PROYECTO ASP.Net parte 4

Inventario

1. Bodega Producto – Modelo Configuración Repositorio

- En SistemaInventario.Modelos creamos una clase llamada BodegaProducto.cs

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

namespace SistemaInventario.Modelos
{
   public class BodegaProducto
   {
      [Key]
      public int Id { get; set; }
      [Required]
      public int BodegaId { get; set; }

      [ForeignKey("BodegaId")]
      public Bodega Bodega { get; set; }
      [Required]
      public int ProductoId { get; set; }

      [ForeignKey("ProductoId")]
      public Producto Producto { get; set; }

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

   }
}
```

- En SistemaInventario.AccesoDatos/Data/ApplicationDbContext.cs adicionamos la clase BodegaProducto.cs

```
public DbSet<BodegaProducto> BodegasProductos { get; set; }
```

- En SistemaInventario.AccesoDatos/Configuracion duplicamos el archivo ProductoConfiguracion.cs lo renombramos BodegaProductoConfiguracion.cs y cambiamos la información relacionada a BodegaProducto

```
namespace SistemaInventario.AccesoDatos.Configuracion
{
  public class BodegaProductoConfiguracion : IEntityTypeConfiguration<BodegaProducto>
  {
    public void Configure(EntityTypeBuilder<BodegaProducto> builder)
    {
      builder.Property(x => x.Id).IsRequired();
      builder.Property(x => x.BodegaId).IsRequired();
      builder.Property(x => x.ProductoId).IsRequired();
      builder.Property(x => x.Cantidad).IsRequired();

      /* Relaciones*/

      builder.HasOne(x => x.Bodega).WithMany()
        .HasForeignKey(x => x.BodegaId)
        .OnDelete(DeleteBehavior.NoAction);

      builder.HasOne(x => x.Producto).WithMany()
        .HasForeignKey(x => x.ProductoId)
        .OnDelete(DeleteBehavior.NoAction);

    }
  }
}
```

- En SistemaInventario.AccesoDatos/Repositorio/IRepositorio duplicamos el archivo IProductoRepositorio.cs lo renombramos IBodegaProductoRepositorio.cs y cambiamos la información relacionada a BodegaProducto

```
namespace SistemaInventario.AccesoDatos.Repositorio.IRepositorio
{
  public interface IBodegaProductoRepositorio : IRepositorio<BodegaProducto>
  {
    void Actualizar(BodegaProducto bodegaProducto);

  }
}
```

- En SistemaInventario.AccesoDatos/Repositorio duplicamos el archivo ProductoRepositorio.cs lo renombramos BodegaProductoRepositorio.cs y cambiamos la información relacionada a BodegaProducto

```
namespace SistemaInventario.AccesoDatos.Repositorio
{
  public class BodegaProductoRepositorio : Repositorio<BodegaProducto>, IBodegaProductoRepositorio
  {

    private readonly ApplicationDbContext _db;

    public BodegaProductoRepositorio(ApplicationDbContext db) : base(db)
    {
```

```
        _db = db;
    }

    public void Actualizar(BodegaProducto bodegaProducto)
    {
        var bodegaProductoBD = _db.BodegasProductos.FirstOrDefault(b => b.Id == bodegaProducto.Id);
        if (bodegaProductoBD != null)
        {

            bodegaProductoBD.Cantidad = bodegaProducto.Cantidad;


            _db.SaveChanges();
        }
    }


  }
}
```

- En SistemaInventario.AccesoDatos/Repositorio/IRepositorio modificamos el archivo IUnidadtrabajo.cs

```
IBodegaProductoRepositorio BodegaProducto { get; }
```

- En SistemaInventario.AccesoDatos/Repositorio modificamos el archivo Unidadtrabajo.cs

```
public IBodegaProductoRepositorio BodegaProducto { get; private set; }
.
.
.
public UnidadTrabajo(ApplicationDbContext db)
{
    _db = db;
    Bodega = new BodegaRepositorio(_db);
    Categoria = new CategoriaRepositorio(_db);
    Marca = new MarcaRepositorio(_db);
    Producto = new ProductoRepositorio(_db);
    UsuarioAplicacion = new UsuarioAplicacionRepositorio(_db);
    BodegaProducto = new BodegaProductoRepositorio(_db);
}
```

- Realizamos la migración del modelo
    - add-migration AgregarBP
    - update-database

- Verificamos en la base de datos que se creó la tabla BodegaProducto

2. Inventario Cabecera – Modelo Configuración Repositorio

- En SistemaInventario.Modelos creamos una clase llamada Inventario.cs

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

namespace SistemaInventario.Modelos
{
   public class Inventario
   {
      [Key]
      public int Id { get; set; }

      [Required]
      public string UsuarioAplicacionId { get; set; }

      [ForeignKey("UsuarioAplicacionId")]
      public UsuarioAplicacion UsuarioAplicacion { get; set; }

      [Required]
      public DateTime FechaInicial { get; set; }

      [Required]
      public DateTime FechaFinal { get; set; }

      [Required(ErrorMessage ="La Bodega es Obligatoria")]
      public int BodegaId { get; set; }

      [ForeignKey("BodegaId")]
      public Bodega Bodega { get; set; }

      [Required]
      public bool Estado { get; set; }
   }
}
```

- En SistemaInventario.AccesoDatos/Data/ApplicationDbContext.cs adicionamos la clase Inventario.cs

```
public DbSet<Inventario> Inventarios { get; set; }
```

- En SistemaInventario.AccesoDatos/Configuracion duplicamos el archivo ProductoConfiguracion.cs lo renombramos Inventario Configuracion.cs y cambiamos la información relacionada a Inventario

```
using Microsoft.EntityFrameworkCore;
using Microsoft.EntityFrameworkCore.Metadata.Builders;
using SistemaInventario.Modelos;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace SistemaInventario.AccesoDatos.Configuracion
{
   public class InventarioConfiguracion : IEntityTypeConfiguration<Inventario>
   {
      public void Configure(EntityTypeBuilder<Inventario> builder)
      {
         builder.Property(x => x.Id).IsRequired();
         builder.Property(x => x.BodegaId).IsRequired();
         builder.Property(x => x.UsuarioAplicacionId).IsRequired();
         builder.Property(x => x.FechaFinal).IsRequired();
         builder.Property(x => x.FechaInicial).IsRequired();
         builder.Property(x => x.Estado).IsRequired();

         /* Relaciones*/

         builder.HasOne(x => x.Bodega).WithMany()
            .HasForeignKey(x => x.BodegaId)
            .OnDelete(DeleteBehavior.NoAction);

         builder.HasOne(x => x.UsuarioAplicacion).WithMany()
           .HasForeignKey(x => x.UsuarioAplicacionId)
           .OnDelete(DeleteBehavior.NoAction);
      }
   }
}
```

- En SistemaInventario.AccesoDatos/Repositorio/IRepositorio duplicamos el archivo IProductoRepositorio.cs lo renombramos IInventarioRepositorio.cs y cambiamos la información relacionada a Inventario

```
using Microsoft.AspNetCore.Mvc.Rendering;
using SistemaInventario.Modelos;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
```

```
namespace SistemaInventario.AccesoDatos.Repositorio.IRepositorio
{
  public interface IInventarioRepositorio : IRepositorio<Inventario>
  {
    void Actualizar(Inventario inventario);

    IEnumerable<SelectListItem> ObtenerTodosDropdownLista(string obj);
  }
}
```

- En SistemaInventario.AccesoDatos/Repositorio duplicamos el archivo ProductoRepositorio.cs lo renombramos InventarioRepositorio.cs y cambiamos la información relacionada a Inventario

```
using Microsoft.AspNetCore.Mvc.Rendering;
using SistemaInventario.AccesoDatos.Data;
using SistemaInventario.AccesoDatos.Repositorio.IRepositorio;
using SistemaInventario.Modelos;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace SistemaInventario.AccesoDatos.Repositorio
{
  public class InventarioRepositorio : Repositorio<Inventario>, IInventarioRepositorio
  {

    private readonly ApplicationDbContext _db;

    public InventarioRepositorio(ApplicationDbContext db) : base(db)
    {
      _db = db;
    }

    public void Actualizar(Inventario inventario)
    {
      var inventarioBD = _db.Inventarios.FirstOrDefault(b => b.Id == inventario.Id);
      if(inventarioBD != null)
      {

        inventarioBD.BodegaId = inventario.BodegaId;
        inventarioBD.FechaFinal = inventario.FechaFinal;
        inventarioBD.Estado = inventario.Estado;


        _db.SaveChanges();
      }
    }

    public IEnumerable<SelectListItem> ObtenerTodosDropdownLista(string obj)
```

```
    {
      if(obj == "Bodega")
      {
        return _db.Bodegas.Where(b => b.Estado == true).Select(b => new SelectListItem
        {
          Text = b.Nombre,
          Value = b.Id.ToString()
        });
      }
      return null;
    }
  }
}
```

- En SistemaInventario.AccesoDatos/Repositorio/IRepositorio modificamos el archivo IUnidadtrabajo.cs

```
IInventarioRepositorio Inventario { get; }
```

- En SistemaInventario.AccesoDatos/Repositorio modificamos el archivo Unidadtrabajo.cs

```
public IInventarioRepositorio Inventario { get; private set; }
.
.
.
public UnidadTrabajo(ApplicationDbContext db)
{
    _db = db;
    Bodega = new BodegaRepositorio(_db);
    Categoria = new CategoriaRepositorio(_db);
    Marca = new MarcaRepositorio(_db);
    Producto = new ProductoRepositorio(_db);
    UsuarioAplicacion = new UsuarioAplicacionRepositorio(_db);
    BodegaProducto = new BodegaProductoRepositorio(_db);
    Inventario = new InventarioRepositorio(_db);
}
```

- Realizamos la migración del modelo
    - add-migration AgregarInventario
    - update-database

- Verificamos en la base de datos que se creó la tabla Inventario

3. Inventario Detalle – Modelo Configuración Repositorio

- En SistemaInventario.Modelos creamos una clase llamada InventarioDetalle.cs

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

namespace SistemaInventario.Modelos
{
   public class InventarioDetalle
   {
      [Key]
      public int Id { get; set; }

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

      [ForeignKey("InventarioId")]
      public Inventario Inventario { get; set; }

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

      [ForeignKey("ProductoId")]
      public Producto Producto { get; set; }

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

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

   }
}
```

- En SistemaInventario.AccesoDatos/Data/ApplicationDbContext.cs adicionamos la clase InventarioDetalle.cs

```
public DbSet<InventarioDetalle> InventarioDetalles { get; set; }
```

- En SistemaInventario.AccesoDatos/Configuracion duplicamos el archivo InventarioConfiguracion.cs lo renombramos InventarioDetalleConfiguracion.cs y cambiamos la información relacionada a InventarioDetalle

```
using Microsoft.EntityFrameworkCore;
```

```
using Microsoft.EntityFrameworkCore.Metadata.Builders;
using SistemaInventario.Modelos;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace SistemaInventario.AccesoDatos.Configuracion
{
   public class InventarioDetalleConfiguracion : IEntityTypeConfiguration<InventarioDetalle>
   {
      public void Configure(EntityTypeBuilder<InventarioDetalle> builder)
      {
         builder.Property(x => x.Id).IsRequired();
         builder.Property(x => x.InventarioId).IsRequired();
         builder.Property(x => x.ProductoId).IsRequired();
         builder.Property(x => x.StockAnterior).IsRequired();
         builder.Property(x => x.Cantidad).IsRequired();


         /* Relaciones*/

         builder.HasOne(x => x.Inventario).WithMany()
            .HasForeignKey(x => x.InventarioId)
            .OnDelete(DeleteBehavior.NoAction);

         builder.HasOne(x => x.Producto).WithMany()
           .HasForeignKey(x => x.ProductoId)
           .OnDelete(DeleteBehavior.NoAction);
      }
   }
}
```

- En SistemaInventario.AccesoDatos/Repositorio/IRepositorio duplicamos el archivo I InventarioRepositorio.cs lo renombramos IInventarioDetalleRepositorio.cs y cambiamos la información relacionada a InventarioDetalle

```
using Microsoft.AspNetCore.Mvc.Rendering;
using SistemaInventario.Modelos;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace SistemaInventario.AccesoDatos.Repositorio.IRepositorio
{
   public interface IInventarioDetalleRepositorio : IRepositorio<InventarioDetalle>
   {
      void Actualizar(InventarioDetalle inventarioDetalle);
   }
```

}

- En SistemaInventario.AccesoDatos/Repositorio duplicamos el archivo InventarioRepositorio.cs lo renombramos InventarioDetalleRepositorio.cs y cambiamos la información relacionada a InventarioDetalle

```
using Microsoft.AspNetCore.Mvc.Rendering;
using SistemaInventario.AccesoDatos.Data;
using SistemaInventario.AccesoDatos.Repositorio.IRepositorio;
using SistemaInventario.Modelos;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace SistemaInventario.AccesoDatos.Repositorio
{
   public class InventarioDetalleRepositorio : Repositorio<InventarioDetalle>, IInventarioDetalleRepositorio
   {

      private readonly ApplicationDbContext _db;

      public InventarioDetalleRepositorio(ApplicationDbContext db) : base(db)
      {
         _db = db;
      }

      public void Actualizar(InventarioDetalle inventarioDetalle)
      {
        var inventarioDetalleBD = _db.InventarioDetalles.FirstOrDefault(b => b.Id == inventarioDetalle.Id);
         if(inventarioDetalleBD != null)
         {

            inventarioDetalleBD.StockAnterior = inventarioDetalle.StockAnterior;
            inventarioDetalleBD.Cantidad = inventarioDetalle.Cantidad;

            _db.SaveChanges();
         }
      }


   }
}
```

- En SistemaInventario.AccesoDatos/Repositorio/IRepositorio modificamos el archivo IUnidadtrabajo.cs

```
IInventarioDetalleRepositorio InventarioDetalle { get; }
```

- En SistemaInventario.AccesoDatos/Repositorio modificamos el archivo Unidadtrabajo.cs

```
public IInventarioDetalleRepositorio InventarioDetalle { get; private set; }
.
.
.
public UnidadTrabajo(ApplicationDbContext db)
{
   _db = db;
   Bodega = new BodegaRepositorio(_db);
   Categoria = new CategoriaRepositorio(_db);
   Marca = new MarcaRepositorio(_db);
   Producto = new ProductoRepositorio(_db);
   UsuarioAplicacion = new UsuarioAplicacionRepositorio(_db);
   BodegaProducto = new BodegaProductoRepositorio(_db);
   Inventario = new InventarioRepositorio(_db);
   InventarioDetalle = new InventarioDetalleRepositorio(_db);
}
```

- Realizamos la migración del modelo
    - add-migration AgregarInventarioDetalle
    - update-database

- Verificamos en la base de datos que se creó la tabla InventarioDetalle

4. Kardex – Modelo Configuración Repositorio

- En SistemaInventario.Modelos creamos una clase llamada KardexInventario.cs

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

namespace SistemaInventario.Modelos
{
   public class KardexInventario
   {
      [Key]
      public int Id { get; set; }

      [Required]
      public int BodegaProductoId { get; set; }
```

```
        [ForeignKey("BodegaProductoId")]
        public BodegaProducto BodegaProducto { get; set; }

        [Required]
        [MaxLength(100)]
        public string Tipo { get; set; }    // ENT- SAL

        [Required]
        public string Detalle { get; set; }

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

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

        [Required]
        public double Costo { get; set; }

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

        public double Total { get; set; }

        [Required]
        public string UsuarioAplicacionId { get; set; }

        [ForeignKey("UsuarioAplicacionId")]
        public UsuarioAplicacion UsuarioAplicacion { get; set; }

        public DateTime FechaRegistro { get; set; }
    }
}
```

- En SistemaInventario.AccesoDatos/Data/ApplicationDbContext.cs adicionamos la clase KardexInventario.cs

```
public DbSet<KardexInventario> KardexInventario { get; set; }
```

- En SistemaInventario.AccesoDatos/Configuracion duplicamos el archivo InventarioConfiguracion.cs lo renombramos KardexInventarioConfiguracion.cs y cambiamos la información relacionada a KardexInventario

```
using Microsoft.EntityFrameworkCore;
using Microsoft.EntityFrameworkCore.Metadata.Builders;
using SistemaInventario.Modelos;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
```

```
using System.Threading.Tasks;

namespace SistemaInventario.AccesoDatos.Configuracion
{
   public class KardexInventarioConfiguracion : IEntityTypeConfiguration<KardexInventario>
   {
      public void Configure(EntityTypeBuilder<KardexInventario> builder)
      {
         builder.Property(x => x.Id).IsRequired();
         builder.Property(x => x.BodegaProductoId).IsRequired();
         builder.Property(x => x.Tipo).IsRequired();
         builder.Property(x => x.Detalle).IsRequired();
         builder.Property(x => x.StockAnterior).IsRequired();
         builder.Property(x => x.Cantidad).IsRequired();
         builder.Property(x => x.Costo).IsRequired();
         builder.Property(x => x.Stock).IsRequired();
         builder.Property(x => x.Total).IsRequired();
         builder.Property(x => x.UsuarioAplicacionId).IsRequired();
         builder.Property(x => x.FechaRegistro).IsRequired();


         /* Relaciones*/

         builder.HasOne(x => x.BodegaProducto).WithMany()
            .HasForeignKey(x => x.BodegaProductoId)
            .OnDelete(DeleteBehavior.NoAction);

         builder.HasOne(x => x.UsuarioAplicacion).WithMany()
           .HasForeignKey(x => x.UsuarioAplicacionId)
           .OnDelete(DeleteBehavior.NoAction);
      }
   }
}
```

- En SistemaInventario.AccesoDatos/Repositorio/IRepositorio duplicamos el archivo IInventarioRepositorio.cs lo renombramos IKardexInventarioRepositorio.cs y cambiamos la información relacionada a KardexInventario

```
using Microsoft.AspNetCore.Mvc.Rendering;
using SistemaInventario.Modelos;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace SistemaInventario.AccesoDatos.Repositorio.IRepositorio
{
   public interface IKardexInventarioRepositorio : IRepositorio<KardexInventario>
   {
      Task RegistrarKardex(int bodegaProductoId, string tipo, string detalle, int stockAnterior, int cantidad, string
usuarioId);
```

```
    }
}
```

- En SistemaInventario.AccesoDatos/Repositorio duplicamos el archivo InventarioRepositorio.cs lo renombramos KardexInventarioRepositorio.cs y cambiamos la información relacionada a KardexInventario

```
using Microsoft.AspNetCore.Mvc.Rendering;
using Microsoft.EntityFrameworkCore;
using SistemaInventario.AccesoDatos.Data;
using SistemaInventario.AccesoDatos.Repositorio.IRepositorio;
using SistemaInventario.Modelos;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Security.Claims;
using System.Text;
using System.Threading.Tasks;

namespace SistemaInventario.AccesoDatos.Repositorio
{
    public class KardexInventarioRepositorio : Repositorio<KardexInventario>, IKardexInventarioRepositorio
    {

        private readonly ApplicationDbContext _db;

        public KardexInventarioRepositorio(ApplicationDbContext db) : base(db)
        {
            _db = db;
        }

        public async Task RegistrarKardex(int bodegaProductoId, string tipo, string detalle, int stockAnterior, int cantidad, string usuarioId)
        {
            var bodegaProducto = await _db.BodegasProductos.Include(b => b.Producto).FirstOrDefaultAsync(b => b.Id == bodegaProductoId);

            if(tipo=="Entrada")
            {
                KardexInventario kardex = new KardexInventario();
                kardex.BodegaProductoId = bodegaProductoId;
                kardex.Tipo = tipo;
                kardex.Detalle = detalle;
                kardex.StockAnterior = stockAnterior;
                kardex.Cantidad = cantidad;
                kardex.Costo = bodegaProducto.Producto.Costo;
                kardex.Stock = stockAnterior + cantidad;
                kardex.Total = kardex.Stock * kardex.Costo;
                kardex.UsuarioAplicacionId = usuarioId;
                kardex.FechaRegistro = DateTime.Now;

                await _db.KardexInventarios.AddAsync(kardex);
```

```
            await _db.SaveChangesAsync();
        }
        if (tipo == "Salida")
        {
            KardexInventario kardex = new KardexInventario();
            kardex.BodegaProductoId = bodegaProductoId;
            kardex.Tipo = tipo;
            kardex.Detalle = detalle;
            kardex.StockAnterior = stockAnterior;
            kardex.Cantidad = cantidad;
            kardex.Costo = bodegaProducto.Producto.Costo;
            kardex.Stock = stockAnterior- cantidad;
            kardex.Total = kardex.Stock * kardex.Costo;
            kardex.UsuarioAplicacionId = usuarioId;
            kardex.FechaRegistro = DateTime.Now;

            await _db.KardexInventarios.AddAsync(kardex);
            await _db.SaveChangesAsync();
        }
    }
  }
}
```

- En SistemaInventario.AccesoDatos/Repositorio/IRepositorio modificamos el archivo IUnidadtrabajo.cs

```
IKardexInventarioRepositorio KardexInventario { get; }
```

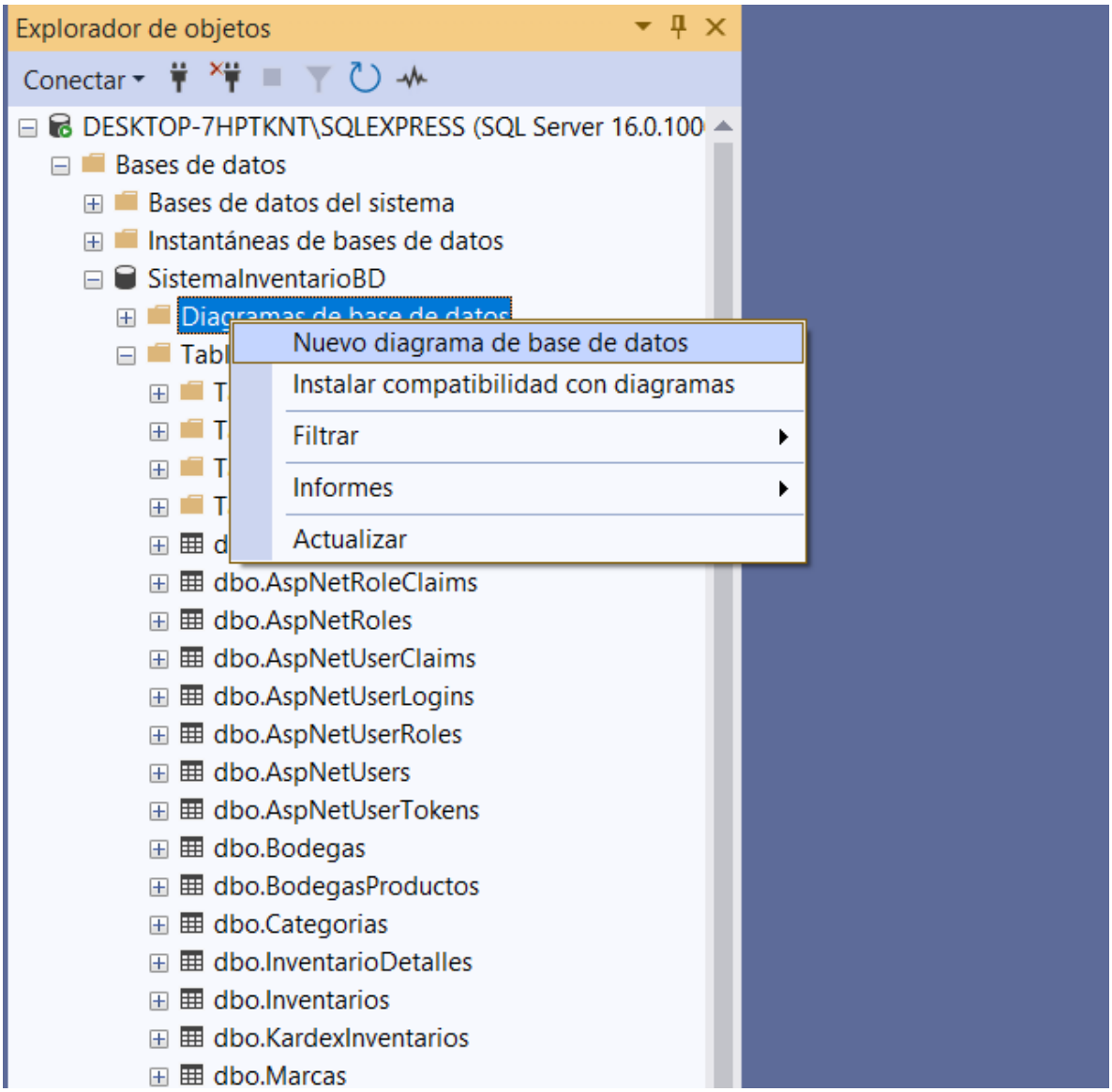
- En SistemaInventario.AccesoDatos/Repositorio modificamos el archivo Unidadtrabajo.cs

```
public IKardexInventarioRepositorio KardexInventario { get; private set; }
.
.
.
public UnidadTrabajo(ApplicationDbContext db)
{
  _db = db;
  Bodega = new BodegaRepositorio(_db);
  Categoria = new CategoriaRepositorio(_db);
  Marca = new MarcaRepositorio(_db);
  Producto = new ProductoRepositorio(_db);
  UsuarioAplicacion = new UsuarioAplicacionRepositorio(_db);
  BodegaProducto = new BodegaProductoRepositorio(_db);
  Inventario = new InventarioRepositorio(_db);
  InventarioDetalle = new InventarioDetalleRepositorio(_db);
  KardexInventario = new KardexInventarioRepositorio(_db);
}
```

- Realizamos la migración del modelo

- add-migration AgregarKardexInventario
- update-database

- Verificamos en la base de datos que se creó la tabla KardexInventario
- Agregar diagrama a la base de datos para verificar relaciones.



5. Inventario Controlador y Vista Index

- En SistemaInventario/Areas/Inventario/Controllers creamos controlador tipo Controlador de MVC: en blanco llamado InventarioController.cs

```
namespace SistemaInventario.Areas.Inventario.Controllers
{
    [Area("Inventario")]
    [Authorize(Roles = DS.Role_Admin + "," + DS.Role_Inventario)]
    public class InventarioController : Controller
    {
        private readonly IUnidadTrabajo _unidadTrabajo;

        public InventarioController(IUnidadTrabajo unidadTrabajo)
        {
            _unidadTrabajo = unidadTrabajo;
        }

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

        #region API
        [HttpGet]
        public async Task<IActionResult> ObtenerTodos()
        {
            var todos = await _unidadTrabajo.BodegaProducto.ObtenerTodos(incluirPropiedades:
"Bodega,Producto");
            return Json(new { data = todos });
        }

        #endregion
    }
}
```

- Desde el método Index de SistemaInventario/Areas/Inventario/Controllers/InventarioController.cs agregamos la vista tipo Vista de Razor

- Para la vista creada Copiamos el código de SistemaInventario/Areas/Admin/Views/Categoria/Index.cshtml.

- En SistemaInventario/wwwroot/js creamos el archivo inventario.js

```
let datatable;

$(document).ready(function () {
    loadDataTable();
});

function loadDataTable() {
    datatable = $('#tblDatos').DataTable({
        "language": {
            "lengthMenu": "Mostrar _MENU_ Registros Por Pagina",
            "zeroRecords": "Ningun Registro",
```

```
            "info": "Mostrar page _PAGE_ de _PAGES_",
            "infoEmpty": "no hay registros",
            "infoFiltered": "(filtered from _MAX_ total registros)",
            "search": "Buscar",
            "paginate": {
                "first": "Primero",
                "last": "Último",
                "next": "Siguiente",
                "previous": "Anterior"
            }
        },
        "ajax": {
            "url": "/Inventario/Inventario/ObtenerTodos"
        },
        "columns": [
            { "data": "bodega.nombre" },
            { "data": "producto.descripcion" },
            {
                "data": "producto.costo", "className": "text-end",
                "render": function (data) {
                    var d = data.toFixed(2).replace(/\d(?=(\d{3})+\.)/g, '$&,');
                    return d;
                }
            },
            { "data": "cantidad", "className": "text-end" },
        ]

    });
}
```

- Modificamos SistemaInventario/Views/Shared/_Layout.cshtml adicionando el enlace de Inventario

```
@if (User.IsInRole(DS.Role_Admin))
{
    <li class="nav-item dropdown">
        <a class="nav-link dropdown-toggle" href="#" data-bs-toggle="dropdown" aria-expanded="false">
            Usuario
        </a>
        <ul class="dropdown-menu">
            <li><a class="dropdown-item" asp-area="Admin" asp-controller="Usuario" asp-action="Index">Usuarios</a></li>
        </ul>
    </li>
}
@if (User.IsInRole(DS.Role_Admin) || User.IsInRole(DS.Role_Inventario))
{
    <li class="nav-item dropdown">
        <a class="nav-link dropdown-toggle" href="#" data-bs-toggle="dropdown" aria-expanded="false">
            Inventario
        </a>
        <ul class="dropdown-menu">
```

```
        <li><a class="dropdown-item" asp-area="Inventario" asp-controller="Inventario" asp-action="Index">Stock Productos</a></li>
      </ul>
    </li>
 }
```

6. Inventario ViewModel

- En SistemaInventario.Modelos/ViewModels creamos una clase llamada InventarioVM.cs

```
using Microsoft.AspNetCore.Mvc.Rendering;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace SistemaInventario.Modelos.ViewModels
{
   public class InventarioVM
   {

      public Inventario Inventario { get; set; }

      public InventarioDetalle InventarioDetalle { get; set; }

      public IEnumerable<InventarioDetalle> InventarioDetalles { get; set; }
      public IEnumerable<SelectListItem> BodegaLista { get; set; }

   }
}
```

7. Nuevo Inventario Metodo y Vista

- En SistemaInventario/Areas/Inventario/Controllers/InventarioController.cs creamos los método InventarioVM y NuevoInventario.

```
public class InventarioController : Controller
{

   private readonly IUnidadTrabajo _unidadTrabajo;

   [BindProperty]
   public InventarioVM inventarioVM { get; set; }

   public InventarioController(IUnidadTrabajo unidadTrabajo)
   {
      _unidadTrabajo = unidadTrabajo;
   }
```

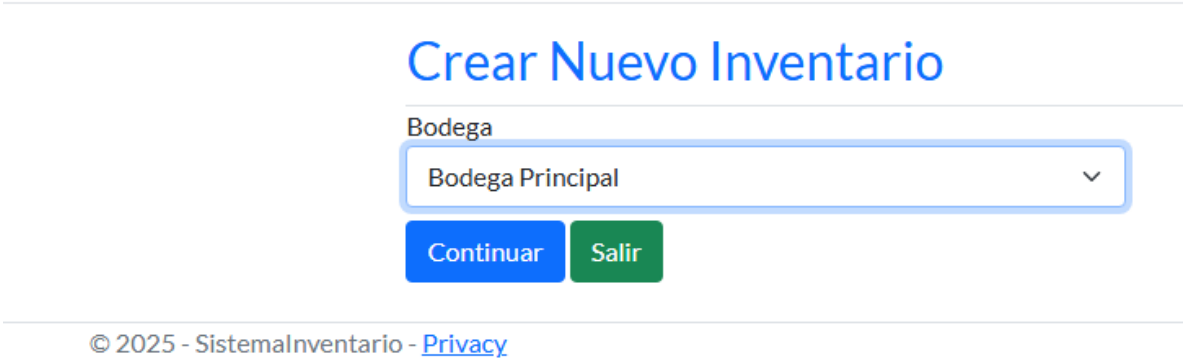
```
    public IActionResult Index()
    {
        return View();
    }

    public IActionResult NuevoInventario()
    {
        inventarioVM = new InventarioVM()
        {
            Inventario = new Modelos.Inventario(),
            BodegaLista = _unidadTrabajo.Inventario.ObtenerTodosDropdownLista("Bodega")
        };

        inventarioVM.Inventario.Estado = false;
        // Obtener el Id del Usuario desde la sesion
        var claimIdentity = (ClaimsIdentity)User.Identity;
        var claim = claimIdentity.FindFirst(ClaimTypes.NameIdentifier);
        inventarioVM.Inventario.UsuarioAplicacionId = claim.Value;
        inventarioVM.Inventario.FechaInicial = DateTime.Now;
        inventarioVM.Inventario.FechaFinal = DateTime.Now;

        return View(inventarioVM);
    }
.
.
.
```

- Desde el método NuevoInventario() creamos una vista de tipo Vista de Razor.

```
@model SistemaInventario.Modelos.ViewModels.InventarioVM
@{
    ViewData["Title"] = "NuevoInventario";
    Layout = "~/Views/Shared/_Layout.cshtml";
    var titulo = "Crear Nuevo Inventario";
}

<form method="post">
    <div style="padding-left:15%; padding-right:15%; padding-bottom: 0.5rem;">
        <div class="row border-0">
            <div asp-validation-summary="ModelOnly" class="text-danger"></div>
        </div>

        <input asp-for="Inventario.Estado" hidden/>
        <input asp-for="Inventario.UsuarioAplicacionId" hidden />
        <input asp-for="Inventario.FechaInicial" hidden />
        <input asp-for="Inventario.FechaFinal" hidden />

        <div class="col-12 border-bottom p-0">
            <h2 class="text-primary">@titulo</h2>
        </div>

        <div class="row mb-2">
```

```
        <div class="col-md-6">
            <label>Bodega</label>
            <select asp-for="Inventario.BodegaId" asp-items="@Model.BodegaLista" class="form-select">
                <option disabled selected>-- Seleccione una Bodega</option>
            </select>
            <span asp-validation-for="Inventario.BodegaId" class="text-danger"></span>
        </div>
    </div>

    <div class="d-grid gap-2 d-md-block">
        <button type="submit" class="btn btn-primary">Continuar</button>
        <a asp-action="Index" class="btn btn-success"> Salir</a>
    </div>

</div>


</form>


@section Scripts{
    <partial name="_ValidationScriptsPartial" />
}
```

- Ejecutamos la aplicación para verificar funcionalidad del botón Crear Nuevo Inventario



8. Nuevo Inventario Metodo Post

- En SistemaInventario/Areas/Inventario/Controllers/InventarioController.cs creamos el método Post NuevoInventario y el método DetalleInventario.

```
public class InventarioController : Controller
{
    private readonly IUnidadTrabajo _unidadTrabajo;
```

```
    [BindProperty]
    public InventarioVM inventarioVM { get; set; }

    public InventarioController(IUnidadTrabajo unidadTrabajo)
    {
        _unidadTrabajo = unidadTrabajo;
    }

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

    public IActionResult NuevoInventario()
    {
        inventarioVM = new InventarioVM()
        {
            Inventario = new Modelos.Inventario(),
            BodegaLista = _unidadTrabajo.Inventario.ObtenerTodosDropdownLista("Bodega")
        };

        inventarioVM.Inventario.Estado = false;
        // Obtener el Id del Usuario desde la sesion
        var claimIdentity = (ClaimsIdentity)User.Identity;
        var claim = claimIdentity.FindFirst(ClaimTypes.NameIdentifier);
        inventarioVM.Inventario.UsuarioAplicacionId = claim.Value;
        inventarioVM.Inventario.FechaInicial = DateTime.Now;
        inventarioVM.Inventario.FechaFinal = DateTime.Now;

        return View(inventarioVM);
    }

    [HttpPost]
    [ValidateAntiForgeryToken]
    public async Task<IActionResult> NuevoInventario(InventarioVM inventarioVM)
    {
        if (ModelState.IsValid)
        {
            inventarioVM.Inventario.FechaInicial = DateTime.Now;
            inventarioVM.Inventario.FechaFinal = DateTime.Now;
            await _unidadTrabajo.Inventario.Agregar(inventarioVM.Inventario);
            await _unidadTrabajo.Guardar();
            return RedirectToAction("DetalleInventario", new { id = inventarioVM.Inventario.Id });
        }
        inventarioVM.BodegaLista = _unidadTrabajo.Inventario.ObtenerTodosDropdownLista("Bodega");
        return View(inventarioVM);
    }

    public async Task<IActionResult> DetalleInventario(int id)
    {
        inventarioVM = new InventarioVM();
        inventarioVM.Inventario = await _unidadTrabajo.Inventario.ObtenerPrimero(i => i.Id == id,
incluirPropiedades: "Bodega");
```

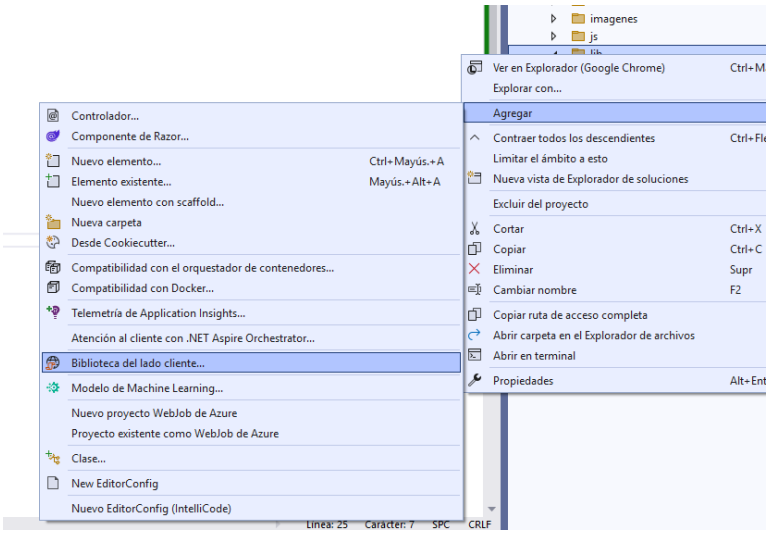
```
        inventarioVM.InventarioDetalles = await _unidadTrabajo.InventarioDetalle.ObtenerTodos(d =>
d.InventarioId == id,
                                                                incluirPropiedades: "Producto,Producto.Marca");
        return View(inventarioVM);
    }
.
.
.
```

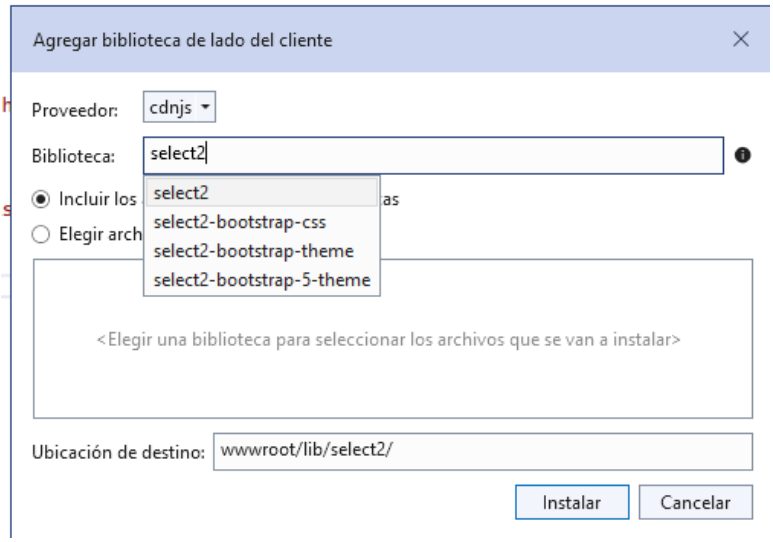
- Desde el método DetalleInventario() creamos una vista de tipo Vista de Razor.

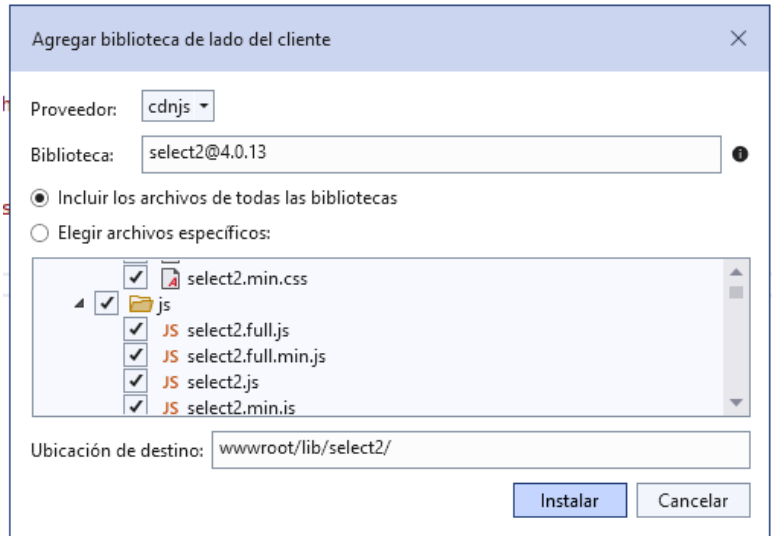
```
@model SistemaInventario.Modelos.ViewModels.InventarioVM
@{
   ViewData["Title"] = "DetalleInventario";
   Layout = "~/Views/Shared/_Layout.cshtml";
}
<div class="container p-2">
   <div class="card-header bg-dark text-light m-lg-0 row container">
      <div class="col-6">
         Agregar Detalle inventario
      </div>
      <div class="col-6 text-end">
         <a asp-action="Index" class="text-white" style="text-decoration:none;"> Salir</a>
      </div>
   </div>


   <form method="post" class="p-2">
      <input asp-for="Inventario.Id" id="inventarioId" name="InventarioId" hidden />

      <div class="form-group mb-4">
         <label>Bodega</label>
         <input asp-for="Inventario.Bodega.Nombre" class="form-control" disabled />
      </div>
   </form>
</div>
```

9. Librería Select2

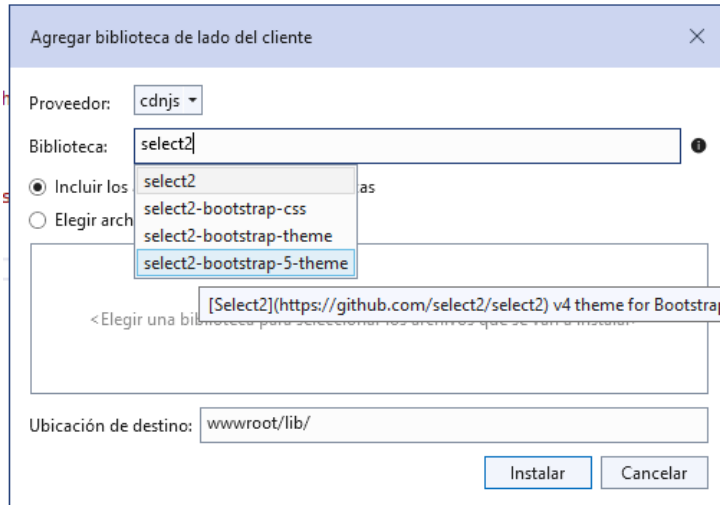
- En SistemaInventario/wwwroot/lib escojemos agregar Biblioteca del lado del cliente

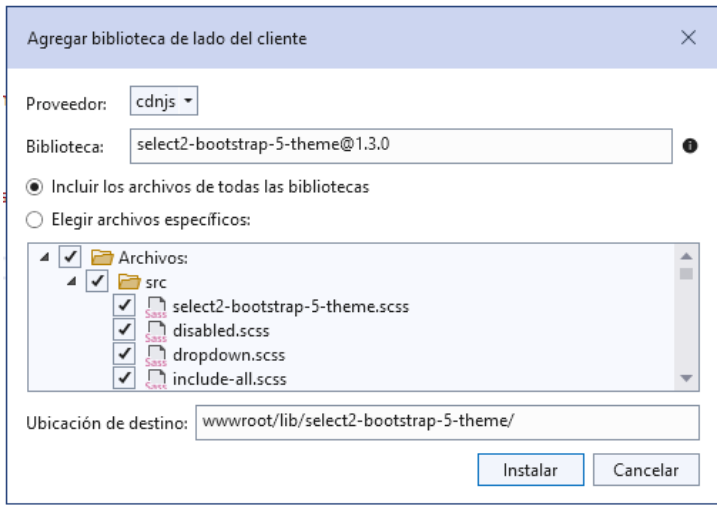

imagenes
js
Ver en Explorador (Google Chrome) Ctrl+M
Explorar con...
Agregar
Contraer todos los descendientes Ctrl+Fl
Limitar el ámbito a esto
Nueva vista de Explorador de soluciones
Excluir del proyecto
Cortar Ctrl+X
Copiar Ctrl+C
Eliminar Supr
Cambiar nombre F2
Copiar ruta de acceso completa
Abrir carpeta en el Explorador de archivos
Abrir en terminal
Propiedades Alt+Ent
Controlador...
Componente de Razor...
Nuevo elemento... Ctrl+Mayús.+A
Elemento existente... Mayús.+Alt+A
Nuevo elemento con scaffold...
Nueva carpeta
Desde Cookiecutter...
Compatibilidad con el orquestador de contenedores...
Compatibilidad con Docker...
Telemetría de Application Insights...
Atención al cliente con .NET Aspire Orchestrator...
Biblioteca del lado cliente...
Modelo de Machine Learning...
Nuevo proyecto WebJob de Azure
Proyecto existente como WebJob de Azure
Clase...
New EditorConfig
Nuevo EditorConfig (IntelliCode)
Línea: 25 Carácter: 7 SPC CRLF


Agregar biblioteca de lado del cliente
Proveedor: cdnjs
Biblioteca: select2
select2
select2-bootstrap-css
select2-bootstrap-theme
select2-bootstrap-5-theme
Incluir los
as
Elegir arch
<Elegir una biblioteca para seleccionar los archivos que se van a instalar>
Ubicación de destino: wwwroot/lib/select2/
Instalar
Cancelar


Agregar biblioteca de lado del cliente
Proveedor: cdnjs
Biblioteca: select2@4.0.13
Incluir los archivos de todas las bibliotecas
Elegir archivos específicos:
select2.min.css
js
select2.full.js
select2.full.min.js
select2.js
select2.min.js
Ubicación de destino: wwwroot/lib/select2/
Instalar
Cancelar

- De la misma manera agregamos otra librería select2





- En SistemaInventario/Views/Shared/_Layout.cshtml adicionamos las librerías select2

```
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>@ViewData["Title"]- SistemaInventario</title>
    <link rel="shortcut icon" href="~/imagenes//icono.ico" />
    <link rel="stylesheet" href="~/lib/bootstrap/dist/css/bootstrap.min.css" />
    <link rel="stylesheet" href="~/css/site.css" asp-append-version="true" />
    <link rel="stylesheet" href="~/SistemaInventario.styles.css" asp-append-version="true" />
    <link rel="stylesheet" href="https://cdn.datatables.net/1.13.3/css/jquery.dataTables.min.css" />
    <link rel="stylesheet" href="https://cdn.datatables.net/1.13.3/css/jquery.dataTables.min.css" />
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/toastr.min.css" />
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.10.3/font/bootstrap-
icons.css">
    <link rel="preconnect" href="https://fonts.googleapis.com">
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
```

```
    <link
href="https://fonts.googleapis.com/css2?family=Lato:ital,wght@0,300;0,400;0,700;0,900;1,400&display=swa
p" rel="stylesheet">
    <link href="~/lib/select2-bootstrap-5-theme/select2-bootstrap-5-theme.min.css" rel="stylesheet" />
</head>
.
.
.
</footer>
<script src="~/lib/jquery/dist/jquery.min.js"></script>
<script src="~/lib/bootstrap/dist/js/bootstrap.bundle.min.js"></script>
<script src="~/js/site.js" asp-append-version="true"></script>
<script src="https://cdn.datatables.net/1.13.3/js/jquery.dataTables.min.js"></script>
<script src="https://unpkg.com/sweetalert/dist/sweetalert.min.js"></script>
<script                                                                    type="text/javascript"
src="https://cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/toastr.min.js"></script>
<script src="~/lib/select2/js/select2.min.js"></script>
```

- Dentro SistemaInventario/Areas/Inventario/Controllers/InventarioController.cs, en la región API creamos el método BuscarProducto()

```
public async Task<IActionResult> BuscarProducto(string term)
{
    if(!string.IsNullOrEmpty(term))
    {
        var listaProductos = await _unidadTrabajo.Producto.ObtenerTodos(p => p.Estado == true);
        var data = listaProductos.Where(x => x.NumeroSerie.Contains(term, StringComparison.OrdinalIgnoreCase)
||
                                    x.Descripcion.Contains(term, StringComparison.OrdinalIgnoreCase)).ToList();
        return Ok(data);
    }
    return Ok();
}
```

- Modificamos la vista SistemaInventario/Areas/Inventario/Views/Inventario/DetalleInventario.cshtml

```
@model SistemaInventario.Modelos.ViewModels.InventarioVM
@{
    ViewData["Title"] = "DetalleInventario";
    Layout = "~/Views/Shared/_Layout.cshtml";
}
<div class="container p-2">
    <div class="card-header bg-dark text-light m-lg-0 row container">
        <div class="col-6">
            Agregar Detalle inventario
        </div>
        <div class="col-6 text-end">
            <a asp-action="Index" class="text-white" style="text-decoration:none;"> Salir</a>
        </div>
    </div>
```

```
    <form method="post" class="p-2">
        <input asp-for="Inventario.Id" id="inventarioId" name="InventarioId" hidden />

        <div class="form-group mb-4">
            <label>Bodega</label>
            <input asp-for="Inventario.Bodega.Nombre" class="form-control" disabled />
        </div>

        <div class="row mb-2">
            <div class="form-group col-md-3">
                <label class="text-primary">Agregar Detalle</label>
            </div>
        </div>

        <div class="row mb-2">
            <div class="form-group col-md-6 mb-2">
                <select class="form-select" id="productoId" name="productoId">
                </select>
            </div>
            <div class="form-group col-md-2 mb-2">
                <input class="form-control text-end" placeholder="Cantidad" type="number" min="1" id="cantidadId"
name="cantidadId" />
            </div>
            <div class="form-group col-md-2 mb-2">
                <button type="submit" class="btn btn-primary" onfocus="false" id="btnAgregar">
                    Agregar
                </button>
            </div>
        </div>
    </form>
</div>

@section Scripts {

    <script>
        // Select 2
        $("#productoId").select2({
            placeholder: "Seleccionar Producto",
            allowClear: true,
            theme: "bootstrap-5",
            ajax: {
                url: "/inventario/inventario/BuscarProducto",
                contentType: "application/json; charset=utf-8",
                data: function (params) {
                    var query =
                    {
                        term: params.term
                    };
                    return query;
                },
                processResults: function (result) {
```

```
            return {
                results: $.map(result, function (item) {
                    return {
                        id: item.id,
                        text: item.numeroSerie + ' ' + item.descripcion
                    };
                })
            };
        }
    }
});

</script>
}
```

- Ejecutamos la aplicación para verificar funcionalidad

10. Detalle Inventario método Post

- Dentro SistemaInventario/Areas/Inventario/Controllers/InventarioController.cs adicionamos el método Post de DetalleInventario

```
public async Task<IActionResult> DetalleInventario(int id)
{
    inventarioVM = new InventarioVM();
    inventarioVM.Inventario = await _unidadTrabajo.Inventario.ObtenerPrimero(i => i.Id == id, incluirPropiedades: "Bodega");
    inventarioVM.InventarioDetalles = await _unidadTrabajo.InventarioDetalle.ObtenerTodos(d => d.InventarioId == id,
                                                                                          incluirPropiedades: "Producto,Producto.Marca");
    return View(inventarioVM);
}

[HttpPost]
[ValidateAntiForgeryToken]
public async Task<IActionResult> DetalleInventario(int InventarioId, int productoId, int cantidadId)
{
    inventarioVM = new InventarioVM();
    inventarioVM.Inventario = await _unidadTrabajo.Inventario.ObtenerPrimero(i => i.Id == InventarioId);
    var bodegaProducto = await _unidadTrabajo.BodegaProducto.ObtenerPrimero(b => b.ProductoId == productoId &&
                                                                            b.BodegaId == inventarioVM.Inventario.BodegaId);
    var detalle = await _unidadTrabajo.InventarioDetalle.ObtenerPrimero(d => d.InventarioId == InventarioId &&
                                                                        d.ProductoId == productoId);
    if (detalle == null)
    {
        inventarioVM.InventarioDetalle = new InventarioDetalle();
        inventarioVM.InventarioDetalle.ProductoId = productoId;
        inventarioVM.InventarioDetalle.InventarioId = InventarioId;
        if (bodegaProducto != null)
        {
```

```
            inventarioVM.InventarioDetalle.StockAnterior = bodegaProducto.Cantidad;
        }
        else
        {
            inventarioVM.InventarioDetalle.StockAnterior = 0;
        }
        inventarioVM.InventarioDetalle.Cantidad = cantidadId;
        await _unidadTrabajo.InventarioDetalle.Agregar(inventarioVM.InventarioDetalle);
        await _unidadTrabajo.Guardar();
    }
    else
    {
        detalle.Cantidad += cantidadId;
        await _unidadTrabajo.Guardar();
    }
    return RedirectToAction("DetalleInventario", new { id = InventarioId });
}
```

- Modificamos la vista SistemaInventario/Areas/Inventario/Views/Inventario/DetalleInventario.cshtml agregamos un table después del form

```
<table class="table table-striped border">
    <thead class="thead-dark">
        <tr class="table-secondary thead-dark">
            <th>Producto</th>
            <th>Marca</th>
            <th>Costo</th>
            <th>Stock</th>
            <th>Cantidad</th>
            <th></th>
        </tr>
    </thead>
    <tbody>
        @foreach (var item in Model.InventarioDetalles)
        {
            <tr>
                <td>
                    @Html.DisplayFor(d => item.Producto.Descripcion)
                </td>
                <td>
                    @Html.DisplayFor(d => item.Producto.Marca.Nombre)
                </td>
                <td>
                    @Html.DisplayFor(d => item.Producto.Costo)
                </td>
                <td>
                    @Html.DisplayFor(d => item.StockAnterior)
                </td>
                <td>
                    @Html.DisplayFor(d => item.Cantidad)
                </td>
```

```
            <td>

            </td>
          </tr>
        }

    </tbody>
</table>
```

- En SistemaInventario/Areas/Inventario/Views/Inventario/DetalleInventario.cshtml agregamos un nuevo script

```
<script>
    $("#btnAgregar").click(function () {
        let cantidad = document.getElementById("cantidadId").value;
        let producto = document.getElementById("productoId").value;

        if(cantidad.toString()=='' || cantidad<1)
        {
            swal("Error", "Ingrese una Cantidad Correcta","error");
            return false;
        }
        if (producto.toString() == '' ) {
            swal("Error", "Seleccione un Producto", "error");
            return false;
        }
    });
</script>
```

- Ejecutamos la aplicación para verificar funcionalidad

11. Botones Mas y Menos y Métodos de Accion

- En SistemaInventario/Areas/Inventario/Controllers/InventarioController.cs agregamos métodos mas y menos y lo pegamos antes de la región API

```
public async Task<IActionResult> Mas(int id)  // recibe el id del detalle
{
    inventarioVM = new InventarioVM();
    var detalle = await _unidadTrabajo.InventarioDetalle.Obtener(id);
    inventarioVM.Inventario = await _unidadTrabajo.Inventario.Obtener(detalle.InventarioId);

    detalle.Cantidad += 1;
    await _unidadTrabajo.Guardar();
    return RedirectToAction("DetalleInventario", new {id = inventarioVM.Inventario.Id});
}

public async Task<IActionResult> Menos(int id)  // recibe el id del detalle
{
    inventarioVM = new InventarioVM();
    var detalle = await _unidadTrabajo.InventarioDetalle.Obtener(id);
```

```
    inventarioVM.Inventario = await _unidadTrabajo.Inventario.Obtener(detalle.InventarioId);
    if(detalle.Cantidad==1)
    {
        _unidadTrabajo.InventarioDetalle.Remover(detalle);
        await _unidadTrabajo.Guardar();
    }
    else
    {
        detalle.Cantidad-= 1;
        await _unidadTrabajo.Guardar();
    }

    return RedirectToAction("DetalleInventario", new { id = inventarioVM.Inventario.Id });
}
```

- En SistemaInventario/Areas/Inventario/Views/Inventario/DetalleInventario.cshtml agregamos botones +/-

```
@foreach (var item in Model.InventarioDetalles)
{
    <tr>
        <td>
            @Html.DisplayFor(d => item.Producto.Descripcion)
        </td>
        <td>
            @Html.DisplayFor(d => item.Producto.Marca.Nombre)
        </td>
        <td>
            @Html.DisplayFor(d => item.Producto.Costo)
        </td>
        <td>
            @Html.DisplayFor(d => item.StockAnterior)
        </td>
        <td>
            @Html.DisplayFor(d => item.Cantidad)
        </td>
        <td>
            <a class="btn btn-primary text-white" asp-action="Mas" asp-route-id="@item.Id">
                <i class="bi bi-file-plus-fill"></i>
            </a>
            <a class="btn btn-primary text-white" asp-action="Menos" asp-route-id="@item.Id">
                <i class="bi bi-file-minus-fill"></i>
            </a>
        </td>
    </tr>
}
.
.
.
</table>

 @if(Model.InventarioDetalles.Count()>0)
```

```
{
   <div class="align-content-lg-center">
      <div class="col-2">
         <a asp-action="GenerarStock" class="btn btn-warning form-control" asp-route-id="@Model.Inventario.Id">
            Generar Stock
         </a>
      </div>
   </div>
}
```

- Ejecutamos la aplicación para verificar funcionalidad de los botons +/-

12. Método Generar Stock

- En SistemaInventario/Areas/Inventario/Controllers/InventarioController.cs agregamos el método GenerarStock y lo pegamos antes de la región API

```
public async Task<IActionResult> GenerarStock(int id)
{
   var inventario = await _unidadTrabajo.Inventario.Obtener(id);
   var detalleLista = await _unidadTrabajo.InventarioDetalle.ObtenerTodos(d =>d.InventarioId == id);
   // Obtener el Id del Usuario desde la sesion
   var claimIdentity = (ClaimsIdentity)User.Identity;
   var claim = claimIdentity.FindFirst(ClaimTypes.NameIdentifier);

   foreach (var item in detalleLista)
   {
      var bodegaProducto = new BodegaProducto();
      bodegaProducto = await _unidadTrabajo.BodegaProducto.ObtenerPrimero(b => b.ProductoId == item.ProductoId &&
                                                b.BodegaId == inventario.BodegaId);
      if(bodegaProducto != null) //  El registro de Stock existe, hay que actualizar las cantidades
      {
         await _unidadTrabajo.KardexInventario.RegistrarKardex(bodegaProducto.Id, "Entrada", "Registro de Inventario",
                                          bodegaProducto.Cantidad, item.Cantidad, claim.Value);
         bodegaProducto.Cantidad += item.Cantidad;
         await _unidadTrabajo.Guardar();

      }
      else  // Registro de Stock no existe, hay que crearlo
      {
         bodegaProducto = new BodegaProducto();
         bodegaProducto.BodegaId = inventario.BodegaId;
         bodegaProducto.ProductoId = item.ProductoId;
         bodegaProducto.Cantidad = item.Cantidad;
         await _unidadTrabajo.BodegaProducto.Agregar(bodegaProducto);
         await _unidadTrabajo.Guardar();
         await _unidadTrabajo.KardexInventario.RegistrarKardex(bodegaProducto.Id, "Entrada", "Inventario Inicial",
```

```
                                        0, item.Cantidad, claim.Value);
        }

    }
    // Actualizar la Cabecera de Inventario
    inventario.Estado = true;
    inventario.FechaFinal = DateTime.Now;
    await _unidadTrabajo.Guardar();
    TempData[DS.Exitosa] = "Stock Generado con Exito";
    return RedirectToAction("Index");

}
```

13. Vista Kardex por Producto

- En SistemaInventario/Areas/Inventario/Controllers/InventarioController.cs agregamos el método KardexProducto antes de la región API

```
public IActionResult KardexProducto()
{
    return View();
}
```

- Agregamos la vista de tipo Vista de Razor del metodo KardexProducto

```
@{
    ViewData["Title"] = "KardexProducto";
    Layout = "~/Views/Shared/_Layout.cshtml";
    var titulo = "Kardex por Producto";
}

<form method="post">
    <div style="padding-left:15%; padding-right:15%; padding-bottom:0.4rem;">
        <div class="col-12 border-bottom p-0">
            <h2 class="text-primary">@titulo </h2>
        </div>

        <div class="row mb-2">
            <div class="form-group col-md-3" style="padding-top:14px;">
                <label>Fecha Inicio</label>
                <input type="date" class="form-control" id="fechaInicioId" name="fechaInicioId">
            </div>
            <div class="form-group col-md-3" style="padding-top:14px;">
                <label>Fecha Final</label>
                <input type="date" class="form-control" id="fechaFinalId" name="fechaFinalId" autocomplete="off">
            </div>
        </div>
        <div class="row mb-2">
            <div class="form-group col-md-6 mb-2">
                <select class="form-select" id="productoId" name="productoId">
                </select>
```

```
        </div>
    </div>

    <div class="d-grid gap-2 d-md-block">
        <button type="submit" class="btn btn-success" onfocus="false" id="btnConsultar"> Consultar</button>
        <a asp-action="Index" class="btn btn-primary">Regresar</a>
    </div>
  </div>
</form>


@section Scripts{

  <script>
    // Select 2
    $("#productoId").select2({
        placeholder: "Seleccionar Producto",
        allowClear: true,
        theme: "bootstrap-5",
        ajax: {
            url: "/inventario/inventario/BuscarProducto",
            contentType: "application/json; charset=utf-8",
            data: function (params) {
                var query =
                {
                    term: params.term
                };
                return query;
            },
            processResults: function (result) {
                return {
                    results: $.map(result, function (item) {
                        return {
                            id: item.id,
                            text: item.numeroSerie + ' ' + item.descripcion
                        };
                    })
                };
            }
        }
    });

  </script>

  <script>
    $("#btnConsultar").click(function () {
        let fechaInicialId = document.getElementById("fechaInicioId").value;
        let fechaFinalId = document.getElementById("fechaFinalId").value;
        let productoId = document.getElementById("productoId").value;

        if (fechaInicialId.toString() == '') {
            swal("Error", "Ingrese una Fecha de Inicio", "error");
            return false;
```

```
        }
        if (fechaFinalId.toString() == '') {
            swal("Error", "Ingrese una Fecha Final", "error");
            return false;
        }

        if (productoId.toString() == '') {
            swal("Error", "Ingrese un Producto", "error");
            return false;
        }
    });
  </script>
}
```

- Ejecutamos la aplicación para verificar funcionalidad

14. Kardex por Producto Método Post

- En SistemaInventario.Modelos/ViewModels creamos una clase llamada KardexInventarioVM.cs

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

namespace SistemaInventario.Modelos.ViewModels
{
  public class KardexInventarioVM
  {
    public Producto Producto { get; set; }

    public IEnumerable<KardexInventario> KardexInventarioLista { get; set; }

    public DateTime FechaInicio { get; set; }

    public DateTime FechaFinal { get; set; }
  }
}
```

- En SistemaInventario/Areas/Inventario/Controllers/InventarioController.cs agregamos el método Post de KardexProduct

```
public IActionResult KardexProducto()
{
  return View();
}

[HttpPost]
public IActionResult KardexProducto(string fechaInicioId, string fechaFinalId, int productoId)
```

```
{
   return RedirectToAction("KardexProductoResultado", new { fechaInicioId, fechaFinalId, productoId });
}
```

- En SistemaInventario/Areas/Inventario/Controllers/InventarioController.cs agregamos el método KardexProductResultado

```
public IActionResult KardexProducto(string fechaInicioId, string fechaFinalId, int productoId)
{
   return RedirectToAction("KardexProductoResultado", new { fechaInicioId, fechaFinalId, productoId });
}

public async Task<IActionResult> KardexProductoResultado(string fechaInicioId, string fechaFinalId, int productoId)
{
   KardexInventarioVM kardexInventarioVM = new KardexInventarioVM();
   kardexInventarioVM.Producto = new Producto();
   kardexInventarioVM.Producto = await _unidadTrabajo.Producto.Obtener(productoId);

   kardexInventarioVM.FechaInicio = DateTime.Parse(fechaInicioId); //  00:00:00
   kardexInventarioVM.FechaFinal  = DateTime.Parse(fechaFinalId).AddHours(23).AddMinutes(59);

   kardexInventarioVM.KardexInventarioLista = await _unidadTrabajo.KardexInventario.ObtenerTodos(
                                        k => k.BodegaProducto.ProductoId == productoId &&
                                           (k.FechaRegistro >= kardexInventarioVM.FechaInicio &&
                                            k.FechaRegistro <= kardexInventarioVM.FechaFinal),
                           incluirPropiedades:
"BodegaProducto,BodegaProducto.Producto,BodegaProducto.Bodega",
                           orderBy: o =>o.OrderBy( o=> o.FechaRegistro)
      );

   return View(kardexInventarioVM);
}
```

15. Vista Kardex por Producto Resultado

- En SistemaInventario/Areas/Inventario/Controllers/InventarioController.cs agregamos la vista de Tipo de Razor del método KardexProductResultado

```
@model SistemaInventario.Modelos.ViewModels.KardexInventarioVM
@using SistemaInventario.Utilidades

@{
   ViewData["Title"] = "KardexProductoResultado";
   Layout = "~/Views/Shared/_Layout.cshtml";
   var titulo = "Kardex Por Producto";
}

<div class="container">

   <div class="col-12 border-bottom p-0">
```

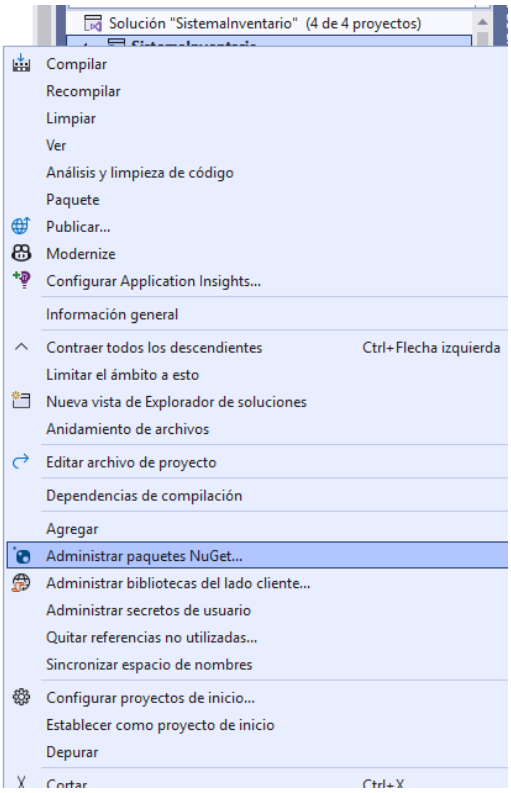
```
    <h2 class="text-primary">@titulo</h2>
  </div>

  <div class="row">
    <div class="col-10">
      <div class="row mb-2">
        <div class="form-group col-md-8 mt-2">
          <label><strong class="m-2">Desde:</strong>@Model.FechaInicio.ToShortDateString() <strong
class="m-2">Hasta: </strong> @Model.FechaFinal.ToShortDateString() </label>
        </div>
      </div>
      <div class="row mb-2">
        <div class="form-group col-md-8 mt-2">
          <label><strong class="m-2">Producto:</strong> @Model.Producto.Descripcion</label>
        </div>
      </div>
    </div>
    <div class="col-2">
      <img src="@DS.ImagenRuta@Model.Producto.ImagenUrl" style="border-radius5px; width:100px;
height:100px;" />
    </div>
  </div>
  <div class="row mb-2">
    <div class="form-group col-md-6 mt-2">
      <a asp-action="KardexProducto" class="btn btn-primary">Nueva Consulta</a>
      <a asp-action="ImprimirKardex"
         asp-route-fechaInicio="@Model.FechaInicio" asp-route-fechaFinal="@Model.FechaFinal"
         asp-route-productoId="@Model.Producto.Id" class="btn btn-primary">
        <i class="bi bi-printer"></i> Imprimir
       </a>
      <a asp-action="Index" class="btn btn-primary">Salir</a>
    </div>
  </div>

  <table class="table table-responsive table-hover table-bordered">
    @foreach (var bodega in Model.KardexInventarioLista.GroupBy(x =>x.BodegaProducto.Bodega.Nombre))
    {
      <thead class="table-secondary">
        <tr>
          <th colspan="6">@bodega.Key</th>
          <th colspan="3" class="text-center">Saldo</th>
        </tr>
        <tr>
          <th class="text-center">Fecha</th>
          <th class="text-center">Tipo</th>
          <th class="text-center">Detalle</th>
          <th class="text-center">Stock Anterior</th>
          <th class="text-center">Entrada</th>
          <th class="text-center">Salida</th>
          <th class="text-center">Stock</th>
          <th class="text-center">Costo</th>
          <th class="text-center">Total</th>
        </tr>
```

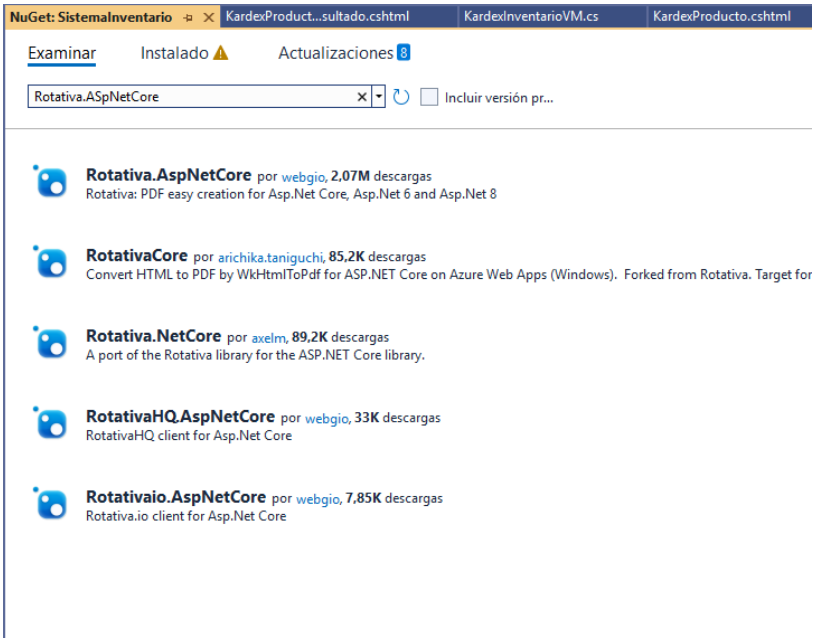
```
        </thead>
        @foreach (var datos in Model.KardexInventarioLista.Where(d => d.BodegaProducto.Bodega.Nombre ==
bodega.Key))
        {
          <tr>
            <td class="text-center">@datos.FechaRegistro.ToShortDateString()</td>
            <td class="text-center">@datos.Tipo</td>
            <td>@datos.Detalle</td>
            <td class="text-center">@datos.StockAnterior</td>
            @if(datos.Tipo == "Entrada")
            {
              <td class="text-center">@datos.Cantidad</td>
            }
            else
            {
              <td class="text-center">--</td>
            }
            @if (datos.Tipo == "Salida")
            {
              <td class="text-center">@datos.Cantidad</td>
            }
            else
            {
              <td class="text-center">--</td>
            }
            <td class="text-center">@datos.Stock</td>
            <td class="text-center">@datos.Costo</td>
            <td class="text-center">@datos.Total</td>
          </tr>
        }


      }
    </table>

    @if(Model.KardexInventarioLista.Count() == 0)
    {
      <div class="col-12 border-bottom p-0">
        <h3 class="text-primary text-center">No hay Datos para Mostrar</h3>
      </div>

    }

</div>
```

Generar Reportes con Rotativa

1. Configuración y Método para Generar Impresión de Kardex de Producto

- En el proyecto SistemaInventario instalamos un paquete NuGet





- En SistemaInventario creamos carpeta llamada Rotativa/Windows
- En SistemaInventario/Rotativa/Windows Agregamos archivos .exe
- Cambiamos la propiedad Copiar en el directorio de salida de los dos archivos a Copiar siempre
- Modificamos SistemaInventario/Program.cs

```
IWebHostEnvironment env = app.Environment;
Rotativa.AspNetCore.RotativaConfiguration.Setup(env.WebRootPath, "..\\Rotativa\\Windows\\");

app.Run();
```

- En SistemaInventario/Areas/Inventario/Controllers/InventarioController.cs agregamos el método de ImprimirKardex sobre la región API

```
public async Task<IActionResult> ImprimirKardex(string fechaInicio,string fechaFinal, int productoId)
{
   KardexInventarioVM kardexInventarioVM = new KardexInventarioVM();
   kardexInventarioVM.Producto = new Producto();
   kardexInventarioVM.Producto = await _unidadTrabajo.Producto.Obtener(productoId);

   kardexInventarioVM.FechaInicio = DateTime.Parse(fechaInicio);
   kardexInventarioVM.FechaFinal = DateTime.Parse(fechaFinal);

   kardexInventarioVM.KardexInventarioLista = await _unidadTrabajo.KardexInventario.ObtenerTodos(
                                        k => k.BodegaProducto.ProductoId == productoId &&
                                           (k.FechaRegistro >= kardexInventarioVM.FechaInicio &&
                                            k.FechaRegistro <= kardexInventarioVM.FechaFinal),
                           incluirPropiedades:
"BodegaProducto,BodegaProducto.Producto,BodegaProducto.Bodega",
                           orderBy: o => o.OrderBy(o => o.FechaRegistro)
      );

   return new ViewAsPdf("ImprimirKardex", kardexInventarioVM)
   {
      FileName = "KardexProducto.pdf",
      PageOrientation = Rotativa.AspNetCore.Options.Orientation.Portrait,
      PageSize = Rotativa.AspNetCore.Options.Size.A4,
      CustomSwitches = "--page-offset 0--footer-center [page]--footer-font-size 12"
   };
}

#region API
 [HttpGet]
```

2. Imprimir Kardex - Vista y PDF

- En SistemaInventario/Areas/Inventario/Controllers/InventarioController.cs agregamos la vista de Tipo de Razor del método ImprimirKardex

```
@model SistemaInventario.Modelos.ViewModels.KardexInventarioVM
@using SistemaInventario.Utilidades

@{
  ViewData["Title"] = "ImprimirKardex";
  Layout = null;
  var titulo = "Kardex Por Producto";
}


<!DOCTYPE html>
<html>
<head>
  <link rel="stylesheet" href="~/lib/bootstrap/dist/css/bootstrap.min.css" />
</head>

<body>


  <table style="width:100%; border : 1px solid white;">
    <thead class="thead-dark">
      <tr>
        <th style="text-align:left;">
          <table style="width:100%; border : 1px solid white;">
            <tr>
              <td style="text-align:center; width:40%;"><h4><strong>@titulo</strong></h4></td>

            </tr>
            <tr>
              <td style="text-align:right; width:30%;"><img
src="@DS.ImagenRuta@Model.Producto.ImagenUrl" style="border-radius:5px; width:100px; height:100px;"
/></td>
            </tr>
          </table>
        </th>

      </tr>
    <thead>
  </table>

  <div class="row">
    <div class="col-10">
      <div class="row mb-2">
        <div class="form-group col-md-8 mt-2">
          <label><strong class="m-2">Desde:</strong>@Model.FechaInicio.ToShortDateString() <strong
class="m-2">Hasta: </strong> @Model.FechaFinal.ToShortDateString() </label>
        </div>
```

```
        </div>
        <div class="row mb-2">
          <div class="form-group col-md-8 mt-2">
            <label><strong class="m-2">Producto:</strong> @Model.Producto.Descripcion</label>
          </div>
        </div>
      </div>

    </div>


    <table class="table table-responsive table-hover table-bordered">
      @foreach (var bodega in Model.KardexInventarioLista.GroupBy(x => x.BodegaProducto.Bodega.Nombre))
      {
        <thead class="table-secondary">
          <tr>
            <th colspan="6">@bodega.Key</th>
            <th colspan="3" class="text-center">Saldo</th>
          </tr>
          <tr>
            <th class="text-center">Fecha</th>
            <th class="text-center">Tipo</th>
            <th class="text-center">Detalle</th>
            <th class="text-center">Stock Anterior</th>
            <th class="text-center">Entrada</th>
            <th class="text-center">Salida</th>
            <th class="text-center">Stock</th>
            <th class="text-center">Costo</th>
            <th class="text-center">Total</th>
          </tr>
        </thead>
        @foreach (var datos in Model.KardexInventarioLista.Where(d => d.BodegaProducto.Bodega.Nombre ==
bodega.Key))
        {
          <tr>
            <td class="text-center">@datos.FechaRegistro.ToShortDateString()</td>
            <td class="text-center">@datos.Tipo</td>
            <td>@datos.Detalle</td>
            <td class="text-center">@datos.StockAnterior</td>
            @if (datos.Tipo == "Entrada")
            {
              <td class="text-center">@datos.Cantidad</td>
            }
            else
            {
              <td class="text-center">--</td>
            }
            @if (datos.Tipo == "Salida")
            {
              <td class="text-center">@datos.Cantidad</td>
            }
            else
            {
```

```
                    <td class="text-center">--</td>
                }
                <td class="text-center">@datos.Stock</td>
                <td class="text-center">@datos.Costo</td>
                <td class="text-center">@datos.Total</td>
            </tr>
        }


    }
</table>

@if (Model.KardexInventarioLista.Count() == 0)
{
    <div class="col-12 border-bottom p-0">
        <h3 class="text-primary text-center">No hay Datos para Mostrar</h3>
    </div>

}

</body>
</html>
```