PROYECTO ASP.Net parte 5

Compania

1. Compania Modelo y Repositorio

- En SistemaInventario.Modelos creamos una clase llamada Compania.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 Compania
   {
      [Key]
      public int Id { get; set; }

      [Required(ErrorMessage ="Nombre es Requerido")]
      [MaxLength(80)]
      public string Nombre { get; set; }

      [Required(ErrorMessage = "Descripcion es Requerido")]
      [MaxLength(200)]
      public string Descripcion { get; set; }

      [Required(ErrorMessage = "Pais es Requerido")]
      [MaxLength(60)]
      public string Pais { get; set; }

      [Required(ErrorMessage = "Ciudad es Requerido")]
      [MaxLength(60)]
      public string Ciudad { get; set; }

      [Required(ErrorMessage = "Direccion es Requerido")]
      [MaxLength(100)]
      public string Direccion { get; set; }

      [Required(ErrorMessage = "Telefono es Requerido")]
      [MaxLength(40)]
      public string Telefono { get; set; }

      [Required(ErrorMessage ="Bodega de Venta es Requerida")]
      public int BodegaVentaId { get; set; }

      [ForeignKey("BodegaVentaId")]
```

```
        public Bodega Bodega { get; set; }

        public string CreadoPorId { get; set; }

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

        public string ActualizadoPorId { get; set; }

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

        public DateTime FechaCreacion { get; set; }

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

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

```
public DbSet<Compania> Companias { get; set; }
```

- En SistemaInventario.AccesoDatos/Configuracion duplicamos el archivo BodegaConfiguracion.cs lo renombramos CompaniaConfiguracion.cs y cambiamos la información relacionada a Compania

```
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 CompaniaConfiguracion : IEntityTypeConfiguration<Compania>
    {
        public void Configure(EntityTypeBuilder<Compania> builder)
        {
            builder.Property(x => x.Id).IsRequired();
            builder.Property(x =>x.Nombre).IsRequired().HasMaxLength(60);
            builder.Property(x =>x.Descripcion).IsRequired().HasMaxLength(100);
            builder.Property(x => x.Pais).IsRequired();
            builder.Property(x => x.Ciudad).IsRequired();
            builder.Property(x => x.Direccion).IsRequired();
            builder.Property(x => x.Telefono).IsRequired();
            builder.Property(x => x.BodegaVentaId).IsRequired();
            builder.Property(x => x.CreadoPorId).IsRequired(false);
```

```
            builder.Property(x => x.ActualizadoPorId).IsRequired(false);

            /* Relaciones*/

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

            builder.HasOne(x => x.CreadoPor).WithMany()
                .HasForeignKey(x => x.CreadoPorId)
                .OnDelete(DeleteBehavior.NoAction);

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

- En SistemaInventario.AccesoDatos/Repositorio/IRepositorio duplicamos el archivo IBodegaRepositorio.cs lo renombramos ICompaniaRepositorio.cs y cambiamos la información relacionada a Compania

```
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 ICompaniaRepositorio : IRepositorio<Compania>
    {
        void Actualizar(Compania compania);

    }
}
```

- En SistemaInventario.AccesoDatos/Repositorio duplicamos el archivo BodegaRepositorio.cs lo renombramos CompaniaRepositorio.cs y cambiamos la información relacionada a Compania

```
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 CompaniaRepositorio : Repositorio<Compania>, ICompaniaRepositorio
   {

      private readonly ApplicationDbContext _db;

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

      public void Actualizar(Compania compania)
      {
        var companiaBD = _db.Companias.FirstOrDefault(b => b.Id == compania.Id);
         if(companiaBD != null)
         {
            companiaBD.Nombre= compania.Nombre;
            companiaBD.Descripcion = compania.Descripcion;
            companiaBD.Pais = compania.Pais;
            companiaBD.Ciudad = compania.Ciudad;
            companiaBD.Direccion = compania.Direccion;
            companiaBD.Telefono = compania.Telefono;
            companiaBD.BodegaVentaId = compania.BodegaVentaId;
            companiaBD.ActualizadoPorId = compania.ActualizadoPorId;
            companiaBD.FechaActualizacion = compania.FechaActualizacion;
            _db.SaveChanges();
         }
      }
   }
}
```

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

```
ICompaniaRepositorio Compania { get; }
```

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

```
public ICompaniaRepositorio Compania { 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);
    Compania = new CompaniaRepositorio(_db);
}
```

2. Migracion y Compania ViewModel

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

- En SistemaInventario.Modelos/ViewModels creamos una clase llamada CompaniaVM.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 CompaniaVM
    {
        public Compania Compania { get; set; }
        public IEnumerable<SelectListItem> BodegaLista { get; set; }
    }
}
```

3. Compania Controlador

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

```
using Microsoft.AspNetCore.Authorization;
using Microsoft.AspNetCore.Mvc;
using SistemaInventario.AccesoDatos.Repositorio.IRepositorio;
using SistemaInventario.Modelos.ViewModels;
using SistemaInventario.Utilidades;
using System.Security.Claims;

namespace SistemaInventario.Areas.Admin.Controllers
{
    [Area("Admin")]
```

```
[Authorize(Roles =DS.Role_Admin)]
public class CompaniaController : Controller
{

    private readonly IUnidadTrabajo _unidadTrabajo;

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

    public async Task<IActionResult> Upsert()
    {
        CompaniaVM companiaVM = new CompaniaVM()
        {
            Compania = new Modelos.Compania(),
            BodegaLista = _unidadTrabajo.Inventario.ObtenerTodosDropdownLista("Bodega")
        };

        companiaVM.Compania = await _unidadTrabajo.Compania.ObtenerPrimero();

        if(companiaVM.Compania == null)
        {
            companiaVM.Compania = new Modelos.Compania();
        }

        return View(companiaVM);
    }

    [HttpPost]
    [ValidateAntiForgeryToken]
    public async Task<IActionResult> Upsert(CompaniaVM companiaVM)
    {

        if(ModelState.IsValid)
        {
            TempData[DS.Exitosa] = "Compania grabada Exitosamente";
            var claimIdentity = (ClaimsIdentity)User.Identity;
            var claim = claimIdentity.FindFirst(ClaimTypes.NameIdentifier);

            if(companiaVM.Compania.Id == 0)  // Crear la Compania
            {
                companiaVM.Compania.CreadoPorId = claim.Value;
                companiaVM.Compania.ActualizadoPorId = claim.Value;
                companiaVM.Compania.FechaCreacion = DateTime.Now;
                companiaVM.Compania.FechaActualizacion = DateTime.Now;
                await _unidadTrabajo.Compania.Agregar(companiaVM.Compania);
            }
            else  // Actualizar Compania
            {
                companiaVM.Compania.ActualizadoPorId = claim.Value;
                companiaVM.Compania.FechaActualizacion = DateTime.Now;
                _unidadTrabajo.Compania.Actualizar(companiaVM.Compania);
```

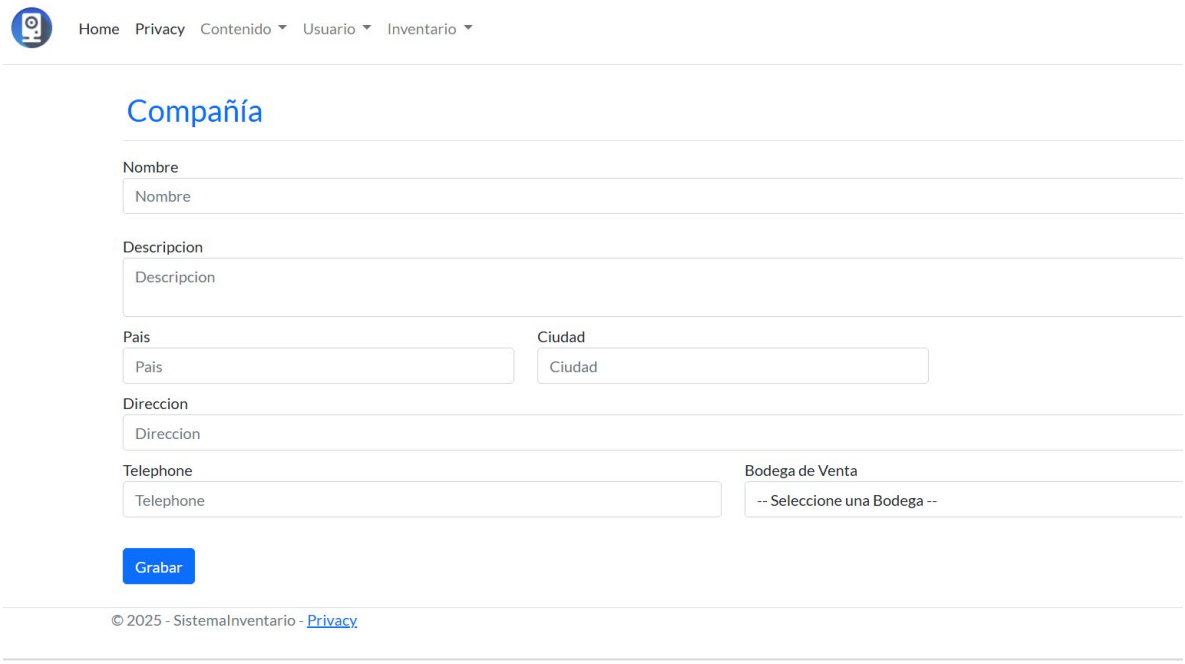
```
            }
            await _unidadTrabajo.Guardar();
            return RedirectToAction("Index", "Home", new {area="Inventario"});
        }
        TempData[DS.Error] = "Error al Grabar Compania";
        return View(companiaVM);
    }

  }
}
```

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

```
@model SistemaInventario.Modelos.ViewModels.CcompaniaVM
@{
  ViewData["Title"] = "Upsert";
  Layout = "~/Views/Shared/_Layout.cshtml";
}

<form method="post">
  <div class="container" style="padding-bottom:0.5rem;">
    <div class="row p-1 border-0">
      <div asp-validation-summary="ModelOnly" class="text-danger"></div>
    </div>

    @if (Model.Compania.Id != 0)
    {
      <input type="hidden" asp-for="Compania.Id" />
    }

    <div class="col-12 border-bottom p-1">
      <h2 class="text-primary">Compañía<i class="fas fa-user-tie"></i></h2>
    </div>

    <div class="form-group mb-2" style="padding-top:14px;">
      <label>Nombre</label>
      <input type="text" asp-for="Compania.Nombre" class="form-control" placeholder="Nombre" />
      <span asp-validation-for="Compania.Nombre" class="text-danger"></span>
    </div>

    <div class="form-group mb-2" style="padding-top:14px;">
      <label>Descripcion</label>
      <textarea type="text" asp-for="Compania.Descripcion" class="form-control" placeholder="Descripcion"
rows=2></textarea>
      <span asp-validation-for="Compania.Descripcion" class="text-danger"></span>
    </div>

    <div class="row mb-2">
      <div class="form-group col-md-4">
        <label>Pais</label>
```

```
            <input type="text" asp-for="Compania.Pais" class="form-control" placeholder="Pais" />
            <span asp-validation-for="Compania.Pais" class="text-danger"></span>
          </div>
          <div class="form-group col-md-4">
            <label>Ciudad</label>
            <input type="text" asp-for="Compania.Ciudad" class="form-control" placeholder="Ciudad" />
            <span asp-validation-for="Compania.Ciudad" class="text-danger"></span>
          </div>
        </div>

        <div class="form-group mb-2">
          <label>Direccion</label>
          <input type="text" asp-for="Compania.Direccion" class="form-control" placeholder="Direccion" />
          <span asp-validation-for="Compania.Direccion" class="text-danger"></span>
        </div>

        <div class="row mb-2">
          <div class="form-group col-md-6">
            <label>Telephone</label>
            <input type="text" asp-for="Compania.Telefono" class="form-control" placeholder="Telephone" />
            <span asp-validation-for="Compania.Telefono" class="text-danger"></span>
          </div>
          <div class="form-group col-md-6">
            <label>Bodega de Venta</label>
            <select asp-for="Compania.BodegaVentaId" asp-items="@Model.BodegaLista" class="form-select">
              <option value="">-- Seleccione una Bodega--</option>
            </select>
            <span asp-validation-for="Compania.BodegaVentaId" class="text-danger"></span>
          </div>

        </div>

        <br />

        <div class="d-grid gap-2 d-md-block">
          <button type="submit" class="btn btn-primary"><i class="fas fa-plus"></i> Grabar</button>
        </div>

      </div>
    </form>

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

- Probamos la ejecución de la aplicación para compania



Carro de Compras - Modelos y Repositorio

1. Compania Modelo y Repositorio

- En SistemaInventario.Modelos creamos una clase llamada CarroCompra.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 CarroCompra
   {
      [Key]
      public int Id { get; set; }

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

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

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

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

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

        [NotMapped]
        public double Precio { get; set; }
    }
}
```

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

```
public DbSet<CarroCompra> CarroCompras { get; set; }
```

- En SistemaInventario.AccesoDatos/Configuracion duplicamos el archivo CompaniaConfiguracion.cs lo renombramos CarroCompraConfiguracion.cs y cambiamos la información relacionada a CarroCompra

```
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 CarroCompraConfiguracion : IEntityTypeConfiguration<CarroCompra>
    {
        public void Configure(EntityTypeBuilder<CarroCompra> builder)
        {
            builder.Property(x => x.Id).IsRequired();
            builder.Property(x => x.UsuarioAplicacionId).IsRequired();
            builder.Property(x => x.ProductoId).IsRequired();
            builder.Property(x => x.Cantidad).IsRequired();

            /* Relaciones*/


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

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

```
    }
  }
}
```

- En SistemaInventario.AccesoDatos/Repositorio/IRepositorio duplicamos el archivo ICompaniaRepositorio.cs lo renombramos ICarroCompraRepositorio.cs y cambiamos la información relacionada a CarroCompra

```
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 ICarroCompraRepositorio : IRepositorio<CarroCompra>
  {
    void Actualizar(CarroCompra carroCompra);

  }
}
```

- En SistemaInventario.AccesoDatos/Repositorio duplicamos el archivo CompaniaRepositorio.cs lo renombramos CarroCompraRepositorio.cs y cambiamos la información relacionada a CarroCompra

```
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 CarroCompraRepositorio : Repositorio<CarroCompra>, ICarroCompraRepositorio
  {

    private readonly ApplicationDbContext _db;

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

    public void Actualizar(CarroCompra carroCompra)
```

```
      {
        _db.Update(carroCompra);
      }
    }
}
```

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

```
ICarroCompraRepositorio CarroCompra { get; }
```

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

```
public ICarroCompraRepositorio CarroCompra { 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);
   Compania = new CompaniaRepositorio(_db);
   CarroCompra = new CarroCompraRepositorio(_db);
}
```

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

2. Orden Modelo y Repositorio

- En SistemaInventario.Modelos creamos una clase llamada Orden.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 Orden
   {
      [Key]
      public int Id { get; set; }

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

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

      public DateTime FechaOrden { get; set; }

      public DateTime FechaEnvio { get; set; }

      public string NumeroEnvio { get; set; }

      public string Carrier { get; set; }

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

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

      public string EstadoPago { get; set; }

      public DateTime FechaPago { get; set; }

      public DateTime FechaMaximaPago { get; set; }

      public string TransaccionId { get; set; }

      public string SessionId { get; set; }

      public string Telefono { get; set; }

      public string Direccion { get; set; }

      public string Ciudad { get; set; }

      public string Pais { get; set; }

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

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

```
public DbSet<Orden> Ordenes { get; set; }
```

- En SistemaInventario.AccesoDatos/Configuracion duplicamos el archivo CarroCompraConfiguracion.cs lo renombramos OrdenConfiguracion.cs y cambiamos la información relacionada a Orden

```
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 OrdenConfiguracion : IEntityTypeConfiguration<Orden>
   {
      public void Configure(EntityTypeBuilder<Orden> builder)
      {
         builder.Property(x => x.Id).IsRequired();
         builder.Property(x => x.UsuarioAplicacionId).IsRequired();
         builder.Property(x => x.FechaOrden).IsRequired();
         builder.Property(x => x.TotalOrden).IsRequired();
         builder.Property(x => x.EstadoOrden).IsRequired();
         builder.Property(x => x.EstadoPago).IsRequired();
         builder.Property(x => x.NombresCliente).IsRequired();
         builder.Property(x => x.NumeroEnvio).IsRequired(false);
         builder.Property(x => x.Carrier).IsRequired(false);
         builder.Property(x => x.TransaccionId).IsRequired(false);
         builder.Property(x => x.Telefono).IsRequired(false);
         builder.Property(x => x.Direccion).IsRequired(false);
         builder.Property(x => x.Ciudad).IsRequired(false);
         builder.Property(x => x.Pais).IsRequired(false);
         builder.Property(x => x.SessionId).IsRequired(false);


         /* Relaciones*/


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

      }
   }
}
```

- En SistemaInventario.AccesoDatos/Repositorio/IRepositorio duplicamos el archivo ICarroCompraRepositorio.cs lo renombramos IOrdenRepositorio.cs y cambiamos la información relacionada a Orden

```
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 IOrdenRepositorio : IRepositorio<Orden>
   {
      void Actualizar(Orden orden);
      void ActualizarEstado(int id, string ordenEstado, string pagoEstado);
      void ActualizarPagoStripeId(int id, string sessionId, string transaccionId);
   }
}
```

- En SistemaInventario.AccesoDatos/Repositorio duplicamos el archivo CarroCompraRepositorio.cs lo renombramos OrdenRepositorio.cs y cambiamos la información relacionada a Orden

```
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 OrdenRepositorio : Repositorio<Orden>, IOrdenRepositorio
   {

      private readonly ApplicationDbContext _db;

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

      public void Actualizar(Orden orden)
      {
        _db.Update(orden);
      }
```

```
        public void ActualizarEstado(int id, string ordenEstado, string pagoEstado)
        {
            var ordenBD = _db.Ordenes.FirstOrDefault(o => o.Id == id);
            if(ordenBD != null)
            {
                ordenBD.EstadoOrden = ordenEstado;
                ordenBD.EstadoPago = pagoEstado;
            }
        }

        public void ActualizarPagoStripeId(int id, string sessionId, string transaccionId)
        {
            var ordenBD = _db.Ordenes.FirstOrDefault(o => o.Id == id);
            if (ordenBD != null)
            {
                if(!String.IsNullOrEmpty(sessionId))
                {
                    ordenBD.SessionId = sessionId;
                }
                if(!string.IsNullOrEmpty(transaccionId))
                {
                    ordenBD.TransaccionId = transaccionId;
                    ordenBD.FechaPago = DateTime.Now;
                }
            }
        }
    }
}
```

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

```
IOrdenRepositorio Orden { get; }
```

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

```
public IOrdenRepositorio Orden { 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);
    Compania = new CompaniaRepositorio(_db);
    CarroCompra = new CarroCompraRepositorio(_db);
    Orden = new OrdenRepositorio(_db);
}
```

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

3. Orden Detalle Modelo y Repositorio

- En SistemaInventario.Modelos creamos una clase llamada OrdenDetalle.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 OrdenDetalle
    {
        [Key]
        public int Id { get; set; }

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

        [ForeignKey("OrdenId")]
        public Orden Orden { get; set; }

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

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

        public int Cantidad { get; set; }

        public double Precio { get; set; }
    }
}
```

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

```
public DbSet<OrdenDetalle> OrdenDetalles { get; set; }
```

- En SistemaInventario.AccesoDatos/Configuracion duplicamos el archivo OrdenConfiguracion.cs lo renombramos OrdenDetalleConfiguracion.cs y cambiamos la información relacionada a OrdenDetalle

```
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 OrdenDetalleConfiguracion : IEntityTypeConfiguration<OrdenDetalle>
   {
      public void Configure(EntityTypeBuilder<OrdenDetalle> builder)
      {
         builder.Property(x => x.Id).IsRequired();
         builder.Property(x => x.OrdenId).IsRequired();
         builder.Property(x => x.ProductoId).IsRequired();
         builder.Property(x => x.Cantidad).IsRequired();
         builder.Property(x => x.Precio).IsRequired();


         /* Relaciones*/


         builder.HasOne(x => x.Orden).WithMany()
               .HasForeignKey(x => x.OrdenId)
               .OnDelete(DeleteBehavior.NoAction);

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

      }
   }
}
```

- En SistemaInventario.AccesoDatos/Repositorio/IRepositorio duplicamos el archivo IOrdenRepositorio.cs lo renombramos IOrdenDetalleRepositorio.cs y cambiamos la información relacionada a OrdenDetalle

```
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 IOrdenDetalleRepositorio : IRepositorio<OrdenDetalle>
   {
      void Actualizar(OrdenDetalle ordenDetalle);

   }
}
```

- En SistemaInventario.AccesoDatos/Repositorio duplicamos el archivo OrdenRepositorio.cs lo renombramos OrdenDetalleRepositorio.cs y cambiamos la información relacionada a OrdenDetalle

```
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 OrdenDetalleRepositorio : Repositorio<OrdenDetalle>, IOrdenDetalleRepositorio
   {

      private readonly ApplicationDbContext _db;

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

      public void Actualizar(OrdenDetalle ordenDetalle)
      {
        _db.Update(ordenDetalle);
      }
   }
}
```

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

```
IOrdenDetalleRepositorio OrdenDetalle { get; }
```

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

```
public IOrdenDetalleRepositorio OrdenDetalle { 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);
   Compania = new CompaniaRepositorio(_db);
   CarroCompra = new CarroCompraRepositorio(_db);
   Orden = new OrdenRepositorio(_db);
   OrdenDetalle = new OrdenDetalleRepositorio(_db);
}
```

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

Carro de Compras - Detalle

1. Detalle Productos - Metodo Get

- Modificamos el archivo SistemaInventario/Areas/Inventario/Views/Home/Index.cshtml para el botón Detalle

```
<a asp-action="Detalle" asp-route-id="@producto.Id" class="btn btn-outline-primary">
   <i class="bi bi-tags-fill"></i> Detalle
</a>
```

En SistemaInventario.Modelos/ViewModels creamos una clase llamada CarroCompraVM.cs

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

namespace SistemaInventario.Modelos.ViewModels
{
```

```
  public class CarroCompraVM
  {
    public Compania Compania { get; set; }
    public Producto Producto { get; set; }

    public int Stock { get; set; }

    public CarroCompra CarroCompra { get; set; }

    public IEnumerable<CarroCompra> CarroCompraLista { get; set; }
    public Orden Orden { get; set; }
  }
}
```

- En SistemaInventario/Areas/Inventario/Controllers/HomeController.cs agregamos el método Detalle después del index

```
private readonly ILogger<HomeController> _logger;
private readonly IUnidadTrabajo _unidadTrabajo;
[BindProperty]
public CarroCompraVM carroCompraVM { get; set; }
.
.
.
public async Task<IActionResult> Detalle(int id)
{
  carroCompraVM = new CarroCompraVM();
  carroCompraVM.Compania = await _unidadTrabajo.Compania.ObtenerPrimero();
  carroCompraVM.Producto = await _unidadTrabajo.Producto.ObtenerPrimero(p => p.Id == id,
                              incluirPropiedades: "Marca,Categoria");
  var bodegaProducto = await _unidadTrabajo.BodegaProducto.ObtenerPrimero(b=>b.ProductoId == id &&
                                      b.BodegaId== carroCompraVM.Compania.BodegaVentaId);
  if (bodegaProducto==null)
  {
    carroCompraVM.Stock = 0;
  }
  else
  {
    carroCompraVM.Stock = bodegaProducto.Cantidad;
  }
  carroCompraVM.CarroCompra = new CarroCompra()
  {
    Producto = carroCompraVM.Producto,
    ProductoId = carroCompraVM.Producto.Id
  };

  return View(carroCompraVM);
}
```

2. Detalle Productos - Vista

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

```
@model SistemaInventario.Modelos.ViewModels.CarroCompraVM
@using SistemaInventario.Utilidades
@{
   ViewData["Title"] = "Detalle";
   Layout = "~/Views/Shared/_Layout.cshtml";
}

<form method="post">
   <main class="mt-3 pt-2">
      <div class="container mt-5">
         <input asp-for="Stock" hidden id="stockId" />
         <input hidden asp-for="CarroCompra.ProductoId" />

         <div class="row">
            <div class="col-md-4 mb-4">
               <img src="@DS.ImagenRuta@Model.Producto.ImagenUrl" class="img-fluid" alt="" />
               <div class="row ">
                  <a asp-action="Index" class="btn btn-outline-primary ms-1">
                     Salir
                  </a>
               </div>
            </div>


            <div class="col-md-6 mb-4">
               <div class="p-4">
                  <div class="mb-3">
                     <span class="badge bg-dark p-2 me-2">@Model.Producto.Categoria.Nombre</span>
                     <span class="badge bg-info p-2">@Model.Producto.Marca.Nombre</span>
                  </div>

                  <p class="lead text-black">
                     <h2>$ @String.Format("{0:#,##0.00}", Model.Producto.Precio)</h2>
                  </p>

                  <strong style="font-size: 20px;">@Model.Producto.Descripcion</strong>

                  <p>Serie: @Model.Producto.NumeroSerie</p>

                  @if (Model.Stock == 0)
                  {
                     <div p-4>
                        <strong class="text-danger">Sin Stock</strong>
                     </div>


                  }
                  else
```

```
                {
                    <div p-4>
                        <strong class="text-primary">En Stock: @Model.Stock</strong>
                    </div>
                }


                <div class="d-flex justify-content-left mt-2">
                    <div class="form-outline me-1" style="width: 100px;">
                        <input asp-for="CarroCompra.Cantidad" value="1" class="form-control" id="cantidadId" />
                    </div>
                    <button type="submit" class="btn btn-primary ms-1" id="btnAgregar">
                        Agregar al carro
                        <i class="bi bi-cart"></i>
                    </button>

                </div>
            </div>
        </div>
    </div>


</main>
</form>

@section Scripts{

    <script>
        $("#btnAgregar").click(function () {
            let stock = document.getElementById("stockId").value;
            let cantidad = document.getElementById("cantidadId").value;
            if (cantidad.toString() == '' || cantidad < 1 || parseInt(cantidad) > parseInt(stock)) {
                swal("Error", "Seleccione una Cantidad correcta", "error");
                return false;
            }
        });
    </script>
}
```

3. Agregar Productos al Carro de Compras.

- En SistemaInventario/Areas/Inventario/Controllers/HomeController.cs agregamos el método Post de Detalle

```
[HttpPost]
[ValidateAntiForgeryToken]
[Authorize]
public async Task<IActionResult> Detalle(CarroCompraVM carroCompraVM)
{
    var claimIdentity = (ClaimsIdentity)User.Identity;
```

```
    var claim = claimIdentity.FindFirst(ClaimTypes.NameIdentifier);
    carroCompraVM.CarroCompra.UsuarioAplicacionId = claim.Value;

    CarroCompra carroBD = await _unidadTrabajo.CarroCompra.ObtenerPrimero(c => c.UsuarioAplicacionId ==
claim.Value &&
                                                c.ProductoId == carroCompraVM.CarroCompra.ProductoId);
    if(carroBD == null)
    {
        await _unidadTrabajo.CarroCompra.Agregar(carroCompraVM.CarroCompra);
    }
    else
    {
        carroBD.Cantidad += carroCompraVM.CarroCompra.Cantidad;
        _unidadTrabajo.CarroCompra.Actualizar(carroBD);
    }
    await _unidadTrabajo.Guardar();
    TempData[DS.Exitosa] = "Producto agregado al Carro de Compras";

    // Agregar valor a la Sesion
    var carroLista = await _unidadTrabajo.CarroCompra.ObtenerTodos(c => c.UsuarioAplicacionId == claim.Value);
    var numeroProductos = carroLista.Count();  // Numero de Registros
    HttpContext.Session.SetInt32(DS.ssCarroCompras, numeroProductos);

    return RedirectToAction("Index");

}
```

4. Configurar Sesion en el Proyecto

- En SistemaInventario/Programs agregamos el servicio AddSession

```
builder.Services.AddSingleton<IEmailSender, EmailSender>();

builder.Services.AddSession(options =>
{
    options.IdleTimeout = TimeSpan.FromMinutes(30);
    options.Cookie.HttpOnly = true;
    options.Cookie.IsEssential = true;
});
.
.
.
app.UseRouting();

app.UseSession();
```

5. Carro de Compras - Refrescar Numero de Productos

- Modificamos el método Index de SistemaInventario/Areas/Inventario/Controllers/HomeController.cs

```
public async Task<IActionResult> Index(int pageNumber = 1, string busqueda="", string busquedaActual="")
{
   // Controlar sesion
   var claimIdentity = (ClaimsIdentity)User.Identity;
   var claim = claimIdentity.FindFirst(ClaimTypes.NameIdentifier);
   if(claim!=null)
   {
      var carroLista = await _unidadTrabajo.CarroCompra.ObtenerTodos(c => c.UsuarioAplicacionId ==
claim.Value);
      var numeroProductos = carroLista.Count();  // Numero de Registros
      HttpContext.Session.SetInt32(DS.ssCarroCompras, numeroProductos);
   }

   //
   if (!String.IsNullOrEmpty(busqueda))
   {
      pageNumber = 1;
   }
   else
   {
      busqueda = busquedaActual;
   }
   ViewData["BusquedaActual"] = busqueda;

   if (pageNumber < 1) { pageNumber = 1; }

   Parametros parametros = new Parametros()
   {
      PageNumber = pageNumber,
      PageSize = 4
   };

   var resultado = _unidadTrabajo.Producto.ObtenerTodosPaginado(parametros);

   if(!String.IsNullOrEmpty(busqueda))
   {
      resultado = _unidadTrabajo.Producto.ObtenerTodosPaginado(parametros, p
=>p.Descripcion.Contains(busqueda));
   }

   ViewData["TotalPaginas"] = resultado.MetaData.TotalPages;
   ViewData["TotalRegistros"] = resultado.MetaData.TotalCount;
   ViewData["PageSize"] = resultado.MetaData.PageSize;
   ViewData["PageNumber"] = pageNumber;
   ViewData["Previo"] = "disabled";  // clase css para desactivar el boton
   ViewData["Siguiente"] = "";

   if (pageNumber > 1) { ViewData["Previo"] = ""; }
   if (resultado.MetaData.TotalPages <= pageNumber) { ViewData["Siguiente"] = "disabled"; }

   return View(resultado);
}
```

- Modificamos SistemaInventario/Areas/Identity/Pages/Account/Login.cshtml.cs

```
public class LoginModel : PageModel
{
   private readonly SignInManager<IdentityUser> _signInManager;
   private readonly ILogger<LoginModel> _logger;
   private readonly IUnidadTrabajo _unidadTrabajo;

   public LoginModel(SignInManager<IdentityUser> signInManager, ILogger<LoginModel> logger,
IUnidadTrabajo unidadTrabajo)
   {
      _signInManager = signInManager;
      _logger = logger;
      _unidadTrabajo = unidadTrabajo;
   }
.
.
.
public async Task<IActionResult> OnPostAsync(string returnUrl = null)
{
   returnUrl ??= Url.Content("~/");

   ExternalLogins = (await _signInManager.GetExternalAuthenticationSchemesAsync()).ToList();

   if (ModelState.IsValid)
   {
      // This doesn't count login failures towards account lockout
      // To enable password failures to trigger account lockout, set lockoutOnFailure: true
      var result = await _signInManager.PasswordSignInAsync(Input.Email, Input.Password, Input.RememberMe,
lockoutOnFailure: false);
      if (result.Succeeded)
      {
         var usuario = await _unidadTrabajo.UsuarioAplicacion.ObtenerPrimero(u => u.UserName ==
Input.Email);
         var carroLista = await _unidadTrabajo.CarroCompra.ObtenerTodos(c => c.UsuarioAplicacionId ==
usuario.Id);
         var numeroProductos = carroLista.Count();
         HttpContext.Session.SetInt32(DS.ssCarroCompras, numeroProductos);
         _logger.LogInformation("User logged in.");
         return LocalRedirect(returnUrl);
      }
      if (result.RequiresTwoFactor)
      {
         return RedirectToPage("./LoginWith2fa", new { ReturnUrl = returnUrl, RememberMe =
Input.RememberMe });
      }
      if (result.IsLockedOut)
      {
         _logger.LogWarning("User account locked out.");
         return RedirectToPage("./Lockout");
      }
      else
```

```
        {
            ModelState.AddModelError(string.Empty, "Usuario no esta Registrado o la Cuenta no ha sido confirmada.");
            return Page();
        }
    }

    // If we got this far, something failed, redisplay form
    return Page();
}
```

- Modificamos el metodo OnPost de SistemaInventario/Areas/Identity/Pages/Account/Logout.cshtml.cs

```
public async Task<IActionResult> OnPost(string returnUrl = null)
 {
    HttpContext.Session.SetInt32(DS.ssCarroCompras, 0);
.
.
.
```

- Modificamos SistemaInventario/Views/Shared/_Layout.cshtml

```
    @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>
    }
    @if (httpContextAccessor.HttpContext.Session.GetInt32(DS.ssCarroCompras) != null)
    {
        <li>
            <a asp-area="Inventario" asp-controller="Carro" asp-action="Index" class="nav-link">
                @{
                    var numeroProductos = httpContextAccessor.HttpContext.Session.GetInt32(DS.ssCarroCompras);
                }
                <i class="bi bi-cart-fill text-primary">  (@numeroProductos) </i>
            </a>
        </li>
    }
    else
    {
        <li>
            <a href="#" class="nav-link">
                <i class="bi bi-cart-fill text-primary">  (0) </i>
            </a>
```

```
    </li>
  }
</ul>
```

Carro de Compras - Administración

1. Carro Compras - Index Metodo Get

- En SistemaInventario/Areas/Inventario/Controllers Creamos el controlador de tipo Controlador de MVC en blanco llamado CarroController.cs

```
using Microsoft.AspNetCore.Authorization;
using Microsoft.AspNetCore.Mvc;
using SistemaInventario.AccesoDatos.Repositorio.IRepositorio;
using SistemaInventario.Modelos;
using SistemaInventario.Modelos.ViewModels;
using SistemaInventario.Utilidades;
using System.Collections.Generic;
using System.Security.Claims;

namespace SistemaInventario.Areas.Inventario.Controllers
{
  [Area("Inventario")]
  public class CarroController : Controller
  {

    private readonly IUnidadTrabajo _unidadTrabajo;

    [BindProperty]
    public CarroCompraVM carroCompraVM { get; set; }

    public CarroController(IUnidadTrabajo unidadTrabajo, IConfiguration configuration)
    {
      _unidadTrabajo = unidadTrabajo;
    }

    [Authorize]
    public async Task<IActionResult> Index()
    {
      var claimIdentity = (ClaimsIdentity)User.Identity;
      var claim = claimIdentity.FindFirst(ClaimTypes.NameIdentifier);

      carroCompraVM = new CarroCompraVM();
      carroCompraVM.Orden = new Modelos.Orden();
      carroCompraVM.CarroCompraLista = await _unidadTrabajo.CarroCompra.ObtenerTodos(
                                u => u.UsuarioAplicacionId == claim.Value,
                                incluirPropiedades: "Producto");

      carroCompraVM.Orden.TotalOrden = 0;
      carroCompraVM.Orden.UsuarioAplicacionId = claim.Value;
```

```
        foreach (var lista in carroCompraVM.CarroCompraLista)
        {
            lista.Precio = lista.Producto.Precio;  // Siempre mostrar el Precio actual del Producto
            carroCompraVM.Orden.TotalOrden += (lista.Precio * lista.Cantidad);
        }

        return View(carroCompraVM);
    }
  }
}
```

2. Carro Compras - Index Vista

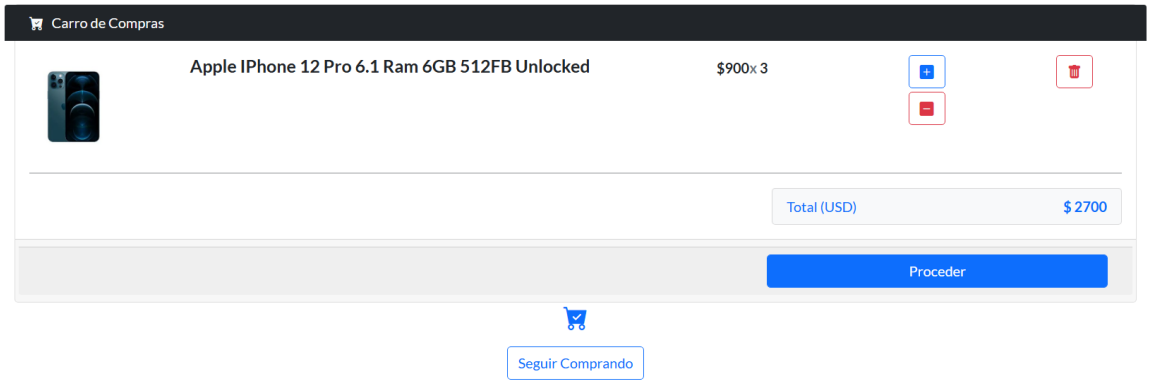
- Desde el método Index En SistemaInventario/Areas/Inventario/Controllers/CarroController.cs agregamos la vista de tipo Vista de Razor.

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


<form method="post">
  <br />
  @if (Model.CarroCompraLista.Count() > 0)
  {
    <div>
      <div class="container">
        <div class="card">
          <div class="card-header bg-dark text-light ml-0 row ">
            <div class="col-6">
              <i class="bi bi-cart-check-fill"></i>  
              Carro de Compras
            </div>

          </div>
          <div class="card-body">
            @foreach (var item in Model.CarroCompraLista)
            {
              <div class="row">
                <div class="d-none d-lg-block col-lg-1 text-center py-2">
                  <img src="@DS.ImagenRuta@item.Producto.ImagenUrl" class="rounded" width="100" />
                </div>
                <div class="col-12 text-sm-center col-lg-6 text-lg-left">
                  <h5><strong>@item.Producto.Descripcion</strong></h5>
                </div>
                <div class="col-12 text-sm-center col-lg-5 text-lg-right row">
                  <div class="col-4 text-md-right" style="padding-top:5px;">
                    <h6><strong>$@item.Producto.Precio<span class="text-muted">x</span>
@item.Cantidad </strong></h6>
                  </div>
                  <div class="col-6 col-sm-4 col-lg-6">
                    <div class="float-right mx-1 mb-1">
```

```
                        <a asp-action="mas" asp-route-carroId="@item.Id" class="btn btn-outline-primary">
                          <i class="bi bi-plus-square-fill"></i>
                        </a>
                      </div>
                      <div class="float-right mx-1">
                        <a asp-action="menos" asp-route-carroId="@item.Id" class="btn btn-outline-
danger">
                          <i class="bi bi-dash-square-fill"></i>
                        </a>
                      </div>
                    </div>
                    <div class="col-2 col-sm-4 col-lg-2 text-right">
                      <a asp-action="remover" asp-route-carroId="@item.Id" class="btn btn-outline-danger">
                        <i class="bi bi-trash-fill"></i>
                      </a>
                    </div>
                  </div>
                </div>
                <hr />
              }

              <div class="row">
                <div class="col-12 col-md-6 offset-md-6 col-lg-4 offset-lg-8 pr-4">
                  <ul class="list-group">
                    <li class="list-group-item d-flex justify-content-between bg-light">
                      <span class="text-primary"> Total (USD)</span>
                      <strong class="text-primary">$ <span
id="txtTotalOrden">@Model.Orden.TotalOrden</span></strong>
                    </li>
                  </ul>
                </div>
              </div>

            </div>
            <div class="card-footer">
              <div class="card-footer row">

                <div class="col-sm-12 col-lg-4 col-md-6 offset-lg-8 offset-md-6 ">

                  <a asp-area="Inventario" asp-controller="Carro" asp-action="Proceder" class="btn btn-
primary form-control">Proceder</a>
                </div>
              </div>
            </div>
          </div>
        </div>
      </div>
    }
    <div class="col-12  p-0">
      <h3 class="text-primary text-center"><i class="bi bi-cart-check-fill"></i></h3>
    </div>
    <div class="col-12 p-2">
      @if (Model.CarroCompraLista.Count() == 0)
```

```
    {
      <h3 class="text-primary text-center">No hay Datos para Mostrar</h3>
    }

    <div class="text-center border-0">
      <a asp-area="Inventario" asp-controller="Home" asp-action="Index" class="btn btn-outline-primary
">Seguir Comprando</a>
    </div>

  </div>

</form>
```

- Verificamos funcionamiento de la aplicación.



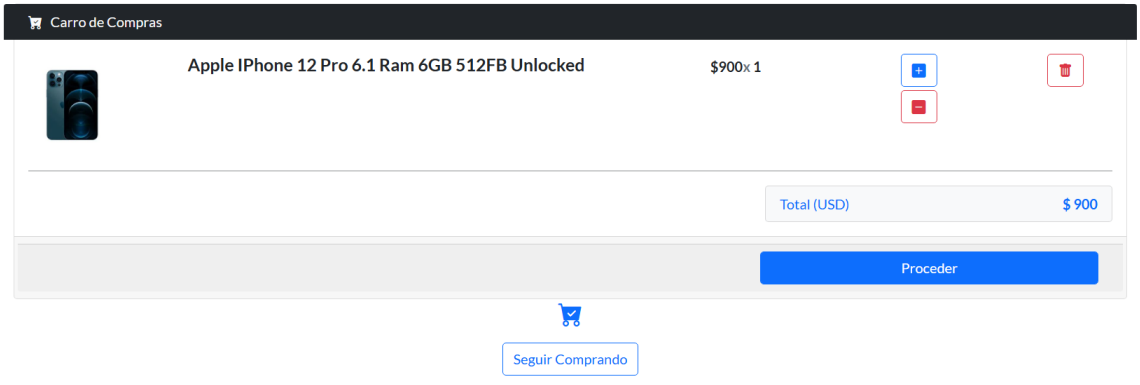
3. Carro Compras - Incremenetar_ Disminuir_ Remover

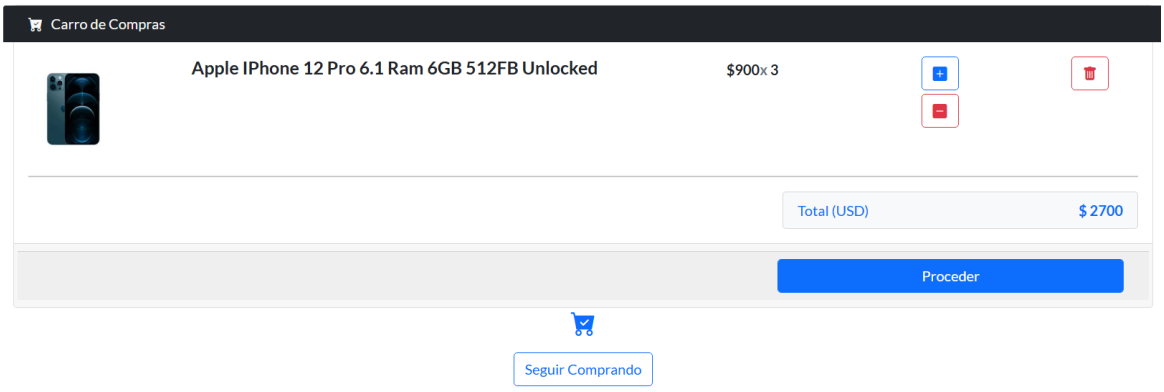
- En SistemaInventario/Areas/Inventario/Controllers/CarroController.cs Creamos métodos mas, menos y remover.

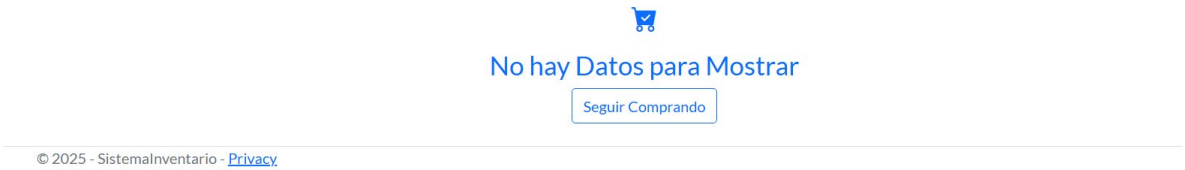
```
public async Task<IActionResult> mas(int carroId)
{
  var carroCompras = await _unidadTrabajo.CarroCompra.ObtenerPrimero(c => c.Id == carroId);
  carroCompras.Cantidad += 1;
  await _unidadTrabajo.Guardar();
  return RedirectToAction("Index");
}

public async Task<IActionResult> menos(int carroId)
{
  var carroCompras = await _unidadTrabajo.CarroCompra.ObtenerPrimero(c => c.Id == carroId);

  if (carroCompras.Cantidad == 1)
  {
    // Removemos el Registro del Carro de Compras y actualizamos la sesion
    var carroLista = await _unidadTrabajo.CarroCompra.ObtenerTodos(
```

```
                        c => c.UsuarioAplicacionId == carroCompras.UsuarioAplicacionId);
        var numeroProductos = carroLista.Count();
        _unidadTrabajo.CarroCompra.Remover(carroCompras);
        await _unidadTrabajo.Guardar();
        HttpContext.Session.SetInt32(DS.ssCarroCompras, numeroProductos- 1);
    }
    else
    {
        carroCompras.Cantidad-= 1;
        await _unidadTrabajo.Guardar();
    }
    return RedirectToAction("Index");
}

public async Task<IActionResult> remover(int carroId)
{
    // Remueve el Registro del Carro de Compras y Actualiza la sesion
    var carroCompras = await _unidadTrabajo.CarroCompra.ObtenerPrimero(c => c.Id == carroId);
    var carroLista = await _unidadTrabajo.CarroCompra.ObtenerTodos(
                        c => c.UsuarioAplicacionId == carroCompras.UsuarioAplicacionId);
    var numeroProductos = carroLista.Count();
    _unidadTrabajo.CarroCompra.Remover(carroCompras);
    await _unidadTrabajo.Guardar();
    HttpContext.Session.SetInt32(DS.ssCarroCompras, numeroProductos- 1);
    return RedirectToAction("Index");
}
```

- Verificamos funcionamiento de la aplicación.







Carro de Compras - Pagos

1. Proceder Metodo Get

- En SistemaInventario/Areas/Inventario/Controllers/CarroController.cs Creamos método proceder

```
public async Task<IActionResult> Proceder()
{
    var claimIdentidad = (ClaimsIdentity)User.Identity;
    var claim = claimIdentidad.FindFirst(ClaimTypes.NameIdentifier);

    carroCompraVM = new CarroCompraVM()
    {
        Orden = new Modelos.Orden(),
        CarroCompraLista = await _unidadTrabajo.CarroCompra.ObtenerTodos(
                c => c.UsuarioAplicacionId == claim.Value, incluirPropiedades: "Producto"),
        Compania = await _unidadTrabajo.Compania.ObtenerPrimero()
    };

    carroCompraVM.Orden.TotalOrden = 0;
    carroCompraVM.Orden.UsuarioAplicacion = await
_unidadTrabajo.UsuarioAplicacion.ObtenerPrimero(u=>u.Id==
                                                claim.Value);

    foreach (var lista in carroCompraVM.CarroCompraLista)
    {
        lista.Precio = lista.Producto.Precio;
        carroCompraVM.Orden.TotalOrden += (lista.Precio * lista.Cantidad);
```

```
    }
    carroCompraVM.Orden.NombresCliente = carroCompraVM.Orden.UsuarioAplicacion.Nombres + " " +
                                carroCompraVM.Orden.UsuarioAplicacion.Apellidos;
    carroCompraVM.Orden.Telefono = carroCompraVM.Orden.UsuarioAplicacion.PhoneNumber;
    carroCompraVM.Orden.Direccion = carroCompraVM.Orden.UsuarioAplicacion.Direccion;
    carroCompraVM.Orden.Pais = carroCompraVM.Orden.UsuarioAplicacion.Pais;
    carroCompraVM.Orden.Ciudad = carroCompraVM.Orden.UsuarioAplicacion.Ciudad;

    // Controlar Stock
    foreach (var lista in carroCompraVM.CarroCompraLista)
    {
      // Capturar el Stock de cada Producto
      var producto = await _unidadTrabajo.BodegaProducto.ObtenerPrimero(b => b.ProductoId ==
lista.ProductoId &&
                                                b.BodegaId == carroCompraVM.Compania.BodegaVentaId);
      if(lista.Cantidad > producto.Cantidad)
      {
        TempData[DS.Error] = "La Cantidad del Producto " + lista.Producto.Descripcion +
                    " Excede al Stock actual (" + producto.Cantidad + ")";
        return RedirectToAction("Index");
      }
    }
    return View(carroCompraVM);
  }
```

2. Proceder Vista

- Desde el método Proceder En SistemaInventario/Areas/Inventario/Controllers/CarroController.cs agregamos la vista de tipo Vista de Razor.

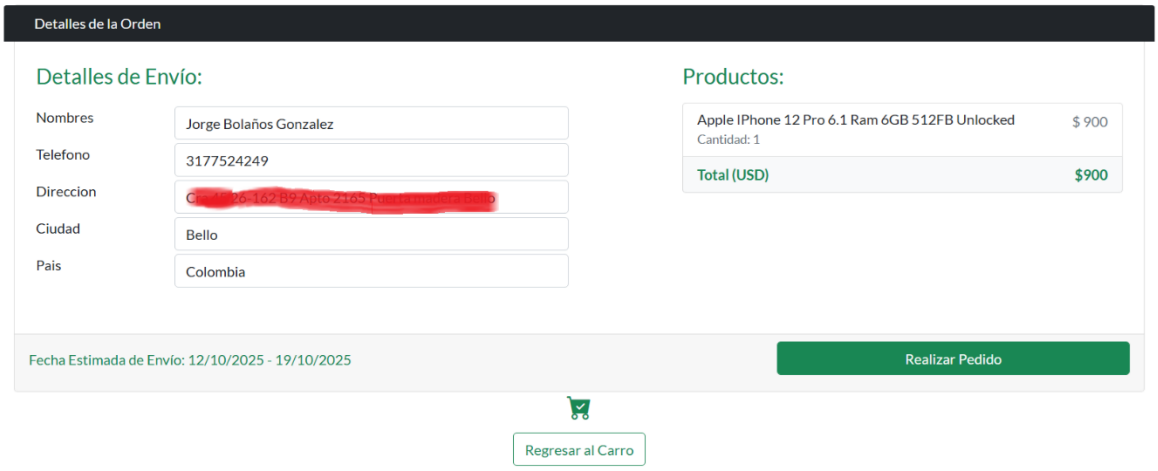
```
@model SistemaInventario.Modelos.ViewModels.CarroCompraVM
<form method="post">
  <br />
  <div class="backgroundWhiteBorder">
    <div class="container">
      <div class="card">
        <div class="card-header bg-dark text-light ml-0 row">
          <div class="col-6">
            <i class="fa fa-shopping-cart"></i>  
            Detalles de la Orden
          </div>

        </div>
        <div class="card-body">
          <div class="container rounded p-2">
            <div class="row">
              <div class="col-12 col-lg-6 pb-4">
                <div class="row">
                  <h4 class="d-flex justify-content-between align-items-center mb-3">
                    <span class="text-success">Detalles de Envío:</span>
                  </h4>
```

```
                    </div>
                    <div class="row my-1">
                      <div class="col-3">
                        <label>Nombres</label>
                      </div>
                      <div class="col-9">
                        <input asp-for="Orden.NombresCliente" type="text" class="form-control"
id="nombresId" />
                      </div>
                    </div>
                    <div class="row my-1">
                      <div class="col-3">
                        <label>Telefono</label>
                      </div>
                      <div class="col-9">
                        <input asp-for="Orden.Telefono" type="text" class="form-control" id="telefonoId" />
                      </div>
                    </div>
                    <div class="row my-1">
                      <div class="col-3">
                        <label>Direccion</label>
                      </div>
                      <div class="col-9">
                        <input asp-for="Orden.Direccion" type="text" class="form-control" id="direccionId"/>
                      </div>
                    </div>
                    <div class="row my-1">
                      <div class="col-3">
                        <label>Ciudad</label>
                      </div>
                      <div class="col-9">
                        <input asp-for="Orden.Ciudad" type="text" class="form-control" id="ciudadId"/>
                      </div>
                    </div>
                    <div class="row my-1">
                      <div class="col-3">
                        <label>Pais</label>
                      </div>
                      <div class="col-9">
                        <input asp-for="Orden.Pais" type="text" class="form-control" id="paisId"/>
                      </div>
                    </div>
                  </div>
                  <div class="col-12 col-lg-5 offset-lg-1">
                    <h4 class="d-flex justify-content-between align-items-center mb-3">
                      <span class="text-success">Productos:</span>
                    </h4>
                    <ul class="list-group mb-3">
                      @foreach (var item in Model.CarroCompraLista)
                      {
                        <li class="list-group-item d-flex justify-content-between">
                          <div>
                            <h6 class="my-0">@item.Producto.Descripcion</h6>
```

```
                        <small class="text-muted">Cantidad: @item.Cantidad</small>
                      </div>
                      <span class="text-muted">$ @(item.Producto.Precio * item.Cantidad)</span>
                    </li>
                  }

                  <li class="list-group-item d-flex justify-content-between bg-light">
                    <strong class="text-success">Total (USD)</strong>
                    <strong class="text-success">$@(Model.Orden.TotalOrden)</strong>
                  </li>
                </ul>
              </div>
            </div>
          </div>
          <div class="card-footer">
            <div class="row">
              <div class="col-12 col-md-8 pt-2">
                <p class="text-success">Fecha Estimada de Envío: @DateTime.Now.AddDays(7).ToShortDateString()- @DateTime.Now.AddDays(14).ToShortDateString()</p>
              </div>
              <div class="col-12 col-md-4">

                <button type="submit" value="Realizar Pedido" class="btn btn-success form-control" id="procederId">Realizar Pedido</button>
              </div>
            </div>
          </div>
        </div>
      </div>
    </div>
    <div class="col-12  p-0">
      <h3 class="text-success text-center"><i class="bi bi-cart-check-fill"></i></h3>
    </div>
    <div class="col-12 p-1">
      <div class="text-center border-0">
        <a asp-area="Inventario" asp-controller="Carro" asp-action="Index" class="btn btn-outline-success ">Regresar al Carro</a>
      </div>

    </div>
</form>

 @section Scripts {
    <script>

      $("#procederId").click(function () {
        let nombresId = document.getElementById("nombresId").value;
        let telefonoId = document.getElementById("telefonoId").value;
        let direccionId = document.getElementById("direccionId").value;
        let ciudadId = document.getElementById("ciudadId").value;
        let paisId = document.getElementById("paisId").value;
```

```
        if (nombresId == '') {
            swal("Error", "Ingresar Nombres", "error");
            return false;
        }
        if (telefonoId == '') {
            swal("Error", "Ingresar Telefono", "error");
            return false;
        }
        if (direccionId == '') {
            swal("Error", "Ingresar Direccion", "error");
            return false;
        }
        if (ciudadId == '') {
            swal("Error", "Ingresar Ciudad", "error");
            return false;
        }
        if (paisId == '') {
            swal("Error", "Ingresar Pais", "error");
            return false;
        }
    });

</script>

}
```

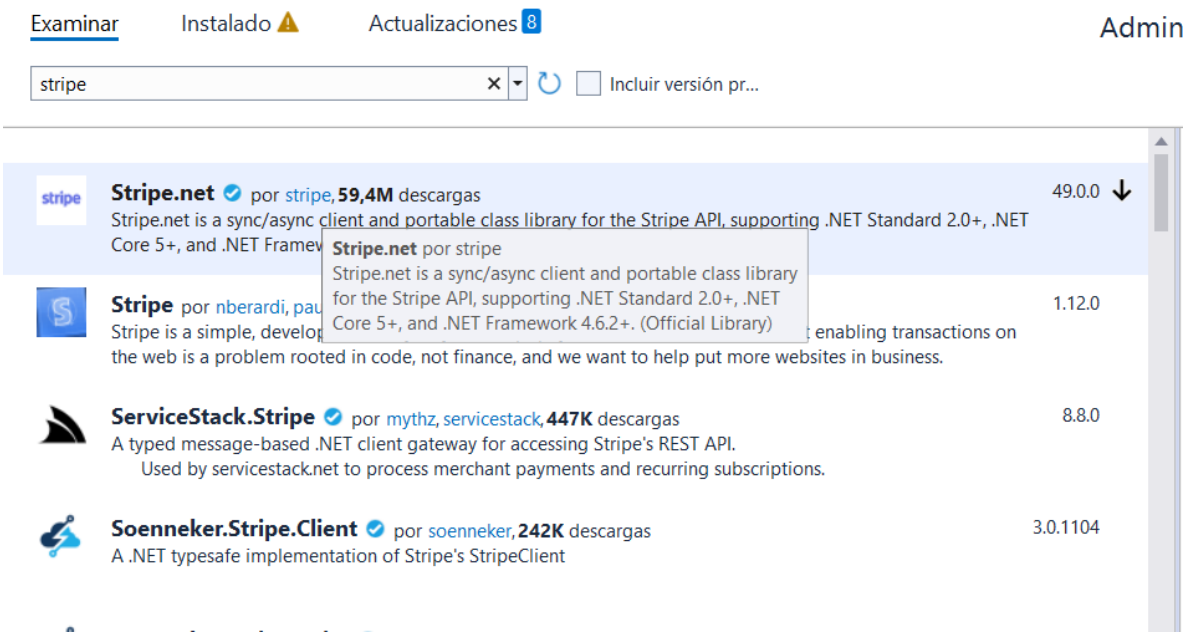
- Verificamos funcionalidad de la aplicación para el botón proceder.



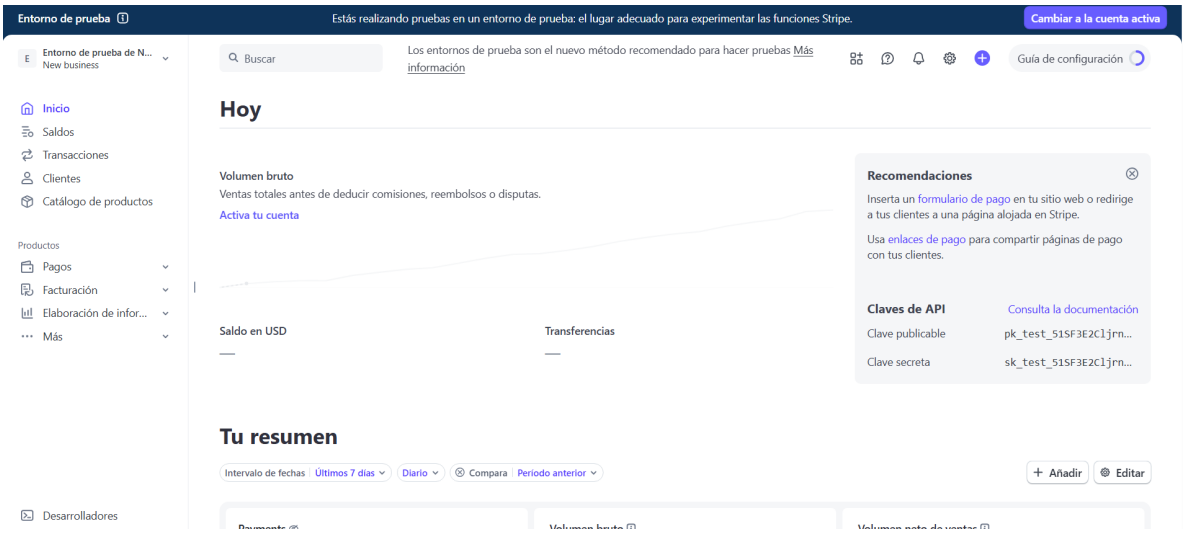
3. Configuracion Stripe

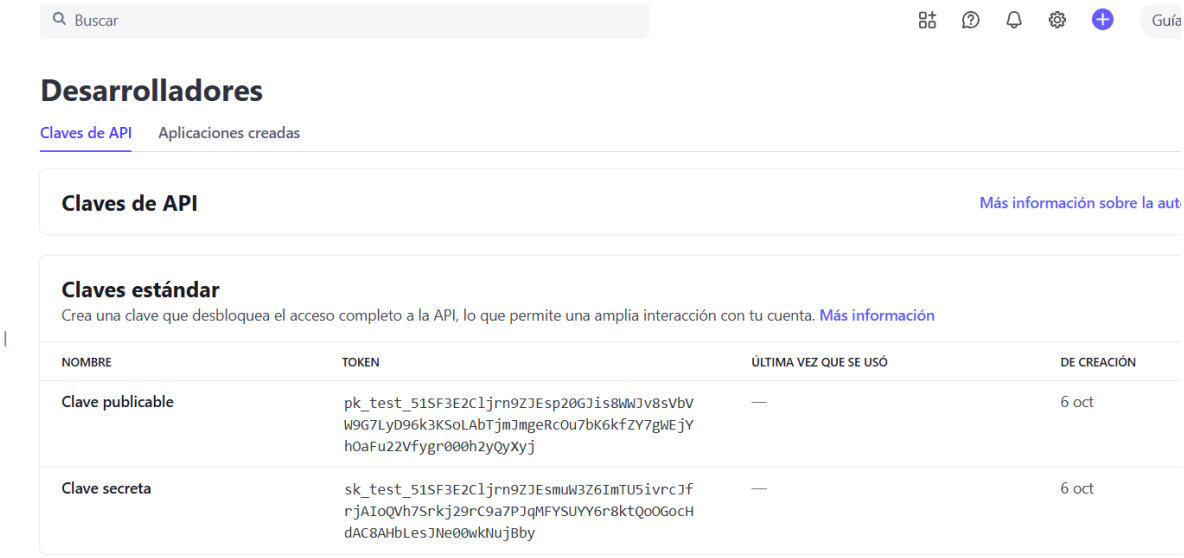
- Agregamos configuración para aceptar pagos en el sitio web, usando un servicio llamado stripe.

- Desde la Administración de Paquetes NuGet



- Crear una cuenta en stripe.com





- En SistemaInventario/appsettings.json agregamos la información relacionada.

```
 },

 "Stripe": {
  "SecretKey": "sk_test_5",
  "PublishableKey": "pk_test_5"
 }
}
```

- En SistemaInventario.Utilidades creamos una clase llamada StripeSettings.cs

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

namespace SistemaInventario.Utilidades
{
   public class StripeSettings
   {
      public string SecretKey { get; set; }
      public string PublishableKey { get; set; }
   }
}
```

- En SistemaInventario/Program.cs agregamos el servicio stripe

```
builder.Services.Configure<StripeSettings>(builder.Configuration.GetSection("Stripe"));

var app = builder.Build();
```

.

```
.
.
app.UseStaticFiles();

StripeConfiguration.ApiKey = builder.Configuration.GetSection("Stripe:SecretKey").Get<string>();
```

4. Configuracion Inicial antes del Metodo Proceder Post

- Modificamos SistemaInventario/appsettings.json

```
  "Stripe": {
    "SecretKey":
"sk_test_51SF3E2Cljrn9ZJEsmuW3Z6ImTU5ivrcJfrjAIoQVh7Srkj29rC9a7PJqMFYSUYY6r8ktQoOGocHdAC8AHbL
esJNe00wkNujBby",
    "PublishableKey":
"pk_test_51SF3E2Cljrn9ZJEsp20GJis8WWJv8sVbVW9G7LyD96k3KSoLAbTjmJmgeRcOu7bK6kfZY7gWEjYhOaFu
22Vfygr000h2yQyXyj"
  },

  "DomainUrls": {
    "WEB_URL": "https://localhost:7130/"
  }
}
```

- Modificamos SistemaInventario/Areas/Inventario/Controllers/CarroController.cs

```
private readonly IUnidadTrabajo _unidadTrabajo;
private string _webUrl;
.
.
.
public CarroController(IUnidadTrabajo unidadTrabajo, IConfiguration configuration)
{
    _unidadTrabajo = unidadTrabajo;
    _webUrl = configuration.GetValue<string>("DomainUrls:WEB_URL");
}
.
.
.
```

5. Metodo Proceder Post

- Modificamos SistemaInventario/Areas/Inventario/Controllers/CarroController.cs agregando el método Post Proceder

```
[HttpPost]
[ValidateAntiForgeryToken]
public async Task<IActionResult> Proceder(CarroCompraVM carroCompraVM)
{
    var claimIdentidad = (ClaimsIdentity)User.Identity;
```

```
    var claim = claimIdentidad.FindFirst(ClaimTypes.NameIdentifier);

    carroCompraVM.CarroCompraLista = await _unidadTrabajo.CarroCompra.ObtenerTodos(
                                c => c.UsuarioAplicacionId == claim.Value,
                                incluirPropiedades: "Producto");
    carroCompraVM.Compania = await _unidadTrabajo.Compania.ObtenerPrimero();
    carroCompraVM.Orden.TotalOrden = 0;
    carroCompraVM.Orden.UsuarioAplicacionId = claim.Value;
    carroCompraVM.Orden.FechaOrden = DateTime.Now;

    foreach (var lista in carroCompraVM.CarroCompraLista)
    {
        lista.Precio = lista.Producto.Precio;
        carroCompraVM.Orden.TotalOrden += (lista.Precio + lista.Cantidad);
    }
    // Controlar Stock
    foreach (var lista in carroCompraVM.CarroCompraLista)
    {
        var producto = await _unidadTrabajo.BodegaProducto.ObtenerPrimero(b => b.ProductoId ==
lista.ProductoId &&
                                            b.BodegaId == carroCompraVM.Compania.BodegaVentaId);
        if (lista.Cantidad > producto.Cantidad)
        {
            TempData[DS.Error] = "La Cantidad del Producto " + lista.Producto.Descripcion +
                        " Excede al Stock actual (" + producto.Cantidad + ")";
            return RedirectToAction("Index");
        }
    }
    carroCompraVM.Orden.EstadoOrden = DS.EstadoPendiente;
    carroCompraVM.Orden.EstadoPago = DS.PagoEstadoPendiente;
    await _unidadTrabajo.Orden.Agregar(carroCompraVM.Orden);
    await _unidadTrabajo.Guardar();
    // Grabar Detalle Orden
    foreach (var lista in carroCompraVM.CarroCompraLista)
    {
        OrdenDetalle ordenDetalle = new OrdenDetalle()
        {
            ProductoId = lista.ProductoId,
            OrdenId = carroCompraVM.Orden.Id,
            Precio = lista.Precio,
            Cantidad = lista.Cantidad
        };
        await _unidadTrabajo.OrdenDetalle.Agregar(ordenDetalle);
        await _unidadTrabajo.Guardar();
    }
    // Stripe

    return new RedirectToAction("OrdenConfirmacion", new { id = carroCompraVM.Orden.Id });

}
public async Task<IActionResult> OrdenConfirmacion(int id)
{
    return View(id);
```

```
}
```

6. Stripe en Proceder Post

- Modificamos el método Post Proceder de SistemaInventario/Areas/Inventario/Controllers/CarroController.cs

```
// Stripe
var usuario = await _unidadTrabajo.UsuarioAplicacion.ObtenerPrimero(u => u.Id == claim.Value);
var options = new SessionCreateOptions
{
   SuccessUrl = _webUrl + $"inventario/carro/OrdenConfirmacion?id={carroCompraVM.Orden.Id}",
   CancelUrl =  _webUrl + "inventario/carro/index",
   LineItems = new List<SessionLineItemOptions>(),
   Mode = "payment",
   CustomerEmail = usuario.Email
};
foreach (var lista in carroCompraVM.CarroCompraLista)
{
   var sessionLineItem = new SessionLineItemOptions
   {
      PriceData = new SessionLineItemPriceDataOptions()
      {
         UnitAmount = (long)(lista.Precio * 100),  // $20  => 200
         Currency = "usd",
         ProductData = new SessionLineItemPriceDataProductDataOptions
         {
            Name = lista.Producto.Descripcion
         }
      },
      Quantity = lista.Cantidad
   };
   options.LineItems.Add(sessionLineItem);
}
var service = new SessionService();
Session session = service.Create(options);
_unidadTrabajo.Orden.ActualizarPagoStripeId(carroCompraVM.Orden.Id, session.Id, session.PaymentIntentId);
await _unidadTrabajo.Guardar();
Response.Headers.Add("Location", session.Url);  // Redirecciona a Stripe
return new StatusCodeResult(303);

public async Task<IActionResult> OrdenConfirmacion(int id)
{
   var orden = await _unidadTrabajo.Orden.ObtenerPrimero(o => o.Id == id, incluirPropiedades: 
"UsuarioAplicacion");
   var service = new SessionService();
   Session session = service.Get(orden.SessionId);
   var carroCompra = await _unidadTrabajo.CarroCompra.ObtenerTodos(u => u.UsuarioAplicacionId == 
orden.UsuarioAplicacionId);
   if (session.PaymentStatus.ToLower() == "paid")
   {
      _unidadTrabajo.Orden.ActualizarPagoStripeId(id, session.Id, session.PaymentIntentId);
```

```
        _unidadTrabajo.Orden.ActualizarEstado(id, DS.EstadoAprobado, DS.PagoEstadoAprobado);
        await _unidadTrabajo.Guardar();

        // Disminuir Stock de la Bodega de Venta
        var compania = await _unidadTrabajo.Compania.ObtenerPrimero();
        foreach (var lista in carroCompra)
        {
            var bodegaProducto = new BodegaProducto();
            bodegaProducto = await _unidadTrabajo.BodegaProducto.ObtenerPrimero(b => b.ProductoId ==
lista.ProductoId
                                                && b.BodegaId == compania.BodegaVentaId);
            await _unidadTrabajo.KardexInventario.RegistrarKardex(bodegaProducto.Id, "Salida",
                                            "Venta- Orden# " + id,
                                            bodegaProducto.Cantidad,
                                            lista.Cantidad,
                                            orden.UsuarioAplicacionId);
            bodegaProducto.Cantidad-= lista.Cantidad;
            await _unidadTrabajo.Guardar();
        }

    }
    // Borramos el Carro de Compra y la Session del Carro de Compras

    List<CarroCompra> carroCompraLista = carroCompra.ToList();
    _unidadTrabajo.CarroCompra.RemoverRango(carroCompraLista);
    await _unidadTrabajo.Guardar();
    HttpContext.Session.SetInt32(DS.ssCarroCompras, 0);


    return View(id);
}
```

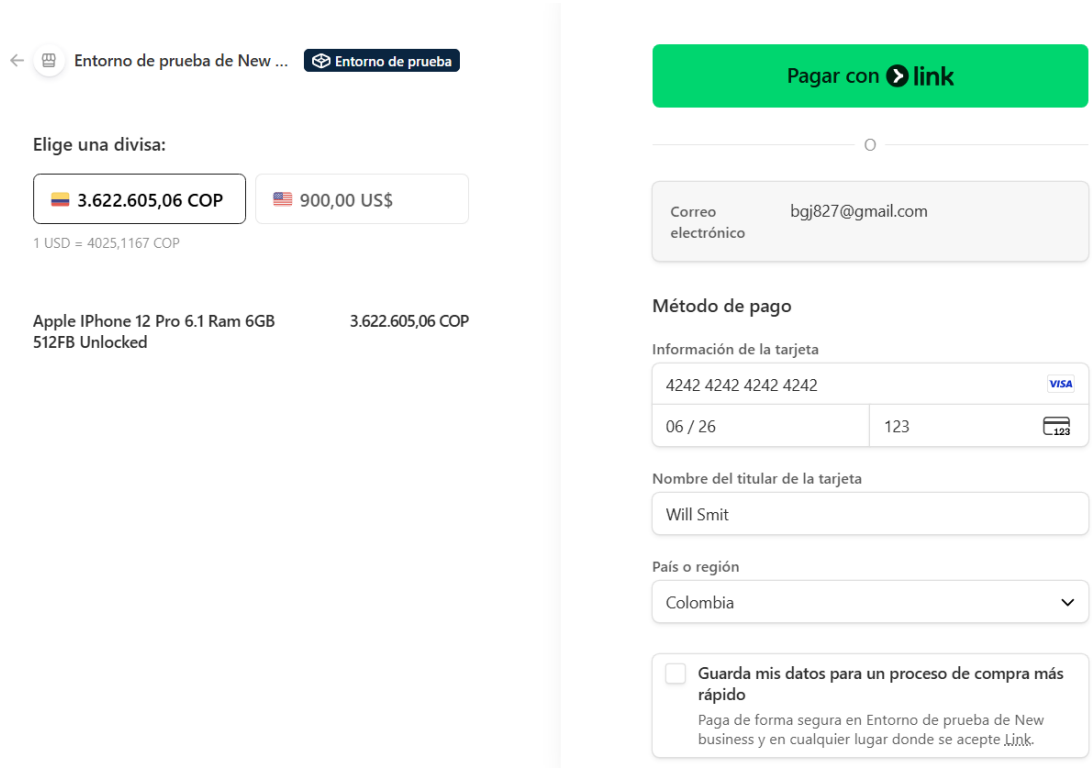
7. Disminuir Stock y Orden Confirmacion Vista

- Desde el método OrdenConfirmacion En SistemaInventario/Areas/Inventario/Controllers/CarroController.cs agregamos la vista de tipo Vista de Razor.

```
@model int
<div class="container row">
    <div class="col-12 text-center">
        <h1 class="text-primary text-center">Orden Procesada Correctamente</h1><br />
        <img src="~/imagenes/confirmacion.jpg" width="50%" />
    </div>
    <div class="col-12 text-center" style="color:maroon">
        <br />
        Gracias Por tu Orden! <br />
        Hemos recibido tu Orden y Nosotros te enviaremos un email pronto!
        <br />
        Tu numero de Orden es : @Model

        <div class="text-center">
```

```
        <a asp-area="Inventario" asp-controller="Home" asp-action="Index" class="btn btn-success">Salir</a>
    </div>

</div>

</div>
```

- Ejecutamos la aplicación para verificar el funcionamiento de Stripe con el botón Realizar Pedido



Orden Procesada Correctamente

Gracias Por tu Orden!
Hemos recibido tu Orden y Nosotros te enviaremos un email pronto!
Tu numero de Orden es : 2

Kardex Por Producto

Desde: 30/09/2025 **Hasta:** 5/10/2025

Producto: Apple IPhone 12 Pro 6.1 Ram 6GB 512FB Unlocked

Nueva Consulta | Imprimir | Salir

Bodega Principal						Saldo		
Fecha	Tipo	Detalle	Stock Anterior	Entrada	Salida	Stock	Costo	Total
4/10/2025	Entrada	Inventario Inicial	0	10	--	10	600	6000
5/10/2025	Salida	Venta - Orden# 2	10	--	1	9	600	5400

Orden Administración

1. Orden Controller - Index_ API

- En SistemaInventario/Areas/Admin/Controllers Creamos el controlador de tipo Controlador de MVC en blanco llamado OrdenController.cs

```
using Microsoft.AspNetCore.Authorization;
using Microsoft.AspNetCore.Mvc;
```

```
using SistemaInventario.AccesoDatos.Repositorio.IRepositorio;

namespace SistemaInventario.Areas.Admin.Controllers
{
    [Area("Admin")]
    [Authorize]
    public class OrdenController : Controller
    {
        private readonly IUnidadTrabajo _unidadTrabajo;

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

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

        #region
        [HttpGet]
        public async Task<IActionResult> ObtenerOrdenLista(string estado)
        {
            var todos = await _unidadTrabajo.Orden.ObtenerTodos(incluirPropiedades: "UsuarioAplicacion");
            return Json(new { data = todos });
        }
        #endregion

    }
}
```

2. Orden Index Vista Parte 1

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

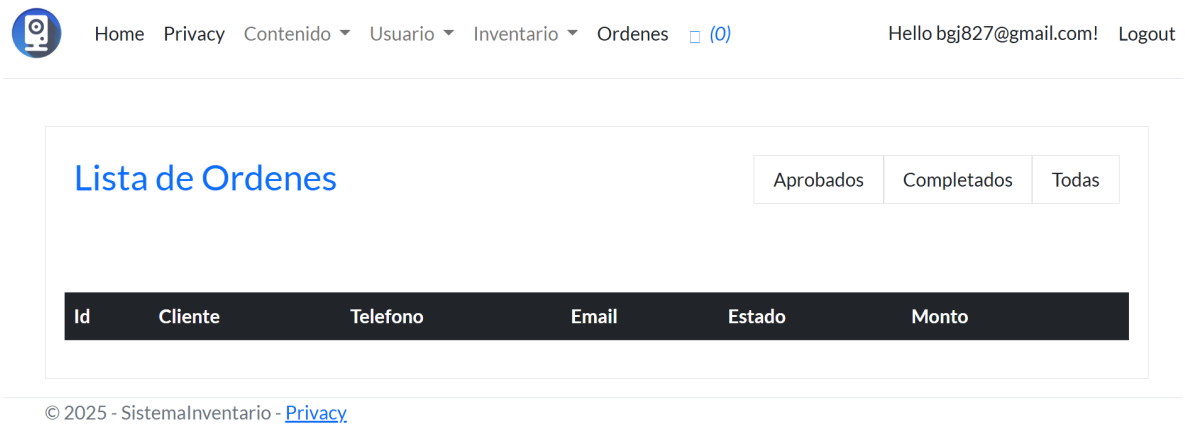
```
@{
    ViewData["Title"] = "Index";
    Layout = "~/Views/Shared/_Layout.cshtml";
}

<br />
<div class="border p-3">
    <div class="d-lg-flex justify-content-between mb-3">
        <div class="p-2">
            <h2 class="text-primary">Lista de Ordenes</h2>
        </div>
        <div class="p-2">
            <ul class="list-group list-group-horizontal-sm">
```

```
            <a      style="text-decoration:none;"      asp-controller="Orden"      asp-action="Index"      asp-route-
estado="aprobado">
            <li class="list-group-item">Aprobados</li>
            </a>
            <a      style="text-decoration:none;"      asp-controller="Orden"      asp-action="Index"      asp-route-
estado="completado">
            <li class="list-group-item">Completados</li>
            </a>
            <a      style="text-decoration:none;"      asp-controller="Orden"      asp-action="Index"      asp-route-
estado="todas">
            <li class="list-group-item">Todas</li>
            </a>
          </ul>
      </div>
    </div>
    <br /><br />
    <table id="tblDatos" class="table table-responsive table-hover display nowrap" style="width:100%">
      <thead class="table-dark">
        <tr>
          <th>Id</th>
          <th>Cliente</th>
          <th>Telefono</th>
          <th>Email</th>
          <th>Estado</th>
          <th>Monto</th>
          <th></th>
        </tr>
      </thead>
    </table>

</div>

@section Scripts {
    <script src="~/js/orden.js"></script>
}
```

- Modificamos SistemaInventario/Views/Shared/_Layout.cshtml para Orden

```
<li class="nav-item">
    <a class="nav-link text-dark" asp-area="Admin" asp-controller="Orden" asp-action="Index">Ordenes</a>
</li>

@if (httpContextAccessor.HttpContext.Session.GetInt32(DS.ssCarroCompras) != null)
{
.
.
.
```

- Ejecutamos la aplicación para verificar el funcionamiento de Orden



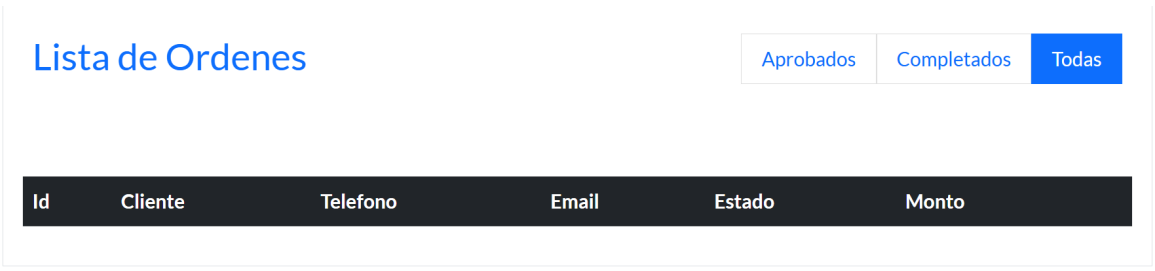
3. Orden Index Vista Parte 2

- Modificamos la vista SistemaInventario/Areas/Admin/Views/Orden/Index.cshtml

```
@{
    ViewData["Title"] = "Index";
    Layout = "~/Views/Shared/_Layout.cshtml";

    var estado = Context.Request.Query["estado"];
    var aprobado = "text-primary";
    var completado = "text-primary";
    var todas = "text-primary";

    switch (estado)
    {
        case "aprobado":
            aprobado = "active text-white";
            break;
        case "completado":
            completado = "active text-white";
            break;
        default:
            todas = "active text-white";
            break;
    }
}

<br />
<div class="border p-3">
    <div class="d-lg-flex justify-content-between mb-3">
        <div class="p-2">
            <h2 class="text-primary">Lista de Ordenes</h2>
        </div>
        <div class="p-2">
            <ul class="list-group list-group-horizontal-sm">
                <a style="text-decoration:none;" asp-controller="Orden" asp-action="Index" asp-route-estado="aprobado">
```

```
            <li class="list-group-item @aprobado">Aprobados</li>
            </a>
            <a      style="text-decoration:none;"      asp-controller="Orden"      asp-action="Index"      asp-route-
estado="completado">
            <li class="list-group-item @completado">Completados</li>
            </a>
            <a      style="text-decoration:none;"      asp-controller="Orden"      asp-action="Index"      asp-route-
estado="todas">
            <li class="list-group-item @todas">Todas</li>
            </a>
          </ul>
        </div>
    </div>
    <br /><br />
    <table id="tblDatos" class="table table-responsive table-hover display nowrap" style="width:100%">
        <thead class="table-dark">
            <tr>
                <th>Id</th>
                <th>Cliente</th>
                <th>Telefono</th>
                <th>Email</th>
                <th>Estado</th>
                <th>Monto</th>
                <th></th>
            </tr>
        </thead>
    </table>

</div>

@section Scripts {
    <script src="~/js/orden.js"></script>
}
```

- Ejecutamos la aplicación para verificar el funcionamiento de Orden



4. Modificacion Orden Controller

- Modificamos SistemaInventario/Areas/Admin/Controllers/OrdenController.cs

```
#region
[HttpGet]
public async Task<IActionResult> ObtenerOrdenLista(string estado)
{
   var claimIdentidad = (ClaimsIdentity)User.Identity;
   var claim = claimIdentidad.FindFirst(ClaimTypes.NameIdentifier);
   IEnumerable<Orden> todos;
   if (User.IsInRole(DS.Role_Admin))  // Validar el Rol del Usuario
   {
      todos = await _unidadTrabajo.Orden.ObtenerTodos(incluirPropiedades: "UsuarioAplicacion");
   }
   else
   {
      todos = await _unidadTrabajo.Orden.ObtenerTodos(o => o.UsuarioAplicacionId == claim.Value,
incluirPropiedades: "UsuarioAplicacion");
   }
   // Validar el Estado
   switch (estado)
   {
      case "aprobado":
         todos = todos.Where(o => o.EstadoOrden == DS.EstadoAprobado);
         break;
      case "completado":
         todos = todos.Where(o => o.EstadoOrden == DS.EstadoEnviado);
         break;
      default:
         break;
   }

   return Json(new { data = todos });
}
#endregion
```

5. Orden JavaScript API

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

```
var datatable;

$(document).ready(function () {
   var url = window.location.search;
   if (url.includes("aprobado")) {
      loadDataTable("ObtenerOrdenLista?estado=aprobado");
   }
   else {
      if (url.includes("completado")) {
         loadDataTable("ObtenerOrdenLista?estado=completado");
      }
      else {
         loadDataTable("ObtenerOrdenLista?estado=todas");
      }
```

```
    }

});

function loadDataTable(url) {
    datatable = $('#tblDatos').DataTable({
        "language": {
            "lengthMenu": "Mostrar _MENU_ Registros Por Pagina",
            "zeroRecords": "Ningun Registro",
            "info": "Mostrar page _PAGE_ de _PAGES_",
            "infoEmpty": "no hay registros",
            "infoFiltered": "(filtered from _MAX_ total registros)",
            "search": "Buscar",
            "paginate": {
                "first": "Primero",
                "last": "Último",
                "next": "Siguiente",
                "previous": "Anterior"
            }
        },
        "ajax": {
            "url": "/Admin/Orden/" + url
        },
        "columns": [
            { "data": "id" },
            { "data": "nombresCliente" },
            { "data": "telefono" },
            { "data": "usuarioAplicacion.email" },
            { "data": "estadoOrden" },
            {
                "data": "totalOrden", "className": "text-end",
                "render": function (data) {
                    var d = data.toFixed(2).replace(/\d(?=(\d{3})+\.)/g, '$&,');
                    return d;
                }
            },
            {
                "data": "id",
                "render": function (data) {
                    return `
                        <div class="text-center">
                            <a href="/Admin/Orden/Detalle/${data}" class="btn btn-success text-white"
style="cursor:pointer">
                                <i class="bi bi-ticket-detailed"></i>
                            </a>
                        </div>
                        `;
                }
            }
        ]
    });
}
```

6. Orden Detalle View Model y Metodo de Accion Get

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

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

namespace SistemaInventario.Modelos.ViewModels
{
   public class OrdenDetalleVM
   {
      public Compania Compania { get; set; }

      public Orden Orden { get; set; }

      public IEnumerable<OrdenDetalle> OrdenDetalleLista { get; set; }
   }
}
```

- Modificamos SistemaInventario/Areas/Admin/Controllers/OrdenController.cs

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

   [BindProperty]
   public OrdenDetalleVM ordenDetalleVM { get; set; }
.
.
.
public IActionResult Index()
{
   return View();
}

public async Task<IActionResult> Detalle(int id)
{
   ordenDetalleVM = new OrdenDetalleVM()
   {
      Orden = await _unidadTrabajo.Orden.ObtenerPrimero(o => o.Id == id, incluirPropiedades:
"UsuarioAplicacion"),
      OrdenDetalleLista = await _unidadTrabajo.OrdenDetalle.ObtenerTodos(d => d.OrdenId == id,
                                                incluirPropiedades: "Producto")
   };
   return View(ordenDetalleVM);
}
```

7. Orden Detalle - Vista

- Desde el método Detalle de SistemaInventario/Areas/Admin/Controllers/OrdenController.cs agregamos la vista tipo Vista de Razor

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

<form method="post">
  <input hidden asp-for="Orden.Id" />
  <br />
  <div class="container">
    <div class="card">
      <div class="card-header bg-dark text-light ml-0 row ">
        <div class="col-12 d-none d-md-block col-md-6 pb-1">
          <i class="fas fa-shopping-cart"></i>   Resumen de la Orden
        </div>
        <div class="col-12 col-md-4 offset-md-2 text-right">
          <a asp-area="Admin" asp-controller="Orden" asp-action="Index" class="btn btn-outline-info form-
control btn-sm">Regresar</a>
        </div>
      </div>
      <div class="card-body">
        <div class="container rounded p-2">
          <div class="row">
            <div class="col-12 col-lg-6 pb-4">
              <div class="row">
                <h4 class="d-flex justify-content-between align-items-center mb-3">
                  <span class="text-secondary">Detalles de Envío:</span>
                </h4>
              </div>
              <div class="row my-1">
                <div class="col-3">Nombres</div>
                <div class="col-9">
                  @if (User.IsInRole(DS.Role_Admin))
                  {
                    <input asp-for="Orden.NombresCliente" class="form-control" />
                  }
                  else
                  {
                    <input asp-for="Orden.NombresCliente" readonly type="text" class="form-control" />
                  }

                </div>
              </div>
              <div class="row my-1">
                <div class="col-3">telefono</div>
                <div class="col-9">
                  @if (User.IsInRole(DS.Role_Admin))
                  {
```

```
            <input asp-for="Orden.Telefono" class="form-control" />
        }
        else
        {
            <input asp-for="Orden.Telefono" readonly type="text" class="form-control" />
        }
    </div>
</div>
<div class="row my-1">
    <div class="col-3">Direccion</div>
    <div class="col-9">
        @if (User.IsInRole(DS.Role_Admin))
        {
            <input asp-for="Orden.Direccion" class="form-control" />
        }
        else
        {
            <input asp-for="Orden.Direccion" readonly type="text" class="form-control" />
        }
    </div>
</div>
<div class="row my-1">
    <div class="col-3">Ciudad</div>
    <div class="col-9">
        @if (User.IsInRole(DS.Role_Admin))
        {
            <input asp-for="Orden.Ciudad" class="form-control" />
        }
        else
        {
            <input asp-for="Orden.Ciudad" readonly type="text" class="form-control" />
        }
    </div>
</div>
<div class="row my-1">
    <div class="col-3">Pais</div>
    <div class="col-9">
        @if (User.IsInRole(DS.Role_Admin))
        {
            <input asp-for="Orden.Pais" class="form-control" />
        }
        else
        {
            <input asp-for="Orden.Pais" readonly type="text" class="form-control" />
        }
    </div>
</div>
<div class="row my-1">
    <div class="col-3">Email</div>
    <div class="col-9">
        @if (User.IsInRole(DS.Role_Admin))
        {
            <input asp-for="Orden.UsuarioAplicacion.Email" class="form-control" />
```

```
                    }
                    else
                    {
                        <input asp-for="Orden.UsuarioAplicacion.Email" readonly type="text" class="form-control" />
                    }
                </div>
            </div>
            <div class="row my-1">
                <div class="col-3">Fecha Orden</div>
                <div class="col-9">
                    <input value="@Model.Orden.FechaOrden.ToShortDateString()" readonly type="text" class="form-control" />
                </div>
            </div>
            <div class="row my-1">
                <div class="col-3">Carrier</div>
                <div class="col-9">
                    @if (User.IsInRole(DS.Role_Admin))
                    {
                        <input asp-for="Orden.Carrier" class="form-control" id="carrierId"/>
                    }
                    else
                    {
                        <input asp-for="Orden.Carrier" readonly type="text" class="form-control" />
                    }
                </div>
            </div>
            <div class="row my-1">
                <div class="col-3">Numero Envio</div>
                <div class="col-9">
                    @if (User.IsInRole(DS.Role_Admin))
                    {
                        <input asp-for="Orden.NumeroEnvio" class="form-control" id="numeroEnvioId"/>
                    }
                    else
                    {
                        <input asp-for="Orden.NumeroEnvio" readonly type="text" class="form-control" />
                    }
                </div>
            </div>
            <div class="row my-1">
                <div class="col-3">Fecha Envio</div>
                <div class="col-9">
                    @if (Model.Orden.FechaEnvio.Year < 1900)
                    {
                        <input id="fechaEnvio" type="text" readonly class="form-control" />
                    }
                    else
                    {
                        <input value="@Model.Orden.FechaEnvio.ToShortDateString()" id="fechaEnvio" type="text" readonly class="form-control" />
                    }
```

```
                    </div>
                </div>
                @if (User.IsInRole(DS.Role_Admin))
                {
                    <div class="row my-1">
                        <div class="col-3">Transaccion ID</div>
                        <div class="col-9">
                            <input asp-for="Orden.TransaccionId" type="text" readonly class="form-control" />
                        </div>
                    </div>
                    <div class="row my-1">

                        <div class="col-3">Fecha de Pago</div>
                        <div class="col-9">
                            <input value="@Model.Orden.FechaPago.ToShortDateString()" readonly
id="paymentDate" type="text" class="form-control" />
                        </div>
                    </div>
                    <div class="row my-1">
                        <div class="col-3">Estado Pago</div>
                        <div class="col-9">
                            <input asp-for="Orden.EstadoPago" type="text" readonly class="form-control" />
                        </div>
                    </div>
                }

            </div>
            <div class="col-12 col-lg-5 offset-lg-1">
                <h4 class="d-flex justify-content-between align-items-center mb-3">
                    <span class="text-secondary">Detalle de la Orden</span>
                </h4>
                <ul class="list-group mb-3">
                    @foreach (var item in Model.OrdenDetalleLista)
                    {
                        <li class="list-group-item d-flex justify-content-between p-2">
                            <div class="row container">
                                <div class="col-8">
                                    <input type="hidden" asp-for="@item.Id" />
                                    <h6 class="my-0 text-secondary">@item.Producto.Descripcion</h6>
                                    <small class="text-muted">Precio : @item.Precio</small><br />
                                    <small class="text-muted">Cantidad : @item.Cantidad</small>
                                </div>
                                <div class="col-4 text-end">
                                    <p class="text-success">@(String.Format("{0:C}", item.Precio * item.Cantidad))</p>
                                </div>
                            </div>
                        </li>
                    }

                    <li class="list-group-item bg-info">
                        <div class="row container">
                            <div class="col-8">
```

```
                        <h6 class="text-white">TOTAL </h6>
                    </div>
                    <div class="col-4 text-end">
                        <h6 class="text-white">@(String.Format("{0:C}", Model.Orden.TotalOrden))</h6>
                    </div>
                </div>
            </li>
        </ul>
    </div>
</div>
</form>
```

- Ejecutamos la aplicación para verificar el funcionamiento de OrdenDetalle.

8. Botones para Procesar Ordenes

- Modificamos SistemaInventario/Areas/Admin/Views/Orden/Detalle.cshtml

```
.
.
.
                </ul>

                @if (User.IsInRole(DS.Role_Admin))
                {
                    @if (Model.Orden.EstadoOrden == DS.EstadoAprobado)
                    {
                        <a asp-action="Procesar" asp-route-id="@Model.Orden.Id" class="btn btn-primary form-
control">
                            Empezar a Procesar
                        </a>
                    }
                    @if (Model.Orden.EstadoOrden == DS.EstadoEnProceso)
                    {
                        <input type="submit" value="Enviar Orden" onclick="return validarIngreso()"
                            class="btn btn-success form-control" formaction="/Admin/Orden/EnviarOrden"
                            formmethod="post" />
                    }
                }
                else
                {
                    <label class="btn btn-primary form-control">@Model.Orden.EstadoOrden</label>
                }

            </div>
        </div>
```

```
            </div>
          </div>
        </div>
      </div>
</form>
```

9. Metodos Procesar_ Enviar Orden y Cancelar Orden

- En SistemaInventario/Areas/Admin/Controllers/OrdenController.cs creamos métodos Procesar y EnviarOrden

```
[Authorize(Roles = DS.Role_Admin)]
public async Task<IActionResult> Procesar(int id)
{
   var orden = await _unidadTrabajo.Orden.ObtenerPrimero(o => o.Id == id);
   orden.EstadoOrden = DS.EstadoEnProceso;
   await _unidadTrabajo.Guardar();
   TempData[DS.Exitosa] = "Orden cambiada a Estado en Proceso";
   return RedirectToAction("Detalle", new { id = id });
}

[HttpPost]
[Authorize(Roles = DS.Role_Admin)]
public async Task<IActionResult> EnviarOrden(OrdenDetalleVM ordenDetalleVM)
{
   var orden = await _unidadTrabajo.Orden.ObtenerPrimero(o => o.Id == ordenDetalleVM.Orden.Id);
   orden.EstadoOrden = DS.EstadoEnviado;
   orden.Carrier = ordenDetalleVM.Orden.Carrier;
   orden.NumeroEnvio = ordenDetalleVM.Orden.NumeroEnvio;
   orden.FechaEnvio = DateTime.Now;
   await _unidadTrabajo.Guardar();
   TempData[DS.Exitosa] = "Orden cambiada a Estado Enviado";
   return RedirectToAction("Detalle", new { id = ordenDetalleVM.Orden.Id });
}

#region
[HttpGet]
```

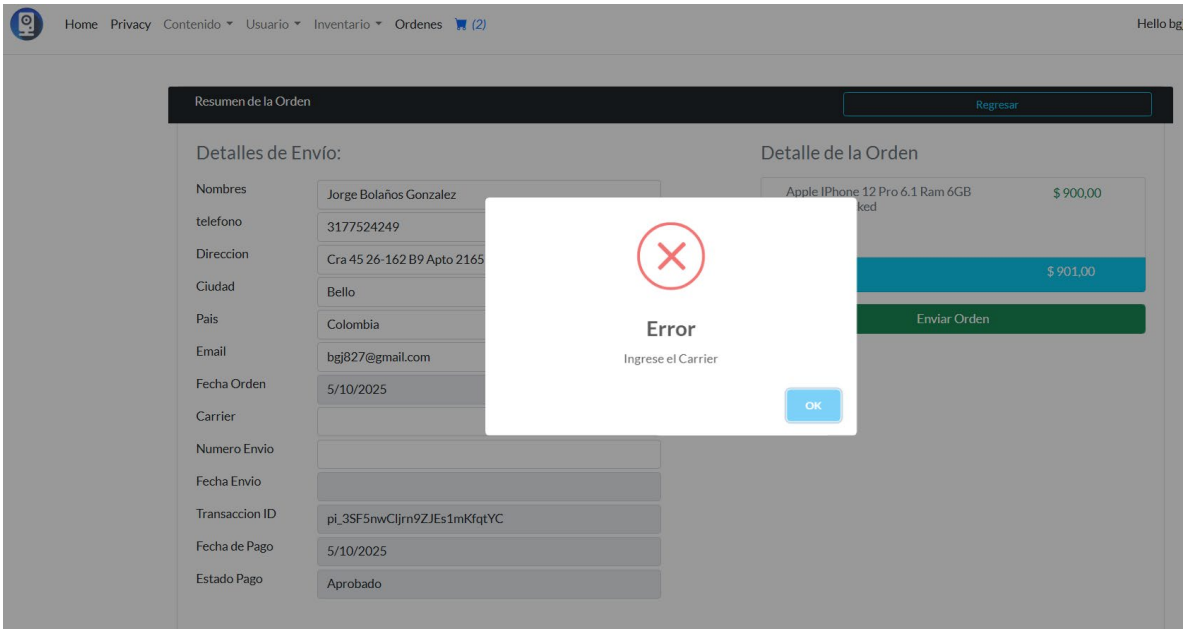
10. Procesar y Enviar Orden

- En SistemaInventario/Areas/Admin/Views/Orden/Detalle.cshtml Adicionamos al final el método validarIngreso()

```
@section Scripts {
   <script>

      function validarIngreso() {
         let carrierId = document.getElementById("carrierId").value;
         let numeroEnvioId = document.getElementById("numeroEnvioId").value;

         if (carrierId == '') {
```

```
            swal("Error", "Ingrese el Carrier", "error")
            return false;
        }
        if (numeroEnvioId == '') {
            swal("Error", "Ingrese el Numero de Envio", "error")
            return false;
        }

        return true;
    }

  </script>
}
```

- Ejecutamos la aplicación para verificar el funcionamiento



Resumen de la Orden

Regresar

Detalles de Envío:

Nombres	Jorge Bolaños Gonzalez
telefono	3177524249
Direccion	Cra 45 26-162 B9 Apto 2165 Puerta madera Bello
Ciudad	Bello
Pais	Colombia
Email	bgj827@gmail.com
Fecha Orden	5/10/2025
Carrier	Fedex
Numero Envio	454545454545
Fecha Envio	9/10/2025
Transaccion ID	pi_3SF5nwCljrn9ZJEs1mKfqtYC
Fecha de Pago	5/10/2025
Estado Pago	Aprobado

Detalle de la Orden

Apple IPhone 12 Pro 6.1 Ram 6GB 512FB Unlocked Precio : 900 Cantidad : 1	$ 900,00
TOTAL	$ 901,00