**Universidad ORT Uruguay**

**Facultad de Ingeniería**

**Escuela de Tecnología**

**OBLIGATORIO PROGRAMACIÓN 3**

**DOCUMENTO**

Hombre con barba y bigote

Descripción generada automáticamente

**Nicolás Giménez – 291950**

Un hombre con una playera de color azul

Descripción generada automáticamente

**Cristian García – 317010**

**Grupo N3D**

**Docente: Liliana Pino**

**Analista en tecnologías de la información**

**21/10/2024**

​​**Índice**

[1. Diagrama de casos de uso. 3](#_Toc183696109)

[2. Diagrama de clases (UML). 4](#_Toc183696110)

[2.1 Lógica de negocio. 4](#_Toc183696111)

[2.2 Lógica de acceso a datos. 6](#_Toc183696112)

[2.3 Lógica de aplicación. 6](#_Toc183696113)

[2.4 Compartido. 9](#_Toc183696114)

[2.5 WebApi. 11](#_Toc183696115)

[2.6 MVC. 14](#_Toc183696116)

[3. Código fuente. 16](#_Toc183696117)

[4. Listas de datos y prompts. 251](#_Toc183696118)

[4.1 Listas de datos precargados. 251](#_Toc183696119)

[4.2 Prompts para generar datos. 270](#_Toc183696120)

# Diagrama de casos de uso.

Diagrama

Descripción generada automáticamente

# Diagrama de clases (UML).

## 2.1 Lógica de negocio.

Diagrama

Descripción generada automáticamente

Excepciones e interface de entity.

Diagrama

Descripción generada automáticamente

Interfaces de repositorios.

Diagrama

Descripción generada automáticamente

Entidades y value objects.

Diagrama

Descripción generada automáticamente

## 2.2 Lógica de acceso a datos.

Escala de tiempo

Descripción generada automáticamente

## 2.3 Lógica de aplicación.

Diagrama

Descripción generada automáticamenteUsuario.

Diagrama

Descripción generada automáticamenteAtleta.

Diagrama

Descripción generada automáticamenteEvento.

Diagrama

Descripción generada automáticamenteDisciplina.

Diagrama

Descripción generada automáticamentePaís.

Auditoría.

Diagrama

Descripción generada automáticamente

## 2.4 Compartido.

Usuario.

Diagrama

Descripción generada automáticamente

País.

Texto, Carta

Descripción generada automáticamente

Diagrama

Descripción generada automáticamenteAtleta, evento y disciplina.

Auditoría.

Diagrama

Descripción generada automáticamente

## 2.5 WebApi.

AtletaController.

Diagrama

Descripción generada automáticamente

DisciplinaController.

Diagrama

Descripción generada automáticamente

EventoController.

Diagrama

Descripción generada automáticamente

UsuarioController.

Diagrama

Descripción generada automáticamente

## 2.6 MVC.

UsuarioController, ErrorController y HomeController.

Diagrama

Descripción generada automáticamente

AtletaController.

Diagrama

Descripción generada automáticamente

EventoController.

Diagrama

Descripción generada automáticamente

DisciplinaController.

Diagrama

Descripción generada automáticamente

# Código fuente.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: JuegosOlimpicosDBContext.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAccesoDatos\JuegosOlimpicosDBContext.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaNegocio.Entidades;

using Microsoft.EntityFrameworkCore;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAccesoDatos

{

public class JuegosOlimpicosDBContext : DbContext

{

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

public DbSet<Atleta> Atletas { get; set; }

public DbSet<Disciplina> Disciplinas { get; set; }

public DbSet<Evento> Eventos { get; set; }

public DbSet<Pais> Paises { get; set; }

public DbSet<Auditoria> Auditorias { get; set; }

public JuegosOlimpicosDBContext(DbContextOptions opt) : base(opt)

{

//Pregunta si la base está creada

if (Database.EnsureCreated())

{

//Se ejecuta el método para cargar los datos desde el principio

InicializarDatos();

}

}

protected override void OnModelCreating(ModelBuilder modelBuilder)

{

modelBuilder.Entity<Usuario>()

.OwnsOne(u => u.Email)

.HasIndex(e => e.Valor)

.IsUnique();

modelBuilder.Entity<Disciplina>()

.OwnsOne(d => d.Nombre)

.HasIndex(n => n.Valor)

.IsUnique();

modelBuilder.Entity<Auditoria>()

.Property(a => a.Accion)

.HasConversion<string>();

}

private void InicializarDatos()

{

//Consultar si las tablas ya tienen datos, en caso de estar vacías se cargan datos

if (!Usuarios.Any())

{

EjecutarScript("Usuarios\_Admin.sql");

}

if (!Paises.Any())

{

EjecutarScript("Paises.sql");

}

if (!Atletas.Any())

{

EjecutarScript("Atletas.sql");

EjecutarScript("Disciplinas.sql");

EjecutarScript("AtletasDisciplina.sql");

}

if(!Eventos.Any())

{

EjecutarScript("Eventos.sql");

EjecutarScript("PuntajeEventoAtleta.sql");

}

}

private void EjecutarScript(string nombreScript)

{

//Ruta relativa para acceder a los scripts

string rutaCompleta = Path.Combine(Directory.GetCurrentDirectory(), "..", "ScriptsDatos", nombreScript);

//Leer el archivo

string sql = File.ReadAllText(rutaCompleta);

//Ejecutar el archivo en la base

Database.ExecuteSqlRaw(sql);

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: AtletaMapper.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\Mappers\AtletaMapper.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Atletas;

using LogicaNegocio.Entidades;

using LogicaNegocio.ExcepcionesEntidades;

using LogicaNegocio.Enums;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Compartido.Mappers

{

public static class AtletaMapper

{

public static IEnumerable<AtletaListaDTO> AtletasToListaDTO(List<Atleta> atletas)

{

return atletas.Select(a => new AtletaListaDTO

{

Id = a.Id,

Nombre = a.Nombre,

Apellido = a.Apellido,

NombreCompleto = a.Nombre + " " + a.Apellido,

NombrePais = a.Pais.Nombre,

Sexo = a.Sexo.ToString(),

});

}

public static AtletaDTO AtletaToDTO(Atleta atleta)

{

if (atleta == null)

{

throw new AtletaException("Atleta vacío en mapper");

}

AtletaDTO res = new AtletaDTO

{

Id = atleta.Id,

Nombre = atleta.Nombre,

Apellido = atleta.Apellido,

NombrePais = atleta.Pais.Nombre,

Sexo = atleta.Sexo.ToString(),

DisciplinasAtleta = DisciplinaMapper.DisciplinasToListaDTO(atleta.LiDisciplinas)

};

return res;

}

public static AtletaDTO AtletaToPuntajeEventoDTO(Atleta atleta)

{

if (atleta == null)

{

throw new AtletaException("Atleta vacío en mapper");

}

AtletaDTO res = new AtletaDTO

{

Id = atleta.Id,

Nombre = atleta.Nombre,

Apellido = atleta.Apellido,

NombrePais = atleta.Pais.Nombre,

Sexo = atleta.Sexo.ToString()

};

return res;

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: AuditoriaMapper.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\Mappers\AuditoriaMapper.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Auditorias;

using LogicaNegocio.Entidades;

using LogicaNegocio.Enums;

using LogicaNegocio.ExcepcionesEntidades;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Compartido.Mappers

{

public static class AuditoriaMapper

{

public static Auditoria DtoInsertToAuditoria(AuditoriaInsertDTO dto)

{

if (dto == null) throw new AuditoriaException("DTO vacío en Auditoria mapper");

return new Auditoria

{

Accion = dto.Accion,

EmailUsuario = dto.EmailUsuario,

Entidad = dto.Entidad,

EntidadId = dto.EntidadId,

Fecha = DateTime.Now

};

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: DisciplinaMapper.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\Mappers\DisciplinaMapper.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Disciplinas;

using LogicaNegocio.Entidades;

using LogicaNegocio.ExcepcionesEntidades;

using LogicaNegocio.ValueObjects.Disciplina;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Compartido.Mappers

{

public static class DisciplinaMapper

{

public static IEnumerable<DisciplinaDTO> DisciplinasToListaDTO(List<Disciplina> disciplinas)

{

IEnumerable<DisciplinaDTO> disciplinasListaDTO = disciplinas.Select(d => new DisciplinaDTO

{

Id = d.Id,

Nombre = d.Nombre.Valor,

AnioIntegracion = d.AnioIntegracion

});

return disciplinasListaDTO;

}

public static Disciplina InsertDTOToDisciplina(DisciplinaInsertDTO disciplinaInsertDTO)

{

if (disciplinaInsertDTO == null) throw new DisciplinaException("Discplina insert vacia en mapper");

Disciplina res = new Disciplina

{

Nombre = new RDisciplinaNombre(disciplinaInsertDTO.Nombre),

AnioIntegracion = disciplinaInsertDTO.AnioIntegracion,

};

return res;

}

public static Disciplina DTOToDisciplina(DisciplinaUpdateDTO dto)

{

if (dto == null) throw new DisciplinaException("Discplina update vacia en mapper");

return new Disciplina

{

Id = dto.Id,

Nombre = new RDisciplinaNombre(dto.Nombre),

AnioIntegracion = dto.AnioIntegracion

};

}

public static DisciplinaDTO DisciplinaToDTO(Disciplina disciplina)

{

if (disciplina == null) throw new DisciplinaException("Discplina vacia en mapper");

return new DisciplinaDTO

{

Id = disciplina.Id,

Nombre = disciplina.Nombre.Valor,

AnioIntegracion = disciplina.AnioIntegracion

};

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: EventoMapper.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\Mappers\EventoMapper.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Eventos;

using LogicaNegocio.Entidades;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using LogicaNegocio.ExcepcionesEntidades;

namespace Compartido.Mappers

{

public static class EventoMapper

{

public static Evento DtoToEvento(EventoInsertDTO dto)

{

if (dto == null) throw new EventoException("Evento insert DTO vacío en mapper");

Evento res = new Evento

{

DisciplinaId = dto.DisciplinaId,

FchInicio = dto.FchInicio,

FchFin = dto.FchFin,

NombrePrueba = dto.NombrePrueba,

LiPuntajes = dto.AtletasId.Select(id => new PuntajeEventoAtleta

{

AtletaId = id,

Puntaje = 0

}).ToList()

};

return res;

}

public static IEnumerable<EventoListaDTO> EventosToListaDTO(List<Evento> listaEventos)

{

IEnumerable<EventoListaDTO> eventos = listaEventos.Select(e => new EventoListaDTO

{

EventoId = e.Id,

NombrePrueba = e.NombrePrueba,

FchInicio = e.FchInicio,

FchFin = e.FchFin

});

return eventos;

}

public static EventoDTO EventoToDTO(Evento evento)

{

if (evento == null) throw new EventoException("Evento vacío en mapper");

EventoDTO eventoDto = new EventoDTO

{

Id = evento.Id,

FchInicio = evento.FchInicio,

FchFin = evento.FchFin,

NombrePrueba = evento.NombrePrueba,

LiAtletas = evento.LiPuntajes.Select(p => new PuntajeEventoAtletaDTO

{

Atleta = AtletaMapper.AtletaToPuntajeEventoDTO(p.Atleta),

Puntaje = p.Puntaje

})

};

return eventoDto;

}

public static List<PuntajeEventoAtleta> DtoListaModificadaToListaPuntaje(IEnumerable<PEAUpdateDTO> listaModificada)

{

if (listaModificada == null) throw new EventoException("Lista de puntajes DTO vacía en mapper");

return listaModificada.Select(p => new PuntajeEventoAtleta

{

AtletaId = p.AtletaId,

Puntaje = p.Puntaje,

}).ToList();

}

public static IEnumerable<AtletaEventoListaDTO> EventoToListaAtletaEventoDTO(List<Evento> lista)

{

if (lista == null) throw new EventoException("Lista de eventos vacía en mapper");

return lista.Select(e => new AtletaEventoListaDTO

{

Id = e.Id,

NombrePrueba = e.NombrePrueba,

NombreDisciplina = e.Disciplina.Nombre.Valor,

FchFin = e.FchFin,

FchInicio = e.FchInicio

});

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: PaisMapper.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\Mappers\PaisMapper.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Compartido.Mappers

{

public static class PaisMapper

{

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: UsuarioMapper.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\Mappers\UsuarioMapper.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Usuarios;

using LogicaNegocio.Entidades;

using LogicaNegocio.Enums;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using LogicaNegocio.ExcepcionesEntidades;

using LogicaNegocio.ValueObjects.Usuario;

namespace Compartido.Mappers

{

public static class UsuarioMapper

{

public static UsuarioDTO UsuarioToDTO(Usuario usuario)

{

if (usuario == null)

{

throw new UsuarioException("Error en mapper: el Usuario ingresado está vacío");

}

return new UsuarioDTO

{

Id = usuario.Id,

Email = usuario.Email.Valor,

RolUsuario = usuario.RolUsuario.ToString(),

Nombre = usuario.Nombre,

FechaRegistro = usuario.FechaRegistro.ToString("dd/MM/yyyy")

};

}

public static Usuario InsertDTOToUsuario(UsuarioInsertDTO usuarioInsertDto)

{

if (usuarioInsertDto == null)

{

throw new UsuarioException("Error en mapper: el Usuario insert está vacío");

}

return new Usuario

{

Email = new RUsuarioEmail(usuarioInsertDto.Email),

Nombre = usuarioInsertDto.Nombre,

Contrasena = new RUsuarioContrasena(usuarioInsertDto.Contrasena),

RolUsuario = usuarioInsertDto.RolUsuario == 0 ? Rol.Administrador : Rol.Digitador,

IdAdminRegistro = usuarioInsertDto.IdAdminRegistro

};

}

public static IEnumerable<UsuarioDTO> ListaUsuariosToDTOListaUsuarios(List<Usuario> usuarios)

{

IEnumerable<UsuarioDTO> usuariosDTOs = usuarios.Select(u => new UsuarioDTO()

{

Email = u.Email.Valor,

Nombre = u.Nombre,

Id = u.Id,

RolUsuario = u.RolUsuario.ToString(),

FechaRegistro = u.FechaRegistro.ToString("dd/MM/yyyy")

});

return usuariosDTOs;

}

public static UsuarioUpdateDTO UsuarioToUpdateDTO(Usuario usuario)

{

if (usuario == null)

{

throw new UsuarioException("Error en mapper: el Usuario está vacío");

}

return new UsuarioUpdateDTO

{

Email = usuario.Email.Valor,

Nombre = usuario.Nombre

};

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: 20241127023058\_init.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAccesoDatos\Migrations\20241127023058\_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: "Auditorias",

columns: table => new

{

Id = table.Column<int>(type: "int", nullable: false)

.Annotation("SqlServer:Identity", "1, 1"),

Fecha = table.Column<DateTime>(type: "datetime2", nullable: false),

Accion = table.Column<string>(type: "nvarchar(max)", nullable: false),

Entidad = table.Column<string>(type: "nvarchar(max)", nullable: false),

EntidadId = table.Column<int>(type: "int", nullable: false),

EmailUsuario = table.Column<string>(type: "nvarchar(max)", nullable: false)

},

constraints: table =>

{

table.PrimaryKey("PK\_Auditorias", x => x.Id);

});

migrationBuilder.CreateTable(

name: "Disciplinas",

columns: table => new

{

Id = table.Column<int>(type: "int", nullable: false)

.Annotation("SqlServer:Identity", "1, 1"),

Nombre\_Valor = table.Column<string>(type: "nvarchar(50)", maxLength: 50, nullable: false),

AnioIntegracion = table.Column<int>(type: "int", nullable: false)

},

constraints: table =>

{

table.PrimaryKey("PK\_Disciplinas", x => x.Id);

});

migrationBuilder.CreateTable(

name: "Paises",

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),

Habitantes = table.Column<int>(type: "int", nullable: false),

NombreDelegado = table.Column<string>(type: "nvarchar(max)", nullable: false),

TelDelegado = table.Column<string>(type: "nvarchar(max)", nullable: false)

},

constraints: table =>

{

table.PrimaryKey("PK\_Paises", x => x.Id);

});

migrationBuilder.CreateTable(

name: "Usuarios",

columns: table => new

{

Id = table.Column<int>(type: "int", nullable: false)

.Annotation("SqlServer:Identity", "1, 1"),

Email\_Valor = table.Column<string>(type: "nvarchar(255)", maxLength: 255, nullable: false),

RolUsuario = table.Column<int>(type: "int", nullable: false),

FechaRegistro = table.Column<DateTime>(type: "datetime2", nullable: false),

IdAdminRegistro = table.Column<int>(type: "int", nullable: false),

Nombre = table.Column<string>(type: "nvarchar(50)", maxLength: 50, nullable: true),

Contrasena\_Valor = table.Column<string>(type: "nvarchar(max)", nullable: false)

},

constraints: table =>

{

table.PrimaryKey("PK\_Usuarios", x => x.Id);

});

migrationBuilder.CreateTable(

name: "Eventos",

columns: table => new

{

Id = table.Column<int>(type: "int", nullable: false)

.Annotation("SqlServer:Identity", "1, 1"),

NombrePrueba = table.Column<string>(type: "nvarchar(max)", nullable: false),

DisciplinaId = table.Column<int>(type: "int", nullable: false),

FchInicio = table.Column<DateTime>(type: "datetime2", nullable: false),

FchFin = table.Column<DateTime>(type: "datetime2", nullable: false)

},

constraints: table =>

{

table.PrimaryKey("PK\_Eventos", x => x.Id);

table.ForeignKey(

name: "FK\_Eventos\_Disciplinas\_DisciplinaId",

column: x => x.DisciplinaId,

principalTable: "Disciplinas",

principalColumn: "Id",

onDelete: ReferentialAction.Cascade);

});

migrationBuilder.CreateTable(

name: "Atletas",

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),

Sexo = table.Column<int>(type: "int", nullable: false),

PaisId = table.Column<int>(type: "int", nullable: false)

},

constraints: table =>

{

table.PrimaryKey("PK\_Atletas", x => x.Id);

table.ForeignKey(

name: "FK\_Atletas\_Paises\_PaisId",

column: x => x.PaisId,

principalTable: "Paises",

principalColumn: "Id",

onDelete: ReferentialAction.Cascade);

});

migrationBuilder.CreateTable(

name: "AtletaDisciplina",

columns: table => new

{

LiAtletasId = table.Column<int>(type: "int", nullable: false),

LiDisciplinasId = table.Column<int>(type: "int", nullable: false)

},

constraints: table =>

{

table.PrimaryKey("PK\_AtletaDisciplina", x => new { x.LiAtletasId, x.LiDisciplinasId });

table.ForeignKey(

name: "FK\_AtletaDisciplina\_Atletas\_LiAtletasId",

column: x => x.LiAtletasId,

principalTable: "Atletas",

principalColumn: "Id",

onDelete: ReferentialAction.Cascade);

table.ForeignKey(

name: "FK\_AtletaDisciplina\_Disciplinas\_LiDisciplinasId",

column: x => x.LiDisciplinasId,

principalTable: "Disciplinas",

principalColumn: "Id",

onDelete: ReferentialAction.Cascade);

});

migrationBuilder.CreateTable(

name: "PuntajeEventoAtleta",

columns: table => new

{

AtletaId = table.Column<int>(type: "int", nullable: false),

EventoId = table.Column<int>(type: "int", nullable: false),

Puntaje = table.Column<decimal>(type: "decimal(18,2)", nullable: false)

},

constraints: table =>

{

table.PrimaryKey("PK\_PuntajeEventoAtleta", x => new { x.AtletaId, x.EventoId });

table.ForeignKey(

name: "FK\_PuntajeEventoAtleta\_Atletas\_AtletaId",

column: x => x.AtletaId,

principalTable: "Atletas",

principalColumn: "Id",

onDelete: ReferentialAction.Cascade);

table.ForeignKey(

name: "FK\_PuntajeEventoAtleta\_Eventos\_EventoId",

column: x => x.EventoId,

principalTable: "Eventos",

principalColumn: "Id",

onDelete: ReferentialAction.Cascade);

});

migrationBuilder.CreateIndex(

name: "IX\_AtletaDisciplina\_LiDisciplinasId",

table: "AtletaDisciplina",

column: "LiDisciplinasId");

migrationBuilder.CreateIndex(

name: "IX\_Atletas\_PaisId",

table: "Atletas",

column: "PaisId");

migrationBuilder.CreateIndex(

name: "IX\_Disciplinas\_Nombre\_Valor",

table: "Disciplinas",

column: "Nombre\_Valor",

unique: true);

migrationBuilder.CreateIndex(

name: "IX\_Eventos\_DisciplinaId",

table: "Eventos",

column: "DisciplinaId");

migrationBuilder.CreateIndex(

name: "IX\_PuntajeEventoAtleta\_EventoId",

table: "PuntajeEventoAtleta",

column: "EventoId");

migrationBuilder.CreateIndex(

name: "IX\_Usuarios\_Email\_Valor",

table: "Usuarios",

column: "Email\_Valor",

unique: true);

}

/// <inheritdoc />

protected override void Down(MigrationBuilder migrationBuilder)

{

migrationBuilder.DropTable(

name: "AtletaDisciplina");

migrationBuilder.DropTable(

name: "Auditorias");

migrationBuilder.DropTable(

name: "PuntajeEventoAtleta");

migrationBuilder.DropTable(

name: "Usuarios");

migrationBuilder.DropTable(

name: "Atletas");

migrationBuilder.DropTable(

name: "Eventos");

migrationBuilder.DropTable(

name: "Paises");

migrationBuilder.DropTable(

name: "Disciplinas");

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: 20241127023058\_init.Designer.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAccesoDatos\Migrations\20241127023058\_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(JuegosOlimpicosDBContext))]

[Migration("20241127023058\_init")]

partial class init

{

/// <inheritdoc />

protected override void BuildTargetModel(ModelBuilder modelBuilder)

{

#pragma warning disable 612, 618

modelBuilder

.HasAnnotation("ProductVersion", "8.0.11")

.HasAnnotation("Relational:MaxIdentifierLength", 128);

SqlServerModelBuilderExtensions.UseIdentityColumns(modelBuilder);

modelBuilder.Entity("AtletaDisciplina", b =>

{

b.Property<int>("LiAtletasId")

.HasColumnType("int");

b.Property<int>("LiDisciplinasId")

.HasColumnType("int");

b.HasKey("LiAtletasId", "LiDisciplinasId");

b.HasIndex("LiDisciplinasId");

b.ToTable("AtletaDisciplina");

});

modelBuilder.Entity("LogicaNegocio.Entidades.Atleta", 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>("Nombre")

.IsRequired()

.HasColumnType("nvarchar(max)");

b.Property<int>("PaisId")

.HasColumnType("int");

b.Property<int>("Sexo")

.HasColumnType("int");

b.HasKey("Id");

b.HasIndex("PaisId");

b.ToTable("Atletas");

});

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<string>("EmailUsuario")

.IsRequired()

.HasColumnType("nvarchar(max)");

b.Property<string>("Entidad")

.IsRequired()

.HasColumnType("nvarchar(max)");

b.Property<int>("EntidadId")

.HasColumnType("int");

b.Property<DateTime>("Fecha")

.HasColumnType("datetime2");

b.HasKey("Id");

b.ToTable("Auditorias");

});

modelBuilder.Entity("LogicaNegocio.Entidades.Disciplina", b =>

{

b.Property<int>("Id")

.ValueGeneratedOnAdd()

.HasColumnType("int");

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

b.Property<int>("AnioIntegracion")

.HasColumnType("int");

b.HasKey("Id");

b.ToTable("Disciplinas");

});

modelBuilder.Entity("LogicaNegocio.Entidades.Evento", b =>

{

b.Property<int>("Id")

.ValueGeneratedOnAdd()

.HasColumnType("int");

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

b.Property<int>("DisciplinaId")

.HasColumnType("int");

b.Property<DateTime>("FchFin")

.HasColumnType("datetime2");

b.Property<DateTime>("FchInicio")

.HasColumnType("datetime2");

b.Property<string>("NombrePrueba")

.IsRequired()

.HasColumnType("nvarchar(max)");

b.HasKey("Id");

b.HasIndex("DisciplinaId");

b.ToTable("Eventos");

});

modelBuilder.Entity("LogicaNegocio.Entidades.Pais", b =>

{

b.Property<int>("Id")

.ValueGeneratedOnAdd()

.HasColumnType("int");

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

b.Property<int>("Habitantes")

.HasColumnType("int");

b.Property<string>("Nombre")

.IsRequired()

.HasColumnType("nvarchar(max)");

b.Property<string>("NombreDelegado")

.IsRequired()

.HasColumnType("nvarchar(max)");

b.Property<string>("TelDelegado")

.IsRequired()

.HasColumnType("nvarchar(max)");

b.HasKey("Id");

b.ToTable("Paises");

});

modelBuilder.Entity("LogicaNegocio.Entidades.PuntajeEventoAtleta", b =>

{

b.Property<int>("AtletaId")

.HasColumnType("int");

b.Property<int>("EventoId")

.HasColumnType("int");

b.Property<decimal>("Puntaje")

.HasColumnType("decimal(18,2)");

b.HasKey("AtletaId", "EventoId");

b.HasIndex("EventoId");

b.ToTable("PuntajeEventoAtleta");

});

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

{

b.Property<int>("Id")

.ValueGeneratedOnAdd()

.HasColumnType("int");

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

b.Property<DateTime>("FechaRegistro")

.HasColumnType("datetime2");

b.Property<int>("IdAdminRegistro")

.HasColumnType("int");

b.Property<string>("Nombre")

.HasMaxLength(50)

.HasColumnType("nvarchar(50)");

b.Property<int>("RolUsuario")

.HasColumnType("int");

b.ComplexProperty<Dictionary<string, object>>("Contrasena", "LogicaNegocio.Entidades.Usuario.Contrasena#RUsuarioContrasena", b1 =>

{

b1.IsRequired();

b1.Property<string>("Valor")

.IsRequired()

.HasColumnType("nvarchar(max)");

});

b.HasKey("Id");

b.ToTable("Usuarios");

});

modelBuilder.Entity("AtletaDisciplina", b =>

{

b.HasOne("LogicaNegocio.Entidades.Atleta", null)

.WithMany()

.HasForeignKey("LiAtletasId")

.OnDelete(DeleteBehavior.Cascade)

.IsRequired();

b.HasOne("LogicaNegocio.Entidades.Disciplina", null)

.WithMany()

.HasForeignKey("LiDisciplinasId")

.OnDelete(DeleteBehavior.Cascade)

.IsRequired();

});

modelBuilder.Entity("LogicaNegocio.Entidades.Atleta", b =>

{

b.HasOne("LogicaNegocio.Entidades.Pais", "Pais")

.WithMany()

.HasForeignKey("PaisId")

.OnDelete(DeleteBehavior.Cascade)

.IsRequired();

b.Navigation("Pais");

});

modelBuilder.Entity("LogicaNegocio.Entidades.Disciplina", b =>

{

b.OwnsOne("LogicaNegocio.ValueObjects.Disciplina.RDisciplinaNombre", "Nombre", b1 =>

{

b1.Property<int>("DisciplinaId")

.HasColumnType("int");

b1.Property<string>("Valor")

.IsRequired()

.HasMaxLength(50)

.HasColumnType("nvarchar(50)");

b1.HasKey("DisciplinaId");

b1.HasIndex("Valor")

.IsUnique();

b1.ToTable("Disciplinas");

b1.WithOwner()

.HasForeignKey("DisciplinaId");

});

b.Navigation("Nombre")

.IsRequired();

});

modelBuilder.Entity("LogicaNegocio.Entidades.Evento", b =>

{

b.HasOne("LogicaNegocio.Entidades.Disciplina", "Disciplina")

.WithMany()

.HasForeignKey("DisciplinaId")

.OnDelete(DeleteBehavior.Cascade)

.IsRequired();

b.Navigation("Disciplina");

});

modelBuilder.Entity("LogicaNegocio.Entidades.PuntajeEventoAtleta", b =>

{

b.HasOne("LogicaNegocio.Entidades.Atleta", "Atleta")

.WithMany()

.HasForeignKey("AtletaId")

.OnDelete(DeleteBehavior.Cascade)

.IsRequired();

b.HasOne("LogicaNegocio.Entidades.Evento", null)

.WithMany("LiPuntajes")

.HasForeignKey("EventoId")

.OnDelete(DeleteBehavior.Cascade)

.IsRequired();

b.Navigation("Atleta");

});

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

{

b.OwnsOne("LogicaNegocio.ValueObjects.Usuario.RUsuarioEmail", "Email", b1 =>

{

b1.Property<int>("UsuarioId")

.HasColumnType("int");

b1.Property<string>("Valor")

.IsRequired()

.HasMaxLength(255)

.HasColumnType("nvarchar(255)");

b1.HasKey("UsuarioId");

b1.HasIndex("Valor")

.IsUnique();

b1.ToTable("Usuarios");

b1.WithOwner()

.HasForeignKey("UsuarioId");

});

b.Navigation("Email")

.IsRequired();

});

modelBuilder.Entity("LogicaNegocio.Entidades.Evento", b =>

{

b.Navigation("LiPuntajes");

});

#pragma warning restore 612, 618

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: JuegosOlimpicosDBContextModelSnapshot.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAccesoDatos\Migrations\JuegosOlimpicosDBContextModelSnapshot.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(JuegosOlimpicosDBContext))]

partial class JuegosOlimpicosDBContextModelSnapshot : ModelSnapshot

{

protected override void BuildModel(ModelBuilder modelBuilder)

{

#pragma warning disable 612, 618

modelBuilder

.HasAnnotation("ProductVersion", "8.0.11")

.HasAnnotation("Relational:MaxIdentifierLength", 128);

SqlServerModelBuilderExtensions.UseIdentityColumns(modelBuilder);

modelBuilder.Entity("AtletaDisciplina", b =>

{

b.Property<int>("LiAtletasId")

.HasColumnType("int");

b.Property<int>("LiDisciplinasId")

.HasColumnType("int");

b.HasKey("LiAtletasId", "LiDisciplinasId");

b.HasIndex("LiDisciplinasId");

b.ToTable("AtletaDisciplina");

});

modelBuilder.Entity("LogicaNegocio.Entidades.Atleta", 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>("Nombre")

.IsRequired()

.HasColumnType("nvarchar(max)");

b.Property<int>("PaisId")

.HasColumnType("int");

b.Property<int>("Sexo")

.HasColumnType("int");

b.HasKey("Id");

b.HasIndex("PaisId");

b.ToTable("Atletas");

});

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<string>("EmailUsuario")

.IsRequired()

.HasColumnType("nvarchar(max)");

b.Property<string>("Entidad")

.IsRequired()

.HasColumnType("nvarchar(max)");

b.Property<int>("EntidadId")

.HasColumnType("int");

b.Property<DateTime>("Fecha")

.HasColumnType("datetime2");

b.HasKey("Id");

b.ToTable("Auditorias");

});

modelBuilder.Entity("LogicaNegocio.Entidades.Disciplina", b =>

{

b.Property<int>("Id")

.ValueGeneratedOnAdd()

.HasColumnType("int");

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

b.Property<int>("AnioIntegracion")

.HasColumnType("int");

b.HasKey("Id");

b.ToTable("Disciplinas");

});

modelBuilder.Entity("LogicaNegocio.Entidades.Evento", b =>

{

b.Property<int>("Id")

.ValueGeneratedOnAdd()

.HasColumnType("int");

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

b.Property<int>("DisciplinaId")

.HasColumnType("int");

b.Property<DateTime>("FchFin")

.HasColumnType("datetime2");

b.Property<DateTime>("FchInicio")

.HasColumnType("datetime2");

b.Property<string>("NombrePrueba")

.IsRequired()

.HasColumnType("nvarchar(max)");

b.HasKey("Id");

b.HasIndex("DisciplinaId");

b.ToTable("Eventos");

});

modelBuilder.Entity("LogicaNegocio.Entidades.Pais", b =>

{

b.Property<int>("Id")

.ValueGeneratedOnAdd()

.HasColumnType("int");

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

b.Property<int>("Habitantes")

.HasColumnType("int");

b.Property<string>("Nombre")

.IsRequired()

.HasColumnType("nvarchar(max)");

b.Property<string>("NombreDelegado")

.IsRequired()

.HasColumnType("nvarchar(max)");

b.Property<string>("TelDelegado")

.IsRequired()

.HasColumnType("nvarchar(max)");

b.HasKey("Id");

b.ToTable("Paises");

});

modelBuilder.Entity("LogicaNegocio.Entidades.PuntajeEventoAtleta", b =>

{

b.Property<int>("AtletaId")

.HasColumnType("int");

b.Property<int>("EventoId")

.HasColumnType("int");

b.Property<decimal>("Puntaje")

.HasColumnType("decimal(18,2)");

b.HasKey("AtletaId", "EventoId");

b.HasIndex("EventoId");

b.ToTable("PuntajeEventoAtleta");

});

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

{

b.Property<int>("Id")

.ValueGeneratedOnAdd()

.HasColumnType("int");

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

b.Property<DateTime>("FechaRegistro")

.HasColumnType("datetime2");

b.Property<int>("IdAdminRegistro")

.HasColumnType("int");

b.Property<string>("Nombre")

.HasMaxLength(50)

.HasColumnType("nvarchar(50)");

b.Property<int>("RolUsuario")

.HasColumnType("int");

b.ComplexProperty<Dictionary<string, object>>("Contrasena", "LogicaNegocio.Entidades.Usuario.Contrasena#RUsuarioContrasena", b1 =>

{

b1.IsRequired();

b1.Property<string>("Valor")

.IsRequired()

.HasColumnType("nvarchar(max)");

});

b.HasKey("Id");

b.ToTable("Usuarios");

});

modelBuilder.Entity("AtletaDisciplina", b =>

{

b.HasOne("LogicaNegocio.Entidades.Atleta", null)

.WithMany()

.HasForeignKey("LiAtletasId")

.OnDelete(DeleteBehavior.Cascade)

.IsRequired();

b.HasOne("LogicaNegocio.Entidades.Disciplina", null)

.WithMany()

.HasForeignKey("LiDisciplinasId")

.OnDelete(DeleteBehavior.Cascade)

.IsRequired();

});

modelBuilder.Entity("LogicaNegocio.Entidades.Atleta", b =>

{

b.HasOne("LogicaNegocio.Entidades.Pais", "Pais")

.WithMany()

.HasForeignKey("PaisId")

.OnDelete(DeleteBehavior.Cascade)

.IsRequired();

b.Navigation("Pais");

});

modelBuilder.Entity("LogicaNegocio.Entidades.Disciplina", b =>

{

b.OwnsOne("LogicaNegocio.ValueObjects.Disciplina.RDisciplinaNombre", "Nombre", b1 =>

{

b1.Property<int>("DisciplinaId")

.HasColumnType("int");

b1.Property<string>("Valor")

.IsRequired()

.HasMaxLength(50)

.HasColumnType("nvarchar(50)");

b1.HasKey("DisciplinaId");

b1.HasIndex("Valor")

.IsUnique();

b1.ToTable("Disciplinas");

b1.WithOwner()

.HasForeignKey("DisciplinaId");

});

b.Navigation("Nombre")

.IsRequired();

});

modelBuilder.Entity("LogicaNegocio.Entidades.Evento", b =>

{

b.HasOne("LogicaNegocio.Entidades.Disciplina", "Disciplina")

.WithMany()

.HasForeignKey("DisciplinaId")

.OnDelete(DeleteBehavior.Cascade)

.IsRequired();

b.Navigation("Disciplina");

});

modelBuilder.Entity("LogicaNegocio.Entidades.PuntajeEventoAtleta", b =>

{

b.HasOne("LogicaNegocio.Entidades.Atleta", "Atleta")

.WithMany()

.HasForeignKey("AtletaId")

.OnDelete(DeleteBehavior.Cascade)

.IsRequired();

b.HasOne("LogicaNegocio.Entidades.Evento", null)

.WithMany("LiPuntajes")

.HasForeignKey("EventoId")

.OnDelete(DeleteBehavior.Cascade)

.IsRequired();

b.Navigation("Atleta");

});

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

{

b.OwnsOne("LogicaNegocio.ValueObjects.Usuario.RUsuarioEmail", "Email", b1 =>

{

b1.Property<int>("UsuarioId")

.HasColumnType("int");

b1.Property<string>("Valor")

.IsRequired()

.HasMaxLength(255)

.HasColumnType("nvarchar(255)");

b1.HasKey("UsuarioId");

b1.HasIndex("Valor")

.IsUnique();

b1.ToTable("Usuarios");

b1.WithOwner()

.HasForeignKey("UsuarioId");

});

b.Navigation("Email")

.IsRequired();

});

modelBuilder.Entity("LogicaNegocio.Entidades.Evento", b =>

{

b.Navigation("LiPuntajes");

});

#pragma warning restore 612, 618

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: RepositorioAtleta.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAccesoDatos\Repositorios\RepositorioAtleta.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaNegocio.Entidades;

using LogicaNegocio.IRepositorios;

using Microsoft.EntityFrameworkCore;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using LogicaNegocio.ExcepcionesEntidades;

namespace LogicaAccesoDatos.Repositorios

{

public class RepositorioAtleta : IRepositorioAtleta

{

private readonly JuegosOlimpicosDBContext \_context;

public RepositorioAtleta(JuegosOlimpicosDBContext context)

{

\_context = context;

}

public void Add(Atleta item)

{

throw new NotImplementedException();

}

public void Delete(Atleta item)

{

throw new NotImplementedException();

}

public List<Atleta> GetAll() =>

\_context.Atletas

.Include(a => a.Pais)

.OrderBy(a => a.Pais.Nombre)

.ThenBy(a => a.Apellido)

.ThenBy(a => a.Nombre)

.ToList();

public Atleta GetById(int id)

{

Atleta atleta = \_context.Atletas

.Include(a => a.Pais)

.Include(a => a.LiDisciplinas)

.SingleOrDefault(a => a.Id == id);

if (atleta == null)

{

throw new AtletaException("Atleta no encontrado por id");

}

return atleta;

}

public void Update(Atleta item)

{

throw new NotImplementedException();

}

public void GuardarCambios()

{

\_context.SaveChanges();

}

public List<Atleta>? SelectByDisciplinaId(int disciplinaId)

{

Disciplina? encontrado = \_context.Disciplinas

.Include(d => d.LiAtletas)

.ThenInclude(a => a.Pais)

.FirstOrDefault(d => d.Id == disciplinaId);

return encontrado == null

? null

: encontrado.LiAtletas;

}

//RF2 – Listado de Atletas por Disciplina(API Web + HttpClient) – Sin autenticación

//- Crear un listado de atletas filtrado por disciplina.Este endpoint permitirá consultar todos los atletas

//que están registrados en una disciplina dado su Id.El listado se retornará ordenado alfabéticamente

//por nombre completo de atleta.Se incluirán como mínimo su Id (o número, según haya utilizado), su

//nombre completo y el nombre de su país.

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: RepositorioAuditoria.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAccesoDatos\Repositorios\RepositorioAuditoria.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaNegocio.Entidades;

using LogicaNegocio.IRepositorios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAccesoDatos.Repositorios

{

public class RepositorioAuditoria:IRepositorioAuditoria

{

private readonly JuegosOlimpicosDBContext \_context;

public RepositorioAuditoria(JuegosOlimpicosDBContext context)

{

\_context = context;

}

public void Add(Auditoria item)

{

\_context.Add(item);

\_context.SaveChanges();

}

public void Delete(Auditoria item)

{

throw new NotImplementedException();

}

public List<Auditoria> GetAll()

{

throw new NotImplementedException();

}

public Auditoria? GetById(int id)

{

throw new NotImplementedException();

}

public void Update(Auditoria item)

{

throw new NotImplementedException();

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: RepositorioDisciplina.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAccesoDatos\Repositorios\RepositorioDisciplina.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaNegocio.Entidades;

using LogicaNegocio.ExcepcionesEntidades;

using LogicaNegocio.IRepositorios;

using Microsoft.EntityFrameworkCore;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAccesoDatos.Repositorios

{

public class RepositorioDisciplina : IRepositorioDisciplina

{

private readonly JuegosOlimpicosDBContext \_context;

public RepositorioDisciplina(JuegosOlimpicosDBContext context)

{

\_context = context;

}

public Disciplina Insert(Disciplina item)

{

\_context.Disciplinas.Add(item);

\_context.SaveChanges();

return item;

}

public void Delete(Disciplina item)

{

\_context.Disciplinas.Remove(item);

\_context.SaveChanges();

}

public List<Disciplina> GetAll() =>

\_context.Disciplinas

.OrderBy(d => d.Nombre.Valor)

.ToList();

public List<Atleta> GetAtletasDisciplina(int idDisciplina)

{

Disciplina buscar = \_context.Disciplinas

.Include(d => d.LiAtletas)

.ThenInclude(a => a.Pais)

.SingleOrDefault(d => d.Id == idDisciplina);

if (buscar == null) throw new DisciplinaException("Disciplina no encontrada con ese id");

return buscar.LiAtletas;

}

public Disciplina? GetById(int id) =>

\_context.Disciplinas

.Include(d => d.LiAtletas)

.SingleOrDefault(d => d.Id == id);

public void Update(Disciplina item)

{

\_context.Disciplinas.Update(item);

\_context.SaveChanges();

}

public Disciplina? GetByNombre(string nombre) =>

\_context.Disciplinas.SingleOrDefault(disiplina => disiplina.Nombre.Valor == nombre);

public void Add(Disciplina item)

{

\_context.Disciplinas.Add(item);

\_context.SaveChanges();

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: RepositorioEvento.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAccesoDatos\Repositorios\RepositorioEvento.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaNegocio.Entidades;

using LogicaNegocio.IRepositorios;

using Microsoft.EntityFrameworkCore;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAccesoDatos.Repositorios

{

public class RepositorioEvento : IRepositorioEvento

{

private readonly JuegosOlimpicosDBContext \_dbContext;

public RepositorioEvento(JuegosOlimpicosDBContext dbContext)

{

\_dbContext = dbContext;

}

public void Add(Evento item)

{

\_dbContext.Eventos.Add(item);

\_dbContext.SaveChanges();

}

public void Delete(Evento item)

{

throw new NotImplementedException();

}

public List<Evento> GetAll()

{

throw new NotImplementedException();

}

public Evento GetById(int id) =>

\_dbContext.Eventos

.Include(e => e.LiPuntajes)

.ThenInclude(p => p.Atleta)

.ThenInclude(a => a.Pais)

.SingleOrDefault(e => e.Id == id);

public Evento GetByNombre(string nombre) => \_dbContext.Eventos.SingleOrDefault(e => e.NombrePrueba == nombre);

public List<Evento> GetByFecha(DateTime fecha)

{

return \_dbContext.Eventos.Where(e => e.FchInicio <= fecha && e.FchFin >= fecha).ToList();

}

public void Update(Evento item)

{

\_dbContext.Eventos.Update(item);

\_dbContext.SaveChanges();

}

public List<Evento> GetEventosAtleta(int atletaId) =>

\_dbContext.Eventos

.Include(e => e.Disciplina)

.Where(e => e.LiPuntajes.Any(p => p.AtletaId == atletaId))

.OrderBy(e => e.Disciplina.Nombre.Valor)

.ToList();

public List<Evento> SelectByBusqueda

(int? disciplinaId, DateTime? fchInicio, DateTime? fchFin, string? nombreEvento, int? puntajeMin, int? puntajeMax)

{

IQueryable<Evento> eventos = \_dbContext.Eventos.AsQueryable();

if (disciplinaId.HasValue) eventos = eventos.Where(e => e.DisciplinaId == disciplinaId);

if (fchInicio.HasValue)

{

if (fchInicio == fchFin) eventos = eventos.Where(e => e.FchInicio == fchInicio || e.FchFin == fchInicio);

else eventos = eventos.Where(e => e.FchInicio >= fchInicio && e.FchFin <= fchFin);

}

if (!string.IsNullOrEmpty(nombreEvento)) eventos = eventos.Where(e => e.NombrePrueba.Contains(nombreEvento));

if (puntajeMin.HasValue)

eventos = eventos.Where(e => e.LiPuntajes.Any(pea => pea.Puntaje >= puntajeMin && pea.Puntaje <= puntajeMax));

return eventos.ToList();

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: RepositorioPais.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAccesoDatos\Repositorios\RepositorioPais.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaNegocio.Entidades;

using LogicaNegocio.IRepositorios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAccesoDatos.Repositorios

{

public class RepositorioPais : IRepositorioPais

{

private readonly JuegosOlimpicosDBContext \_context;

public RepositorioPais(JuegosOlimpicosDBContext context)

{

\_context = context;

}

public void Add(Pais item)

{

throw new NotImplementedException();

}

public void Delete(Pais item)

{

throw new NotImplementedException();

}

public List<Pais> GetAll()

{

throw new NotImplementedException();

}

public Pais GetById(int id)

{

throw new NotImplementedException();

}

public void Update(Pais item)

{

throw new NotImplementedException();

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: RepositorioUsuario.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAccesoDatos\Repositorios\RepositorioUsuario.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaNegocio.Entidades;

using LogicaNegocio.ExcepcionesEntidades;

using LogicaNegocio.IRepositorios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAccesoDatos.Repositorios

{

public class RepositorioUsuario : IRepositorioUsuario

{

private JuegosOlimpicosDBContext \_context;

public RepositorioUsuario(JuegosOlimpicosDBContext context)

{

\_context = context;

}

public void Add(Usuario item)

{

if (item == null)

{

throw new UsuarioException("El usuario no puede ser vacío");

}

\_context.Add(item);

\_context.SaveChanges();

}

public void Delete(Usuario item)

{

\_context.Remove(item);

\_context.SaveChanges();

}

public List<Usuario> GetAll()

{

return \_context.Usuarios.ToList();

}

public Usuario? GetById(int id) =>

\_context.Usuarios.SingleOrDefault(usuario => usuario.Id == id);

public Usuario? GetByEmail(string email) =>

\_context.Usuarios.SingleOrDefault(usuario => usuario.Email.Valor == email);

public void Update(Usuario item)

{

\_context.Usuarios.Update(item);

\_context.SaveChanges();

}

public Usuario? LoginUsuario(string email, string password)

{

List<Usuario> preRes = \_context.Usuarios

.Where(usuario => usuario.Email.Valor == email && usuario.Contrasena.Valor == password)

.ToList();

return preRes.FirstOrDefault(u => u.Email.Valor == email && u.Contrasena.Valor == password);

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: ValidarDisciplina.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\Validadores\ValidarDisciplina.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaNegocio.ExcepcionesEntidades;

using LogicaNegocio.ValueObjects.Disciplina;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.Validadores

{

public static class ValidarDisciplina

{

public static void Nombre(string nombre)

{

if (nombre.Length < 10) throw new DisciplinaException("El nombre de la disciplina debe contener al menos 10 caracteres");

if (nombre.Length > 50) throw new DisciplinaException("El nombre de la disciplina puede contener hasta 50 caracteres");

}

public static void Anio(int anio)

{

if (anio > DateTime.Now.Year) throw new DisciplinaException("El año no puede ser posterior al actual");

if(anio <= 0) throw new DisciplinaException("Año incorrecto");

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: ValidarEvento.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\Validadores\ValidarEvento.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Eventos;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using LogicaNegocio.ExcepcionesEntidades;

using LogicaNegocio.Entidades;

namespace LogicaAplicacion.Validadores

{

public static class ValidarEvento

{

public static void CantidadAtletas(EventoInsertDTO dto)

{

if (dto.AtletasId != null ? dto.AtletasId.Count() < 3 : true)

throw new EventoException("Se requieren al menos 3 Atletas");

}

public static void AtletasRegistradosEnDisciplina(List<Atleta> atletas, int[] dtoIds)

{

int verificador = atletas.Where(a => dtoIds.Any(id => id == a.Id)).Count();

if (verificador != dtoIds.Length) throw new EventoException("Hay atletas no registrados en esta Disciplina");

}

public static void Fechas(EventoInsertDTO dto)

{

if (dto.FchInicio > dto.FchFin)

throw new EventoException("Fechas incorrectas");

}

public static void Puntajes(EventoUpdatePuntajesDTO eventoUpdatePuntajesDTO)

{

if (eventoUpdatePuntajesDTO.LiAtletas.Any(p => p.Puntaje < 1))

throw new EventoException("Los puntajes no pueden ser menores a 1");

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: ValidarUsuario.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\Validadores\ValidarUsuario.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Usuarios;

using LogicaNegocio.Entidades;

using LogicaNegocio.Enums;

using LogicaNegocio.ExcepcionesEntidades;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

using static System.Runtime.InteropServices.JavaScript.JSType;

namespace LogicaAplicacion.Validadores

{

public static class ValidarUsuario

{

public static void Email(string email)

{

if (!Regex.IsMatch(email, @"^[a-zA-Z0-9]+@[a-zA-Z0-9]+\.[a-zA-Z0-9]+$"))

{

throw new UsuarioException("El formato del mail no es correcto");

}

if (email.Length > 255)

{

throw new UsuarioException("El email puede contener hasta 255 caracteres");

}

}

public static void Contrasena(string contrasena)

{

if (!Regex.IsMatch(contrasena, @"^(?=.\*[a-z])(?=.\*[A-Z])(?=.\*[0-9])(?=.\*[\.;,!]).{6,}$"))

{

throw new UsuarioException("La contraseña debe contener al menos 6 caracteres y al menos una mayúscula, minúscula, dígito y [. ; , !]");

}

}

public static void Rol(int rolId)

{

if (rolId < 0 || rolId > 1)

{

throw new UsuarioException("El rol solamente puede ser como Administrador o Digitador");

}

}

public static void Nombre(string nombre)

{

if (nombre != null && nombre.Length > 50)

{

throw new UsuarioException("El Nombre puede contener hasta 50 caracteres");

}

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: Atleta.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\Entidades\Atleta.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaNegocio.Enums;

using LogicaNegocio.IEntidades;

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 class Atleta : IEntity

{

public int Id { get; set; }

public string Nombre { get; set; }

public string Apellido { get; set; }

public Sexo Sexo { get; set; }

[ForeignKey("Pais")]

public int PaisId { get; set; }

public Pais Pais { get; set; }

public List<Disciplina> LiDisciplinas { get; set; }

public Atleta() { }

public Atleta(int id, string nombre, string apellido, Sexo sexo, Pais pais)

{

Id = id;

Nombre = nombre;

Apellido = apellido;

Sexo = sexo;

Pais = pais;

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: Auditoria.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\Entidades\Auditoria.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaNegocio.Enums;

using LogicaNegocio.IEntidades;

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("Auditorias")]

public class Auditoria : IEntity

{

public int Id { get ; set; }

public DateTime Fecha { get; set; }

public Accion Accion { get; set; }

public string Entidad { get; set; }

public int EntidadId { get; set; }

public string EmailUsuario { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: Disciplina.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\Entidades\Disciplina.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaNegocio.IEntidades;

using LogicaNegocio.ValueObjects.Disciplina;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaNegocio.Entidades

{

public class Disciplina : IEntity

{

public int Id { get; set; }

public RDisciplinaNombre Nombre { get; set; }

public int AnioIntegracion { get; set; }

public List<Atleta> LiAtletas { get; set; }

public Disciplina() { }

public Disciplina(int id, string nombre, int anioIntegracion)

{

Id = id;

Nombre = new RDisciplinaNombre(nombre);

AnioIntegracion = anioIntegracion;

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: Evento.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\Entidades\Evento.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaNegocio.IEntidades;

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 class Evento : IEntity

{

public int Id { get; set; }

public string NombrePrueba { get; set; }

public int DisciplinaId { get; set; }

public DateTime FchInicio { get; set; }

public DateTime FchFin { get; set; }

public List<PuntajeEventoAtleta> LiPuntajes { get; set; } = new List<PuntajeEventoAtleta>();

[ForeignKey("DisciplinaId")]

public Disciplina Disciplina { get; set; }

public Evento() { }

public Evento(int id, Disciplina disciplina, string nombrePrueba, DateTime fchInicio, DateTime fchFin)

{

Id = id;

Disciplina = disciplina;

NombrePrueba = nombrePrueba;

FchInicio = fchInicio;

FchFin = fchFin;

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: Pais.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\Entidades\Pais.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaNegocio.IEntidades;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaNegocio.Entidades

{

public class Pais : IEntity

{

public int Id { get; set; }

public string Nombre { get; set; }

public int Habitantes { get; set; }

public string NombreDelegado { get; set; }

public string TelDelegado { get; set; }

public Pais() { }

Pais(int id, string nombre, int habitantes, string nombreDelegado, string telDelegado)

{

Id = id;

Nombre = nombre;

Habitantes = habitantes;

NombreDelegado = nombreDelegado;

TelDelegado = telDelegado;

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: PuntajeEventoAtleta.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\Entidades\PuntajeEventoAtleta.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Microsoft.EntityFrameworkCore;

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations.Schema;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaNegocio.Entidades

{

[PrimaryKey(nameof(AtletaId), nameof(EventoId))]

public class PuntajeEventoAtleta

{

public decimal Puntaje { get; set; }

public int AtletaId { get; set; }

public int EventoId { get; set; }

[ForeignKey("AtletaId")]

public Atleta Atleta { get; set; }

public PuntajeEventoAtleta() { }

public PuntajeEventoAtleta(Atleta atleta, decimal puntaje)

{

Atleta = atleta;

Puntaje = puntaje;

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: Usuario.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\Entidades\Usuario.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaNegocio.Enums;

using LogicaNegocio.IEntidades;

using LogicaNegocio.ValueObjects.Usuario;

using Microsoft.EntityFrameworkCore;

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.Entidades

{

public class Usuario : IEntity

{

public int Id { get; set; }

public RUsuarioEmail Email { get; set; }

public RUsuarioContrasena Contrasena { get; set; }

public Rol RolUsuario { get; set; }

public DateTime FechaRegistro { get; private set; } = DateTime.Now;

public int IdAdminRegistro { get; set; }

[MaxLength(50)]

public string? Nombre { get; set; }

public Usuario() { }

public Usuario(string email, string contrasena, Rol rolUsuario, int idAdminRegistro)

{

Email = new RUsuarioEmail(email);

Contrasena = new RUsuarioContrasena(contrasena);

RolUsuario = rolUsuario;

IdAdminRegistro = idAdminRegistro;

}

public Usuario(string email, string contrasena, Rol rolUsuario, int idAdminRegistro, string nombre)

{

Email = new RUsuarioEmail(email);

Contrasena = new RUsuarioContrasena(contrasena);

RolUsuario = rolUsuario;

IdAdminRegistro = idAdminRegistro;

Nombre = nombre;

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: Accion.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\Enums\Accion.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaNegocio.Enums

{

public enum Accion

{

Create = 1,

Update = 2,

Delete = 3

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: Rol.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\Enums\Rol.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaNegocio.Enums

{

public enum Rol

{

Administrador = 0,

Digitador = 1

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: Sexo.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\Enums\Sexo.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaNegocio.Enums

{

public enum Sexo

{

Masculino = 0,

Femenino = 1

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: AtletaException.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\ExcepcionesEntidades\AtletaException.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaNegocio.ExcepcionesEntidades

{

public class AtletaException : Exception

{

public AtletaException() { }

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

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

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: AuditoriaException.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\ExcepcionesEntidades\AuditoriaException.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaNegocio.ExcepcionesEntidades

{

public class AuditoriaException : Exception

{

public AuditoriaException() { }

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

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

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: ConflictException.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\ExcepcionesEntidades\ConflictException.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaNegocio.ExcepcionesEntidades

{

public class ConflictException:Exception

{

public ConflictException() { }

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

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

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: DisciplinaException.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\ExcepcionesEntidades\DisciplinaException.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaNegocio.ExcepcionesEntidades

{

public class DisciplinaException : Exception

{

public DisciplinaException() { }

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

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

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: EventoException.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\ExcepcionesEntidades\EventoException.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaNegocio.ExcepcionesEntidades

{

public class EventoException : Exception

{

public EventoException() { }

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

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

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: PaisException.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\ExcepcionesEntidades\PaisException.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaNegocio.ExcepcionesEntidades

{

public class PaisException : Exception

{

public PaisException() { }

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

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

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: UsuarioException.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\ExcepcionesEntidades\UsuarioException.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaNegocio.ExcepcionesEntidades

{

public class UsuarioException : Exception

{

public UsuarioException() { }

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

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

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IEntity.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\IEntidades\IEntity.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaNegocio.IEntidades

{

public interface IEntity

{

int Id { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IRepositorio.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\IRepositorios\IRepositorio.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaNegocio.IRepositorios

{

public interface IRepositorio<T>

{

List<T> GetAll();

T? GetById(int id);

void Add(T item);

void Update(T item);

void Delete(T item);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IRepositorioAtleta.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\IRepositorios\IRepositorioAtleta.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaNegocio.Entidades;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaNegocio.IRepositorios

{

public interface IRepositorioAtleta : IRepositorio<Atleta>

{

public void GuardarCambios();

List<Atleta>? SelectByDisciplinaId(int disciplinaId);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IRepositorioAuditoria.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\IRepositorios\IRepositorioAuditoria.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaNegocio.Entidades;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaNegocio.IRepositorios

{

public interface IRepositorioAuditoria: IRepositorio<Auditoria>

{

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IRepositorioDisciplina.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\IRepositorios\IRepositorioDisciplina.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaNegocio.Entidades;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaNegocio.IRepositorios

{

public interface IRepositorioDisciplina : IRepositorio<Disciplina>

{

Disciplina Insert(Disciplina item);

Disciplina? GetByNombre(string nombre);

List<Atleta> GetAtletasDisciplina(int idDisciplina);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IRepositorioEvento.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\IRepositorios\IRepositorioEvento.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaNegocio.Entidades;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaNegocio.IRepositorios

{

public interface IRepositorioEvento : IRepositorio<Evento>

{

Evento GetByNombre(string nombre);

List<Evento> GetByFecha (DateTime fecha);

List<Evento> GetEventosAtleta(int atletaId);

List<Evento> SelectByBusqueda

(int? disciplinaId, DateTime? fchInicio, DateTime? fchFin, string? nombreEvento, int? puntajeMin, int? puntajeMax);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IRepositorioPais.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\IRepositorios\IRepositorioPais.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaNegocio.Entidades;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaNegocio.IRepositorios

{

public interface IRepositorioPais : IRepositorio<Pais>

{

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IRepositorioUsuario.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\IRepositorios\IRepositorioUsuario.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaNegocio.Entidades;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaNegocio.IRepositorios

{

public interface IRepositorioUsuario : IRepositorio<Usuario>

{

Usuario? LoginUsuario(string email, string password);

Usuario? GetByEmail(string email);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: AtletaController.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\WebApi\Controllers\AtletaController.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Atletas;

using Compartido.DTOs.Eventos;

using LogicaAplicacion.ICasosDeUso.Atletas;

using LogicaNegocio.ExcepcionesEntidades;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

namespace WebApi.Controllers

{

[Route("api/[controller]")]

[ApiController]

public class AtletaController : ControllerBase

{

private readonly ISelectByDisciplinaId \_selectByDisciplinaId;

public AtletaController(ISelectByDisciplinaId selectByDisciplinaId)

{

\_selectByDisciplinaId = selectByDisciplinaId;

}

/// <summary>

/// Permite obtener el listado de atletas de una disciplina

/// </summary>

/// <param name="disciplinaId"></param>

/// <returns></returns>

[ProducesResponseType(StatusCodes.Status200OK)]

[ProducesResponseType(StatusCodes.Status400BadRequest)]

[ProducesResponseType(StatusCodes.Status404NotFound)]

[ProducesResponseType(StatusCodes.Status500InternalServerError)]

[HttpGet("{disciplinaId}")]

public IActionResult GetByDisciplina(int disciplinaId)

{

try

{

IEnumerable<AtletaListaDTO> res = \_selectByDisciplinaId.Ejecutar(disciplinaId);

return res == null

? NotFound("No existe disciplina con esa id")

: Ok(res);

}

catch (AtletaException aex)

{

return BadRequest(aex.Message);

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

return StatusCode(500, "Algo no salió correctamente");

}

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: DisciplinaController.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\WebApi\Controllers\DisciplinaController.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Disciplinas;

using Compartido.DTOs.Auditorias;

using LogicaAplicacion.ICasosDeUso.Atletas;

using LogicaAplicacion.ICasosDeUso.Auditorias;

using LogicaAplicacion.ICasosDeUso.Disciplinas;

using LogicaNegocio.ExcepcionesEntidades;

using Microsoft.AspNetCore.Authorization;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using LogicaNegocio.Enums;

using LogicaNegocio.Entidades;

using System.Security.Claims;

namespace WebApi.Controllers

{

[Route("api/[controller]")]

[ApiController]

public class DisciplinaController : ControllerBase

{

private readonly IFindAllDisciplinas \_finAllDisciplinas;

private readonly IInsertDisciplina \_altaDisciplina;

private readonly IDeleteDisciplina \_deleteDisciplina;

private readonly IDisciplinaUpdate \_disciplinaUpdate;

private readonly IDisciplinaSelectById \_disciplinaSelectById;

private readonly IAuditoriaInsert \_auditoriaInsert;

private readonly IDisciplinaSelectByNombre \_disciplinaSelectByNombre;

public DisciplinaController(

IFindAllDisciplinas findAllDisciplinas,

IInsertDisciplina altaDisciplina,

IDeleteDisciplina deleteDisciplina,

IDisciplinaSelectById disciplinaSelectById,

IAuditoriaInsert auditoriaInsert,

IDisciplinaSelectByNombre disciplinaSelectByNombre,

IDisciplinaUpdate disciplinaUpdate)

{

\_finAllDisciplinas = findAllDisciplinas;

\_altaDisciplina = altaDisciplina;

\_deleteDisciplina = deleteDisciplina;

\_disciplinaSelectById = disciplinaSelectById;

\_auditoriaInsert = auditoriaInsert;

\_disciplinaSelectByNombre = disciplinaSelectByNombre;

\_disciplinaUpdate = disciplinaUpdate;

}

/// <summary>

/// Permite obtener una disciplina por ID

/// </summary>

/// <param name="id"></param>

/// <returns></returns>

[ProducesResponseType(StatusCodes.Status200OK)]

[ProducesResponseType(StatusCodes.Status400BadRequest)]

[ProducesResponseType(StatusCodes.Status404NotFound)]

[ProducesResponseType(StatusCodes.Status500InternalServerError)]

[Authorize(Roles = "Digitador")]

[HttpGet("{id}")]

public IActionResult Get(int id)

{

try

{

DisciplinaDTO? res = \_disciplinaSelectById.Ejecutar(id);

return res == null ? NotFound("No se encontró disciplina con ese Id") : Ok(res);

}

catch (DisciplinaException dex)

{

return BadRequest(dex.Message);

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

return StatusCode(500, "Algo no salió correctamente");

}

}

/// <summary>

/// Permite obtener una disciplina por nombre

/// </summary>

/// <param name="nombre"></param>

/// <returns></returns>

[ProducesResponseType(StatusCodes.Status200OK)]

[ProducesResponseType(StatusCodes.Status400BadRequest)]

[ProducesResponseType(StatusCodes.Status404NotFound)]

[ProducesResponseType(StatusCodes.Status500InternalServerError)]

[Authorize(Roles = "Digitador")]

[HttpGet("Nombre/{nombre}")]

public IActionResult GetByNombre(string nombre)

{

try

{

DisciplinaDTO? res = \_disciplinaSelectByNombre.Ejecutar(nombre);

return res == null ? NotFound("No se encontró disciplina con ese nombre") : Ok(res);

}

catch (DisciplinaException dex)

{

return BadRequest(dex.Message);

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

return StatusCode(500, "Algo no salió correctamente");

}

}

/// <summary>

/// Permite dar de alta una disciplina

/// </summary>

/// <param name="disciplinaInsertDto"></param>

/// <returns></returns>

[ProducesResponseType(StatusCodes.Status201Created)]

[ProducesResponseType(StatusCodes.Status400BadRequest)]

[ProducesResponseType(StatusCodes.Status409Conflict)]

[ProducesResponseType(StatusCodes.Status500InternalServerError)]

[Authorize(Roles = "Digitador")]

[HttpPost]

public IActionResult Post([FromBody] DisciplinaInsertDTO disciplinaInsertDto)

{

try

{

int idDisciplina = \_altaDisciplina.Ejecutar(disciplinaInsertDto);

\_auditoriaInsert.Ejecutar(new AuditoriaInsertDTO

{

Accion = Accion.Create,

EmailUsuario = User.FindFirst(ClaimTypes.Email)?.Value,

Entidad = "Disciplina",

EntidadId = idDisciplina

});

return Created();

}

catch (ConflictException cex)

{

return Conflict(cex.Message);

}

catch (DisciplinaException dex)

{

return BadRequest(dex.Message);

}

catch (AuditoriaException aex)

{

// Esto es un tema interno de servidor, si llego hasta \_auditoriaInsert.Ejecutar ya hizo la inserción

Console.WriteLine(aex.Message);

return Created();

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

return StatusCode(500, "Algo no salió correctamente");

}

}

/// <summary>

/// Permite modificar una disciplina

/// </summary>

/// <param name="id"></param>

/// <param name="disciplinaUpdateDTO"></param>

/// <returns></returns>

[ProducesResponseType(StatusCodes.Status200OK)]

[ProducesResponseType(StatusCodes.Status400BadRequest)]

[ProducesResponseType(StatusCodes.Status409Conflict)]

[ProducesResponseType(StatusCodes.Status500InternalServerError)]

[Authorize(Roles = "Digitador")]

[HttpPut("{id}")]

public IActionResult Put(int id, [FromBody] DisciplinaUpdateDTO disciplinaUpdateDTO)

{

DisciplinaDTO disciplinaDTO = new DisciplinaDTO();

try

{

disciplinaDTO = \_disciplinaUpdate.Ejecutar(id, disciplinaUpdateDTO);

\_auditoriaInsert.Ejecutar(new AuditoriaInsertDTO

{

Accion = Accion.Update,

EmailUsuario = User.FindFirst(ClaimTypes.Email)?.Value,

Entidad = "Disciplina",

EntidadId = id

});

return Ok(disciplinaDTO);

}

catch (ConflictException cex)

{

return Conflict(cex.Message);

}

catch (DisciplinaException dex)

{

return BadRequest(dex.Message);

}

catch (AuditoriaException aex)

{

Console.WriteLine(aex.Message);

return Ok(disciplinaDTO);

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

return StatusCode(500, "Algo no salió correctamente");

}

}

/// <summary>

/// Permite eliminar una disciplina

/// </summary>

/// <param name="id"></param>

/// <returns></returns>

[ProducesResponseType(StatusCodes.Status204NoContent)]

[ProducesResponseType(StatusCodes.Status400BadRequest)]

[ProducesResponseType(StatusCodes.Status409Conflict)]

[ProducesResponseType(StatusCodes.Status500InternalServerError)]

[Authorize(Roles = "Digitador")]

[HttpDelete("{id}")]

public IActionResult Delete(int id)

{

try

{

\_deleteDisciplina.Ejecutar(id);

\_auditoriaInsert.Ejecutar(new AuditoriaInsertDTO

{

Accion = Accion.Delete,

EmailUsuario = User.FindFirst(ClaimTypes.Email)?.Value,

Entidad = "Disciplina",

EntidadId = id

});

return NoContent();

}

catch (ConflictException cex)

{

return Conflict(cex.Message);

}

catch (DisciplinaException dex)

{

return BadRequest(dex.Message);

}

catch (AuditoriaException aex)

{

Console.WriteLine(aex.Message);

return NoContent();

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

return StatusCode(500, "Algo no salió correctamente");

}

}

/// <summary>

/// Permite obtener todas las disciplinas

/// </summary>

/// <returns></returns>

[ProducesResponseType(StatusCodes.Status200OK)]

[ProducesResponseType(StatusCodes.Status500InternalServerError)]

//Ver en este el tema de la autenticación, es utilizado para obtener el listado para el anterior que es sin autenticación

[HttpGet]

public IActionResult GetAll()

{

try

{

IEnumerable<DisciplinaDTO> disciplinas = \_finAllDisciplinas.Ejecutar();

return Ok(disciplinas);

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

return StatusCode(500, "Algo no salió correctamente");

}

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: EventoController.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\WebApi\Controllers\EventoController.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Eventos;

using LogicaAplicacion.ICasosDeUso.Eventos;

using LogicaNegocio.ExcepcionesEntidades;

using Microsoft.AspNetCore.Authorization;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

namespace WebApi.Controllers

{

[Route("api/[controller]")]

[ApiController]

public class EventoController : ControllerBase

{

private readonly ISelectByBusqueda \_selectByBusqueda;

public EventoController(ISelectByBusqueda selectByBusqueda)

{

\_selectByBusqueda = selectByBusqueda;

}

/// <summary>

/// Permite obtener el listado de eventos de acuerdo a los filtros cargados.

/// </summary>

/// <param name="dto"></param>

/// <returns></returns>

[ProducesResponseType(StatusCodes.Status200OK)]

[ProducesResponseType(StatusCodes.Status400BadRequest)]

[ProducesResponseType(StatusCodes.Status500InternalServerError)]

[HttpGet]

[Authorize(Roles = "Digitador")]

public IActionResult GetByBusqueda([FromQuery] EventoBusquedaDTO dto)

{

try

{

return Ok(\_selectByBusqueda.Ejecutar(dto));

}

catch (EventoException eex)

{

return BadRequest(eex.Message);

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

return StatusCode(500, "Algo no salió correctamente");

}

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: UsuarioController.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\WebApi\Controllers\UsuarioController.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Usuarios;

using LogicaAplicacion.ICasosDeUso.Usuarios;

using LogicaNegocio.ExcepcionesEntidades;

using Microsoft.AspNetCore.Authorization;

using Microsoft.AspNetCore.Mvc;

using System.Security.Claims;

using WebApi.Tokens;

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

namespace WebApi.Controllers

{

[Route("api/[controller]")]

[ApiController]

public class UsuarioController : ControllerBase

{

private readonly ILoginUsuario \_login;

private readonly IGetByIdUsuario \_getById;

private readonly IFindAllUsuarios \_getAll;

private readonly IAltaUsuario \_alta;

private readonly IUpdateUsuario \_update;

private readonly IDeleteUsuario \_delete;

private readonly IConfiguration \_config;

public UsuarioController(

ILoginUsuario login,

IAltaUsuario alta,

IUpdateUsuario update,

IGetByIdUsuario getById,

IFindAllUsuarios getAll,

IDeleteUsuario delete,

IConfiguration config)

{

\_login = login;

\_alta = alta;

\_update = update;

\_getById = getById;

\_getAll = getAll;

\_delete = delete;

\_config = config;

}

/// <summary>

/// Permite realizar el login a la página

/// </summary>

/// <param name="dto"></param>

/// <returns></returns>

[ProducesResponseType(StatusCodes.Status200OK)]

[ProducesResponseType(StatusCodes.Status400BadRequest)]

[ProducesResponseType(StatusCodes.Status500InternalServerError)]

[HttpPost("iniciar-sesion")]

public IActionResult Login([FromBody] CredencialesDTO dto)

{

Token token = new Token(\_config);

try

{

UsuarioDTO usuario = \_login.Ejecutar(dto.Email, dto.Contrasena);

UsuarioLogueadoDTO usuarioRes = new UsuarioLogueadoDTO

{

Id = usuario.Id,

Rol = usuario.RolUsuario,

Token = token.Crear(usuario)

};

return Ok(usuarioRes);

}

catch (UsuarioException uex)

{

return BadRequest(uex.Message);

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

return StatusCode(500, ex.Message);

}

}

/// <summary>

/// Permite obtener el listado de usuarios

/// </summary>

/// <returns></returns>

// GET: api/<UsuarioController>

[HttpGet]

public IActionResult Get()

{

try

{

return Ok(\_getAll.Ejecutar());

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

return StatusCode(500, "Algo no salió correctamente");

}

}

/// <summary>

/// Permite obtener un usuario por id

/// </summary>

/// <param name="id"></param>

/// <returns></returns>

[ProducesResponseType(StatusCodes.Status200OK)]

[ProducesResponseType(StatusCodes.Status400BadRequest)]

[ProducesResponseType(StatusCodes.Status500InternalServerError)]

[Authorize(Roles = "Administrador,Digitador")]

// GET api/<UsuarioController>/5

[HttpGet("{id}")]

public IActionResult Get(int id)

{

try

{

string idUsuario = User.FindFirst(ClaimTypes.NameIdentifier)?.Value;

string rol = User.FindFirst(ClaimTypes.Role)?.Value;

if (rol == "Digitador")

if (idUsuario != id.ToString()) return Unauthorized("No tiene permisos para acceder a estos recursos");

return Ok(\_getById.Ejecutar(id));

}

catch (UsuarioException uex)

{

return BadRequest(uex);

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

return StatusCode(500, "Algo no salió correctamente");

}

}

/// <summary>

/// Permite realizar el alta de un usuario

/// </summary>

/// <param name="usuarioInsert"></param>

/// <returns></returns>

[ProducesResponseType(StatusCodes.Status201Created)]

[ProducesResponseType(StatusCodes.Status400BadRequest)]

[ProducesResponseType(StatusCodes.Status409Conflict)]

[ProducesResponseType(StatusCodes.Status500InternalServerError)]

[Authorize(Roles = "Administrador")]

// POST api/<UsuarioController>

[HttpPost()]

public IActionResult Post([FromBody] UsuarioInsertDTO usuarioInsert)

{

try

{

\_alta.Ejecutar(usuarioInsert);

return Created();

}

catch (ConflictException cex)

{

return Conflict(cex.Message);

}

catch (UsuarioException uex)

{

return BadRequest(uex.Message);

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

return StatusCode(500, "Algo no salió correctamente");

}

}

/// <summary>

/// Permite modificar los datos de un usuario

/// </summary>

/// <param name="id"></param>

/// <param name="dtoUpdate"></param>

/// <returns></returns>

[ProducesResponseType(StatusCodes.Status200OK)]

[ProducesResponseType(StatusCodes.Status400BadRequest)]

[ProducesResponseType(StatusCodes.Status409Conflict)]

[ProducesResponseType(StatusCodes.Status500InternalServerError)]

[Authorize(Roles = "Administrador,Digitador")]

[HttpPut("{id}")]

public IActionResult Put(int id, [FromBody] UsuarioUpdateDTO dtoUpdate)

{

try

{

string idUsuario = User.FindFirst(ClaimTypes.NameIdentifier)?.Value;

string rol = User.FindFirst(ClaimTypes.Role)?.Value;

if (rol == "Digitador")

if (idUsuario != id.ToString()) return Unauthorized("No tiene permisos para acceder a estos recursos");

return Ok(\_update.Ejecutar(id, dtoUpdate));

}

catch (ConflictException cex)

{

return Conflict(cex.Message);

}

catch (UsuarioException uex)

{

return BadRequest(uex);

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

return StatusCode(500, "Algo no salió correctamente");

}

}

/// <summary>

/// Permite modificar la contraseña de un usuario

/// </summary>

/// <param name="id"></param>

/// <param name="dto"></param>

/// <returns></returns>

[ProducesResponseType(StatusCodes.Status200OK)]

[ProducesResponseType(StatusCodes.Status400BadRequest)]

[ProducesResponseType(StatusCodes.Status500InternalServerError)]

[Authorize(Roles = "Administrador,Digitador")]

[HttpPatch("{id}")]

public IActionResult Patch(int id, [FromBody] UpdateContrasenaDTO dto)

{

try

{

string idUsuario = User.FindFirst(ClaimTypes.NameIdentifier)?.Value;

string rol = User.FindFirst(ClaimTypes.Role)?.Value;

if (rol == "Digitador")

if (idUsuario != id.ToString()) return Unauthorized("No tiene permisos para acceder a estos recursos");

if (rol == "Digitador")

{

return Ok(\_update.Ejecutar(id, dto.ContrasenaNueva, dto.ContrasenaAnterior));

}

return Ok(\_update.Ejecutar(id, dto.ContrasenaNueva));

}

catch (UsuarioException uex)

{

return BadRequest(uex);

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

return StatusCode(500, "Algo no salió correctamente");

}

}

/// <summary>

/// Permite borrar un usuario

/// </summary>

/// <param name="id"></param>

/// <returns></returns>

[ProducesResponseType(StatusCodes.Status204NoContent)]

[ProducesResponseType(StatusCodes.Status400BadRequest)]

[ProducesResponseType(StatusCodes.Status500InternalServerError)]

[Authorize(Roles = "Administrador")]

[HttpDelete("{id}")]

public IActionResult Delete(int id)

{

try

{

\_delete.Ejecutar(id);

return NoContent();

}

catch (UsuarioException uex)

{

return BadRequest(uex);

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

return StatusCode(500, "Algo no salió correctamente");

}

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: Token.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\WebApi\Tokens\Token.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Usuarios;

using Microsoft.IdentityModel.Tokens;

using System.IdentityModel.Tokens.Jwt;

using System.Security.Claims;

using System.Text;

namespace WebApi.Tokens

{

public class Token

{

private readonly string \_secret;

public Token(IConfiguration config)

{

\_secret = config.GetValue<string>("JWT:secret");

}

public string Crear(UsuarioDTO dto)

{

byte[] clave = Encoding.ASCII.GetBytes(\_secret);

JwtSecurityTokenHandler tokenHandler = new JwtSecurityTokenHandler();

//Se incluye un claim para el email

SecurityTokenDescriptor tokenDescriptor = new SecurityTokenDescriptor

{

Subject = new ClaimsIdentity(new Claim[]

{

new Claim(ClaimTypes.Email, dto.Email),

new Claim(ClaimTypes.Role, dto.RolUsuario),

new Claim(ClaimTypes.NameIdentifier, dto.Id.ToString())

}),

Expires = DateTime.UtcNow.AddMinutes(30),

SigningCredentials = new SigningCredentials(new SymmetricSecurityKey(clave),

SecurityAlgorithms.HmacSha256)

};

var token = tokenHandler.CreateToken(tokenDescriptor);

return tokenHandler.WriteToken(token);

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: AtletaDTO.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\DTOs\Atletas\AtletaDTO.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Disciplinas;

using LogicaNegocio.Entidades;

using LogicaNegocio.Enums;

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations.Schema;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Compartido.DTOs.Atletas

{

public class AtletaDTO

{

public int Id { get; set; }

public string Nombre { get; set; }

public string Apellido { get; set; }

public string Sexo { get; set; }

public string NombrePais { get; set; }

public IEnumerable<DisciplinaDTO> DisciplinasAtleta { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: AtletaInsertDTO.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\DTOs\Atletas\AtletaInsertDTO.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Compartido.DTOs.Atletas

{

public class AtletaInsertDTO

{

public string Nombre { get; set; }

public string Apellido { get; set; }

public int Sexo { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: AtletaListaDTO.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\DTOs\Atletas\AtletaListaDTO.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Compartido.DTOs.Atletas

{

public class AtletaListaDTO

{

public int Id { get; set; }

public string Nombre { get; set; }

public string Apellido { get; set; }

public string NombreCompleto { get; set; }

public string Sexo { get; set; }

public string NombrePais { get; set; }

}

//RF2 – Listado de Atletas por Disciplina(API Web + HttpClient) – Sin autenticación

//- Crear un listado de atletas filtrado por disciplina.Este endpoint permitirá consultar todos los atletas

//que están registrados en una disciplina dado su Id.El listado se retornará ordenado alfabéticamente

//por nombre completo de atleta.Se incluirán como mínimo su Id (o número, según haya utilizado), su

//nombre completo y el nombre de su país.

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: AuditoriaInsertDTO.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\DTOs\Auditorias\AuditoriaInsertDTO.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaNegocio.Enums;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Compartido.DTOs.Auditorias

{

public class AuditoriaInsertDTO

{

public Accion Accion { get; set; }

public string Entidad { get; set; }

public int EntidadId { get; set; }

public string EmailUsuario { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: DisciplinaDTO.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\DTOs\Disciplinas\DisciplinaDTO.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaNegocio.ValueObjects.Disciplina;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Compartido.DTOs.Disciplinas

{

public class DisciplinaDTO

{

public int Id { get; set; }

public string Nombre { get; set; }

public int AnioIntegracion { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: DisciplinaInsertDTO.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\DTOs\Disciplinas\DisciplinaInsertDTO.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Compartido.DTOs.Disciplinas

{

public class DisciplinaInsertDTO

{

public string Nombre { get; set; }

public int AnioIntegracion { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: DisciplinaUpdateDTO.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\DTOs\Disciplinas\DisciplinaUpdateDTO.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Compartido.DTOs.Disciplinas

{

public class DisciplinaUpdateDTO

{

public int Id { get; set; }

public string Nombre { get; set; }

public int AnioIntegracion { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: AtletaEventoListaDTO.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\DTOs\Eventos\AtletaEventoListaDTO.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaNegocio.Entidades;

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations.Schema;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Compartido.DTOs.Eventos

{

public class AtletaEventoListaDTO

{

public int Id { get; set; }

public string NombrePrueba { get; set; }

public string NombreDisciplina { get; set; }

public DateTime FchInicio { get; set; }

public DateTime FchFin { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: EventoBusquedaDTO.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\DTOs\Eventos\EventoBusquedaDTO.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Compartido.DTOs.Eventos

{

public class EventoBusquedaDTO

{

public int? DisciplinaId { get; set; }

public DateTime? FchInicio { get; set; }

public DateTime? FchFin { get; set; }

public string? NombreEvento { get; set; }

public int? PuntajeMin { get; set; }

public int? PuntajeMax { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: EventoDTO.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\DTOs\Eventos\EventoDTO.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaNegocio.Entidades;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Compartido.DTOs.Eventos

{

public class EventoDTO

{

public int Id { get; set; }

public string NombrePrueba { get; set; }

public DateTime FchInicio { get; set; }

public DateTime FchFin { get; set; }

public IEnumerable<PuntajeEventoAtletaDTO> LiAtletas { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: EventoInsertDTO.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\DTOs\Eventos\EventoInsertDTO.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Compartido.DTOs.Eventos

{

public class EventoInsertDTO

{

public int DisciplinaId { get; set; }

public string NombrePrueba { get; set; }

public DateTime FchInicio { get; set; }

public DateTime FchFin { get; set; }

public int[]? AtletasId { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: EventoListaDTO.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\DTOs\Eventos\EventoListaDTO.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.ComponentModel;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Compartido.DTOs.Eventos

{

public class EventoListaDTO

{

public int EventoId { get; set; }

public string NombrePrueba { get; set; }

public DateTime FchInicio { get; set; }

public DateTime FchFin { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: EventoUpdatePuntajesDTO.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\DTOs\Eventos\EventoUpdatePuntajesDTO.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Compartido.DTOs.Eventos

{

public class EventoUpdatePuntajesDTO

{

public int Id { get; set; }

public IEnumerable<PEAUpdateDTO> LiAtletas { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: PEAUpdateDTO.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\DTOs\Eventos\PEAUpdateDTO.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Compartido.DTOs.Eventos

{

public class PEAUpdateDTO

{

public int AtletaId { get; set; }

public decimal Puntaje { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: PuntajeEventoAtletaDTO.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\DTOs\Eventos\PuntajeEventoAtletaDTO.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Atletas;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Compartido.DTOs.Eventos

{

public class PuntajeEventoAtletaDTO

{

public AtletaDTO Atleta { get; set; }

public decimal Puntaje { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: PaisInsertDTO.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\DTOs\Paises\PaisInsertDTO.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Compartido.DTOs.Paises

{

public class PaisInsertDTO

{

public string Nombre { get; set; }

public int Habitantes { get; set; }

public string NombreDelegado { get; set; }

public string TelDelegado { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: CredencialesDTO.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\DTOs\Usuarios\CredencialesDTO.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Compartido.DTOs.Usuarios

{

public class CredencialesDTO

{

public string Email { get; set; }

public string Contrasena { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: UpdateContrasenaDTO.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\DTOs\Usuarios\UpdateContrasenaDTO.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Compartido.DTOs.Usuarios

{

public class UpdateContrasenaDTO

{

public string ContrasenaNueva { get; set; }

public string ContrasenaAnterior { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: UsuarioDTO.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\DTOs\Usuarios\UsuarioDTO.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaNegocio.Enums;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Compartido.DTOs.Usuarios

{

public class UsuarioDTO

{

public int Id { get; set; }

public string Email { get; set; }

public string RolUsuario { get; set; }

public string? Nombre { get; set; }

public string FechaRegistro { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: UsuarioInsertDTO.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\DTOs\Usuarios\UsuarioInsertDTO.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Compartido.DTOs.Usuarios

{

public class UsuarioInsertDTO

{

public string Email { get; set; }

public string Contrasena { get; set; }

public int RolUsuario { get; set; }

public int IdAdminRegistro { get; set; }

public string? Nombre { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: UsuarioLogueadoDTO.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\DTOs\Usuarios\UsuarioLogueadoDTO.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Compartido.DTOs.Usuarios

{

public class UsuarioLogueadoDTO

{

public int Id { get; set; }

public string Rol { get; set; }

public string Token { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: UsuarioUpdateDTO.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\Compartido\DTOs\Usuarios\UsuarioUpdateDTO.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Compartido.DTOs.Usuarios

{

public class UsuarioUpdateDTO

{

public string Email { get; set; }

public string Nombre { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: AgregarDisciplina.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\CasosDeUso\Atletas\AgregarDisciplina.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaAplicacion.ICasosDeUso.Atletas;

using LogicaNegocio.Entidades;

using LogicaNegocio.ExcepcionesEntidades;

using LogicaNegocio.IRepositorios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.CasosDeUso.Atletas

{

public class AgregarDisciplina : IAgregarDisciplina

{

private readonly IRepositorioAtleta \_repositorioAtleta;

private readonly IRepositorioDisciplina \_repositorioDisciplina;

public AgregarDisciplina(IRepositorioAtleta repositorioAtleta, IRepositorioDisciplina repositorioDisciplina)

{

\_repositorioAtleta = repositorioAtleta;

\_repositorioDisciplina = repositorioDisciplina;

}

public void Ejecutar(int id, int? idDisciplina)

{

if (id == 0)

{

throw new AtletaException("Id de atleta incorrecto");

}

if (idDisciplina == null || idDisciplina == 0)

{

throw new AtletaException("Id de disciplina incorrecto");

}

Atleta atleta = \_repositorioAtleta.GetById(id);

Disciplina tieneDisciplina = atleta.LiDisciplinas.FirstOrDefault(d => d.Id == idDisciplina);

if (tieneDisciplina != null)

{

throw new AtletaException("El atleta ya está registrado en esta disciplina");

};

Disciplina disciplinaAgregar = \_repositorioDisciplina.GetById(idDisciplina.Value);

atleta.LiDisciplinas.Add(disciplinaAgregar);

\_repositorioAtleta.GuardarCambios();

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: AltaAtleta.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\CasosDeUso\Atletas\AltaAtleta.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Atletas;

using LogicaAccesoDatos;

using LogicaAplicacion.ICasosDeUso.Atletas;

using LogicaNegocio.IRepositorios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.CasosDeUso.Atletas

{

public class AltaAtleta : IAltaAtleta

{

private readonly IRepositorioAtleta \_repositorioAtleta;

public AltaAtleta(IRepositorioAtleta repositorioAtleta)

{

\_repositorioAtleta = repositorioAtleta;

}

public void Ejecutar(AtletaInsertDTO atletaInsertDTO)

{

throw new NotImplementedException();

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: EventosAtleta.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\CasosDeUso\Atletas\EventosAtleta.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Eventos;

using Compartido.Mappers;

using LogicaAplicacion.ICasosDeUso.Atletas;

using LogicaNegocio.IRepositorios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.CasosDeUso.Atletas

{

public class EventosAtleta : IEventosAtleta

{

private readonly IRepositorioEvento \_repositorioEvento;

public EventosAtleta(IRepositorioEvento repositorioEvento)

{

\_repositorioEvento = repositorioEvento;

}

public IEnumerable<AtletaEventoListaDTO> Ejecutar(int atletaId)

=> EventoMapper.EventoToListaAtletaEventoDTO(\_repositorioEvento.GetEventosAtleta(atletaId));

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: FindAllAtletas.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\CasosDeUso\Atletas\FindAllAtletas.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Atletas;

using Compartido.Mappers;

using LogicaAplicacion.ICasosDeUso.Atletas;

using LogicaNegocio.IRepositorios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.CasosDeUso.Atletas

{

public class FindAllAtletas : IFindAllAtletas

{

private readonly IRepositorioAtleta \_repositorioAtleta;

public FindAllAtletas(IRepositorioAtleta repositorioAtleta)

{

\_repositorioAtleta = repositorioAtleta;

}

public IEnumerable<AtletaListaDTO> Ejectuar()

{

return AtletaMapper.AtletasToListaDTO(\_repositorioAtleta.GetAll());

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: GetByIdAtleta.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\CasosDeUso\Atletas\GetByIdAtleta.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Atletas;

using Compartido.Mappers;

using LogicaAplicacion.ICasosDeUso.Atletas;

using LogicaNegocio.Entidades;

using LogicaNegocio.IRepositorios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.CasosDeUso.Atletas

{

public class GetByIdAtleta : IGetByIdAtleta

{

private readonly IRepositorioAtleta \_repositorioAtleta;

public GetByIdAtleta(IRepositorioAtleta repositorioAtleta)

{

\_repositorioAtleta = repositorioAtleta;

}

public AtletaDTO Ejecutar(int id) => AtletaMapper.AtletaToDTO(\_repositorioAtleta.GetById(id));

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: SelectByDisciplinaId.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\CasosDeUso\Atletas\SelectByDisciplinaId.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Atletas;

using Compartido.Mappers;

using LogicaAplicacion.ICasosDeUso.Atletas;

using LogicaNegocio.Entidades;

using LogicaNegocio.ExcepcionesEntidades;

using LogicaNegocio.IRepositorios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.CasosDeUso.Atletas

{

public class SelectByDisciplinaId : ISelectByDisciplinaId

{

private readonly IRepositorioAtleta \_repositorioAtleta;

public SelectByDisciplinaId(IRepositorioAtleta repositorioAtleta)

{

\_repositorioAtleta = repositorioAtleta;

}

public IEnumerable<AtletaListaDTO> Ejecutar(int disciplinaId)

{

if (disciplinaId <= 0) throw new AtletaException("Id incorrecto");

List<Atleta>? resdb = \_repositorioAtleta.SelectByDisciplinaId(disciplinaId);

return resdb == null

? null

: AtletaMapper.AtletasToListaDTO(resdb).OrderBy(a => a.NombreCompleto);

}

//RF2 – Listado de Atletas por Disciplina(API Web + HttpClient) – Sin autenticación

//- Crear un listado de atletas filtrado por disciplina.Este endpoint permitirá consultar todos los atletas

//que están registrados en una disciplina dado su Id.El listado se retornará ordenado alfabéticamente

//por nombre completo de atleta.Se incluirán como mínimo su Id (o número, según haya utilizado), su

//nombre completo y el nombre de su país.

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: AuditoriaInsert.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\CasosDeUso\Auditorias\AuditoriaInsert.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Auditorias;

using Compartido.Mappers;

using LogicaAplicacion.ICasosDeUso.Auditorias;

using LogicaNegocio.ExcepcionesEntidades;

using LogicaNegocio.IRepositorios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.CasosDeUso.Auditorias

{

public class AuditoriaInsert:IAuditoriaInsert

{

private readonly IRepositorioAuditoria \_repositorioAuditoria;

public AuditoriaInsert(IRepositorioAuditoria repositorioAuditoria)

{

\_repositorioAuditoria = repositorioAuditoria;

}

public void Ejecutar(AuditoriaInsertDTO dto)

{

if (dto == null) throw new AuditoriaException("Error al registrar la auditoría");

\_repositorioAuditoria.Add(AuditoriaMapper.DtoInsertToAuditoria(dto));

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: DeleteDisciplina.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\CasosDeUso\Disciplinas\DeleteDisciplina.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaAplicacion.ICasosDeUso.Disciplinas;

using LogicaNegocio.Entidades;

using LogicaNegocio.IRepositorios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using LogicaNegocio.ExcepcionesEntidades;

namespace LogicaAplicacion.CasosDeUso.Disciplinas

{

public class DeleteDisciplina : IDeleteDisciplina

{

private readonly IRepositorioDisciplina \_repositorioDisciplina;

public DeleteDisciplina(IRepositorioDisciplina repositorioDisciplina)

{

\_repositorioDisciplina = repositorioDisciplina;

}

public void Ejecutar(int id)

{

if (id <= 0) throw new DisciplinaException("Id incorrecto");

Disciplina buscarDisciplina = \_repositorioDisciplina.GetById(id)

?? throw new DisciplinaException($"No se encontró una Disciplina con id: {id}");

if (buscarDisciplina.LiAtletas.Count > 0)

throw new ConflictException("No se puede eliminar una disciplina con atletlas registrados");

\_repositorioDisciplina.Delete(buscarDisciplina);

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: DisciplinaSelectById.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\CasosDeUso\Disciplinas\DisciplinaSelectById.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Disciplinas;

using LogicaAplicacion.ICasosDeUso.Disciplinas;

using LogicaNegocio.Entidades;

using LogicaNegocio.ExcepcionesEntidades;

using LogicaNegocio.IRepositorios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.CasosDeUso.Disciplinas

{

public class DisciplinaSelectById: IDisciplinaSelectById

{

private readonly IRepositorioDisciplina \_repositorioDisciplina;

public DisciplinaSelectById(IRepositorioDisciplina repositorioDisciplina)

{

\_repositorioDisciplina = repositorioDisciplina;

}

public DisciplinaDTO? Ejecutar(int id)

{

if (id <= 0) throw new DisciplinaException("Id Incorrecto");

Disciplina? res = \_repositorioDisciplina.GetById(id);

return res == null ? null : new DisciplinaDTO

{

Id = id,

AnioIntegracion = res.AnioIntegracion,

Nombre = res.Nombre.Valor

};

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: DisciplinaSelectByNombre.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\CasosDeUso\Disciplinas\DisciplinaSelectByNombre.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Disciplinas;

using LogicaAplicacion.ICasosDeUso.Disciplinas;

using LogicaNegocio.Entidades;

using LogicaNegocio.ExcepcionesEntidades;

using LogicaNegocio.IRepositorios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.CasosDeUso.Disciplinas

{

public class DisciplinaSelectByNombre : IDisciplinaSelectByNombre

{

private readonly IRepositorioDisciplina \_repositorioDisciplina;

public DisciplinaSelectByNombre(IRepositorioDisciplina repositorioDisciplina)

{

\_repositorioDisciplina = repositorioDisciplina;

}

public DisciplinaDTO? Ejecutar(string nombre)

{

if (string.IsNullOrEmpty(nombre)) throw new DisciplinaException("Se requieren un Nombre");

Disciplina? res = \_repositorioDisciplina.GetByNombre(nombre);

return res == null ? null : new DisciplinaDTO

{

Id = res.Id,

AnioIntegracion = res.AnioIntegracion,

Nombre = nombre

};

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: DisciplinaUpdate.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\CasosDeUso\Disciplinas\DisciplinaUpdate.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Disciplinas;

using Compartido.Mappers;

using LogicaAplicacion.ICasosDeUso.Disciplinas;

using LogicaAplicacion.Validadores;

using LogicaNegocio.Entidades;

using LogicaNegocio.ExcepcionesEntidades;

using LogicaNegocio.IRepositorios;

using LogicaNegocio.ValueObjects.Disciplina;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.CasosDeUso.Disciplinas

{

public class DisciplinaUpdate : IDisciplinaUpdate

{

private readonly IRepositorioDisciplina \_repositorioDisciplina;

public DisciplinaUpdate(IRepositorioDisciplina repositorioDisciplina)

{

\_repositorioDisciplina = repositorioDisciplina;

}

public DisciplinaDTO Ejecutar(int id, DisciplinaUpdateDTO dto)

{

ValidarDisciplina.Nombre(dto.Nombre);

ValidarDisciplina.Anio(dto.AnioIntegracion);

Disciplina Existe = \_repositorioDisciplina.GetById(id)

?? throw new DisciplinaException("Disciplina no encontrada");

Disciplina? ExisteNombre = \_repositorioDisciplina.GetByNombre(dto.Nombre);

if(ExisteNombre != null)

{

if (dto.Nombre == ExisteNombre.Nombre.Valor && id != ExisteNombre.Id)

throw new ConflictException("Ya existe una disciplina con ese nombre");

}

Existe.Nombre = new RDisciplinaNombre( dto.Nombre);

Existe.AnioIntegracion = dto.AnioIntegracion;

\_repositorioDisciplina.Update(Existe);

return DisciplinaMapper.DisciplinaToDTO(Existe);

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: FindAllDisciplinas.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\CasosDeUso\Disciplinas\FindAllDisciplinas.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Disciplinas;

using Compartido.Mappers;

using LogicaAccesoDatos;

using LogicaAplicacion.ICasosDeUso.Disciplinas;

using LogicaNegocio.IRepositorios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.CasosDeUso.Disciplinas

{

public class FindAllDisciplinas : IFindAllDisciplinas

{

private readonly IRepositorioDisciplina \_repositorioDisciplina;

public FindAllDisciplinas(IRepositorioDisciplina repositorioDisciplina)

{

\_repositorioDisciplina = repositorioDisciplina;

}

public IEnumerable<DisciplinaDTO> Ejecutar() => DisciplinaMapper.DisciplinasToListaDTO(\_repositorioDisciplina.GetAll());

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: FindAtletasDisciplina.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\CasosDeUso\Disciplinas\FindAtletasDisciplina.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Atletas;

using Compartido.Mappers;

using LogicaAplicacion.ICasosDeUso.Disciplinas;

using LogicaNegocio.ExcepcionesEntidades;

using LogicaNegocio.IRepositorios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.CasosDeUso.Disciplinas

{

public class FindAtletasDisciplina : IFindAtletasDisciplina

{

private IRepositorioDisciplina \_repositorioDisciplina;

public FindAtletasDisciplina(IRepositorioDisciplina repositorioDisciplina)

{

\_repositorioDisciplina = repositorioDisciplina;

}

public IEnumerable<AtletaListaDTO> Ejecutar(int idDisciplina)

{

if (idDisciplina == 0)

{

throw new DisciplinaException("Id Incorrecto");

}

return AtletaMapper.AtletasToListaDTO(\_repositorioDisciplina.GetAtletasDisciplina(idDisciplina));

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: InsertDisciplina.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\CasosDeUso\Disciplinas\InsertDisciplina.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Disciplinas;

using Compartido.Mappers;

using LogicaAccesoDatos;

using LogicaAplicacion.ICasosDeUso.Disciplinas;

using LogicaAplicacion.Validadores;

using LogicaNegocio.Entidades;

using LogicaNegocio.ExcepcionesEntidades;

using LogicaNegocio.IRepositorios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.CasosDeUso.Disciplinas

{

public class InsertDisciplina : IInsertDisciplina

{

private IRepositorioDisciplina \_repositorioDisciplina;

public InsertDisciplina(IRepositorioDisciplina repositorioDisciplina)

{

\_repositorioDisciplina = repositorioDisciplina;

}

public int Ejecutar(DisciplinaInsertDTO disciplinaInsertDTO)

{

ValidarDisciplina.Nombre(disciplinaInsertDTO.Nombre);

ValidarDisciplina.Anio(disciplinaInsertDTO.AnioIntegracion);

Disciplina? BuscarSiExiste = \_repositorioDisciplina.GetByNombre(disciplinaInsertDTO.Nombre);

if (BuscarSiExiste != null) throw new ConflictException("Ya existe una disciplina con ese nombre");

Disciplina disciplina = \_repositorioDisciplina.Insert(DisciplinaMapper.InsertDTOToDisciplina(disciplinaInsertDTO));

return disciplina.Id;

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: AltaEvento.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\CasosDeUso\Eventos\AltaEvento.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Eventos;

using LogicaAccesoDatos;

using LogicaAccesoDatos.Repositorios;

using LogicaAplicacion.ICasosDeUso.Eventos;

using LogicaNegocio.IRepositorios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using LogicaNegocio.ExcepcionesEntidades;

using LogicaAplicacion.Validadores;

using Compartido.Mappers;

namespace LogicaAplicacion.CasosDeUso.Eventos

{

public class AltaEvento : IAltaEvento

{

private readonly IRepositorioEvento \_repositorioEvento;

private readonly IRepositorioDisciplina \_repositorioDisciplina;

public AltaEvento(IRepositorioEvento repositorioEvento, IRepositorioDisciplina repositorioDisciplina)

{

\_repositorioEvento = repositorioEvento;

\_repositorioDisciplina = repositorioDisciplina;

}

public void Ejecutar(EventoInsertDTO eventoInsertDTO)

{

if (eventoInsertDTO == null)

throw new EventoException("El Evento DTO se encuentra vacío");

if (\_repositorioEvento.GetByNombre(eventoInsertDTO.NombrePrueba) != null)

throw new EventoException("Ya existe un evento con ese nombre");

ValidarEvento.CantidadAtletas(eventoInsertDTO);

ValidarEvento.AtletasRegistradosEnDisciplina

(\_repositorioDisciplina.GetAtletasDisciplina(eventoInsertDTO.DisciplinaId), eventoInsertDTO.AtletasId);

ValidarEvento.Fechas(eventoInsertDTO);

\_repositorioEvento.Add(EventoMapper.DtoToEvento(eventoInsertDTO));

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: CargarPuntajes.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\CasosDeUso\Eventos\CargarPuntajes.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Eventos;

using Compartido.Mappers;

using LogicaAplicacion.ICasosDeUso.Eventos;

using LogicaNegocio.Entidades;

using LogicaNegocio.IRepositorios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using LogicaNegocio.ExcepcionesEntidades;

using LogicaAplicacion.Validadores;

namespace LogicaAplicacion.CasosDeUso.Eventos

{

public class CargarPuntajes : ICargarPuntajes

{

private readonly IRepositorioEvento \_repositorioEvento;

public CargarPuntajes(IRepositorioEvento repositorioEvento)

{

\_repositorioEvento = repositorioEvento;

}

public EventoDTO Ejecutar(EventoUpdatePuntajesDTO eventoUpdatePuntajesDTO)

{

// Validar puntajes >= 0

ValidarEvento.Puntajes(eventoUpdatePuntajesDTO);

// Validar que existe el Evento

Evento eventoAModificar = \_repositorioEvento.GetById(eventoUpdatePuntajesDTO.Id);

if (eventoAModificar == null) throw new EventoException("No se encontró un evento con ese Id");

// Update

eventoAModificar.LiPuntajes = EventoMapper.DtoListaModificadaToListaPuntaje(eventoUpdatePuntajesDTO.LiAtletas);

\_repositorioEvento.Update(eventoAModificar);

return EventoMapper.EventoToDTO(eventoAModificar);

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: FindById.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\CasosDeUso\Eventos\FindById.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Eventos;

using Compartido.Mappers;

using LogicaAplicacion.ICasosDeUso.Eventos;

using LogicaNegocio.ExcepcionesEntidades;

using LogicaNegocio.IRepositorios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.CasosDeUso.Eventos

{

public class FindById : IFindById

{

private readonly IRepositorioEvento \_repositorioEvento;

public FindById(IRepositorioEvento repositorioEvento)

{

\_repositorioEvento = repositorioEvento;

}

public EventoDTO Ejecutar(int id)

{

if (id < 1) throw new EventoException("No existe atleta con ese id");

return EventoMapper.EventoToDTO(\_repositorioEvento.GetById(id));

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: FindEventosFecha.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\CasosDeUso\Eventos\FindEventosFecha.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Eventos;

using Compartido.Mappers;

using LogicaAplicacion.ICasosDeUso.Eventos;

using LogicaNegocio.IRepositorios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.CasosDeUso.Eventos

{

public class FindEventosFecha : IFindEventosFecha

{

private readonly IRepositorioEvento \_repositorioEvento;

public FindEventosFecha(IRepositorioEvento repositorioEvento)

{

\_repositorioEvento = repositorioEvento;

}

public IEnumerable<EventoListaDTO> Ejecutar(DateTime fecha) => EventoMapper.EventosToListaDTO(\_repositorioEvento.GetByFecha(fecha));

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: SelectByBusqueda.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\CasosDeUso\Eventos\SelectByBusqueda.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Eventos;

using Compartido.Mappers;

using LogicaAplicacion.ICasosDeUso.Eventos;

using LogicaNegocio.ExcepcionesEntidades;

using LogicaNegocio.IRepositorios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.CasosDeUso.Eventos

{

public class SelectByBusqueda : ISelectByBusqueda

{

private readonly IRepositorioEvento \_repositorioEvento;

public SelectByBusqueda(IRepositorioEvento repositorioEvento)

{

\_repositorioEvento = repositorioEvento;

}

public IEnumerable<EventoListaDTO> Ejecutar(EventoBusquedaDTO busqueda)

{

// Validaciones de la búsqueda

if(busqueda.DisciplinaId <= 0)

throw new EventoException("Id incorrecto");

if (

(busqueda.FchInicio != null && busqueda.FchFin == null) ||

(busqueda.FchFin != null && busqueda.FchInicio == null) ||

(busqueda.PuntajeMin != null && busqueda.PuntajeMax == null) ||

(busqueda.PuntajeMax != null && busqueda.PuntajeMin == null)

) throw new EventoException("Si ingresa una fecha debe ingresar ambas, lo mismo para los puntajes");

if (busqueda.FchInicio > busqueda.FchFin) throw new EventoException("Rango de fechas incorrecto");

if (busqueda.PuntajeMax < busqueda.PuntajeMin) throw new EventoException("Rango de puntajes incorrecto");

return EventoMapper.EventosToListaDTO(\_repositorioEvento.SelectByBusqueda(

busqueda.DisciplinaId,

busqueda.FchInicio,

busqueda.FchFin,

busqueda.NombreEvento,

busqueda.PuntajeMin,

busqueda.PuntajeMax

));

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: AltaPais.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\CasosDeUso\Paises\AltaPais.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Paises;

using LogicaAccesoDatos;

using LogicaAplicacion.ICasosDeUso.Paises;

using LogicaNegocio.IRepositorios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.CasosDeUso.Paises

{

public class AltaPais : IAltaPais

{

private readonly IRepositorioPais \_repositorioPais;

public AltaPais(IRepositorioPais repositorioPais)

{

\_repositorioPais = repositorioPais;

}

public void Ejecutar(PaisInsertDTO paisInsertDTO)

{

throw new NotImplementedException();

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: AltaUsuario.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\CasosDeUso\Usuarios\AltaUsuario.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Usuarios;

using Compartido.Mappers;

using LogicaAccesoDatos;

using LogicaAplicacion.ICasosDeUso.Usuarios;

using LogicaAplicacion.Validadores;

using LogicaNegocio.Entidades;

using LogicaNegocio.ExcepcionesEntidades;

using LogicaNegocio.IRepositorios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

namespace LogicaAplicacion.CasosDeUso.Usuarios

{

public partial class AltaUsuario : IAltaUsuario

{

private readonly IRepositorioUsuario \_repositorioUsuario;

public AltaUsuario(IRepositorioUsuario repoUsuario)

{

\_repositorioUsuario = repoUsuario;

}

public void Ejecutar(UsuarioInsertDTO usuarioInsertDTO)

{

if (usuarioInsertDTO == null)

{

throw new UsuarioException("El usuario no puede ser vacío");

}

Usuario? buscarSiExiste = \_repositorioUsuario.GetByEmail(usuarioInsertDTO.Email);

if (buscarSiExiste != null)

{

throw new ConflictException("Ya se registró un usuario con ese email");

}

// Validaciones

var (email, contrasena, rol, nombre) =

(usuarioInsertDTO.Email, usuarioInsertDTO.Contrasena, usuarioInsertDTO.RolUsuario, usuarioInsertDTO.Nombre);

ValidarUsuario.Email(email);

ValidarUsuario.Contrasena(contrasena);

ValidarUsuario.Rol(rol);

ValidarUsuario.Nombre(nombre);

\_repositorioUsuario.Add(UsuarioMapper.InsertDTOToUsuario(usuarioInsertDTO));

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: DeleteUsuario.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\CasosDeUso\Usuarios\DeleteUsuario.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaAplicacion.ICasosDeUso.Usuarios;

using LogicaNegocio.Entidades;

using LogicaNegocio.IRepositorios;

using LogicaNegocio.ExcepcionesEntidades;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.CasosDeUso.Usuarios

{

public class DeleteUsuario : IDeleteUsuario

{

private readonly IRepositorioUsuario \_repositorioUsuario;

public DeleteUsuario(IRepositorioUsuario repositorioUsuario)

{

\_repositorioUsuario = repositorioUsuario;

}

public void Ejecutar(int id)

{

Usuario resDB = \_repositorioUsuario.GetById(id);

if (resDB == null)

{

throw new UsuarioException("Usuario no encontrado por ese ID");

}

\_repositorioUsuario.Delete(resDB);

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: FindAllUsuarios.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\CasosDeUso\Usuarios\FindAllUsuarios.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Usuarios;

using Compartido.Mappers;

using LogicaAplicacion.ICasosDeUso.Usuarios;

using LogicaNegocio.Entidades;

using LogicaNegocio.IRepositorios;

namespace LogicaAplicacion.CasosDeUso.Usuarios

{

public class FindAllUsuarios : IFindAllUsuarios

{

private readonly IRepositorioUsuario \_repositorioUsuario;

public FindAllUsuarios(IRepositorioUsuario repoUsuario)

{

\_repositorioUsuario = repoUsuario;

}

public IEnumerable<UsuarioDTO> Ejecutar()

{

List<Usuario> usuarios = \_repositorioUsuario.GetAll();

return UsuarioMapper.ListaUsuariosToDTOListaUsuarios(usuarios);

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: GetByIdUsuario.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\CasosDeUso\Usuarios\GetByIdUsuario.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Usuarios;

using LogicaAplicacion.ICasosDeUso.Usuarios;

using LogicaNegocio.IRepositorios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using LogicaNegocio.ExcepcionesEntidades;

using Compartido.Mappers;

namespace LogicaAplicacion.CasosDeUso.Usuarios

{

public class GetByIdUsuario : IGetByIdUsuario

{

private readonly IRepositorioUsuario \_repositorioUsuario;

public GetByIdUsuario(IRepositorioUsuario repositorioUsuario)

{

\_repositorioUsuario = repositorioUsuario;

}

public UsuarioDTO Ejecutar(int id)

{

LogicaNegocio.Entidades.Usuario? res = \_repositorioUsuario.GetById(id);

if (res == null)

{

throw new UsuarioException("Usuario no encontrado por ID");

}

return UsuarioMapper.UsuarioToDTO(res);

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: LoginUsuario.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\CasosDeUso\Usuarios\LoginUsuario.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaAccesoDatos;

using LogicaNegocio.Entidades;

using LogicaNegocio.IRepositorios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using LogicaNegocio.Entidades;

using LogicaAplicacion.ICasosDeUso.Usuarios;

using Compartido.DTOs.Usuarios;

using Compartido.Mappers;

using LogicaNegocio.ExcepcionesEntidades;

namespace LogicaAplicacion.CasosDeUso.Usuarios

{

public class LoginUsuario : ILoginUsuario

{

private readonly IRepositorioUsuario \_repositorioUsuario;

public LoginUsuario(IRepositorioUsuario repositorioUsuario)

{

\_repositorioUsuario = repositorioUsuario;

}

public UsuarioDTO Ejecutar(string email, string password)

{

Usuario resBD = \_repositorioUsuario.LoginUsuario(email, password)

?? throw new UsuarioException("Usuario o contraseña incorrectos");

return UsuarioMapper.UsuarioToDTO(resBD);

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: UpdateUsuario.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\CasosDeUso\Usuarios\UpdateUsuario.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Usuarios;

using LogicaAplicacion.ICasosDeUso.Usuarios;

using LogicaAplicacion.Validadores;

using LogicaNegocio.Entidades;

using LogicaNegocio.IRepositorios;

using LogicaNegocio.ExcepcionesEntidades;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Compartido.Mappers;

using LogicaNegocio.ValueObjects.Usuario;

using LogicaNegocio.Enums;

namespace LogicaAplicacion.CasosDeUso.Usuarios

{

public class UpdateUsuario : IUpdateUsuario

{

private readonly IRepositorioUsuario \_repositorioUsuario;

public UpdateUsuario(IRepositorioUsuario repositorioUsuario)

{

\_repositorioUsuario = repositorioUsuario;

}

public UsuarioUpdateDTO Ejecutar(int id, UsuarioUpdateDTO usuarioUpdateDTO)

{

var (email, nombre) = (usuarioUpdateDTO.Email, usuarioUpdateDTO.Nombre);

ValidarUsuario.Email(email);

ValidarUsuario.Nombre(nombre);

Usuario actualizarUsuario = \_repositorioUsuario.GetById(id)

?? throw new UsuarioException("No se encontró el usuario que intenta actualizar");

if (\_repositorioUsuario.GetByEmail(email) != null)

throw new ConflictException("Este email ya pertenece a un usuario");

actualizarUsuario.Email = new RUsuarioEmail(email);

actualizarUsuario.Nombre = nombre;

\_repositorioUsuario.Update(actualizarUsuario);

return UsuarioMapper.UsuarioToUpdateDTO(actualizarUsuario);

}

public UsuarioUpdateDTO Ejecutar(int id, string contrasena)

{

ValidarUsuario.Contrasena(contrasena);

Usuario actualizarUsuario = \_repositorioUsuario.GetById(id)

?? throw new UsuarioException("No se encontró el usuario que intenta actualizar");

actualizarUsuario.Contrasena = new RUsuarioContrasena(contrasena);

\_repositorioUsuario.Update(actualizarUsuario);

return UsuarioMapper.UsuarioToUpdateDTO(actualizarUsuario);

}

public UsuarioUpdateDTO Ejecutar(int id, string contrasena, string contrasenaAnterior)

{

ValidarUsuario.Contrasena(contrasena);

Usuario actualizarUsuario = \_repositorioUsuario.GetById(id)

?? throw new UsuarioException("No se encontró el usuario que intenta actualizar");

if (contrasenaAnterior != actualizarUsuario.Contrasena.Valor)

throw new UsuarioException("La contraseña actual que ingresó es incorrecta");

if (contrasenaAnterior == contrasena)

throw new UsuarioException("La nueva contraseña que intenta ingresar es igual a la actual");

actualizarUsuario.Contrasena = new RUsuarioContrasena(contrasena);

\_repositorioUsuario.Update(actualizarUsuario);

return UsuarioMapper.UsuarioToUpdateDTO(actualizarUsuario);

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IAgregarDisciplina.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\ICasosDeUso\Atletas\IAgregarDisciplina.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.ICasosDeUso.Atletas

{

public interface IAgregarDisciplina

{

void Ejecutar(int id, int? idDisciplina);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IAltaAtleta.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\ICasosDeUso\Atletas\IAltaAtleta.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Atletas;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.ICasosDeUso.Atletas

{

public interface IAltaAtleta

{

void Ejecutar(AtletaInsertDTO atletaInsertDTO);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IEventosAtleta.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\ICasosDeUso\Atletas\IEventosAtleta.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Eventos;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.ICasosDeUso.Atletas

{

public interface IEventosAtleta

{

IEnumerable<AtletaEventoListaDTO> Ejecutar(int atletaId);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IFindAllAtletas.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\ICasosDeUso\Atletas\IFindAllAtletas.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using LogicaNegocio.Entidades;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Compartido.DTOs.Atletas;

namespace LogicaAplicacion.ICasosDeUso.Atletas

{

public interface IFindAllAtletas

{

IEnumerable<AtletaListaDTO> Ejectuar();

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IGetByIdAtleta.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\ICasosDeUso\Atletas\IGetByIdAtleta.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Atletas;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.ICasosDeUso.Atletas

{

public interface IGetByIdAtleta

{

AtletaDTO Ejecutar(int id);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: ISelectByDisciplinaId.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\ICasosDeUso\Atletas\ISelectByDisciplinaId.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Atletas;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.ICasosDeUso.Atletas

{

public interface ISelectByDisciplinaId

{

IEnumerable<AtletaListaDTO> Ejecutar(int disciplinaId);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IAuditoriaInsert.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\ICasosDeUso\Auditorias\IAuditoriaInsert.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Auditorias;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.ICasosDeUso.Auditorias

{

public interface IAuditoriaInsert

{

void Ejecutar(AuditoriaInsertDTO dto);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IDeleteDisciplina.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\ICasosDeUso\Disciplinas\IDeleteDisciplina.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.ICasosDeUso.Disciplinas

{

public interface IDeleteDisciplina

{

void Ejecutar(int id);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IDisciplinaSelectById.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\ICasosDeUso\Disciplinas\IDisciplinaSelectById.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Disciplinas;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.ICasosDeUso.Disciplinas

{

public interface IDisciplinaSelectById

{

DisciplinaDTO? Ejecutar(int id);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IDisciplinaSelectByNombre.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\ICasosDeUso\Disciplinas\IDisciplinaSelectByNombre.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Disciplinas;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.ICasosDeUso.Disciplinas

{

public interface IDisciplinaSelectByNombre

{

DisciplinaDTO? Ejecutar(string nombre);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IDisciplinaUpdate.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\ICasosDeUso\Disciplinas\IDisciplinaUpdate.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Disciplinas;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.ICasosDeUso.Disciplinas

{

public interface IDisciplinaUpdate

{

DisciplinaDTO Ejecutar(int id,DisciplinaUpdateDTO dto);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IFindAllDisciplinas.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\ICasosDeUso\Disciplinas\IFindAllDisciplinas.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Disciplinas;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.ICasosDeUso.Disciplinas

{

public interface IFindAllDisciplinas

{

IEnumerable<DisciplinaDTO> Ejecutar();

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IFindAtletasDisciplina.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\ICasosDeUso\Disciplinas\IFindAtletasDisciplina.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Atletas;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.ICasosDeUso.Disciplinas

{

public interface IFindAtletasDisciplina

{

IEnumerable<AtletaListaDTO> Ejecutar(int idDisciplina);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IInsertDisciplina.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\ICasosDeUso\Disciplinas\IInsertDisciplina.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Disciplinas;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.ICasosDeUso.Disciplinas

{

public interface IInsertDisciplina

{

int Ejecutar(DisciplinaInsertDTO disciplinaInsertDTO);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IAltaEvento.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\ICasosDeUso\Eventos\IAltaEvento.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Eventos;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.ICasosDeUso.Eventos

{

public interface IAltaEvento

{

void Ejecutar(EventoInsertDTO eventoInsertDTO);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: ICargarPuntajes.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\ICasosDeUso\Eventos\ICargarPuntajes.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Eventos;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.ICasosDeUso.Eventos

{

public interface ICargarPuntajes

{

EventoDTO Ejecutar(EventoUpdatePuntajesDTO eventoUpdatePuntajesDTO);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IFindById.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\ICasosDeUso\Eventos\IFindById.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Eventos;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.ICasosDeUso.Eventos

{

public interface IFindById

{

EventoDTO Ejecutar(int id);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IFindEventosFecha.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\ICasosDeUso\Eventos\IFindEventosFecha.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Compartido.DTOs.Eventos;

namespace LogicaAplicacion.ICasosDeUso.Eventos

{

public interface IFindEventosFecha

{

IEnumerable<EventoListaDTO> Ejecutar(DateTime fecha);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: ISelectByBusqueda.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\ICasosDeUso\Eventos\ISelectByBusqueda.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Eventos;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.ICasosDeUso.Eventos

{

public interface ISelectByBusqueda

{

IEnumerable<EventoListaDTO> Ejecutar(EventoBusquedaDTO busqueda);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IAltaPais.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\ICasosDeUso\Paises\IAltaPais.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Paises;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.ICasosDeUso.Paises

{

public interface IAltaPais

{

void Ejecutar(PaisInsertDTO paisInsertDTO);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IAltaUsuario.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\ICasosDeUso\Usuarios\IAltaUsuario.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Usuarios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.ICasosDeUso.Usuarios

{

public interface IAltaUsuario

{

void Ejecutar(UsuarioInsertDTO usuarioInsertDTO);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IDeleteUsuario.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\ICasosDeUso\Usuarios\IDeleteUsuario.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.ICasosDeUso.Usuarios

{

public interface IDeleteUsuario

{

void Ejecutar(int id);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IFindAllUsuarios.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\ICasosDeUso\Usuarios\IFindAllUsuarios.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Usuarios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.ICasosDeUso.Usuarios

{

public interface IFindAllUsuarios

{

IEnumerable<UsuarioDTO> Ejecutar();

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IGetByIdUsuario.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\ICasosDeUso\Usuarios\IGetByIdUsuario.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Usuarios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.ICasosDeUso.Usuarios

{

public interface IGetByIdUsuario

{

UsuarioDTO Ejecutar(int id);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: ILoginUsuario.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\ICasosDeUso\Usuarios\ILoginUsuario.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Usuarios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.ICasosDeUso.Usuarios

{

public interface ILoginUsuario

{

UsuarioDTO Ejecutar(string email, string password);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: IUpdateUsuario.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaAplicacion\ICasosDeUso\Usuarios\IUpdateUsuario.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Compartido.DTOs.Usuarios;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaAplicacion.ICasosDeUso.Usuarios

{

public interface IUpdateUsuario

{

UsuarioUpdateDTO Ejecutar(int id, UsuarioUpdateDTO usuarioUpdateDTO);

UsuarioUpdateDTO Ejecutar(int id, string contrasena, string contrasenaAnterior);

UsuarioUpdateDTO Ejecutar(int id, string contrasena);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: RDisciplinaNombre.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\ValueObjects\Disciplina\RDisciplinaNombre.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaNegocio.ValueObjects.Disciplina

{

[ComplexType]

public record RDisciplinaNombre

{

[Required]

[StringLength(50, MinimumLength = 10)]

public string Valor { get; init; }

public RDisciplinaNombre(string valor)

{

Valor = valor;

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: RUsuarioContrasena.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\ValueObjects\Usuario\RUsuarioContrasena.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaNegocio.ValueObjects.Usuario

{

[ComplexType]

public record RUsuarioContrasena

{

[Required]

public string Valor { get; init; }

public RUsuarioContrasena(string valor)

{

Valor = valor;

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: RUsuarioEmail.cs

Carpeta: C:\Users\cri\_1\OneDrive\Escritorio\Obligatorio1\_Prog3\ObligatorioP3v2\Entrega Obligatorio 2\SolObligatorioP3\LogicaNegocio\ValueObjects\Usuario\RUsuarioEmail.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LogicaNegocio.ValueObjects.Usuario

{

[ComplexType]

public record RUsuarioEmail

{

[Required]

[MaxLength(255)]

public string Valor { get; init; }

public RUsuarioEmail(string valor)

{

Valor = valor;

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: AtletaController.cs

Carpeta: C:\Users\cri\_1\Downloads\COPIAMVC\MVC\Controllers\AtletaController.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using MVC.Models.Atleta;

using MVC.Models.Disciplina;

using MVC.Utils;

using Newtonsoft.Json;

using System.Net.Http.Headers;

namespace MVC.Controllers

{

public class AtletaController : Controller

{

private readonly string \_url;

public AtletaController(IConfiguration config)

{

\_url = config.GetConnectionString("API");

}

public async Task<ActionResult> ListAtletasDisciplina(int? disciplinaId)

{

try

{

(string, HttpResponseMessage) disciplinas =

await ConexionServidor.ClientSinBody(\_url + "/api/Disciplina", "GET");

if (disciplinas.Item2.IsSuccessStatusCode)

{

IEnumerable<DisciplinaVM> listaDiscplinas = JsonConvert.DeserializeObject<IEnumerable<DisciplinaVM>>(disciplinas.Item1) ??

throw new Exception("Fallo en obtener listado de disciplinas");

ViewBag.Disciplinas = listaDiscplinas;

}

else

{

TempData["ErrorMessage"] = disciplinas.Item1;

return View();

}

if (disciplinaId != null)

{

(string, HttpResponseMessage) atletas =

await ConexionServidor.ClientSinBody(\_url + "/api/Atleta/" + disciplinaId, "GET");

if (atletas.Item2.IsSuccessStatusCode)

{

IEnumerable<AtletaListaVM> listaAtletasvm = JsonConvert.DeserializeObject<IEnumerable<AtletaListaVM>>(atletas.Item1) ??

throw new Exception("Fallo en obtener listado de atletas");

return View(listaAtletasvm);

}

else

{

TempData["ErrorMessage"] = atletas.Item1;

}

}

return View();

}

catch (Exception ex)

{

TempData["ErrorMessage"] = ex.Message;

}

return View();

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: DisciplinaController.cs

Carpeta: C:\Users\cri\_1\Downloads\COPIAMVC\MVC\Controllers\DisciplinaController.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using MVC.Models.Disciplina;

using MVC.Utils;

using Newtonsoft.Json;

using static System.Runtime.InteropServices.JavaScript.JSType;

namespace MVC.Controllers

{

public class DisciplinaController : Controller

{

private readonly string \_url;

public DisciplinaController(IConfiguration config)

{

\_url = config.GetConnectionString("API");

}

public async Task<ActionResult> Index()

{

if (ManejoSession.GetRolLogueado(HttpContext) == "Digitador")

{

try

{

(string, HttpResponseMessage) disciplinas =

await ConexionServidor.ClientSinBody(\_url + "/api/Disciplina", "GET");

if (disciplinas.Item2.IsSuccessStatusCode)

{

IEnumerable<DisciplinaVM> listaDiscplinas = JsonConvert.DeserializeObject<IEnumerable<DisciplinaVM>>(disciplinas.Item1) ??

throw new Exception("Error al intentar obtener listado de disciplinas");

return View(listaDiscplinas);

}

else

{

throw new Exception(disciplinas.Item1);

}

}

catch (Exception ex)

{

return RedirectToAction("Index", "Error", new { code = 500, message = ex.Message });

}

}

else

{

return RedirectToAction("Index", "Error", new { code = 401, message = "No tiene permisos para ver esta información" });

}

}

public async Task<ActionResult> Details(int id)

{

if (id == 0)

{

TempData["ErrorMessage"] = "Id incorrecto";

return View();

}

if (ManejoSession.GetRolLogueado(HttpContext) == "Digitador")

{

try

{

string token = ManejoSession.GetToken(HttpContext)

?? throw new Exception("Fallo en la obtención del token");

(string, HttpResponseMessage) disciplinas =

await ConexionServidor.ClientSinBody(\_url + "/api/Disciplina/" + id.ToString(), "GET", token);

if (disciplinas.Item2.IsSuccessStatusCode)

{

DisciplinaVM res = JsonConvert.DeserializeObject<DisciplinaVM>(disciplinas.Item1);

return View(res);

}

else if ((int)disciplinas.Item2.StatusCode == StatusCodes.Status500InternalServerError)

{

throw new Exception(disciplinas.Item1);

}

else

{

TempData["ErrorMessage"] = disciplinas.Item1;

return View();

}

}

catch (Exception ex)

{

return RedirectToAction("Index", "Error", new { code = 500, message = ex.Message });

}

}

else

{

return RedirectToAction("Index", "Error", new { code = 401, message = "No tiene permisos para ver esta información" });

}

}

[HttpGet]

public async Task<ActionResult> Buscar(int? tipoBusqueda, int? iId, string? iNa)

{

if (ManejoSession.GetRolLogueado(HttpContext) == "Digitador")

{

try

{

if (tipoBusqueda is null || tipoBusqueda <= 0)

{

return View();

}

else if (tipoBusqueda == 1)

{

if(iId is null || iId <= 0)

{

TempData["ErrorMessage"] = "Error: Id incorrecto";

return View();

}

string token = ManejoSession.GetToken(HttpContext)

?? throw new Exception("Fallo en la obtención del token");

(string, HttpResponseMessage) disciplinas =

await ConexionServidor.ClientSinBody(\_url + "/api/Disciplina/" + iId, "GET", token);

if (disciplinas.Item2.IsSuccessStatusCode)

{

DisciplinaVM res = JsonConvert.DeserializeObject<DisciplinaVM>(disciplinas.Item1);

return View(res);

}

else if ((int)disciplinas.Item2.StatusCode == StatusCodes.Status500InternalServerError)

{

throw new Exception(disciplinas.Item1);

}

else

{

TempData["ErrorMessage"] = disciplinas.Item1;

return RedirectToAction("Buscar");

}

}

else

{

if(string.IsNullOrEmpty(iNa))

{

TempData["ErrorMessage"] = "Error: Nombre vacío";

return View();

}

iNa = iNa.Trim();

string token = ManejoSession.GetToken(HttpContext)

?? throw new Exception("Fallo en la obtención del token");

(string, HttpResponseMessage) disciplinas =

await ConexionServidor.ClientSinBody(\_url + "/api/Disciplina/Nombre/" + iNa, "GET", token);

if (disciplinas.Item2.IsSuccessStatusCode)

{

DisciplinaVM res = JsonConvert.DeserializeObject<DisciplinaVM>(disciplinas.Item1);

return View(res);

}

else if ((int)disciplinas.Item2.StatusCode == StatusCodes.Status500InternalServerError)

{

throw new Exception(disciplinas.Item1);

}

else

{

TempData["ErrorMessage"] = disciplinas.Item1;

return View();

}

}

}

catch (Exception ex)

{

return RedirectToAction("Index", "Error", new { code = 500, message = ex.Message });

}

}

else

{

return RedirectToAction("Index", "Error", new { code = 401, message = "No tiene permisos para ver esta información" });

}

}

[HttpGet]

public async Task<ActionResult> Create()

{

if (ManejoSession.GetRolLogueado(HttpContext) == "Digitador")

{

return View();

}

else

{

return RedirectToAction("Index", "Error", new { code = 401, message = "No tiene permisos para ver esta información" });

}

}

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<ActionResult> Create(DisciplinaInsertVM disciplinaInsertVM)

{

if (ManejoSession.GetRolLogueado(HttpContext) == "Digitador")

{

try

{

string token = ManejoSession.GetToken(HttpContext)

?? throw new Exception("Fallo en la obtención del token");

(string, HttpResponseMessage) disciplinas =

await ConexionServidor.ClientConBody(\_url + "/api/Disciplina/", "POST", disciplinaInsertVM, token);

if (disciplinas.Item2.IsSuccessStatusCode)

{

TempData["Message"] = "Disciplina creada correctamente";

return RedirectToAction(nameof(Create));

}

else if ((int)disciplinas.Item2.StatusCode == StatusCodes.Status500InternalServerError)

{

throw new Exception(disciplinas.Item1);

}

else

{

TempData["ErrorMessage"] = disciplinas.Item1;

return View();

}

}

catch (Exception ex)

{

return RedirectToAction("Index", "Error", new { code = 500, message = ex.Message });

}

}

else

{

return RedirectToAction("Index", "Error", new { code = 401, message = "No tiene permisos para ver esta información" });

}

}

public async Task<ActionResult> Edit(int id)

{

if (id == 0)

{

TempData["ErrorMessage"] = "Id incorrecto";

return View();

}

if (ManejoSession.GetRolLogueado(HttpContext) == "Digitador")

{

try

{

string token = ManejoSession.GetToken(HttpContext)

?? throw new Exception("Fallo en la obtención del token");

(string, HttpResponseMessage) disciplinas =

await ConexionServidor.ClientSinBody(\_url + "/api/Disciplina/" + id.ToString(), "GET", token);

if (disciplinas.Item2.IsSuccessStatusCode)

{

DisciplinaUpdateVM res = JsonConvert.DeserializeObject<DisciplinaUpdateVM>(disciplinas.Item1);

return View(res);

}

else if ((int)disciplinas.Item2.StatusCode == StatusCodes.Status500InternalServerError)

{

throw new Exception(disciplinas.Item1);

}

else

{

TempData["ErrorMessage"] = disciplinas.Item1;

return View();

}

}

catch (Exception ex)

{

return RedirectToAction("Index", "Error", new { code = 500, message = ex.Message });

}

}

else

{

return RedirectToAction("Index", "Error", new { code = 401, message = "No tiene permisos para ver esta información" });

}

}

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<ActionResult> Edit(int id, DisciplinaUpdateVM disciplinaUpdate)

{

if (id == 0)

{

TempData["ErrorMessage"] = "Id incorrecto";

return View();

}

if (ManejoSession.GetRolLogueado(HttpContext) == "Digitador")

{

try

{

disciplinaUpdate.Nombre = disciplinaUpdate.Nombre.Trim();

string token = ManejoSession.GetToken(HttpContext)

?? throw new Exception("Fallo en la obtención del token");

(string, HttpResponseMessage) updated =

await ConexionServidor.ClientConBody(\_url + "/api/Disciplina/" + id.ToString(), "PUT", disciplinaUpdate, token);

if (updated.Item2.IsSuccessStatusCode)

{

DisciplinaUpdateVM res = JsonConvert.DeserializeObject<DisciplinaUpdateVM>(updated.Item1);

TempData["Message"] = "Disciplina actualizada correctamente";

return RedirectToAction("Edit", new { Id = id });

}

else if ((int)updated.Item2.StatusCode == StatusCodes.Status500InternalServerError)

{

throw new Exception(updated.Item1);

}

else

{

TempData["ErrorMessage"] = updated.Item1;

return RedirectToAction("Edit", new { Id = id });

}

}

catch (Exception ex)

{

return RedirectToAction("Index", "Error", new { code = 500, message = ex.Message });

}

}

else

{

return RedirectToAction("Index", "Error", new { code = 401, message = "No tiene permisos para ver esta información" });

}

}

public async Task<ActionResult> DeleteDesdeBuscar(int id)

{

if (ManejoSession.GetIdLogueado(HttpContext) != null && ManejoSession.GetRolLogueado(HttpContext) == "Digitador")

{

try

{

string token = ManejoSession.GetToken(HttpContext)

?? throw new Exception("Fallo en la obtención del token");

(string, HttpResponseMessage) disciplinas =

await ConexionServidor.ClientSinBody(\_url + "/api/Disciplina/" + id.ToString(), "DELETE", token);

if (disciplinas.Item2.IsSuccessStatusCode)

{

TempData["Message"] = "Disciplina eliminada correctamente";

return RedirectToAction("Buscar");

}

else if ((int)disciplinas.Item2.StatusCode == StatusCodes.Status500InternalServerError)

{

throw new Exception(disciplinas.Item1);

}

else

{

TempData["ErrorMessage"] = disciplinas.Item1;

return RedirectToAction("Buscar");

}

}

catch (Exception ex)

{

return RedirectToAction("Index", "Error", new { code = 500, message = ex.Message });

}

}

else

{

return RedirectToAction("Index", "Error", new { code = 401, message = "No tiene permisos para ver esta información" });

}

}

public async Task<ActionResult> Delete(int id)

{

if (ManejoSession.GetIdLogueado(HttpContext) != null && ManejoSession.GetRolLogueado(HttpContext) == "Digitador")

{

try

{

string token = ManejoSession.GetToken(HttpContext)

?? throw new Exception("Fallo en la obtención del token");

(string, HttpResponseMessage) disciplinas =

await ConexionServidor.ClientSinBody(\_url + "/api/Disciplina/" + id.ToString(), "DELETE", token);

if (disciplinas.Item2.IsSuccessStatusCode)

{

TempData["Message"] = "Disciplina eliminada correctamente";

return RedirectToAction("Index");

}

else if ((int)disciplinas.Item2.StatusCode == StatusCodes.Status500InternalServerError)

{

throw new Exception(disciplinas.Item1);

}

else

{

TempData["ErrorMessage"] = disciplinas.Item1;

return RedirectToAction("Index");

}

}

catch (Exception ex)

{

return RedirectToAction("Index", "Error", new { code = 500, message = ex.Message });

}

}

else

{

return RedirectToAction("Index", "Error", new { code = 401, message = "No tiene permisos para ver esta información" });

}

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: ErrorController.cs

Carpeta: C:\Users\cri\_1\Downloads\COPIAMVC\MVC\Controllers\ErrorController.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Microsoft.AspNetCore.Mvc;

namespace MVC.Controllers

{

public class ErrorController : Controller

{

public IActionResult Index(int? code, string? message)

{

if (code != null)

{

ViewBag.Message = new { code, message };

}

return View();

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: EventoController.cs

Carpeta: C:\Users\cri\_1\Downloads\COPIAMVC\MVC\Controllers\EventoController.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using MVC.Models.Atleta;

using MVC.Models.Disciplina;

using MVC.Models.Evento;

using MVC.Utils;

using Newtonsoft.Json;

namespace MVC.Controllers

{

public class EventoController : Controller

{

private readonly string \_url;

public EventoController(IConfiguration config)

{

\_url = config.GetConnectionString("API");

}

[HttpGet]

public async Task<ActionResult> Busqueda(EventoBusquedaVM? vm)

{

if (ManejoSession.GetRolLogueado(HttpContext) == "Digitador")

{

// En teoría vm nunc debería ser null, pero por las dudas

if (vm is null || vm.GetType().GetProperties().All(prop => prop.GetValue(vm) == null))

{

return View();

}

else

{

try

{

// Validaciones del VM

if (vm.DisciplinaId <= 0)

{

ViewBag.ErrorMessage = "Id incorrecto";

return View();

}

if (

(vm.FchInicio != null && vm.FchFin == null) ||

(vm.FchFin != null && vm.FchInicio == null) ||

(vm.PuntajeMin != null && vm.PuntajeMax == null) ||

(vm.PuntajeMax != null && vm.PuntajeMin == null)

)

{

ViewBag.ErrorMessage = "Si ingresa una fecha debe ingresar ambas, lo mismo para los puntajes";

return View();

}

if (vm.FchInicio > vm.FchFin)

{

ViewBag.ErrorMessage = "Rango de fechas incorrecto";

return View();

}

if (vm.PuntajeMax < vm.PuntajeMin ||

vm.PuntajeMin < 0 ||

vm.PuntajeMin > 10 ||

vm.PuntajeMax < 0 ||

vm.PuntajeMax > 10

)

{

ViewBag.ErrorMessage = "Rango de puntajes incorrecto";

return View();

}

// Reflección para generar URL dinámicamente:

List<string> queries = vm.GetType()

.GetProperties()

.Where(p => p.GetValue(vm) != null)

.Select(p =>

{

if (p.PropertyType == typeof(DateTime?))

{

DateTime? otroP = p.GetValue(vm) as DateTime?;

return $"{p.Name}={otroP.Value.ToString("yyyy-MM-dd")}";

}

else

{

return $"{p.Name}={p.GetValue(vm)}";

}

})

.ToList();

// Ej. api/Evento?DisciplinaId=2&NombreEvento=Salto | Si no hay filtro que traiga todos

string queryString = queries.Any() ? "?" + string.Join("&", queries) : "";

string token = ManejoSession.GetToken(HttpContext)

?? throw new Exception("Fallo en la obtención del token");

(string, HttpResponseMessage) disciplinas =

await ConexionServidor.ClientSinBody(\_url + "/api/Evento" + queryString, "GET", token);

if (disciplinas.Item2.IsSuccessStatusCode)

{

var res = JsonConvert.DeserializeObject<IEnumerable<EventoListaVM>>(disciplinas.Item1)

?? throw new Exception("Fallo en obtener listado de disciplinas");

return View(res);

}

else if ((int)disciplinas.Item2.StatusCode == StatusCodes.Status500InternalServerError)

{

throw new Exception(disciplinas.Item1);

}

else

{

ViewBag.ErrorMessage = disciplinas.Item2;

return View();

}

}

catch (Exception ex)

{

return RedirectToAction("Index", "Error", new { code = 500, message = ex.Message });

}

}

}

else

{

return RedirectToAction("Index", "Error", new { code = 401, message = "No tiene permisos para ver esta información" });

}

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: HomeController.cs

Carpeta: C:\Users\cri\_1\Downloads\COPIAMVC\MVC\Controllers\HomeController.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Microsoft.AspNetCore.Mvc;

namespace MVC.Controllers

{

public class HomeController : Controller

{

public IActionResult Index()

{

return View();

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: UsuarioController.cs

Carpeta: C:\Users\cri\_1\Downloads\COPIAMVC\MVC\Controllers\UsuarioController.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using MVC.Models.Usuario;

using Microsoft.VisualStudio.Web.CodeGenerators.Mvc.Templates.BlazorIdentity.Pages.Manage;

using MVC.Utils;

using System.Net.Http.Headers;

using System.Security.Policy;

using Newtonsoft.Json;

using static System.Runtime.InteropServices.JavaScript.JSType;

using Microsoft.VisualStudio.Web.CodeGenerators.Mvc.View;

namespace MVC.Controllers

{

public class UsuarioController : Controller

{

private readonly string \_url;

public UsuarioController(IConfiguration config)

{

\_url = config.GetConnectionString("API");

}

public ActionResult Index()

{

int? usuarioId = ManejoSession.GetIdLogueado(HttpContext);

if (usuarioId != null) return RedirectToAction("Index", "Home");

return View();

}

[HttpPost]

public async Task<ActionResult> Index(string email, string password)

{

try

{

if(string.IsNullOrEmpty(email) || string.IsNullOrEmpty(password))

{

ViewBag.ErrorMessage = "Debe completar todos los campos";

return View();

}

CredencialesVM cred = new CredencialesVM

{

Email = email,

Contrasena = password

};

(string, HttpResponseMessage) res =

await ConexionServidor.ClientConBody<CredencialesVM>(\_url + "/api/Usuario/iniciar-sesion", "POST", cred);

if (res.Item2.IsSuccessStatusCode)

{

UsuarioLogueadoVM vm = JsonConvert.DeserializeObject<UsuarioLogueadoVM>(res.Item1) ??

throw new Exception("Fallo en iniciar-sesion de API");

HttpContext.Session.SetInt32("idLogueado", vm.Id);

HttpContext.Session.SetString("rolLogueado", vm.Rol);

HttpContext.Session.SetString("token", vm.Token);

return RedirectToAction("Index", "Home");

}

else

{

ViewBag.ErrorMessage = res.Item1;

}

}

catch (Exception ex)

{

ViewBag.ErrorMessage = ex.Message;

}

return View();

}

[HttpGet]

public ActionResult CerrarSesion()

{

HttpContext.Session.Clear();

return RedirectToAction("Index");

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: ErrorViewModel.cs

Carpeta: C:\Users\cri\_1\Downloads\COPIAMVC\MVC\Models\ErrorViewModel.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

namespace MVC.Models

{

public class ErrorViewModel

{

public string? RequestId { get; set; }

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

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: ConexionServidor.cs

Carpeta: C:\Users\cri\_1\Downloads\COPIAMVC\MVC\Utils\ConexionServidor.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Microsoft.VisualStudio.Web.CodeGenerators.Mvc.Templates.BlazorIdentity.Pages.Manage;

using MVC.Models.Usuario;

using System.Net.Http.Headers;

using System.Security.Cryptography;

using System.Security.Policy;

namespace MVC.Utils

{

public static class ConexionServidor

{

public static async Task<(string, HttpResponseMessage)> ClientSinBody(string url, string method)

{

HttpClient cliente = new HttpClient();

HttpResponseMessage respuesta;

switch (method)

{

case "GET":

respuesta = await cliente.GetAsync(url);

break;

case "DELETE":

respuesta = await cliente.DeleteAsync(url);

break;

default:

throw new Exception("Method incorrecto");

}

string resBody = await respuesta.Content.ReadAsStringAsync();

return (resBody, respuesta);

}

public static async Task<(string, HttpResponseMessage)> ClientSinBody(string url, string method, string token)

{

HttpClient cliente = new HttpClient();

cliente.DefaultRequestHeaders.Authorization = new AuthenticationHeaderValue("Bearer", token);

HttpResponseMessage respuesta;

switch (method)

{

case "GET":

respuesta = await cliente.GetAsync(url);

break;

case "DELETE":

respuesta = await cliente.DeleteAsync(url);

break;

default:

throw new Exception("Method incorrecto");

}

string resBody = await respuesta.Content.ReadAsStringAsync();

return (resBody, respuesta);

}

public static async Task<(string, HttpResponseMessage)> ClientConBody<T>(string url, string method, T VM)

{

HttpClient cliente = new HttpClient();

HttpResponseMessage respuesta;

switch (method)

{

case "POST":

respuesta = await cliente.PostAsJsonAsync(url, VM);

break;

case "PUT":

respuesta = await cliente.PutAsJsonAsync(url, VM);

break;

case "PATCH":

respuesta = await cliente.PatchAsJsonAsync(url, VM);

break;

default:

throw new Exception("Method incorrecto");

}

string resBody = await respuesta.Content.ReadAsStringAsync();

return (resBody, respuesta);

}

public static async Task<(string, HttpResponseMessage)> ClientConBody<T>(string url, string method, T VM, string token)

{

HttpClient cliente = new HttpClient();

cliente.DefaultRequestHeaders.Authorization = new AuthenticationHeaderValue("Bearer", token);

HttpResponseMessage respuesta;

switch (method)

{

case "POST":

respuesta = await cliente.PostAsJsonAsync(url, VM);

break;

case "PUT":

respuesta = await cliente.PutAsJsonAsync(url, VM);

break;

case "PATCH":

respuesta = await cliente.PatchAsJsonAsync(url, VM);

break;

default:

throw new Exception("Method incorrecto");

}

string resBody = await respuesta.Content.ReadAsStringAsync();

return (resBody, respuesta);

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: ManejoSession.cs

Carpeta: C:\Users\cri\_1\Downloads\COPIAMVC\MVC\Utils\ManejoSession.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

namespace MVC.Utils

{

public static class ManejoSession

{

public static int? GetIdLogueado(HttpContext httpContext)

{

return httpContext.Session.GetInt32("idLogueado");

}

public static string? GetRolLogueado(HttpContext httpContext)

{

return httpContext.Session.GetString("rolLogueado");

}

public static string? GetToken(HttpContext httpContext)

{

return httpContext.Session.GetString("token");

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: AtletaListaVM.cs

Carpeta: C:\Users\cri\_1\Downloads\COPIAMVC\MVC\Models\Atleta\AtletaListaVM.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System.ComponentModel;

using System.ComponentModel.DataAnnotations.Schema;

namespace MVC.Models.Atleta

{

public class AtletaListaVM

{

public int Id { get; set; }

public string Nombre { get; set; }

public string Apellido { get; set; }

[DisplayName("Nombre Completo")]

public string NombreCompleto { get; set; }

public string Sexo { get; set; }

[DisplayName("País")]

public string NombrePais { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: DisciplinaInsertVM.cs

Carpeta: C:\Users\cri\_1\Downloads\COPIAMVC\MVC\Models\Disciplina\DisciplinaInsertVM.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System.ComponentModel;

using System.ComponentModel.DataAnnotations;

namespace MVC.Models.Disciplina

{

public class DisciplinaInsertVM

{

[DisplayName("Nombre de disciplina")]

[Length(10,50,ErrorMessage = "El nombre de la disciplina debe tener entre 10 y 50 caracteres")]

[Required(ErrorMessage = "El nombre es requerido")]

public string Nombre { get; set; }

[DisplayName("Año de integración")]

[Required(ErrorMessage = "El año es requerido")]

[Range(1, 2024, ErrorMessage = "Año fuera del rango")]

public int AnioIntegracion { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: DisciplinaUpdateVM.cs

Carpeta: C:\Users\cri\_1\Downloads\COPIAMVC\MVC\Models\Disciplina\DisciplinaUpdateVM.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

namespace MVC.Models.Disciplina

{

public class DisciplinaUpdateVM

{

public int Id { get; set; }

public string Nombre { get; set; }

public int AnioIntegracion { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: DisciplinaVM.cs

Carpeta: C:\Users\cri\_1\Downloads\COPIAMVC\MVC\Models\Disciplina\DisciplinaVM.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System.ComponentModel;

namespace MVC.Models.Disciplina

{

public class DisciplinaVM

{

public int Id { get; set; }

public string Nombre { get; set; }

[DisplayName("Año integración")]

public int AnioIntegracion { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: EventoBusquedaVM.cs

Carpeta: C:\Users\cri\_1\Downloads\COPIAMVC\MVC\Models\Evento\EventoBusquedaVM.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System.ComponentModel.DataAnnotations;

using System.ComponentModel;

namespace MVC.Models.Evento

{

public class EventoBusquedaVM

{

[DisplayName("Disciplina Id")]

public int? DisciplinaId { get; set; }

[DisplayName("Fecha Inicio")]

[DataType(DataType.Date)]

public DateTime? FchInicio { get; set; }

[DisplayName("Fecha Final")]

[DataType(DataType.Date)]

public DateTime? FchFin { get; set; }

public string? NombreEvento { get; set; }

[DisplayName("Puntaje Mínimo")]

[Range(0, 10, ErrorMessage = "Los puntajes van desde 0 a 10")]

public int? PuntajeMin { get; set; }

[DisplayName("Puntaje Máximo")]

[Range(0, 10, ErrorMessage = "Los puntajes van desde 0 a 10")]

public int? PuntajeMax { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: EventoListaVM.cs

Carpeta: C:\Users\cri\_1\Downloads\COPIAMVC\MVC\Models\Evento\EventoListaVM.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using MVC.Models.Atleta;

using System.ComponentModel.DataAnnotations;

using System.ComponentModel;

namespace MVC.Models.Evento

{

public class EventoListaVM

{

public int EventoId { get; set; }

[DisplayName("Nombre de la Prueba")]

public string NombrePrueba { get; set; }

[DisplayName("Fecha Inicio")]

[DataType(DataType.Date)]

public DateTime FchInicio { get; set; }

[DataType(DataType.Date)]

[DisplayName("Fecha Finalización")]

public DateTime FchFin { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: CredencialesVM.cs

Carpeta: C:\Users\cri\_1\Downloads\COPIAMVC\MVC\Models\Usuario\CredencialesVM.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

namespace MVC.Models.Usuario

{

public class CredencialesVM

{

public string Email { get; set; }

public string Contrasena { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Archivo: UsuarioLogueadoVM.cs

Carpeta: C:\Users\cri\_1\Downloads\COPIAMVC\MVC\Models\Usuario\UsuarioLogueadoVM.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

namespace MVC.Models.Usuario

{

public class UsuarioLogueadoVM

{

public int Id { get; set; }

public string Rol { get; set; }

public string Token { get; set; }

}

}

# Listas de datos y prompts.

Los scripts con los datos se encuentran en la ruta SolObligatorioP3\ScriptsDatos. Estos archivos son leídos y ejecutados por el proyecto cuando se realiza el add-migration y/o el update-database en caso de que no exista la base creada.

De esta forma se contará con todos los datos precargados en la base para poder realizar las pruebas.

## 4.1 Listas de datos precargados.

Usuarios.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Id** | **Email** | **Rol** | **FechaRegistro** | **AdminRegistro** | **Nombre** | **Contraseña** |
| 1 | hola@gmail.com | 0 | 21/9/2024 0:00 | 0 | NULL | Hola1234! |
| 2 | hola2@gmail.com | 1 | 25/9/2024 8:00 | 3 | NULL | Hola1234! |
| 3 | nicolas@gmail.com | 0 | 25/9/2024 0:00 | 0 | Nicolas Gimenez | Nico@Gimenez12 |
| 4 | cristian@gmail.com | 0 | 25/9/2024 0:00 | 0 | Cristian Garcia | Cris@Garcia12 |

Países.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Id** | **Nombre** | **Habitantes** | **NomDelegado** | **TelDelegado** |
| 1 | Estados Unidos | 331000000 | John Smith | +1 123 456 7890 |
| 2 | Jamaica | 2970000 | Usain Bennett | +1 876 456 7890 |
| 3 | Rumania | 19000000 | Mihai Popescu | +40 123 456 789 |
| 4 | Rusia | 146000000 | Alexei Ivanov | +7 495 123 4567 |
| 5 | Finlandia | 5500000 | Kalle Virtanen | +358 123 456 789 |
| 6 | Arabia Saudita | 35000000 | Fahad Al-Saud | +966 123 456 789 |
| 7 | República Checa | 10700000 | Jan Novák | +420 123 456 789 |
| 8 | Argentina | 45000000 | Juan Pérez | +54 11 1234 5678 |
| 9 | Uruguay | 3500000 | Diego Rodríguez | +598 1234 5678 |
| 10 | Brasil | 213000000 | Carlos Silva | +55 21 1234 5678 |
| 11 | Canadá | 38000000 | Sophie Trudeau | +1 613 123 4567 |
| 12 | México | 128000000 | Alejandro Hernández | +52 55 1234 5678 |
| 13 | Alemania | 83000000 | Hans Müller | +49 30 1234 5678 |
| 14 | Francia | 67000000 | Pierre Dubois | +33 1 2345 6789 |
| 15 | Italia | 60000000 | Marco Rossi | +39 06 1234 5678 |
| 16 | España | 47000000 | Antonio García | +34 91 123 4567 |
| 17 | Japón | 126000000 | Takeshi Nakamura | +81 3 1234 5678 |
| 18 | China | 1400000000 | Li Wei | +86 10 1234 5678 |
| 19 | Australia | 25000000 | Jack Thompson | +61 2 1234 5678 |
| 20 | Sudáfrica | 59000000 | Sipho Ndlovu | +27 12 123 4567 |
| 21 | Nueva Zelanda | 5000000 | Lucy Smith | +64 4 123 4567 |
| 22 | Kenia | 54000000 | David Mwangi | +254 20 123 4567 |
| 23 | Corea del Sur | 51000000 | Kim Joon | +82 2 1234 5678 |
| 24 | Grecia | 10000000 | Nikos Papadopoulos | +30 21 1234 5678 |
| 25 | Egipto | 102000000 | Ahmed Hassan | +20 2 123 4567 |
| 26 | Turquía | 82000000 | Mehmet Yildiz | +90 212 123 4567 |
| 27 | Suecia | 10000000 | Erik Johansson | +46 8 123 4567 |
| 28 | Noruega | 5400000 | Ole Larsen | +47 23 123 4567 |
| 29 | India | 1366000000 | Rajesh Kumar | +91 11 1234 5678 |
| 30 | Reino Unido | 67000000 | James Brown | +44 20 1234 5678 |

Atletas.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Id** | **Nombre** | **Apellido** | **Sexo** | **PaísId** |
| 1 | Michael | Phelps | 0 | 1 |
| 2 | Usain | Bolt | 0 | 2 |
| 3 | Nadia | Comaneci | 1 | 3 |
| 4 | Yelena | Isinbayeva | 1 | 4 |
| 5 | Paavo | Nurmi | 0 | 5 |
| 6 | Hadi | Souan | 0 | 6 |
| 7 | Ester | Ledecká | 1 | 7 |
| 8 | Luciana | Aymar | 1 | 8 |
| 9 | Enzo | Francescoli | 0 | 9 |
| 10 | Pelé | Nascimento | 0 | 10 |
| 11 | Donovan | Bailey | 0 | 11 |
| 12 | Soraya | Jiménez | 1 | 12 |
| 13 | Birgit | Fischer | 1 | 13 |
| 14 | Marie-José | Pérec | 1 | 14 |
| 15 | Valentino | Rossi | 0 | 15 |
| 16 | Pau | Gasol | 0 | 16 |
| 17 | Kohei | Uchimura | 0 | 17 |
| 18 | Liu | Xiang | 0 | 18 |
| 19 | Cathy | Freeman | 1 | 19 |
| 20 | Caster | Semenya | 1 | 20 |
| 21 | Valerie | Adams | 1 | 21 |
| 22 | Eliud | Kipchoge | 0 | 22 |
| 23 | Kim | Yuna | 1 | 23 |
| 24 | Pyrros | Dimas | 0 | 24 |
| 25 | Hossam | Abdalla | 0 | 25 |
| 26 | Naim | Süleymanoglu | 0 | 26 |
| 27 | Gert | Fredriksson | 0 | 27 |
| 28 | Ole | Einar Bjørndalen | 0 | 28 |
| 29 | Milkha | Singh | 0 | 29 |
| 30 | Mo | Farah | 0 | 30 |
| 31 | Carl | Lewis | 0 | 1 |
| 32 | Shelly-Ann | Fraser-Pryce | 1 | 2 |
| 33 | Gabriela | Szabó | 1 | 3 |
| 34 | Aleksandr | Popov | 0 | 4 |
| 35 | Kaisa | Mäkäräinen | 1 | 5 |
| 36 | Saeed | Abdulghani | 0 | 6 |
| 37 | Jan | Železný | 0 | 7 |
| 38 | Lionel | Messi | 0 | 8 |
| 39 | Santiago | Urrutia | 0 | 9 |
| 40 | Gustavo | Kuerten | 0 | 10 |
| 41 | Clara | Hughes | 1 | 11 |
| 42 | Ana | Guevara | 1 | 12 |
| 43 | Katja | Seizinger | 1 | 13 |
| 44 | Renaud | Lavillenie | 0 | 14 |
| 45 | Federica | Pellegrini | 1 | 15 |
| 46 | Rafael | Nadal | 0 | 16 |
| 47 | Saori | Yoshida | 1 | 17 |
| 48 | Ma | Long | 0 | 18 |
| 49 | Ian | Thorpe | 0 | 19 |
| 50 | Wayde | van Niekerk | 0 | 20 |
| 51 | Florence | Griffith-Joyner | 1 | 1 |
| 52 | Maurice | Greene | 0 | 1 |
| 53 | Yohan | Blake | 0 | 2 |
| 54 | Melaine | Walker | 1 | 2 |
| 55 | Alina | Astafei | 1 | 3 |
| 56 | Andrei | Rădulescu | 0 | 3 |
| 57 | Dina | Asher-Smith | 1 | 30 |
| 58 | Jessica | Ennis-Hill | 1 | 30 |
| 59 | Daria | Klishina | 1 | 4 |
| 60 | Ilya | Zakharov | 0 | 4 |
| 61 | Tero | Pitkämäki | 0 | 5 |
| 62 | Minna | Nikkanen | 1 | 5 |
| 63 | Haider | Al-Muwallad | 0 | 6 |
| 64 | Reem | Al-Jizawi | 1 | 6 |
| 65 | Barbora | Špotáková | 1 | 7 |
| 66 | Tomáš | Dvořák | 0 | 7 |
| 67 | Gabriela | Sabatini | 1 | 8 |
| 68 | Marcos | Maidana | 0 | 8 |
| 69 | Luis | Suárez | 0 | 9 |
| 70 | Virginia | Saravia | 1 | 9 |
| 71 | Gisele | Schmidt | 1 | 10 |
| 72 | Adriano | Leite | 0 | 10 |
| 73 | Christine | Sinclair | 1 | 11 |
| 74 | Simon | Whitfield | 0 | 11 |
| 75 | Guadalupe | Luna | 1 | 12 |
| 76 | Omar | Fierro | 0 | 12 |
| 77 | Monica | Seles | 1 | 13 |
| 78 | Steffi | Graf | 1 | 13 |
| 79 | Armand | Duplantis | 0 | 27 |
| 80 | Daniel | Stähl | 0 | 27 |
| 81 | Karsten | Warholm | 0 | 28 |
| 82 | Therese | Johaug | 1 | 28 |
| 83 | Zhang | Ning | 0 | 18 |
| 84 | Wang | Meng | 1 | 18 |
| 85 | Neeraj | Chopra | 0 | 29 |
| 86 | PV | Sindhu | 1 | 29 |
| 87 | Ezekiel | Kemboi | 0 | 22 |
| 88 | Vivian | Cheruiyot | 1 | 22 |
| 89 | John | Walker | 0 | 21 |
| 90 | Dame | Valerie Adams | 1 | 21 |
| 91 | Miruts | Yifter | 0 | 22 |
| 92 | Tirunesh | Dibaba | 1 | 22 |
| 93 | Yuzuru | Hanyu | 0 | 17 |
| 94 | Ryoko | Tani | 1 | 17 |
| 95 | Ashleigh | Barty | 1 | 19 |
| 96 | Ian | Millar | 0 | 19 |
| 97 | Mpumelelo | Mbangwa | 0 | 20 |
| 98 | Zola | Budd | 1 | 20 |
| 99 | Elvan | Abeylegesse | 1 | 26 |
| 100 | Riza | Kayaalp | 0 | 26 |

Disciplinas.

|  |  |  |
| --- | --- | --- |
| **Id** | **Nombre** | **AnioIntegracion** |
| 1 | Natacion Olimpica | 1999 |
| 2 | Atletismo de Larga Distancia | 2005 |
| 3 | Triatlón Internacional Competitivo | 2012 |
| 4 | Escalada en Roca para Competencias | 2016 |
| 5 | Ciclismo de Montaña Extremo | 2009 |
| 6 | Danza Deportiva Internacional | 2014 |
| 7 | Levantamiento de Pesas Olímpico | 2001 |
| 8 | Esgrima de Alto Rendimiento | 2003 |
| 9 | Remando en Competencias Internacionales | 2010 |
| 10 | Tenis de Mesa Profesional | 2007 |
| 11 | Baloncesto 3x3 Competitivo | 2015 |
| 12 | Karate Olímpico Internacional | 2021 |
| 13 | Fútbol Sala Profesional | 2018 |
| 14 | Patinaje Artístico en Hielo | 2006 |
| 15 | Tiro con Arco Olímpico | 1997 |
| 16 | Voleibol de Playa Profesional | 1999 |
| 17 | Judo Olímpico Internacional | 1993 |
| 18 | Boxeo de Alto Rendimiento | 1998 |
| 19 | Snowboard en Medio Tubo | 2014 |
| 20 | Halterofilia Femenil Competitiva | 2000 |
| 21 | Rugby 7s Internacional | 2016 |
| 22 | Bádminton Competitivo Olímpico | 1992 |
| 23 | Polo Acuático Profesional | 1986 |
| 24 | Esquí Alpino en Competencia | 1976 |
| 25 | Vela Competitiva Internacional | 1980 |
| 26 | Surfing Olímpico en Olas Grandes | 2020 |
| 27 | Ciclismo de Pista Olímpico | 1988 |
| 28 | Tiro Deportivo de Precisión | 1991 |
| 29 | Equitación de Alto Nivel | 1972 |
| 30 | Pentatlón Moderno Olímpico | 1984 |
| 31 | Hockey Sobre Césped Profesional | 1988 |
| 32 | Esquí de Fondo Competitivo | 1994 |
| 33 | Curling Profesional en Equipos | 1998 |
| 34 | Escalada Deportiva en Búlder | 2021 |
| 35 | Skateboarding en Competencia | 2020 |
| 36 | Gimnasia Artística Internacional | 1952 |
| 37 | Lucha Libre Olímpica Profesional | 2008 |
| 38 | Natación Sincronizada de Equipo | 1984 |
| 39 | Triatlón Corto para Competencia | 2000 |
| 40 | Handball Olímpico Internacional | 1996 |

Atletas.

|  |  |
| --- | --- |
| **AtletaId** | **DisciplinaId** |
| 1 | 1 |
| 2 | 1 |
| 3 | 1 |
| 4 | 1 |
| 5 | 1 |
| 6 | 1 |
| 7 | 1 |
| 8 | 1 |
| 9 | 1 |
| 10 | 1 |
| 11 | 1 |
| 12 | 1 |
| 13 | 1 |
| 14 | 1 |
| 15 | 1 |
| 11 | 2 |
| 12 | 2 |
| 13 | 2 |
| 14 | 2 |
| 15 | 2 |
| 16 | 2 |
| 17 | 2 |
| 18 | 2 |
| 19 | 2 |
| 20 | 2 |
| 21 | 2 |
| 22 | 2 |
| 23 | 2 |
| 24 | 2 |
| 25 | 2 |
| 21 | 3 |
| 22 | 3 |
| 23 | 3 |
| 24 | 3 |
| 25 | 3 |
| 26 | 3 |
| 27 | 3 |
| 28 | 3 |
| 29 | 3 |
| 30 | 3 |
| 31 | 3 |
| 32 | 3 |
| 33 | 3 |
| 34 | 3 |
| 35 | 3 |
| 26 | 4 |
| 27 | 4 |
| 28 | 4 |
| 29 | 4 |
| 30 | 4 |
| 31 | 4 |
| 32 | 4 |
| 33 | 4 |
| 34 | 4 |
| 35 | 4 |
| 36 | 4 |
| 37 | 4 |
| 38 | 4 |
| 39 | 4 |
| 40 | 4 |
| 1 | 5 |
| 2 | 5 |
| 3 | 5 |
| 4 | 5 |
| 5 | 5 |
| 16 | 5 |
| 17 | 5 |
| 18 | 5 |
| 19 | 5 |
| 20 | 5 |
| 6 | 6 |
| 7 | 6 |
| 8 | 6 |
| 9 | 6 |
| 10 | 6 |
| 21 | 6 |
| 22 | 6 |
| 23 | 6 |
| 24 | 6 |
| 25 | 6 |
| 11 | 7 |
| 12 | 7 |
| 13 | 7 |
| 14 | 7 |
| 15 | 7 |
| 31 | 7 |
| 32 | 7 |
| 33 | 7 |
| 34 | 7 |
| 35 | 7 |
| 16 | 8 |
| 17 | 8 |
| 18 | 8 |
| 19 | 8 |
| 20 | 8 |
| 36 | 8 |
| 37 | 8 |
| 38 | 8 |
| 39 | 8 |
| 40 | 8 |
| 1 | 9 |
| 2 | 9 |
| 3 | 9 |
| 4 | 9 |
| 5 | 9 |
| 26 | 9 |
| 27 | 9 |
| 28 | 9 |
| 29 | 9 |
| 30 | 9 |
| 6 | 10 |
| 7 | 10 |
| 8 | 10 |
| 9 | 10 |
| 10 | 10 |
| 31 | 10 |
| 32 | 10 |
| 33 | 10 |
| 34 | 10 |
| 35 | 10 |
| 51 | 11 |
| 52 | 11 |
| 53 | 11 |
| 54 | 11 |
| 55 | 11 |
| 56 | 12 |
| 57 | 12 |
| 58 | 12 |
| 59 | 12 |
| 60 | 12 |
| 61 | 13 |
| 62 | 13 |
| 63 | 13 |
| 64 | 13 |
| 65 | 13 |
| 66 | 14 |
| 67 | 14 |
| 68 | 14 |
| 69 | 14 |
| 70 | 14 |
| 71 | 15 |
| 72 | 15 |
| 73 | 15 |
| 74 | 15 |
| 75 | 15 |
| 76 | 16 |
| 77 | 16 |
| 78 | 16 |
| 79 | 16 |
| 80 | 16 |
| 81 | 17 |
| 82 | 17 |
| 83 | 17 |
| 84 | 17 |
| 85 | 17 |
| 86 | 18 |
| 87 | 18 |
| 88 | 18 |
| 89 | 18 |
| 90 | 18 |
| 91 | 19 |
| 92 | 19 |
| 93 | 19 |
| 94 | 19 |
| 95 | 19 |
| 96 | 20 |
| 97 | 20 |
| 98 | 20 |
| 99 | 20 |
| 100 | 20 |
| 51 | 21 |
| 52 | 21 |
| 53 | 21 |
| 54 | 21 |
| 55 | 21 |
| 56 | 22 |
| 57 | 22 |
| 58 | 22 |
| 59 | 22 |
| 60 | 22 |
| 61 | 23 |
| 62 | 23 |
| 63 | 23 |
| 64 | 23 |
| 65 | 23 |
| 66 | 24 |
| 67 | 24 |
| 68 | 24 |
| 69 | 24 |
| 70 | 24 |
| 71 | 25 |
| 72 | 25 |
| 73 | 25 |
| 74 | 25 |
| 75 | 25 |
| 76 | 26 |
| 77 | 26 |
| 78 | 26 |
| 79 | 26 |
| 80 | 26 |
| 81 | 27 |
| 82 | 27 |
| 83 | 27 |
| 84 | 27 |
| 85 | 27 |
| 86 | 28 |
| 87 | 28 |
| 88 | 28 |
| 89 | 28 |
| 90 | 28 |
| 91 | 29 |
| 92 | 29 |
| 93 | 29 |
| 94 | 29 |
| 95 | 29 |
| 96 | 30 |
| 97 | 30 |
| 98 | 30 |
| 99 | 30 |
| 100 | 30 |
| 51 | 31 |
| 52 | 31 |
| 53 | 31 |
| 54 | 31 |
| 55 | 31 |
| 56 | 32 |
| 57 | 32 |
| 58 | 32 |
| 59 | 32 |
| 60 | 32 |
| 61 | 33 |
| 62 | 33 |
| 63 | 33 |
| 64 | 33 |
| 65 | 33 |
| 66 | 34 |
| 67 | 34 |
| 68 | 34 |
| 69 | 34 |
| 70 | 34 |
| 71 | 35 |
| 72 | 35 |
| 73 | 35 |
| 74 | 35 |
| 75 | 35 |
| 76 | 36 |
| 77 | 36 |
| 78 | 36 |
| 79 | 36 |
| 80 | 36 |
| 81 | 37 |
| 82 | 37 |
| 83 | 37 |
| 84 | 37 |
| 85 | 37 |
| 86 | 38 |
| 87 | 38 |
| 88 | 38 |
| 89 | 38 |
| 90 | 38 |
| 91 | 39 |
| 92 | 39 |
| 93 | 39 |
| 94 | 39 |
| 95 | 39 |
| 96 | 40 |
| 97 | 40 |
| 98 | 40 |
| 99 | 40 |
| 100 | 40 |

Eventos.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Id** | **Nombre** | **DisciplinaId** | **FchInicio** | **FchFin** |
| 1 | Natación Olímpica 100m Prueba | 1 | 1/11/2024 0:00 | 2/11/2024 0:00 |
| 2 | Carrera de 10km | 2 | 5/11/2024 0:00 | 6/11/2024 0:00 |
| 3 | Triatlón Largo Internacional | 3 | 10/11/2024 0:00 | 12/11/2024 0:00 |
| 4 | Campeonato Nacional de Escalada | 4 | 15/11/2024 0:00 | 16/11/2024 0:00 |
| 5 | Copa Montaña Extrema | 5 | 20/11/2024 0:00 | 22/11/2024 0:00 |
| 6 | Danza Internacional Clásica | 6 | 25/11/2024 0:00 | 26/11/2024 0:00 |
| 7 | Levantamiento Pesas Juvenil | 7 | 1/12/2024 0:00 | 2/12/2024 0:00 |
| 8 | Esgrima Campeonato Regional | 8 | 5/12/2024 0:00 | 6/12/2024 0:00 |
| 9 | Remo Olímpico Abierto | 9 | 10/12/2024 0:00 | 11/12/2024 0:00 |
| 10 | Tenis de Mesa Pro Challenge | 10 | 15/12/2024 0:00 | 16/12/2024 0:00 |
| 11 | Torneo Baloncesto 3x3 | 11 | 20/12/2024 0:00 | 21/12/2024 0:00 |
| 12 | Karate Internacional Open | 12 | 25/12/2024 0:00 | 26/12/2024 0:00 |
| 13 | Torneo de Futsal | 13 | 5/1/2025 0:00 | 6/1/2025 0:00 |
| 14 | Patinaje Artístico Sobre Hielo | 14 | 10/1/2025 0:00 | 11/1/2025 0:00 |
| 15 | Torneo de Tiro con Arco | 15 | 15/1/2025 0:00 | 16/1/2025 0:00 |
| 16 | Copa Voleibol de Playa | 16 | 20/1/2025 0:00 | 21/1/2025 0:00 |
| 17 | Competencia de Judo Olímpico | 17 | 25/1/2025 0:00 | 26/1/2025 0:00 |
| 18 | Torneo de Boxeo | 18 | 1/2/2025 0:00 | 2/2/2025 0:00 |
| 19 | Copa Snowboard | 19 | 5/2/2025 0:00 | 6/2/2025 0:00 |
| 20 | Halterofilia Femenina | 20 | 10/2/2025 0:00 | 11/2/2025 0:00 |
| 21 | Rugby 7s Internacional | 21 | 15/2/2025 0:00 | 16/2/2025 0:00 |
| 22 | Campeonato de Bádminton | 22 | 20/2/2025 0:00 | 21/2/2025 0:00 |
| 23 | Torneo de Polo Acuático | 23 | 25/2/2025 0:00 | 26/2/2025 0:00 |
| 24 | Copa de Esquí Alpino | 24 | 1/3/2025 0:00 | 2/3/2025 0:00 |
| 25 | Regatas Internacionales de Vela | 25 | 5/3/2025 0:00 | 6/3/2025 0:00 |
| 26 | Torneo de Surfing Olímpico | 26 | 10/3/2025 0:00 | 11/3/2025 0:00 |
| 27 | Ciclismo de Pista Nacional | 27 | 15/3/2025 0:00 | 16/3/2025 0:00 |
| 28 | Campeonato de Tiro Deportivo | 28 | 20/3/2025 0:00 | 21/3/2025 0:00 |
| 29 | Concurso de Equitación | 29 | 25/3/2025 0:00 | 26/3/2025 0:00 |
| 30 | Pentatlón Moderno | 30 | 30/3/2025 0:00 | 31/3/2025 0:00 |
| 31 | Hockey Sobre Césped Abierto | 31 | 5/4/2025 0:00 | 6/4/2025 0:00 |
| 32 | Esquí de Fondo Olímpico | 32 | 10/4/2025 0:00 | 11/4/2025 0:00 |
| 33 | Copa Internacional de Curling | 33 | 15/4/2025 0:00 | 16/4/2025 0:00 |
| 34 | Torneo de Escalada en Búlder | 34 | 20/4/2025 0:00 | 21/4/2025 0:00 |
| 35 | Skateboarding Pro Series | 35 | 25/4/2025 0:00 | 26/4/2025 0:00 |
| 36 | Gimnasia Artística Clásica | 36 | 1/5/2025 0:00 | 2/5/2025 0:00 |
| 37 | Competencia de Lucha Libre | 37 | 5/5/2025 0:00 | 6/5/2025 0:00 |
| 38 | Nado Sincronizado en Equipo | 38 | 10/5/2025 0:00 | 11/5/2025 0:00 |
| 39 | Triatlón Corto Olímpico | 39 | 15/5/2025 0:00 | 16/5/2025 0:00 |
| 40 | Torneo Olímpico de Handball | 40 | 20/5/2025 0:00 | 21/5/2025 0:00 |
| 41 | Carrera 20km | 2 | 25/5/2025 0:00 | 26/5/2025 0:00 |
| 42 | Carrera ciclismo montaña | 5 | 30/5/2025 0:00 | 31/5/2025 0:00 |
| 43 | Tiro al arco 100m | 15 | 5/6/2025 0:00 | 6/6/2025 0:00 |
| 44 | Campeonato Natación 500m | 1 | 10/6/2025 0:00 | 11/6/2025 0:00 |
| 45 | Nado Sincronizado Individual | 38 | 15/6/2025 0:00 | 16/6/2025 0:00 |
| 46 | Esgrima Internacional | 8 | 20/6/2025 0:00 | 21/6/2025 0:00 |
| 47 | Danza Deportiva Showdown | 6 | 25/6/2025 0:00 | 26/6/2025 0:00 |
| 48 | Ciclismo de Montaña Open | 5 | 30/6/2025 0:00 | 1/7/2025 0:00 |
| 49 | Atletismo Maratón Olímpico | 2 | 5/7/2025 0:00 | 6/7/2025 0:00 |
| 50 | Torneo de Boxeo Femenino | 18 | 10/7/2025 0:00 | 11/7/2025 0:00 |
| 51 | Competencia Snowboard | 19 | 15/7/2025 0:00 | 16/7/2025 0:00 |
| 52 | Tiro con Arco Precisión | 15 | 20/7/2025 0:00 | 21/7/2025 0:00 |
| 53 | Ciclismo Pista Elite | 27 | 25/7/2025 0:00 | 26/7/2025 0:00 |
| 54 | Esquí Alpino Nacional | 24 | 30/7/2025 0:00 | 31/7/2025 0:00 |
| 55 | Triatlón Olímpico Clásico | 3 | 5/8/2025 0:00 | 6/8/2025 0:00 |
| 56 | Levantamiento Pesas Internacional | 7 | 10/8/2025 0:00 | 11/8/2025 0:00 |
| 57 | Remo Competitivo Nacional | 9 | 15/8/2025 0:00 | 16/8/2025 0:00 |
| 58 | Torneo de Judo Mixto | 17 | 20/8/2025 0:00 | 21/8/2025 0:00 |
| 59 | Baloncesto 3x3 Internacional | 11 | 25/8/2025 0:00 | 26/8/2025 0:00 |
| 60 | Halterofilia Regional | 20 | 30/8/2025 0:00 | 31/8/2025 0:00 |
| 61 | Torneo Internacional de Rugby 7s | 21 | 5/9/2025 0:00 | 6/9/2025 0:00 |
| 62 | Abierto Nacional de Bádminton | 22 | 10/9/2025 0:00 | 11/9/2025 0:00 |
| 63 | Copa de Polo Acuático | 23 | 15/9/2025 0:00 | 16/9/2025 0:00 |
| 64 | Torneo de Vela Olímpica | 25 | 20/9/2025 0:00 | 21/9/2025 0:00 |
| 65 | Campeonato de Surf en Olas Grandes | 26 | 25/9/2025 0:00 | 26/9/2025 0:00 |
| 66 | Copa Nacional de Curling | 33 | 30/9/2025 0:00 | 1/10/2025 0:00 |
| 67 | Esquí de Fondo Elite | 32 | 5/10/2025 0:00 | 6/10/2025 0:00 |
| 68 | Torneo de Skateboarding Olímpico | 35 | 10/10/2025 0:00 | 11/10/2025 0:00 |
| 69 | Competencia de Gimnasia Artística | 36 | 15/10/2025 0:00 | 16/10/2025 0:00 |
| 70 | Abierto de Lucha Libre | 37 | 20/10/2025 0:00 | 21/10/2025 0:00 |
| 71 | Nado Sincronizado Dúo | 38 | 25/10/2025 0:00 | 26/10/2025 0:00 |
| 72 | Ciclismo de Montaña Internacional | 5 | 30/10/2025 0:00 | 31/10/2025 0:00 |
| 73 | Triatlón Abierto de Velocidad | 39 | 5/11/2025 0:00 | 6/11/2025 0:00 |
| 74 | Torneo Nacional de Handball | 40 | 10/11/2025 0:00 | 11/11/2025 0:00 |
| 75 | Desafío Nacional de Escalada en Roca | 4 | 15/11/2025 0:00 | 16/11/2025 0:00 |
| 76 | Levantamiento de Pesas Femenil Pro | 20 | 20/11/2025 0:00 | 21/11/2025 0:00 |
| 77 | Tiro con Arco de Alta Precisión | 15 | 25/11/2025 0:00 | 26/11/2025 0:00 |
| 78 | Remo Olímpico de Velocidad | 9 | 30/11/2025 0:00 | 1/12/2025 0:00 |
| 79 | Esgrima de Alto Rendimiento | 8 | 5/12/2025 0:00 | 6/12/2025 0:00 |
| 80 | Torneo Internacional de Voleibol de Playa | 16 | 10/12/2025 0:00 | 11/12/2025 0:00 |

Puntaje evento atleta.

|  |  |  |
| --- | --- | --- |
| **AtletaId** | **EventoId** | **Puntaje** |
| 2 | 5 | 1 |
| 2 | 9 | 3 |
| 3 | 5 | 8 |
| 3 | 9 | 3 |
| 4 | 5 | 8 |
| 4 | 9 | 2 |
| 5 | 48 | 5 |
| 5 | 57 | 10 |
| 7 | 6 | 5 |
| 7 | 10 | 10 |
| 8 | 6 | 1 |
| 8 | 10 | 4 |
| 9 | 6 | 10 |
| 9 | 10 | 1 |
| 10 | 47 | 9 |
| 11 | 1 | 2 |
| 12 | 7 | 3 |
| 13 | 1 | 3 |
| 13 | 7 | 10 |
| 14 | 1 | 8 |
| 14 | 7 | 8 |
| 15 | 2 | 1 |
| 15 | 56 | 3 |
| 16 | 2 | 2 |
| 16 | 48 | 5 |
| 17 | 2 | 10 |
| 17 | 8 | 9 |
| 17 | 48 | 2 |
| 18 | 8 | 1 |
| 18 | 49 | 9 |
| 18 | 72 | 5 |
| 19 | 8 | 4 |
| 19 | 49 | 6 |
| 19 | 72 | 1 |
| 20 | 46 | 6 |
| 20 | 49 | 9 |
| 20 | 72 | 5 |
| 21 | 47 | 0 |
| 22 | 3 | 5 |
| 22 | 47 | 0 |
| 23 | 3 | 8 |
| 24 | 3 | 5 |
| 25 | 55 | 9 |
| 26 | 55 | 0 |
| 26 | 57 | 7 |
| 27 | 4 | 2 |
| 27 | 55 | 5 |
| 27 | 57 | 6 |
| 28 | 4 | 8 |
| 28 | 78 | 5 |
| 29 | 4 | 10 |
| 29 | 78 | 1 |
| 30 | 75 | 8 |
| 30 | 78 | 7 |
| 31 | 56 | 3 |
| 31 | 75 | 7 |
| 32 | 56 | 6 |
| 32 | 75 | 7 |
| 36 | 46 | 7 |
| 37 | 46 | 7 |
| 38 | 79 | 8 |
| 39 | 79 | 5 |
| 40 | 79 | 6 |
| 51 | 60 | 7 |
| 52 | 11 | 10 |
| 52 | 21 | 0 |
| 52 | 31 | 6 |
| 52 | 60 | 3 |
| 53 | 11 | 1 |
| 53 | 21 | 4 |
| 53 | 31 | 5 |
| 53 | 76 | 6 |
| 54 | 11 | 6 |
| 54 | 21 | 9 |
| 54 | 31 | 2 |
| 54 | 76 | 1 |
| 55 | 59 | 6 |
| 55 | 61 | 10 |
| 55 | 76 | 5 |
| 56 | 59 | 9 |
| 56 | 61 | 5 |
| 57 | 12 | 0 |
| 57 | 22 | 8 |
| 57 | 32 | 8 |
| 57 | 59 | 9 |
| 57 | 61 | 2 |
| 58 | 12 | 1 |
| 58 | 22 | 6 |
| 58 | 32 | 0 |
| 59 | 12 | 8 |
| 59 | 22 | 1 |
| 59 | 32 | 10 |
| 60 | 62 | 4 |
| 60 | 67 | 3 |
| 61 | 62 | 2 |
| 61 | 67 | 8 |
| 62 | 13 | 8 |
| 62 | 23 | 5 |
| 62 | 33 | 3 |
| 62 | 62 | 10 |
| 62 | 67 | 6 |
| 63 | 13 | 9 |
| 63 | 23 | 7 |
| 63 | 33 | 10 |
| 64 | 13 | 0 |
| 64 | 23 | 5 |
| 64 | 33 | 7 |
| 65 | 63 | 9 |
| 65 | 66 | 1 |
| 66 | 63 | 3 |
| 66 | 66 | 0 |
| 67 | 14 | 2 |
| 67 | 24 | 1 |
| 67 | 34 | 10 |
| 67 | 63 | 0 |
| 67 | 66 | 0 |
| 68 | 14 | 9 |
| 68 | 24 | 9 |
| 68 | 34 | 9 |
| 69 | 14 | 2 |
| 69 | 24 | 4 |
| 69 | 34 | 7 |
| 70 | 54 | 9 |
| 71 | 54 | 1 |
| 72 | 15 | 7 |
| 72 | 25 | 2 |
| 72 | 35 | 7 |
| 72 | 54 | 0 |
| 73 | 15 | 3 |
| 73 | 25 | 7 |
| 73 | 35 | 3 |
| 74 | 15 | 10 |
| 74 | 25 | 10 |
| 74 | 35 | 3 |
| 75 | 52 | 4 |
| 75 | 64 | 1 |
| 75 | 68 | 9 |
| 76 | 52 | 10 |
| 76 | 64 | 4 |
| 76 | 68 | 6 |
| 77 | 16 | 2 |
| 77 | 26 | 1 |
| 77 | 36 | 5 |
| 77 | 52 | 10 |
| 77 | 64 | 9 |
| 77 | 68 | 0 |
| 78 | 16 | 9 |
| 78 | 26 | 9 |
| 78 | 36 | 9 |
| 78 | 77 | 6 |
| 79 | 16 | 5 |
| 79 | 26 | 5 |
| 79 | 36 | 10 |
| 79 | 77 | 3 |
| 80 | 65 | 0 |
| 80 | 69 | 6 |
| 80 | 77 | 4 |
| 80 | 80 | 10 |
| 81 | 65 | 7 |
| 81 | 69 | 9 |
| 81 | 80 | 0 |
| 82 | 17 | 9 |
| 82 | 27 | 6 |
| 82 | 37 | 5 |
| 82 | 65 | 9 |
| 82 | 69 | 5 |
| 82 | 80 | 3 |
| 83 | 17 | 1 |
| 83 | 27 | 2 |
| 83 | 37 | 6 |
| 84 | 17 | 8 |
| 84 | 27 | 10 |
| 84 | 37 | 5 |
| 85 | 53 | 9 |
| 85 | 58 | 7 |
| 85 | 70 | 4 |
| 86 | 53 | 3 |
| 86 | 58 | 8 |
| 86 | 70 | 2 |
| 87 | 18 | 4 |
| 87 | 28 | 1 |
| 87 | 38 | 4 |
| 87 | 53 | 9 |
| 87 | 58 | 5 |
| 87 | 70 | 10 |
| 88 | 18 | 7 |
| 88 | 28 | 10 |
| 88 | 38 | 4 |
| 89 | 18 | 5 |
| 89 | 28 | 2 |
| 89 | 38 | 6 |
| 90 | 45 | 9 |
| 90 | 50 | 9 |
| 91 | 45 | 3 |
| 91 | 50 | 3 |
| 92 | 19 | 6 |
| 92 | 29 | 1 |
| 92 | 39 | 9 |
| 92 | 45 | 10 |
| 92 | 50 | 6 |
| 93 | 19 | 10 |
| 93 | 29 | 0 |
| 93 | 39 | 8 |
| 93 | 71 | 9 |
| 94 | 19 | 4 |
| 94 | 29 | 0 |
| 94 | 39 | 5 |
| 94 | 71 | 7 |
| 95 | 51 | 1 |
| 95 | 71 | 3 |
| 95 | 73 | 1 |
| 96 | 51 | 10 |
| 96 | 73 | 3 |
| 97 | 20 | 1 |
| 97 | 30 | 5 |
| 97 | 40 | 2 |
| 97 | 51 | 9 |
| 97 | 73 | 5 |
| 97 | 74 | 4 |
| 98 | 20 | 9 |
| 98 | 30 | 9 |
| 98 | 40 | 8 |
| 98 | 74 | 8 |
| 99 | 20 | 7 |
| 99 | 30 | 2 |
| 99 | 40 | 8 |
| 99 | 74 | 4 |
| 100 | 60 | 1 |

## 4.2 Prompts para generar datos.

Promp para generar disiciplinas.

“Tengo un proyecto de Juegos Olimpicos y cuento con este listado de Disciplinas:

SET IDENTITY\_INSERT [dbo].[Disciplinas] ON

INSERT INTO [dbo].[Disciplinas] ([Id], [Nombre\_Valor], [AnioIntegracion])

VALUES

(1, N'Natacion Olimpica', 1999),

(2, N'Atletismo de Larga Distancia', 2005),

(3, N'Triatlón Internacional Competitivo', 2012),

(4, N'Escalada en Roca para Competencias', 2016),

(5, N'Ciclismo de Montaña Extremo', 2009),

(6, N'Danza Deportiva Internacional', 2014),

(7, N'Levantamiento de Pesas Olímpico', 2001),

(8, N'Esgrima de Alto Rendimiento', 2003),

(9, N'Remando en Competencias Internacionales', 2010),

(10, N'Tenis de Mesa Profesional', 2007);

SET IDENTITY\_INSERT [dbo].[Disciplinas] OFF

Consideración: el nombre de la disciplina es único y debe tener entre 10 y 50 caracteres

Agregame 30 disciplinas”

Texto

Descripción generada automáticamente

Respuesta.

Captura de pantalla de computadora

Descripción generada automáticamente

SET IDENTITY\_INSERT [dbo].[Disciplinas] ON

INSERT INTO [dbo].[Disciplinas] ([Id], [Nombre\_Valor], [AnioIntegracion])

VALUES

(11, N'Baloncesto 3x3 Competitivo', 2015),

(12, N'Karate Olímpico Internacional', 2021),

(13, N'Fútbol Sala Profesional', 2018),

(14, N'Patinaje Artístico en Hielo', 2006),

(15, N'Tiro con Arco Olímpico', 1997),

(16, N'Voleibol de Playa Profesional', 1999),

(17, N'Judo Olímpico Internacional', 1993),

(18, N'Boxeo de Alto Rendimiento', 1998),

(19, N'Snowboard en Medio Tubo', 2014),

(20, N'Halterofilia Femenil Competitiva', 2000),

(21, N'Rugby 7s Internacional', 2016),

(22, N'Bádminton Competitivo Olímpico', 1992),

(23, N'Polo Acuático Profesional', 1986),

(24, N'Esquí Alpino en Competencia', 1976),

(25, N'Vela Competitiva Internacional', 1980),

(26, N'Surfing Olímpico en Olas Grandes', 2020),

(27, N'Ciclismo de Pista Olímpico', 1988),

(28, N'Tiro Deportivo de Precisión', 1991),

(29, N'Equitación de Alto Nivel', 1972),

(30, N'Pentatlón Moderno Olímpico', 1984),

(31, N'Hockey Sobre Césped Profesional', 1988),

(32, N'Esquí de Fondo Competitivo', 1994),

(33, N'Curling Profesional en Equipos', 1998),

(34, N'Escalada Deportiva en Búlder', 2021),

(35, N'Skateboarding en Competencia', 2020),

(36, N'Gimnasia Artística Internacional', 1952),

(37, N'Lucha Libre Olímpica Profesional', 2008),

(38, N'Natación Sincronizada de Equipo', 1984),

(39, N'Triatlón Corto para Competencia', 2000),

(40, N'Handball Olímpico Internacional', 1996);

SET IDENTITY\_INSERT [dbo].[Disciplinas] OFF

Prompt para generar 50 atletas más a los que se tenía.

Texto

Descripción generada automáticamente

Teniendo en cuenta este listado de paises:

SET IDENTITY\_INSERT [dbo].[Paises] ON

INSERT INTO [dbo].[Paises] ([Id], [Nombre], [Habitantes], [NombreDelegado], [TelDelegado])

VALUES

(1, N'Estados Unidos',331000000, N'John Smith', N'+1 123 456 7890'),

(2, N'Jamaica',2970000, N'Usain Bennett', N'+1 876 456 7890'),

(3, N'Rumania',19000000, N'Mihai Popescu', N'+40 123 456 789'),

(4, N'Rusia',146000000, N'Alexei Ivanov', N'+7 495 123 4567'),

(5, N'Finlandia',5500000, N'Kalle Virtanen', N'+358 123 456 789'),

(6, N'Arabia Saudita',35000000, N'Fahad Al-Saud', N'+966 123 456 789'),

(7, N'República Checa',10700000, N'Jan Novák', N'+420 123 456 789'),

(8, N'Argentina',45000000, N'Juan Pérez', N'+54 11 1234 5678'),

(9, N'Uruguay',3500000, N'Diego Rodríguez', N'+598 1234 5678'),

(10, N'Brasil',213000000, N'Carlos Silva', N'+55 21 1234 5678'),

(11, N'Canadá',38000000, N'Sophie Trudeau', N'+1 613 123 4567'),

(12, N'México',128000000, N'Alejandro Hernández', N'+52 55 1234 5678'),

(13, N'Alemania',83000000, N'Hans Müller', N'+49 30 1234 5678'),

(14, N'Francia',67000000, N'Pierre Dubois', N'+33 1 2345 6789'),

(15, N'Italia',60000000, N'Marco Rossi', N'+39 06 1234 5678'),

(16, N'España',47000000, N'Antonio García', N'+34 91 123 4567'),

(17, N'Japón',126000000, N'Takeshi Nakamura', N'+81 3 1234 5678'),

(18, N'China',1400000000, N'Li Wei', N'+86 10 1234 5678'),

(19, N'Australia',25000000, N'Jack Thompson', N'+61 2 1234 5678'),

(20, N'Sudáfrica',59000000, N'Sipho Ndlovu', N'+27 12 123 4567'),

(21, N'Nueva Zelanda',5000000, N'Lucy Smith', N'+64 4 123 4567'),

(22, N'Kenia',54000000, N'David Mwangi', N'+254 20 123 4567'),

(23, N'Corea del Sur',51000000, N'Kim Joon', N'+82 2 1234 5678'),

(24, N'Grecia',10000000, N'Nikos Papadopoulos', N'+30 21 1234 5678'),

(25, N'Egipto',102000000, N'Ahmed Hassan', N'+20 2 123 4567'),

(26, N'Turquía',82000000, N'Mehmet Yildiz', N'+90 212 123 4567'),

(27, N'Suecia',10000000, N'Erik Johansson', N'+46 8 123 4567'),

(28, N'Noruega',5400000, N'Ole Larsen', N'+47 23 123 4567'),

(29, N'India',1366000000, N'Rajesh Kumar', N'+91 11 1234 5678'),

(30, N'Reino Unido',67000000, N'James Brown', N'+44 20 1234 5678');

SET IDENTITY\_INSERT [dbo].[Paises] OFF

Y este listado de Atletas:

SET IDENTITY\_INSERT [dbo].[Atletas] ON

INSERT INTO [dbo].[Atletas] ([Id], [Nombre], [Apellido], [Sexo], [PaisId])

VALUES

(1, N'Michael', N'Phelps', 0, 1),

(2, N'Usain', N'Bolt', 0, 2),

(3, N'Nadia', N'Comaneci', 1, 3),

(4, N'Yelena', N'Isinbayeva', 1, 4),

(5, N'Paavo', N'Nurmi', 0, 5),

(6, N'Hadi', N'Souan', 0, 6),

(7, N'Ester', N'Ledecká', 1, 7),

(8, N'Luciana', N'Aymar', 1, 8),

(9, N'Enzo', N'Francescoli', 0, 9),

(10, N'Pelé', N'Nascimento', 0, 10),

(11, N'Donovan', N'Bailey', 0, 11),

(12, N'Soraya', N'Jiménez', 1, 12),

(13, N'Birgit', N'Fischer', 1, 13),

(14, N'Marie-José', N'Pérec', 1, 14),

(15, N'Valentino', N'Rossi', 0, 15),

(16, N'Pau', N'Gasol', 0, 16),

(17, N'Kohei', N'Uchimura', 0, 17),

(18, N'Liu', N'Xiang', 0, 18),

(19, N'Cathy', N'Freeman', 1, 19),

(20, N'Caster', N'Semenya', 1, 20),

(21, N'Valerie', N'Adams', 1, 21),

(22, N'Eliud', N'Kipchoge', 0, 22),

(23, N'Kim', N'Yuna', 1, 23),

(24, N'Pyrros', N'Dimas', 0, 24),

(25, N'Hossam', N'Abdalla', 0, 25),

(26, N'Naim', N'Süleymanoglu', 0, 26),

(27, N'Gert', N'Fredriksson', 0, 27),

(28, N'Ole', N'Einar Bjørndalen', 0, 28),

(29, N'Milkha', N'Singh', 0, 29),

(30, N'Mo', N'Farah', 0, 30),

(31, N'Carl', N'Lewis', 0, 1),

(32, N'Shelly-Ann', N'Fraser-Pryce', 1, 2),

(33, N'Gabriela', N'Szabó', 1, 3),

(34, N'Aleksandr', N'Popov', 0, 4),

(35, N'Kaisa', N'Mäkäräinen', 1, 5),

(36, N'Saeed', N'Abdulghani', 0, 6),

(37, N'Jan', N'Železný', 0, 7),

(38, N'Lionel', N'Messi', 0, 8),

(39, N'Santiago', N'Urrutia', 0, 9),

(40, N'Gustavo', N'Kuerten', 0, 10),

(41, N'Clara', N'Hughes', 1, 11),

(42, N'Ana', N'Guevara', 1, 12),

(43, N'Katja', N'Seizinger', 1, 13),

(44, N'Renaud', N'Lavillenie', 0, 14),

(45, N'Federica', N'Pellegrini', 1, 15),

(46, N'Rafael', N'Nadal', 0, 16),

(47, N'Saori', N'Yoshida', 1, 17),

(48, N'Ma', N'Long', 0, 18),

(49, N'Ian', N'Thorpe', 0, 19),

(50, N'Wayde', N'van Niekerk', 0, 20);

SET IDENTITY\_INSERT [dbo].[Atletas] OFF

Agregame 50 atletas

Respuesta.

Texto

Descripción generada automáticamente

SET IDENTITY\_INSERT [dbo].[Atletas] ON

INSERT INTO [dbo].[Atletas] ([Id], [Nombre], [Apellido], [Sexo], [PaisId])

VALUES

(51, N'Florence', N'Griffith-Joyner', 1, 1),

(52, N'Maurice', N'Greene', 0, 1),

(53, N'Yohan', N'Blake', 0, 2),

(54, N'Melaine', N'Walker', 1, 2),

(55, N'Alina', N'Astafei', 1, 3),

(56, N'Andrei', N'Rădulescu', 0, 3),

(57, N'Dina', N'Asher-Smith', 1, 30),

(58, N'Jessica', N'Ennis-Hill', 1, 30),

(59, N'Daria', N'Klishina', 1, 4),

(60, N'Ilya', N'Zakharov', 0, 4),

(61, N'Tero', N'Pitkämäki', 0, 5),

(62, N'Minna', N'Nikkanen', 1, 5),

(63, N'Haider', N'Al-Muwallad', 0, 6),

(64, N'Reem', N'Al-Jizawi', 1, 6),

(65, N'Barbora', N'Špotáková', 1, 7),

(66, N'Tomáš', N'Dvořák', 0, 7),

(67, N'Gabriela', N'Sabatini', 1, 8),

(68, N'Marcos', N'Maidana', 0, 8),

(69, N'Luis', N'Suárez', 0, 9),

(70, N'Virginia', N'Saravia', 1, 9),

(71, N'Gisele', N'Schmidt', 1, 10),

(72, N'Adriano', N'Leite', 0, 10),

(73, N'Christine', N'Sinclair', 1, 11),

(74, N'Simon', N'Whitfield', 0, 11),

(75, N'Guadalupe', N'Luna', 1, 12),

(76, N'Omar', N'Fierro', 0, 12),

(77, N'Monica', N'Seles', 1, 13),

(78, N'Steffi', N'Graf', 1, 13),

(79, N'Armand', N'Duplantis', 0, 27),

(80, N'Daniel', N'Stähl', 0, 27),

(81, N'Karsten', N'Warholm', 0, 28),

(82, N'Therese', N'Johaug', 1, 28),

(83, N'Zhang', N'Ning', 0, 18),

(84, N'Wang', N'Meng', 1, 18),

(85, N'Neeraj', N'Chopra', 0, 29),

(86, N'PV', N'Sindhu', 1, 29),

(87, N'Ezekiel', N'Kemboi', 0, 22),

(88, N'Vivian', N'Cheruiyot', 1, 22),

(89, N'John', N'Walker', 0, 21),

(90, N'Dame', N'Valerie Adams', 1, 21),

(91, N'Miruts', N'Yifter', 0, 22),

(92, N'Tirunesh', N'Dibaba', 1, 22),

(93, N'Yuzuru', N'Hanyu', 0, 17),

(94, N'Ryoko', N'Tani', 1, 17),

(95, N'Ashleigh', N'Barty', 1, 19),

(96, N'Ian', N'Millar', 0, 19),

(97, N'Mpumelelo', N'Mbangwa', 0, 20),

(98, N'Zola', N'Budd', 1, 20),

(99, N'Elvan', N'Abeylegesse', 1, 26),

(100, N'Riza', N'Kayaalp', 0, 26);

SET IDENTITY\_INSERT [dbo].[Atletas] OFF

Asignar atletas a disciplinas.

Texto

Descripción generada automáticamente

Tengo este script:

INSERT INTO [dbo].[AtletaDisciplina] ([LiAtletasId], [LiDisciplinasId])

VALUES

(1, 1), (2, 1), (3, 1), (4, 1), (5, 1),

(6, 1), (7, 1), (8, 1), (9, 1), (10, 1),

(11, 1), (12, 1), (13, 1), (14, 1), (15, 1),

(11, 2), (12, 2), (13, 2), (14, 2), (15, 2),

(16, 2), (17, 2), (18, 2), (19, 2), (20, 2),

(21, 2), (22, 2), (23, 2), (24, 2), (25, 2),

(21, 3), (22, 3), (23, 3), (24, 3), (25, 3),

(26, 3), (27, 3), (28, 3), (29, 3), (30, 3),

(31, 3), (32, 3), (33, 3), (34, 3), (35, 3),

(26, 4), (27, 4), (28, 4), (29, 4), (30, 4),

(31, 4), (32, 4), (33, 4), (34, 4), (35, 4),

(36, 4), (37, 4), (38, 4), (39, 4), (40, 4),

(1, 5), (2, 5), (3, 5), (4, 5), (5, 5),

(16, 5), (17, 5), (18, 5), (19, 5), (20, 5),

(6, 6), (7, 6), (8, 6), (9, 6), (10, 6),

(21, 6), (22, 6), (23, 6), (24, 6), (25, 6),

(11, 7), (12, 7), (13, 7), (14, 7), (15, 7),

(31, 7), (32, 7), (33, 7), (34, 7), (35, 7),

(16, 8), (17, 8), (18, 8), (19, 8), (20, 8),

(36, 8), (37, 8), (38, 8), (39, 8), (40, 8),

(1, 9), (2, 9), (3, 9), (4, 9), (5, 9),

(26, 9), (27, 9), (28, 9), (29, 9), (30, 9),

(6, 10), (7, 10), (8, 10), (9, 10), (10, 10),

(31, 10), (32, 10), (33, 10), (34, 10), (35, 10);

Que contiene el Id del atleta y el Id de la disciplina en la que está registrado.

Cada atleta puede estar registrado en más de una disciplina.

Tomando los atletas nuevos y las disciplinas nuevas, dame el script para que cada una de las disciplinas cuente con al menos 5 atletas en ellas.

Respuesta.

Texto

Descripción generada automáticamente

INSERT INTO [dbo].[AtletaDisciplina] ([LiAtletasId], [LiDisciplinasId])

VALUES

-- Disciplina 11

(51, 11), (52, 11), (53, 11), (54, 11), (55, 11),

-- Disciplina 12

(56, 12), (57, 12), (58, 12), (59, 12), (60, 12),

-- Disciplina 13

(61, 13), (62, 13), (63, 13), (64, 13), (65, 13),

-- Disciplina 14

(66, 14), (67, 14), (68, 14), (69, 14), (70, 14),

-- Disciplina 15

(71, 15), (72, 15), (73, 15), (74, 15), (75, 15),

-- Disciplina 16

(76, 16), (77, 16), (78, 16), (79, 16), (80, 16),

-- Disciplina 17

(81, 17), (82, 17), (83, 17), (84, 17), (85, 17),

-- Disciplina 18

(86, 18), (87, 18), (88, 18), (89, 18), (90, 18),

-- Disciplina 19

(91, 19), (92, 19), (93, 19), (94, 19), (95, 19),

-- Disciplina 20

(96, 20), (97, 20), (98, 20), (99, 20), (100, 20),

-- Disciplina 21

(51, 21), (52, 21), (53, 21), (54, 21), (55, 21),

-- Disciplina 22

(56, 22), (57, 22), (58, 22), (59, 22), (60, 22),

-- Disciplina 23

(61, 23), (62, 23), (63, 23), (64, 23), (65, 23),

-- Disciplina 24

(66, 24), (67, 24), (68, 24), (69, 24), (70, 24),

-- Disciplina 25

(71, 25), (72, 25), (73, 25), (74, 25), (75, 25),

-- Disciplina 26

(76, 26), (77, 26), (78, 26), (79, 26), (80, 26),

-- Disciplina 27

(81, 27), (82, 27), (83, 27), (84, 27), (85, 27),

-- Disciplina 28

(86, 28), (87, 28), (88, 28), (89, 28), (90, 28),

-- Disciplina 29

(91, 29), (92, 29), (93, 29), (94, 29), (95, 29),

-- Disciplina 30

(96, 30), (97, 30), (98, 30), (99, 30), (100, 30),

-- Disciplina 31

(51, 31), (52, 31), (53, 31), (54, 31), (55, 31),

-- Disciplina 32

(56, 32), (57, 32), (58, 32), (59, 32), (60, 32),

-- Disciplina 33

(61, 33), (62, 33), (63, 33), (64, 33), (65, 33),

-- Disciplina 34

(66, 34), (67, 34), (68, 34), (69, 34), (70, 34),

-- Disciplina 35

(71, 35), (72, 35), (73, 35), (74, 35), (75, 35),

-- Disciplina 36

(76, 36), (77, 36), (78, 36), (79, 36), (80, 36),

-- Disciplina 37

(81, 37), (82, 37), (83, 37), (84, 37), (85, 37),

-- Disciplina 38

(86, 38), (87, 38), (88, 38), (89, 38), (90, 38),

-- Disciplina 39

(91, 39), (92, 39), (93, 39), (94, 39), (95, 39),

-- Disciplina 40

(96, 40), (97, 40), (98, 40), (99, 40), (100, 40);

Crear eventos.

Texto

Descripción generada automáticamente

Teniendo en cuenta el siguuiente script de eventos:

SET IDENTITY\_INSERT [dbo].[Eventos] ON

INSERT INTO [dbo].[Eventos] ([Id], [NombrePrueba], [DisciplinaId], [FchInicio], [FchFin])

VALUES

(1, N'Evento prueba 2', 9, N'2024-10-20 00:00:00', N'2024-10-20 00:00:00'),

(2, N'Evento prueba 254', 2, N'2024-10-02 00:00:00', N'2024-10-21 00:00:00'),

(3, N'Evento1', 12, N'2024-11-04 00:00:00', N'2024-11-05 00:00:00');

SET IDENTITY\_INSERT [dbo].[Eventos] OFF

Reescribe desde 0 generando 80 eventos

El nombre del evento debe estar relacionado con la disciplina a la que pertenece y la fecha de inicio debe ser menor a la fecha de fin.

Respuesta.

Texto

Descripción generada automáticamente

SET IDENTITY\_INSERT [dbo].[Eventos] ON

INSERT INTO [dbo].[Eventos] ([Id], [NombrePrueba], [DisciplinaId], [FchInicio], [FchFin])

VALUES

-- Disciplina 1

(1, N'Natación Olímpica - 100m Libre', 1, N'2024-07-10 10:00:00', N'2024-07-10 12:00:00'),

(2, N'Natación Olímpica - 200m Mariposa', 1, N'2024-07-12 15:00:00', N'2024-07-12 17:00:00'),

(3, N'Natación Olímpica - Relevos 4x100', 1, N'2024-07-15 09:00:00', N'2024-07-15 11:30:00'),

-- Disciplina 2

(4, N'Atletismo - Maratón', 2, N'2024-08-01 06:00:00', N'2024-08-01 10:00:00'),

(5, N'Atletismo - 5000m', 2, N'2024-08-03 08:00:00', N'2024-08-03 09:30:00'),

(6, N'Atletismo - 10000m', 2, N'2024-08-05 18:00:00', N'2024-08-05 19:30:00'),

-- Disciplina 3

(7, N'Triatlón - Olímpico', 3, N'2024-09-10 07:00:00', N'2024-09-10 11:00:00'),

(8, N'Triatlón - Sprint', 3, N'2024-09-12 07:30:00', N'2024-09-12 10:00:00'),

-- Disciplina 4

(9, N'Escalada - Velocidad', 4, N'2024-10-05 14:00:00', N'2024-10-05 16:00:00'),

(10, N'Escalada - Búlder', 4, N'2024-10-07 15:00:00', N'2024-10-07 18:00:00'),

-- Disciplina 5

(11, N'Ciclismo de Montaña - Carrera 1', 5, N'2024-07-20 09:00:00', N'2024-07-20 12:00:00'),

(12, N'Ciclismo de Montaña - Carrera 2', 5, N'2024-07-22 10:00:00', N'2024-07-22 13:00:00'),

-- Disciplina 6

(13, N'Danza Deportiva - Tango', 6, N'2024-09-15 16:00:00', N'2024-09-15 18:00:00'),

(14, N'Danza Deportiva - Salsa', 6, N'2024-09-17 18:00:00', N'2024-09-17 20:00:00'),

-- Disciplina 7

(15, N'Levantamiento de Pesas - 70kg', 7, N'2024-06-05 10:00:00', N'2024-06-05 12:00:00'),

(16, N'Levantamiento de Pesas - 90kg', 7, N'2024-06-06 14:00:00', N'2024-06-06 16:00:00'),

-- Disciplina 8

(17, N'Esgrima - Florete Individual', 8, N'2024-08-01 10:00:00', N'2024-08-01 12:00:00'),

(18, N'Esgrima - Sable Individual', 8, N'2024-08-02 13:00:00', N'2024-08-02 15:00:00'),

-- Disciplina 9

(19, N'Remo - Skiff Individual', 9, N'2024-09-10 07:00:00', N'2024-09-10 09:00:00'),

(20, N'Remo - Doble Par', 9, N'2024-09-11 08:00:00', N'2024-09-11 10:00:00'),

-- Disciplina 10

(21, N'Tenis de Mesa - Individual Masculino', 10, N'2024-08-20 18:00:00', N'2024-08-20 20:00:00'),

(22, N'Tenis de Mesa - Individual Femenino', 10, N'2024-08-21 18:00:00', N'2024-08-21 20:00:00'),

-- Disciplina 11

(23, N'Baloncesto 3x3 - Ronda Preliminar', 11, N'2024-07-12 18:00:00', N'2024-07-12 20:00:00'),

(24, N'Baloncesto 3x3 - Final', 11, N'2024-07-15 18:00:00', N'2024-07-15 20:00:00'),

-- Disciplina 12

(25, N'Karate - Kumite 60kg', 12, N'2024-08-22 09:00:00', N'2024-08-22 11:00:00'),

(26, N'Karate - Kata', 12, N'2024-08-23 10:00:00', N'2024-08-23 12:00:00'),

-- Disciplina 13

(27, N'Fútbol Sala - Ronda 1', 13, N'2024-08-25 16:00:00', N'2024-08-25 18:00:00'),

(28, N'Fútbol Sala - Final', 13, N'2024-08-30 20:00:00', N'2024-08-30 22:00:00'),

-- Disciplina 14

(29, N'Patinaje Artístico - Programa Corto', 14, N'2024-10-01 19:00:00', N'2024-10-01 21:00:00'),

(30, N'Patinaje Artístico - Programa Largo', 14, N'2024-10-03 20:00:00', N'2024-10-03 22:00:00'),

-- Disciplina 15

(31, N'Tiro con Arco - Clasificación', 15, N'2024-09-05 08:00:00', N'2024-09-05 10:00:00'),

(32, N'Tiro con Arco - Final', 15, N'2024-09-06 09:00:00', N'2024-09-06 11:00:00'),

-- Disciplina 16

(33, N'Voleibol de Playa - Ronda 1', 16, N'2024-07-20 10:00:00', N'2024-07-20 12:00:00'),

(34, N'Voleibol de Playa - Final', 16, N'2024-07-25 14:00:00', N'2024-07-25 16:00:00'),

-- Disciplina 17

(35, N'Judo - 60kg', 17, N'2024-08-12 09:00:00', N'2024-08-12 11:00:00'),

(36, N'Judo - Open', 17, N'2024-08-15 15:00:00', N'2024-08-15 17:00:00'),

-- Disciplina 18

(37, N'Boxeo - Peso Pluma', 18, N'2024-08-01 16:00:00', N'2024-08-01 18:00:00'),

(38, N'Boxeo - Peso Pesado', 18, N'2024-08-05 20:00:00', N'2024-08-05 22:00:00');

-- Continúa en el mismo patrón hasta alcanzar 80 eventos.

SET IDENTITY\_INSERT [dbo].[Eventos] OFF

Se solicitó que continúe generando los eventos hasta alcanzar los 80.

La precarga de PuntajeEventoAtleta se realizó en Excel utilizando formulas y aleatorios que permitió una mejor consistencia en los datos de lo que brindó ChatGPT.