Universidad ORT Uruguay
Facultad de Ingeniería
Escuela de Tecnología

Agustina Istebot - 203375
Bruno Benvenuto - 312195

Grupo: N2D
Docente: *Lucas López*

Analista en Tecnologías de la Información
Octubre 2024

LETRA OBLIGATORIO

Se nos ha asignado el desarrollo de una plataforma de venta de artículos que operará en todo el país. La empresa anuncia que tendrá un enorme catálogo de productos a la venta, y promete muchos beneficios a los usuarios que se registren en la aplicación.

En la plataforma conviven 2 tipos de usuarios, los Clientes y los Administradores.

De todos los usuarios se conoce **id (autonumérico), nombre, apellido, email y contraseña**. Pero de los clientes se conoce además el **saldo disponible en su billetera electrónica.**

Posee un catálogo de artículos y de cada uno se conoce un **id (autonumérico), el nombre, la categoría, y el precio de venta**.

El modelo de negocio de esta empresa consiste en realizar publicaciones de los artículos que pone a la venta. No los vende de forma individual, sino que se venden a través de estas publicaciones.

De cada publicación se conoce **un id (autonumérico), un nombre, el estado de la publicación** que puede ser **ABIERTA, CERRADA, CANCELADA**, **la fecha de publicación, y la lista de artículos que se venden en ella**.

Además, posee **el cliente que realizó la compra**, **el usuario que finalizó** la compra y **la fecha en que finalizó**, estos datos permanecen vacíos mientras se encuentre en estado ABIERTA.

Existen **dos tipos de publicaciones, las ventas y las subastas**. De las **ventas se sabe si está en "oferta relámpago"**, y de las subastas se conoce además una lista de ofertas que son las que realizan los usuarios para "ganar" la subasta.

De cada **oferta** se conoce **el id (autonumérico), usuario, el monto, y la fecha en que la realizó.**

Ejemplo de publicaciones:

Suponiendo que existen los artículos: Balde, sombrilla, protector solar y salvavidas, se ofrecen todos juntos en una publicación de nombre "Verano en la playa".

Y para los artículos: Bicicleta de carrera, malla y zapatilla de ciclismo, se ofrecen en una subasta de nombre "Vuelta ciclista", para la cual se realizarán una serie de ofertas y se le adjudicará al que haya hecho la oferta más alta al día que la empresa decida finalizar la subasta.

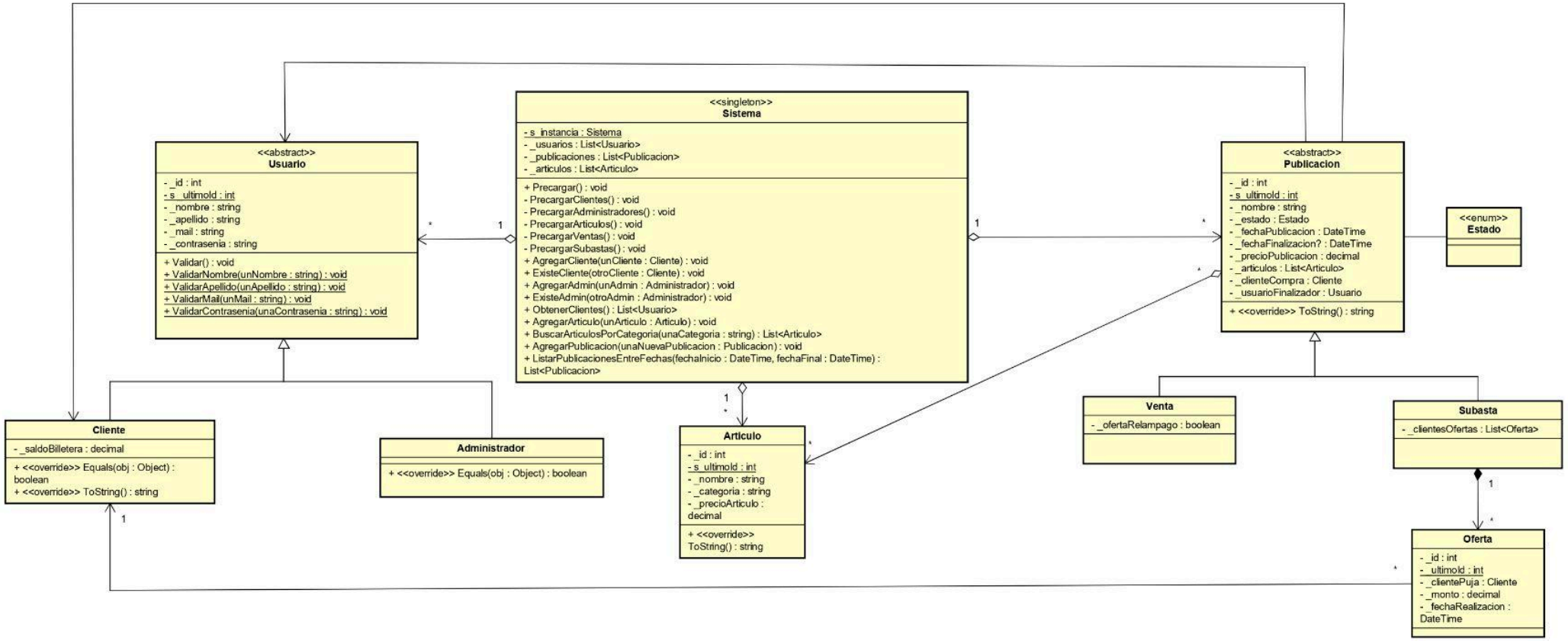
El precio de la publicación depende de su tipo. Para las ventas es la suma de los precios de los artículos, teniendo en cuenta que si está en “oferta relámpago” aplica un 20% de descuento.

El precio final de una subasta es siempre la mejor oferta realizada hasta el momento.

Los clientes podrán comprar cualquier publicación de tipo venta, sin embargo, a las subastas solamente podrán realizar una oferta, y será un administrador quien pueda cerrarla disparando la lógica de adjudicación de la compra al usuario que hizo la mejor oferta.

ÍNDICE

DIAGRAMA UML



CÓDIGO COMENTADO

Acceso a Repositorio GitHub https://github.com/BrunoBenvenuto/ObligatorioP2.git

Administrador.cs

```
namespace LogicaNegocio
{
    public class Administrador : Usuario
    {
        public Administrador() : base() { }

        public Administrador(
            string unNombre,
            string unApellido,
            string unMail,
            string unaContrasenia
            ) : base(unNombre, unApellido, unMail, unaContrasenia) { }

        public override bool Equals(object? obj)
        {
            if (obj == null || !(obj is Administrador))
            {
                return false;
            }
            Administrador otroAdmin = obj as Administrador;
            return otroAdmin.Id == this.Id;
        }
    }
}
```

Articulo.cs

```
namespace LogicaNegocio
{
   public class Articulo
   {
      /* Atributo estatico */
      private static int s_ultimoId = 0;

      /* Atributos */
      private int _id;
      private string _nombre;
      private string _categoria;
      private decimal _precioArticulo;

      /* Métodos */
      public int Id
      {
         get { return _id; }
      }

      public string Nombre
      {
         get { return _nombre; }
         set { _nombre = value; }
      }

      public string Categoria
      {
         get { return _categoria; }
         set { _categoria = value; }
      }

      public decimal PrecioArticulo
      {
         get { return _precioArticulo; }
         set { _precioArticulo = value; }
      }

      /* Constructores */
      public Articulo()
      {
         this._id = Articulo.s_ultimoId++;
      }

      public Articulo(string unNombre, string unaCategoria, decimal unPrecioArticulo)
```

```
        {
            this._id = Articulo.s_ultimoId++;
            this._nombre = unNombre;
            this._categoria = unaCategoria;
            this._precioArticulo = unPrecioArticulo;
        }

        /*---------------------- Metodo ToString ----------------------*/
        public override string ToString()
        {
            return " - Id: " + this.Id +
                "\n - Nombre: " + this.Nombre +
                "\n - Categoria: " + this.Categoria +
                "\n - Precio: " + this.PrecioArticulo;
        }
    }
}
```

Cliente.cs

```
namespace LogicaNegocio
{
   public class Cliente : Usuario
   {
      /*Atributos*/
      private decimal _saldoBilletera;

      /*Propiedades*/
      public decimal SaldoBilletera
      {
         get { return _saldoBilletera; }
         set { _saldoBilletera = value; }
      }

      /*Constructores*/
      public Cliente() : base() { }

      public Cliente(
         string unNombre,
         string unApellido,
         string unMail,
         string unaContrasenia,
         decimal unSaldo) : base(unNombre, unApellido, unMail, unaContrasenia)
      {
         this._saldoBilletera = unSaldo;
      }

      public override bool Equals(object? obj)
      {
         if(obj == null || !(obj is Cliente))
         {
            return false;
         }
         Cliente otroCliente = obj as Cliente;
         return otroCliente.Id == this.Id;
      }

      /*----------------------- Metodo ToString ----------------------*/
      public override string ToString()
      {
         return " - Id: " + this.Id +
            "\n - Nombre: " + this.Nombre +
            "\n - Email: " + this.Mail +
            "\n - Saldo: " + this.SaldoBilletera;
```

```
        }
    }
}
```

Estado.cs

```
namespace LogicaNegocio
{
    public enum Estado
    {
        Abierta = 0,
        Cerrada = 1,
        Cancelada = 2
    }
}
```

Oferta.cs

```
namespace LogicaNegocio
{
    public class Oferta
    {
        // Atributo estatico
        private static int s_ultimoId = 0;

        // Atributos
        private int _id;
        private decimal _monto;
        private Cliente _clientePuja;
        private DateTime _fechaRealizacion;

        // Propiedades
        public int Id
        {
            get { return _id; }
        }

        public decimal Monto
        {
            get { return _monto; }
            set { _monto = value; }
        }

        public DateTime FechaRealizacion
```

```
        {
            get { return _fechaRealizacion; }
            set { _fechaRealizacion = value; }
        }

        public Cliente ClientePuja
        {
            get { return _clientePuja; }
            set { _clientePuja = value; }
        }

        /********************** CONSTRUCTORES *******************************/
        public Oferta()
        {
            this._id = Oferta.s_ultimoId++;
        }

        public Oferta(decimal unMonto, DateTime unaFechaRealizacion, Cliente
unClientePuja)
        {
            this._id = Oferta.s_ultimoId++;
            this._monto = unMonto;
            this._fechaRealizacion = unaFechaRealizacion;
            this._clientePuja = unClientePuja;
        }
    }
}
```

Publicacion.cs

```
namespace LogicaNegocio
{
   public abstract class Publicacion
   {
      /* Atributo estatico */
      private static int s_ultimoId = 0;

      /* Atributos */
      private int _id;
      private string _nombre;
      private Estado _estado;
      private DateTime _fechaPublicacion;
      private DateTime? _fechaFinalizacion;
      private decimal _precioPublicacion;
      private List<Articulo> _articulos;
      private Cliente? _clienteCompra;
      private Usuario? _usuarioFinalizador;


      /* Propiedades */
      public int Id
      {
         get { return _id; }
      }

      public string Nombre
      {
         get { return _nombre; }
         set { _nombre = value; }
      }

      public Estado Estado
      {
         get { return _estado; }
         set { _estado = value; }
      }

      public DateTime FechaPublicacion
      {
         get { return _fechaPublicacion; }
         set { _fechaPublicacion = value; }
      }

      public DateTime? FechaFinalizacion
```

```
{
   get { return _fechaFinalizacion; }
   set { _fechaFinalizacion = value; }
}

public decimal PrecioPublicacion
{
   get { return _precioPublicacion; }
   set { _precioPublicacion = value; }
}

public List<Articulo> Articulos
{
   get { return _articulos; }
}

public Cliente? ClienteCompra
{
   get { return _clienteCompra; }
   set { _clienteCompra = value;}
}

public Usuario? UsuarioFinalizador
{
   get { return _usuarioFinalizador; }
   set { _usuarioFinalizador = value; }
}


/* Constructores */

public Publicacion()
{
   this._id = Publicacion.s_ultimoId++;
}

public Publicacion(string unNombre,
                   Estado unEstado,
                   DateTime unaFechaPub,
                   DateTime? unaFechaFin,
                   decimal unPrecioPub,
                   List<Articulo> articulos,
                   Cliente? clienteCompra,
                   Usuario? unUsuarioFinalizador)
{
   this._id = Publicacion.s_ultimoId++;
```

```
            this._nombre = unNombre;
            this._estado = unEstado;
            this._fechaPublicacion = unaFechaPub;
            this._fechaFinalizacion = unaFechaFin;
            this._precioPublicacion = unPrecioPub;
            this._articulos = articulos;
            this._clienteCompra = clienteCompra;
            this._usuarioFinalizador = unUsuarioFinalizador;
        }

        /*---------------------- Metodo ToString ----------------------*/
        public override string ToString()
        {
            return " - Id: " + this.Id +
                "\n - Nombre: " + this.Nombre +
                "\n - Estado: " + this.Estado +
                "\n - Fecha Publicacion: " + this.FechaPublicacion;
        }
    }
}
```

Sistema.cs

```
namespace LogicaNegocio
{
   public class Sistema
   {
      /*SINGLETON*/
      private static Sistemas _instancia;

      /*Atributos*/
      private List<Usuario> _usuarios = new List<Usuario>();
      private List<Articulo> _articulos = new List<Articulo>();
      private List<Publicacion> _publicaciones = new List<Publicacion>();

      public static Sistema Instancia
      {
         get
         {
            if (s_instancia == null)
            {
               s_instancia = new Sistema();
            }
            return s_instancia;
         }
      }

      public List<Usuario> Usuarios { get { return _usuarios; } }
      public List<Articulo> Articulos { get { return _articulos; } }
      public List<Publicacion> Publicaciones { get { return _publicaciones; } }

      private Sistema()
      {
         this.Precargar();
      }

      public void Precargar()
      {
         this.PrecargarClientes();
         this.PrecargarAdministradores();
         this.PrecargarArticulos();
         this.PrecargarVentas();
         this.PrecargarSubastas();
      }

      /*------------------------Precarga Usuarios------------------------*/
      private void PrecargarClientes()
```

```
        {
            List<Cliente> clientesAux = new List<Cliente>
            {
             new Cliente(
                "Bruno", "Benvenuto", "brunob@test.com", "1234bruno", 123),
            new Cliente(
                "Agustina", "Istebot", "agu@test.com", "1234agu", 1234),
            new Cliente(
                "María", "García López", "maria.garcia@email.com", "password1", 1250),
            new Cliente(
               "Juan", "Rodríguez Fernández", "juan.rodriguez@email.com", "password2", 8500),
            new Cliente(
                 "Sofía", "Martínez Gómez", "sofia.martinez@email.com", "password3", 1450),
            new Cliente(
                  "Pedro", "Sánchez Herrera", "pedro.sanchez@email.com", "password4", 620),
            new Cliente(
                  "Laura", "Fernández Ruiz", "laura.fernandez@email.com", "password5", 920),
            new Cliente(
                  "Andrés", "Gutiérrez Díaz", "andres.gutierrez@email.com", "password6", 1130),
            new Cliente(
                       "Carmen", "Torres Moreno", "carmen.torres@email.com", "password7", 785),
            new Cliente(
                     "Alejandro", "Mendoza Vargas", "alejandro.mendoza@email.com",
"password10", 990)
        };

            foreach (Cliente c in clientesAux)
            {
                this.AgregarCliente(c);
            }

        }

        /*------------------------Precarga Administradores------------------------*/
        private void PrecargarAdministradores()
        {
            Administrador administrador1 = new Administrador(
                "Jose", "Rodriguez", "jose@gmail.com", "1234jose");
            Administrador administrador2 = new Administrador(
                "Mauro", "Daverio", "mdaverio@tremendomail.com", "ItsMeMario");

            this.AgregarAdmin(administrador1);
            this.AgregarAdmin(administrador2);
        }

        /*------------------------Precarga Articulos------------------------*/
```

```
private void PrecargarArticulos()
{
    List<Articulo> articulosAux = new List<Articulo>
    {
        new Articulo("Bicicleta de montaña", "Deportes", 1200.99m),
        new Articulo("Patineta", "Deportes", 85.50m),
        new Articulo("Balón de fútbol", "Deportes", 25.75m),
        new Articulo("Raqueta de tenis", "Deportes", 145.99m),
        new Articulo("Camiseta de running", "Ropa", 35.90m),
        new Articulo("Zapatillas deportivas", "Ropa", 79.99m),
        new Articulo("Pantalones cortos", "Ropa", 29.50m),
        new Articulo("Gorra de béisbol", "Accesorios", 19.99m),
        new Articulo("Taza", "Hogar", 27.50m),
        new Articulo("Gafas de sol", "Accesorios", 49.99m),
        new Articulo("Bicicleta de carrera", "Ciclismo", 1500.00m),
        new Articulo("Casco de ciclismo", "Ciclismo", 89.99m),
        new Articulo("Guantes de ciclismo", "Ciclismo", 25.49m),
        new Articulo("Bomba de aire", "Ciclismo", 15.75m),
        new Articulo("Botella de agua", "Accesorios", 12.50m),
        new Articulo("Mochila deportiva", "Accesorios", 45.00m),
        new Articulo("Kit de herramientas", "Automotriz", 75.99m),
        new Articulo("Linterna LED", "Hogar", 22.90m),
        new Articulo("Cargador portátil", "Electrónica", 35.50m),
        new Articulo("Auriculares Bluetooth", "Electrónica", 59.99m),
        new Articulo("Teclado mecánico", "Electrónica", 89.50m),
        new Articulo("Ratón inalámbrico", "Electrónica", 25.99m),
        new Articulo("Monitor LED", "Electrónica", 199.99m),
        new Articulo("Silla de oficina", "Muebles", 149.99m),
        new Articulo("Escritorio", "Muebles", 249.50m),
        new Articulo("Estantería", "Muebles", 99.75m),
        new Articulo("Lámpara de escritorio", "Hogar", 29.99m),
        new Articulo("Cafetera", "Hogar", 79.99m),
        new Articulo("Tetera eléctrica", "Hogar", 45.50m),
        new Articulo("Reloj de pared", "Decoración", 35.00m),
        new Articulo("Alfombra", "Decoración", 89.99m),
        new Articulo("Cojín decorativo", "Decoración", 19.50m),
        new Articulo("Espejo de pared", "Decoración", 129.99m),
        new Articulo("Marco de fotos", "Decoración", 15.99m),
        new Articulo("Sofá de 3 plazas", "Muebles", 499.99m),
        new Articulo("Mesa de centro", "Muebles", 125.00m),
        new Articulo("Taburete", "Muebles", 49.50m),
        new Articulo("Ventilador de torre", "Electrodomésticos", 89.99m),
        new Articulo("Aspiradora", "Electrodomésticos", 159.99m),
        new Articulo("Plancha de vapor", "Electrodomésticos", 39.90m),
        new Articulo("Set de cuchillos", "Cocina", 25.99m),
```

```
            new Articulo("Sartenes", "Cocina", 55.75m),
            new Articulo("Olla a presión", "Cocina", 79.50m),
            new Articulo("Tupperware", "Cocina", 19.99m),
            new Articulo("Cubertería", "Cocina", 45.00m),
            new Articulo("Jarra de vidrio", "Cocina", 15.75m),
            new Articulo("Set de vasos", "Cocina", 20.99m),
            new Articulo("Toallas de baño", "Baño", 29.50m),
            new Articulo("Esponjas", "Baño", 9.99m),
            new Articulo("Alfombra de baño", "Baño", 19.50m)
        };

        foreach (Articulo a in articulosAux)
        {
            this.AgregarArticulo(a);
        }
    }


    /*------------------------Precarga Ventas------------------------*/
    private void PrecargarVentas()
    {
        // Crear diferentes listas de artículos utilizando los artículos ya cargados en el
sistema
        List<Articulo> articulos1 = new List<Articulo> { _articulos[0], _articulos[1] };
        List<Articulo> articulos2 = new List<Articulo> { _articulos[2], _articulos[3],
_articulos[4] };
        List<Articulo> articulos3 = new List<Articulo> { _articulos[5], _articulos[6] };
        List<Articulo> articulos4 = new List<Articulo> { _articulos[7] };
        List<Articulo> articulos5 = new List<Articulo> { _articulos[8], _articulos[9],
_articulos[10] };
        List<Articulo> articulos6 = new List<Articulo> { _articulos[11], _articulos[12] };
        List<Articulo> articulos7 = new List<Articulo> { _articulos[13], _articulos[14],
_articulos[15], _articulos[16] };
        List<Articulo> articulos8 = new List<Articulo> { _articulos[17], _articulos[18] };
        List<Articulo> articulos9 = new List<Articulo> { _articulos[19], _articulos[20],
_articulos[21] };
        List<Articulo> articulos10 = new List<Articulo> { _articulos[22], _articulos[23],
_articulos[24], _articulos[25] };

        // Crear 10 ventas y agregarlas al sistema utilizando el método AgregarPublicacion
        AgregarPublicacion(new Venta("Deportes Extremos", Estado.Abierta,
DateTime.Parse("08-03-2012"), null, 299.99m, articulos1, null, null, true));
        AgregarPublicacion(new Venta("Equipamiento Deportivo", Estado.Abierta,
DateTime.Parse("30-12-2021"), null, 199.99m, articulos2, null, null, false));
        AgregarPublicacion(new Venta("Aventura en la Montaña", Estado.Abierta,
DateTime.Parse("15-06-2010"), null, 349.99m, articulos3, null, null, true));
```

```
        AgregarPublicacion(new Venta("Ropa para Running", Estado.Abierta,
DateTime.Parse("03-03-2018"), null, 159.99m, articulos4, null, null, false));
        AgregarPublicacion(new Venta("Ciclismo y Más", Estado.Abierta,
DateTime.Parse("22-01-2020"), null, 299.99m, articulos5, null, null, true));
        AgregarPublicacion(new Venta("Verano Activo", Estado.Abierta,
DateTime.Parse("03-03-2015"), null, 249.99m, articulos6, null, null, false));
        AgregarPublicacion(new Venta("Accesorios para Deportes", Estado.Abierta,
DateTime.Parse("05-04-2017"), null, 189.99m, articulos7, null, null, true));
        AgregarPublicacion(new Venta("Entrenamiento al Aire Libre", Estado.Abierta,
DateTime.Parse("08-08-2020"), null, 139.99m, articulos8, null, null, false));
        AgregarPublicacion(new Venta("Fitness Total", Estado.Abierta,
DateTime.Parse("11-08-2023"), null, 219.99m, articulos9, null, null, true));
        AgregarPublicacion(new Venta("Ciclismo Profesional", Estado.Abierta,
DateTime.Parse("11-01-2019"), null, 389.99m, articulos10, null, null, false));
    }

    /*-----------------------Precarga Subastas ----------------------------*/
    private void PrecargarSubastas()
    {
        // Filtrar solo los clientes de la lista de usuarios del sistema
        List<Cliente> clientes = _usuarios.OfType<Cliente>().ToList();

        // Crear diferentes listas de artículos utilizando los artículos ya cargados en el
sistema
        List<Articulo> articulos1 = new List<Articulo> { _articulos[26], _articulos[27],
_articulos[28] };
        List<Articulo> articulos2 = new List<Articulo> { _articulos[29], _articulos[30] };
        List<Articulo> articulos3 = new List<Articulo> { _articulos[31], _articulos[32],
_articulos[33], _articulos[34] };
        List<Articulo> articulos4 = new List<Articulo> { _articulos[35] };
        List<Articulo> articulos5 = new List<Articulo> { _articulos[36], _articulos[37] };
        List<Articulo> articulos6 = new List<Articulo> { _articulos[38], _articulos[39],
_articulos[40] };
        List<Articulo> articulos7 = new List<Articulo> { _articulos[41], _articulos[42] };
        List<Articulo> articulos8 = new List<Articulo> { _articulos[43], _articulos[44],
_articulos[45] };
        List<Articulo> articulos9 = new List<Articulo> { _articulos[46] };
        List<Articulo> articulos10 = new List<Articulo> { _articulos[47], _articulos[48],
_articulos[49] };

        // Crear listas de ofertas
        List<Oferta> ofertasSubasta1 = new List<Oferta>
        {
            new Oferta(150m, DateTime.Parse("04-11-2021"), clientes[0]),
            new Oferta(200m, DateTime.Parse("23-01-2024"), clientes[1])
        };
```

```
        List<Oferta> ofertasSubasta2 = new List<Oferta>
        {
            new Oferta(300m, DateTime.Parse("29-06-2019"), clientes[2]),
            new Oferta(350m, DateTime.Parse("13-08-2020"), clientes[3]),
            new Oferta(500m, DateTime.Parse("15-08-2020"), clientes[2])
        };

        // Crear 10 subastas y agregarlas al sistema utilizando el método AgregarPublicacion
        AgregarPublicacion(new Subasta("Vuelta Ciclista", Estado.Abierta,
DateTime.Parse("02-11-2021"), null, 500m, articulos1, null, ofertasSubasta1, null));
        AgregarPublicacion(new Subasta("Equipo de Camping", Estado.Abierta,
DateTime.Parse("23-02-2019"), null, 250m, articulos2, null, ofertasSubasta2, null));
        AgregarPublicacion(new Subasta("Caza y Pesca", Estado.Abierta,
DateTime.Parse("03-02-2023"), null, 600m, articulos3, null, null, null));
        AgregarPublicacion(new Subasta("Ropa de Montaña", Estado.Abierta,
DateTime.Parse("18-03-2010"), null, 350m, articulos4, null, new List<Oferta>(), null));
        AgregarPublicacion(new Subasta("Equipamiento de Escalada", Estado.Abierta,
DateTime.Parse("02-12-2022"), null, 450m, articulos5, null, new List<Oferta>(), null));
        AgregarPublicacion(new Subasta("Set de Viaje", Estado.Abierta,
DateTime.Parse("03-09-2016"), null, 200m, articulos6, null, new List<Oferta>(), null));
        AgregarPublicacion(new Subasta("Camping de Lujo", Estado.Abierta,
DateTime.Parse("12-08-2021"), null, 700m, articulos7, null, new List<Oferta>(), null));
        AgregarPublicacion(new Subasta("Aventura Extrema", Estado.Abierta,
DateTime.Parse("27-08-2017"), null, 800m, articulos8, null, new List<Oferta>(), null));
        AgregarPublicacion(new Subasta("Mountain Bike", Estado.Abierta,
DateTime.Parse("23-04-2015"), null, 900m, articulos9, null, new List<Oferta>(), null));
        AgregarPublicacion(new Subasta("Ropa para Esquí", Estado.Abierta,
DateTime.Parse("15-05-2022"), null, 550m, articulos10, null, new List<Oferta>(), null));
    }


    /*------------------------Metodos para clientes-------------------------*/
    public void AgregarCliente(Cliente unCliente)
    {
        try
        {
            this.ExisteCliente(unCliente);
            this._usuarios.Add(unCliente);
        }
        catch (Exception e)
        {
            throw new Exception(e.Message);
        }
    }
```

```
public void ExisteCliente(Cliente otroCliente)
{
   if (this._usuarios.Contains(otroCliente))
   {
      throw new Exception("El usuario ya existe.");
   }
}

/*------------------------Metodos para administradores------------------------*/
public void AgregarAdmin(Administrador unAdmin)
{
   try
   {
      this.ExisteAdmin(unAdmin);
      this._usuarios.Add(unAdmin);
   }
   catch (Exception e)
   {
      throw new Exception(e.Message);
   }
}

public void ExisteAdmin(Administrador otroAdmin)
{
   if (this._usuarios.Contains(otroAdmin))
   {
      throw new Exception("El usuario ya existe.");
   }
}

/*------------------------Lista de Clientes------------------------*/
public List<Usuario> ObtenerClientes()
{
   List<Usuario> aRetornar = new List<Usuario>();
   foreach (Usuario u in this._usuarios)
   {
      if (u is Cliente)
      {
         aRetornar.Add(u);
      }
   }
   return aRetornar;
}

/*------------------------Metodo para Articulos------------------------*/
```

```
    public void AgregarArticulo(Articulo unArticulo)
    {
        try
        {
            this._articulos.Add(unArticulo);
        }
        catch (Exception e)
        {
            throw new Exception(e.Message);
        }
    }

    public List<Articulo> BuscarArticulosPorCategoria(string unaCategoria)
    {
        List<Articulo> aRetornar = new List<Articulo>();
        foreach (Articulo unArticulo in this.Articulos)
        {
            if (unArticulo.Categoria.ToLower() == unaCategoria.ToLower())
            {
                aRetornar.Add(unArticulo);
            }
        }
        if (aRetornar.Count == 0) throw new Exception("No existen publicaciones con ese
genero");
        return aRetornar;
    }

    /*------------------------Metodo para Publicaciones------------------------*/
    public void AgregarPublicacion(Publicacion unaNuevaPublicacion)
    {
        try
        {
            this._publicaciones.Add(unaNuevaPublicacion);
        }
        catch (Exception e)
        {
            throw new Exception(e.Message);
        }
    }

    public List<Publicacion> ListarPublicacionesEntreFechas(DateTime fechaInicio,
DateTime fechaFinal)
    {
        List<Publicacion> listaPublicacionesFiltradas = new List<Publicacion>();
        foreach (Publicacion p in this._publicaciones)
        {
```

```
            if (p.FechaPublicacion >= fechaInicio && p.FechaPublicacion <= fechaFinal)
            {
                listaPublicacionesFiltradas.Add(p);

            }
        }
        return listaPublicacionesFiltradas;
    }
  }
}
```

Subasta.cs

```
namespace LogicaNegocio
{
   public class Subasta : Publicacion
   {
      //Atributo
      private List<Oferta> _clientesOfertas;

      //Propiedad
      public List<Oferta> ClientesOfertas
      {
         get { return _clientesOfertas; }
      }

      //Constructores
      public Subasta() : base() { }

      public Subasta(
         string unNombre,
         Estado unEstado,
         DateTime unaFechaPub,
         DateTime? unaFechaFin,
         decimal unPrecioPub,
         List<Articulo> articulos,
         Cliente? clienteCompra,
         List<Oferta> clientesOfertas,
         Usuario? unUsuarioFinalizador
         ) : base(unNombre, unEstado, unaFechaPub, unaFechaFin, unPrecioPub, articulos,
clienteCompra, unUsuarioFinalizador)
      {
         this._clientesOfertas = clientesOfertas;
      }
   }
}
```

Usuario.cs

```
namespace LogicaNegocio
{
   public abstract class Usuario
   {
      /* Atributo estatico */
      private static int s_ultimoId = 0;

      /* Atributos */
      private int _id;
      private string _nombre;
      private string _apellido;
      private string _mail;
      private string _contrasenia;


      /* Propiedades */
      public int Id
      {
         get { return _id; }
      }

      public string Nombre
      {
         get { return _nombre; }
         set { _nombre = value; }
      }

      public string Apellido
      {
         get { return _apellido; }
         set { _apellido = value; }
      }

      public string Mail
      {
         get { return _mail; }
         set { _mail = value; }
      }

      public string Contrasenia
      {
         get { return _contrasenia; }
         set { _contrasenia = value;}
      }
```

```
    /* Constructores */

    public Usuario()
    {
        this._id = Usuario.s_ultimoId++;
    }

    public Usuario(string unNombre, string unApellido, string unMail, string
unaContrasenia)
    {
        this._id = Usuario.s_ultimoId++;
        this._nombre = unNombre;
        this._apellido = unApellido;
        this._mail = unMail;
        this._contrasenia = unaContrasenia;
        this.Validar();
    }

    /*Validaciones*/
    public void Validar()
    {
        Usuario.ValidarNombre(this._nombre);
        Usuario.ValidarApellido(this._apellido);
        Usuario.ValidarMail(this._mail);
        Usuario.ValidarContrasenia(this._contrasenia);
    }

    public static void ValidarNombre(string unNombre)
    {
        if(unNombre.Length < 3)
        {
            throw new Exception("Nombre incorrecto.");
        }
    }

    public static void ValidarApellido(string unApellido)
    {
        if (unApellido.Length < 3)
        {
            throw new Exception("Apellido incorrecto.");
        }
    }

    public static void ValidarMail(string unMail)
    {
```

```
            bool contieneArroba = false;
            foreach(char c in unMail)
            {
                if(!contieneArroba && c == '@')
                {
                    contieneArroba = true;
                    break;
                }
            }
            if (!contieneArroba)
            {
                throw new Exception("Mail incorrecto.");
            }
        }

        public static void ValidarContrasenia(string unaContrasenia)
        {
            if (unaContrasenia.Length < 7)
            {
                throw new Exception("Contraseña incorrecta.");
            }
        }

    }
}
```

Venta.cs

```
namespace LogicaNegocio
{
   public class Venta : Publicacion
   {
      // Atributos
      private bool _ofertaRelampago;

      public bool OfertaRelampago
      {
         get { return _ofertaRelampago; }
         set { _ofertaRelampago = value; }
      }

      /*Constructores*/
      public Venta() : base() { }

      public Venta(
         string unNombre,
         Estado unEstado,
         DateTime unaFechaPub,
         DateTime? unaFechaFin,
         decimal unPrecioPub,
         List<Articulo> articulos,
         Cliente? clienteCompra,
         Usuario? unUsuarioFinalizador,
         bool unaOferta) : base(unNombre, unEstado, unaFechaPub, unaFechaFin,
unPrecioPub, articulos, clienteCompra, unUsuarioFinalizador)
      {
         this._ofertaRelampago = unaOferta;
      }

   }
}
```

Program.cs

```
using LogicaNegocio;

namespace Obligatorio
{
   internal class Program
   {
      static void Main(string[] args)
      {
         Sistema sistema = Sistema.Instancia;
         string seleccion = "";
         while (seleccion != "0")
         {
            Console.Clear();
            MostrarMenu();
            seleccion = Console.ReadLine();
            switch(seleccion)
            {
               case "0":
                  Console.Clear();
                  Console.WriteLine("Adios Profe! No se olvide poner buena nota :)");
                  break;
               case "1":
                  Console.Clear();
                  Console.WriteLine("Listado de Clientes:\n");
                  MostrarClientes();
                  break;
               case "2":
                  Console.Clear();
                  Console.WriteLine("Dado un nombre de categoría listar todos los artículos de
esa categoría");
                  MostrarCategorias();
                  break;
               case "3":
                  Console.Clear();
                  Console.WriteLine("Alta de Articulo: ");
                  bool seIngreso = false;
                  while (!seIngreso)
                  {
                     try
                     {
                        Articulo articulo = new Articulo();
                        Console.WriteLine("Ingrese el nombre: ");
                        string nombre = Console.ReadLine();
                        articulo.Nombre = nombre;
```

```
                Console.WriteLine("Ingrese la categoria: ");
                string categoria = Console.ReadLine();
                articulo.Categoria = categoria;
                Console.WriteLine("Ingrese el precio: ");
                decimal precioArticulo = decimal.Parse(Console.ReadLine());
                articulo.PrecioArticulo = precioArticulo;

                sistema.AgregarArticulo(articulo);
                seIngreso = true;
                Console.Clear();
                Console.WriteLine("Se ingresó el artículo: \n" + articulo);
                Console.WriteLine("Presione enter para volver al menu.");
                Console.ReadLine();
            }
            catch (Exception ex)
            {
                Console.WriteLine(ex.Message);
            }
        }
        break;
    case "4":
        Console.Clear();
        Console.WriteLine("Publicaciones entre dos fechas.");
        MostrarPublicaciones();
        break;
    default:
        Console.Clear();
        Console.WriteLine("Opción inválida. Presione enter para volver al menu.");
        Console.ReadLine();
        break;
  }
 }
}

static void MostrarClientes()
{
  Sistema sistema = Sistema.Instancia;

  if (sistema.ObtenerClientes().Count == 0)
  {
    Console.WriteLine("No existen Clientes");
  }
  else
  {
    foreach (Usuario unUsuario in sistema.ObtenerClientes())
    {
```

```
                Console.WriteLine(unUsuario);
                Console.WriteLine("----------------------------");
            }
        }
        Console.ReadLine();
    }

    static void MostrarMenu()
    {
        Console.WriteLine("Obligatorio");
        Console.WriteLine("1 - Ver listado de clientes");
        Console.WriteLine("2 - Ver artículos de una categoría");
        Console.WriteLine("3 - Alta de artículo");
        Console.WriteLine("4 - Publicaciones existentes en fechas especificas");
        Console.WriteLine("0 - Salir");
    }

    static void MostrarCategorias()
    {
        Sistema sistema = Sistema.Instancia;
        List<string> categoriasUnicas = new List<string>();

        Console.WriteLine("Lista de Categorias:");
        foreach (Articulo unArticulo in sistema.Articulos)
        {
            if (!categoriasUnicas.Contains(unArticulo.Categoria))
            {
                categoriasUnicas.Add(unArticulo.Categoria);
                Console.WriteLine("- " + unArticulo.Categoria);
            }
        }

        Console.WriteLine("---------------------------------");
        Console.WriteLine("Escriba el nombre de la categoria que desea filtrar:");
        string categoriaABuscar = Console.ReadLine();
        Console.Clear();
        Console.WriteLine("Resultado de la busqueda: " + categoriaABuscar + "\n");

        try {
            foreach (Articulo unArticulo in
sistema.BuscarArticulosPorCategoria(categoriaABuscar))
            {
                Console.WriteLine(unArticulo);
                Console.WriteLine("----------------------");
            }
            Console.WriteLine("\n\nPreciona cualquier tecla para volver al menu");
```

```
            Console.ReadLine();
        }
        catch (Exception e) {
            Console.WriteLine("Error Categoria invalida... Presione cualquier letra para volver
al menu");
            Console.ReadLine();
        }

    }

    static void MostrarPublicaciones() {
        Sistema sistema = Sistema.Instancia;
        bool seIngreso4 = false;
        while (!seIngreso4)
        {
            try
            {
                Console.WriteLine("Teniendo en cuenta el siguiente formato
dd-mm-aaaa\nIngrese la fecha de inicio:");
                string fechaIngresada1 = Console.ReadLine();
                DateTime fechaInicio = DateTime.Parse(fechaIngresada1);

                Console.WriteLine("Ingrese la fecha de finalizacion:");
                string fechaIngresada2 = Console.ReadLine();
                DateTime fechaFinal = DateTime.Parse(fechaIngresada2);

                Console.WriteLine("\n\nLas fechas ingresada son:\nFecha Inicial: " + fechaInicio +
"\nFecha Final: " + fechaFinal);
                Console.WriteLine("\n----------\nSi la solicitud es correcta escriba SI");
                string respuesta = Console.ReadLine();
                if (respuesta.ToLower() == "si")
                {
                    seIngreso4 = true;
                }
                try
                {
                    List<Publicacion> lista = sistema.ListarPublicacionesEntreFechas(fechaInicio,
fechaFinal);

                    if (lista.Count == 0)
                    {
                        Console.WriteLine("\nNo se encontraron resultados\nVuelve a intentarlo
con nuevas fechas...");
                    }
                    else
                    {
```

```
                    foreach (Publicacion unaPublicacion in lista)
                    {
                        Console.WriteLine(unaPublicacion);
                        Console.WriteLine("----------------------");
                    }
                        Console.WriteLine("\n\nPreciona cualquier tecla para volver al menu");
                }
                Console.ReadLine();
            }
            catch (Exception e)
            {
                Console.WriteLine("Error Categoria invalida... Presione cualquier letra para
volver al menu");
                Console.ReadLine();
            }
        }
        catch (Exception ex)
        {
            Console.WriteLine(ex.Message);
        }
    }
  }
}
```

TABLA DE PRECARGA

Usuarios

Clientes

Nombre	Apellido	Email	Password	Saldo
Bruno	Benvenuto	brunob@test.com	1234bruno	123
Agustina	Istebot	agu@test.com	1234agu	1234
María	García López	maria.garcia@email.com	password1	1250
Juan	Rodríguez Fernández	juan.rodriguez@email.com	password2	8500
Sofía	Martínez Gómez	sofia.martinez@email.com	password3	1450
Pedro	Sánchez Herrera	pedro.sanchez@email.com	password4	620
Laura	Fernández Ruiz	laura.fernandez@email.com	password5	920
Andrés	Gutiérrez Díaz	andres.gutierrez@email.com	password6	1130
Carmen	Torres Moreno	carmen.torres@email.com	password7	785
Alejandro	Mendoza Vargas	alejandro.mendoza@email.com	password10	990

Administradores

Nombre	Apellido	Email	Password
Jose	Rodriguez	jose@gmail.com	1234jose
Mauro	Daverio	mdaverio@tremendomail.com	ItsMeMario

Artículos

Nombre	Categoría	Precio
Bicicleta de montaña	Deportes	1200.99
Patineta	Deportes	85.50
Balón de fútbol	Deportes	25.75
Raqueta de tenis	Deportes	145.99
Camiseta de running	Ropa	35.90
Zapatillas deportivas	Ropa	79.99
Pantalones cortos	Ropa	29.50
Gorra de béisbol	Accesorios	19.99
Taza	Hogar	27.50
Gafas de sol	Accesorios	49.99
Bicicleta de carrera	Ciclismo	1500.00
Casco de ciclismo	Ciclismo	89.99
Guantes de ciclismo	Ciclismo	25.49
Bomba de aire	Ciclismo	15.75
Botella de agua	Accesorios	12.50
Mochila deportiva	Accesorios	45.00
Kit de herramientas	Automotriz	75.99
Linterna LED	Hogar	22.90
Cargador portátil	Electrónica	35.50
Auriculares Bluetooth	Electrónica	59.99
Teclado mecánico	Electrónica	89.50
Ratón inalámbrico	Electrónica	25.99
Monitor LED	Electrónica	199.99
Silla de oficina	Muebles	149.99

Escritorio	Muebles	249.50
Estantería	Muebles	99.75
Lámpara de escritorio	Hogar	29.99
Cafetera	Hogar	79.99
Tetera eléctrica	Hogar	45.50
Reloj de pared	Decoración	35.00
Alfombra	Decoración	89.99
Cojín decorativo	Decoración	19.50
Espejo de pared	Decoración	129.99
Marco de fotos	Decoración	15.99
Sofá de 3 plazas	Muebles	499.99
Mesa de centro	Muebles	125.00
Taburete	Muebles	49.50
Ventilador de torre	Electrodomésticos	89.99
Aspiradora	Electrodomésticos	159.99
Plancha de vapor	Electrodomésticos	39.90
Set de cuchillos	Cocina	25.99
Sartenes	Cocina	55.75
Olla a presión	Cocina	79.50
Tupperware	Cocina	19.99
Cubertería	Cocina	45.00
Jarra de vidrio	Cocina	15.75
Set de vasos	Cocina	20.99
Toallas de baño	Baño	29.50
Esponjas	Baño	9.99
Alfombra de baño	Baño	19.50

Publicaciones

Venta

Título	Estado	Fecha	Precio	Artículos	Destacado
Deportes Extremos	Abierta	08-03-2012	299.99	articulos1	Sí
Equipamiento Deportivo	Abierta	30-12-2021	199.99	articulos2	No
Aventura en la Montaña	Abierta	15-06-2010	349.99	articulos3	Sí
Ropa para Running	Abierta	03-03-2018	159.99	articulos4	No
Ciclismo y Más	Abierta	22-01-2020	299.99	articulos5	Sí
Verano Activo	Abierta	03-03-2015	249.99	articulos6	No
Accesorios para Deportes	Abierta	05-04-2017	189.99	articulos7	Sí
Entrenamiento al Aire Libre	Abierta	08-08-2020	139.99	articulos8	No
Fitness Total	Abierta	11-08-2023	219.99	articulos9	Sí
Ciclismo Profesional	Abierta	11-01-2019	389.99	articulos10	No

Subasta

Título	Estado	Fecha	Precio Base	Artículos	Ofertas
Vuelta Ciclista	Abierta	02-11-2021	500	articulos1	ofertasSubasta1
Equipo de Camping	Abierta	23-2-2019	250	articulos2	ofertasSubasta2
Caza y Pesca	Abierta	03-02-2023	600	articulos3	Ninguna
Ropa de Montaña	Abierta	18-03-2010	350	articulos4	Lista vacía
Equipamiento de Escalada	Abierta	02-12-2022	450	articulos5	Lista vacía
Set de Viaje	Abierta	03-09-2016	200	articulos6	Lista vacía
Camping de Lujo	Abierta	12-08-2021	700	articulos7	Lista vacía
Aventura Extrema	Abierta	27-08-2017	800	articulos8	Lista vacía
Mountain Bike	Abierta	23-04-2015	900	articulos9	Lista vacía
Ropa para Esquí	Abierta	15-05-2022	550	articulos10	Lista vacía

Oferta - ofertasSubasta1

Monto	Fecha	Cliente
150m	04-11-2021	clientes[0]
200m	23-01-2024	clientes[1]

Oferta - ofertasSubasta2

Monto	Fecha	Cliente
300m	29-06-2019	clientes[2]
350m	13-08-2020	clientes[3]
500m	15-08-2020	clientes[2]

FUNCIONALIDADES

- ➔ **F01** - Ver listado de Cliente
- ➔ **F02** - Ver artículos de una categoría
- ➔ **F03** - Alta de artículo
- ➔ **F04** - Publicaciones existentes en fechas específicas

BIBLIOGRAFÍA

Uso de ChatGPT

Link de consulta: ChatGPT - Precarga de datos en C#