Universidad ORT Uruguay

Facultad de Ingeniería

Escuela de Tecnología

OBLIGATORIO

Programación 3

Mario Gonzalez - 241618

M3A

Docente: Liliana Pino

Analista En Tecnología De La Información

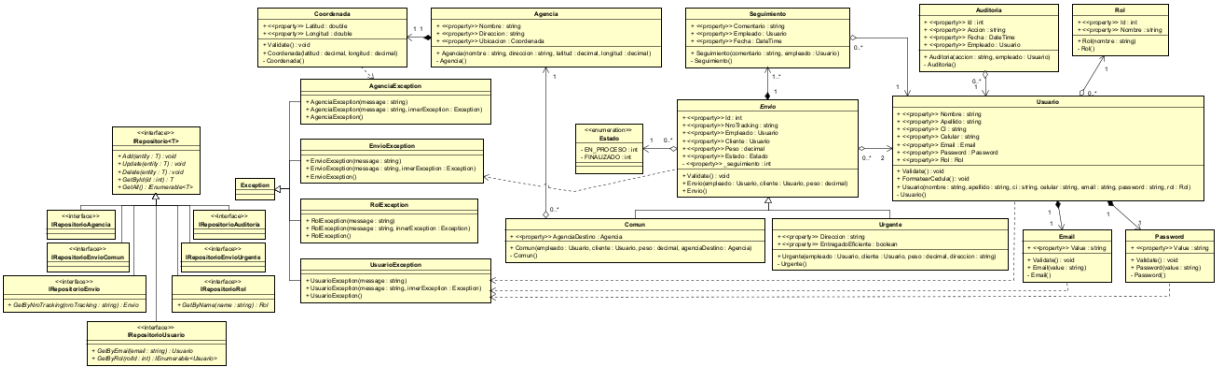
19/05/2025

Contenido

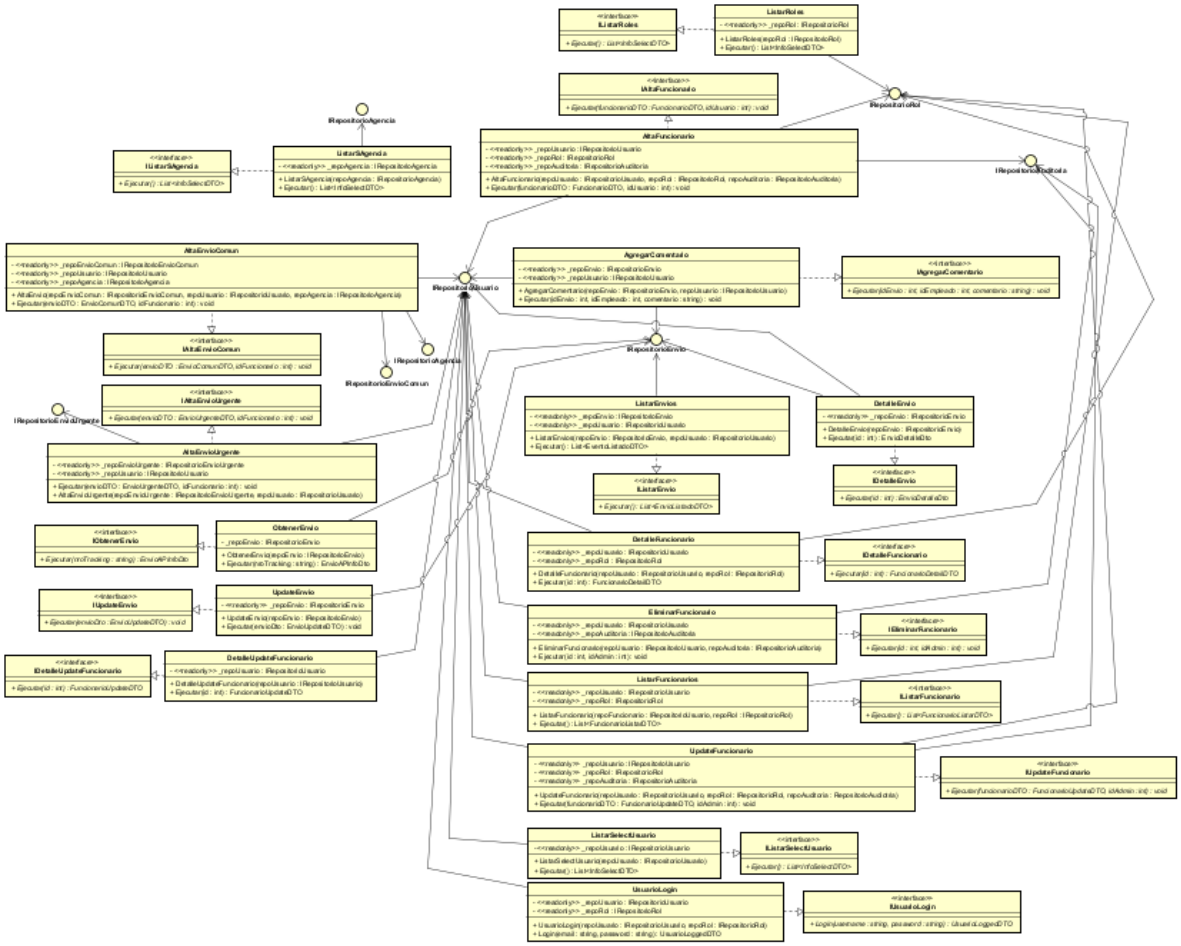
Diagramas

De clase

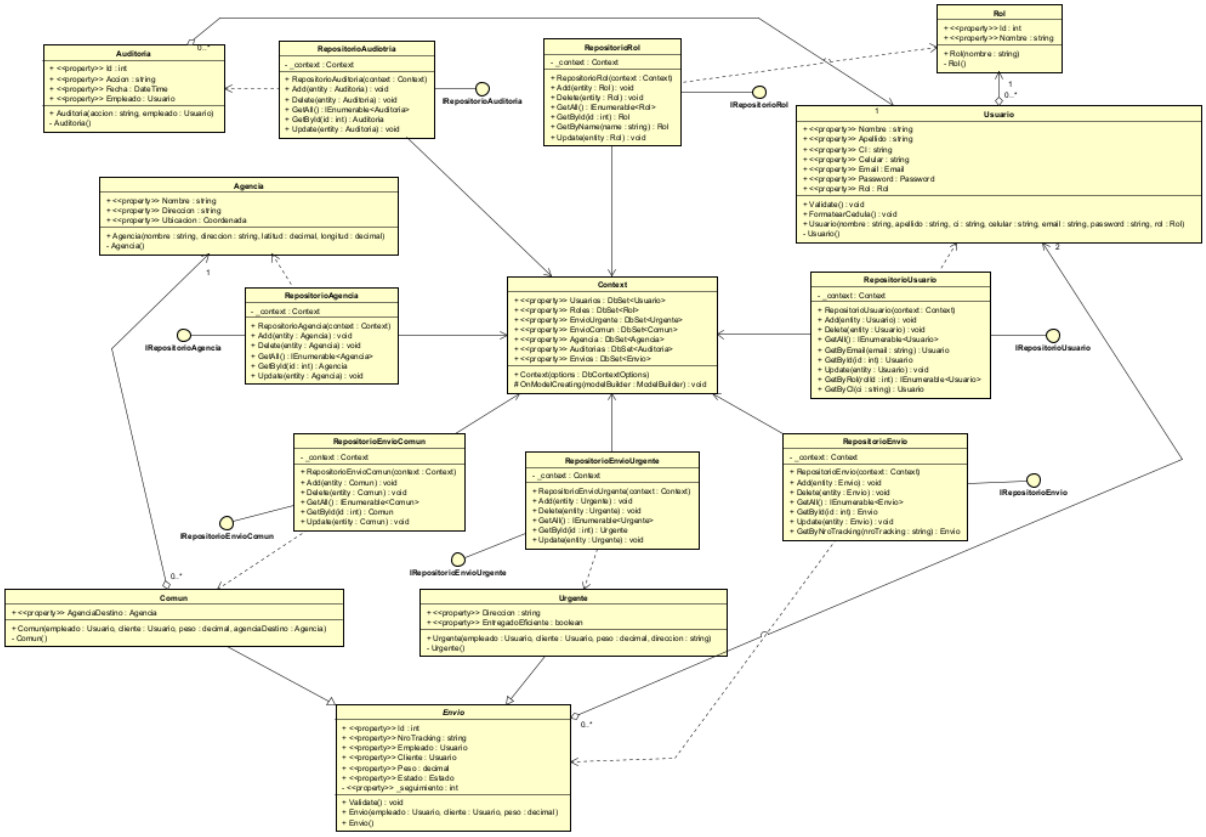
LogicaNegocio



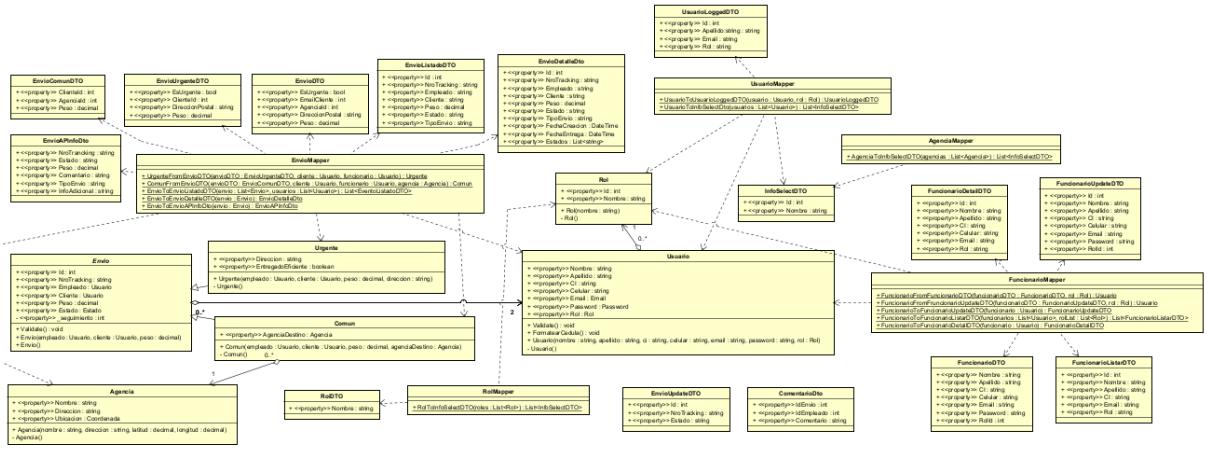
LogicaAplicacion



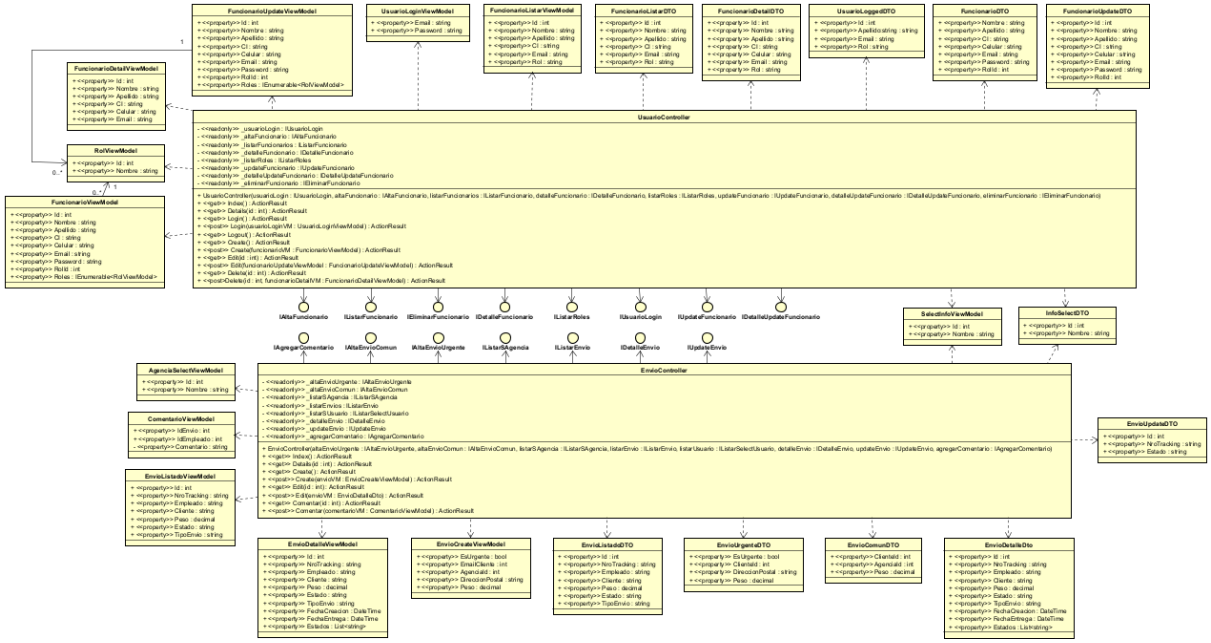
LogicaAccesoDatos



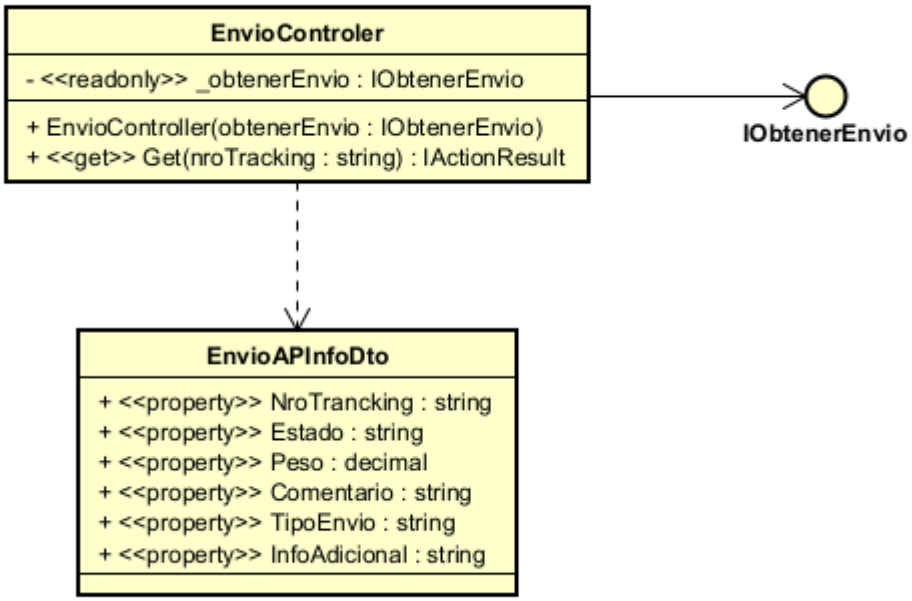
Compartido



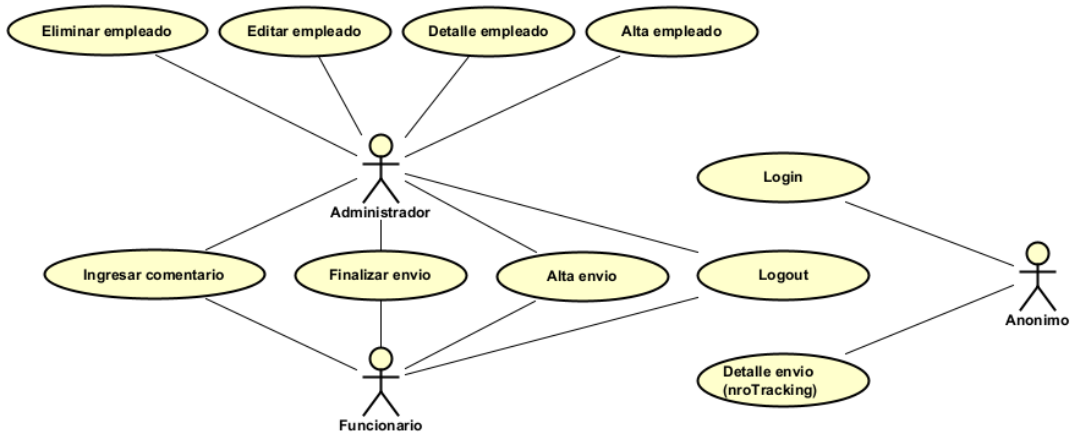
MVC



WebApi


EnvioControler
- <<readonly>> _obtenerEnvio : IObtenerEnvio
+ EnvioController(obtenerEnvio : IObtenerEnvio)
+ <<get>> Get(nroTracking : string) : IActionResult
IObtenerEnvio
EnvioAPInfoDto
+ <<property>> NroTrancking : string
+ <<property>> Estado : string
+ <<property>> Peso : decimal
+ <<property>> Comentario : string
+ <<property>> TipoEnvio : string
+ <<property>> InfoAdicional : string

Caso de uso



Caso de uso narrativo

Identificador: CU-01	**Nombre: Alta de envío.**
Autor:	Mario Gonzalez
Fecha:	19/05/2025
Descripción: Permite dar de alta un nuevo envío.	
Actor: Usuario administrador o funcionario.	
Precondiciones: El actor debe estar logueado en el sistema y debe datos de las agencias.	
Poscondiciones: El envío se crea con éxito y se da de alta en la base de datos.	
Flujo normal: 1. El actor ingresa en crear envío. 2. El sistema le permite ingresar los datos correspondientes según el envío. 3. El actor ingresa los datos y los envía. 4. El sistema valida los datos ingresados, deja valores por defecto, marca el estado del envío en proceso y lo guarda.	
Flujo/s Alternativo/s: 3. El actor no envía los datos a. El sistema descarta los datos ingresados. 4. Los datos ingresados no son correctos a. Muestra un mensaje avisando donde esta el dato incorrecto.	
Flujo/s Excepcionales/s: Se interrumpe la comunicación con el servidor. Los datos no son guardados.	

Identificador: CU-02	**Nombre: Finalizar un envío.**
Autor:	Mario Gonzalez
Fecha:	19/05/2025
Descripción: Permite finalizar un envío.	
Actor: Usuario administrador o funcionario.	
Precondiciones: El actor debe estar logueado en el sistema y debe haber envíos disponibles.	
Poscondiciones: El envío se finaliza con éxito.	
Flujo normal: 1. El actor ingresa a un envío. 2. El sistema le muestra el envío a finalizar. 3. El actor selecciona finalizar envío. 4. El sistema da como finalizado el envío, agregando fecha de entrega.	
Flujo/s Alternativo/s: 2. El sistema no muestra el envío. a. El envío solicitado no existe. 3. El envío ya está finalizado a. El sistema no permite cambiar el estado del envío.	
Flujo/s Excepcionales/s: Se interrumpe la comunicación con el servidor. Los datos no son guardados.	

Link

GitHub: https://github.com/Mariocgf/Obligatorio-P3

Prompt ChatGPT: https://chatgpt.com/share/682b9eb7-2f28-8011-b653-b4172e02cccc

Código

```
**********************************
Archivo: Context.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaAccesoDatos\Context.cs
**********************************
using LogicaNegocio.Entidades;
using Microsoft.EntityFrameworkCore;
namespace LogicaAccesoDatos
{
   public class Context : DbContext
   {
      public DbSet<Usuario> Usuarios { get; set; }
      public DbSet<Rol> Roles { get; set; }
      public DbSet<Urgente> EnvioUrgente { get; set; }
      public DbSet<Comun> EnvioComun { get; set; }
      public DbSet<Agencia> Agencias { get; set; }
      public DbSet<Auditoria> Auditorias { get; set; }
      public DbSet<Envio> Envios { get; set; }
      public Context(DbContextOptions options) : base(options)
      {
      }
      protected override void OnModelCreating(ModelBuilder modelBuilder)
      {
         modelBuilder.Entity<Usuario>().HasOne(u =>
u.Rol).WithMany().OnDelete(DeleteBehavior.NoAction);
         modelBuilder.Entity<Envio>().HasOne(e =>
e.Cliente).WithMany().OnDelete(DeleteBehavior.NoAction);
         modelBuilder.Entity<Envio>().HasOne(e =>
e.Empleado).WithMany().OnDelete(DeleteBehavior.NoAction);
         modelBuilder.Entity<Envio>().UseTptMappingStrategy();
         modelBuilder.Entity<Envio>().Property(e => e.Peso).HasPrecision(5, 2);
         modelBuilder.Entity<Auditoria>().HasOne(a =>
a.Empleado).WithMany().OnDelete(DeleteBehavior.NoAction);
         modelBuilder.Entity<Seguimiento>().HasOne(s =>
s.Empleado).WithMany().OnDelete(DeleteBehavior.NoAction);
         base.OnModelCreating(modelBuilder);
      }
   }
}
**********************************
Archivo: InfoSelectDTO.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\Compartido\DTOs\InfoSelectDTO.cs
**********************************
using System;
using System.Collections.Generic;
using System.Linq;
```

```
using System.Text;
using System.Threading.Tasks;
namespace Compartido.DTOs
{
   public class InfoSelectDTO
   {
      public int Id { get; set; }
      public string Nombre { get; set; }
   }
}
```

****

Archivo: RolDTO.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\Compartido\DTOs\RolDTO.cs

****

```
namespace Compartido.DTOs
{
   public class RolDTO
   {
      public string Nombre { get; set; }
   }
}
```

****

Archivo: UsuarioLoggedDTO.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\Compartido\DTOs\UsuarioLoggedDTO.cs

****

```
namespace Compartido.DTOs
{
   public class UsuarioLoggedDTO
   {
      public int Id { get; set; }
      public string Nombre { get; set; }
      public string Apellido { get; set; }
      public string Email { get; set; }
      public string Rol { get; set; }
   }
}
```

****

Archivo: AgenciaMapper.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\Compartido\Mappers\AgenciaMapper.cs

****

```
using Compartido.DTOs;
using Compartido.DTOs.Agencia;
using LogicaNegocio.Entidades;
namespace Compartido.Mappers
{
```

```
   public class AgenciaMapper
   {
      public static List<InfoSelectDTO> AgenciaToInfoSelectDTO(List<Agencia> agencias)
      {
         return agencias.Select(a => new InfoSelectDTO()
         {
            Id = a.Id,
            Nombre = a.Nombre,
         }).ToList();
      }
   }
}
**********
Archivo: EnvioMapper.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\Compartido\Mappers\EnvioMapper.cs
**********
using Compartido.DTOs.Envio;
using LogicaNegocio.Entidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace Compartido.Mappers
{
   public class EnvioMapper
   {
      public static Urgente UrgenteFromEnvioDTO(EnvioUrgenteDTO envioDTO, Usuario cliente,
Usuario funcionario)
      {
         return new Urgente(funcionario, cliente, envioDTO.Peso, envioDTO.DireccionPostal);
      }
      public static Comun ComunFromEnvioDTO(EnvioComunDTO envioDTO, Usuario cliente, Usuario
funcionario, Agencia agencia)
      {
         return new Comun(funcionario, cliente, envioDTO.Peso, agencia);
      }
      public static List<EnvioListadoDTO> EnvioTOEnvioListadoDTO(List<Envio> envio, List<Usuario>
usuarios)
      {
         return envio.Select(e => new EnvioListadoDTO
         {
            Id = e.Id,
            NroTracking = e.NroTracking,
            Empleado = $"{usuarios.FirstOrDefault(u => u.Id == e.Empleado.Id).Nombre}
{usuarios.FirstOrDefault(u => u.Id == e.Empleado.Id).Apellido}",
```

```
            Cliente = $"{usuarios.FirstOrDefault(u => u.Id == e.Cliente.Id).Nombre}
{usuarios.FirstOrDefault(u => u.Id == e.Cliente.Id).Apellido}",
            Peso = e.Peso,
            Estado = e.Estado.ToString(),
            TipoEnvio = e is Comun ? "Comun" : "Urgente",
        }).ToList();
    }
    public static EnvioDetalleDto EnvioTOEnvioDetalleDTO(Envio envio)
    {
        List<string> estados = [.. Enum.GetNames(typeof(Estados))];

        return new EnvioDetalleDto
        {
            Id = envio.Id,
            NroTracking = envio.NroTracking,
            Empleado = $"{envio.Empleado.Nombre} {envio.Empleado.Apellido}",
            Cliente = $"{envio.Cliente.Nombre} {envio.Cliente.Apellido}",
            Peso = envio.Peso,
            Estado = envio.Estado.ToString(),
            TipoEnvio = envio is Comun ? "Comun" : "Urgente",
            FechaCreacion = envio.FechaCreacion,
            FechaEntrega = envio.FechaEntrega,
            Estados = estados
        };
    }

    public static EnvioAPInfoDto EnvioToEnvioAPInfoDto(Envio envio)
    {
        EnvioAPInfoDto envioAPInfoDto = new EnvioAPInfoDto
        {
            NroTracking = envio.NroTracking,
            Peso = envio.Peso,
            Estado = envio.Estado.ToString(),
            Comentario = envio.ListaSeguimiento.LastOrDefault()?.Comentario
        };
        if (envio is Comun comun)
        {
            envioAPInfoDto.TipoEnvio = "Comun";
            envioAPInfoDto.InfoAdicional = comun.AgenciaDestino.Nombre;
        }
        else
        {
            envioAPInfoDto.TipoEnvio = "Urgente";
            envioAPInfoDto.InfoAdicional = ((Urgente)envio).Direccion;
        }
        return envioAPInfoDto;
    }
}
```

```
}
**********
Archivo: FuncionarioMapper.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\Compartido\Mappers\FuncionarioMapper.cs
**********
using LogicaNegocio.Entidades;
using Compartido.DTOs.Funcionario;
namespace Compartido.Mappers
{
   public class FuncionarioMapper
   {
      public static Usuario FuncionarioFromFuncionarioDTO(FuncionarioDTO funcionarioDTO, Rol rol)
      {
         return new Usuario
            (funcionarioDTO.Nombre,
             funcionarioDTO.Apellido,
             funcionarioDTO.CI,
             funcionarioDTO.Celular,
             funcionarioDTO.Email,
             funcionarioDTO.Password,
             rol);
      }
      public static Usuario FuncionarioFromFuncionarioUpdateDTO(FuncionarioUpdateDTO
funcionarioDTO, Rol rol)
      {
         return new Usuario
            (funcionarioDTO.Nombre,
             funcionarioDTO.Apellido,
             funcionarioDTO.CI,
             funcionarioDTO.Celular,
             funcionarioDTO.Email,
             funcionarioDTO.Password,
             rol);
      }
      public static FuncionarioUpdateDTO FuncionarioToFuncionarioUpdateDTO(Usuario funcionario)
      {
         return new FuncionarioUpdateDTO()
         {
            Id = funcionario.Id,
            Nombre = funcionario.Nombre,
            Apellido = funcionario.Apellido,
            CI = funcionario.CI,
            Celular = funcionario.Celular,
            Email = funcionario.Email.Value,
            Password = funcionario.Password.Value,
            RolId = funcionario.RolId
         };
```

```
    }
    public static List<FuncionarioListarDTO> FuncionarioToFuncionarioListarDTO(List<Usuario>
funcionarios, List<Rol> rolList)
    {
      return funcionarios.Select(f => new FuncionarioListarDTO()
      {
        Id = f.Id,
        Nombre = f.Nombre,
        Apellido = f.Apellido,
        CI = f.CI,
        Email = f.Email.Value,
        Rol = f.Rol.Nombre,
      }).ToList();
    }
    public static FuncionarioDetailDTO FuncionarioToFuncionarioDetailDTO(Usuario funcionario)
    {
      return new FuncionarioDetailDTO()
      {
        Id = funcionario.Id,
        Nombre = funcionario.Nombre,
        Apellido = funcionario.Apellido,
        CI = funcionario.CI,
        Celular = funcionario.Celular,
        Email = funcionario.Email.Value
      };
    }
    //public static Usuario FuncionarioFromFuncionarioDTO(FuncionarioDTO funcionarioDTO)
    //{
    //   return new Usuario()
    //   {
    //      Nombre = funcionarioDTO.Nombre,
    //      Apellido = funcionarioDTO.Apellido,
    //      CI = funcionarioDTO.CI,
    //      Celular = funcionarioDTO.Celular,
    //      Email = funcionarioDTO.Email,
    //      Password = funcionarioDTO.Password
    //   };
    //}
  }
}
**********
Archivo: RolMapper.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\Compartido\Mappers\RolMapper.cs
**********
using Compartido.DTOs;
using LogicaNegocio.Entidades;
using System;
```

```
using System.Collections.Generic;
using System.Linq;
using System.Security.Cryptography.X509Certificates;
using System.Text;
using System.Threading.Tasks;
namespace Compartido.Mappers
{
   public class RolMapper
   {
      public static List<InfoSelectDTO> RolToRolDetallesDTO(List<Rol> roles)
      {
         return roles.Select(rol => new InfoSelectDTO()
         {
            Id = rol.Id,
            Nombre = rol.Nombre
         }).ToList();
      }
   }
}
**********
Archivo: UsuarioMapper.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\Compartido\Mappers\UsuarioMapper.cs
**********
using Compartido.DTOs;
using LogicaNegocio.Entidades;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace Compartido.Mappers
{
   public class UsuarioMapper
   {
      public static UsuarioLoggedDTO UsuarioTOUsurioLoggedDTO(Usuario usuario, Rol rol)
      {
         return new UsuarioLoggedDTO()
         {
            Id = usuario.Id,
            Nombre = usuario.Nombre,
            Apellido = usuario.Apellido,
            Email = usuario.Email.Value,
            Rol = rol.Nombre
         };
      }
      public static List<InfoSelectDTO> UsuarioToInfoSelectDto(List<Usuario> usuarios)
      {
         return [..usuarios.Select(u => new InfoSelectDTO()
```

```
        {
            Id = u.Id,
            Nombre = u.Email.Value
        })];
    }
  }
}
**********
Archivo: 20250519115132_init.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaAccesoDatos\Migrations\20250519115132_init.cs
**********
using System;
using Microsoft.EntityFrameworkCore.Migrations;
#nullable disable
namespace LogicaAccesoDatos.Migrations
{
  /// <inheritdoc />
  public partial class init : Migration
  {
    /// <inheritdoc />
    protected override void Up(MigrationBuilder migrationBuilder)
    {
      migrationBuilder.CreateTable(
        name: "Agencias",
        columns: table => new
        {
          Id = table.Column<int>(type: "int", nullable: false)
            .Annotation("SqlServer:Identity", "1, 1"),
          Nombre = table.Column<string>(type: "nvarchar(max)", nullable: false),
          Direccion = table.Column<string>(type: "nvarchar(max)", nullable: false),
          Coordenada_Latitud = table.Column<decimal>(type: "decimal(18,2)", nullable: false),
          Coordenada_Longitud = table.Column<decimal>(type: "decimal(18,2)", nullable: false)
        },
        constraints: table =>
        {
          table.PrimaryKey("PK_Agencias", x => x.Id);
        });
      migrationBuilder.CreateTable(
        name: "Roles",
        columns: table => new
        {
          Id = table.Column<int>(type: "int", nullable: false)
            .Annotation("SqlServer:Identity", "1, 1"),
          Nombre = table.Column<string>(type: "nvarchar(max)", nullable: false)
        },
        constraints: table =>
        {
```

```
            table.PrimaryKey("PK_Roles", x => x.Id);
        });
    migrationBuilder.CreateTable(
        name: "Usuarios",
        columns: table => new
        {
            Id = table.Column<int>(type: "int", nullable: false)
                .Annotation("SqlServer:Identity", "1, 1"),
            Nombre = table.Column<string>(type: "nvarchar(max)", nullable: false),
            Apellido = table.Column<string>(type: "nvarchar(max)", nullable: false),
            CI = table.Column<string>(type: "nvarchar(max)", nullable: false),
            Celular = table.Column<string>(type: "nvarchar(max)", nullable: false),
            RolId = table.Column<int>(type: "int", nullable: false),
            Email_Value = table.Column<string>(type: "nvarchar(max)", nullable: false),
            Password_Value = table.Column<string>(type: "nvarchar(max)", nullable: false)
        },
        constraints: table =>
        {
            table.PrimaryKey("PK_Usuarios", x => x.Id);
            table.ForeignKey(
                name: "FK_Usuarios_Roles_RolId",
                column: x => x.RolId,
                principalTable: "Roles",
                principalColumn: "Id");
        });
    migrationBuilder.CreateTable(
        name: "Auditorias",
        columns: table => new
        {
            Id = table.Column<int>(type: "int", nullable: false)
                .Annotation("SqlServer:Identity", "1, 1"),
            Accion = table.Column<string>(type: "nvarchar(max)", nullable: false),
            Fecha = table.Column<DateTime>(type: "datetime2", nullable: false),
            EmpleadoId = table.Column<int>(type: "int", nullable: false)
        },
        constraints: table =>
        {
            table.PrimaryKey("PK_Auditorias", x => x.Id);
            table.ForeignKey(
                name: "FK_Auditorias_Usuarios_EmpleadoId",
                column: x => x.EmpleadoId,
                principalTable: "Usuarios",
                principalColumn: "Id");
        });
    migrationBuilder.CreateTable(
        name: "Envio",
        columns: table => new
        {
```

```
            Id = table.Column<int>(type: "int", nullable: false)
                .Annotation("SqlServer:Identity", "1, 1"),
            NroTracking = table.Column<string>(type: "nvarchar(max)", nullable: false),
            EmpleadoId = table.Column<int>(type: "int", nullable: false),
            ClienteId = table.Column<int>(type: "int", nullable: false),
            Peso = table.Column<decimal>(type: "decimal(5,2)", precision: 5, scale: 2, nullable: false),
            Estado = table.Column<int>(type: "int", nullable: false),
            FechaCreacion = table.Column<DateTime>(type: "datetime2", nullable: false),
            FechaEntrega = table.Column<DateTime>(type: "datetime2", nullable: false)
        },
        constraints: table =>
        {
            table.PrimaryKey("PK_Envio", x => x.Id);
            table.ForeignKey(
                name: "FK_Envio_Usuarios_ClienteId",
                column: x => x.ClienteId,
                principalTable: "Usuarios",
                principalColumn: "Id");
            table.ForeignKey(
                name: "FK_Envio_Usuarios_EmpleadoId",
                column: x => x.EmpleadoId,
                principalTable: "Usuarios",
                principalColumn: "Id");
        });
    migrationBuilder.CreateTable(
        name: "EnvioComun",
        columns: table => new
        {
            Id = table.Column<int>(type: "int", nullable: false),
            AgenciaDestinoId = table.Column<int>(type: "int", nullable: false)
        },
        constraints: table =>
        {
            table.PrimaryKey("PK_EnvioComun", x => x.Id);
            table.ForeignKey(
                name: "FK_EnvioComun_Agencias_AgenciaDestinoId",
                column: x => x.AgenciaDestinoId,
                principalTable: "Agencias",
                principalColumn: "Id",
                onDelete: ReferentialAction.Cascade);
            table.ForeignKey(
                name: "FK_EnvioComun_Envio_Id",
                column: x => x.Id,
                principalTable: "Envio",
                principalColumn: "Id",
                onDelete: ReferentialAction.Cascade);
        });
    migrationBuilder.CreateTable(
```

```
        name: "EnvioUrgente",
        columns: table => new
        {
            Id = table.Column<int>(type: "int", nullable: false),
            Direccion = table.Column<string>(type: "nvarchar(max)", nullable: false),
            EntregadoEficiente = table.Column<bool>(type: "bit", nullable: false)
        },
        constraints: table =>
        {
            table.PrimaryKey("PK_EnvioUrgente", x => x.Id);
            table.ForeignKey(
                name: "FK_EnvioUrgente_Envio_Id",
                column: x => x.Id,
                principalTable: "Envio",
                principalColumn: "Id",
                onDelete: ReferentialAction.Cascade);
        });
    migrationBuilder.CreateTable(
        name: "Seguimiento",
        columns: table => new
        {
            Id = table.Column<int>(type: "int", nullable: false)
                .Annotation("SqlServer:Identity", "1, 1"),
            Comentario = table.Column<string>(type: "nvarchar(max)", nullable: false),
            EmpleadoId = table.Column<int>(type: "int", nullable: false),
            Fecha = table.Column<DateTime>(type: "datetime2", nullable: false),
            EnvioId = table.Column<int>(type: "int", nullable: true)
        },
        constraints: table =>
        {
            table.PrimaryKey("PK_Seguimiento", x => x.Id);
            table.ForeignKey(
                name: "FK_Seguimiento_Envio_EnvioId",
                column: x => x.EnvioId,
                principalTable: "Envio",
                principalColumn: "Id");
            table.ForeignKey(
                name: "FK_Seguimiento_Usuarios_EmpleadoId",
                column: x => x.EmpleadoId,
                principalTable: "Usuarios",
                principalColumn: "Id");
        });
    migrationBuilder.CreateIndex(
        name: "IX_Auditorias_EmpleadoId",
        table: "Auditorias",
        column: "EmpleadoId");
    migrationBuilder.CreateIndex(
        name: "IX_Envio_ClienteId",
```

```
            table: "Envio",
            column: "ClienteId");
        migrationBuilder.CreateIndex(
            name: "IX_Envio_EmpleadoId",
            table: "Envio",
            column: "EmpleadoId");
        migrationBuilder.CreateIndex(
            name: "IX_EnvioComun_AgenciaDestinoId",
            table: "EnvioComun",
            column: "AgenciaDestinoId");
        migrationBuilder.CreateIndex(
            name: "IX_Seguimiento_EmpleadoId",
            table: "Seguimiento",
            column: "EmpleadoId");
        migrationBuilder.CreateIndex(
            name: "IX_Seguimiento_EnvioId",
            table: "Seguimiento",
            column: "EnvioId");
        migrationBuilder.CreateIndex(
            name: "IX_Usuarios_RolId",
            table: "Usuarios",
            column: "RolId");
    }
    /// <inheritdoc />
    protected override void Down(MigrationBuilder migrationBuilder)
    {
        migrationBuilder.DropTable(
            name: "Auditorias");
        migrationBuilder.DropTable(
            name: "EnvioComun");
        migrationBuilder.DropTable(
            name: "EnvioUrgente");
        migrationBuilder.DropTable(
            name: "Seguimiento");
        migrationBuilder.DropTable(
            name: "Agencias");
        migrationBuilder.DropTable(
            name: "Envio");
        migrationBuilder.DropTable(
            name: "Usuarios");
        migrationBuilder.DropTable(
            name: "Roles");
    }
  }
}
```

****

Archivo: 20250519115132_init.Designer.cs

Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaAccesoDatos\Migrations\20250519115132_init.Designer.cs
****

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

#nullable disable

namespace LogicaAccesoDatos.Migrations
{
   [DbContext(typeof(Context))]
   [Migration("20250519115132_init")]
   partial class init
   {
      /// <inheritdoc />
      protected override void BuildTargetModel(ModelBuilder modelBuilder)
      {
#pragma warning disable 612, 618
         modelBuilder
            .HasAnnotation("ProductVersion", "8.0.10")
            .HasAnnotation("Relational:MaxIdentifierLength", 128);

         SqlServerModelBuilderExtensions.UseIdentityColumns(modelBuilder);

         modelBuilder.Entity("LogicaNegocio.Entidades.Agencia", b =>
            {
               b.Property<int>("Id")
                  .ValueGeneratedOnAdd()
                  .HasColumnType("int");

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

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

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

```
            b.ComplexProperty<Dictionary<string, object>>("Coordenada",
"LogicaNegocio.Entidades.Agencia.Coordenada#Coordenada", b1 =>
                {
                    b1.IsRequired();

                    b1.Property<decimal>("Latitud")
                        .HasColumnType("decimal(18,2)");

                    b1.Property<decimal>("Longitud")
                        .HasColumnType("decimal(18,2)");
                });

            b.HasKey("Id");

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

        modelBuilder.Entity("LogicaNegocio.Entidades.Auditoria", b =>
        {
            b.Property<int>("Id")
                .ValueGeneratedOnAdd()
                .HasColumnType("int");

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

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

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

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

            b.HasKey("Id");

            b.HasIndex("EmpleadoId");

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

        modelBuilder.Entity("LogicaNegocio.Entidades.Envio", b =>
        {
            b.Property<int>("Id")
                .ValueGeneratedOnAdd()
                .HasColumnType("int");
```

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

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

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

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

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

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

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

            b.Property<decimal>("Peso")
                .HasPrecision(5, 2)
                .HasColumnType("decimal(5,2)");

            b.HasKey("Id");

            b.HasIndex("ClienteId");

            b.HasIndex("EmpleadoId");

            b.ToTable("Envio");

            b.UseTptMappingStrategy();
        });

    modelBuilder.Entity("LogicaNegocio.Entidades.Rol", b =>
        {
            b.Property<int>("Id")
                .ValueGeneratedOnAdd()
                .HasColumnType("int");

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

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

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

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

    modelBuilder.Entity("LogicaNegocio.Entidades.Seguimiento", b =>
        {
            b.Property<int>("Id")
                .ValueGeneratedOnAdd()
                .HasColumnType("int");

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

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

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

            b.Property<int?>("EnvioId")
                .HasColumnType("int");

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

            b.HasKey("Id");

            b.HasIndex("EmpleadoId");

            b.HasIndex("EnvioId");

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

    modelBuilder.Entity("LogicaNegocio.Entidades.Usuario", b =>
        {
            b.Property<int>("Id")
                .ValueGeneratedOnAdd()
                .HasColumnType("int");

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

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

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

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

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

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

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

            b.ComplexProperty<Dictionary<string, object>>("Email",
"LogicaNegocio.Entidades.Usuario.Email#Email", b1 =>
                {
                    b1.IsRequired();

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

            b.ComplexProperty<Dictionary<string, object>>("Password",
"LogicaNegocio.Entidades.Usuario.Password#Password", b1 =>
                {
                    b1.IsRequired();

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

            b.HasKey("Id");

            b.HasIndex("RolId");

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

        modelBuilder.Entity("LogicaNegocio.Entidades.Comun", b =>
            {
                b.HasBaseType("LogicaNegocio.Entidades.Envio");

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

```
        b.HasIndex("AgenciaDestinoId");

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

modelBuilder.Entity("LogicaNegocio.Entidades.Urgente", b =>
    {
        b.HasBaseType("LogicaNegocio.Entidades.Envio");

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

        b.Property<bool>("EntregadoEficiente")
            .HasColumnType("bit");

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

modelBuilder.Entity("LogicaNegocio.Entidades.Auditoria", b =>
    {
        b.HasOne("LogicaNegocio.Entidades.Usuario", "Empleado")
            .WithMany()
            .HasForeignKey("EmpleadoId")
            .OnDelete(DeleteBehavior.NoAction)
            .IsRequired();

        b.Navigation("Empleado");
    });

modelBuilder.Entity("LogicaNegocio.Entidades.Envio", b =>
    {
        b.HasOne("LogicaNegocio.Entidades.Usuario", "Cliente")
            .WithMany()
            .HasForeignKey("ClienteId")
            .OnDelete(DeleteBehavior.NoAction)
            .IsRequired();

        b.HasOne("LogicaNegocio.Entidades.Usuario", "Empleado")
            .WithMany()
            .HasForeignKey("EmpleadoId")
            .OnDelete(DeleteBehavior.NoAction)
            .IsRequired();

        b.Navigation("Cliente");

        b.Navigation("Empleado");
    });
```

```
modelBuilder.Entity("LogicaNegocio.Entidades.Seguimiento", b =>
  {
    b.HasOne("LogicaNegocio.Entidades.Usuario", "Empleado")
      .WithMany()
      .HasForeignKey("EmpleadoId")
      .OnDelete(DeleteBehavior.NoAction)
      .IsRequired();

    b.HasOne("LogicaNegocio.Entidades.Envio", null)
      .WithMany("ListaSeguimiento")
      .HasForeignKey("EnvioId");

    b.Navigation("Empleado");
  });

modelBuilder.Entity("LogicaNegocio.Entidades.Usuario", b =>
  {
    b.HasOne("LogicaNegocio.Entidades.Rol", "Rol")
      .WithMany()
      .HasForeignKey("RolId")
      .OnDelete(DeleteBehavior.NoAction)
      .IsRequired();

    b.Navigation("Rol");
  });

modelBuilder.Entity("LogicaNegocio.Entidades.Comun", b =>
  {
    b.HasOne("LogicaNegocio.Entidades.Agencia", "AgenciaDestino")
      .WithMany()
      .HasForeignKey("AgenciaDestinoId")
      .OnDelete(DeleteBehavior.Cascade)
      .IsRequired();

    b.HasOne("LogicaNegocio.Entidades.Envio", null)
      .WithOne()
      .HasForeignKey("LogicaNegocio.Entidades.Comun", "Id")
      .OnDelete(DeleteBehavior.Cascade)
      .IsRequired();

    b.Navigation("AgenciaDestino");
  });

modelBuilder.Entity("LogicaNegocio.Entidades.Urgente", b =>
  {
    b.HasOne("LogicaNegocio.Entidades.Envio", null)
      .WithOne()
```

```
                .HasForeignKey("LogicaNegocio.Entidades.Urgente", "Id")
                .OnDelete(DeleteBehavior.Cascade)
                .IsRequired();
            });

        modelBuilder.Entity("LogicaNegocio.Entidades.Envio", b =>
            {
                b.Navigation("ListaSeguimiento");
            });
#pragma warning restore 612, 618
      }
   }
}
```

****

Archivo: ContextModelSnapshot.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaAccesoDatos\Migrations\ContextModelSnapshot.cs

****

```
// <auto-generated />
using System;
using System.Collections.Generic;
using LogicaAccesoDatos;
using Microsoft.EntityFrameworkCore;
using Microsoft.EntityFrameworkCore.Infrastructure;
using Microsoft.EntityFrameworkCore.Metadata;
using Microsoft.EntityFrameworkCore.Storage.ValueConversion;

#nullable disable

namespace LogicaAccesoDatos.Migrations
{
   [DbContext(typeof(Context))]
   partial class ContextModelSnapshot : ModelSnapshot
   {
      protected override void BuildModel(ModelBuilder modelBuilder)
      {
#pragma warning disable 612, 618
         modelBuilder
            .HasAnnotation("ProductVersion", "8.0.10")
            .HasAnnotation("Relational:MaxIdentifierLength", 128);

         SqlServerModelBuilderExtensions.UseIdentityColumns(modelBuilder);

         modelBuilder.Entity("LogicaNegocio.Entidades.Agencia", b =>
            {
               b.Property<int>("Id")
                  .ValueGeneratedOnAdd()
```

```
                .HasColumnType("int");

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

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

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

            b.ComplexProperty<Dictionary<string, object>>("Coordenada",
"LogicaNegocio.Entidades.Agencia.Coordenada#Coordenada", b1 =>
                {
                    b1.IsRequired();

                    b1.Property<decimal>("Latitud")
                        .HasColumnType("decimal(18,2)");

                    b1.Property<decimal>("Longitud")
                        .HasColumnType("decimal(18,2)");
                });

            b.HasKey("Id");

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

        modelBuilder.Entity("LogicaNegocio.Entidades.Auditoria", b =>
            {
                b.Property<int>("Id")
                    .ValueGeneratedOnAdd()
                    .HasColumnType("int");

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

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

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

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

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

```
        b.HasIndex("EmpleadoId");

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

modelBuilder.Entity("LogicaNegocio.Entidades.Envio", b =>
    {
        b.Property<int>("Id")
            .ValueGeneratedOnAdd()
            .HasColumnType("int");

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

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

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

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

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

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

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

        b.Property<decimal>("Peso")
            .HasPrecision(5, 2)
            .HasColumnType("decimal(5,2)");

        b.HasKey("Id");

        b.HasIndex("ClienteId");

        b.HasIndex("EmpleadoId");

        b.ToTable("Envio");

        b.UseTptMappingStrategy();
    });
```

```
modelBuilder.Entity("LogicaNegocio.Entidades.Rol", b =>
    {
        b.Property<int>("Id")
            .ValueGeneratedOnAdd()
            .HasColumnType("int");

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

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

        b.HasKey("Id");

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

modelBuilder.Entity("LogicaNegocio.Entidades.Seguimiento", b =>
    {
        b.Property<int>("Id")
            .ValueGeneratedOnAdd()
            .HasColumnType("int");

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

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

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

        b.Property<int?>("EnvioId")
            .HasColumnType("int");

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

        b.HasKey("Id");

        b.HasIndex("EmpleadoId");

        b.HasIndex("EnvioId");

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

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

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

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

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

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

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

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

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

            b.ComplexProperty<Dictionary<string, object>>("Email",
"LogicaNegocio.Entidades.Usuario.Email#Email", b1 =>
                {
                    b1.IsRequired();

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

            b.ComplexProperty<Dictionary<string, object>>("Password",
"LogicaNegocio.Entidades.Usuario.Password#Password", b1 =>
                {
                    b1.IsRequired();

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

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

```
        b.HasIndex("RolId");

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

modelBuilder.Entity("LogicaNegocio.Entidades.Comun", b =>
    {
        b.HasBaseType("LogicaNegocio.Entidades.Envio");

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

        b.HasIndex("AgenciaDestinoId");

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

modelBuilder.Entity("LogicaNegocio.Entidades.Urgente", b =>
    {
        b.HasBaseType("LogicaNegocio.Entidades.Envio");

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

        b.Property<bool>("EntregadoEficiente")
            .HasColumnType("bit");

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

modelBuilder.Entity("LogicaNegocio.Entidades.Auditoria", b =>
    {
        b.HasOne("LogicaNegocio.Entidades.Usuario", "Empleado")
            .WithMany()
            .HasForeignKey("EmpleadoId")
            .OnDelete(DeleteBehavior.NoAction)
            .IsRequired();

        b.Navigation("Empleado");
    });

modelBuilder.Entity("LogicaNegocio.Entidades.Envio", b =>
    {
        b.HasOne("LogicaNegocio.Entidades.Usuario", "Cliente")
            .WithMany()
            .HasForeignKey("ClienteId")
            .OnDelete(DeleteBehavior.NoAction)
```

```
            .IsRequired();

        b.HasOne("LogicaNegocio.Entidades.Usuario", "Empleado")
            .WithMany()
            .HasForeignKey("EmpleadoId")
            .OnDelete(DeleteBehavior.NoAction)
            .IsRequired();

        b.Navigation("Cliente");

        b.Navigation("Empleado");
    });

modelBuilder.Entity("LogicaNegocio.Entidades.Seguimiento", b =>
    {
        b.HasOne("LogicaNegocio.Entidades.Usuario", "Empleado")
            .WithMany()
            .HasForeignKey("EmpleadoId")
            .OnDelete(DeleteBehavior.NoAction)
            .IsRequired();

        b.HasOne("LogicaNegocio.Entidades.Envio", null)
            .WithMany("ListaSeguimiento")
            .HasForeignKey("EnvioId");

        b.Navigation("Empleado");
    });

modelBuilder.Entity("LogicaNegocio.Entidades.Usuario", b =>
    {
        b.HasOne("LogicaNegocio.Entidades.Rol", "Rol")
            .WithMany()
            .HasForeignKey("RolId")
            .OnDelete(DeleteBehavior.NoAction)
            .IsRequired();

        b.Navigation("Rol");
    });

modelBuilder.Entity("LogicaNegocio.Entidades.Comun", b =>
    {
        b.HasOne("LogicaNegocio.Entidades.Agencia", "AgenciaDestino")
            .WithMany()
            .HasForeignKey("AgenciaDestinoId")
            .OnDelete(DeleteBehavior.Cascade)
            .IsRequired();

        b.HasOne("LogicaNegocio.Entidades.Envio", null)
```

```
                .WithOne()
                .HasForeignKey("LogicaNegocio.Entidades.Comun", "Id")
                .OnDelete(DeleteBehavior.Cascade)
                .IsRequired();

            b.Navigation("AgenciaDestino");
          });

        modelBuilder.Entity("LogicaNegocio.Entidades.Urgente", b =>
          {
            b.HasOne("LogicaNegocio.Entidades.Envio", null)
              .WithOne()
              .HasForeignKey("LogicaNegocio.Entidades.Urgente", "Id")
              .OnDelete(DeleteBehavior.Cascade)
              .IsRequired();
          });

        modelBuilder.Entity("LogicaNegocio.Entidades.Envio", b =>
          {
            b.Navigation("ListaSeguimiento");
          });
#pragma warning restore 612, 618
      }
    }
}
```

****
Archivo: RepositorioAgencia.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaAccesoDatos\Repositorio\RepositorioAgencia.cs
****

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

namespace LogicaAccesoDatos.Repositorio
{
  public class RepositorioAgencia : IRepositorioAgencia
  {
    private readonly Context _context;
    public RepositorioAgencia(Context context)
    {
      _context = context;
    }
```

```
        public void Add(Agencia entity)
        {
            if (entity == null)
                throw new ArgumentNullException("Los datos de la agencia son nulos");
            _context.Agencias.Add(entity);
            _context.SaveChanges();
        }

        public void Delete(Agencia entity)
        {
            throw new NotImplementedException();
        }

        public IEnumerable<Agencia> GetAll()
        {
            return _context.Agencias;
        }

        public Agencia? GetById(int id)
        {
            return _context.Agencias.Find(id);
        }

        public void Update(Agencia entity)
        {
            throw new NotImplementedException();
        }
    }
}
```

****
Archivo: RepositorioAuditoria.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaAccesoDatos\Repositorio\RepositorioAuditoria.cs
****

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

namespace LogicaAccesoDatos.Repositorio
{
    public class RepositorioAuditoria : IRepositorioAuditoria
    {
        private readonly Context _context;
```

```
        public RepositorioAuditoria(Context context)
        {
            _context = context;
        }
        public void Add(Auditoria entity)
        {
            if (entity == null)
                throw new ArgumentNullException("Datos vacios.");
            _context.Auditorias.Add(entity);
            _context.SaveChanges();
        }

        public void Delete(Auditoria entity)
        {
            throw new NotImplementedException();
        }

        public IEnumerable<Auditoria> GetAll()
        {
            throw new NotImplementedException();
        }

        public Auditoria? GetById(int id)
        {
            throw new NotImplementedException();
        }

        public void Update(Auditoria entity)
        {
            throw new NotImplementedException();
        }
    }
}
```

****
Archivo: RepositorioEnvio.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaAccesoDatos\Repositorio\RepositorioEnvio.cs
****

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

```
namespace LogicaAccesoDatos.Repositorio
{
   public class RepositorioEnvio : IRepositorioEnvio
   {
      private readonly Context _context;
      public RepositorioEnvio(Context context)
      {
         _context = context;
      }
      public void Add(Envio entity)
      {
         throw new NotImplementedException();
      }

      public void Delete(Envio entity)
      {
         throw new NotImplementedException();
      }

      public IEnumerable<Envio> GetAll()
      {
         return _context.Envios;
      }

      public Envio? GetById(int id)
      {
         return _context.Envios.Include(e => e.Cliente).Include(e => e.Empleado).Include(e =>
e.ListaSeguimiento).FirstOrDefault(e => e.Id == id);
      }

      public void Update(Envio entity)
      {
         ArgumentNullException.ThrowIfNull(entity);
         Envio envio = GetById(entity.Id) ?? throw new EnvioException("El envio a actualizar no
existe.");
         _context.Entry(envio).State = EntityState.Detached;
         _context.Envios.Update(entity);
         _context.SaveChanges();
      }

      public Envio? GetByNroTracking(string nroTracking)
      {
         return _context.Envios.Include(e => e.Cliente).Include(e => e.Empleado).Include(e =>
e.ListaSeguimiento).FirstOrDefault(e => e.NroTracking == nroTracking);
      }
   }
}
```

```
**********************************
Archivo: RepositorioEnvioComun.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaAccesoDatos\Repositorio\RepositorioEnvioComun.cs
**********************************
using LogicaNegocio.Entidades;
using LogicaNegocio.ExepcionesEntidades;
using LogicaNegocio.InterfacesRepositorio;
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAccesoDatos.Repositorio
{
   public class RepositorioEnvioComun : IRepositorioEnvioComun
   {
      private readonly Context _context;
      public RepositorioEnvioComun(Context context)
      {
         _context = context;
      }

      public void Add(Comun entity)
      {
         if (entity == null)
            throw new ArgumentNullException("Los datos del envio son nulos");
         _context.EnvioComun.Add(entity);
         _context.SaveChanges();
      }

      public void Delete(Comun entity)
      {
         throw new NotImplementedException();
      }

      public IEnumerable<Comun> GetAll()
      {
         throw new NotImplementedException();
      }

      public Comun? GetById(int id)
      {
         throw new NotImplementedException();
      }
```

```
        public void Update(Comun entity)
        {
            ArgumentNullException.ThrowIfNull(entity);
            Comun envio = GetById(entity.Id) ?? throw new EnvioException("El envio a actualizar no
existe.");
            _context.Entry(envio).State = EntityState.Detached;
            _context.EnvioComun.Update(entity);
            _context.SaveChanges();
        }
    }
}

**********
Archivo: RepositorioEnvioUrgente.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaAccesoDatos\Repositorio\RepositorioEnvioUrgente.cs
**********
using LogicaNegocio.Entidades;
using LogicaNegocio.InterfacesRepositorio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAccesoDatos.Repositorio
{
    public class RepositorioEnvioUrgente : IRepositorioEnvioUrgente
    {
        private readonly Context _context;
        public RepositorioEnvioUrgente(Context context)
        {
            _context = context;
        }
        public void Add(Urgente entity)
        {
            if(entity == null)
                throw new ArgumentNullException("Los datos del envio son nulos");
            _context.EnvioUrgente.Add(entity);
            _context.SaveChanges();
        }

        public void Delete(Urgente entity)
        {
            throw new NotImplementedException();
        }
```

```
        public IEnumerable<Urgente> GetAll()
        {
            throw new NotImplementedException();
        }

        public Urgente? GetById(int id)
        {
            throw new NotImplementedException();
        }

        public void Update(Urgente entity)
        {
            throw new NotImplementedException();
        }
    }
}
```

****
Archivo: RepositorioRol.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaAccesoDatos\Repositorio\RepositorioRol.cs
****

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

namespace LogicaAccesoDatos.Repositorio
{
    public class RepositorioRol : IRepositorioRol
    {
        private Context _context { get; set; }
        public RepositorioRol(Context context)
        {
            _context = context;
        }
        public void Add(Rol entity)
        {
            throw new NotImplementedException();
        }

        public void Delete(Rol entity)
        {
            throw new NotImplementedException();
        }
```

```
        public IEnumerable<Rol> GetAll()
        {
            return _context.Roles;
        }

        public Rol? GetById(int id)
        {
            return _context.Roles.Find(id);
        }
        public Rol? GetByName(string name)
        {
            return _context.Roles.FirstOrDefault(rol => rol.Nombre == name);
        }

        public void Update(Rol entity)
        {
            throw new NotImplementedException();
        }
    }
}
```

****
Archivo: RepositorioUsuario.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaAccesoDatos\Repositorio\RepositorioUsuario.cs
****

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

namespace LogicaAccesoDatos.Repositorio
{
    public class RepositorioUsuario : IRepositorioUsuario
    {
        private readonly Context _context;
        public RepositorioUsuario(Context context)
        {
            _context = context;
        }
        public void Add(Usuario entity)
        {
```

```
        if(entity == null)
            throw new ArgumentNullException("Datos vacios.");
        Usuario? usuario = GetByEmail(entity.Email.Value);
        if (usuario != null)
            throw new UsuarioException("El usuario o mail ya esta registrado!");
        if (GetByCI(entity.CI) != null)
            throw new UsuarioException("Ya existe un usuario con esa cedula de identidad.");
        _context.Usuarios.Add(entity);
        _context.SaveChanges();
    }

    public void Delete(Usuario entity)
    {
        if (entity == null)
            throw new ArgumentNullException("Datos vacios.");
        Usuario? usuario = GetById(entity.Id) ?? throw new UsuarioException("El usuario a eliminar no
existe.");
        _context.Usuarios.Remove(entity);
        _context.SaveChanges();
    }

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

    public Usuario? GetByEmail(string email)
    {
        return _context.Usuarios.FirstOrDefault(user => user.Email.Value == email);
    }

    public Usuario? GetById(int id)
    {
        return _context.Usuarios.Find(id);
    }

    public void Update(Usuario entity)
    {
        if (entity == null)
            throw new ArgumentNullException("Datos vacios, no se puede actualizar los cambios.");
        Usuario? usuario = GetById(entity.Id) ?? throw new UsuarioException("El usuario no esta
registrado.");
        if (usuario.Id != entity.Id && usuario.Email.Value == entity.Email.Value)
            throw new UsuarioException("Ya existe un usuario con ese email.");
        _context.Entry(usuario).State = EntityState.Detached;
        _context.Usuarios.Update(entity);
        _context.SaveChanges();
    }
```

```
        public IEnumerable<Usuario> GetByRol(int rolId)
        {
            return _context.Usuarios.Where(user => user.Rol.Id == rolId);
        }
        private Usuario? GetByCI(string ci)
        {
            return _context.Usuarios.FirstOrDefault(user => user.CI == ci);
        }
    }
}
```

****
Archivo: ListarRoles.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaAplicacion\CasosUso\ListarRoles.cs
****

```
using Compartido.DTOs;
using Compartido.Mappers;
using LogicaAplicacion.InterfacesCasosUso;
using LogicaNegocio.Entidades;
using LogicaNegocio.InterfacesRepositorio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso
{
    public class ListarRoles : IListarRoles
    {
        private IRepositorioRol _repoRol { get; set; }
        public ListarRoles(IRepositorioRol repoRol)
        {
            _repoRol = repoRol;
        }
        public List<InfoSelectDTO> Ejecutar()
        {
            List<Rol> roles = _repoRol.GetAll().ToList();
            return RolMapper.RolToRolDetallesDTO(roles);
        }
    }
}
```

****
Archivo: UsuarioLogin.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaAplicacion\CasosUso\UsuarioLogin.cs

```
**********
using Compartido.DTOs;
using LogicaAplicacion.InterfacesCasosUso;
using LogicaNegocio.InterfacesRepositorio;
using LogicaNegocio.Entidades;
using LogicaNegocio.ExepcionesEntidades;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Compartido.Mappers;

namespace LogicaAplicacion.CasosUso
{
   public class UsuarioLogin : IUsuarioLogin
   {
      private readonly IRepositorioUsuario _repoUsuario;
      private readonly IRepositorioRol _repoRol;
      public UsuarioLogin(IRepositorioUsuario repositorioUsuario, IRepositorioRol repositorioRol)
      {
         _repoUsuario = repositorioUsuario;
         _repoRol = repositorioRol;
      }
      public UsuarioLoggedDTO Login(string email, string password)
      {
         Usuario? usuario = _repoUsuario.GetByEmail(email);
         if(usuario == null && usuario.Password.Value != password)
            throw new UsuarioException("Credenciales invalidas");

         Rol rol = _repoRol.GetById(usuario.RolId) ?? throw new RolException("El rol no existe.");
         if (rol.Nombre == "Cliente")
            throw new UsuarioException("El cliente no tiene permisos para acceder a esta
funcionalidad.");
         return UsuarioMapper.UsuarioTOUsurioLoggedDTO(usuario, rol);
      }
   }
}

**********
Archivo: IListarRoles.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaAplicacion\InterfacesCasosUso\IListarRoles.cs
**********
using Compartido.DTOs;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
```

```
namespace LogicaAplicacion.InterfacesCasosUso
{
   public interface IListarRoles
   {
      List<InfoSelectDTO> Ejecutar();
   }
}

**********
Archivo: IUsuarioLogin.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaAplicacion\InterfacesCasosUso\IUsuarioLogin.cs
**********
using Compartido.DTOs;
namespace LogicaAplicacion.InterfacesCasosUso
{
   public interface IUsuarioLogin
   {
      UsuarioLoggedDTO Login(string username, string password);
   }
}

**********
Archivo: Agencia.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaNegocio\Entidades\Agencia.cs
**********
using LogicaNegocio.ValueObject;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaNegocio.Entidades
{
   public class Agencia
   {
      public int Id { get; set; }
      public string Nombre { get; set; }
      public string Direccion { get; set; }
      public Coordenada Coordenada { get; set; }
      public Agencia(string nombre, string direccion, decimal latitud, decimal longitud)
      {
         Nombre = nombre;
         Direccion = direccion;
         Coordenada = new Coordenada(latitud, longitud);
```

```
        }
        private Agencia() { }
    }
}

**********
Archivo: Auditoria.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaNegocio\Entidades\Auditoria.cs
**********

namespace LogicaNegocio.Entidades
{
    public class Auditoria
    {
        public int Id { get; set; }
        public string Accion { get; set; }
        public DateTime Fecha { get; set; }
        public Usuario Empleado { get; set; }
        public Auditoria(string accion, Usuario empleado)
        {
            Accion = accion;
            Fecha = DateTime.Now;
            Empleado = empleado;
        }
        private Auditoria() { }
    }
}

**********
Archivo: Comun.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaNegocio\Entidades\Comun.cs
**********
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations.Schema;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaNegocio.Entidades
{
    [Table("EnvioComun")]
    public class Comun : Envio
    {
        public Agencia AgenciaDestino { get; set; }
```

```
      public Comun(Usuario empleado, Usuario cliente, decimal peso, Agencia agenciaDestino) : base(
empleado, cliente, peso)
      {
         AgenciaDestino = agenciaDestino;
      }
      private Comun() { }

      public override void FinalizarEnvio()
      {
         base.FinalizarEnvio();
      }
   }
}

**********
Archivo: Envio.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaNegocio\Entidades\Envio.cs
**********
using LogicaNegocio.ExepcionesEntidades;
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations.Schema;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaNegocio.Entidades
{
   public enum Estados
   {
      EN_PROCESO,
      FINALIZADO
   }
   [Table("Envio")]
   public abstract class Envio
   {
      public int Id { get; set; }
      public string NroTracking { get; set; }
      public Usuario Empleado { get; set; }
      public Usuario Cliente { get; set; }
      public decimal Peso { get; set; }
      public Estados Estado { get; set; }
      public DateTime FechaCreacion { get; set; }
      public DateTime FechaEntrega { get; set; }
      public List<Seguimiento> ListaSeguimiento { get; set; } = new List<Seguimiento>();
      public Envio(Usuario empleado, Usuario cliente, decimal peso)
      {
```

```
            NroTracking = Guid.NewGuid().ToString();
            Empleado = empleado;
            Cliente = cliente;
            Peso = peso;
            Estado = Estados.EN_PROCESO;
            FechaCreacion = DateTime.Now;
            Validate();
        }
        public void Validate()
        {
            if (Peso <= 0)
                throw new EnvioException("El peso debe ser mayor a 0");
            if (Empleado == null)
                throw new EnvioException("El empleado no puede ser nulo");
            if (Cliente == null)
                throw new EnvioException("El cliente no puede ser nulo");
        }
        public Envio() { }

        public virtual void FinalizarEnvio()
        {
            FechaEntrega = DateTime.Now;
            Estado = Estados.FINALIZADO;
        }
    }
}
```

****
Archivo: Rol.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaNegocio\Entidades\Rol.cs
****

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

namespace LogicaNegocio.Entidades
{
    public class Rol
    {
        public int Id { get; set; }
        public string Nombre { get; set; }
        public Rol(string nombre)
        {
            Nombre = nombre;
        }
```

```
        private Rol() { }
    }
}

**********
Archivo: Seguimiento.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaNegocio\Entidades\Seguimiento.cs
**********
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaNegocio.Entidades
{
    public class Seguimiento
    {
        public int Id { get; set; }
        public string Comentario { get; set; }
        public Usuario Empleado { get; set; }
        public DateTime Fecha { get; set; }
        public Seguimiento(string comentario, Usuario empleado)
        {
            Comentario = comentario;
            Empleado = empleado;
            Fecha = DateTime.Now;
        }
        private Seguimiento() { }
    }
}

**********
Archivo: Urgente.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaNegocio\Entidades\Urgente.cs
**********
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations.Schema;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaNegocio.Entidades
{
    [Table("EnvioUrgente")]
```

```
   public class Urgente : Envio
   {
      public string Direccion { get; set; }
      public bool EntregadoEficiente { get; set; }
      public Urgente(Usuario empleado, Usuario cliente, decimal peso, string direccion) :
base(empleado, cliente, peso)
      {
         Direccion = direccion;
      }
      private Urgente() { }

      public override void FinalizarEnvio()
      {
         base.FinalizarEnvio();
         TimeSpan timeSpan = DateTime.Now - FechaCreacion;
         if (timeSpan.TotalHours < 24)
            EntregadoEficiente = true;
      }
   }
}

**********
Archivo: Usuario.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaNegocio\Entidades\Usuario.cs
**********
using LogicaNegocio.ExepcionesEntidades;
using LogicaNegocio.ValueObject;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaNegocio.Entidades
{
   public class Usuario : IEquatable<Usuario>
   {
      public int Id { get; set; }
      public string Nombre { get; set; }
      public string Apellido { get; set; }
      public string CI { get; set; }
      public string Celular { get; set; }
      public Email Email { get; set; }
      public Password Password { get; set; }
      public Rol Rol { get; set; }
      public int RolId { get; set; }
```

```
    public Usuario(string nombre, string apellido, string cI, string celular, string email, string
password, Rol rol)
    {
        Nombre = nombre;
        Apellido = apellido;
        CI = cI;
        Celular = celular;
        Email = new Email(email);
        Password = new Password(password);
        Rol = rol;
        RolId = rol.Id;
        Validate();
        FormatearCedula();
    }
    public Usuario() { }
    public void Validate()
    {
        if(string.IsNullOrEmpty(CI))
            throw new UsuarioException("CI no puede estar vacío");
        if(CI.Length != 8)
            throw new UsuarioException("CI debe tener 8 dígitos");
    }
    public void FormatearCedula()
    {
        string cedula = $"{CI[0]}.";
        for (int i = 1; i < CI.Length; i++)
        {
            if (i == 3)
                cedula += $"{CI[i]}.";
            else if (i == 6)
                cedula += $"{CI[i]}-";
            else
                cedula += CI[i];
        }
        CI = cedula;
    }
    public bool Equals(Usuario? other)
    {
        return Email.Equals(other?.Email);
    }
  }
}
```

****

Archivo: AgenciaException.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaNegocio\ExepcionesEntidades\AgenciaException.cs
****

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

namespace LogicaNegocio.ExepcionesEntidades
{
   public class AgenciaException : Exception
   {
      public AgenciaException() { }
      public AgenciaException(string message) : base(message) { }
      public AgenciaException(string message, Exception inner) : base(message, inner) { }
   }
}
```

****
Archivo: EnvioException.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaNegocio\ExepcionesEntidades\EnvioException.cs
****

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

namespace LogicaNegocio.ExepcionesEntidades
{
   public class EnvioException : Exception
   {
      public EnvioException() { }
      public EnvioException(string message) : base(message) { }
      public EnvioException(string message, Exception inner) : base(message, inner) { }
   }
}
```

****
Archivo: RolException.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaNegocio\ExepcionesEntidades\RolException.cs
****

```
namespace LogicaNegocio.ExepcionesEntidades
{
   public class RolException : Exception
   {
      public RolException(string message) : base(message)
```

```
        {
        }

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

****
Archivo: UsuarioException.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaNegocio\ExepcionesEntidades\UsuarioException.cs
****

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

namespace LogicaNegocio.ExepcionesEntidades
{
    public class UsuarioException : Exception
    {
        public UsuarioException() { }
        public UsuarioException(string message) : base(message) { }
        public UsuarioException(string message, Exception inner) : base(message, inner) { }
    }
}
```

****
Archivo: IRepositorio.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaNegocio\InterfacesRepositorio\IRepositorio.cs
****

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

namespace LogicaNegocio.InterfacesRepositorio
{
    public interface IRepositorio<T>
    {
        void Add(T entity);
        void Update(T entity);
```

```
        void Delete(T entity);
        T? GetById(int id);
        IEnumerable<T> GetAll();
    }
}
```

```
**********
Archivo: IRepositorioAgencia.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaNegocio\InterfacesRepositorio\IRepositorioAgencia.cs
**********
using LogicaNegocio.Entidades;

namespace LogicaNegocio.InterfacesRepositorio
{
    public interface IRepositorioAgencia : IRepositorio<Agencia>
    {
    }
}
```

```
**********
Archivo: IRepositorioAuditoria.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaNegocio\InterfacesRepositorio\IRepositorioAuditoria.cs
**********
using LogicaNegocio.Entidades;

namespace LogicaNegocio.InterfacesRepositorio
{
    public interface IRepositorioAuditoria : IRepositorio<Auditoria>
    {
    }
}
```

```
**********
Archivo: IRepositorioEnvio.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaNegocio\InterfacesRepositorio\IRepositorioEnvio.cs
**********
using LogicaNegocio.Entidades;

namespace LogicaNegocio.InterfacesRepositorio
{
    public interface IRepositorioEnvio : IRepositorio<Envio>
    {
        Envio? GetByNroTracking(string nroTracking);
    }
}
```

```
**********
Archivo: IRepositorioEnvioComun.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaNegocio\InterfacesRepositorio\IRepositorioEnvioComun.cs
**********

using LogicaNegocio.Entidades;

namespace LogicaNegocio.InterfacesRepositorio
{
   public interface IRepositorioEnvioComun : IRepositorio<Comun>
   {
   }
}

**********
Archivo: IRepositorioEnvioUrgente.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaNegocio\InterfacesRepositorio\IRepositorioEnvioUrgente.cs
**********
using LogicaNegocio.Entidades;

namespace LogicaNegocio.InterfacesRepositorio
{
   public interface IRepositorioEnvioUrgente : IRepositorio<Urgente>
   {
   }
}

**********
Archivo: IRepositorioRol.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaNegocio\InterfacesRepositorio\IRepositorioRol.cs
**********
using LogicaNegocio.Entidades;

namespace LogicaNegocio.InterfacesRepositorio
{
   public interface IRepositorioRol : IRepositorio<Rol>
   {
      Rol? GetByName(string name);
   }
}

**********
Archivo: IRepositorioUsuario.cs
```

Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaNegocio\InterfacesRepositorio\IRepositorioUsuario.cs
****

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using LogicaNegocio.Entidades;
namespace LogicaNegocio.InterfacesRepositorio
{
   public interface IRepositorioUsuario : IRepositorio<Usuario>
   {
      Usuario? GetByEmail(string email);
      IEnumerable<Usuario> GetByRol(int rolId);
   }
}
```

****

Archivo: Coordenada.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaNegocio\ValueObject\Coordenada.cs
****

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

namespace LogicaNegocio.ValueObject
{
   [ComplexType]
   public record Coordenada
   {
      public decimal Latitud { get; init; }
      public decimal Longitud { get; init; }
      private Coordenada() { }
      public Coordenada(decimal latitud, decimal longitud)
      {
         Latitud = latitud;
         Longitud = longitud;
         Validate();
      }
      public void Validate()
      {
         if (Latitud < -90 || Latitud > 90)
```

```
        {
            throw new AgenciaException("Latitud fuera de rango");
        }
        if (Longitud < -180 || Longitud > 180)
        {
            throw new AgenciaException("Longitud fuera de rango");
        }
    }
  }
}
```

****
Archivo: Email.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaNegocio\ValueObject\Email.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;
using LogicaNegocio.ExepcionesEntidades;

namespace LogicaNegocio.ValueObject
{
  [ComplexType]
  public record Email
  {

    public string Value { get; init; }
    public Email() { }
    public Email(string value)
    {
      Value = value;
      Validate();
    }
    private void Validate()
    {
      if (string.IsNullOrEmpty(Value))
      {
        throw new UsuarioException("El email no puede estar vacio.");
      }
      if (!Value.Contains("@"))
      {
        throw new UsuarioException("El email debe contener un @.");
      }
```

```
            if (!Value.EndsWith(".com"))
            {
                throw new UsuarioException("El email debe terminar con .com.");
            }
        }
    }
}

**********
Archivo: Password.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaNegocio\ValueObject\Password.cs
**********
using LogicaNegocio.ExepcionesEntidades;
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 LogicaNegocio.ValueObject
{
    [ComplexType]
    public class Password
    {
        public string Value { get; private set; }
        public Password() { }
        public Password(string value)
        {
            Value = value;
            Validate();
        }
        private void Validate()
        {
            if (string.IsNullOrEmpty(Value))
            {
                throw new UsuarioException("La contraseña no puede estar vacia.");
            }
            if (Value.Length < 8)
            {
                throw new UsuarioException("La contraseña debe tener al menos 8 caracteres");
            }
        }
    }
}
```

```
**********************************
Archivo: EnvioController.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\MVC\Controllers\EnvioController.cs
**********************************
using Compartido.DTOs.Envio;
using LogicaAplicacion.InterfacesCasosUso.AgenciaCU;
using LogicaAplicacion.InterfacesCasosUso.EnvioCU;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using MVC.Helpers;
using MVC.Models;
using MVC.Models.Envio;
using MVC.Filter;
using LogicaAplicacion.InterfacesCasosUso.UsuarioCU;
using LogicaNegocio.ExepcionesEntidades;
using Microsoft.EntityFrameworkCore.Metadata.Internal;

namespace MVC.Controllers
{
   public class EnvioController : Controller
   {
      private readonly IAltaEnvioUrgente _altaEnvioUrgente;
      private readonly IAltaEnvioComun _altaEnvioComun;
      private readonly IListarSAgencia _listarSAgencia;
      private readonly IListarEnvios _listarEnvios;
      private readonly IListarSelectUsuario _listarSUsuario;
      private readonly IDetalleEnvio _detalleEnvio;
      private readonly IUpdateEnvio _updateEnvio;
      private readonly IAgregarComentario _agregarComentario;
      public EnvioController(IAltaEnvioUrgente altaEnvioUrgente, IAltaEnvioComun altaEnvioComun,
                        IListarSAgencia listarSAgencia, IListarEnvios listarEnvios,
                        IListarSelectUsuario listarSUsuario, IDetalleEnvio detalleEnvio,
                        IUpdateEnvio updateEnvio, IAgregarComentario agregarComentario)
      {
         _altaEnvioUrgente = altaEnvioUrgente;
         _altaEnvioComun = altaEnvioComun;
         _listarSAgencia = listarSAgencia;
         _listarEnvios = listarEnvios;
         _listarSUsuario = listarSUsuario;
         _detalleEnvio = detalleEnvio;
         _updateEnvio = updateEnvio;
         _agregarComentario = agregarComentario;
      }
      // GET: EnvioController
      [Funcionarios]
      [IsAuthenticated]
      public ActionResult Index()
```

```
{
    List<EnvioListadoViewModel> enviosVM = new List<EnvioListadoViewModel>();
    try
    {
        List<EnvioListadoDTO> envios = _listarEnvios.Ejecutar();
        enviosVM = envios.Select(e => new EnvioListadoViewModel
        {
            Id = e.Id,
            NroTracking = e.NroTracking,
            Empleado = e.Empleado,
            Cliente = e.Cliente,
            Peso = e.Peso,
            Estado = e.Estado,
            TipoEnvio = e.TipoEnvio,
        }).ToList();
    }
    catch (Exception ex)
    {
        ViewBag.Msg = ex.Message;
    }
    return View(enviosVM);
}

// GET: EnvioController/Details/5
public ActionResult Details(int id)
{
    return View();
}
[Funcionarios]
[IsAuthenticated]
// GET: EnvioController/Create
public ActionResult Create()
{
    EnvioCreateViewModel envioVM = new EnvioCreateViewModel();
    try
    {
        CargarDatos.AgenciaSelect(envioVM, _listarSAgencia);
        CargarDatos.UsuarioSelect(envioVM, _listarSUsuario);
    }
    catch (Exception)
    {

        ViewBag.Msg = "Error al cargar las agencias";
    }
    return View(envioVM);
}

// POST: EnvioController/Create
```

```
[HttpPost]
[ValidateAntiForgeryToken]
[Funcionarios]
[IsAuthenticated]
public ActionResult Create(EnvioCreateViewModel envioVM)
{

   try
   {
      CargarDatos.AgenciaSelect(envioVM, _listarSAgencia);
      CargarDatos.UsuarioSelect(envioVM, _listarSUsuario);

      int idFuncionario = HttpContext.Session.GetInt32("Id").Value;
      if (envioVM.EsUrgente)
      {
         EnvioUrgenteDTO envioUDto = new EnvioUrgenteDTO
         {
            EsUrgente = envioVM.EsUrgente,
            ClienteId = envioVM.EmailCliente,
            DireccionPostal = envioVM.DireccionPostal,
            Peso = envioVM.Peso
         };
         _altaEnvioUrgente.Ejecutar(envioUDto, idFuncionario);
      }
      else
      {
         EnvioComunDTO envioCDto = new EnvioComunDTO
         {
            AgenciaId = envioVM.AgenciaId,
            ClienteId = envioVM.EmailCliente,
            Peso = envioVM.Peso
         };
         _altaEnvioComun.Ejecutar(envioCDto, idFuncionario);
      }
      return RedirectToAction(nameof(Index));
   }
   catch (ArgumentException ex)
   {
      ViewBag.Msg = ex.Message;
   }
   catch (Exception ex)
   {
      ViewBag.Msg = ex.Message;
   }
   return View(envioVM);
}

// GET: EnvioController/Edit/5
```

```
public ActionResult Edit(int id)
{
   EnvioDetalleViewModel envioVM = new EnvioDetalleViewModel();
   try
   {
      EnvioDetalleDto envioDto = _detalleEnvio.Ejecutar(id);
      envioVM = new EnvioDetalleViewModel
      {
         Id = envioDto.Id,
         NroTracking = envioDto.NroTracking,
         Empleado = envioDto.Empleado,
         Cliente = envioDto.Cliente,
         Peso = envioDto.Peso,
         Estado = envioDto.Estado,
         TipoEnvio = envioDto.TipoEnvio,
         FechaCreacion = envioDto.FechaCreacion,
         FechaEntrega = envioDto.FechaEntrega,
         Estados = envioDto.Estados
      };
   }
   catch (EnvioException e)
   {
      ViewBag.Msg = e.Message;
   }
   catch (Exception)
   {
      ViewBag.Msg = "Error al cargar el envio";
   }
   return View(envioVM);
}

// POST: EnvioController/Edit/5
[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult Edit(EnvioDetalleViewModel envioVM)
{
   try
   {
      if (!ModelState.IsValid)
         throw new ArgumentException("Datos invalidos");
      EnvioUpdateDTO envioUDto = new EnvioUpdateDTO
      {
         Id = envioVM.Id
      };
      _updateEnvio.Ejecutar(envioUDto);
      return RedirectToAction(nameof(Index));
   }
   catch (EnvioException e)
```

```
        {
            ViewBag.Msg = e.Message;
        }
        catch (ArgumentException)
        {
            ViewBag.Msg = "Valores vacios";
        }
        catch (Exception)
        {
            ViewBag.Msg = "Error al cargar el envio";
        }
        return View();
    }



    public ActionResult Comentar(int id)
    {
        ComentarioViewModel comentarioVM = new ComentarioViewModel();
        comentarioVM.IdEnvio = id;
        comentarioVM.IdEmpleado = HttpContext.Session.GetInt32("Id").Value;
        return View(comentarioVM);
    }
    [HttpPost]
    [ValidateAntiForgeryToken]
    public ActionResult Comentar(ComentarioViewModel comentarioVM)
    {
        try
        {
            _agregarComentario.Ejecutar(comentarioVM.IdEnvio, comentarioVM.IdEmpleado,
comentarioVM.Comentario);
        }
        catch (Exception e)
        {
            ViewBag.Msg = e.Message;
        }
        return View();
    }
  }
}
```

****

Archivo: HomeController.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Controllers\HomeController.cs
****

```
using Microsoft.AspNetCore.Mvc;
using MVC.Models;
```

```
using System.Diagnostics;

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

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

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

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

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

****
Archivo: UsuarioController.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Controllers\UsuarioController.cs
****

```
using Compartido.DTOs;
using Compartido.DTOs.Funcionario;
using LogicaAplicacion.InterfacesCasosUso;
using LogicaAplicacion.InterfacesCasosUso.FuncionarioCU;
using LogicaNegocio.ExepcionesEntidades;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using MVC.Filter;
using MVC.Helpers;
using MVC.Models.Funcionario;
using MVC.Models.Rol;
```

```
using MVC.Models.Usuario;

namespace MVC.Controllers
{
   public class UsuarioController : Controller
   {
      private readonly IUsuarioLogin _usuarioLogin;
      private readonly IAltaFuncionario _altaFuncionario;
      private readonly IListarFuncionarios _listarFuncionarios;
      private readonly IDetalleFuncionario _detalleFuncionario;
      private readonly IListarRoles _listarRoles;
      private readonly IUpdateFuncionario _updateFuncionario;
      private readonly IDetalleUpdateFuncionario _detalleUpdateFuncionario;
      private readonly IEliminarFuncionario _eliminarFuncionario;
      public UsuarioController(IUsuarioLogin usuarioLogin,
                               IAltaFuncionario altaFuncionario,
                               IListarFuncionarios listarFuncionarios,
                               IDetalleFuncionario detalleFuncionario,
                               IListarRoles listarRoles,
                               IUpdateFuncionario updateFuncionario,
                               IDetalleUpdateFuncionario detalleUpdateFuncionario,
                               IEliminarFuncionario eliminarFuncionario)
      {
         _usuarioLogin = usuarioLogin;
         _altaFuncionario = altaFuncionario;
         _listarFuncionarios = listarFuncionarios;
         _detalleFuncionario = detalleFuncionario;
         _listarRoles = listarRoles;
         _updateFuncionario = updateFuncionario;
         _detalleUpdateFuncionario = detalleUpdateFuncionario;
         _eliminarFuncionario = eliminarFuncionario;
      }
      // GET: UsuarioController
      [Admin]
      public ActionResult Index()
      {
         List<FuncionarioListarViewModel> listaFuncionarioVM = new
List<FuncionarioListarViewModel>();
         List<FuncionarioListarDTO> listaFuncionarioDTO = _listarFuncionarios.Ejecutar();
         try
         {
            listaFuncionarioVM = listaFuncionarioDTO.Select(f => new FuncionarioListarViewModel()
            {
               Id = f.Id,
               Nombre = f.Nombre,
               Apellido = f.Apellido,
               CI = f.CI,
               Email = f.Email,
```

```
            Rol = f.Rol
        }).ToList();
    }
    catch (Exception ex)
    {
        ViewBag.Msg = ex.Message;
    }
    return View(listaFuncionarioVM);
}

// GET: UsuarioController/Details/5
[Admin]
public ActionResult Details(int id)
{
    FuncionarioDetailViewModel funcionarioVM = new FuncionarioDetailViewModel();
    try
    {
        FuncionarioDetailDTO funcionarioDetailDTO = _detalleFuncionario.Ejecutar(id);
        funcionarioVM = new FuncionarioDetailViewModel()
        {
            Id = funcionarioDetailDTO.Id,
            Nombre = funcionarioDetailDTO.Nombre,
            Apellido = funcionarioDetailDTO.Apellido,
            CI = funcionarioDetailDTO.CI,
            Celular = funcionarioDetailDTO.Celular,
            Email = funcionarioDetailDTO.Email
        };
    }
    catch (UsuarioException ex)
    {
        ViewBag.Msg = ex.Message;
    }
    catch (ArgumentNullException ex)
    {
        ViewBag.Msg = ex.Message;
    }
    catch (Exception ex)
    {
        ViewBag.Msg = "Error inesperado.";
    }
    return View(funcionarioVM);
}
// GET: UsuarioController/Login
public ActionResult Login()
{
    return View();
}
```

```
        // POST: UsuarioController/Login
        [HttpPost]
        [ValidateAntiForgeryToken]
        public ActionResult Login(UsuarioLoginViewModel usuarioLoginVM)
        {
            try
            {
                if (ModelState.IsValid)
                {
                    UsuarioLoggedDTO usuario = _usuarioLogin.Login(usuarioLoginVM.Email,
usuarioLoginVM.Password);
                    HttpContext.Session.SetInt32("Id", usuario.Id);
                    HttpContext.Session.SetString("Usuario", usuario.Nombre);
                    HttpContext.Session.SetString("Apellido", usuario.Apellido);
                    HttpContext.Session.SetString("Email", usuario.Email);
                    HttpContext.Session.SetString("Rol", usuario.Rol);
                }

                return Redirect("/");
            }
            catch (UsuarioException ex)
            {
                ViewBag.Msg = ex.Message;
            }
            return View();
        }
        // GET: UsuarioController/Logout
        [HttpGet]
        public ActionResult Logout()
        {
            HttpContext.Session.Clear();
            return RedirectToAction("Index", "Home");
        }

        // GET: UsuarioController/Create
        [Admin]
        public ActionResult Create()
        {
            FuncionarioViewModel funcionarioVM = new FuncionarioViewModel();
            CargarRoles(funcionarioVM);
            return View(funcionarioVM);
        }

        // POST: UsuarioController/Create
        [HttpPost]
        [Admin]
        [ValidateAntiForgeryToken]
        public ActionResult Create(FuncionarioViewModel funcionarioVM)
```

```
{
    try
    {
        if (!ModelState.IsValid)
            throw new ArgumentException("Datos invalidos.");

        FuncionarioDTO funcionarioDTO = new FuncionarioDTO()
        {
            Nombre = funcionarioVM.Nombre,
            Apellido = funcionarioVM.Apellido,
            CI = funcionarioVM.CI,
            Celular = funcionarioVM.Celular,
            Email = funcionarioVM.Email,
            Password = funcionarioVM.Password,
            RolId = funcionarioVM.RolId
        };
        int idUsuario = HttpContext.Session.GetInt32("Id").Value;
        _altaFuncionario.Ejecutar(funcionarioDTO, idUsuario);
        return RedirectToAction(nameof(Index));
    }
    catch (UsuarioException ex)
    {
        ViewBag.Msg = ex.Message;
    }
    catch (RolException ex)
    {
        ViewBag.Msg = ex.Message;
    }
    catch (ArgumentException ex)
    {
        ViewBag.Msg = ex.Message;
    }
    catch (Exception ex)
    {
        ViewBag.Msg = "Ups, acaba de ocurrir un error inesperado.";
    }
    CargarRoles(funcionarioVM);
    return View(funcionarioVM);
}
private void CargarRoles(FuncionarioViewModel funcionarioVM)
{
    try
    {
        List<InfoSelectDTO> listaRolDTO = _listarRoles.Ejecutar();
        funcionarioVM.Roles = listaRolDTO.Select(r => new RolViewModel()
        {
            Id = r.Id,
            Nombre = r.Nombre
```

```
            }).ToList();
        }
        catch (Exception ex)
        {
            ViewBag.Msg = ex.Message;
        }
    }

    // GET: UsuarioController/Edit/5
    [Admin]
    public ActionResult Edit(int id)
    {
        FuncionarioUpdateViewModel funcionarioUpdateViewModel = new
FuncionarioUpdateViewModel();
        try
        {
            CargarDatos.RolesSelect(funcionarioUpdateViewModel, _listarRoles);
            FuncionarioUpdateDTO funcionarioDTO = _detalleUpdateFuncionario.Ejecutar(id);
            funcionarioUpdateViewModel.Id = funcionarioDTO.Id;
            funcionarioUpdateViewModel.Nombre = funcionarioDTO.Nombre;
            funcionarioUpdateViewModel.Apellido = funcionarioDTO.Apellido;
            funcionarioUpdateViewModel.CI = funcionarioDTO.CI;
            funcionarioUpdateViewModel.Celular = funcionarioDTO.Celular;
            funcionarioUpdateViewModel.Email = funcionarioDTO.Email;
            funcionarioUpdateViewModel.Password = funcionarioDTO.Password;
            funcionarioUpdateViewModel.RolId = funcionarioDTO.RolId;
        }
        catch (UsuarioException ex)
        {
            ViewBag.Msg = ex.Message;
        }
        return View(funcionarioUpdateViewModel);
    }

    // POST: UsuarioController/Edit/5
    [HttpPost]
    [Admin]
    [ValidateAntiForgeryToken]
    public ActionResult Edit(FuncionarioUpdateViewModel funcionarioUpdateViewModel)
    {
        try
        {
            if (!ModelState.IsValid)
                throw new ArgumentException("Datos invalidos.");
            FuncionarioUpdateDTO funcionarioDTO = new FuncionarioUpdateDTO()
            {
                Id = funcionarioUpdateViewModel.Id,
                Nombre = funcionarioUpdateViewModel.Nombre,
```

```
            Apellido = funcionarioUpdateViewModel.Apellido,
            CI = funcionarioUpdateViewModel.CI,
            Celular = funcionarioUpdateViewModel.Celular,
            Email = funcionarioUpdateViewModel.Email,
            Password = funcionarioUpdateViewModel.Password,
            RolId = funcionarioUpdateViewModel.RolId
        };
        int idAdmin = HttpContext.Session.GetInt32("Id").Value;
        _updateFuncionario.Ejecutar(funcionarioDTO, idAdmin);
        return RedirectToAction(nameof(Index));
    }
    catch (UsuarioException ex)
    {
        ViewBag.Msg = ex.Message;
    }
    catch (RolException ex)
    {
        ViewBag.Msg = ex.Message;
    }
    catch (ArgumentException ex)
    {
        ViewBag.Msg = ex.Message;
    }
    catch (Exception ex)
    {
        ViewBag.Msg = "Ups, acaba de ocurrir un error inesperado.";
    }
    CargarDatos.RolesSelect(funcionarioUpdateViewModel, _listarRoles);
    return View(funcionarioUpdateViewModel);
}

// GET: UsuarioController/Delete/5
[Admin]
public ActionResult Delete(int id)
{
    try
    {
        FuncionarioDetailDTO funcionarioDetailDTO = _detalleFuncionario.Ejecutar(id);
        FuncionarioDetailViewModel funcionarioVM = new FuncionarioDetailViewModel()
        {
            Id = funcionarioDetailDTO.Id,
            Nombre = funcionarioDetailDTO.Nombre,
            Apellido = funcionarioDetailDTO.Apellido,
            CI = funcionarioDetailDTO.CI,
            Celular = funcionarioDetailDTO.Celular,
            Email = funcionarioDetailDTO.Email
        };
        return View(funcionarioVM);
```

```
        }
        catch (UsuarioException ex)
        {
            ViewBag.Msg = ex.Message;
        }
        catch (ArgumentNullException ex)
        {
            ViewBag.Msg = ex.Message;
        }
        catch (Exception ex)
        {
            ViewBag.Msg = "Error inesperado.";
        }
        return View(new FuncionarioDetailViewModel());
    }

    // POST: UsuarioController/Delete/5
    [HttpPost]
    [Admin]
    [ValidateAntiForgeryToken]
    public ActionResult Delete(int id, FuncionarioDetailViewModel funcionarioDetailVM)
    {
        try
        {
            int idAdmin = HttpContext.Session.GetInt32("Id").Value;
            _eliminarFuncionario.Ejecutar(id, idAdmin);
            return RedirectToAction(nameof(Index));
        }
        catch (UsuarioException ex)
        {
            ViewBag.Msg = ex.Message;
        }
        catch(Exception ex)
        {
            ViewBag.Msg = "Error inesperado.";
        }
        return View();
    }
  }
}
```

****
Archivo: Admin.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Filter\Admin.cs
****

```
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Mvc.Filters;
```

```
namespace MVC.Filter
{
   public class Admin : Attribute, IAuthorizationFilter
   {
      public void OnAuthorization(AuthorizationFilterContext context)
      {
         if (context.HttpContext.Session.GetString("Rol") != "Administrador")
            context.Result = new RedirectResult("/");
      }
   }


}
```

****
Archivo: Funcionarios.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Filter\Funcionarios.cs
****

```
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Mvc.Filters;

namespace MVC.Filter
{
   public class Funcionarios : Attribute, IAuthorizationFilter
   {
      public void OnAuthorization(AuthorizationFilterContext context)
      {
         if(context.HttpContext.Session.GetString("Rol") == "Cliente")
            context.Result = new RedirectResult("/Home/Index");
      }
   }
}
```

****
Archivo: IsAuthenticated.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Filter\IsAuthenticated.cs
****

```
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Mvc.Filters;

namespace MVC.Filter
{
   public class IsAuthenticated : Attribute, IAuthorizationFilter
   {
      public void OnAuthorization(AuthorizationFilterContext context)
```

```
        {
            if (context.HttpContext.Session.GetString("Rol") == null)
                context.Result = new RedirectResult("/");
        }
    }
}
```

****
Archivo: CargarDatos.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Helpers\CargarDatos.cs
****

```
using Compartido.DTOs;
using Compartido.DTOs.Agencia;
using LogicaAplicacion.InterfacesCasosUso;
using LogicaAplicacion.InterfacesCasosUso.AgenciaCU;
using LogicaAplicacion.InterfacesCasosUso.UsuarioCU;
using MVC.Models;
using MVC.Models.Agencia;
using MVC.Models.Funcionario;
using MVC.Models.Rol;

namespace MVC.Helpers
{
    public class CargarDatos
    {
        public static void RolesSelect(FuncionarioViewModel funcionarioVM, IListarRoles listaRoles)
        {
            List<InfoSelectDTO> listaRolDTO = listaRoles.Ejecutar();
            funcionarioVM.Roles = listaRolDTO.Select(r => new RolViewModel()
            {
                Id = r.Id,
                Nombre = r.Nombre
            }).ToList();
        }
        public static void RolesSelect(FuncionarioUpdateViewModel funcionarioVM, IListarRoles
listaRoles)
        {
            List<InfoSelectDTO> listaRolDTO = listaRoles.Ejecutar();
            funcionarioVM.Roles = listaRolDTO.Select(r => new RolViewModel()
            {
                Id = r.Id,
                Nombre = r.Nombre
            }).ToList();
        }
        public static void AgenciaSelect(EnvioCreateViewModel envioVM, IListarSAgencia listaSAgencia)
        {
            List<InfoSelectDTO> listaAgenciaDTO = listaSAgencia.Ejecutar();
```

```
            envioVM.Agencias = listaAgenciaDTO.Select(r => new AgenciaSelectViewModel()
            {
                Id = r.Id,
                Nombre = r.Nombre
            }).ToList();
        }
        public static void UsuarioSelect(EnvioCreateViewModel envioVM, IListarSelectUsuario
listaSUsuario)
        {
            List<InfoSelectDTO> listaAgenciaDTO = listaSUsuario.Ejecutar();
            envioVM.Usuarios = listaAgenciaDTO.Select(r => new SelectInfoViewModel()
            {
                Id = r.Id,
                Nombre = r.Nombre
            }).ToList();
        }
    }
}
```

****
Archivo: EnvioCreateViewModel.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Models\EnvioCreateViewModel.cs
****

```
using Compartido.DTOs.Agencia;
using MVC.Models.Agencia;

namespace MVC.Models
{
    public class EnvioCreateViewModel
    {
        public bool EsUrgente { get; set; }
        public int EmailCliente { get; set; }
        public int AgenciaId { get; set; }
        public string DireccionPostal { get; set; }
        public decimal Peso { get; set; }

        public IEnumerable<AgenciaSelectViewModel> Agencias { get; set; } = new
List<AgenciaSelectViewModel>();
        public IEnumerable<SelectInfoViewModel> Usuarios { get; set; } = new
List<SelectInfoViewModel>();

    }
}
```

****
Archivo: ErrorViewModel.cs

Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Models\ErrorViewModel.cs
****
```
namespace MVC.Models
{
    public class ErrorViewModel
    {
        public string? RequestId { get; set; }

        public bool ShowRequestId => !string.IsNullOrEmpty(RequestId);
    }
}
```

****
Archivo: SelectInfoViewModel.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Models\SelectInfoViewModel.cs
****
```
namespace MVC.Models
{
    public class SelectInfoViewModel
    {
        public int Id { get; set; }
        public string Nombre { get; set; }
    }
}
```

****
Archivo: EnvioController.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\WebAPI\Controllers\EnvioController.cs
****
```
using LogicaAplicacion.InterfacesCasosUso.EnvioCU;
using LogicaNegocio.ExepcionesEntidades;
using Microsoft.AspNetCore.Mvc;

namespace WebAPI.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class EnvioController : ControllerBase
    {
        private readonly IObtenerEnvio _obtenerEnvio;
        public EnvioController(IObtenerEnvio obtenerEnvio)
        {
            _obtenerEnvio = obtenerEnvio;
        }
        // GET: api/<EnvioController>
```

```
        [HttpGet]
        public IEnumerable<string> Get()
        {
            return new string[] { "value1", "value2" };
        }

        // GET api/<EnvioController>/5
        /// <summary>
        /// Permite consultar un envio por su numero de tracking.
        /// </summary>
        /// <param name="nroTracking"> Numero de tracking del envio a consultar.</param>
        /// <returns></returns>
        [ProducesResponseType(StatusCodes.Status200OK)]
        [ProducesResponseType(StatusCodes.Status400BadRequest)]
        [ProducesResponseType(StatusCodes.Status500InternalServerError)]
        [HttpGet("{nroTracking}")]
        public IActionResult Get(string nroTracking)
        {
            try
            {
                if (String.IsNullOrEmpty(nroTracking))
                    throw new ArgumentNullException("El numero de tracking no puede ser nulo o vacio.");
                return Ok(_obtenerEnvio.Ejecutar(nroTracking));
            }
            catch (EnvioException error)
            {
                return BadRequest(error.Message);
            }
            catch (ArgumentNullException error)
            {
                return BadRequest(error.Message);
            }
            catch (Exception error)
            {
                return StatusCode(500, "Error interno :(");
            }
        }
    }
}

**********
Archivo: AgenciaSelectDTO.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\Compartido\DTOs\Agencia\AgenciaSelectDTO.cs
**********

namespace Compartido.DTOs.Agencia
{
```

```
    public class AgenciaSelectDTO
    {
        public int Id { get; set; }
        public string Nombre { get; set; }
    }
}

**********
Archivo: ComentarioDto.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\Compartido\DTOs\Envio\ComentarioDto.cs
**********
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Compartido.DTOs.Envio
{
    public class ComentarioDto
    {
        public int IdEnvio { get; set; }
        public int IdEmpleado { get; set; }
        public string Comentario { get; set; }
    }
}

**********
Archivo: EnvioAPInfoDto.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\Compartido\DTOs\Envio\EnvioAPInfoDto.cs
**********
namespace Compartido.DTOs.Envio
{
    public class EnvioAPInfoDto
    {
        public string NroTracking { get; set; }
        public string Estado { get; set; }
        public decimal Peso { get; set; }
        public string Comentario { get; set; }
        public string TipoEnvio { get; set; }
        public string InfoAdicional { get; set; }
}
}

**********
Archivo: EnvioComunDTO.cs
```

Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\Compartido\DTOs\Envio\EnvioComunDTO.cs
****

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

namespace Compartido.DTOs.Envio
{
   public class EnvioComunDTO
   {
      public int ClienteId { get; set; }
      public int AgenciaId { get; set; }
      public decimal Peso { get; set; }
   }
}
```

****

Archivo: EnvioDetalleDto.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\Compartido\DTOs\Envio\EnvioDetalleDto.cs
****

```
namespace Compartido.DTOs.Envio
{
   public class EnvioDetalleDto
   {
      public int Id { get; set; }
      public string NroTracking { get; set; }
      public string Empleado { get; set; }
      public string Cliente { get; set; }
      public decimal Peso { get; set; }
      public string Estado { get; set; }
      public string TipoEnvio { get; set; }
      public DateTime FechaCreacion { get; set; }
      public DateTime FechaEntrega { get; set; }
      public List<string> Estados = new List<string>();
   }
}
```

****

Archivo: EnvioDTO.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\Compartido\DTOs\Envio\EnvioDTO.cs
****

```
namespace Compartido.DTOs.Envio
```

```
{
   public class EnvioDTO
   {
      public bool EsUrgente { get; set; }
      public int EmailCliente { get; set; }
      public int AgenciaId { get; set; }
      public string DireccionPostal { get; set; }
      public decimal Peso { get; set; }
   }
}

**********
Archivo: EnvioListadoDTO.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\Compartido\DTOs\Envio\EnvioListadoDTO.cs
**********

namespace Compartido.DTOs.Envio
{
   public class EnvioListadoDTO
   {
      public int Id { get; set; }
      public string NroTracking { get; set; }
      public string Empleado { get; set; }
      public string Cliente { get; set; }
      public decimal Peso { get; set; }
      public string Estado { get; set; }
      public string TipoEnvio { get; set; }
   }
}

**********
Archivo: EnvioUpdateDTO.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\Compartido\DTOs\Envio\EnvioUpdateDTO.cs
**********
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Compartido.DTOs.Envio
{
   public class EnvioUpdateDTO
   {
      public int Id { get; set; }
```

```
    }
}

**********
Archivo: EnvioUrgenteDTO.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\Compartido\DTOs\Envio\EnvioUrgenteDTO.cs
**********

namespace Compartido.DTOs.Envio
{
    public class EnvioUrgenteDTO
    {
        public bool EsUrgente { get; set; }
        public int ClienteId { get; set; }
        public string DireccionPostal { get; set; }
        public decimal Peso { get; set; }
    }
}

**********
Archivo: FuncionarioAllDataDTO.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\Compartido\DTOs\Funcionario\FuncionarioAllDataDTO.cs
**********
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Compartido.DTOs.Funcionario
{
    public class FuncionarioAllDataDTO
    {
        public int Id { get; set; }
        public string Nombre { get; set; }
        public string Apellido { get; set; }
        public string CI { get; set; }
        public string Celular { get; set; }
        public string Email { get; set; }
        public string Password { get; set; }
        public int RolId { get; set; }
    }
}

**********
Archivo: FuncionarioDetailDTO.cs
```

Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\Compartido\DTOs\Funcionario\FuncionarioDetailDTO.cs
****
```
namespace Compartido.DTOs.Funcionario
{
   public class FuncionarioDetailDTO
   {
      public int Id { get; set; }
      public string Nombre { get; set; }
      public string Apellido { get; set; }
      public string CI { get; set; }
      public string Celular { get; set; }
      public string Email { get; set; }
      public string Rol { get; set; }
   }
}
```

****
Archivo: FuncionarioDTO.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\Compartido\DTOs\Funcionario\FuncionarioDTO.cs
****
```
namespace Compartido.DTOs.Funcionario
{
   public class FuncionarioDTO
   {
      public string Nombre { get; set; }
      public string Apellido { get; set; }
      public string CI { get; set; }
      public string Celular { get; set; }
      public string Email { get; set; }
      public string Password { get; set; }
      public int RolId { get; set; }
   }
}
```
****
Archivo: FuncionarioListarDTO.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\Compartido\DTOs\Funcionario\FuncionarioListarDTO.cs
****
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Compartido.DTOs.Funcionario
{
```

```
    public class FuncionarioListarDTO
    {
        public int Id { get; set; }
        public string Nombre { get; set; }
        public string Apellido { get; set; }
        public string CI { get; set; }
        public string Email { get; set; }
        public string Rol { get; set; }
    }
}

**********
Archivo: FuncionarioUpdateDTO.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\Compartido\DTOs\Funcionario\FuncionarioUpdateDTO.cs
**********
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Compartido.DTOs.Funcionario
{
    public class FuncionarioUpdateDTO
    {
        public int Id { get; set; }
        public string Nombre { get; set; }
        public string Apellido { get; set; }
        public string CI { get; set; }
        public string Celular { get; set; }
        public string Email { get; set; }
        public string Password { get; set; }
        public int RolId { get; set; }
    }
}

**********
Archivo: ListarSAgencia.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaAplicacion\CasosUso\AgenciaCU\ListarSAgencia.cs
**********
using Compartido.DTOs;
using Compartido.DTOs.Agencia;
using Compartido.Mappers;
using LogicaAplicacion.InterfacesCasosUso.AgenciaCU;
using LogicaNegocio.Entidades;
using LogicaNegocio.InterfacesRepositorio;
```

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

namespace LogicaAplicacion.CasosUso.AgenciaCU
{
   public class ListarSAgencia : IListarSAgencia
   {
      private readonly IRepositorioAgencia _repoAgencia;
      public ListarSAgencia(IRepositorioAgencia repoAgencia)
      {
         _repoAgencia = repoAgencia;
      }

      public List<InfoSelectDTO> Ejecutar()
      {
         List<Agencia> agencias = _repoAgencia.GetAll().ToList();
         return AgenciaMapper.AgenciaToInfoSelectDTO(agencias);
      }
   }
}

**********
Archivo: AgregarComentario.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaAplicacion\CasosUso\EnvioCU\AgregarComentario.cs
**********
using LogicaAplicacion.InterfacesCasosUso.EnvioCU;
using LogicaNegocio.Entidades;
using LogicaNegocio.InterfacesRepositorio;
using LogicaNegocio.ExepcionesEntidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.EnvioCU
{
   public class AgregarComentario : IAgregarComentario
   {
      private readonly IRepositorioEnvio _repoEnvio;
      private readonly IRepositorioUsuario _repoUsuario;
      public AgregarComentario(IRepositorioEnvio repoEnvio, IRepositorioUsuario repoUsuario)
      {
         _repoEnvio = repoEnvio;
```

```
        _repoUsuario = repoUsuario;
    }
    void IAgregarComentario.Ejecutar(int idEnvio, int idEmpleado, string comentario)
    {
        Usuario usuario = _repoUsuario.GetById(idEmpleado) ?? throw new UsuarioException();
        Envio envio = _repoEnvio.GetById(idEnvio) ?? throw new EnvioException();
        if (envio.Estado == Estados.FINALIZADO)
            throw new EnvioException("No se puede agregar un comentario a un envio finalizado");
        List<Seguimiento> seguimientos = envio.ListaSeguimiento;
        seguimientos.Add(new Seguimiento(comentario, usuario));
        _repoEnvio.Update(envio);
    }
  }
}
```

****
Archivo: AltaEnvioComun.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaAplicacion\CasosUso\EnvioCU\AltaEnvioComun.cs
****

```
using Compartido.DTOs.Envio;
using Compartido.Mappers;
using LogicaAplicacion.InterfacesCasosUso.EnvioCU;
using LogicaNegocio.Entidades;
using LogicaNegocio.ExepcionesEntidades;
using LogicaNegocio.InterfacesRepositorio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.EnvioCU
{
  public class AltaEnvioComun : IAltaEnvioComun
  {
    private readonly IRepositorioEnvioComun _repoEnvioComun;
    private readonly IRepositorioUsuario _repoUsuario;
    private readonly IRepositorioAgencia _repoAgencia;
    public AltaEnvioComun(IRepositorioEnvioComun repoEnvioComun,
                IRepositorioUsuario repoUsuario,
                IRepositorioAgencia repoAgencia)
    {
      _repoEnvioComun = repoEnvioComun;
      _repoUsuario = repoUsuario;
      _repoAgencia = repoAgencia;
    }
    public void Ejecutar(EnvioComunDTO envioDTO, int idFuncionario)
```

```
      {
         ArgumentNullException.ThrowIfNull(envioDTO);
         Usuario cliente = _repoUsuario.GetById(envioDTO.ClienteId) ?? throw new
UsuarioException("Cliente no registrado.");
         Usuario funcionario = _repoUsuario.GetById(idFuncionario) ?? throw new
UsuarioException("Funcionario no registrado.");
         Agencia agencia = _repoAgencia.GetById(envioDTO.AgenciaId) ?? throw new
AgenciaException("La agencia no fue encontrada.");
         Comun envioComun = EnvioMapper.ComunFromEnvioDTO(envioDTO, cliente, funcionario,
agencia);
         _repoEnvioComun.Add(envioComun);

      }
   }
}

**********
Archivo: AltaEnvioUrgente.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaAplicacion\CasosUso\EnvioCU\AltaEnvioUrgente.cs
**********
using Compartido.DTOs.Envio;
using Compartido.Mappers;
using LogicaAplicacion.InterfacesCasosUso.EnvioCU;
using LogicaNegocio.Entidades;
using LogicaNegocio.ExepcionesEntidades;
using LogicaNegocio.InterfacesRepositorio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.EnvioCU
{
   public class AltaEnvioUrgente : IAltaEnvioUrgente
   {
      private readonly IRepositorioEnvioUrgente _repoEnvioUrgente;
      private readonly IRepositorioUsuario _repoUsuario;
      public AltaEnvioUrgente(IRepositorioEnvioUrgente repoEnvioUrgente,
                  IRepositorioUsuario repoUsuario)
      {
         _repoEnvioUrgente = repoEnvioUrgente;
         _repoUsuario = repoUsuario;
      }
      public void Ejecutar(EnvioUrgenteDTO envioDTO, int idFuncionario)
      {
         ArgumentNullException.ThrowIfNull(envioDTO);
```

```
        Usuario cliente = _repoUsuario.GetById(envioDTO.ClienteId) ?? throw new
UsuarioException("Cliente no registrado.");
        Usuario funcionario = _repoUsuario.GetById(idFuncionario) ?? throw new
UsuarioException("Funcionario no registrado.");
        Urgente envioUrgente = EnvioMapper.UrgenteFromEnvioDTO(envioDTO, cliente, funcionario);
        _repoEnvioUrgente.Add(envioUrgente);
      }
    }
}
```

****
Archivo: DetalleEnvio.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaAplicacion\CasosUso\EnvioCU\DetalleEnvio.cs
****

```
using Compartido.DTOs.Envio;
using LogicaAplicacion.InterfacesCasosUso.EnvioCU;
using LogicaNegocio.Entidades;
using LogicaNegocio.InterfacesRepositorio;
using LogicaNegocio.ExepcionesEntidades;
using Compartido.Mappers;

namespace LogicaAplicacion.CasosUso.EnvioCU
{
    public class DetalleEnvio : IDetalleEnvio
    {
        private readonly IRepositorioEnvio _repositorioEnvio;
        public DetalleEnvio(IRepositorioEnvio repositorioEnvio)
        {
            _repositorioEnvio = repositorioEnvio;
        }
        public EnvioDetalleDto Ejecutar(int id)
        {
            Envio envio = _repositorioEnvio.GetById(id) ?? throw new EnvioException();
            return EnvioMapper.EnvioTOEnvioDetalleDTO(envio);
        }
    }
}
```

****
Archivo: ListarEnvios.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaAplicacion\CasosUso\EnvioCU\ListarEnvios.cs
****

```
using Compartido.DTOs.Envio;
using Compartido.Mappers;
using LogicaAplicacion.InterfacesCasosUso.EnvioCU;
using LogicaNegocio.Entidades;
```

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

namespace LogicaAplicacion.CasosUso.EnvioCU
{
   public class ListarEnvios : IListarEnvios
   {
      private readonly IRepositorioEnvio _repoEnvio;
      private readonly IRepositorioUsuario _repoUsuario;

      public ListarEnvios(IRepositorioEnvio repositorioEnvio, IRepositorioUsuario repoUsuario)
      {
         _repoEnvio = repositorioEnvio;
         _repoUsuario = repoUsuario;
      }

      public List<EnvioListadoDTO> Ejecutar()
      {
         List<Usuario> usuarios = _repoUsuario.GetAll().ToList();
         List<Envio> envios = _repoEnvio.GetAll().ToList();
         return EnvioMapper.EnvioTOEnvioListadoDTO(envios, usuarios);
      }
   }
}
```

****
Archivo: ObtenerEnvio.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaAplicacion\CasosUso\EnvioCU\ObtenerEnvio.cs
****

```
using Compartido.DTOs.Envio;
using Compartido.Mappers;
using LogicaAplicacion.InterfacesCasosUso.EnvioCU;
using LogicaNegocio.Entidades;
using LogicaNegocio.ExepcionesEntidades;
using LogicaNegocio.InterfacesRepositorio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.EnvioCU
{
```

```
  public class ObtenerEnvio : IObtenerEnvio
  {
    private readonly IRepositorioEnvio _repoEnvio;
    public ObtenerEnvio(IRepositorioEnvio repoEnvio)
    {
      _repoEnvio = repoEnvio;
    }
    public EnvioAPInfoDto Ejecutar(string nroTracking)
    {
      if(String.IsNullOrEmpty(nroTracking))
        throw new ArgumentNullException("El numero de tracking no puede ser nulo o vacio.");
      Envio envio = _repoEnvio.GetByNroTracking(nroTracking) ?? throw new EnvioException("El
envio no existe.");
      return EnvioMapper.EnvioToEnvioAPInfoDto(envio);
    }
  }
}
```

****

Archivo: UpdateEnvio.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaAplicacion\CasosUso\EnvioCU\UpdateEnvio.cs

****

```
using Compartido.DTOs.Envio;
using Compartido.Mappers;
using LogicaAplicacion.InterfacesCasosUso.EnvioCU;
using LogicaNegocio.Entidades;
using LogicaNegocio.ExepcionesEntidades;
using LogicaNegocio.InterfacesRepositorio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.EnvioCU
{
  public class UpdateEnvio : IUpdateEnvio
  {
    private readonly IRepositorioEnvio _repoEnvio;
    public UpdateEnvio(IRepositorioEnvio repoEnvio)
    {
      _repoEnvio = repoEnvio;
    }
    public void Ejecutar(EnvioUpdateDTO envioDto)
    {
      ArgumentNullException.ThrowIfNull(envioDto);
```

```
        Envio envio = _repoEnvio.GetById(envioDto.Id) ?? throw new EnvioException("Envio no
encontrado.");
        envio.FinalizarEnvio();
        _repoEnvio.Update(envio);
      }
   }
}

**********
Archivo: AltaFuncionario.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaAplicacion\CasosUso\FuncionarioCU\AltaFuncionario.cs
**********
using LogicaNegocio.Entidades;
using LogicaNegocio.InterfacesRepositorio;
using Compartido.Mappers;
using LogicaAplicacion.InterfacesCasosUso.FuncionarioCU;
using LogicaNegocio.ExepcionesEntidades;
using Compartido.DTOs.Funcionario;

namespace LogicaAplicacion.CasosUso.FuncionarioCU
{
   public class AltaFuncionario : IAltaFuncionario
   {
      private readonly IRepositorioUsuario _repoUsuario;
      private readonly IRepositorioRol _repoRol;
      private readonly IRepositorioAuditoria _repoAuditoria;
      public AltaFuncionario(IRepositorioUsuario repoUsuario, IRepositorioRol repoRol,
IRepositorioAuditoria repoAuditoria)
      {
         _repoUsuario = repoUsuario;
         _repoRol = repoRol;
         _repoAuditoria = repoAuditoria;
      }
      public void Ejecutar(FuncionarioDTO funcionarioDTO, int idUsuario)
      {
         if (funcionarioDTO == null)
            throw new ArgumentNullException("Datos invalidos.");
         Rol? rol = _repoRol.GetById(funcionarioDTO.RolId) ?? throw new RolException("Rol no
encontrado.");
         Usuario funcionario = FuncionarioMapper.FuncionarioFromFuncionarioDTO(funcionarioDTO,
rol);
         Usuario administrador = _repoUsuario.GetById(idUsuario) ?? throw new
UsuarioException("Usuario no encontrado.");
         _repoUsuario.Add(funcionario);
         _repoAuditoria.Add(new Auditoria($"Alta de funcionario {funcionario.Id}", administrador));
      }
   }
```

```
}
```

```
**********
Archivo: DetalleFuncionario.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaAplicacion\CasosUso\FuncionarioCU\DetalleFuncionario.cs
**********
using Compartido.DTOs.Funcionario;
using Compartido.Mappers;
using LogicaAplicacion.InterfacesCasosUso.FuncionarioCU;
using LogicaNegocio.Entidades;
using LogicaNegocio.ExepcionesEntidades;
using LogicaNegocio.InterfacesRepositorio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.FuncionarioCU
{
  public class DetalleFuncionario : IDetalleFuncionario
  {
    private readonly IRepositorioUsuario _repoUsuario;
    private readonly IRepositorioRol _repoRol;
    public DetalleFuncionario(IRepositorioUsuario repoUsuario, IRepositorioRol repoRol)
    {
      _repoUsuario = repoUsuario;
      _repoRol = repoRol;
    }

    public FuncionarioDetailDTO Ejecutar(int id)
    {
      Usuario? funcionario = _repoUsuario.GetById(id) ?? throw new UsuarioException("Funcionario no encontrado.");
      return FuncionarioMapper.FuncionarioToFuncionarioDetailDTO(funcionario);
    }
  }
}
```

```
**********
Archivo: DetalleUpdateFuncionario.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaAplicacion\CasosUso\FuncionarioCU\DetalleUpdateFuncionario.cs
**********
using Compartido.DTOs.Funcionario;
using Compartido.Mappers;
```

```
using LogicaAplicacion.InterfacesCasosUso.FuncionarioCU;
using LogicaNegocio.Entidades;
using LogicaNegocio.ExepcionesEntidades;
using LogicaNegocio.InterfacesRepositorio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.FuncionarioCU
{
   public class DetalleUpdateFuncionario : IDetalleUpdateFuncionario
   {
      private readonly IRepositorioUsuario _repoUsuario;
      public DetalleUpdateFuncionario(IRepositorioUsuario repoUsuario)
      {
         _repoUsuario = repoUsuario;
      }
      public FuncionarioUpdateDTO Ejecutar(int id)
      {
         Usuario usuario = _repoUsuario.GetById(id) ?? throw new UsuarioException("Funcionario no
encontrado.");
         return FuncionarioMapper.FuncionarioToFuncionarioUpdateDTO(usuario);
      }
   }
}
```

****

Archivo: EliminarFuncionario.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaAplicacion\CasosUso\FuncionarioCU\EliminarFuncionario.cs

****

```
using LogicaAplicacion.InterfacesCasosUso.FuncionarioCU;
using LogicaNegocio.Entidades;
using LogicaNegocio.ExepcionesEntidades;
using LogicaNegocio.InterfacesRepositorio;

namespace LogicaAplicacion.CasosUso.FuncionarioCU
{
   public class EliminarFuncionario : IEliminarFuncionario
   {
      private readonly IRepositorioUsuario _repoUsuario;
      private readonly IRepositorioAuditoria _repoAuditoria;
      public EliminarFuncionario(IRepositorioUsuario repoUsuario, IRepositorioAuditoria
repoAuditoria)
      {
         _repoUsuario = repoUsuario;
```

```
        _repoAuditoria = repoAuditoria;
    }
    public void Ejecutar(int id, int idAdmin)
    {
        Usuario usuario = _repoUsuario.GetById(id) ?? throw new UsuarioException("El funcionario a
eliminar no existe.");
        Usuario administrador = _repoUsuario.GetById(idAdmin) ?? throw new
UsuarioException("Administrador no encontrado.");
        _repoUsuario.Delete(usuario);
        _repoAuditoria.Add(new Auditoria($"Eliminacion de funcionario {usuario.Id}", administrador));
    }
  }
}
```

****
Archivo: ListarFuncionarios.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaAplicacion\CasosUso\FuncionarioCU\ListarFuncionarios.cs
****

```
using LogicaNegocio.Entidades;
using LogicaNegocio.InterfacesRepositorio;
using LogicaNegocio.ExepcionesEntidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using LogicaAplicacion.InterfacesCasosUso.FuncionarioCU;
using Compartido.Mappers;
using Compartido.DTOs.Funcionario;

namespace LogicaAplicacion.CasosUso.FuncionarioCU
{
  public class ListarFuncionarios : IListarFuncionarios
  {
    private readonly IRepositorioUsuario _repoUsuario;
    private readonly IRepositorioRol _repoRol;
    public ListarFuncionarios(IRepositorioUsuario repoFuncionario, IRepositorioRol repoRol)
    {
      _repoUsuario = repoFuncionario;
      _repoRol = repoRol;
    }
    public List<FuncionarioListarDTO> Ejecutar()
    {
      Rol? rol = _repoRol.GetByName("Cliente");
      List<Rol> rolList = _repoRol.GetAll().Where(r => r.Nombre != "Cliente").ToList();
      List<Usuario> funcionarios = _repoUsuario.GetAll().Where(u => u.RolId != rol.Id).ToList();
      return
```

```
            rol == null
            ?
            throw new RolException("El rol no fue encontrado.")
            :
            FuncionarioMapper.FuncionarioToFuncionarioListarDTO(funcionarios, rolList);
      }
   }
}
```

****
Archivo: UpdateFuncionario.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaAplicacion\CasosUso\FuncionarioCU\UpdateFuncionario.cs
****

```
using Compartido.DTOs.Funcionario;
using Compartido.Mappers;
using LogicaAplicacion.InterfacesCasosUso.FuncionarioCU;
using LogicaNegocio.Entidades;
using LogicaNegocio.ExepcionesEntidades;
using LogicaNegocio.InterfacesRepositorio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.FuncionarioCU
{
   public class UpdateFuncionario : IUpdateFuncionario
   {
      private readonly IRepositorioUsuario _repoUsuario;
      private readonly IRepositorioRol _repoRol;
      private readonly IRepositorioAuditoria _repoAuditoria;
      public UpdateFuncionario(IRepositorioUsuario repoUsuario, IRepositorioRol repoRol,
IRepositorioAuditoria repoAuditoria)
      {
         _repoUsuario = repoUsuario;
         _repoRol = repoRol;
         _repoAuditoria = repoAuditoria;
      }
      public void Ejecutar(FuncionarioUpdateDTO funcionarioDTO, int idAdmin)
      {
         if (funcionarioDTO == null)
            throw new ArgumentNullException("Datos vacios, no se puede actualizar los cambios.");
         Rol? rol = _repoRol.GetById(funcionarioDTO.RolId) ?? throw new RolException("El rol no
existe.");
         Usuario usuario =
FuncionarioMapper.FuncionarioFromFuncionarioUpdateDTO(funcionarioDTO, rol);
```

```
            Usuario administrador = _repoUsuario.GetById(idAdmin) ?? throw new
UsuarioException("Administrador no encontrado.");
            usuario.Id = funcionarioDTO.Id;
            _repoUsuario.Update(usuario);
            _repoAuditoria.Add(new Auditoria($"Actualizacion de funcionario {usuario.Id}",
administrador));
        }
    }
}

**********
Archivo: ListarSelectUsuario.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaAplicacion\CasosUso\UsuarioCU\ListarSelectUsuario.cs
**********
using Compartido.DTOs;
using Compartido.Mappers;
using LogicaAplicacion.InterfacesCasosUso.UsuarioCU;
using LogicaNegocio.Entidades;
using LogicaNegocio.InterfacesRepositorio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.CasosUso.UsuarioCU
{
    public class ListarSelectUsuario : IListarSelectUsuario
    {
        private readonly IRepositorioUsuario _repoUsuario;
        public ListarSelectUsuario(IRepositorioUsuario repoUsuario)
        {
            _repoUsuario = repoUsuario;
        }
        public List<InfoSelectDTO> Ejecutar()
        {
            List<Usuario> usuarios = [.. _repoUsuario.GetAll()];
            return UsuarioMapper.UsuarioToInfoSelectDto(usuarios);
        }
    }
}

**********
Archivo: IListarSAgencia.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaAplicacion\InterfacesCasosUso\AgenciaCU\IListarSAgencia.cs
**********
```

```
using Compartido.DTOs;

namespace LogicaAplicacion.InterfacesCasosUso.AgenciaCU
{
   public interface IListarSAgencia
   {
      List<InfoSelectDTO> Ejecutar();
   }
}
```

****
Archivo: IAgregarComentario.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaAplicacion\InterfacesCasosUso\EnvioCU\IAgregarComentario.cs
****

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

namespace LogicaAplicacion.InterfacesCasosUso.EnvioCU
{
   public interface IAgregarComentario
   {
      void Ejecutar(int idEnvio, int idEmpleado, string comentario);
   }
}
```

****
Archivo: IAltaEnvioComun.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaAplicacion\InterfacesCasosUso\EnvioCU\IAltaEnvioComun.cs
****

```
using Compartido.DTOs.Envio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.InterfacesCasosUso.EnvioCU
{
   public interface IAltaEnvioComun
   {
      void Ejecutar(EnvioComunDTO envioDTO, int idFuncionario);
   }
}
```

```
**********
Archivo: IAltaEnvioUrgente.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaAplicacion\InterfacesCasosUso\EnvioCU\IAltaEnvioUrgente.cs
**********
using Compartido.DTOs.Envio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.InterfacesCasosUso.EnvioCU
{
   public interface IAltaEnvioUrgente
   {
      void Ejecutar(EnvioUrgenteDTO envioDTO, int idFuncionario);
   }
}

**********
Archivo: IDetalleEnvio.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaAplicacion\InterfacesCasosUso\EnvioCU\IDetalleEnvio.cs
**********
using Compartido.DTOs.Envio;

namespace LogicaAplicacion.InterfacesCasosUso.EnvioCU
{
   public interface IDetalleEnvio
   {
      EnvioDetalleDto Ejecutar(int id);
   }
}

**********
Archivo: IListarEnvios.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaAplicacion\InterfacesCasosUso\EnvioCU\IListarEnvios.cs
**********
using Compartido.DTOs.Envio;

namespace LogicaAplicacion.InterfacesCasosUso.EnvioCU
{
   public interface IListarEnvios
   {
      List<EnvioListadoDTO> Ejecutar();
```

```
    }
}

**********
Archivo: IObtenerEnvio.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaAplicacion\InterfacesCasosUso\EnvioCU\IObtenerEnvio.cs
**********
using Compartido.DTOs.Envio;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.InterfacesCasosUso.EnvioCU
{
    public interface IObtenerEnvio
    {
        EnvioAPInfoDto Ejecutar(string nroTracking);
    }
}

**********
Archivo: IUpdateEnvio.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaAplicacion\InterfacesCasosUso\EnvioCU\IUpdateEnvio.cs
**********
using Compartido.DTOs.Envio;
namespace LogicaAplicacion.InterfacesCasosUso.EnvioCU
{
    public interface IUpdateEnvio
    {
        void Ejecutar(EnvioUpdateDTO envioDto);
    }
}

**********
Archivo: IAltaFuncionario.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaAplicacion\InterfacesCasosUso\FuncionarioCU\IAltaFuncionario.cs
**********
using Compartido.DTOs.Funcionario;

namespace LogicaAplicacion.InterfacesCasosUso.FuncionarioCU
{
    public interface IAltaFuncionario
    {
```

```
        void Ejecutar(FuncionarioDTO funcionarioDTO, int idUsuario);
    }
}

**********
Archivo: IDetalleFuncionario.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaAplicacion\InterfacesCasosUso\FuncionarioCU\IDetalleFuncionario.cs
**********
using Compartido.DTOs.Funcionario;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.InterfacesCasosUso.FuncionarioCU
{
    public interface IDetalleFuncionario
    {
        FuncionarioDetailDTO Ejecutar(int id);
    }
}

**********
Archivo: IDetalleUpdateFuncionario.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaAplicacion\InterfacesCasosUso\FuncionarioCU\IDetalleUpdateFuncionario.cs
**********
using Compartido.DTOs.Funcionario;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.InterfacesCasosUso.FuncionarioCU
{
    public interface IDetalleUpdateFuncionario
    {
        FuncionarioUpdateDTO Ejecutar(int id);
    }
}

**********
Archivo: IEliminarFuncionario.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaAplicacion\InterfacesCasosUso\FuncionarioCU\IEliminarFuncionario.cs
```

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

namespace LogicaAplicacion.InterfacesCasosUso.FuncionarioCU
{
    public interface IEliminarFuncionario
    {
        void Ejecutar(int id, int idAdmin);
    }
}

**********
Archivo: IListarFuncionarios.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaAplicacion\InterfacesCasosUso\FuncionarioCU\IListarFuncionarios.cs
**********
using Compartido.DTOs.Funcionario;

namespace LogicaAplicacion.InterfacesCasosUso.FuncionarioCU
{
    public interface IListarFuncionarios
    {
        List<FuncionarioListarDTO> Ejecutar();
    }
}

**********
Archivo: IUpdateFuncionario.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\LogicaAplicacion\InterfacesCasosUso\FuncionarioCU\IUpdateFuncionario.cs
**********
using Compartido.DTOs.Funcionario;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace LogicaAplicacion.InterfacesCasosUso.FuncionarioCU
{
    public interface IUpdateFuncionario
    {
        void Ejecutar(FuncionarioUpdateDTO funcionarioDTO, int idAdmin);
    }
```

```
}

**********
Archivo: IListarSelectUsuario.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\LogicaAplicacion\InterfacesCasosUso\UsuarioCU\IListarSelectUsuario.cs
**********
using Compartido.DTOs;
namespace LogicaAplicacion.InterfacesCasosUso.UsuarioCU
{
   public interface IListarSelectUsuario
   {
      List<InfoSelectDTO> Ejecutar();
   }
}

**********
Archivo: AgenciaSelectViewModel.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Models\Agencia\AgenciaSelectViewModel.cs
**********
namespace MVC.Models.Agencia
{
   public class AgenciaSelectViewModel
   {
      public int Id { get; set; }
      public string Nombre { get; set; }
   }
}

**********
Archivo: ComentarioViewModel.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Models\Envio\ComentarioViewModel.cs
**********
namespace MVC.Models.Envio
{
   public class ComentarioViewModel
   {
      public int IdEnvio { get; set; }
      public int IdEmpleado { get; set; }
      public string Comentario { get; set; }
   }
}

**********
Archivo: EnvioDetalleViewModel.cs
```

Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Models\Envio\EnvioDetalleViewModel.cs
****
```
namespace MVC.Models.Envio
{
   public class EnvioDetalleViewModel
   {
      public int Id { get; set; }
      public string NroTracking { get; set; }
      public string Empleado { get; set; }
      public string Cliente { get; set; }
      public decimal Peso { get; set; }
      public string Estado { get; set; }
      public string TipoEnvio { get; set; }
      public DateTime FechaCreacion { get; set; }
      public DateTime FechaEntrega { get; set; }
      public List<string> Estados  = new List<string>();
   }
}
```

****
Archivo: EnvioListadoViewModel.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Models\Envio\EnvioListadoViewModel.cs
****
```
namespace MVC.Models.Envio
{
   public class EnvioListadoViewModel
   {
      public int Id { get; set; }
      public string NroTracking { get; set; }
      public string Empleado { get; set; }
      public string Cliente { get; set; }
      public decimal Peso { get; set; }
      public string Estado { get; set; }
      public string TipoEnvio { get; set; }
   }
}
```

****
Archivo: FuncionarioDetailViewModel.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Models\Funcionario\FuncionarioDetailViewModel.cs
****
```
namespace MVC.Models.Funcionario
{
   public class FuncionarioDetailViewModel
   {
```

```
        public int Id { get; set; }
        public string Nombre { get; set; }
        public string Apellido { get; set; }
        public string CI { get; set; }
        public string Celular { get; set; }
        public string Email { get; set; }
    }
}
```

****
Archivo: FuncionarioListarViewModel.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Models\Funcionario\FuncionarioListarViewModel.cs
****

```
namespace MVC.Models.Funcionario
{
    public class FuncionarioListarViewModel
    {
        public int Id { get; set; }
        public string Nombre { get; set; }
        public string Apellido { get; set; }
        public string CI { get; set; }
        public string Email { get; set; }
        public string Rol { get; set; }
    }
}
```

****
Archivo: FuncionarioUpdateViewModel.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Models\Funcionario\FuncionarioUpdateViewModel.cs
****

```
using MVC.Models.Rol;

namespace MVC.Models.Funcionario
{
    public class FuncionarioUpdateViewModel
    {
        public int Id { get; set; }
        public string Nombre { get; set; }
        public string Apellido { get; set; }
        public string CI { get; set; }
        public string Celular { get; set; }
        public string Email { get; set; }
        public string Password { get; set; }
        public int RolId { get; set; }
        public IEnumerable<RolViewModel> Roles { get; set; } = new List<RolViewModel>();
    }
```

```
}

**********
Archivo: FuncionarioViewModel.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Models\Funcionario\FuncionarioViewModel.cs
**********
using MVC.Models.Rol;

namespace MVC.Models.Funcionario
{
   public class FuncionarioViewModel
   {
      public string Nombre { get; set; }
      public string Apellido { get; set; }
      public string CI { get; set; }
      public string Celular { get; set; }
      public string Email { get; set; }
      public string Password { get; set; }
      public int RolId { get; set; }

      public IEnumerable<RolViewModel> Roles { get; set; } = new List<RolViewModel>();
   }
}

**********
Archivo: RolViewModel.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Models\Rol\RolViewModel.cs
**********
namespace MVC.Models.Rol
{
   public class RolViewModel
   {
      public int Id { get; set; }
      public string Nombre { get; set; }

   }
}

**********
Archivo: UsuarioController.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Models\Usuario\UsuarioController.cs
**********
namespace MVC.Models.Usuario
{
   public class UsuarioController
```

```
    {
    }
}

**********
Archivo: UsuarioLoginViewModel.cs
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Models\Usuario\UsuarioLoginViewModel.cs
**********
namespace MVC.Models.Usuario
{
    public class UsuarioLoginViewModel
    {
        public string Email { get; set; }
        public string Password { get; set; }
    }
}
```

Vista

```
**********************************
Archivo: _ViewImports.cshtml
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Views\_ViewImports.cshtml
**********************************
@using MVC
@using MVC.Models
@addTagHelper *, Microsoft.AspNetCore.Mvc.TagHelpers

**********************************
Archivo: _ViewStart.cshtml
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Views\_ViewStart.cshtml
**********************************
@{
    Layout = "_Layout";
}

**********************************
Archivo: Comentar.cshtml
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Views\Envio\Comentar.cshtml
**********************************
@model MVC.Models.Envio.ComentarioViewModel

@{
    ViewData["Title"] = "Comentar";
}

<h1>Comentar</h1>

<h4>ComentarioViewModel</h4>
<hr />
<div class="row">
    <div class="col-md-4">
        <form asp-action="Comentar">
            <div asp-validation-summary="ModelOnly" class="text-danger"></div>
            <div class="form-group">
                <label asp-for="IdEnvio" class="control-label"></label>
                <p>@Model.IdEnvio</p>
                <input asp-for="IdEnvio" class="form-control" style="display: none;"/>
                <span asp-validation-for="IdEnvio" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="IdEmpleado" class="control-label"></label>
                <p>@Model.IdEmpleado</p>
                <input asp-for="IdEmpleado" class="form-control" style="display: none;" />
```

```
            <span asp-validation-for="IdEmpleado" class="text-danger"></span>
        </div>
        <div class="form-group">
            <label asp-for="Comentario" class="control-label"></label>
            <input asp-for="Comentario" class="form-control" />
            <span asp-validation-for="Comentario" class="text-danger"></span>
        </div>
        <div class="form-group">
            <input type="submit" value="Create" class="btn btn-primary" />
        </div>
    </form>
  </div>
</div>

<div>
  <a asp-action="Index">Back to List</a>
</div>

@section Scripts {
  @{await Html.RenderPartialAsync("_ValidationScriptsPartial");}
}
```

****
Archivo: Create.cshtml
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Views\Envio\Create.cshtml
****

```
@model MVC.Models.EnvioCreateViewModel

@{
  ViewData["Title"] = "Create";
}

<h1>Create</h1>

<h4>Envio</h4>
<hr />
<div class="row">
  <div class="col-md-6">
    <form asp-action="Create" id="formulario" novalidate>
      <div asp-validation-summary="ModelOnly" class="text-danger"></div>

      <!-- Es urgente -->
      <div class="form-group form-check">
        <label class="form-check-label">
          <input class="form-check-input" asp-for="EsUrgente" id="esUrgenteCheck" />
@Html.DisplayNameFor(model => model.EsUrgente)
        </label>
```

```
        </div>

        <!-- Email del cliente -->
        <div class="form-group">
          <label asp-for="EmailCliente" class="control-label"></label>
          <select asp-for="EmailCliente" class="form-control" required>
            <option value="">Seleccione usuario</option>
            @foreach (var usuario in Model.Usuarios)
            {
              @if (usuario.Nombre != Context.Session.GetString("Email"))
              {
                <option value="@usuario.Id">@usuario.Nombre</option>
              }
            }
          </select>
          <span asp-validation-for="EmailCliente" class="text-danger"></span>
        </div>

        <!-- Campo Agencia o Dirección (dinámico) -->
        <div id="urgenteFields"></div>

        <!-- Peso -->
        <div class="form-group">
          <label asp-for="Peso" class="control-label"></label>
          <input asp-for="Peso" class="form-control" required />
          <span asp-validation-for="Peso" class="text-danger"></span>
        </div>

        <div class="form-group">
          <input type="submit" value="Crear" class="btn btn-primary" />
        </div>
      </form>
    </div>
</div>

<p>@ViewBag.Msg</p>

<div>
    <a asp-action="Index">Volver al listado</a>
</div>

@section Scripts {
    @{
      await Html.RenderPartialAsync("_ValidationScriptsPartial");
    }

    <script>
      const check = document.getElementById("esUrgenteCheck");
```

```
    const container = document.getElementById("urgenteFields");

    function actualizarCampos() {
      if (check.checked) {
        // Mostrar dirección y ocultar agencia
        container.innerHTML = `
                    <div class="form-group">
                      <label for="DireccionPostal" class="control-label">Dirección Postal</label>
                      <input type="text" name="DireccionPostal" class="form-control"
id="DireccionPostal" required />
                      <span class="text-danger field-validation-valid" data-valmsg-
for="DireccionPostal" data-valmsg-replace="true"></span>
                    </div>
                  `;
      } else {
        // Mostrar agencia y ocultar dirección
        container.innerHTML = `
                    <div class="form-group">
                      <label for="AgenciaId" class="control-label">Agencia</label>
                      <select name="AgenciaId" class="form-control" id="AgenciaId" required>
                        <option value="">Seleccione agencia</option>
    @foreach (var agencia in Model.Agencias)
    {
                                <option value="@agencia.Id">@agencia.Nombre</option>
    }
                      </select>
                      <span class="text-danger field-validation-valid" data-valmsg-for="AgenciaId"
data-valmsg-replace="true"></span>
                    </div>
                  `;
      }
    }

    check.addEventListener("change", actualizarCampos);

    // Al cargar la vista, inicializamos los campos
    actualizarCampos();
  </script>
}

**********
Archivo: Edit.cshtml
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\MVC\Views\Envio\Edit.cshtml
**********
@model MVC.Models.Envio.EnvioDetalleViewModel

@{
```

```
    ViewData["Title"] = "Edit";
}

<h1>Edit</h1>

<h4>EnvioDetalleViewModel</h4>
<hr />
<div class="row">
    <div class="col-md-4">
        <form asp-action="Edit">
            <div asp-validation-summary="ModelOnly" class="text-danger"></div>
            <div class="form-group">
                <label asp-for="Id" class="control-label"></label>
                <p>@Model.Id</p>
                <input asp-for="Id" class="form-control" style="display: none" />
                <span asp-validation-for="Id" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="NroTracking" class="control-label"></label>
                <p>@Model.NroTracking</p>
                <input asp-for="NroTracking" class="form-control" style="display:none" />
                <span asp-validation-for="NroTracking" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="Empleado" class="control-label"></label>
                <p>@Model.Empleado</p>
                <input asp-for="Empleado" class="form-control" style="display: none" />
                <span asp-validation-for="Empleado" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="Cliente" class="control-label"></label>
                <p>@Model.Cliente</p>
                <input asp-for="Cliente" class="form-control" style="display: none" />
                <span asp-validation-for="Cliente" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="Peso" class="control-label"></label>
                <p>@Model.Peso</p>
                <input asp-for="Peso" class="form-control" style="display: none" />
                <span asp-validation-for="Peso" class="text-danger"></span>
            </div>

            <div class="form-group">
                <label asp-for="Estado" class="control-label"></label>
                <p>@Model.Estado</p>
                <input asp-for="Estado" class="form-control" style="display: none" />
                <span asp-validation-for="Estado" class="text-danger"></span>
            </div>
```

```
        <div class="form-group">
            <label asp-for="TipoEnvio" class="control-label"></label>
            <p>@Model.TipoEnvio</p>
            <input asp-for="TipoEnvio" class="form-control" style="display: none" />
            <span asp-validation-for="TipoEnvio" class="text-danger"></span>
        </div>
        <div class="form-group">
            <label asp-for="FechaCreacion" class="control-label"></label>
            <p>@Model.FechaCreacion</p>
            <input asp-for="FechaCreacion" class="form-control" style="display: none" />
            <span asp-validation-for="FechaCreacion" class="text-danger"></span>
        </div>
        @if (!(Model.FechaEntrega == DateTime.MinValue))
        {
            <div class="form-group">
                <label asp-for="FechaEntrega" class="control-label"></label>
                <p>@Model.FechaEntrega</p>
                <input asp-for="FechaEntrega" class="form-control" style="display: none" />
                <span asp-validation-for="FechaEntrega" class="text-danger"></span>
            </div>
        }
        <div class="form-group">
            <input type="submit" value="Finalizar" class="btn btn-primary" />
        </div>
    </form>
  </div>
</div>

<div>
  <a asp-action="Index">Back to List</a>
</div>

@section Scripts {
  @{
    await Html.RenderPartialAsync("_ValidationScriptsPartial");
  }
}

**********
Archivo: Edit2.cshtml
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\MVC\Views\Envio\Edit2.cshtml
**********
@model MVC.Models.Envio.EnvioDetalleViewModel

@{
  ViewData["Title"] = "Edit";
}
```

```
<h1>Edit</h1>

<h4>EnvioDetalleViewModel</h4>
<hr />
<div class="row">
  <div class="col-md-4">
    <form asp-action="Edit">
      <div asp-validation-summary="ModelOnly" class="text-danger"></div>
      <div class="form-group">
        <label asp-for="Id" class="control-label"></label>
        <p>@Model.Id</p>
        <input asp-for="Id" class="form-control" style="display: none"/>
        <span asp-validation-for="Id" class="text-danger"></span>
      </div>
      <div class="form-group">
        <label asp-for="NroTracking" class="control-label"></label>
        <p>@Model.NroTracking</p>
        <input asp-for="NroTracking" class="form-control" style="display:none" />
        <span asp-validation-for="NroTracking" class="text-danger"></span>
      </div>
      <div class="form-group">
        <label asp-for="Empleado" class="control-label"></label>
        <p>@Model.Empleado</p>
        <input asp-for="Empleado" class="form-control" style="display: none"/>
        <span asp-validation-for="Empleado" class="text-danger"></span>
      </div>
      <div class="form-group">
        <label asp-for="Cliente" class="control-label"></label>
        <p>@Model.Cliente</p>
        <input asp-for="Cliente" class="form-control" style="display: none"/>
        <span asp-validation-for="Cliente" class="text-danger"></span>
      </div>
      <div class="form-group">
        <label asp-for="Peso" class="control-label"></label>
        <p>@Model.Peso</p>
        <input asp-for="Peso" class="form-control" style="display: none"/>
        <span asp-validation-for="Peso" class="text-danger"></span>
      </div>

      <div class="form-group">
        <label asp-for="Estado" class="control-label"></label>
        <select asp-for="Estado" class="form-control">
          <option value="">Seleccione agencia</option>
          @foreach (var estado in Model.Estados)
          {
            <option value="@estado">@estado</option>
          }
```

```
            </select>
            <span asp-validation-for="Estado" class="text-danger"></span>
        </div>
        <div class="form-group">
            <label asp-for="TipoEnvio" class="control-label"></label>
            <p>@Model.TipoEnvio</p>
            <input asp-for="TipoEnvio" class="form-control" style="display: none"/>
            <span asp-validation-for="TipoEnvio" class="text-danger"></span>
        </div>
        <div class="form-group">
            <input type="submit" value="Save" class="btn btn-primary" />
        </div>
    </form>
  </div>
  <p>@ViewBag.Msg</p>
</div>

<div>
  <a asp-action="Index">Back to List</a>
</div>

@section Scripts {
  @{await Html.RenderPartialAsync("_ValidationScriptsPartial");}
}

**********
Archivo: Index.cshtml
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\MVC\Views\Envio\Index.cshtml
**********
@model IEnumerable<MVC.Models.Envio.EnvioListadoViewModel>

@{
  ViewData["Title"] = "Index";
}

<h1>Index</h1>

<p>
  <a asp-action="Create">Create New</a>
</p>
<table class="table">
  <thead>
    <tr>
      <th>
        @Html.DisplayNameFor(model => model.Id)
      </th>
      <th>
```

```
            @Html.DisplayNameFor(model => model.NroTracking)
          </th>
          <th>
            @Html.DisplayNameFor(model => model.Empleado)
          </th>
          <th>
            @Html.DisplayNameFor(model => model.Cliente)
          </th>
          <th>
            @Html.DisplayNameFor(model => model.Peso)
          </th>
          <th>
            @Html.DisplayNameFor(model => model.Estado)
          </th>
          <th>
            @Html.DisplayNameFor(model => model.TipoEnvio)
          </th>
          <th></th>
        </tr>
      </thead>
<tbody>
        @if (Model == null || !Model.Any())
        {
          <tr>
            <td colspan="8" class="text-center">No hay envios en proceso</td>
          </tr>
        }
        else
        {
          @foreach (var item in Model)
          {
            if (item.Estado == "EN_PROCESO")
            {
              <tr>
                <td>
                  @Html.DisplayFor(modelItem => item.Id)
                </td>
                <td>
                  @Html.DisplayFor(modelItem => item.NroTracking)
                </td>
                <td>
                  @Html.DisplayFor(modelItem => item.Empleado)
                </td>
                <td>
                  @Html.DisplayFor(modelItem => item.Cliente)
                </td>
                <td>
                  @Html.DisplayFor(modelItem => item.Peso)
```

```
                </td>
                <td>
                    @Html.DisplayFor(modelItem => item.Estado)
                </td>
                <td>
                    @Html.DisplayFor(modelItem => item.TipoEnvio)
                </td>
                <td>
                    @Html.ActionLink("Finalizar", "Edit", new { id = item.Id }) |
                    @* @Html.ActionLink("Details", "Details", new { id = item.Id }) | *@
                    @Html.ActionLink("Comentar", "Comentar", new { id = item.Id }) |
                </td>
            </tr>
        }
    }  }
</tbody></table>
```

****
Archivo: Index.cshtml
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Views\Home\Index.cshtml
****

```
@{
    ViewData["Title"] = "Home Page";
}

<div class="text-center">
    <h1 class="display-4">Welcome</h1>
    <p>Learn about <a href="https://learn.microsoft.com/aspnet/core">building Web apps with ASP.NET Core</a>.</p>
</div>
```

****
Archivo: Privacy.cshtml
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Views\Home\Privacy.cshtml
****

```
@{
    ViewData["Title"] = "Privacy Policy";
}
<h1>@ViewData["Title"]</h1>

<p>Use this page to detail your site's privacy policy.</p>
```

****
Archivo: Error.cshtml
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Views\Shared\Error.cshtml

```
**********************************
@model ErrorViewModel
@{
  ViewData["Title"] = "Error";
}

<h1 class="text-danger">Error.</h1>
<h2 class="text-danger">An error occurred while processing your request.</h2>

@if (Model.ShowRequestId)
{
  <p>
    <strong>Request ID:</strong> <code>@Model.RequestId</code>
  </p>
}

<h3>Development Mode</h3>
<p>
  Swapping to <strong>Development</strong> environment will display more detailed information
about the error that occurred.
</p>
<p>
  <strong>The Development environment shouldn't be enabled for deployed applications.</strong>
  It can result in displaying sensitive information from exceptions to end users.
  For local debugging, enable the <strong>Development</strong> environment by setting the
<strong>ASPNETCORE_ENVIRONMENT</strong> environment variable to
<strong>Development</strong>
  and restarting the app.
</p>

**********************************
Archivo: _Layout.cshtml
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\MVC\Views\Shared\_Layout.cshtml
**********************************
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>@ViewData["Title"] - MVC</title>
  <link rel="stylesheet" href="~/lib/bootstrap/dist/css/bootstrap.min.css" />
  <link rel="stylesheet" href="~/css/site.css" asp-append-version="true" />
  <link rel="stylesheet" href="~/MVC.styles.css" asp-append-version="true" />
</head>
<body>
  <header>
```

```
    <nav class="navbar navbar-expand-sm navbar-toggleable-sm navbar-light bg-white border-
bottom box-shadow mb-3">
        <div class="container-fluid">
            <a class="navbar-brand" asp-area="" asp-controller="Home" asp-action="Index">MVC</a>
            <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
target=".navbar-collapse" aria-controls="navbarSupportedContent"
            aria-expanded="false" aria-label="Toggle navigation">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="navbar-collapse collapse d-sm-inline-flex justify-content-between">
                <ul class="navbar-nav flex-grow-1">
                    <li class="nav-item">
                        <a class="nav-link text-dark" asp-area="" asp-controller="Home" asp-
action="Index">Home</a>
                    </li>
                    @if(String.IsNullOrEmpty(Context.Session.GetString("Rol")))
                    {
                        <li class="nav-item">
                            <a class="nav-link text-dark" asp-area="" asp-controller="Usuario" asp-
action="Login">Login</a>
                        </li>
                    } else
                    {
                        <li class="nav-item">
                            @if (Context.Session.GetString("Rol") == "Administrador")
                            {
                                <a class="nav-link text-dark" asp-area="" asp-controller="Usuario" asp-
action="Index">Empleados</a>
                            }
                        </li>
                        @if (Context.Session.GetString("Rol") != "Cliente" &&
!String.IsNullOrEmpty(Context.Session.GetString("Rol")))
                        {
                            <li class="nav-item">
                                <a class="nav-link text-dark" asp-area="" asp-controller="Envio" asp-
action="Index">Envio</a>
                            </li>
                        }
                        <li class="nav-item">
                            <a class="nav-link text-dark" asp-area="" asp-controller="Usuario" asp-
action="Logout">Logout</a>
                        </li>
                    }
                </ul>
            </div>
        </div>
    </nav>
</header>
```

```
    <div class="container">
        <main role="main" class="pb-3">
            @RenderBody()
        </main>
    </div>

    <footer class="border-top footer text-muted">
        <div class="container">
            &copy; 2025 - MVC - <a asp-area="" asp-controller="Home" asp-action="Privacy">Privacy</a>
        </div>
    </footer>
    <script src="~/lib/jquery/dist/jquery.min.js"></script>
    <script src="~/lib/bootstrap/dist/js/bootstrap.bundle.min.js"></script>
    <script src="~/js/site.js" asp-append-version="true"></script>
    @await RenderSectionAsync("Scripts", required: false)
</body>
</html>

**********
Archivo: _ValidationScriptsPartial.cshtml
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Views\Shared\_ValidationScriptsPartial.cshtml
**********
<script src="~/lib/jquery-validation/dist/jquery.validate.min.js"></script>
<script src="~/lib/jquery-validation-unobtrusive/jquery.validate.unobtrusive.min.js"></script>

**********
Archivo: Create.cshtml
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Views\Usuario\Create.cshtml
**********
@model MVC.Models.Funcionario.FuncionarioViewModel

@{
    ViewData["Title"] = "Create";
}

<h1>Create</h1>

<h4>FuncionarioViewModel</h4>
<hr />
<div class="row">
    <div class="col-md-4">
        <form asp-action="Create">
            <div asp-validation-summary="ModelOnly" class="text-danger"></div>
            <div class="form-group">
                <label asp-for="Nombre" class="control-label"></label>
                <input asp-for="Nombre" class="form-control" />
```

```
            <span asp-validation-for="Nombre" class="text-danger"></span>
        </div>
        <div class="form-group">
            <label asp-for="Apellido" class="control-label"></label>
            <input asp-for="Apellido" class="form-control" />
            <span asp-validation-for="Apellido" class="text-danger"></span>
        </div>
        <div class="form-group">
            <label asp-for="CI" class="control-label"></label>
            <input asp-for="CI" class="form-control" />
            <span asp-validation-for="CI" class="text-danger"></span>
        </div>
        <div class="form-group">
            <label asp-for="Celular" class="control-label"></label>
            <input asp-for="Celular" class="form-control" />
            <span asp-validation-for="Celular" class="text-danger"></span>
        </div>
        <div class="form-group">
            <label asp-for="Email" class="control-label"></label>
            <input asp-for="Email" class="form-control" />
            <span asp-validation-for="Email" class="text-danger"></span>
        </div>
        <div class="form-group">
            <label asp-for="Password" class="control-label"></label>
            <input asp-for="Password" class="form-control" />
            <span asp-validation-for="Password" class="text-danger"></span>
        </div>
        <div class="form-group">
            <label asp-for="RolId" class="control-label"></label>
            <select asp-for="RolId" class="form-control" >
                <option value="">Select a role</option>
                @foreach (var role in Model.Roles)
                {
                    if(role.Nombre != "Cliente")
                    {
                        <option value="@role.Id">@role.Nombre</option>
                    }
                }
            </select>
            <span asp-validation-for="RolId" class="text-danger"></span>
        </div>
        <div class="form-group">
            <input type="submit" value="Create" class="btn btn-primary" />
        </div>
    </form>
  </div>
  <p>@ViewBag.Msg</p>
</div>
```

```
<div>
    <a asp-action="Index">Back to List</a>
</div>

@section Scripts {
    @{await Html.RenderPartialAsync("_ValidationScriptsPartial");}
}

**********
Archivo: Delete.cshtml
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\MVC\Views\Usuario\Delete.cshtml
**********
@model MVC.Models.Funcionario.FuncionarioDetailViewModel

@{
    ViewData["Title"] = "Delete";
}

<h1>Delete</h1>

<h3>Are you sure you want to delete this?</h3>
<div>
    <h4>FuncionarioDetailViewModel</h4>
    <hr />
    <dl class="row">
        <dt class = "col-sm-2">
            @Html.DisplayNameFor(model => model.Id)
        </dt>
        <dd class = "col-sm-10">
            @Html.DisplayFor(model => model.Id)
        </dd>
        <dt class = "col-sm-2">
            @Html.DisplayNameFor(model => model.Nombre)
        </dt>
        <dd class = "col-sm-10">
            @Html.DisplayFor(model => model.Nombre)
        </dd>
        <dt class = "col-sm-2">
            @Html.DisplayNameFor(model => model.Apellido)
        </dt>
        <dd class = "col-sm-10">
            @Html.DisplayFor(model => model.Apellido)
        </dd>
        <dt class = "col-sm-2">
            @Html.DisplayNameFor(model => model.CI)
        </dt>
```

```
        <dd class = "col-sm-10">
            @Html.DisplayFor(model => model.CI)
        </dd>
        <dt class = "col-sm-2">
            @Html.DisplayNameFor(model => model.Celular)
        </dt>
        <dd class = "col-sm-10">
            @Html.DisplayFor(model => model.Celular)
        </dd>
        <dt class = "col-sm-2">
            @Html.DisplayNameFor(model => model.Email)
        </dt>
        <dd class = "col-sm-10">
            @Html.DisplayFor(model => model.Email)
        </dd>
    </dl>
    <p>@ViewBag.Msg</p>
    <form asp-action="Delete">
        <input type="submit" value="Delete" class="btn btn-danger" /> |
        <a asp-action="Index">Back to List</a>
    </form>
</div>

**********
Archivo: Details.cshtml
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Views\Usuario\Details.cshtml
**********
@model MVC.Models.Funcionario.FuncionarioDetailViewModel

@{
    ViewData["Title"] = "Details";
}

<h1>Details</h1>

<div>
    <h4>FuncionarioDetailViewModel</h4>
    <hr />
    <dl class="row">
        <dt class = "col-sm-2">
            @Html.DisplayNameFor(model => model.Id)
        </dt>
        <dd class = "col-sm-10">
            @Html.DisplayFor(model => model.Id)
        </dd>
        <dt class = "col-sm-2">
            @Html.DisplayNameFor(model => model.Nombre)
```

```
        </dt>
        <dd class = "col-sm-10">
            @Html.DisplayFor(model => model.Nombre)
        </dd>
        <dt class = "col-sm-2">
            @Html.DisplayNameFor(model => model.Apellido)
        </dt>
        <dd class = "col-sm-10">
            @Html.DisplayFor(model => model.Apellido)
        </dd>
        <dt class = "col-sm-2">
            @Html.DisplayNameFor(model => model.CI)
        </dt>
        <dd class = "col-sm-10">
            @Html.DisplayFor(model => model.CI)
        </dd>
        <dt class = "col-sm-2">
            @Html.DisplayNameFor(model => model.Celular)
        </dt>
        <dd class = "col-sm-10">
            @Html.DisplayFor(model => model.Celular)
        </dd>
        <dt class = "col-sm-2">
            @Html.DisplayNameFor(model => model.Email)
        </dt>
        <dd class = "col-sm-10">
            @Html.DisplayFor(model => model.Email)
        </dd>
    </dl>
    <p>@ViewBag.Msg</p>
</div>
<div>
    @Html.ActionLink("Edit", "Edit", new {  id = Model.Id  }) |
    <a asp-action="Index">Back to List</a>
</div>

**********
Archivo: Edit 1.cshtml
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\MVC\Views\Usuario\Edit 1.cshtml
**********
@model MVC.Models.Funcionario.FuncionarioViewModel

@{
    ViewData["Title"] = "Edit";
}

<h1>Edit</h1>
```

```
<h4>FuncionarioViewModel</h4>
<hr />
<div class="row">
  <div class="col-md-4">
    <form asp-action="Edit">
      <div asp-validation-summary="ModelOnly" class="text-danger"></div>
      <div class="form-group">
        <label asp-for="Nombre" class="control-label"></label>
        <input asp-for="Nombre" class="form-control" />
        <span asp-validation-for="Nombre" class="text-danger"></span>
      </div>
      <div class="form-group">
        <label asp-for="Apellido" class="control-label"></label>
        <input asp-for="Apellido" class="form-control" />
        <span asp-validation-for="Apellido" class="text-danger"></span>
      </div>
      <div class="form-group">
        <label asp-for="CI" class="control-label"></label>
        <input asp-for="CI" class="form-control" />
        <span asp-validation-for="CI" class="text-danger"></span>
      </div>
      <div class="form-group">
        <label asp-for="Celular" class="control-label"></label>
        <input asp-for="Celular" class="form-control" />
        <span asp-validation-for="Celular" class="text-danger"></span>
      </div>
      <div class="form-group">
        <label asp-for="Email" class="control-label"></label>
        <input asp-for="Email" class="form-control" />
        <span asp-validation-for="Email" class="text-danger"></span>
      </div>
      <div class="form-group">
        <label asp-for="Password" class="control-label"></label>
        <input asp-for="Password" class="form-control" />
        <span asp-validation-for="Password" class="text-danger"></span>
      </div>
      <div class="form-group">
        <input type="submit" value="Save" class="btn btn-primary" />
      </div>
    </form>
  </div>
</div>

<div>
  <a asp-action="Index">Back to List</a>
</div>
```

```
@section Scripts {
   @{await Html.RenderPartialAsync("_ValidationScriptsPartial");}
}

**********
Archivo: Edit.cshtml
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\MVC\Views\Usuario\Edit.cshtml
**********
@model MVC.Models.Funcionario.FuncionarioUpdateViewModel

@{
   ViewData["Title"] = "Edit";
}

<h1>Edit</h1>

<h4>FuncionarioUpdateViewModel</h4>
<hr />
<div class="row">
   <div class="col-md-4">
      <form asp-action="Edit">
         <div asp-validation-summary="ModelOnly" class="text-danger"></div>
         <div class="form-group" style="display:none">
            <label asp-for="Id" class="control-label"></label>
            <input asp-for="Id" class="form-control" />
            <span asp-validation-for="Id" class="text-danger"></span>
         </div>
         <div class="form-group">
            <label asp-for="Nombre" class="control-label"></label>
            <input asp-for="Nombre" class="form-control" />
            <span asp-validation-for="Nombre" class="text-danger"></span>
         </div>
         <div class="form-group">
            <label asp-for="Apellido" class="control-label"></label>
            <input asp-for="Apellido" class="form-control" />
            <span asp-validation-for="Apellido" class="text-danger"></span>
         </div>
         <div class="form-group">
            <label asp-for="CI" class="control-label"></label>
            <input asp-for="CI" class="form-control" />
            <span asp-validation-for="CI" class="text-danger"></span>
         </div>
         <div class="form-group">
            <label asp-for="Celular" class="control-label"></label>
            <input asp-for="Celular" class="form-control" />
            <span asp-validation-for="Celular" class="text-danger"></span>
         </div>
```

```
        <div class="form-group">
            <label asp-for="Email" class="control-label"></label>
            <input asp-for="Email" class="form-control" />
            <span asp-validation-for="Email" class="text-danger"></span>
        </div>
        <div class="form-group">
            <label asp-for="Password" class="control-label"></label>
            <input asp-for="Password" type="password" class="form-control"
value="@Model.Password"/>
            <span asp-validation-for="Password" class="text-danger"></span>
        </div>
        <div class="form-group">
            <label asp-for="RolId" class="control-label"></label>
            <select asp-for="RolId" class="form-control">
                <option value="">Select a role</option>
                @foreach (var role in Model.Roles)
                {
                    if (role.Nombre != "Cliente")
                    {
                        if(role.Id == Model.RolId)
                        {
                            <option value="@role.Id" selected>@role.Nombre</option>
                        }
                        else
                        {
                            <option value="@role.Id">@role.Nombre</option>
                        }
                    }
                }
            </select>
            <span asp-validation-for="RolId" class="text-danger"></span>
        </div>
        <div class="form-group">
            <input type="submit" value="Save" class="btn btn-primary" />
        </div>
    </form>
  </div>
  <p>@ViewBag.Msg</p>
</div>

<div>
  <a asp-action="Index">Back to List</a>
</div>

@section Scripts {
  @{await Html.RenderPartialAsync("_ValidationScriptsPartial");}
}
```

```
**********************************
Archivo: Index.cshtml
Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-
P3\MVC\Views\Usuario\Index.cshtml
**********************************
@model IEnumerable<MVC.Models.Funcionario.FuncionarioListarViewModel>

@{
   ViewData["Title"] = "Index";
}

<h1>Index</h1>

<p>
   <a asp-action="Create">Create New</a>
</p>
<table class="table">
   <thead>
      <tr>
         <th>
            @Html.DisplayNameFor(model => model.Id)
         </th>
         <th>
            @Html.DisplayNameFor(model => model.Nombre)
         </th>
         <th>
            @Html.DisplayNameFor(model => model.Apellido)
         </th>
         <th>
            @Html.DisplayNameFor(model => model.CI)
         </th>
         <th>
            @Html.DisplayNameFor(model => model.Email)
         </th>
         <th>
            @Html.DisplayNameFor(model => model.Rol)
         </th>
         <th></th>
      </tr>
   </thead>
   <tbody>
@foreach (var item in Model) {
      <tr>
         <td>
            @Html.DisplayFor(modelItem => item.Id)
         </td>
         <td>
            @Html.DisplayFor(modelItem => item.Nombre)
```

```
        </td>
        <td>
            @Html.DisplayFor(modelItem => item.Apellido)
        </td>
        <td>
            @Html.DisplayFor(modelItem => item.CI)
        </td>
        <td>
            @Html.DisplayFor(modelItem => item.Email)
        </td>
        <td>
            @Html.DisplayFor(modelItem => item.Rol)
        </td>
        <td>
            @Html.ActionLink("Edit", "Edit", new { id=item.Id }) |
            @Html.ActionLink("Details", "Details", new { id=item.Id }) |
            @Html.ActionLink("Delete", "Delete", new { id=item.Id })
        </td>
    </tr>
}
  </tbody>
</table>
```

****

Archivo: Login.cshtml

Carpeta: C:\Users\mario\OneDrive\Documentos\ORT\Programacion 3\Obligatorio-P3\MVC\Views\Usuario\Login.cshtml

****

```
@model MVC.Models.Usuario.UsuarioLoginViewModel

@{
  ViewData["Title"] = "Login";
}

<h1>Login</h1>

<h4>UsuarioLoginViewModel</h4>
<hr />
<div class="row">
  <div class="col-md-4">
    <form asp-action="Login">
      <div asp-validation-summary="ModelOnly" class="text-danger"></div>
      <div class="form-group">
        <label asp-for="Email" class="control-label"></label>
        <input asp-for="Email" class="form-control" />
        <span asp-validation-for="Email" class="text-danger"></span>
      </div>
      <div class="form-group">
```

```
            <label asp-for="Password" class="control-label"></label>
            <input asp-for="Password" class="form-control" type="password"/>
            <span asp-validation-for="Password" class="text-danger"></span>
          </div>
          <div class="form-group">
            <input type="submit" value="Login" class="btn btn-primary" />
          </div>
        </form>
        <p>@ViewBag.Msg</p>
    </div>
</div>

<div>
    <a asp-action="Index">Back to List</a>
</div>

@section Scripts {
    @{await Html.RenderPartialAsync("_ValidationScriptsPartial");}
}
```