Guía de Prueba del Programa Principal - CentroEventos

Introducción

Este documento explica como probar las funcionalidades implementadas en el archivo Program.cs del sistema CentroEventos. A través de un menú interactivo por consola, se pueden realizar operaciones cómo agregar, modificar, eliminar y listar personas, eventos deportivos y reservas.

Ejemplo 1: Agregar una persona

Para probar la opción 1 (Agregar Persona), seguir estos pasos:

```
== Agregar persona ==

Nombre: Juan

Apellido: Pérez

DNI: 12345678

Email: juan.perez@example.com

Teléfono: 2211234567

Persona agregada con ID: 1
```

Este flujo de prueba agrega una nueva persona al sistema y muestra su ID asignado.

Program.cs:

RepositorioPersona.cs:

PersonaAltaUseCase.cs:

```
public void Ejecutar(Persona persona, int idUsuario){
    if (!_servicioAutorizacion.PoseeElPermiso(idUsuario, Permiso.UsuarioAlta))
        throw new UnauthorizedAccessException("El usuario no tiene permiso para agregar una persona.");
    _validador.Validar(persona);
    _repo.Agregar(persona);
}
```

ValidadorPersona.cs:

```
public void Validar(Persona persona) {
    if (string.IsNullOrWhiteSpace(persona.Nombre))
        throw new ValidacionException("El Nombre debe completarse");
    if (string.IsNullOrWhiteSpace(persona.Apellido))
        throw new ValidacionException("El Apellido debe completarse");
    if (string.IsNullOrWhiteSpace(persona.DNI))
        throw new ValidacionException("El DNI debe completarse");
    if (string.IsNullOrWhiteSpace(persona.Email))
        throw new ValidacionException("El Email debe completarse");

    var Email = _repositorioPersona.ObtenerPorEmail(persona.Email);
    var DNI = _repositorioPersona.ObtenerPorDni(persona.DNI);

    if (Email != null && Email.Id != persona.Id)
        throw new DuplicadoException("Ya existe una persona con ese Email");
    if (DNI != null && DNI.Id != persona.Id)
        throw new DuplicadoException("Ya existe una persona con ese DNI");
}
```

Ejemplo 2: Agregar un Evento Deportivo

Seleccionar la opción 2 e ingresar los siguientes datos:

```
Nombre del Evento: Torneo Futbol

Descripción del Evento: Torneo universitario

Fecha de inicio del Evento: 2025-06-01 19:00

Duración en horas del Evento: 2

ID del responsable del Evento: 1

Cupo maximo del Evento: 50

Lugar del evento: Estadio Central

Fecha válida: 01/06/2025

Duración correcta

Id valido

Cupo valido
```

Esto crea un evento deportivo si el ID responsable ya existe.

Program.cs:

```
var evento = new EventoDeportivo
{
    Nombre = nombreEvento ?? "",
    Descripcion = desc ?? "",
    FechaHoraInicio = fecha,
    Lugar = donde ?? "",
    DuracionHoras = dura,
    CupoMaximo = cupo,
    ResponsableId = id
};
var crearEvento = new EventoDeportivoAltaUseCase(repoevento, repo,servicioAutorizacion);
crearEvento.Ejecutar(evento,idAutorizado);
```

RepositorioEventoDeportivo.cs:

```
public void Agregar(EventoDeportivo evento) {
    evento.Id = ObtenerNuevoId();
    var linea = $"{evento.Id},{evento.Nombre},{evento.FechaHoraInicio:yyyy-MM-dd HH:mm}
    File.AppendAllLines(archivo, [linea]);
}
```

EventoDeportivoAltaUseCase.cs:

```
public void Ejecutar(EventoDeportivo evento, int idUsuario){
    if (!_servicioAutorizacion.PoseeElPermiso(idUsuario, Permiso.EventoAlta))
            throw new UnauthorizedAccessException("El usuario no tiene permiso para crear eventos.");
    _Validador.Validar(evento);
    _repoEvento.Agregar(evento);
}
```

ValidadorEventoDeportivo.cs:

```
public void Validar(EventoDeportivo eventoDeportivo) {
    var Persona = this.repoPersona.ObtenerPorId(eventoDeportivo.ResponsableId);
    if (Persona == null)
        throw new EntidadNotFoundException("Responsable no encontrado");
    if (eventoDeportivo.CupoMaximo < 0)
        throw new ValidacionException("Cupo Maximo debe ser mayor a 0");
    if (eventoDeportivo.DuracionHoras < 0)
        throw new ValidacionException("El evento debe durar mas de 0 hs");
    if (eventoDeportivo.FechaHoraInicio < DateTime.Now)
        throw new ValidacionException("La Fecha del evento debe ser posterior a la actual");
    if (string.IsNullOrWhiteSpace(eventoDeportivo.Nombre))
        throw new ValidacionException("El evento debe tener un nombre");
    if (string.IsNullOrWhiteSpace(eventoDeportivo.Descripcion))
        throw new ValidacionException("El evento debe tener una descripcion");
}
```

Ejemplo 3: Agregar una Reserva

Seleccionar la opción 3 y completar los datos:

```
Insertar Id de la persona q quiere hacer la reserva: 1

Insertar Id del evento a reservar: 1
```

Este ejemplo realiza una reserva para la persona y eventos creados anteriormente.

Program.cs:

```
var AgregarReserva = new ReservaAltaUseCase(repoReserva, repoevento, repo, servicioAutorizacion);
var reserva = new Reserva()
{
    PersonaId = PersonaReservaId,
    EventoDeportivoId = EventoReservaId
};
AgregarReserva.Ejecutar(reserva, idAutorizado);
```

RepositorioReserva.cs:

```
public void Agregar(Reserva reserva) {
    reserva.Id = ObtenerNuevoId();
    var linea = $"{reserva.Id},{reserva.Id},{reserva.PersonaId},{reserva.FechaAltaReserva:yyyy-MM-dd},{reserva}";
    File.AppendAllLines(archivo, new[] { linea });
}
```

ReservaAltaUseCase.cs:

```
public void Ejecutar(Reserva datosReserva, int idUsuario) {
    if (!_servicioAutorizacion.PoseeElPermiso(idUsuario,Permiso.ReservaAlta))
        throw new FalloAutorizacionException();
    _validador.Validar(datosReserva);
    datosReserva.FechaAltaReserva = DateTime.Now;
    datosReserva.EstadoAsistencia = EstadoAsistencia.Pendiente;
    _repoReserva.Agregar(datosReserva);
}
```

Validador Reserva.cs:

```
public void Validar(Reserva reserva) {
        var Persona = _repoPersona.ObtenerPorId(reserva.PersonaId);
        var EventoDeportivo = _repoEvento.ObtenerPorId(reserva.EventoDeportivoId);
        var cantReservas = _repoReservas.ListarPorEvento(reserva.EventoDeportivoId).Count;
        if (Persona == null)
            throw new EntidadNotFoundException("Persona no encontrada");
        if (EventoDeportivo == null)
            throw new EntidadNotFoundException("Evento Deportivo no encontrado");
        if (cantReservas >= EventoDeportivo.CupoMaximo)
            throw new CupoExcedidoException();
        if (_repoReservas.ExisteReserva(reserva.PersonaId, reserva.EventoDeportivoId))
            throw new DuplicadoException("La Persona ya tiene una reserva para este evento");
}
```

Ejemplo 4: Listar Personas

Seleccionar la opción 10 para ver el listado de personas registradas:

```
Listado de personas: 1: Juan Pérez - DNI: 12345678 - Email:
juan.perez@example.com - Telefono: 2211234567
```

Program.cs:

```
Console.WriteLine("Listado de personas:");
var Lista = new PersonaListarUseCase(repo);
foreach (var p in Lista.Ejecutar())
{
    Console.WriteLine($"{p.Id}: {p.Nombre} {p.Apellido} - DNI: {p.DNI} - Email: {p.Email} - Telefono: {p.Telefono}");
}
```

RepositorioPersona.cs:

```
public List<Persona> Listar() {
    var personas = new List<Persona>();
    if (!File.Exists(archivo)) return personas;
    foreach (var linea in File.ReadAllLines(archivo)) {
        var partes = linea.Split(',');
        if (partes.Length >= 5) {
            personas.Add(new Persona
            {
                // trim: eliminar espacios en blanco al inicio y al final de una cadena de texto
                Id = int.Parse(partes[0]),
                Nombre = partes[1].Trim(),
                Apellido = partes[2].Trim(),
                DNI = partes[3].Trim(),
                Email = partes[4].Trim(),
                Telefono= partes[5].Trim()
            });
        }
    }
    return personas;
}
```

PersonaListarUseCase.cs:

```
public List<Persona> Ejecutar(){
    return repositorioPersona.Listar();
}
```

Ejemplo 5: Listar Eventos con cupo disponible

Seleccionar la opción 14 para ver todos los eventos disponibles.

```
Listado de Eventos con cupo disponible:

Id: 1 - Evento: Torneo Futbol: Torneo universitario - Fecha de inicio:
01/06/2025

19:00:00 - Lugar: Estadio Central
```

Program.cs:

```
//ListarEventoConCupoDisponible
Console.WriteLine("Listado de Eventos con cupo disponible: ");
var ListaEventosDisponibles = new ListarEventosConCupoDisponibleUseCase(repoevento, repoReserva);
foreach (var e in ListaEventosDisponibles.Ejecutar())
    Console.WriteLine($"Id: {e.Id} - Evento: {e.Nombre}: {e.Descripcion} - Fecha de inicio: {e.FechaHoraInicio} - Lugar: {e.Lugar}");
break;
```

RepositorioEventoDeportivo.cs:

```
public List<EventoDeportivo> ListarTodos() {
    if (!File.Exists(archivo)) return new List<EventoDeportivo>();
    return File.ReadAllLines(archivo)
        .Select(l => l.Split(','))
        .Where(p => p.Length >= 4)
        .Select(p => new EventoDeportivo
        {
            Id = int.Parse(p[0]),
            Nombre = p[1].Trim(),
            FechaHoraInicio = DateTime.Parse(p[2]),
            Lugar = p[3].Trim(),
            CupoMaximo = int.Parse(p[4]),
            DuracionHoras = int.Parse(p[5]),
            ResponsableId = int.Parse(p[6]),
            Descripcion=p[7]

        }).ToList();
}
```

ListarEventoConCupoDisponibleUseCase.cs:

```
public List<EventoDeportivo> Ejecutar() {
    var eventosFuturos = _repoEvento.ListarTodos().Where(e => e.FechaHoraInicio > DateTime.Now).ToList();
    var resultado = new List<EventoDeportivo>();
    foreach(var evento in eventosFuturos) {
        var reservas = _repoReserva.ListarPorEvento(evento.Id);
        if (reservas.Count() < evento.CupoMaximo)
            resultado.Add(evento);
    }
    return resultado;
}
```

Ejemplo 6: Listar Eventos

Seleccionar la opción 11 para listar todos los eventos

```
Listado de eventos

Id: 1 - Evento: Torneo Futbol: Torneo universitario - Fecha de inicio:
01/06/2025

19:00:00 - Lugar: Estadio Central

Id: 3 - Evento: F5: Partido de fútbol 5 - Fecha de inicio: 12/9/2025 20:00:00
- Lugar: Las palmas
```

Program.cs:

```
Console.WriteLine("Listado de eventos");
var ListaEventos = new EventoDeportivoListarUseCase(repoevento);
foreach (var e in ListaEventos.Ejecutar())
{
    Console.WriteLine($"Id: {e.Id} - Evento: {e.Nombre}: {e.Descripcion} - Fecha de inicio: {e.FechaHoraInicio} - Lugar: {e.Lugar}");
}
```

RepositorioEventoDeportivo.cs:

```
public List<EventoDeportivo> ListarTodos() {
    if (!File.Exists(archivo)) return new List<EventoDeportivo>();
    return File.ReadAllLines(archivo)
        .Select(l => l.Split(','))
        .Where(p => p.Length >= 4)
        .Select(p => new EventoDeportivo
        {
            Id = int.Parse(p[0]),
            Nombre = p[1].Trim(),
            FechaHoraInicio = DateTime.Parse(p[2]),
            Lugar = p[3].Trim(),
            CupoMaximo = int.Parse(p[4]),
            DuracionHoras = int.Parse(p[5]),
            ResponsableId = int.Parse(p[6]),
            Descripcion=p[7]

        }).ToList();
}
```

EventoDeportivoListarUseCase.cs:

```
public List<EventoDeportivo> Ejecutar(){
    return _repo.ListarTodos();
}
```

Ejemplo 7: Listar Reservas

Seleccionar la opción 12 para listar todas las reservas:

```
Listado de Reservas:

Id de la reserva: 3: Id de la persona que reserva: 6 - Id del evento:
3 - Fecha: 21/5/2025 09:44:00
```

Program.cs:

```
Console.WriteLine("Listado de Reservas:");
var ListaReserva = new ReservaListarUseCase(repoReserva);
foreach (var p in ListaReserva.Ejecutar()) {
    Console.WriteLine($"Id de la reserva: {p.Id}: Id de la persona que reserva: {p.PersonaId} - Id del evento: {p.EventoDeportivoId}
}
```

RepositorioReserva.cs:

```
public List<Reserva> Listar() {
    if (!File.Exists(archivo)) return new List<Reserva>();
    return File.ReadAllLines(archivo)
        .Select(l => l.Split(','))
        .Where(p => p.Length >= 4)
        .Select(p => new Reserva
        {
            Id = int.Parse(p[0]),
            EventoDeportivoId = int.Parse(p[1]),
            PersonaId = int.Parse(p[2]),
            FechaAltaReserva = DateTime.Parse(p[3])

        }).ToList();
```

ReservaListarUseCase.cs:

```
public List<Reserva> Ejecutar(){
    return _repo.Listar();
}
```

Ejemplo 8: Listar Asistencia a Evento:

seleccione la opción 13 para listar los asistidos a un evento:

```
Insertar Id del evento del que se desea listar la asistencia

3

Listado De asistencia Al evento con ID 3

Listado de personas: 1: Juan Pérez - DNI: 12345678 - Email:
juan.perez@example.com - Telefono: 2211234567
```

Program.cs:

```
Console.WriteLine("Insertar Id del evento del que se desea listar la asistencia");
string? idEventoAlistar = Console.ReadLine();
if (int.TryParse(idEventoAlistar, out int idevent)) {
    Console.WriteLine($"Listado De asistencia Al evento con ID {idevent}");
    var ListarAsistencia = new ListarAsistenciaAEventoUseCase(repoevento, repo, repoReserva);
    foreach (var p in ListarAsistencia.Ejecutar(idevent))
        Console.WriteLine($"{p.Id}: {p.Nombre} {p.Apellido} - DNI: {p.DNI} - Email: {p.Email} - Telefono: {p.Telefono}");
}
```

RepositorioReserva.cs:

```
public List<Reserva> ListarPorEvento(int eventoId) {
    return Listar().Where(r => r.EventoDeportivoId == eventoId).ToList();
}
```

ListarAsistenciaAEventoUseCase.cs:

```
public List<Persona> Ejecutar(int Id){
    var evento = _repoEventoDeportivo.ObtenerPorId(Id)??throw new EntidadNotFoundException("El evento no existe");

    if(evento.FechaHoraInicio>DateTime.Now) throw new OperacionInvalidaException("El evento no ha ocurrido");

    var ReservasAsistieron= _repoReserva.ListarPorEvento(Id).Where(r=> r.EstadoAsistencia==EstadoAsistencia.Presente).ToList();
    var Personas =new  List<Persona>();
    foreach(var r in ReservasAsistieron){
        var persona = _repoPersona.ObtenerPorId(r.Id);
        if(persona!= null ){
            Personas.Add(persona);
        }
    }
    return Personas;
}
```

Ejemplo 9: Modificar Persona

seleccionar la opción 7 para modificar el email, en este caso, de una persona:

```
 Ingrese el ID de la persona a buscar:

6

Encontrada: Matias Buscemi

Ingrese nuevo email para esta persona:

matibusss@gmail.com

Email modificado para persona con ID 6
```

Program.cs:

```
Console.Write("\nIngrese el ID de la persona a buscar: ");
string? inputBuscar = Console.ReadLine();
if (int.TryParse(inputBuscar, out int idBuscar)) {
    var encontrada = repo.ObtenerPorId(idBuscar);
    if (encontrada is not null) {
        Console.WriteLine($"Encontrada: {encontrada.Nombre} {encontrada.Apellido}");
        var modificarPersona = new PersonaModificarUseCase(repo,servicioAutorizacion);
        // Modificar email
        Console.Write("Ingrese nuevo email para esta persona: ");
        string? nuevoEmail = Console.ReadLine();
        encontrada.Email = nuevoEmail ?? encontrada.Email;
        modificarPersona.Ejecutar(encontrada,idAutorizado);
        Console.WriteLine($"Email modificado para persona con ID {encontrada.Id}");
    }
    else
        Console.WriteLine($"No se encontró una persona con ID {idBuscar}.");
}
else
    Console.WriteLine("ID inválido.");
```

RepositorioPersona.cs:

```
public void Modificar(Persona persona) {
    var personas = Listar();
    var index = personas.FindIndex(p => p.Id == persona.Id);
    if (index != -1) {
        personas[index] = persona;
        GuardarTodas(personas);
    }
}
```

```
private void GuardarTodas(List<Persona> personas) {
    var lineas = personas.Select(p => $"{p.Id},{p.Nombre},{p.Apellido},{p.DNI},{p.Email},{p.Telefono}");
    File.WriteAllLines(archivo, lineas);
}
```

PersonaModificarUseCase.cs:

```
public void Ejecutar(Persona persona,int idUsuario){
    if (!_servicioAutorizacion.PoseeElPermiso(idUsuario, Permiso.UsuarioModificacion))
        throw new UnauthorizedAccessException("El usuario no tiene permiso para modificar personas.");
    var add = repositorioPersona.ObtenerPorId(persona.Id)?? throw new EntidadNotFoundException("Persona no encontrada");
    _validador.Validar(persona);
    repositorioPersona.Modificar(persona);
}
```

ValidadorPersona.cs:

```
public void Validar(Persona persona) {
    if (string.IsNullOrWhiteSpace(persona.Nombre))
        throw new ValidacionException("El Nombre debe completarse");
    if (string.IsNullOrWhiteSpace(persona.Apellido))
        throw new ValidacionException("El Apellido debe completarse");
    if (string.IsNullOrWhiteSpace(persona.DNI))
        throw new ValidacionException("El DNI debe completarse");
    if (string.IsNullOrWhiteSpace(persona.Email))
        throw new ValidacionException("El Email debe completarse");

    var Email = _repositorioPersona.ObtenerPorEmail(persona.Email);
    var DNI = _repositorioPersona.ObtenerPorDni(persona.DNI);

    if (Email != null && Email.Id != persona.Id)
        throw new DuplicadoException("Ya existe una persona con ese Email");
    if (DNI != null && DNI.Id != persona.Id)
        throw new DuplicadoException("Ya existe una persona con ese DNI");
}
```

Ejemplo 10: Modificar Evento

seleccionar la opción 8 para modificar, en este caso la fecha de un evento.

```
Ingresar id del evento a modificar:

3

Evento Encontrado: F5

Insertar nueva fecha para el evento:

03/09/2025 6:00

Fecha válida

Evento con id 3 modificado con éxito
```

Program.cs:

```
Console.WriteLine("Ingresar id del evento a modificar: ");
string? idEventoModificar = Console.ReadLine();
if (int.TryParse(idEventoModificar, out int idBuscarEvento)) {
    var eventoEncontrado = repoevento.ObtenerPorId(idBuscarEvento);
    if (eventoEncontrado is not null) {
        Console.WriteLine($"Evento Encontrado: {eventoEncontrado.Nombre}");
        var Modificar = new EventoDeportivoModificarUseCase(repoevento, repo,servicioAutorizacion);
        Console.WriteLine("Insertar nueva fecha para el evento: ");
        string? fechanueva = Console.ReadLine();
        if (DateTime.TryParse(fechanueva, out DateTime fechaEvento))
            Console.WriteLine("Fecha valida");
        else
            Console.WriteLine("Fecha Invalida");
        eventoEncontrado.FechaHoraInicio = fechaEvento;
        Modificar.Ejecutar(eventoEncontrado,idAutorizado);
        Console.WriteLine($"Evento con id {eventoEncontrado.Id} modificado con exito");
    }
}
```

RepositorioEventoDeportivo.cs:

```
public void Modificar(EventoDeportivo evento) {
    var eventos = ListarTodos();
    var index = eventos.FindIndex(e => e.Id == evento.Id);
    if (index >= 0) {
        eventos[index] = evento;
        GuardarTodos(eventos);
    }
}
```

```
private void GuardarTodos(List<EventoDeportivo> eventos) {
    var lineas = eventos.Select(e => $"{e.Id},{e.Nombre},{e.FechaHoraInicio:yyyy-MM-dd HH:mm},{e.Lugar},{e.CupoMaximo},{e.DuracionHoras}
    File.WriteAllLines(archivo, lineas);
}
```

EventoDeportivoModificarUseCase.cs:

```
public void Ejecutar(EventoDeportivo evento,int idUsuario)
{
    if (!_servicioAutorizacion.PoseeElPermiso(idUsuario, Permiso.EventoModificacion))
        throw new UnauthorizedAccessException("El usuario no tiene permiso para modificar eventos.");
    var mod = _repoEvento.ObtenerPorId(evento.Id) ?? throw new EntidadNotFoundException("El evento no existe");
    if (evento.FechaHoraInicio < DateTime.Now) throw new OperacionInvalidaException("No se puede modificar un evento terminado");
    _Validar.Validar(evento);
    _repoEvento.Modificar(evento);
}
```

Ejemplo 11: Modificar Reserva

seleccionar la opción 9 para modificar, en este caso, se modifica el evento que se reservó

```
Insertar Id de la reserva a modificar:

9

Reserva Encontrada: 9

Insertar nuevo id de evento:

3

id valido

Reserva con id 9 modificada con éxito
```

Program.cs:

```
Console.WriteLine("Insertar Id de la reserva a modificar: ");
string? idReservaModificar = Console.ReadLine();
if (int.TryParse(idReservaModificar, out int ReservaModificarId)) {
    var encontrarReserva = repoReserva.ObtenerPorId(ReservaModificarId);
    if (encontrarReserva is not null) {
        Console.WriteLine($"Reserva Encontrada: {encontrarReserva.Id}");
        var ModificarReserva = new ReservaModificarUseCase(repoReserva,servicioAutorizacion);
        Console.WriteLine("Insertar nuevo id de evento: ");
        string? idnuevo = Console.ReadLine();
        if (int.TryParse(idnuevo, out int nuevoid))
            Console.WriteLine("id valido");
        else
            Console.WriteLine("id Invalido");
        encontrarReserva.EventoDeportivoId = nuevoid;
        ModificarReserva.Ejecutar(encontrarReserva,idAutorizado);
        Console.WriteLine($"Reserva con id {encontrarReserva.Id} modificada con exito");
    }
}
else
    Console.WriteLine("Id de la reserva invalido");
```

RepositorioReserva.cs:

```
public void Modificar(Reserva reserva) {
    var reservas = Listar();
    var index = reservas.FindIndex(r => r.Id == reserva.Id);
    if (index >= 0)
    {
        reservas[index] = reserva;
        GuardarTodas(reservas);
    }
}
```

```
private void GuardarTodas(List<Reserva> reservas) {
    var lineas = reservas.Select(r => $"{r.Id},{r.EventoDeportivoId},{r.PersonaId},{r.FechaAltaReserva:yyyy-MM-dd HH:mm}");
    File.WriteAllLines(archivo, lineas);
}
```

ReservaModificarUseCase.cs:

```
public void Ejecutar(Persona persona,int idUsuario){
    if (!_servicioAutorizacion.PoseeElPermiso(idUsuario, Permiso.UsuarioModificacion))
            throw new UnauthorizedAccessException("El usuario no tiene permiso para modificar personas.");
    var add = repositorioPersona.ObtenerPorId(persona.Id)?? throw new EntidadNotFoundException("Persona no encontrada");
    _validador.Validar(persona);
    repositorioPersona.Modificar(persona);
}
```

ValidadorReserva.cs:

```
public void Validar(Reserva reserva) {
        var Persona = _repoPersona.ObtenerPorId(reserva.PersonaId);
        var EventoDeportivo = _repoEvento.ObtenerPorId(reserva.EventoDeportivoId);
        var cantReservas = _repoReservas.ListarPorEvento(reserva.EventoDeportivoId).Count;
        if (Persona == null)
            throw new EntidadNotFoundException("Persona no encontrada");
        if (EventoDeportivo == null)
            throw new EntidadNotFoundException("Evento Deportivo no encontrado");
        if (cantReservas >= EventoDeportivo.CupoMaximo)
            throw new CupoExcedidoException();
        if (_repoReservas.ExisteReserva(reserva.EventoDeportivoId,reserva.PersonaId))
            throw new DuplicadoException("La Persona ya tiene una reserva para este evento");
}
```

Ejemplo 12: Eliminar Persona

seleccione la opción 4 para eliminar una persona

```
Ingrese Id de la persona a eliminar:

8

Persona con Id 8 eliminada
```

Program.cs:

```
Console.WriteLine("Ingrese Id de la persona a eliminar: ");
string? inputEliminar = Console.ReadLine();
if (inputEliminar != "") {
    if (int.TryParse(inputEliminar, out int idEliminar)) {
        var eliminarPersona = new PersonaBajaUseCase(repo,servicioAutorizacion);
        eliminarPersona.Ejecutar(idEliminar,idAutorizado);
        Console.WriteLine($"Persona con Id {idEliminar} eliminada");
    }
    else
        Console.WriteLine("Id no encontrado");
}
```

RepositorioPersona.cs:

```
public void Eliminar(int id) {
    if (!File.Exists(archivo)) return;

    var personas = Listar();
    personas = personas.Where(p => p.Id != id).ToList();

    GuardarTodas(personas);
}
```

```
private void GuardarTodas(List<Persona> personas) {
    var lineas = personas.Select(p => $"{p.Id},{p.Nombre},{p.Apellido},{p.DNI},{p.Email},{p.Telefono}");
    File.WriteAllLines(archivo, lineas);
}
```

PersonaBajaUseCase.cs:

```
public void Ejecutar(int Id, int idUsuario){
    if (!_servicioAutorizacion.PoseeElPermiso(idUsuario, Permiso.UsuarioBaja))
        throw new UnauthorizedAccessException("El usuario no tiene permiso eliminar una persona");
    var PersonaEliminar =_repositorioPersona.ObtenerPorId(Id)?? throw new EntidadNotFoundException("Persona no encontrada");
    if(_repositorioPersona.EsResponsable(Id)|| _repositorioPersona.TieneReservas(Id))
        throw new OperacionInvalidaException("La persona no puede eliminarse");
    _repositorioPersona.Eliminar(Id);
}
```

Ejemplo 13: Eliminar Evento Deportivo

seleccione la opción 5 para eliminar un evento deportivo a elección

```
Ingrese el Id del evento a eliminar

3

Evento Eliminado con éxito
```

Program.cs:

```
Console.WriteLine("Ingrese el Id del evento a eliminar");
string? eventoEliminar = Console.ReadLine();
if (eventoEliminar != "") {
    if (int.TryParse(eventoEliminar, out int idEvento)) {
        var bajaEvento = new EventoDeportivoBajaUseCase(repoevento, repoReserva,servicioAutorizacion);
        bajaEvento.Ejecutar(idEvento,idAutorizado);
        Console.WriteLine("Evento Eliminado con exito");

    }
    else
        Console.WriteLine("Id del evento no encontrado");
}
```

RepositorioEventoDeportivo.cs:

```
public void Eliminar(int id) {
    var eventos = ListarTodos().Where(e => e.Id != id).ToList();
    GuardarTodos(eventos);
}
```

```
private void GuardarTodos(List<EventoDeportivo> eventos) {
    var lineas = eventos.Select(e => $"{e.Id},{e.Nombre},{e.FechaHoraInicio:yyyy-MM-dd HH:mm},{e.Lugar},{e.CupoMaximo},{e.DuracionHoras}
    File.WriteAllLines(archivo, lineas);
}
```

EventoDeportivoBajaUseCase.cs:

```
public void Ejecutar(int Id,int idUsuario){
    if (!_servicioAutorizacion.PoseeElPermiso(idUsuario, Permiso.EventoBaja))
        throw new UnauthorizedAccessException("El usuario no tiene permiso para eliminar eventos.");
    var del= _repositorioEvento.ObtenerPorId(Id)??throw new EntidadNotFoundException("Evento no encontrado");
    if(_repositorioReserva.ListarPorEvento(Id).Any())throw new OperacionInvalidaException("No se Puede eliminar un evento con reservas asociadas");
    _repositorioEvento.Eliminar(Id);
}
```

Ejemplo 14: Eliminar Reserva

seleccione la opción 6 para eliminar una reserva a elección

```
Ingresar el Id de la reserva a eliminar:

9

Reserva eliminada con éxito
```

Program.cs:

```
Console.WriteLine("Ingresar el Id de la reserva a eliminar: ");
string? reservaEliminar = Console.ReadLine();
if (reservaEliminar != "") {
    if (int.TryParse(reservaEliminar, out int idReservaEliminar)) {
        var bajaReserva = new ReservaBajaUseCase(repoReserva,servicioAutorizacion);
        bajaReserva.Ejecutar(idReservaEliminar,idAutorizado);
        Console.WriteLine("Reserva eliminada con exito");
    }
    else
        Console.WriteLine("Id de la reserva no encontrado");
}
```

RepositorioReserva.cs:

```
public void Eliminar(int id) {
    var reservas = Listar().Where(r => r.Id != id).ToList();
    GuardarTodas(reservas);
}
```

```
private void GuardarTodas(List<Reserva> reservas) {
    var lineas = reservas.Select(r => $"{r.Id},{r.EventoDeportivoId},{r.PersonaId},{r.FechaAltaReserva:yyyy-MM-dd HH:mm}");
    File.WriteAllLines(archivo, lineas);
}
```

ReservaBajaUseCase.cs:

```
public void Ejecutar(int Id,int idUsuario)
{
    if (!_servicioAutorizacion.PoseeElPermiso(idUsuario, Permiso.ReservaBaja))
        throw new UnauthorizedAccessException("El usuario no tiene permiso para eliminar reservas.");
    var del = _repo.ObtenerPorId(Id) ?? throw new EntidadNotFoundException("Reserva no encontrada");
    _repo.Eliminar(Id);
}
```

Extras:

Servicio de autorización provisorio:

```
public bool PoseeElPermiso(int idUsuario, Permiso permiso) {
    return idUsuario == 1;
}
```