

Universidad ORT Uruguay
Facultad de Ingeniería
Escuela de Tecnología

OBLIGATORIO PROGRAMACIÓN 2
DOCUMENTO FINAL



Facundo Pelufo – 290361



Matheus Melo – 229019

ORT – P2 - N2C

Docente: Santiago

BailloAnalista

Programador

Fecha de entrega del documento (05-10 -2023)

Indice

La tabla de contenido está vacía porque no has seleccionado los estilos de párrafo que deben aparecer en ella.

Código fuente de todas las clases del dominio

IValidable

```
using System;
namespace Dominio.Interface
{
    public interface Ivalidable
    {
        public void Validar();
    }
}
```

Administrador

```
using System;
namespace Dominio
{
    public class Administrador: User
    {
        //private List<Miembro> _listaMiembrosAdm = new List<Miembro>();

        public Administrador(string email, string contrasena):base(email, contrasena)
        {

        }

        public override void Validar()
        {
            base.Validar();

        }

        public void BloquearMiembro(Miembro miem)
        {
            if (miem == null) throw new Exception("Miembro no puede ser nulo!");
            miem.Validar();
            miem.Status = false;
        }

        public void desBloquearMiembro(Miembro miem) // Atualizar en el diagrama
        {
            if (miem == null) throw new Exception("Miembro no puede ser nulo!");
            miem.Validar();
            miem.Status = true;
        }

        public void CensuraPost(Post post)
```

```

        {
            if (post == null) throw new Exception("Post no puede ser nulo!");
                post.Validar();
                post.CambiaCensura();
            // Parecido con el método BloquearMiembro
        }

        public void DesbloquearPost(Post post)
        {
            if (post == null) throw new Exception("Post no puede ser nulo!");
            post.Validar();
            post.CambiaCensuraFalse();
            // Parecido con el método BloquearMiembro
        }

        public override string ToString()
        {
            return $"ADM: mail - {this.Email}";

        }

        public override string Tipo()
        {
            return "Administrador";

        }
    }
}

```

Comentario

```

using System;
namespace Dominio
{
    public class Comentario: Publicacion
    {
        public Comentario(string tituloPubli, Miembro autorPubli, DateTime fechaPubli, string
        contenido, Visibilidad visibilidad) : base(tituloPubli, autorPubli ,fechaPubli, contenido, visibilidad)
        {
        }

        public Visibilidad visibilidadComentario
        {
            set { _visibilidad = value; }
        }

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

```

```

    public void AgregarReaccionAComent(Reaccion reaccion)
    {
        _listaReacciones.Add(reaccion);
    }

    public override string ToString()
    {
        return $"COMENTARIO:\n ID: {this.IdPubli} - Fecha Post:{this.fechaPubli} - Titulo:
{this.TituloPubli} - Contenido: {this.Contenido}.";
    }

    public override double CalcularVA()
    {
        return base.CalcularVA();
    }

    public override string tipoPublicacion() // Actualizar en el diagrama
    {
        return "Comentario";
    }
}

```

Invitacion

```

using System;
using Dominio.Interface;

```

```

namespace Dominio
{
    public class Invitacion: Ivalidable
    {
        private int _idInv;
        private Miembro
        _miembroSolicitante;private Miembro
        _miembroSolicitado; private Status
        _estadoSolicitud; private DateTime
        _fechaSolicitud;

        public Invitacion(int id, Miembro miebroSolicitante, Miembro miembroSolicitado)
        {
            _idInv = id;
            _miembroSolicitante = miebroSolicitante;
            _miembroSolicitado = miembroSolicitado;
            _estadoSolicitud = Status.PENDIENTE_APROBACION;// ? En la precarga
            _fechaSolicitud = DateTime.Now;
        }

        public Miembro MiembroSolicitante
        {
            get { return _miembroSolicitante; }
        }

        public Miembro MiembroSolicitado
        {
            get { return _miembroSolicitado; }
        }

        public int IdInv
        {
            get { return _idInv; }
        }

        public void Validar()
        {
            if (_idInv < 0) throw new Exception("Id no puede ser menor a 0!");
            if (_miembroSolicitante.Status == false || _miembroSolicitado.Status ==
false) throw new Exception("Miembro censurado!"); // controlamos que el estatus del miembro
            if (_estadoSolicitud == Status.RECHAZADA || _estadoSolicitud
==Status.APROBADA) throw new Exception("El estado de la solicitud no es válido!");
            if (!(_fechaSolicitud is DateTime)) throw new Exception("La fecha de
solicitud no es de tipo DateTime.");
        }

        public void AceptarSolicitud(Miembro solicitado)
        {
            if (solicitado == null) throw new Exception("Solicitado no puede ser nulo!");
            if (solicitado.Email != _miembroSolicitado.Email) throw new Exception("No
le corresponde a este usuario aceptar");
            if(solicitado.Status == false) throw new Exception("El solicitante se
encuentra bloqueado");

            solicitado.AgregarMiembro(_miembroSolicitante);
            _miembroSolicitante.AgregarMiembro(solicitado);
        }
    }
}

```

```

        //solicitante.AgregarAmigo(solicitado); // Hace referencia al método AgregarAmigo -- >
        Miembro.
        this._estadoSolicitud = Status.APROBADA;

    }

    public void RechazarInvitacion()
    {
        this._estadoSolicitud = Status.RECHAZADA;
    }

    public override bool Equals(object? obj)
    {
        Invitacion inv = obj as Invitacion;
        return inv != null && this._idInv.Equals(inv._idInv);
    }

    public override string ToString()
    {
        return $"Id: {_idInv} - Miembro Solicitado: {_miembroSolicitado} - Miembro
        Solicitante: {_miembroSolicitante} - Fecha: {_fechaSolicitud} - Estado: {_estadoSolicitud}.";
    }
}

```

Miembro

```

using System;
namespace Dominio
{
    public class Miembro : User, IComparable<Miembro>
    {
        private string _nombre;
        private string _apellido;
        private DateTime
        _fechaNacimiento; private bool
        _status;
        private List<Miembro> _listaAmigos = new List<Miembro>();

        public Miembro(string email, string contrasena, string nombre, string apellido,
        DateTime fechaNacimiento, bool status): base(email, contrasena)
        {
            _nombre = nombre;
            _apellido = apellido;
            _fechaNacimiento = fechaNacimiento;

```

```

        _status = status;
    }

    public bool Status
    {
        get { return _status; }
        set { _status = value; }
    }

    public List<Miembro> ListaAmigos
    {
        get { return _listaAmigos; }
    }

    public override void Validar()
    {
        base.Validar();
        if (string.IsNullOrEmpty(_nombre)) throw new Exception("El nombre no
puede ser vacio");

        if (string.IsNullOrEmpty(_apellido)) throw new Exception("El apellido no puede ser vacio");
    }

    public override bool Equals(object? obj)
    {
        Miembro miembro = obj as Miembro;
        return miembro != null && this.Email.Equals(miembro.Email);
    }

    public override string ToString()
    {
        return $"Nombre: {_nombre} - Apellido: {_apellido} - Fecha de Nacimiento:
{_fechaNacimiento.ToString("dd/MM/yyyy")} - Status: {_status}. ";
    }

    public int CompareTo(Miembro m)
    {
        // Primero comparamos con el nombre

        int nombreComparacion = _apellido.CompareTo(m.Apellido);
        if (nombreComparacion != 0)
        {
            return nombreComparacion;
        }

        // Si los nombres son iguales, entonces comparamos por el apellido
        return _nombre.CompareTo(m.Nombre);
    }

    public bool EsAmigo(string emailAmigo) // Actualizar en el diagrama
    {
        bool resultado = false;

        foreach (Miembro miem in _listaAmigos)
        {
            if (miem.Email == emailAmigo)
            {
                resultado = true;
            }
        }
    }

```



```
        return resultado;
    }
}
```

Post

```
using System;
namespace Dominio
{
    public class Post: Publicacion
    {
        private string _imagenPost;
        private List<Comentario> _listaComentarios = new
        List<Comentario>();private bool _esCensurado;

        public Post(string tituloPubli, Miembro autor, DateTime fechaPubli, string contenido,
        Visibilidad visibilidad, string imagen, bool censurado) : base(tituloPubli, autor, fechaPubli, contenido,
        visibilidad)
```

```

        {
            _imagenPost = imagen;
            _esCensurado = censurado;
        }

        public List<Comentario> ListaComentario
        {
            get { return _listaComentarios; }
        }

        public override void Validar()
        {
            if (this._esCensurado != true && this._esCensurado != false) throw new Exception("Censura
tiene que ser true o false");
            if (string.IsNullOrEmpty(this._imagenPost)) throw new Exception("Nombre de la imagen no
puede estar vacia!");
            string ultimosCuatroCaracteres = this._imagenPost.Substring(this._imagenPost.Length -
4); string ultimosCuatroCaracteresUpper = ultimosCuatroCaracteres.ToUpper();
            if (ultimosCuatroCaracteresUpper != ".PNG" && ultimosCuatroCaracteresUpper != ".JPG")
            {
                throw new Exception("Imagen del post tiene que ser .png o .jpg");
            }
        }

        public bool Censurado
        {
            get { return _esCensurado; }
        }

        public string Imagen
        {
            get { return _imagenPost; }
            set { _imagenPost = value; }
        }

        public Visibilidad Visibilidad
        {
            get { return _visibilidad; }
        }

        private void ValidarImagen() // Atualizar el diagrama
        {
            if (!_imagenPost.EndsWith(".png") && !_imagenPost.EndsWith(".JPG")) throw new
Exception("Extension inválida!");
        }

        public override double CalcularVA()
        {
            double valorFinal = base.CalcularVA();

            if (this._visibilidad == Visibilidad.Publico)
            {
                valorFinal += 10;
            }
        }

```

```

        return valorFinal;

    }

    public override string ToString() //this.Contenido.Substring(0, 49);
    {
        if (this.Contenido.Length > 50)
        {
            string maxContenido = string.Empty;

            int i = 0;
            while (i < 50)
            {
                maxContenido += this.Contenido[i];
                i++;
            }

            this.Contenido = maxContenido;

        }

        return $"POST:\n ID: {this.IdPubli} - Fecha Post: {this.fechaPubli} - Titulo: {this.TituloPubli} -
Contenido: {this.Contenido}.";

    }

    public void AgregarComentario(Comentario coment)
    {
        if (_visibilidad == Visibilidad.Privado)
        {
            coment.visibilidadComentario = Visibilidad.Privado;
        }

        if (_visibilidad == Visibilidad.Publico)
        {
            coment.visibilidadComentario = Visibilidad.Publico;
        }

        _listaComentarios.Add(coment);
    }

```

```
public void CambiaCensura()
{
    _esCensurado = true;
}
```

```
public void CambiaCensuraFalse() // Atualizar en el diagrama
{
    _esCensurado = false;
}
```

```
public void AgregarReaccionAPost(Reaccion reaccion)
{
    _listaReacciones.Add(reaccion)
}
```

```
public List<Comentario> BuscarComentariosMiembro(string email)
{
    List<Comentario> listaAuxComentariosMiembro = new List<Comentario>();
    foreach (Comentario com in _listaComentarios)
    {
        if (com.AutorPubli.Email == email)
        {
            listaAuxComentariosMiembro.Add(com);
        }
    }

    return listaAuxComentariosMiembro;
}
```

```
public bool HayComentarioDelMiembro(string email)
{
    bool hayComent = false;
    foreach(Comentario com in ListaComentario)
    {
        if (com.Autor.Email == email) hayComent = true;
    }

    return hayComent;
}
```

```
public override string tipoPublicacion()
{
    return "Post";
}
```

```
}  
}
```

Publicacion

```
using System;  
using Dominio.Interface;  
  
namespace Dominio  
{  
    public class Publicacion : Ivalidable, IComparable<Publicacion>  
    {  
  
        private static int s_idPubli =  
            1; private int _idPubli;  
        private string _tituloPubli;  
        private Miembro _autorPubli;  
        private DateTime  
            _fechaPubli; private string  
            _contenido;  
        protected Visibilidad _visibilidad;  
        protected List<Reaccion> _listaReacciones = new List<Reaccion>();  
  
        public Publicacion(string tituloPubli, Miembro autor, DateTime fechaPubli, string contenido,  
            Visibilidad visibilidad)  
        {  
            _idPubli = s_idPubli;  
            _tituloPubli = tituloPubli;  
            _autorPubli = autor;  
            _fechaPubli = fechaPubli;  
            _contenido = contenido;  
            _visibilidad =  
                visibilidad; s_idPubli++;  
        }  
  
        public Miembro AutorPubli  
        {  
            get { return _autorPubli; }  
        }  
  
        public string Contenido  
        {  
            get { return _contenido; }  
            set { _contenido = value; }  
        }  
        public int IdPubli  
        {  
            get { return _idPubli; }  
        }  
  
        public Miembro Autor  
        {  
            get { return _autorPubli; }  
        }  
  
        public DateTime fechaPubli  
        {  
            get { return _fechaPubli; }  
        }  
  
        public string TituloPubli  
        {  
            get { return _tituloPubli; }  
        }  
    }  
}
```

```

}

public List<Reaccion> ListaReaccion
{
    get { return _listaReacciones; }
}

public virtual void Validar()
{
    if (_idPubli < 0) throw new Exception("El id de la publicación no puede ser menor a 0!");
    if (string.IsNullOrEmpty(_tituloPubli)) throw new Exception("Titulo inválido!");
    if (string.IsNullOrEmpty(_contenido)) throw new Exception("El contenido no puede ser vacío!");
    if (_autorPubli == null) throw new Exception("Autor no puede ser nulo!");
}

public virtual double CalcularVA()
{
    double contadorLike = 0;
    double contadorDislike = 0;
    double valorFinal;

    foreach (Reaccion reac in _listaReacciones)
    {
        if (reac.Tipo == ReaccionEnum.like)
        {
            contadorLike++;
        }
    }

    foreach (Reaccion reac in _listaReacciones)
    {
        if (reac.Tipo == ReaccionEnum.dislike)
        {
            contadorDislike++;
        }
    }

    valorFinal = (contadorLike * 5) + (contadorDislike * -2);

    return valorFinal;
}

public int CompareTo(Publicacion? other) // Utilizar en Post para ordenar la lista en forma descendente.
{
    return _tituloPubli.CompareTo(other.TituloPubli) * -1;
}

public override string ToString()
{
    return $"id post: {_idPubli} - Titulo: {_tituloPubli}";
}

```

```

public override bool Equals(object? obj)
{
    Publicacion pub = obj as Publicacion;
    return pub != null && this.IdPubli.Equals(pub.IdPubli);
}

public int CantidadLikes() // Actualizar en el diagrama
{
    int contador = 0;
    foreach (Reaccion r in _listaReacciones)
    {
        if(r.Tipo == ReaccionEnum.like)
        {
            contador++;
        }
    }

    return contador;
}

public int CantidadDislikes() // Actualizar en el diagrama
{
    int contador = 0;
    foreach (Reaccion r in _listaReacciones)
    {
        if (r.Tipo == ReaccionEnum.dislike)
        {
            contador++;
        }
    }

    return contador;
}

public abstract string tipoPublicacion(); // Actualizar en el diagrama
}
}

```

Reaccion

```

using System;
using Dominio.Interface;

```

```

namespace Dominio
{
    public class Reaccion: Ivalidable
    {
        private ReaccionEnum _tipo;
        private Miembro _miembroReaccion;

        public ReaccionEnum Tipo
        {
            get { return _tipo; }
        }

        public Miembro MiembroReacciono
        {
            get { return _miembroReaccion; }
        }

        public Reaccion(ReaccionEnum tipo, Miembro miembro)
        {
            _tipo = tipo;
            _miembroReaccion = miembro;
        }

        public void Validar()
        {
            //?
        }

        public override string ToString()
        {
            return $"Tipo {_tipo} - Miembro: {_miembroReaccion}";
        }
    }
}

```

ReaccionEnum

```

using System;
namespace Dominio
{
    public enum ReaccionEnum
    {
        like = 1,
        dislike = 2
    }
}

```


Sistema

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

namespace Dominio
{
    public class Sistema
    {
        private static Sistema _instancia;
        private List<User> _listaUsuarios = new List<User>();
        private List<Invitacion> _listaInvitaciones = new
        List<Invitacion>();private List<Post> _listaPosts = new
        List<Post>();

        public static Sistema Instancia // Singleton
        {
            get
            {
                if (_instancia == null)
                {
                    _instancia = new Sistema();
                }
                return _instancia;
            }
        }

        public List<Post> ListaPost
        {
            get { return _listaPosts; }
        }

        public List<User> ListaUsers
        {
            get { return _listaUsuarios; }
        }

        public List<Invitacion> ListaInvitaciones
        {
            get { return _listaInvitaciones; }
        }

        private Sistema()
        {
        }
    }
}
```

```

//PreCargaAdm();
PreCargaMiembro();

PreCargaPost();
PreCargaReaccionComentario();
PreCargaReaccionPost();
PreCargaInvitaciones();

PreCargaVinculos();

}

```

```

public void PreCargaMiembro()
{
    // PrecargaMiembro
    AltaMiembro(new Miembro("matheus.caique21@gmail.com", "mat123", "Matheus", "Melo",
newDateTime(1994, 12, 08), true));
    AltaMiembro(new Miembro("facupelu26@gmail.com", "facu123", "Facundo", "Peluf",
newDateTime(1996, 12, 08), true));
    AltaMiembro(new Miembro("johndoe@gmail.com", "password1", "John", "Doe",
newDateTime(1985, 7, 25), true));
    AltaMiembro(new Miembro("janedoe@gmail.com", "password2", "Jane", "Doe",
newDateTime(1990, 9, 10), true));
    AltaMiembro(new Miembro("alice.smith@gmail.com", "password3", "Alice", "Smith",
newDateTime(1980, 5, 3), true));
    AltaMiembro(new Miembro("bob.johnson@gmail.com", "password4", "Bob", "Johnson",
newDateTime(1975, 2, 18), true));
    AltaMiembro(new Miembro("susan.wilson@gmail.com", "password5", "Susan", "Wilson", new
DateTime(1992, 11, 8), true));
    AltaMiembro(new Miembro("kevin.martin@gmail.com", "password6", "Kevin", "Martin",
newDateTime(1988, 4, 12), true));
    AltaMiembro(new Miembro("lperez@gmail.com", "password7", "Laura", "Perez",
newDateTime(1987, 6, 30), true));
    AltaMiembro(new Miembro("michael.jackson@gmail.com", "password8", "Michael",
"Jackson",new DateTime(1995, 1, 22), true));

}

```

```

public int RetornaUltimoID()
{
    int max = int.MinValue;
    foreach(Invitacion inv in _listaInvitaciones)
    {
        if(inv.IdInv > max)
        {
            max = inv.IdInv;
        }
    }

    return max += 1;
}

```

```
public void PreCargaInvitaciones()
{
    // Invitaciones Matheus con tod*s
    Invitacion inv1 = new Invitacion(1,
RetornaMiembroPorMail("matheus.caique21@gmail.com"),
RetornaMiembroPorMail("facupelu26@gmail.com"));
    Invitacion inv2 = new Invitacion(2,
RetornaMiembroPorMail("matheus.caique21@gmail.com"),
RetornaMiembroPorMail("johndoe@gmail.com"));
    Invitacion inv3 = new Invitacion(3,
RetornaMiembroPorMail("matheus.caique21@gmail.com"),
RetornaMiembroPorMail("janedoe@gmail.com"));
    Invitacion inv4 = new Invitacion(4,
RetornaMiembroPorMail("matheus.caique21@gmail.com"),
RetornaMiembroPorMail("alice.smith@gmail.com"));
    Invitacion inv5 = new Invitacion(5,
RetornaMiembroPorMail("matheus.caique21@gmail.com"),
RetornaMiembroPorMail("susan.wilson@gmail.com"));
```

```
Invitacion inv6 = new Invitacion(6,
RetornaMiembroPorMail("matheus.caique21@gmail.com"),
RetornaMiembroPorMail("kevin.martin@gmail.com"));
Invitacion inv7 = new Invitacion(7,
RetornaMiembroPorMail("matheus.caique21@gmail.com"),
RetornaMiembroPorMail("lperez@gmail.com"));
Invitacion inv8 = new Invitacion(8,
RetornaMiembroPorMail("matheus.caique21@gmail.com"),
RetornaMiembroPorMail("bob.johnson@gmail.com"));
Invitacion inv9 = new Invitacion(9,
RetornaMiembroPorMail("matheus.caique21@gmail.com"),
RetornaMiembroPorMail("michael.jackson@gmail.com"));
```

//// Invitaciones Facundo con tod*s

```
Invitacion inv10 = new Invitacion(10,
RetornaMiembroPorMail("facupelu26@gmail.com"),
RetornaMiembroPorMail("michael.jackson@gmail.com"));
Invitacion inv11 = new Invitacion(11, RetornaMiembroPorMail("facupelu26@gmail.com"),
RetornaMiembroPorMail("johndoe@gmail.com"));
Invitacion inv12 = new Invitacion(12,
RetornaMiembroPorMail("facupelu26@gmail.com"),
RetornaMiembroPorMail("janedoe@gmail.com"));
Invitacion inv13 = new Invitacion(13,
RetornaMiembroPorMail("facupelu26@gmail.com"),
RetornaMiembroPorMail("alice.smith@gmail.com"));
Invitacion inv14 = new Invitacion(14,
RetornaMiembroPorMail("facupelu26@gmail.com"),
RetornaMiembroPorMail("susan.wilson@gmail.com"));
Invitacion inv15 = new Invitacion(15,
RetornaMiembroPorMail("facupelu26@gmail.com"),
RetornaMiembroPorMail("kevin.martin@gmail.com"));
Invitacion inv16 = new Invitacion(16,
RetornaMiembroPorMail("facupelu26@gmail.com"),
RetornaMiembroPorMail("lperez@gmail.com"));
Invitacion inv17 = new Invitacion(17,
RetornaMiembroPorMail("facupelu26@gmail.com"),
RetornaMiembroPorMail("bob.johnson@gmail.com"));
```

//// Invitaciones rechazadas

```
Invitacion inv18 = new Invitacion(18,
RetornaMiembroPorMail("alice.smith@gmail.com"),
RetornaMiembroPorMail("bob.johnson@gmail.com"));
Invitacion inv19 = new Invitacion(19,
RetornaMiembroPorMail("johndoe@gmail.com"),
RetornaMiembroPorMail("janedoe@gmail.com"));
```

// Invitaciones Pendientes

```
Invitacion inv20 = new Invitacion(20,
RetornaMiembroPorMail("michael.jackson@gmail.com"),
RetornaMiembroPorMail("lperez@gmail.com"));
Invitacion inv21 = new Invitacion(21,
RetornaMiembroPorMail("lperez@gmail.com"),
RetornaMiembroPorMail("kevin.martin@gmail.com"));
```

```
AltaInvitacion(inv1);
AltaInvitacion(inv2);
AltaInvitacion(inv3);
AltaInvitacion(inv4);
AltaInvitacion(inv5);
AltaInvitacion(inv6);
AltaInvitacion(inv7);
AltaInvitacion(inv8);
AltaInvitacion(inv9);
AltaInvitacion(inv10);
AltaInvitacion(inv11);
```

AltaInvitacion(inv12);
AltaInvitacion(inv13);
AltaInvitacion(inv14);
AltaInvitacion(inv15);
AltaInvitacion(inv16);
AltaInvitacion(inv17);
AltaInvitacion(inv18);

```

        AltaInvitacion(inv19)
        ;
        AltaInvitacion(inv20)
        ;
        AltaInvitacion(inv21);

    }

    public void PreCargaVinculos()
    {

        // Aceptadas
        AceptarInvitacion(1,"facupelu26@gmail.com");
        AceptarInvitacion(2, "johndoe@gmail.com");
        AceptarInvitacion(3, "janedoe@gmail.com");
        AceptarInvitacion(4, "alice.smith@gmail.com");
        AceptarInvitacion(5, "susan.wilson@gmail.com");
        AceptarInvitacion(6, "kevin.martin@gmail.com");
        AceptarInvitacion(7, "lperez@gmail.com");
        AceptarInvitacion(8, "bob.johnson@gmail.com");
        AceptarInvitacion(9, "michael.jackson@gmail.com");
        AceptarInvitacion(10,
        "michael.jackson@gmail.com"); AceptarInvitacion(11,
        "johndoe@gmail.com"); AceptarInvitacion(12,
        "janedoe@gmail.com"); AceptarInvitacion(13,
        "alice.smith@gmail.com"); AceptarInvitacion(14,
        "susan.wilson@gmail.com"); AceptarInvitacion(15,
        "kevin.martin@gmail.com"); AceptarInvitacion(16,
        "lperez@gmail.com"); AceptarInvitacion(17,
        "bob.johnson@gmail.com");

        //Rechazadas
        RechazarInvitacion(18)
        ;
        RechazarInvitacion(19);

    }

    public void PreCargaAdm()
    {
        // PrecargaAdm
        AltaAdm(new Administrador("ernesto.perez@gmail.com", "mat123"));
    }

    public void PreCargaReaccionPost()
    {
        Post post = RetornaPostPorId(1);
        Miembro miem =
        RetornaMiembroPorMail("matheus.caique21@gmail.com"); if (!
        HayReaccionPost(1, "matheus.caique21@gmail.com"))
        {
            post.AgregarReaccionAPost(new Reaccion(ReaccionEnum.like, miem));
        }

        Post post2 = RetornaPostPorId(5);
        Miembro miem2 =
        RetornaMiembroPorMail("facupelu26@gmail.com"); if (!
        HayReaccionPost(5, "facupelu26@gmail.com"))
        {
            post2.AgregarReaccionAPost(new Reaccion(ReaccionEnum.like, miem2));
        }
    }

```

```
}
```

```
}
```

```
public void PreCargaReaccionComentario()
```

```
{
```

```
    Comentario com = RetornaComentariosPorId(2);
```

```
    Miembro miem = RetornaMiembroPorMail("facupelu26@gmail.com");
```

```
    if (!HayReaccionComentario(2, "facupelu26@gmail.com"))
```

```
    {
```

```
        com.AgregarReaccionAComent(new Reaccion(ReaccionEnum.like, miem));
```

```
    }
```

```
    Comentario com2 = RetornaComentariosPorId(3);
```

```
    Miembro miem3 = RetornaMiembroPorMail("facupelu26@gmail.com");
```

```
    if (!HayReaccionComentario(3, "facupelu26@gmail.com"))
```

```
    {
```

```
        com2.AgregarReaccionAComent(new Reaccion(ReaccionEnum.like, miem3));
```

```
    }
```

```
}
```

```
public void PreCargaPost()
```

```
{
```

```
    Post post1 = (new Post("Post 1", RetornaMiembroPorMail("matheus.caique21@gmail.com"),  
DateTime.Now, "Contenido post 1: It is a long established fact that a reader will be distracted by the readable  
content of a page when looking at its layout. The point of using Lorem Ipsum is that it has a more-or-less  
normal distribution of letters, as opposed to using 'Content here, content here', making it look like readable  
English. Many desktop publishing packages and web page editors now use Lorem Ipsum as their default  
model text, and a search for 'lorem ipsum' will uncover many web sites still in their infancy. Various versions  
have evolved over the years, sometimes by accident, sometimes on purpose (injected humour and the like).",  
Visibilidad.Publico, "imagenPost1.jpg", false));
```

```
    post1.AgregarComentario(
```

```
        new Comentario("Titulo comentario 2",
```

```
RetornaMiembroPorMail("matheus.caique21@gmail.com"), DateTime.Now, "Contenido comentario 2 post  
1", Visibilidad.Publico));
```

```
    post1.AgregarComentario(
```

```
        new Comentario("Titulo comentario 2",
RetornaMiembroPorMail("matheus.caique21@gmail.com"), DateTime.Now, "Contenido comentario 2 post
1", Visibilidad.Publico));
        post1.AgregarComentario(
            new Comentario("Titulo comentario 3", RetornaMiembroPorMail("facupelu26@gmail.com"),
DateTime.Now, "Contenido comentario 3 post 1", Visibilidad.Publico));
```

```
        Post post2 = (new Post("Post 2",
RetornaMiembroPorMail("matheus.caique21@gmail.com"),DateTime.Now, "Contenido post 2",
Visibilidad.Publico, "imagenPost2.jpg", false));
        post2.AgregarComentario(
            new Comentario("Titulo comentario 1", RetornaMiembroPorMail("facupelu26@gmail.com"),
DateTime.Now, "Contenido comentario 1 post 2", Visibilidad.Publico));
```

```
        post2.AgregarComentario(
            new Comentario("Titulo comentario 2", RetornaMiembroPorMail("facupelu26@gmail.com"),
DateTime.Now, "Contenido comentario 1 post 2", Visibilidad.Publico));
```

```
        post2.AgregarComentario(
            new Comentario("Titulo comentario 3", RetornaMiembroPorMail("facupelu26@gmail.com"),
DateTime.Now, "Contenido comentario 1 post 2", Visibilidad.Publico));
```

```
        post2.AgregarComentario(
            new Comentario("Titulo comentario 4", RetornaMiembroPorMail("facupelu26@gmail.com"),
DateTime.Now, "Contenido comentario 1 post 2", Visibilidad.Publico));
```

```
        post2.AgregarComentario(
            new Comentario("Titulo comentario 5 ", RetornaMiembroPorMail("facupelu26@gmail.com"),
DateTime.Now, "Contenido comentario 2 post 2", Visibilidad.Publico));
```

```
        post2.AgregarComentario(
            new Comentario("Titulo comentario 6",
RetornaMiembroPorMail("matheus.caique21@gmail.com"), DateTime.Now, "Contenido comentario 3 post
2", Visibilidad.Publico));
```

```
        Post post3= (new Post("Post 3",
RetornaMiembroPorMail("susan.wilson@gmail.com"),DateTime.Now, "Contenido post 3",
Visibilidad.Publico, "imagenPost3.jpg", false));
        post3.AgregarComentario(
            new Comentario("Titulo comentario 1", RetornaMiembroPorMail("kevin.martin@gmail.com"),
DateTime.Now, "Contenido comentario 1 post 3", Visibilidad.Publico));
        post3.AgregarComentario(
            new Comentario("Titulo comentario 2", RetornaMiembroPorMail("janedoe@gmail.com"),
DateTime.Now, "Contenido comentario 2 post 3", Visibilidad.Publico));
        post3.AgregarComentario(
            new Comentario("Titulo comentario 3",
RetornaMiembroPorMail("matheus.caique21@gmail.com"), DateTime.Now, "Contenido comentario 3 post
3", Visibilidad.Publico));
```

```
        Post post4 = (new Post("Post 4",
RetornaMiembroPorMail("facupelu26@gmail.com"),DateTime.Now, "Contenido post 4",
Visibilidad.Publico, "imagenPost3.jpg", false));
        post4.AgregarComentario(
            new Comentario("Titulo comentario 1", RetornaMiembroPorMail("kevin.martin@gmail.com"),
DateTime.Now, "Contenido comentario 1 post 4", Visibilidad.Publico));
        post4.AgregarComentario(
```



```

        new Comentario("Titulo comentario 2",
RetornaMiembroPorMail("matheus.caique21@gmail.com"), DateTime.Now, "Contenido comentario 2 post
4", Visibilidad.Publico));
        post4.AgregarComentario(
            new Comentario("Titulo comentario 3",
RetornaMiembroPorMail("matheus.caique21@gmail.com"), DateTime.Now, "Contenido comentario 3 post
4", Visibilidad.Publico));

```

```

        Post post5 = (new Post("Post 5",
RetornaMiembroPorMail("facupelu26@gmail.com"), DateTime.Now, "Contenido post 4",
Visibilidad.Publico, "imagenPost3.jpg", false));
        post5.AgregarComentario(
            new Comentario("Titulo comentario 1", RetornaMiembroPorMail("kevin.martin@gmail.com"),
DateTime.Now, "Contenido comentario 1 post 5", Visibilidad.Publico));
        post5.AgregarComentario(
            new Comentario("Titulo comentario 2",
RetornaMiembroPorMail("matheus.caique21@gmail.com"), DateTime.Now, "Contenido comentario 2 post
5", Visibilidad.Publico));
        post5.AgregarComentario(
            new Comentario("Titulo comentario 3",
RetornaMiembroPorMail("matheus.caique21@gmail.com"), DateTime.Now, "Contenido comentario 3 post
5", Visibilidad.Publico));
        post5.AgregarComentario(
            new Comentario("Titulo comentario 5", RetornaMiembroPorMail("facupelu26@gmail.com"),
DateTime.Now, "Contenido comentario 2 post 2", Visibilidad.Publico));
        post5.AgregarComentario(
            new Comentario("Titulo comentario 5", RetornaMiembroPorMail("facupelu26@gmail.com"),
DateTime.Now, "Contenido comentario 2 post 2", Visibilidad.Publico));

```

```

        AltaPost(post1)
        ;
        AltaPost(post2)
        ;
        AltaPost(post3)
        ;
        AltaPost(post4)
        ;
        AltaPost(post5);
    }

```

```

public bool HayReaccionPost(int id, string email)
// Para validar que no haya una reaccion del miembro que quiere reaccionar en el POST
{
    Miembro m =
    RetornaMiembroPorMail(email); Post publi =
    RetornaPostPorId(id);
    bool hayReaccion = false;

    foreach(Reaccion r in publi.ListaReaccion)
    {
        if (r.MiembroReacciono.Email == email)
        {
            hayReaccion = true;
            throw new Exception("Ya hay una reaccion de esta persona");
        }
        else
        {
            hayReaccion = false;
        }
    }
}

```

```

    }

    return hayReaccion;
}

public bool HayReaccionComentario(int id, string email)
// Para validar que no haya una reaccion del miembro que quiere reaccionar
{
    Miembro m = RetornaMiembroPorMail(email);
    Comentario comen =
    RetornaComentariosPorId(id); bool hayReaccion =
    false;

    foreach (Reaccion r in comen.ListaReaccion)
    {
        if (r.MiembroReacciono.Email == m.Email)
        {
            hayReaccion = true;
            throw new Exception("Ya hay una reaccion de esta persona");
        }
        else
        {
            hayReaccion = false;
        }
    }
}

return hayReaccion;
}

```

```

public void AltaPost(Post post)
{
    if (post == null) throw new Exception("Post no puede ser nulo!");
    if (_listaPosts.Contains(post)) throw new Exception("Ya existe un post en la lista!");
    post.Validar();
    _listaPosts.Add(post);
}

```

```

public void AltaAdm(Administrador adm)
{
    //if (adm == null) throw new Exception("Adm recibido por parametro no es
    válido!"); adm.Validar();
    //if (_listaUsuarios.Contains(adm)) throw new Exception("Ya existe un adm con este email!");
    _listaUsuarios.Add(adm);
}

```

```

public void AltaMiembro(Miembro miembro) // Menu 1
{
    if (miembro == null) throw new Exception("Miembro recibido por parametro no es
    válido!"); miembro.Validar();
    if (_listaUsuarios.Contains(miembro)) throw new Exception("Ya existe un miembro con este
    email!");
}

```

```

    _listaUsuarios.Add(miembro);
}

public void AltaInvitacion(Invitacion invitacion)
{
    if (invitacion == null) throw new Exception("Invitación recibida por parametro!");
    // if(_listaInvitaciones.Contains(invitacion)) throw new Exception("Invitación ya enviada!");
    invitacion.Validar();
    if (HayInvitacion(invitacion)) throw new Exception("Ya hay una invitacion");
    _listaInvitaciones.Add(invitacion);
}

public void AceptarInvitacion(int Idinvitacion, string emailAprobador)
{
    if (Idinvitacion < 0) throw new Exception("El id de la invitación pasado por parametro no es válido!");
    Invitacion invitacionBuscado = RetornaIdInv(Idinvitacion);
    if (invitacionBuscado == null) throw new Exception("La invitación no puede ser nula!");

    //
    Miembro m1 =RetornaMiembroPorMail(emailAprobador);

    invitacionBuscado.AceptarSolicitud(m1);
}

public void AltaComentario(Comentario coment, int idPost)
{
    if (coment == null) throw new Exception("El comentario no puede ser nulo");if (idPost < 0) throw new Exception("Id de post no puede ser menor a 0!"); coment.Validar();
    Post postBuscado =

    RetornaPostPorId(idPost);

    postBuscado.AgregarComentario(coment);
}

public Invitacion RetornaIdInv(int id)
{
    Invitacion invBuscada = null;
    foreach (Invitacion inv in _listaInvitaciones)
    {
        if(inv.IdInv == id)
        {
            invBuscada = inv;
        }
    }
    return invBuscada;
}

public void BloquearMiembro(Administrador adm, Miembro miem) // Bloquear miembro
{
    adm.BloquearMiembro(miem); // BloquearMiembro está en Adminiestrador.
}

```

```

public void RechazarInvitacion(int Idinvitacion)
{
    if (Idinvitacion < 0) throw new Exception("El id de la invitación pasado por parametro no es válido!");
    Invitacion invitacionBuscado = RetornaIdInv(Idinvitacion);
    if (invitacionBuscado == null) throw new Exception("La invitación no puede ser nula!");
    invitacionBuscado.Validar();
    invitacionBuscado.RechazarInvitacion();
}

```

```

public Miembro BuscarMiembro(Miembro miembro)
{
    Miembro miembroBuscado = null;

    foreach (Miembro miem in _listaUsuarios)
    {
        if (miem.Email == miembro.Email)
        {
            miembroBuscado = miem;
        }
    }

    return miembroBuscado;
}

```

```

public Miembro RetornaMiembroPorMail(string email) // Editar en el DIAGRAMA!
{
    Miembro miembroBuscado = null;
    int i = 0;

    while (miembroBuscado == null && i < _listaUsuarios.Count)
    {
        User usuario = _listaUsuarios[i];

        if (usuario is Miembro miem && miem.Email == email) // Casteo para poder iterar sobre la lista de Usuario
        {
            miembroBuscado = miem;
        }

        i++;
    }

    return miembroBuscado;
}

```

```

public Comentario RetornaComentariosPorId(int id)
// retorna Comentario buscado en la lista de comentarios de un post
{
    Comentario comentarioBuscado = null;

    foreach (Post pub in _listaPosts)
    {
        foreach(Comentario comen in pub.ListaComentario)
        {
            if (comen.IdPubli == id) comentarioBuscado = comen;
        }
    }

    return comentarioBuscado;
}

```

// PARA EL POST

```

public Post RetornaPostPorId(int id) // retorna Posts
{
    Post publiBuscada = null;

    foreach (Post pub in _listaPosts)
    {
        if (pub.IdPubli == id)
        {
            publiBuscada = pub;
        }
    }

    return publiBuscada;
}

```

```

public List<Publicacion> RetornaPublicaciones(string email) // Menu 2
{
    Miembro miembroBuscado = RetornaMiembroPorMail(email);
    if (miembroBuscado == null) throw new Exception("Miembro no existe en la lista de usuarios!");

    List<Publicacion> listaFinal = new

    List<Publicacion>();foreach (Post post in _listaPosts)

    {

        if (post.AutorPubli.Email == email)
        {
            listaFinal.Add(post);
        }

        listaFinal.AddRange(post.BuscarComentariosMiembro(email));

    }

    return listaFinal;
}

```

```

public List<Post> RetornaPostsConComentariosDeMiembro(string email) // Menu 3
{
    if (RetornaMiembroPorMail(email) == null) throw new Exception("Miembro no
existe!"); //Validación existe miembro o no (utilizando método auxiliar para buscar el miembro en la
lista).

```

```

    List<Post> postsConComentarios = new List<Post>();

```

```

    foreach (Post post in _listaPosts)
    {
        if (post.HayComentarioDelMiembro(email)) postsConComentarios.Add(post);
    }

```

```

    // if (postsConComentarios.Count == 0) throw new Exception("no hay
registros!");return postsConComentarios;

```

```

}

```

```

public List<Publicacion> RetornaListaPostFechas(DateTime fechaInic, DateTime fechaFinal) // Menu 4
{
    if (fechaInic == null || fechaFinal == null) throw new Exception("Fecha no
válida!");List<Publicacion> listaAuxPost = new List<Publicacion>();

```

```

    foreach (Publicacion publi in _listaPosts)
    {
        if(publi is Post)
        {

```

```

            Post post = ((Post)publi); // casteando

```

```

            if(post.fechaPubli.Date >= fechaInic.Date && post.fechaPubli.Date <=
fechaFinal.Date) //Utilizando DateTime.Date para implementar la condición de las fechas.
            {
                listaAuxPost.Add(post);
            }
        }
    }

```

```

    if (listaAuxPost.Count == 0) throw new Exception("no hay
registros!");listaAuxPost.Sort();

```

```

    return listaAuxPost;
}

```

```

public int RetornaCantidadPostPorMiembro(string email) // Método auxiliar
paraRetornaMiembrosConMasPublicaciones

```

```

{
    int contador = 0;

```

```

    foreach (Publicacion pub in _listaPosts)
    {
        if(pub.AutorPubli.Email == email)
        {
            contador++;
        }
    }

```

```

        if (pub is Post)
        {
            Post post = ((Post)pub);

            foreach (Comentario comen in post.ListaComentario) // casteando acá // Iterando sobre la
listade comentario del post.
            {
                if (comen.AutorPubli.Email == email)
                {
                    contador++;
                }
            }
        }
    }

    return contador;
}

```

```

public List<Miembro> RetornaMiembrosConMasPublicaciones() // Menu 5
{
    List<Miembro> miembrosConMasPublicaciones = new
List<Miembro>(); int maximoPublicaciones = int.MinValue;

    foreach (Miembro miembro in _listaUsuarios)
    {
        int totalPublicaciones = RetornaCantidadPostPorMiembro(miembro.Email);

        if (totalPublicaciones > maximoPublicaciones)
        {
            maximoPublicaciones = totalPublicaciones;
            miembrosConMasPublicaciones.Clear();
            miembrosConMasPublicaciones.Add(miembro);
        }
        else if (totalPublicaciones == maximoPublicaciones)
        {
            miembrosConMasPublicaciones.Add(miembro);
        }
    }
    if (miembrosConMasPublicaciones.Count <= 0) throw new Exception("No hay registros!");

    return miembrosConMasPublicaciones;
}

```

```

public List<Miembro> RetornaListaAmigos(Miembro m)
{
    List<Miembro> listaAmigos = new
List<Miembro>(); Miembro miembroBuscado = null;

    foreach (Miembro miem in ListaUsers)
    {
        if (m.Email == miem.Email)
        {

```

```

        miembroBuscado = miem;
    }
}

foreach(Miembro miembroFound in miembroBuscado.ListaAmigos)
{
    listaAmigos.Add(miembroFound);
}

return listaAmigos;
}

```

```

public List<Reaccion> ListarReacciones()
{
    List<Reaccion> lista = new List<Reaccion>();

    foreach (Post post in _listaPosts)
    {
        foreach (Reaccion reac in post.ListaReaccion)
        {
            lista.Add(reac);
        }

        foreach (Comentario c in post.ListaComentario)
        {
            foreach (Reaccion r in c.ListaReaccion)
            {
                lista.Add(r);
            }
        }
    }
    return lista;
}

```

```

public User Login(string email, string pass) // Actualizar en el diagrama
{
    User usuario = null;
    int i = 0;
    while (usuario == null && i < _listaUsuarios.Count)
    {
        if (_listaUsuarios[i].Email == email && _listaUsuarios[i].Contraseña == pass) usuario =
        _listaUsuarios[i];
        i++;
    }

    return usuario;
}

```

```

public bool HayLikeMiembro(Miembro m, Publicacion p) // Actualizar en el diagrama
{
    bool flag = false;

    foreach (Reaccion reac in p.ListaReaccion)
    {
        if(reac.MiembroReacciono.Email == m.Email && reac.Tipo == ReaccionEnum.like)
        {
            flag = true;
        }
    }

    return flag;
}

```



```

public bool HayDislikeMiembro(Miembro m, Publicacion p) // Actualizar en el diagrama
{
    bool flag = false;

    foreach (Reaccion reac in p.ListaReaccion)
    {
        if (reac.MiembroReacciono.Email == m.Email && reac.Tipo == ReaccionEnum.dislike)
        {
            flag = true;
        }
    }

    return flag;
}

```

```

public List<Publicacion> retornaPostsConTexto(string texto, double numero)
{
    List<Publicacion> listaAux = new List<Publicacion>();

    foreach (Post p in _listaPosts)
    {
        if(p.Contenido.Contains(texto) && p.CalcularVA() > numero)
        {
            listaAux.Add(p);
        }

        foreach (Comentario c in p.ListaComentario)
        {
            if(c.Contenido.Contains(texto) && c.CalcularVA() > numero)
            {
                listaAux.Add(c);
            }
        }
    }

    return listaAux;
}

```

```

public List<Miembro> RetornaSoloMiembros() // Actualizar el diagrama
{
    List<Miembro> miembros = new List<Miembro>();

    foreach (User u in _listaUsuarios )
    {
        if(u is Miembro miem)
        {
            miembros.Add(miem);
        }
    }

    return miembros;
}

```

```

public List<Invitacion> RetornaInvitacionesPendientesMiembro(string email)
{
    List<Invitacion> listAux = new List<Invitacion>();

    foreach (Invitacion inv in _listaInvitaciones)

```

```

        {
            if(inv.MiembroSolicitado.Email == email)
            {
                listAux.Add(inv);
            }
        }

        return listAux;
    }

    public bool HayInvitacion(Invitacion inv)
    {
        bool hay = false; // flag

        foreach (Invitacion i in ListaInvitaciones)
        {
            if (inv.MiembroSolicitante.Equals(i.MiembroSolicitante) &&
                inv.MiembroSolicitado.Equals(i.MiembroSolicitado) && (i.Status == Status.APROBADA || i.Status ==
                Status.PENDIENTE_APROBACION)) hay = true;
            // i.Status.PENDIENTE_APROBACION
        }

        return hay;
    }

}
}

```

Status

```
using System;
namespace Dominio
{
    public enum Status
    {
        PENDIENTE_APROBACION = 1,
        APROBADA = 2,
        RECHAZADA = 3
    }
}
```

User

```
using System;
using Dominio.Interface;

namespace Dominio
{
    public class User: Ivalidable
    {
        private string _email;
        private string _contrasena;

        public User(string email, string contrasena)
        {
            _email = email;
            _contrasena = contrasena;
        }

        public string Email
        {
            get { return _email; }
        }

        public virtual void Validar()
        {
            if (string.IsNullOrEmpty(_email)) throw new Exception("Email no puede ser nulo
vacío!");
            if (string.IsNullOrEmpty(_contrasena)) throw new Exception("Contraseña no
puede ser nula o vacía!");
        }

        public abstract string Tipo();
    }
}
```

Visibilidad

```
using System;
namespace Dominio
{
    public enum Visibilidad
    {
        Publico = 1,
        Privado = 2,
    }
}
```

Tabla de precarga

Miembro
AltaMiembro(new Miembro("matheus.caique21@gmail.com", "mat123", "Matheus", "Melo", newDateTime(1994, 12, 08), true));
AltaMiembro(new Miembro("facupelu26@gmail.com", "facu123", "Facundo", "Pelugo", newDateTime(1996, 12, 08), true));
AltaMiembro(new Miembro("johndoe@gmail.com", "password1", "John", "Doe", newDateTime(1985, 7, 25), true));
AltaMiembro(new Miembro("janedoe@gmail.com", "password2", "Jane", "Doe", newDateTime(1990, 9, 10), true));
AltaMiembro(new Miembro("alice.smith@gmail.com", "password3", "Alice", "Smith", newDateTime(1980, 5, 3), true));
AltaMiembro(new Miembro("bob.johnson@gmail.com", "password4", "Bob", "Johnson", newDateTime(1975, 2, 18), true));
AltaMiembro(new Miembro("susan.wilson@gmail.com", "password5", "Susan", "Wilson", newDateTime(1992, 11, 8), true));
AltaMiembro(new Miembro("kevin.martin@gmail.com", "password6", "Kevin", "Martin", newDateTime(1988, 4, 12), true));
AltaMiembro(new Miembro("f", "password7", "Laura", "Perez", new DateTime(1987, 6, 30), false));
AltaMiembro(new Miembro("michael.jackson@gmail.com", "password8", "Michael", "Jackson", new DateTime(1995, 1, 22), false));


```
// Invitaciones Matheus con tod*s
    Invitacion inv1 = new Invitacion(1,
RetornaMiembroPorMail("matheus.caique21@gmail.com"),
RetornaMiembroPorMail("facupelu26@gmail.com"));
    Invitacion inv2 = new Invitacion(2,
RetornaMiembroPorMail("matheus.caique21@gmail.com"),
RetornaMiembroPorMail("johndoe@gmail.com"));
    Invitacion inv3 = new Invitacion(3,
RetornaMiembroPorMail("matheus.caique21@gmail.com"),
RetornaMiembroPorMail("janedoe@gmail.com"));
    Invitacion inv4 = new Invitacion(4,
RetornaMiembroPorMail("matheus.caique21@gmail.com"),
RetornaMiembroPorMail("alice.smith@gmail.com"));
    Invitacion inv5 = new Invitacion(5,
RetornaMiembroPorMail("matheus.caique21@gmail.com"),
RetornaMiembroPorMail("susan.wilson@gmail.com"));
    Invitacion inv6 = new Invitacion(6,
RetornaMiembroPorMail("matheus.caique21@gmail.com"),
RetornaMiembroPorMail("kevin.martin@gmail.com"));
    Invitacion inv7 = new Invitacion(7,
RetornaMiembroPorMail("matheus.caique21@gmail.com"),
RetornaMiembroPorMail("lperez@gmail.com"));
    Invitacion inv8 = new Invitacion(8,
RetornaMiembroPorMail("matheus.caique21@gmail.com"),
RetornaMiembroPorMail("bob.johnson@gmail.com"));
    Invitacion inv9 = new Invitacion(9,
RetornaMiembroPorMail("matheus.caique21@gmail.com"),
RetornaMiembroPorMail("michael.jackson@gmail.com"));
```

```
//// Invitaciones Facundo con tod*s
    Invitacion inv10 = new Invitacion(10,
RetornaMiembroPorMail("facupelu26@gmail.com"),
RetornaMiembroPorMail("michael.jackson@gmail.com"));
    Invitacion inv11 = new Invitacion(11,
RetornaMiembroPorMail("facupelu26@gmail.com"),
RetornaMiembroPorMail("johndoe@gmail.com"));
    Invitacion inv12 = new Invitacion(12,
RetornaMiembroPorMail("facupelu26@gmail.com"),
RetornaMiembroPorMail("janedoe@gmail.com"));
    Invitacion inv13 = new Invitacion(13,
RetornaMiembroPorMail("facupelu26@gmail.com"),
RetornaMiembroPorMail("alice.smith@gmail.com"));
    Invitacion inv14 = new Invitacion(14,
RetornaMiembroPorMail("facupelu26@gmail.com"),
RetornaMiembroPorMail("susan.wilson@gmail.com"));
    Invitacion inv15 = new Invitacion(15,
RetornaMiembroPorMail("facupelu26@gmail.com"),
RetornaMiembroPorMail("kevin.martin@gmail.com"));
    Invitacion inv16 = new Invitacion(16,
RetornaMiembroPorMail("facupelu26@gmail.com"),
RetornaMiembroPorMail("lperez@gmail.com"));
    Invitacion inv17 = new Invitacion(17,
RetornaMiembroPorMail("facupelu26@gmail.com"),
RetornaMiembroPorMail("bob.johnson@gmail.com"));
```

```
//// Invitaciones rechazadas
    Invitacion inv18 = new Invitacion(18,
RetornaMiembroPorMail("alice.smith@gmail.com"),
RetornaMiembroPorMail("bob.johnson@gmail.com"));
    Invitacion inv19 = new Invitacion(19,
RetornaMiembroPorMail("johndoe@gmail.com"),
RetornaMiembroPorMail("janedoe@gmail.com"));
```

```
// Invitaciones Pendientes
    Invitacion inv20 = new Invitacion(20, RetornaMiembroPorMail("michael.jackson@gmail.com"),
RetornaMiembroPorMail("lperez@gmail.com"));
    Invitacion inv21 = new Invitacion(21,
```



```

Post post1 = (new Post("Post 1", RetornaMiembroPorMail("matheus.caique21@gmail.com"),
DateTime.Now, "Contenido post 1: It is a long established fact that a reader will be distracted by the
readable content of a page when looking at its layout. The point of using Lorem Ipsum is that it has a
more-or-less normal distribution of letters, as opposed to using 'Content here, content here', making it
look like readable English. Many desktop publishing packages and web page editors now use Lorem
Ipsum as their default model text, and a search for 'lorem ipsum' will uncover many web sites still in
theirinfancy. Various versions have evolved over the years, sometimes by accident, sometimes on
purpose (injected humour and the like).", Visibilidad.Publico, "imagenPost1.jpg", false));
    post1.AgregarComentario(
        new Comentario("Titulo comentario 2",
RetornaMiembroPorMail("matheus.caique21@gmail.com"), DateTime.Now, "Contenido comentario 2 post
1", Visibilidad.Publico));

    post1.AgregarComentario(
        new Comentario("Titulo comentario 2",
RetornaMiembroPorMail("matheus.caique21@gmail.com"), DateTime.Now, "Contenido comentario 2 post
1", Visibilidad.Publico));
    post1.AgregarComentario(
        new Comentario("Titulo comentario 3", RetornaMiembroPorMail("facupelu26@gmail.com"),
DateTime.Now, "Contenido comentario 3 post 1", Visibilidad.Publico));

    Post post2 = (new Post("Post 2", RetornaMiembroPorMail("matheus.caique21@gmail.com"),
DateTime.Now, "Contenido post 2", Visibilidad.Publico, "imagenPost2.jpg", false));
    post2.AgregarComentario(
        new Comentario("Titulo comentario 1", RetornaMiembroPorMail("facupelu26@gmail.com"),
DateTime.Now, "Contenido comentario 1 post 2", Visibilidad.Publico));

    post2.AgregarComentario(
        new Comentario("Titulo comentario 2", RetornaMiembroPorMail("facupelu26@gmail.com"),
DateTime.Now, "Contenido comentario 1 post 2", Visibilidad.Publico));

    post2.AgregarComentario(
        new Comentario("Titulo comentario 3", RetornaMiembroPorMail("facupelu26@gmail.com"),
DateTime.Now, "Contenido comentario 1 post 2", Visibilidad.Publico));

    post2.AgregarComentario(
        new Comentario("Titulo comentario 4", RetornaMiembroPorMail("facupelu26@gmail.com"),
DateTime.Now, "Contenido comentario 1 post 2", Visibilidad.Publico));

    post2.AgregarComentario(
        new Comentario("Titulo comentario 5 ", RetornaMiembroPorMail("facupelu26@gmail.com"),
DateTime.Now, "Contenido comentario 2 post 2", Visibilidad.Publico));

    post2.AgregarComentario(
        new Comentario("Titulo comentario 6",
RetornaMiembroPorMail("matheus.caique21@gmail.com"), DateTime.Now, "Contenido comentario 3 post
2", Visibilidad.Publico));

```

```
Post post3= (new Post("Post 3", RetornaMiembroPorMail("susan.wilson@gmail.com"), DateTime.Now,
"Contenido post 3", Visibilidad.Publico, "imagenPost3.jpg", false));
    post3.AgregarComentario(
        new Comentario("Titulo comentario 1", RetornaMiembroPorMail("kevin.martin@gmail.com"),
DateTime.Now, "Contenido comentario 1 post 3", Visibilidad.Publico));
    post3.AgregarComentario(
        new Comentario("Titulo comentario 2", RetornaMiembroPorMail("janedoe@gmail.com"),
DateTime.Now, "Contenido comentario 2 post 3", Visibilidad.Publico));
    post3.AgregarComentario(
        new Comentario("Titulo comentario 3",
RetornaMiembroPorMail("matheus.caique21@gmail.com"), DateTime.Now, "Contenido comentario 3 post
3", Visibilidad.Publico));
```

```
Post post4 = (new Post("Post 4", RetornaMiembroPorMail("facupelu26@gmail.com"),
DateTime.Now, "Contenido post 4", Visibilidad.Publico, "imagenPost3.jpg", false));
    post4.AgregarComentario(
        new Comentario("Titulo comentario 1", RetornaMiembroPorMail("kevin.martin@gmail.com"),
DateTime.Now, "Contenido comentario 1 post 4", Visibilidad.Publico));
    post4.AgregarComentario(
        new Comentario("Titulo comentario 2",
RetornaMiembroPorMail("matheus.caique21@gmail.com"), DateTime.Now, "Contenido comentario 2 post
4", Visibilidad.Publico));
    post4.AgregarComentario(
        new Comentario("Titulo comentario 3",
RetornaMiembroPorMail("matheus.caique21@gmail.com"), DateTime.Now, "Contenido comentario 3 post
4", Visibilidad.Publico));
```

```
Post post5 = (new Post("Post 5", RetornaMiembroPorMail("facupelu26@gmail.com"),
DateTime.Now, "Contenido post 4", Visibilidad.Publico, "imagenPost3.jpg", false));
    post5.AgregarComentario(
        new Comentario("Titulo comentario 1", RetornaMiembroPorMail("kevin.martin@gmail.com"),
DateTime.Now, "Contenido comentario 1 post 5", Visibilidad.Publico));
    post5.AgregarComentario(
        new Comentario("Titulo comentario 2",
RetornaMiembroPorMail("matheus.caique21@gmail.com"), DateTime.Now, "Contenido comentario 2 post
5", Visibilidad.Publico));
    post5.AgregarComentario(
        new Comentario("Titulo comentario 3",
RetornaMiembroPorMail("matheus.caique21@gmail.com"), DateTime.Now, "Contenido comentario 3 post
5", Visibilidad.Publico));
    post5.AgregarComentario(
        new Comentario("Titulo comentario 5 ", RetornaMiembroPorMail("facupelu26@gmail.com"),
DateTime.Now, "Contenido comentario 2 post 2", Visibilidad.Publico));
    post5.AgregarComentario(
        new Comentario("Titulo comentario 5 ", RetornaMiembroPorMail("facupelu26@gmail.com"),
DateTime.Now, "Contenido comentario 2 post 2", Visibilidad.Publico));
```

```

public void PreCargaReaccionPost() // HAY QUE AGREGAR MAS
{
    Post post = RetornaPostPorId(1);
    Miembro miem =
    RetornaMiembroPorMail("matheus.caique21@gmail.com");if (!
    HayReaccionPost(1, "matheus.caique21@gmail.com"))
    {
        post.AgregarReaccionAPost(new Reaccion(ReaccionEnum.like, miem));
    }

    Post post2 = RetornaPostPorId(5);
    Miembro miem2 =
    RetornaMiembroPorMail("facupelu26@gmail.com");if (!
    HayReaccionPost(5, "facupelu26@gmail.com"))
    {
        post2.AgregarReaccionAPost(new Reaccion(ReaccionEnum.like, miem2));
    }
}

```

Vinculos

```

AceptarInvitacion(1,"facupelu26@gmail.com");
AceptarInvitacion(2, "johndoe@gmail.com");
AceptarInvitacion(3, "janedoe@gmail.com");
AceptarInvitacion(4, "alice.smith@gmail.com");
AceptarInvitacion(5, "susan.wilson@gmail.com");
AceptarInvitacion(6, "kevin.martin@gmail.com");
AceptarInvitacion(7, "lperez@gmail.com");
AceptarInvitacion(8, "bob.johnson@gmail.com");
AceptarInvitacion(9, "michael.jackson@gmail.com");
AceptarInvitacion(10,
"michael.jackson@gmail.com");AceptarInvitacion(11,
"johndoe@gmail.com"); AceptarInvitacion(12,
"janedoe@gmail.com"); AceptarInvitacion(13,
"alice.smith@gmail.com"); AceptarInvitacion(14,
"susan.wilson@gmail.com"); AceptarInvitacion(15,
"kevin.martin@gmail.com"); AceptarInvitacion(16,
"lperez@gmail.com"); AceptarInvitacion(17,
"bob.johnson@gmail.com");

//Rechazadas
RechazarInvitacion(18);
RechazarInvitacion(19);

```

```

public void PreCargaReaccionComentario()
{
    Comentario com = RetornaComentariosPorId(2);
    Miembro miem = RetornaMiembroPorMail("facupelu26@gmail.com");

    if (!HayReaccionComentario(2, "facupelu26@gmail.com"))
    {
        com.AgregarReaccionAComent(new Reaccion(ReaccionEnum.like, miem));
    }

    Comentario com2 = RetornaComentariosPorId(3);
    Miembro miem3 = RetornaMiembroPorMail("facupelu26@gmail.com");

    if (!HayReaccionComentario(3, "facupelu26@gmail.com"))
    {
        com2.AgregarReaccionAComent(new Reaccion(ReaccionEnum.like, miem3));
    }
}

```

Evidencia de testing para el alta de un miembro

Caso de prueba 1: Validación de tipos de datos ingresados consistentes.

Descripción: Ingresa datos válidos en todos los campos (nombre, apellido, fecha de nacimiento, email y contraseña) con tipos de datos incorrectos.

Datos de entrada: Nombre: 123, Apellido: 456, Fecha de Nacimiento: abc, Email: emailincorrecto, Contraseña: pass123

Pasos: Ingresa los datos en cada input y presiona ENTER.

Resultado esperado: Debe mostrar mensajes de error para datos incorrectos.

```

Terminal - Consola

Alta miembro
Ingrese el nombre:
123

*** No debe ingresar números, solo palabras ***

Ingrese el nombre:
Matheus
Ingrese el apellido:
456

*** No debe ingresar números, solo palabras ***

Ingrese el apellido:
Melo
Fecha de nacimiento
Ingresar el mes:
abc

*** Debe ingresar solo numeros ***

```

Caso de prueba 2: Validación de ingresos no vacíos.

Descripción: Deja campos en blanco y verifica si se validan correctamente.

Datos de entrada: Nombre: [vacío], Apellido: [vacío], Fecha de Nacimiento: [vacío], Email: [vacío], Contraseña: [vacío]

Pasos: Deja los campos en blanco y presiona ENTER.

Resultado esperado: Debe mostrar mensajes de error indicando que los campos no pueden estar vacíos.

```
Terminal - Consola

Alta miembro
Ingrese el nombre:

**** Debe ingresar una palabra válida ****

Ingrese el nombre:
Matheus
Ingrese el apellido:

**** Debe ingresar una palabra válida ****

Ingrese el apellido:
Melo
Fecha de nacimiento
Ingresar el mes:

**** Debe ingresar solo numeros ****

Ingresar el mes:
12
Ingresar el día:
08
Ingresar el año:

**** Debe ingresar solo numeros ****

Ingresar el año:
2023
Ingresar el mail:
matheus.melo2@gmail.com
Ingresar el password:
mat123

**** Miembro dado de alta ****
```

Caso de prueba 3: Validación de email no duplicado.

Descripción: Intenta registrar un miembro con un email que ya está en uso.

Datos de entrada: Nombre: Matheus, Apellido: Melo, Fecha de Nacimiento: 01/01/1990,
Email:matheus.caique21@gmail.com, Contraseña: password123

Pasos: Ingresa los datos en el formulario y presiona ENTER.

Resultado esperado: Debe mostrar un mensaje de error indicando que el email ya está registrado.

```
Terminal - Consola

Alta miembro

Ingrese el nombre:
Matheus
Ingrese el apellido:
Melo
Fecha de nacimiento
Ingresar el mes:
12
Ingresar el día:
08
Ingresar el año:
1994
Ingresar el mail:
matheus.caique21@gmail.com
Ingresar el password:
password123

*** Ya existe un miembro con este email! ***

=====
*** Menu de opciones ***
=====

Ingresar una opción
1 - Alta miembro
2 - Dado un email de miembro listar todas las publicaciones ...
3 - Dado un email de miembro, listar los posts ...
4 - Dadas dos fechas listar los posts realizados ...
5 - Obtener los miembros que hayan realizados más publicaciones ...
0 - Salir
```

Caso de prueba 4: Alta exitosa.

Descripción: Ingresa datos válidos en todos los campos y verifica si el miembro se da de alta correctamente. Datos de entrada: Nombre: María, Apellido: López, Fecha de Nacimiento: 15/05/1985, Email: nuevocorreo@example.com, Contraseña: contraseñasegura123

Pasos: Ingresa los datos en el formulario y presiona ENTER.

Resultado esperado: Debe mostrar un mensaje de éxito indicando que el miembro se ha registrado correctamente.

```
Terminal - Consola

Alta miembro

Ingrese el nombre:
Maria
Ingrese el apellido:
Lopez
Fecha de nacimiento
Ingresar el mes:
05
Ingresar el día:
15
Ingresar el año:
1985
Ingresar el mail:
nuevocorreo@example.com
Ingresar el password:
contrasena segura123

**** Miembro dado de alta ****

-----
**** Menu de opciones ****
-----

Ingresar una opción
1 - Alta miembro
2 - Dado un email de miembro listar todas las publicaciones ...
3 - Dado un email de miembro, listar los posts ...
4 - Dadas dos fechas listar los posts realizados ...
5 - Obtener los miembros que hayan realizado más publicaciones ...
0 - Salir
```

DEPLOY DEL SITIO EN SOMEЕ

URL: <http://www.obgp2.somee.com/>