }
    return string.Join(" ", palabras);
}
public void esValido()
{
    if (Nombre == null || Apellido == null) {
        throw new NombreNuloException("El nombre o apellido no pueden ser nulos.");
    }
    if (Nombre.Length <= 2 && !Nombre.Any(c => char.IsDigit(c))){
        throw new NombreNoValidoException($"{Nombre}: no es un nombre valido.");
    }
    if (Apellido.Length <= 2 && !Apellido.Any(c => char.IsDigit(c)))
    {
        throw new NombreNoValidoException($"{Apellido}: no es un apellido valido.");
    }
    if (!Regex.IsMatch(Nombre, @"^[a-zA-Z]+([' -]?[a-zA-Z]+)*$")){
        throw new NombreNoValidoException($"{Nombre}: no es un nombre valido.");
    }
```

```
            if (!Regex.IsMatch(Apellido, @"^[a-zA-Z]+([' -]?[a-zA-Z]+)*$")){
                throw new NombreNoValidoException($"{Apellido}: no es un apellido valido.");
            }
        }
    }

}
```

****

Archivo: CodigoProveedorDuplicadoException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Articulo\ArticulosValueObjects\CodigoProveedor\CodigoProveedorDuplicadoException.cs

****

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

namespace Papeleria.LogicaNegocio.Excepciones.Articulo.ArticulosValueObjects.CodigoProveedor
```

```
{
    public class CodigoProveedorDuplicadoException : Exception
    {
        public CodigoProveedorDuplicadoException()
        {
        }

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

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

        protected CodigoProveedorDuplicadoException(SerializationInfo info, StreamingContext context) : base(info, context)
        {
        }
    }
}
```

****

Archivo: CodigoProveedorNoValidoException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Articulo\ArticulosValueObjects\CodigoProveedor\CodigoProveedorNoValidoException.cs

****

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


namespace Papeleria.LogicaNegocio.Excepciones.Articulo.ArticulosValueObjects.CodigoProveedor
{
    public class CodigoProveedorNoValidoException : Exception
    {
        public CodigoProveedorNoValidoException()
        {
        }


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

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

    protected CodigoProveedorNoValidoException(SerializationInfo info, StreamingContext context) : base(info, context)
    {
    }
  }
}
```

****

Archivo: CodigoProveedorNuloException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Articulo\ArticulosValueObjects\CodigoProveedor\CodigoProveedorNuloException.cs

****

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

```
namespace Papeleria.LogicaNegocio.Excepciones.Articulo.ArticulosValueObjects.CodigoProveedor
{
    public class CodigoProveedorNuloException : Exception
    {
        public CodigoProveedorNuloException()
        {
        }

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

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

        protected CodigoProveedorNuloException(SerializationInfo info, StreamingContext context) : base(info, context)
        {
        }
    }
}
```

****

Archivo: DescripcionArticuloDuplicadoException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Articulo\ArticulosValueObjects\DescripcionArticulo\DescripcionArticuloDuplicadoException.cs

****

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

namespace Papeleria.LogicaNegocio.Excepciones.Articulo.ArticulosValueObjects.DescripcionArticulo
{
    public class DescripcionArticuloDuplicadoException : Exception
    {
        public DescripcionArticuloDuplicadoException()
        {
        }

        public DescripcionArticuloDuplicadoException(string? message) : base(message)
```

```
    {

    }

    public DescripcionArticuloDuplicadoException(string? message, Exception? innerException) : base(message, innerException)

    {

    }

    protected DescripcionArticuloDuplicadoException(SerializationInfo info, StreamingContext context) : base(info, context)

    {

    }

  }

}
```

****

Archivo: DescripcionArticuloNoValidoException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Articulo\ArticulosValueObjects\DescripcionArticulo\DescripcionArticuloNoValidoException.cs

****

```
using System;

using System.Collections.Generic;

using System.Linq;
```

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

namespace Papeleria.LogicaNegocio.Excepciones.Articulo.ArticulosValueObjects.DescripcionArticulo
{
    public class DescripcionArticuloNoValidoException : Exception
    {
        public DescripcionArticuloNoValidoException()
        {
        }

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

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

        protected DescripcionArticuloNoValidoException(SerializationInfo info, StreamingContext context) : base(info, context)
        {
```

```
        }
    }
}
```

****

Archivo: DescripcionArticuloNuloException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Articulo\ArticulosValueObjects\DescripcionArticulo\DescripcionArticuloNuloException.cs

****

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

namespace Papeleria.LogicaNegocio.Excepciones.Articulo.ArticulosValueObjects.DescripcionArticulo
{
    public class DescripcionArticuloNuloException : Exception
    {
        public DescripcionArticuloNuloException()
        {
```

```
    }

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

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

    protected DescripcionArticuloNuloException(SerializationInfo info, StreamingContext context) : base(info, context)
    {
    }
  }
}
```

****

Archivo: LineaDuplicadoException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Articulo\ArticulosValueObjects\Linea\LineaDuplicadoException.cs

****

```
using System;
```

```
using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace Papeleria.LogicaNegocio.Excepciones.Articulo.ArticulosValueObjects.Linea

{

    internal class LineaDuplicadoException

    {

    }

}
```

****

Archivo: LineaNoValidoException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Articulo\ArticulosValueObjects\Linea\LineaNoValidoException.cs

****

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;
```

```
namespace Papeleria.LogicaNegocio.Excepciones.Articulo.ArticulosValueObjects.Linea
{
    internal class LineaNoValidoException
    {
    }
}
```

****

Archivo: LineaNuloException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Articulo\ArticulosValueObjects\Linea\LineaNuloException.cs

****

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

namespace Papeleria.LogicaNegocio.Excepciones.Articulo.ArticulosValueObjects.Linea
{
    public class LineaNuloException : Exception
    {
```

```
        public LineaNuloException()
        {
        }

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

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

        protected LineaNuloException(SerializationInfo info, StreamingContext context) : base(info, context)
        {
        }
    }
}
```

****

Archivo: NombreArticuloDuplicadoException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Articulo\ArticulosValueObjects\NombreArticulo\NombreArticuloDuplicadoException.cs

```
**********
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.Text;
using System.Threading.Tasks;

namespace Papeleria.LogicaNegocio.Excepciones.Articulo.ArticulosValueObjects.NombreArticulo
{
    public class NombreArticuloDuplicadoException : Exception
    {
        public NombreArticuloDuplicadoException()
        {
        }

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

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

protected NombreArticuloDuplicadoException(SerializationInfo info, StreamingContext context) : base(info, context)

{

}

}

}

****

Archivo: NombreArticuloNoValidoException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Articulo\ArticulosValueObjects\NombreArticulo\NombreArticuloNoValidoException.cs

****

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

namespace Papeleria.LogicaNegocio.Excepciones.Articulo.ArticulosValueObjects.NombreArticulo
{
    public class NombreArticuloNoValidoException : Exception
```

```
    {
        public NombreArticuloNoValidoException()
        {
        }

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

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

        protected NombreArticuloNoValidoException(SerializationInfo info, StreamingContext context) : base(info, context)
        {
        }
    }
}
```

****

Archivo: NombreArticuloNuloException.cs

Carpeta:
C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Articulo\ArticulosValueObjects\NombreArticulo\NombreArticuloNuloException.cs

****

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.Text;

using System.Threading.Tasks;


namespace Papeleria.LogicaNegocio.Excepciones.Articulo.ArticulosValueObjects.NombreArticulo

{

    public class NombreArticuloNuloException : Exception

    {

        public NombreArticuloNuloException()

        {

        }


        public NombreArticuloNuloException(string? message) : base(message)

        {

        }
```

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

    protected NombreArticuloNuloException(SerializationInfo info, StreamingContext context) : base(info, context)
    {
    }
  }
}
```

****

Archivo: StockArticuloDuplicadoException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Articulo\ArticulosValueObjects\StockArticulo\StockArticuloDuplicadoException.cs

****

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

```
namespace Papeleria.LogicaNegocio.Excepciones.Articulo.ArticulosValueObjects.StockArticulo
{
    public class StockArticuloDuplicadoException : Exception
    {
        public StockArticuloDuplicadoException()
        {
        }

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

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

        protected StockArticuloDuplicadoException(SerializationInfo info, StreamingContext context) : base(info, context)
        {
        }
    }
}
```

****

Archivo: StockArticuloNoValidoException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Articulo\ArticulosValueObjects\StockArticulo\StockArticuloNoValidoException.cs

****

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.Text;

using System.Threading.Tasks;


namespace Papeleria.LogicaNegocio.Excepciones.Articulo.ArticulosValueObjects.StockArticulo

{

    public class StockArticuloNoValidoException : Exception

    {

        public StockArticuloNoValidoException()

        {

        }


        public StockArticuloNoValidoException(string? message) : base(message)

        {

        }
```

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

        protected StockArticuloNoValidoException(SerializationInfo info, StreamingContext context) : base(info, context)
        {
        }
    }
}
```

****

Archivo: StockArticuloNuloException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Articulo\ArticulosValueObjects\StockArticulo\StockArticuloNuloException.cs

****

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

```
namespace Papeleria.LogicaNegocio.Excepciones.Articulo.ArticulosValueObjects.StockArticulo
{
    public class StockArticuloNuloException : Exception
    {
        public StockArticuloNuloException()
        {
        }

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

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

        protected StockArticuloNuloException(SerializationInfo info, StreamingContext context) : base(info, context)
        {
        }
    }
}
```

****

Archivo: ContraseniaNoValidoException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Usuario\UsuarioExcepcions\Constrasenia\ContraseniaNoValidoException.cs

****

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


namespace Papeleria.LogicaNegocio.Excepciones.Usuario.UsuarioExcepcions.Constrasenia
{
    public class ContraseniaNoValidoException : Exception
    {
        public ContraseniaNoValidoException()
        {
        }


        public ContraseniaNoValidoException(string? message) : base(message)
        {
```

```
        }


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


        protected ContraseniaNoValidoException(SerializationInfo info, StreamingContext context) : base(info, context)
        {
        }
    }
}
```

****

Archivo: ContraseniaNuloException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Usuario\UsuarioExcepcions\Constrasenia\ContraseniaNuloException.cs

****

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.Text;
```

```
using System.Threading.Tasks;

namespace Papeleria.LogicaNegocio.Excepciones.Usuario.UsuarioExcepcions.Constrasenia
{
    public class ContraseniaNuloException : Exception
    {
        public ContraseniaNuloException()
        {
        }

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

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

        protected ContraseniaNuloException(SerializationInfo info, StreamingContext context) : base(info, context)
        {
        }
    }
```

}

****

Archivo: EmailDuplicadoException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Usuario\UsuarioExcepcions\Email\EmailDuplicadoException.cs

****

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.Text;

using System.Threading.Tasks;


namespace Papeleria.LogicaNegocio.Excepciones.Usuario.UsuarioExcepcions.Email

{

    public class EmailDuplicadoException : Exception

    {

        public EmailDuplicadoException()

        {

        }


        public EmailDuplicadoException(string? message) : base(message)
```

```
        {

        }


        public EmailDuplicadoException(string? message, Exception? innerException) : base(message, innerException)

        {

        }


        protected EmailDuplicadoException(SerializationInfo info, StreamingContext context) : base(info, context)

        {

        }

    }

}
```

****

Archivo: EmailNoValidoException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Usuario\UsuarioExcepcions\Email\EmailNoValidoException.cs

****

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;
```

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

namespace Papeleria.LogicaNegocio.Excepciones.Usuario.UsuarioExcepcions.Email
{
    public class EmailNoValidoException : Exception
    {
        public EmailNoValidoException()
        {
        }

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

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

        protected EmailNoValidoException(SerializationInfo info, StreamingContext context) : base(info, context)
        {
        }
```

```
    }
}
```

****

Archivo: EmailNuloException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Usuario\UsuarioExcepcions\Email\EmailNuloException.cs

****

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

namespace Papeleria.LogicaNegocio.Excepciones.Usuario.UsuarioExcepcions.Email
{
    public class EmailNuloException : Exception
    {
        public EmailNuloException()
        {
        }
```

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

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

        protected EmailNuloException(SerializationInfo info, StreamingContext context) : base(info, context)
        {
        }
    }
}
```

****

Archivo: NombreNoValidoException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Usuario\UsuarioExcepcions\Nombre\NombreNoValidoException.cs

****

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

```
using System.Runtime.Serialization;

using System.Text;

using System.Threading.Tasks;


namespace Papeleria.LogicaNegocio.Excepciones.Usuario.UsuarioExcepcions.Nombre

{

    public class NombreNoValidoException : Exception

    {

        public NombreNoValidoException()

        {

        }


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

        {

        }


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

        {

        }


        protected NombreNoValidoException(SerializationInfo info, StreamingContext context) : base(info, context)

        {
```

```
        }
    }
}
```

****

Archivo: NombreNuloException.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria_Deposito\Papeleria.LogicaNegocio\Excepciones\Usuario\UsuarioExcepcions\Nombre\NombreNuloException.cs

****

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

namespace Papeleria.LogicaNegocio.Excepciones.Usuario.UsuarioExcepcions.Nombre
{
    public class NombreNuloException : Exception
    {
        public NombreNuloException()
        {
        }
```

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

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

        protected NombreNuloException(SerializationInfo info, StreamingContext context) : base(info, context)
        {
        }
    }
}
```

****

Archivo: ConsultasController.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria.MVC\Controllers\ConsultasController.cs

****

```
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Papeleria.MVC.Models;
using System.Net.Http.Headers;
using System.Text.Json;

namespace Papeleria.MVC.Controllers
{
    public class ConsultasController : Controller
    {
        private readonly HttpClient _httpClient;
        private readonly string _url = "https://localhost:7148/api/";
        private readonly JsonSerializerOptions _jsonOptions
            = new JsonSerializerOptions { PropertyNameCaseInsensitive = true };
        // GET: MovimientoStockController
        public ConsultasController(HttpClient httpClient)
        {
```

```
        _httpClient = httpClient;
        _httpClient.BaseAddress = new Uri(_url);
        _httpClient.DefaultRequestHeaders.Accept.Clear();
        _httpClient.DefaultRequestHeaders.Accept.Add(new
System.Net.Http.Headers.MediaTypeWithQualityHeaderValue("application/json"));
    }

    // GET: ConsultasController
    public ActionResult Index()
    {
        _httpClient.DefaultRequestHeaders.Authorization = new AuthenticationHeaderValue("Bearer",
HttpContext.Session.GetString("Token"));
        if (_httpClient.DefaultRequestHeaders.Authorization.Parameter == null ||
HttpContext.Session.GetString("Rol") == "Administrador")
        {
            return RedirectToAction("Autorizar", "Login");
        }
        HttpResponseMessage articulosRequest = _httpClient.GetAsync("Articulos").Result;
        HttpResponseMessage tipoMovRequest = _httpClient.GetAsync("TipoMovimientos").Result;
        IEnumerable<ArticuloModel> articulos = null;
        IEnumerable<TipoMovimientoModel> tiposMovimientos = null;
        if (articulosRequest.IsSuccessStatusCode)
        {
            var body = articulosRequest.Content.ReadAsStringAsync().Result;
```

```
            var objetos = JsonSerializer.Deserialize<IEnumerable<Models.ArticuloModel>>(body);
            articulos = objetos;
        }
        if (tipoMovRequest.IsSuccessStatusCode)
        {
            var body = tipoMovRequest.Content.ReadAsStringAsync().Result;
            var objetos = JsonSerializer.Deserialize<IEnumerable<Models.TipoMovimientoModel>>(body);
            tiposMovimientos = objetos;
        }
        ViewBag.articulos = articulos;
        ViewBag.tiposMovimientos = tiposMovimientos;
        return View();
    }

    [HttpPost]
    [ValidateAntiForgeryToken]
    public ActionResult Index(int ArticuloID, string tipoMovimientoNombre, DateTime fechaIni, DateTime fechaFin)
    {
        _httpClient.DefaultRequestHeaders.Authorization = new AuthenticationHeaderValue("Bearer", HttpContext.Session.GetString("Token"));
        if (_httpClient.DefaultRequestHeaders.Authorization.Parameter == null || HttpContext.Session.GetString("Rol") == "Administrador")
        {
```

```
            return RedirectToAction("Autorizar", "Login");
        }
        TempData["ResultadoBuscarMovimientos"] = "";
        DateTime fechaNull = new DateTime(01, 01, 0001);
        try
        {
            if (!string.IsNullOrEmpty(tipoMovimientoNombre) && ArticuloID != 0)
            {
                HttpResponseMessage movimientossRequest = _httpClient.GetAsync($"Movimientos/{ArticuloID}/{tipoMovimientoNombre}").Result;
                IEnumerable<MovimientosModel> movimientos = null;

                var body = movimientossRequest.Content.ReadAsStringAsync().Result;
                var objetos = JsonSerializer.Deserialize<IEnumerable<Models.MovimientosModel>>(body);
                movimientos = objetos;

                if (movimientos.Count() == 0)
                {
                    TempData["ResultadoBuscarMovimientos"] = "No se ha encontrado ninguna coincidencia para ese movimiento.";
                    return RedirectToAction("Index", "Consultas");
                }
                return View("Consulta1", movimientos);
            }
```

```
            if (fechaIni != fechaNull && fechaFin != fechaNull)
            {
                HttpResponseMessage movimientossRequest = _httpClient.GetAsync($"Movimientos/articulos-por-fechas?fechaIni={fechaIni:s}&fechaFin={fechaFin:s}").Result;
                IEnumerable<ArticuloModel> articulos = null;

                var body = movimientossRequest.Content.ReadAsStringAsync().Result;
                var objetos = JsonSerializer.Deserialize<IEnumerable<Models.ArticuloModel>>(body);
                articulos = objetos;

                if (articulos.Count() == 0)
                {
                    TempData["ResultadoBuscarMovimientos"] = "No se ha encontrado ninguna coincidencia para ese movimiento.";
                    return RedirectToAction("Index", "Consultas");
                }
                return View("Consulta2", articulos);
            }
            TempData["ResultadoBuscarMovimientos"] = "No se ha encontrado ninguna coincidencia. Verifique los parametros ingresados.";
            return RedirectToAction("Index", "Consultas");
        }
        catch (Exception e)
        {
```

```
            TempData["ResultadoBuscarMovimientos"] = e.Message;
            return RedirectToAction("Index", "Consultas");
        }
    }
    public ActionResult Consultar()
    {
        _httpClient.DefaultRequestHeaders.Authorization = new AuthenticationHeaderValue("Bearer", HttpContext.Session.GetString("Token"));
        if (_httpClient.DefaultRequestHeaders.Authorization.Parameter == null || HttpContext.Session.GetString("Rol") == "Administrador")
        {
            return RedirectToAction("Autorizar", "Login");
        }
        HttpResponseMessage movimientosRequest = _httpClient.GetAsync($"Movimientos/resumen").Result;

        IEnumerable<MovimientoConsultaModel> respuesta = null;

        var body = movimientosRequest.Content.ReadAsStringAsync().Result;
        var objetos = JsonSerializer.Deserialize<IEnumerable<MovimientoConsultaModel>>(body);
        respuesta = objetos;

        if (respuesta.Count() == 0)
        {
```

```
            TempData["ResultadoBuscarMovimientos"] = "No se ha encontrado ninguna coincidencia para ese movimiento.";
            return RedirectToAction("Index", "Consultas");
        }

        foreach (var movimiento in respuesta)
        {
            movimiento.totalCantidadMovida = movimiento.movimientos.Sum(m => m.cantidadMovida);
        }
        return View("Consulta3", respuesta);
    }

    // GET: ConsultasController/Details/5
    public ActionResult Details(int id)
    {
        _httpClient.DefaultRequestHeaders.Authorization = new AuthenticationHeaderValue("Bearer", HttpContext.Session.GetString("Token"));
        if (_httpClient.DefaultRequestHeaders.Authorization.Parameter == null || HttpContext.Session.GetString("Rol") == "Administrador")
        {
            return RedirectToAction("Autorizar", "Login");
        }
        return View();
    }
```

```
// GET: ConsultasController/Create

public ActionResult Create()

{

    _httpClient.DefaultRequestHeaders.Authorization = new AuthenticationHeaderValue("Bearer", HttpContext.Session.GetString("Token"));

    if (_httpClient.DefaultRequestHeaders.Authorization.Parameter == null || HttpContext.Session.GetString("Rol") == "Administrador")

    {

        return RedirectToAction("Autorizar", "Login");

    }

    return View();

}



// POST: ConsultasController/Create

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Create(IFormCollection collection)

{

    _httpClient.DefaultRequestHeaders.Authorization = new AuthenticationHeaderValue("Bearer", HttpContext.Session.GetString("Token"));

    if (_httpClient.DefaultRequestHeaders.Authorization.Parameter == null || HttpContext.Session.GetString("Rol") == "Administrador")

    {

        return RedirectToAction("Autorizar", "Login");
```

```
        }
        try
        {
            return RedirectToAction(nameof(Index));
        }
        catch
        {
            return View();
        }
    }

    // GET: ConsultasController/Edit/5
    public ActionResult Edit(int id)
    {
        _httpClient.DefaultRequestHeaders.Authorization = new AuthenticationHeaderValue("Bearer", HttpContext.Session.GetString("Token"));
        if (_httpClient.DefaultRequestHeaders.Authorization.Parameter == null || HttpContext.Session.GetString("Rol") == "Administrador")
        {
            return RedirectToAction("Autorizar", "Login");
        }
        return View();
    }
```

```
// POST: ConsultasController/Edit/5
[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult Edit(int id, IFormCollection collection)
{
    _httpClient.DefaultRequestHeaders.Authorization = new AuthenticationHeaderValue("Bearer", HttpContext.Session.GetString("Token"));
    if (_httpClient.DefaultRequestHeaders.Authorization.Parameter == null || HttpContext.Session.GetString("Rol") == "Administrador")
    {
        return RedirectToAction("Autorizar", "Login");
    }
    try
    {
        return RedirectToAction(nameof(Index));
    }
    catch
    {
        return View();
    }
}

// GET: ConsultasController/Delete/5
public ActionResult Delete(int id)
```

```
{
    _httpClient.DefaultRequestHeaders.Authorization = new AuthenticationHeaderValue("Bearer", HttpContext.Session.GetString("Token"));
    if (_httpClient.DefaultRequestHeaders.Authorization.Parameter == null || HttpContext.Session.GetString("Rol") == "Administrador")
    {
        return RedirectToAction("Autorizar", "Login");
    }
    return View();
}

// POST: ConsultasController/Delete/5
[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult Delete(int id, IFormCollection collection)
{
    _httpClient.DefaultRequestHeaders.Authorization = new AuthenticationHeaderValue("Bearer", HttpContext.Session.GetString("Token"));
    if (_httpClient.DefaultRequestHeaders.Authorization.Parameter == null || HttpContext.Session.GetString("Rol") == "Administrador")
    {
        return RedirectToAction("Autorizar", "Login");
    }
    try
    {
```

```
            return RedirectToAction(nameof(Index));
        }
        catch
        {
            return View();
        }
    }
}
```

****

Archivo: HomeController.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria.MVC\Controllers\HomeController.cs

****

```
using Papeleria.MVC.Models;
using Microsoft.AspNetCore.Mvc;
using System.Diagnostics;

namespace Papeleria.MVC.Controllers
{
    public class HomeController : Controller
    {
```

```
    private HttpClient _httpClient;
    private readonly ILogger<HomeController> _logger;

    public HomeController(HttpClient httpClient, ILogger<HomeController> logger)
    {
        _httpClient = httpClient;
        _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 });
```

```
        }
    }
}
```

****

Archivo: LoginController.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria.MVC\Controllers\LoginController.cs

****

```
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Newtonsoft.Json.Linq;
using Papeleria.MVC.Models;
using System.Net.Http.Headers;
using System.Text;
using System.Text.Json;
///xxxxx
namespace Papeleria.MVC.Controllers
{
    public class LoginController : Controller
    {
        private HttpClient _httpClient;
        private readonly string _url = "https://localhost:7148/api/";
```

```
//Configuracion para deserializar el json y evitar errores por mayusculas y minusculas
private JsonSerializerOptions _jsonOptions
    = new JsonSerializerOptions { PropertyNameCaseInsensitive = true };
public LoginController(HttpClient httpClient)
{
    _httpClient = httpClient;
    _httpClient.BaseAddress = new Uri(_url);
}

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

// POST: LoginController/Create
[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult Autorizar(LoginModel loginModel)
{
    try
    {
        //loginModel.Nombre = "";
```

```
            //loginModel.Apellido = "";
            var json = JsonSerializer.Serialize(loginModel);
            var body = new StringContent(json, Encoding.UTF8, "application/json");
            var respuesta = _httpClient.PostAsync("Usuarios/login", body).Result;
            if (respuesta.IsSuccessStatusCode)
            {
                var content = respuesta.Content.ReadAsStringAsync().Result;

                var token = JsonSerializer.Deserialize<LoginToken>(content, _jsonOptions);
                if (token == null)
                {
                    ViewBag.Error = "No se ha podido deserializar el token";
                    return View(loginModel);
                }
                HttpContext.Session.SetString("Token", token.Token);
                HttpContext.Session.SetString("Rol", token.Rol);
                HttpContext.Session.SetInt32("UserId", token.UserId);
                //HttpContext.Session.SetString("Email", token.Email);
                //Agregar el token a las cabeceras de las peticiones, para que el servidor lo pueda validar
                //No olvidar que el token debe ser enviado en todas las peticiones
                _httpClient.DefaultRequestHeaders.Add("Authorization", $"Bearer {token.Token}");
                _httpClient.DefaultRequestHeaders.Authorization = new
AuthenticationHeaderValue("Bearer", token.Token);
                return RedirectToAction("Index", "Home");
```

```
        }
        else
        {
            SetError(respuesta);
            return View(loginModel);
        }

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

public ActionResult CerrarSesion()
{
    if (HttpContext.Session.GetString("Token") == null)
    {
        return RedirectToAction("Login");
    }

    HttpContext.Session.Remove("Token");
```

```
            HttpContext.Session.Remove("Rol");
           // HttpContext.Session.Remove("Email");
            _httpClient.DefaultRequestHeaders.Remove("Authorization");
            return RedirectToAction("Autorizar","Login");
        }
        private void SetError(HttpResponseMessage respuesta)
        {
            var contenidoError = respuesta.Content.ReadAsStringAsync().Result;
            dynamic mensajeJson = JObject.Parse(@"{'Message':'" + contenidoError + "'}");
            ViewBag.Error = $"Hubo un error. {respuesta.ReasonPhrase} " + mensajeJson.Message;
        }
    }
}
```

****

Archivo: MovimientoController.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria.MVC\Controllers\MovimientoController.cs

****

```
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Mvc.Rendering;
using Newtonsoft.Json.Linq;
```

```
using NuGet.Common;
using Papeleria.MVC.Models;
using System.Net;
using System.Net.Http;
using System.Net.Http.Headers;
using System.Text;
using System.Text.Json;

namespace Papeleria.MVC.Controllers
{
    public class MovimientoController : Controller
    {
        private readonly HttpClient _httpClient;
        private readonly string _url = "https://localhost:7148/api/";
        private readonly JsonSerializerOptions _jsonOptions
            = new JsonSerializerOptions { PropertyNameCaseInsensitive = true };
        // GET: MovimientoStockController
        public MovimientoController(HttpClient httpClient, IHttpClientFactory httpClientFactory)
        {
            _httpClient = httpClient;
            _httpClient.BaseAddress = new Uri(_url);
            _httpClient.DefaultRequestHeaders.Accept.Add(new
System.Net.Http.Headers.MediaTypeWithQualityHeaderValue("application/json"));
        }
```

```
public ActionResult Index()
{
    _httpClient.DefaultRequestHeaders.Authorization = new AuthenticationHeaderValue("Bearer", HttpContext.Session.GetString("Token"));
    if (_httpClient.DefaultRequestHeaders.Authorization.Parameter == null || HttpContext.Session.GetString("Rol") == "Administrador")
    {
        return RedirectToAction("Autorizar", "Login");
    }

    try
    {
        HttpResponseMessage response = _httpClient.GetAsync("Movimientos").Result;
        if (response.IsSuccessStatusCode)
        {
            var body = response.Content.ReadAsStringAsync().Result;
            var objetos = JsonSerializer.Deserialize<IEnumerable<Models.MovimientosModel>>(body);
            return View(objetos);
        }
        else
        {
            SetError(response);
            if (response.StatusCode == HttpStatusCode.Unauthorized)
            {
```

```
                return RedirectToAction("Autorizar", "Login");
            }
            else
            {

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

// GET: MovimientoController/Details/5
public ActionResult Details(int id)
{
    _httpClient.DefaultRequestHeaders.Authorization = new AuthenticationHeaderValue("Bearer", HttpContext.Session.GetString("Token"));
    if (_httpClient.DefaultRequestHeaders.Authorization.Parameter == null || HttpContext.Session.GetString("Rol") == "Administrador")
    {
```

```
            return RedirectToAction("Autorizar", "Login");
        }
        try
        {
            HttpResponseMessage response = _httpClient.GetAsync("Movimientos/" + id).Result;
            if (response.IsSuccessStatusCode)
            {
                var body = response.Content.ReadAsStringAsync().Result;
                var objetos = JsonSerializer.Deserialize<Models.MovimientosModel>(body);
                return View(objetos);
            }
            else
            {
                SetError(response);
                if (response.StatusCode == HttpStatusCode.Unauthorized || HttpContext.Session.GetString("Rol") == "Administrador")
                {
                    return RedirectToAction("Autorizar", "Login");
                }
                else
                {
                    return View();
                }
            }
```

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

    // GET: MovimientoController/Create
    public ActionResult Create()
    {
        _httpClient.DefaultRequestHeaders.Authorization = new AuthenticationHeaderValue("Bearer", HttpContext.Session.GetString("Token"));
        if (_httpClient.DefaultRequestHeaders.Authorization.Parameter == null || HttpContext.Session.GetString("Rol") == "Administrador")
        {
            return RedirectToAction("Autorizar", "Login");
        }
        try
        {
            HttpResponseMessage articulosRequest = _httpClient.GetAsync("Articulos").Result;
            HttpResponseMessage tipoMovRequest = _httpClient.GetAsync("TipoMovimientos").Result;
            IEnumerable<ArticuloModel> articulos = null;
            IEnumerable<TipoMovimientoModel> tiposMovimientos = null;
```

```
            if (articulosRequest.IsSuccessStatusCode)
            {
                var body = articulosRequest.Content.ReadAsStringAsync().Result;
                var objetos = JsonSerializer.Deserialize<IEnumerable<Models.ArticuloModel>>(body);
                articulos = objetos;
            }
            if (tipoMovRequest.IsSuccessStatusCode)
            {
                var body = tipoMovRequest.Content.ReadAsStringAsync().Result;
                var objetos = JsonSerializer.Deserialize<IEnumerable<Models.TipoMovimientoModel>>(body);
                tiposMovimientos = objetos;
            }
            else
            {
                ViewBag.Error = "No se encontraron tipos de movimiento";
            }
            ViewBag.articulos = articulos;
            ViewBag.tiposMovimientos = tiposMovimientos;
            ViewBag.usuario = HttpContext.Session.GetInt32("UserId");
            return View();
        }
        catch (Exception ex)
        {
```

```
            ViewBag.Error = ex.Message;
            return View();
        }
    }

    // POST: MovimientoController/Create
    [HttpPost]
    [ValidateAntiForgeryToken]
    public ActionResult Create(MovimientosModel movimiento)
    {
        _httpClient.DefaultRequestHeaders.Authorization = new AuthenticationHeaderValue("Bearer", HttpContext.Session.GetString("Token"));
        if (_httpClient.DefaultRequestHeaders.Authorization.Parameter == null || HttpContext.Session.GetString("Rol") == "Administrador")
        {
            return RedirectToAction("Autorizar", "Login");
        }
        try
        {
            HttpResponseMessage articulosRequest = _httpClient.GetAsync("Articulos").Result;
            HttpResponseMessage tipoMovRequest = _httpClient.GetAsync("TipoMovimientos").Result;
            IEnumerable<ArticuloModel> articulos = null;
            IEnumerable<TipoMovimientoModel> tiposMovimientos = null;
            if (articulosRequest.IsSuccessStatusCode)
```

```
        {
            var body = articulosRequest.Content.ReadAsStringAsync().Result;
            var objetos = JsonSerializer.Deserialize<IEnumerable<Models.ArticuloModel>>(body);
            articulos = objetos;
        }
        if (tipoMovRequest.IsSuccessStatusCode)
        {
            var body = tipoMovRequest.Content.ReadAsStringAsync().Result;
            var objetos = JsonSerializer.Deserialize<IEnumerable<Models.TipoMovimientoModel>>(body);
            tiposMovimientos = objetos;
        }
        ViewBag.articulos = articulos;
        ViewBag.tiposMovimientos = tiposMovimientos;
        var json = JsonSerializer.Serialize(movimiento);
        var bodyJson = new StringContent(json, Encoding.UTF8, "application/json");
        var respuesta = _httpClient.PostAsync("Movimientos", bodyJson).Result;

        if (respuesta.IsSuccessStatusCode)
        {
            var content = respuesta.Content.ReadAsStringAsync().Result;
            return RedirectToAction(nameof(Index));
        }
        else
```

```
        {
            SetError(respuesta);
            if (respuesta.StatusCode == HttpStatusCode.Unauthorized)
            {
                return RedirectToAction("Autorizar", "Login");
            }
            else
            {
                return View();
            }
        }
    }
    catch (Exception ex)
    {
        ViewBag.Error = ex.Message;
        return View();
    }
}
private void SetError(HttpResponseMessage respuesta)
{
    var contenidoError = respuesta.Content.ReadAsStringAsync().Result;
    dynamic mensajeJson = JObject.Parse(@"{'Message':'" + contenidoError + "'}");
    ViewBag.Error = $"Hubo un error. {respuesta.ReasonPhrase} " + mensajeJson.Message;
```

```
        }
    }
}
```

****

Archivo: ArticuloModel.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria.MVC\Models\ArticuloModel.cs

****

```
namespace Papeleria.MVC.Models
{
    public class ArticuloModel
    {

        public int id { get; set; }
        public long codigoProveedor { get; set; }
        public string nombreArticulo { get; set; }
        public string descripcion { get; set; }
        public double precioVP { get; set; }
    }
}
```

****

Archivo: ErrorViewModel.cs

Carpeta:
C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria.MVC\Models\ErrorViewModel.cs

****

```
namespace Papeleria.MVC.Models
{
    public class ErrorViewModel
    {
        public string? RequestId { get; set; }


        public bool ShowRequestId => !string.IsNullOrEmpty(RequestId);
    }
}
```

****

Archivo: LoginModel.cs

Carpeta:
C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria.MVC\Models\LoginModel.cs

****

```
namespace Papeleria.MVC.Models
{
    public class LoginModel
    {
        public int Id { get; set; }
        public int Admin { get; set; }
```

```
        public string Email { get; set; }
        public string Nombre { get; set; } = string.Empty;
        public string Apellido { get; set; } = string.Empty;
        public string Contrasenia { get; set; }
        //public string Email { get; set; }
        //public string Contrasenia { get; set; }
    }
}
```

****

Archivo: LoginToken.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria.MVC\Models\LoginToken.cs

****

```
namespace Papeleria.MVC.Models
{
    public class LoginToken
    {
        public int UserId { get; set; }
        public string Email { get; set; }
        public string Rol { get; set; }
        public string Token { get; set; }
    }
}
```

****

Archivo: MovimientoConsultaModel.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria.MVC\Models\MovimientoConsultaModel.cs

****

```
namespace Papeleria.MVC.Models
{
    public class MovimientoConsultaModel
    {
        public int anio { get; set; }
        public List<TipoMovConsultaModel> movimientos { get; set; }
        public int totalCantidadMovida { get; set; }
    }
}
```

****

Archivo: MovimientosModel.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria.MVC\Models\MovimientosModel.cs

****

```
namespace Papeleria.MVC.Models
{
```

```
    public class MovimientosModel
    {
        public int id { get; set; }
        public DateTime fecHorMovRealizado { get; set; } = DateTime.Now;
        public int articuloID { get; set; }
        public long codigoProveedor { get; set; }
        public string nombreArticulo { get; set; } = string.Empty;
        public string descripcion { get; set; } = string.Empty;
        public double precioVP { get; set; }
        public int tipoMovimientoID { get; set; }
        public string tipoMovimientoNombre { get; set; } = string.Empty;
        public int usuarioID { get; set; }
        public string email { get; set; } = string.Empty;
        public string nombre { get; set; } = string.Empty;
        public string apellido { get; set; } = string.Empty;
        public string contrasenia { get; set; } = string.Empty;
        public int ctdUnidadesXMovimiento { get; set; }
    }
}
```

****

Archivo: Prueba.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria.MVC\Models\Prueba.cs

****

```
namespace Papeleria.MVC.Models
{
    public class Prueba
    {
        DateTime prueba { get; set; }


    }
}
```

****

Archivo: TipoMovConsultaModel.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria.MVC\Models\TipoMovConsultaModel.cs

****

```
namespace Papeleria.MVC.Models
{
    public class TipoMovConsultaModel
    {
        public string tipoMovimiento { get; set; }
        public int cantidadMovida { get; set; }
    }
}
```

****

Archivo: TipoMovimientoModel.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria.MVC\Models\TipoMovimientoModel.cs

****

```
namespace Papeleria.MVC.Models
{
    public class TipoMovimientoModel
    {
        public int id { get; set; }
        public string nombre { get; set; }

    }
}
```

****

Archivo: UsuarioModel.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria.MVC\Models\UsuarioModel.cs

****

```
namespace Papeleria.MVC.Models
{
```

```
    public class UsuarioModel
    {
        public int Id { get; set; }
        public int Admin { get; set; }
        public string Email { get; set; }
        public string Nombre { get; set; } = string.Empty;
        public string Apellido { get; set; } = string.Empty;
        public string Contrasenia { get; set; }
    }
}
```

****

Archivo: LineaPedidoModel.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria.MVC\Models\PedidosModels\LineaPedidoModel.cs

****

```
using System.ComponentModel.DataAnnotations;

namespace Papeleria.MVC.Models.PedidosModels
{
    public class LineaPedidoModel
    {
        [Required(ErrorMessage = "El artículo es obligatorio")]
```

```
        public int ArticuloId { get; set; }


        [Required(ErrorMessage = "La cantidad es obligatoria")]
        [Range(1, int.MaxValue, ErrorMessage = "La cantidad debe ser mayor que cero")]
        public int Cantidad { get; set; }
    }
}
```

****

Archivo: PedidoAltaModel.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria.MVC\Models\PedidosModels\PedidoAltaModel.cs

****

```
using Microsoft.AspNetCore.Mvc.Rendering;
using System.ComponentModel.DataAnnotations;


namespace Papeleria.MVC.Models.PedidosModels
{
    public class PedidoAltaModel
    {
        [Required(ErrorMessage = "El cliente es obligatorio")]
        public int ClienteId { get; set; }
```

```
        public List<LineaPedidoModel> LineasPedido { get; set; }


        [Required(ErrorMessage = "La fecha de entrega es obligatoria")]
        [DataType(DataType.Date)]
        public DateTime FechaEntrega { get; set; }
        public int ArticuloId { get; set; }
        public int Cantidad { get; set; }
        public double iva { get; set; }
        public double PrecioUnitario { get; set; }
        public double Subtotal { get; set; }
    }
}
```

****

Archivo: PedidoListModel.cs

Carpeta: C:\Users\moreno\Documents\GitHub\P3_OBL_2_Papeleria\Papeleria.MVC\Models\PedidosModels\PedidoListModel.cs

****

```
using System.ComponentModel.DataAnnotations;


namespace Papeleria.MVC.Models.PedidosModels
{
    public class PedidoListModel
```

```
    {
        public int Id { get; set; }
        [Required(ErrorMessage = "El cliente es obligatorio")]
        public int ClienteId { get; set; }

        public List<LineaPedidoModel> LineasPedido { get; set; }

        [DataType(DataType.Date)]
        public DateTime FechaPedido { get; set; }

        [Required(ErrorMessage = "La fecha de entrega es obligatoria")]
        [DataType(DataType.Date)]
        public DateTime FechaEntrega { get; set; }
        public double IVA { get; set; }
        public double PrecioFinal {  get; set; }

    }
}
```