

Universidad ORT Uruguay



Sebastián Hohl – 327007

M2E



Matías Oreiro – 239479

M2E

Prof. Liliana Pino

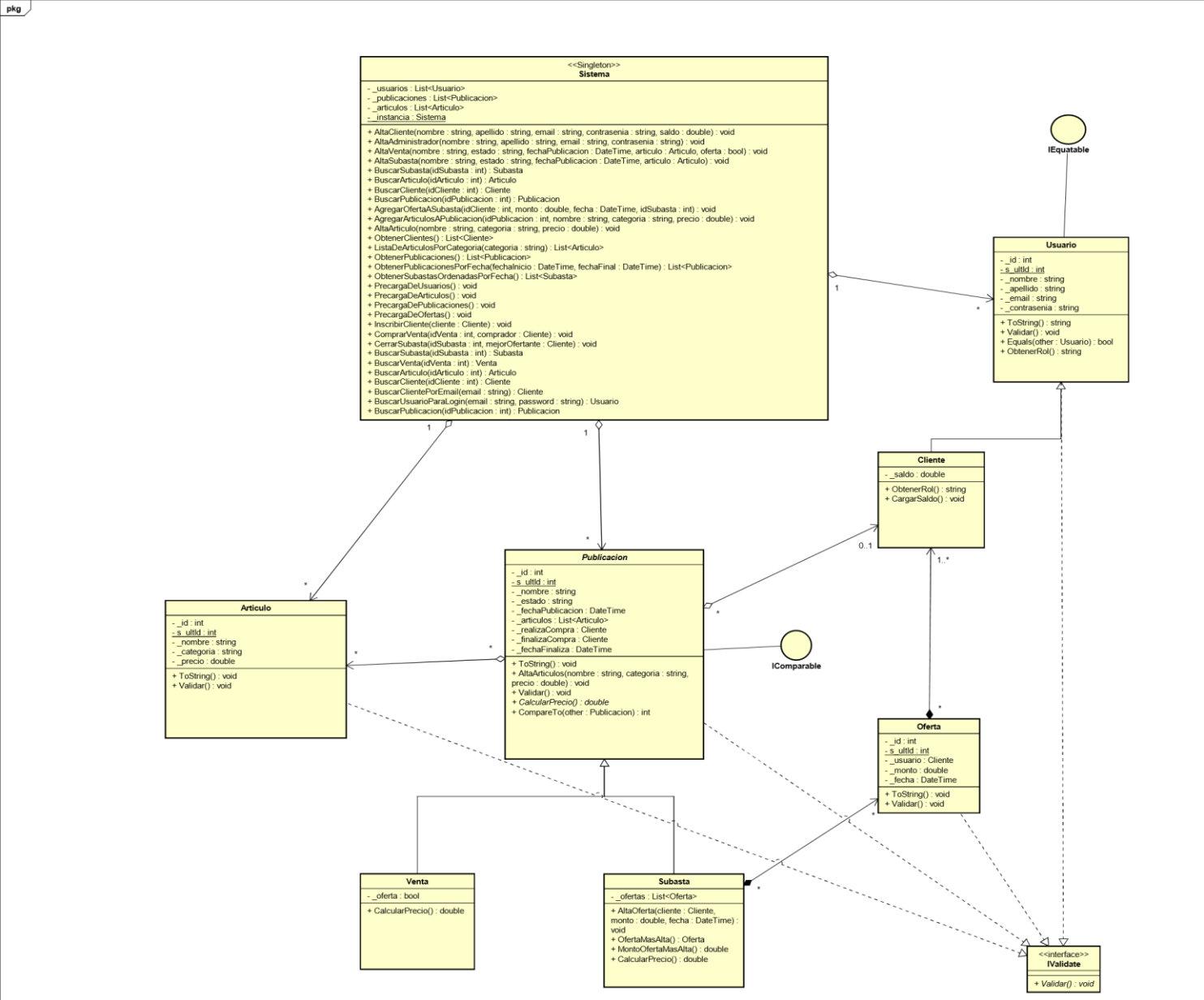
Programación 2

Grupo M2E

UML:

Class Diagram0

2024/11/28



Precargas:

Clientes:

Nombre	Apellido	Email	Contraseña
Juan	Pérez	juan.perez@example.com	contrasena123
Ana	Gómez	ana.gomez@example.com	contrasena456
Carlos	Fernández	carlos.fernandez@example.com	contrasena789
Laura	Martínez	laura.martinez@example.com	laura2023
Javier	López	javier.lopez@example.com	javier2023
Marta	Rodríguez	marta.rodriguez@example.com	marta1234

José	Torres	jose.torres@example.com	jose5678
Sofía	Díaz	sofia.diaz@example.com	sofia910
Luis	Morales	luis.morales@example.com	luis1111
Patricia	Jiménez	patricia.jimenez@example.com	patricia2222

Administradores:

Nombre	Apellido	Email	Contraseña
Ricardo	Álvarez	ricardo.alvarez@example.com	ricardo3333
Elena	Sánchez	elena.sanchez@example.com	elena4444

Artículos:

Nombre del artículo	Categoría	Precio (USD)
Vestido de verano	Ropa	85.60
Collar de perlas	Joyería	200.00
Botas de cuero	Calzado	120.90
Lámpara LED	Hogar	95.00
Bicicleta de montaña	Deportes	550.00
Crema hidratante	Belleza	70.00
Set de herramientas	Jardinería	45.20
Gorro de invierno	Accesorios	33.80
Laptop Ultra Book	Electrónica	1200.40
Pantalones jeans	Ropa	55.25
Pulsera de acero	Joyería	95.60
Sandalias veraniegas	Calzado	65.70
Juego de toallas	Hogar	150.90
Raqueta de tenis	Deportes	85.00
Maquillaje compacto	Belleza	45.50
Kit de jardinería	Jardinería	70.00

Bolso de mano	Accesorios	27.30
Altavoz Bluetooth	Electrónica	400.00
Chaqueta de invierno	Ropa	75.30
Aretes de oro	Joyería	150.00
Sneakers de moda	Calzado	90.40
Cuadro decorativo	Hogar	60.50
Set de pesas	Deportes	110.00
Gel exfoliante	Belleza	55.90
Fertilizante orgánico	Jardinería	40.00
Cinturón de cuero	Accesorios	22.90
Monitor 24"	Electrónica	300.00
Falda larga	Ropa	30.00
Broche de cristal	Joyería	85.60
Chanclas de playa	Calzado	95.80
Manta suave	Hogar	210.50
Kit de yoga	Deportes	75.10
Mascarilla facial	Belleza	35.00
Set de semillas	Jardinería	50.40
Reloj de pulsera	Accesorios	60.70
Proyector portátil	Electrónica	500.00
Pijama de seda	Ropa	45.00
Pulsera de cuero	Joyería	160.00
Botines de gamuza	Calzado	80.50
Mesa de café	Hogar	95.00
Smartphone X10	Electrónica	300.20

Camiseta de algodón	Ropa	25.50
Anillo de plata	Joyería	120.75
Zapatillas deportivas	Calzado	75.90

Publicaciones:

AltaVenta("Venta 1", "ABIERTA", new DateTime(2024, 10, 08),
 BuscarArticulo(3), false);

AltaVenta("Venta 2", "ABIERTA", new DateTime(2023, 05, 18),
 BuscarArticulo(7), true);

AltaVenta("Venta 3", "ABIERTA", new DateTime(2021, 01, 07),
 BuscarArticulo(21), false);

AltaVenta("Venta 4", "ABIERTA", new DateTime(2022, 09, 09),
 BuscarArticulo(17), false);

AltaVenta("Venta 5", "ABIERTA", new DateTime(2023, 12, 03),
 BuscarArticulo(41), false);

AltaVenta("Venta 6", "ABIERTA", new DateTime(2022, 02, 01),
 BuscarArticulo(44), false);

AltaVenta("Venta 7", "ABIERTA", new DateTime(2023, 04, 29),
 BuscarArticulo(7), false);

AltaVenta("Venta 8", "ABIERTA", new DateTime(2021, 01, 23),
 BuscarArticulo(2), true);

AltaVenta("Venta 9", "ABIERTA", new DateTime(2020, 04, 28),
 BuscarArticulo(13), false);

AltaVenta("Venta 10", "ABIERTA", new DateTime(2019, 09, 18),
 BuscarArticulo(33), false);

AltaSubasta("Subasta 1", "ABIERTA", new DateTime(2017, 07, 07),
 BuscarArticulo(5));

AltaSubasta("Subasta 2", "ABIERTA", new DateTime(2020, 11, 07),
 BuscarArticulo(45));

AltaSubasta("Subasta 3", "ABIERTA", new DateTime(2023, 06, 06),
BuscarArticulo(14));

AltaSubasta("Subasta 4", "ABIERTA", new DateTime(2024, 02, 08),
BuscarArticulo(22));

AltaSubasta("Subasta 5", "ABIERTA", new DateTime(2024, 11, 09),
BuscarArticulo(48));

AltaSubasta("Subasta 6", "ABIERTA", new DateTime(2023, 12, 11),
BuscarArticulo(37));

AltaSubasta("Subasta 7", "ABIERTA", new DateTime(2024, 02, 01),
BuscarArticulo(32));

AltaSubasta("Subasta 8", "ABIERTA", new DateTime(2022, 03, 22),
BuscarArticulo(12));

AltaSubasta("Subasta 9", "ABIERTA", new DateTime(2024, 05, 08),
BuscarArticulo(35));

AltaSubasta("Subasta 10", "ABIERTA", new DateTime(2024, 10,
18), BuscarArticulo(36));

Ofertas:

AgregarOfertaASubasta(1, 10, new DateTime(2024, 10, 08), 11);

AgregarOfertaASubasta(3, 44, new DateTime(2024, 07, 18), 15);

Código:

Archivo: Articulo.cs

Carpeta: C:\Users\matia\Desktop\ORT\Programación
2\Obligatorio\Obligatorio\Obligatorio\Articulo.cs

using Obligatorio.Interfaces;

using System;

using System.Collections.Generic;

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

namespace Obligatorio
{
    public class Articulo : IValidate
    {
        private int _id;
        private static int s_ultid = 0;
        private string _nombre;
        private string _categoria;
        private double _precio;

        public string categoria
        {
            get { return _categoria; }
        }

        public int id
        {
            get { return _id; }
        }

        public double precio
        {

```

```
    get { return _precio; }
}

public override string ToString()
{
    return $"{_nombre} - Precio ${_precio}.\n Categoría {_categoria}.";
}

public void Validar()
{
    if (string.IsNullOrEmpty(_nombre))
    {
        throw new Exception("El nombre es obligatorio");
    }
    if (string.IsNullOrEmpty(_categoria))
    {
        throw new Exception("La categoria es obligatoria");
    }
    if (_precio <= 0)
    {
        throw new Exception("El precio no puede ser menor a $0");
    }
}

public Artículo(string nombre, string categoria, double precio)
{
```



```

        _id = Artículo.s_ultild++;
        _nombre = nombre;
        _categoria = categoria;
        _precio = precio;
    }
}
}

```

Archivo: Cliente.cs

Carpeta: C:\Users\matia\Desktop\ORT\Programación
2\Obligatorio\Obligatorio\Obligatorio\Cliente.cs

```

using Obligatorio.Interfaces;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

```

```

namespace Obligatorio
{
    public class Cliente:Usuario
    {
        private double _saldo;

        public double saldo

```

```
{  
    get { return _saldo; }  
    set { _saldo = value; }  
}  
  
    public Cliente(string nombre, string apellido, string email, string  
password) : base(nombre, apellido, email, password)  
    {  
  
    }  
  
    public Cliente() : base(){ }  
  
    public override string ObtenerRol()  
    {  
        return "Cliente";  
    }  
  
    public void CargarSaldo(double saldo)  
    {  
        _saldo += saldo;  
    }  
  
}  
  
}
```

Archivo: Oferta.cs

Carpeta: C:\Users\matia\Desktop\ORT\Programación
2\Obligatorio\Obligatorio\Obligatorio\Oferta.cs

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

```
namespace Obligatorio
{
    public class Oferta : IValidate
    {
        private int _id;
        private static int s_ultId = 0;
        private Cliente _usuario;
        private double _monto;
        private DateTime _fecha;

        public Cliente usuario
        {
            get { return _usuario; }
            set { _usuario = value; }
        }
    }
}
```

```
}

public double monto
{
    get { return _monto; }
    set { _monto = value; }
}

public override string ToString()
{
    return $"{_usuario} ofertó {_monto} el día {_fecha}.";
}

public void Validar()
{
    if (_monto <= 0)
    {
        throw new Exception("La oferta debe ser mayor de 0");
    }
}

public Oferta(Cliente usuario, double monto, DateTime fecha)
{
    _id = s_ultId++;
    _usuario = usuario;
    _monto = monto;
    _fecha = fecha;
}
```

```
}  
}
```

```
*****
```

Archivo: Publicacion.cs

Carpeta: C:\Users\matia\Desktop\ORT\Programación
2\Obligatorio\Obligatorio\Obligatorio\Publicacion.cs

```
*****
```

```
using Obligatorio.Interfaces;  
using System;  
using System.Collections.Generic;  
using System.Linq;  
using System.Text;  
using System.Threading.Tasks;  
  
namespace Obligatorio  
{  
    public abstract class Publicacion : IValidate,  
        IComparable<Publicacion>  
    {  
        private int _id;  
        private static int s_ultId = 0;  
        private string _nombre;  
        private string _estado;  
        private DateTime _fechaPublicacion;
```

```
private List<Articulo> _articulos = new List<Articulo>();
```

```
private Cliente _realizaCompra;
```

```
private Cliente _finalizaCompra;
```

```
private DateTime _fechaFinaliza;
```

```
public int id
```

```
{
```

```
    get { return _id; }
```

```
}
```

```
public string nombre
```

```
{
```

```
    get { return _nombre; }
```

```
}
```

```
public DateTime fecha
```

```
{
```

```
    get { return _fechaPublicacion.Date; }
```

```
}
```

```
public string estado
```

```
{
```

```
    get { return _estado; }
```

```
    set { _estado = value; }
```

```
}
```

```
public List<Articulo> articulos
{
    get { return _articulos; }
}

public Cliente realizaCompra
{
    get { return _realizaCompra; }
    set { _realizaCompra = value; }
}

public Cliente finalizaCompra
{
    get { return _finalizaCompra; }
    set { _finalizaCompra = value; }
}

public DateTime fechaFinaliza
{
    get { return _fechaFinaliza; }
    set { _fechaFinaliza = value; }
}

public override string ToString()
{
    return $"({_nombre} - Estado: {_estado}. \n Fecha de publicación
{_fechaPublicacion}. \n Lista de artículos ";
```

```
}
```

```
public Publicacion(string nombre, string estado, DateTime  
fechaPublicacion, List<Articulo> articulos)
```

```
{
```

```
    _nombre = nombre;
```

```
    _estado = estado;
```

```
    _fechaPublicacion = fechaPublicacion;
```

```
    _articulos = articulos;
```

```
    _id = s_ultId++;
```

```
}
```

```
public void AltaArticulos(string nombre, string categoria, double  
precio)
```

```
{
```

```
    Articulo articulo = new Articulo(nombre, categoria, precio);
```

```
    if (!_articulos.Contains(articulo))
```

```
    {
```

```
        _articulos.Add(articulo);
```

```
    }
```

```
}
```

```
public void Validar()
```

```
{
```

```
    if (string.IsNullOrEmpty(_nombre)){
```

```
        throw new Exception("El nombre es obligatorio");
```

```
    }
```

```
    if (string.IsNullOrEmpty(_estado)) {
```



```

        throw new Exception("El estado es obligatorio");
    }

    else if (_estado.ToUpper().Trim() != "ABIERTA" ||
        _estado.ToUpper().Trim() != "CERRADA" || _estado.ToUpper().Trim() !=
        "CANCELADA")
    {
        throw new Exception("El estado no tiene un valor válido");
    }

}

public abstract double CalcularPrecio();

public int CompareTo(Publicacion? other)
{
    return _fechaPublicacion.CompareTo(other._fechaPublicacion);
}
}
}

```

Archivo: Sistema.cs

Carpeta: C:\Users\matia\Desktop\ORT\Programación
2\Obligatorio\Obligatorio\Obligatorio\Sistema.cs

```

using System;

using System.Collections.Generic;

```

```
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using static System.Runtime.InteropServices.JavaScript.JSType;
```

```
namespace Obligatorio
```

```
{
```

```
    public class Sistema
```

```
    {
```

```
        public List<Usuario> _usuarios = new List<Usuario>();
```

```
        public List<Publicacion> _publicaciones = new
List<Publicacion>();
```

```
        public List<Articulo> _articulos = new List<Articulo>();
```

```
        private static Sistema _instancia;
```

```
        public static Sistema Instancia
```

```
        {
```

```
            get
```

```
            {
```

```
                if (_instancia == null)
```

```
                {
```

```
                    _instancia = new Sistema();
```

```
                }
```

```
                return _instancia;
```

```
}  
}
```

```
public Sistema()  
{  
    PrecargaDeUsuarios();  
    PrecargaDeArticulos();  
    PrecargaDePublicaciones();  
    PrecargaDeOfertas();  
}
```

```
//Métodos para precargar
```

```
public void AltaCliente(string nombre, string apellido, string email,  
string password)  
{  
    Cliente cliente = new Cliente(nombre, apellido, email,  
password);  
    cliente.Validar();  
    if (!_usuarios.Contains(cliente)){  
        _usuarios.Add(cliente);  
    }  
  
}
```

```
private void AltaAdministrador(string nombre, string apellido, string  
email, string password)  
{
```

```
        Usuario administrador = new Usuario(nombre, apellido, email,
password);

        administrador.Validar();

        _usuarios.Add(administrador);

    }
```

```
    private void AltaVenta(string nombre, string estado, DateTime
fechaPublicacion, Artículo articulo, bool oferta)

    {

        List<Artículo> articulos = new List<Artículo>();

        articulos.Add(articulo);

        Venta venta = new Venta(nombre, estado, fechaPublicacion,
articulos, oferta);

        //      venta.Validar();

        _publicaciones.Add(venta);

    }
```

```
    private void AltaSubasta(string nombre, string estado, DateTime
fechaPublicacion, Artículo articulo)

    {

        List<Artículo> articulos = new List<Artículo>();

        articulos.Add(articulo);

        Subasta subasta = new Subasta(nombre, estado,
fechaPublicacion, articulos);

        //      subasta.Validar();

        _publicaciones.Add(subasta);

    }
```

```

    public void AgregarOfertaASubasta(int idCliente, double monto,
DateTime fecha, int idSubasta)
    {
        Subasta subasta = BuscarSubasta(idSubasta);
        Cliente cliente = BuscarCliente(idCliente);
        if (idCliente != null && idSubasta != null)
        {
            subasta.AltaOferta(cliente, monto, fecha);
        }
    }

    public void AgregarArticulosAPublicacion(int idPublicacion, string
nombre, string categoria, double precio)
    {
        Publicacion publicacion = BuscarPublicacion(idPublicacion);
        if (idPublicacion != null)
        {
            publicacion.AltaArticulos(nombre, categoria, precio);
        }
    }

    public void AltaArticulo(string nombre, string categoria, double
precio)
    {
        Artículo articulo = new Artículo(nombre, categoria, precio);
        articulo.Validar();
        if (!_articulos.Contains(articulo))
        {

```

```
        _articulos.Add(articulo);  
    }  
}
```

//Métodos para los controller

```
public void InscribirCliente(Cliente cliente)
```

```
{  
    cliente.Validar();  
    if (!_usuarios.Contains(cliente))  
    {  
        _usuarios.Add(cliente);  
    }  
    else  
    {  
        throw new Exception("El cliente ya existe");  
    }  
}
```

```
public void ComprarVenta(int idVenta, Cliente comprador)
```

```
{  
    Venta venta = BuscarVenta(idVenta);  
    if (venta != null)  
    {  
        if (comprador.saldo >= venta.CalcularPrecio())  
        {  
            comprador.saldo -= venta.CalcularPrecio();  
        }  
    }  
}
```

```
        venta.estado = "CERRADA";
        venta.realizaCompra = comprador;
        venta.fechaFinaliza = DateTime.Now;
    }
    else
    {
        throw new Exception("Saldo insuficiente");
    }
}
}
```

```
public void CerrarSubasta(int idSubasta, Cliente mejorOfertante)
{
    Subasta subasta = BuscarSubasta(idSubasta);
    if (subasta != null)
    {
        if (mejorOfertante.saldo >= subasta.OfertaMasAlta().monto)
        {
            mejorOfertante.saldo -= subasta.OfertaMasAlta().monto;
            subasta.estado = "CERRADA";
            subasta.finalizaCompra = mejorOfertante;
            subasta.fechaFinaliza = DateTime.Now;
        }
        else
        {

```

```
        throw new Exception("Saldo insuficiente");
    }
}
}
```

//Métodos para buscar en las clases

```
public Subasta BuscarSubasta(int idSubasta)
{
    Subasta subasta = null;
    int i = 0;
    while(i<_publicaciones.Count && subasta == null)
    {
        if (_publicaciones[i].id == idSubasta && _publicaciones[i] is
Subasta)
        {
            subasta = (Subasta) _publicaciones[i];
        }
        i++;
    }
    return subasta;
}
```

```
public Venta BuscarVenta(int idVenta)
{
    Venta venta = null;
    int i = 0;
    while (i < _publicaciones.Count && venta == null)
```



```

    {
        if (_publicaciones[i].id == idVenta && _publicaciones[i] is
Venta)
        {
            venta = (Venta)_publicaciones[i];
        }
        i++;
    }
    return venta;
}

```

```

public Artículo BuscarArticulo(int idArticulo)
{
    Artículo articulo = null;
    int i = 0;
    while (i < _articulos.Count && articulo == null)
    {
        if (_articulos[i].id == idArticulo)
        {
            articulo = _articulos[i];
        }
        i++;
    }
    return articulo;
}

```

```

public Cliente BuscarCliente(int idCliente)

```

```
{  
    Cliente cliente = null;  
    int i = 0;  
    while (i < _usuarios.Count && cliente == null)  
    {  
        if (_usuarios[i].id == idCliente && _usuarios[i] is Cliente)  
        {  
            cliente = (Cliente)_usuarios[i];  
        }  
        i++;  
    }  
    return cliente;  
}
```

```
public Cliente BuscarClientePorEmail(string email)  
{  
    Cliente cliente = null;  
    int i = 0;  
    while (i < _usuarios.Count && cliente == null)  
    {  
        if (_usuarios[i].email == email && _usuarios[i] is Cliente)  
        {  
            cliente = (Cliente)_usuarios[i];  
        }  
        i++;  
    }  
}
```

```
    return cliente;
}
```

```
    public Usuario BuscarUsuarioParaLogin(string email, string
password)
    {
        Usuario usuario = null;
        int i = 0;
        while (i < _usuarios.Count && usuario == null)
        {
            if (_usuarios[i].email == email && _usuarios[i].password ==
password)
            {
                usuario = _usuarios[i];
            }
            i++;
        }
        return usuario;
    }
```

```
public Publicacion BuscarPublicacion(int idPublicacion)
{
    Publicacion publicacion = null;
    int i = 0;
    while (i < _publicaciones.Count && publicacion == null)
    {
```

```
        if (_publicaciones[i].id == idPublicacion)
        {
            publicacion = _publicaciones[i];
        }
        i++;
    }
    return publicacion;
}
```

//Métodos para devolver listas

```
public List<Publicacion> ObtenerPublicaciones()
{
    return _publicaciones;
}
```

```
public List<Cliente> ObtenerClientes()
{
    List<Cliente> clientes = new List<Cliente> ();
    int i = 0;
    while (i < _usuarios.Count)
    {
        if (_usuarios[i] is Cliente)
        {
            clientes.Add((Cliente)_usuarios[i]);
        }
        i++;
    }
}
```

```
    }  
    return clientes;  
}
```

```
public List<Articulo> ListaDeArticulosPorCategoria(string  
categoria)  
{  
    List<Articulo> articulos = new List<Articulo> ();  
    int i = 0;  
    while (i < _articulos.Count && !string.IsNullOrEmpty(categoria))  
    {  
        if (_articulos[i].categoria.ToUpper().Trim() ==  
categoria.ToUpper().Trim())  
        {  
            articulos.Add(_articulos[i]);  
        }  
        i++;  
    }  
    return articulos;  
}
```

```
public List<Publicacion>  
ObtenerPublicacionesPorFecha(DateTime fechaInicio, DateTime  
fechaFinal)  
{  
    List<Publicacion> publicaciones = new List<Publicacion>();  
    int i = 0;  
    while (i < _publicaciones.Count)
```

```

    {
        if (_publicaciones[i].fecha >= fechaInicio &&
            _publicaciones[i].fecha <= fechaFinal)
        {
            publicaciones.Add(_publicaciones[i]);
        }
        i++;
    }
    return publicaciones;
}

```

```

public List<Subasta> ObtenerSubastasOrdenadasPorFecha()
{
    List<Subasta> subastas = new List<Subasta>();
    int i = 0;
    while (i < _publicaciones.Count)
    {
        if (_publicaciones[i] is Subasta)
        {
            subastas.Add((Subasta)_publicaciones[i]);
        }
        i++;
    }
    subastas.Sort();
    return subastas;
}

```

```
//Precarga de todos los datos separados por tipo
private void PrecargaDeUsuarios()
{
    AltaCliente("Juan", "Pérez", "juan.perez@example.com",
"contrasena123");

    AltaCliente("Ana", "Gómez", "ana.gomez@example.com",
"contrasena456");

    AltaCliente("Carlos", "Fernández",
"carlos.fernandez@example.com", "contrasena789");

    AltaCliente("Laura", "Martínez", "laura.martinez@example.com",
"laura2023");

    AltaCliente("Javier", "López", "javier.lopez@example.com",
"javier2023");

    AltaCliente("Marta", "Rodríguez",
"marta.rodriguez@example.com", "marta1234");

    AltaCliente("José", "Torres", "jose.torres@example.com",
"jose5678");

    AltaCliente("Sofia", "Díaz", "sofia.diaz@example.com",
"sofia910");

    AltaCliente("Luis", "Morales", "luis.morales@example.com",
"luis1111");

    AltaCliente("Patricia", "Jiménez",
"patricia.jimenez@example.com", "patricia2222");

    AltaAdministrador("Ricardo", "Alvarez",
"ricardo.alvarez@example.com", "ricardo3333");

    AltaAdministrador("Elena", "Sánchez",
"elena.sanchez@example.com", "elena4444");
}
```

```
private void PrecargaDeArticulos()
{
    AltaArticulo("Vestido de verano", "Ropa", 85.60);
    AltaArticulo("Collar de perlas", "Joyería", 200.00);
    AltaArticulo("Botas de cuero", "Calzado", 120.90);
    AltaArticulo("Lampara LED", "Hogar", 95.00);
    AltaArticulo("Bicicleta de montaña", "Deportes", 550.00);
    AltaArticulo("Crema hidratante", "Belleza", 70.00);
    AltaArticulo("Set de herramientas", "Jardinería", 45.20);
    AltaArticulo("Gorro de invierno", "Accesorios", 33.80);
    AltaArticulo("Laptop Ultra Book", "Electrónica", 1200.40);
    AltaArticulo("Pantalones jeans", "Ropa", 55.25); //10
    AltaArticulo("Pulsera de acero", "Joyería", 95.60);
    AltaArticulo("Sandalias veraniegas", "Calzado", 65.70);
    AltaArticulo("Juego de toallas", "Hogar", 150.90);
    AltaArticulo("Raqueta de tenis", "Deportes", 85.00);
    AltaArticulo("Maquillaje compacto", "Belleza", 45.50);
    AltaArticulo("Pantalones jeans", "Ropa", 55.25);
    AltaArticulo("Pantalones jeans", "Ropa", 55.25);
    AltaArticulo("Pantalones jeans", "Ropa", 55.25);
    AltaArticulo("Pantalones jeans", "Ropa", 55.25);
    AltaArticulo("Pantalones jeans", "Ropa", 55.25); //20
    AltaArticulo("Maquillaje compacto", "Belleza", 45.50);
    AltaArticulo("Kit de jardinería", "Jardinería", 70.00);
    AltaArticulo("Bolso de mano", "Accesorios", 27.30);
```


AltaArticulo("Altavoz Bluetooth", "Electrónica", 400.00);
AltaArticulo("Chaqueta de invierno", "Ropa", 75.30);
AltaArticulo("Aretes de oro", "Joyería", 150.00);
AltaArticulo("Sneakers de moda", "Calzado", 90.40);
AltaArticulo("Cuadro decorativo", "Hogar", 60.50);
AltaArticulo("Set de pesas", "Deportes", 110.00);
AltaArticulo("Gel exfoliante", "Belleza", 55.90); //30
AltaArticulo("Fertilizante orgánico", "Jardinería", 40.00);
AltaArticulo("Cinturón de cuero", "Accesorios", 22.90);
AltaArticulo("Monitor 24\"", "Electrónica", 300.00);
AltaArticulo("Falda larga", "Ropa", 30.00);
AltaArticulo("Broche de cristal", "Joyería", 85.60);
AltaArticulo("Chanclas de playa", "Calzado", 95.80);
AltaArticulo("Manta suave", "Hogar", 210.50);
AltaArticulo("Kit de yoga", "Deportes", 75.10);
AltaArticulo("Mascarilla facial", "Belleza", 35.00);
AltaArticulo("Set de semillas", "Jardinería", 50.40); //40
AltaArticulo("Reloj de pulsera", "Accesorios", 60.70);
AltaArticulo("Proyector portátil", "Electrónica", 500.00);
AltaArticulo("Pijama de seda", "Ropa", 45.00);
AltaArticulo("Pulsera de cuero", "Joyería", 160.00);
AltaArticulo("Botines de gamuza", "Calzado", 80.50);
AltaArticulo("Mesa de café", "Hogar", 95.00);
AltaArticulo("Smartphone X10", "Electrónica", 300.20);
AltaArticulo("Camiseta de algodón", "Ropa", 25.50);
AltaArticulo("Anillo de plata", "Joyería", 120.75);

```
AltaArticulo("Zapatillas deportivas", "Calzado", 75.90); //50  
}
```

```
private void PrecargaDePublicaciones()  
{  
    AltaVenta("Venta 1", "ABIERTA", new DateTime(2024, 10, 08),  
BuscarArticulo(3), false);  
  
    AltaVenta("Venta 2", "ABIERTA", new DateTime(2023, 05, 18),  
BuscarArticulo(7), true);  
  
    AltaVenta("Venta 3", "ABIERTA", new DateTime(2021, 01, 07),  
BuscarArticulo(21), false);  
  
    AltaVenta("Venta 4", "ABIERTA", new DateTime(2022, 09, 09),  
BuscarArticulo(17), false);  
  
    AltaVenta("Venta 5", "ABIERTA", new DateTime(2023, 12, 03),  
BuscarArticulo(41), false);  
  
    AltaVenta("Venta 6", "ABIERTA", new DateTime(2022, 02, 01),  
BuscarArticulo(44), false);  
  
    AltaVenta("Venta 7", "ABIERTA", new DateTime(2023, 04, 29),  
BuscarArticulo(7), false);  
  
    AltaVenta("Venta 8", "ABIERTA", new DateTime(2021, 01, 23),  
BuscarArticulo(2), true);  
  
    AltaVenta("Venta 9", "ABIERTA", new DateTime(2020, 04, 28),  
BuscarArticulo(13), false);  
  
    AltaVenta("Venta 10", "ABIERTA", new DateTime(2019, 09, 18),  
BuscarArticulo(33), false);  
  
    AltaSubasta("Subasta 1", "ABIERTA", new DateTime(2017, 07, 07),  
BuscarArticulo(5));  
  
    AltaSubasta("Subasta 2", "ABIERTA", new DateTime(2020, 11, 07),  
BuscarArticulo(45));  
}
```

```
        AltaSubasta("Subasta 3", "ABIERTA", new DateTime(2023, 06, 06),  
        BuscarArticulo(14));
```

```
        AltaSubasta("Subasta 4", "ABIERTA", new DateTime(2024, 02, 08),  
        BuscarArticulo(22));
```

```
        AltaSubasta("Subasta 5", "ABIERTA", new DateTime(2024, 11, 09),  
        BuscarArticulo(48));
```

```
        AltaSubasta("Subasta 6", "ABIERTA", new DateTime(2023, 12, 11),  
        BuscarArticulo(37));
```

```
        AltaSubasta("Subasta 7", "ABIERTA", new DateTime(2024, 02, 01),  
        BuscarArticulo(32));
```

```
        AltaSubasta("Subasta 8", "ABIERTA", new DateTime(2022, 03, 22),  
        BuscarArticulo(12));
```

```
        AltaSubasta("Subasta 9", "ABIERTA", new DateTime(2024, 05, 08),  
        BuscarArticulo(35));
```

```
        AltaSubasta("Subasta 10", "ABIERTA", new DateTime(2024, 10,  
18), BuscarArticulo(36));
```

```
    }
```

```
    private void PrecargaDeOfertas()
```

```
    {
```

```
        AgregarOfertaASubasta(1, 10, new DateTime(2024, 10, 08), 11);
```

```
        AgregarOfertaASubasta(3, 44, new DateTime(2024, 07, 18), 15);
```

```
    }
```

```
}
```

```
}
```

```
*****
```

Archivo: Subasta.cs

Carpeta: C:\Users\matia\Desktop\ORT\Programación
2\Obligatorio\Obligatorio\Obligatorio\Subasta.cs

```
using Obligatorio.Interfaces;
```

```
using System;
```

```
using System.Collections.Generic;
```

```
using System.Linq;
```

```
using System.Text;
```

```
using System.Threading.Tasks;
```

```
namespace Obligatorio
```

```
{
```

```
    public class Subasta:Publicacion
```

```
    {
```

```
        private List<Oferta> _ofertas=new List<Oferta>();
```

```
        public List<Oferta> ofertas
```

```
        {
```

```
            get { return _ofertas; }
```

```
        }
```

```
        public Subasta(string nombre, string estado, DateTime  
fechaPublicacion, List<Articulo> articulos) : base (nombre, estado,  
fechaPublicacion, articulos)
```

```
        {
```

```
        }
```

```
public void AltaOferta(Cliente cliente, double monto, DateTime  
fecha)
```

```
{  
    Oferta oferta = new Oferta(cliente, monto, fecha);  
    if (!_ofertas.Contains(oferta))  
    {  
        _ofertas.Add(oferta);  
    }  
}
```

```
public Oferta OfertaMasAlta()  
{  
    Oferta ofertaAlta = null;  
    foreach (Oferta oferta in _ofertas)  
    {  
        if (oferta.monto == _ofertas.Max(o => o.monto) &&  
oferta.monto > 0)  
        {  
            ofertaAlta = oferta;  
        }  
    }  
    return ofertaAlta;  
}
```

```
public double MontoOfertaMasAlta()  
{
```

```

double monto = 0;
foreach (Oferta oferta in _ofertas)
{
    if (oferta.monto == _ofertas.Max(o => o.monto))
    {
        monto = oferta.monto;
    }
}
return monto;
}

```

```

public override double CalcularPrecio()
{
    return MontoOfertaMasAlta();
}

```

```

    }
}

```

Archivo: Usuario.cs

Carpeta: C:\Users\matia\Desktop\ORT\Programación
2\Obligatorio\Obligatorio\Obligatorio\Usuario.cs

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

```
namespace Obligatorio
```

```
{
```

```
    public class Usuario : IValidate, IEquatable<Usuario>
```

```
    {
```

```
        private int _id;
```

```
        private static int s_ultId = 0;
```

```
        private string _nombre;
```

```
        private string _apellido;
```

```
        private string _email;
```

```
        private string _password;
```

```
        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 email  
{  
    get { return _email; }  
    set { _email = value; }  
}
```

```
public string password  
{  
    get { return _password; }  
    set { _password = value; }  
}
```

```
public override string ToString()  
{  
    return $"{_nombre}{_apellido}. Mail: {_email}.";  
}
```



```
public void Validar()
{
    if (string.IsNullOrEmpty(_nombre))
    {
        throw new Exception("El nombre es obligatorio");
    }
    if (string.IsNullOrEmpty(_apellido))
    {
        throw new Exception("El Apellido es obligatorio");
    }
    if (string.IsNullOrEmpty(_email))
    {
        throw new Exception("El email es obligatorio");
    }
    if (string.IsNullOrEmpty(_password))
    {
        throw new Exception("Ingrese una contraseña válida");
    }
}
```

```
public Usuario(string nombre, string apellido, string email, string
contrasenia)
{
    _nombre = nombre;
```

```

        _apellido = apellido;
        _email = email;
        _password = contrasenia;
        _id = s_ultId++;
    }

    public Usuario()
    {
        _id = s_ultId++;
    }

    public bool Equals(Usuario? other)
    {
        return _email == other._email;
    }

    public virtual string ObtenerRol()
    {
        return "Administrador";
    }
}
}

```

Archivo: Venta.cs

Carpeta: C:\Users\matia\Desktop\ORT\Programación
2\Obligatorio\Obligatorio\Obligatorio\Venta.cs

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

namespace Obligatorio
{
    public class Venta:Publicacion
    {
        private bool _oferta;

        public Venta(string nombre, string estado, DateTime
fechaPublicacion, List<Articulo> articulos, bool oferta) : base(nombre,
estado, fechaPublicacion, articulos)
        {
            _oferta = oferta;
        }

        public override double CalcularPrecio()
        {
            double precio = 0;
            foreach (Articulo articulo in articulos)
            {
                precio += articulo.precio;
            }
        }
    }
}
```

```

        if (_oferta)
        {
            precio = precio - (precio * 0.2);
        }

        return precio;
    }
}
}

```

Archivo: IValidate.cs

Carpeta: C:\Users\matia\Desktop\ORT\Programación
2\Obligatorio\Obligatorio\Obligatorio\Interfaces\IValidate.cs

```

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

```

```

namespace Obligatorio.Interfaces

```

```

{
    internal interface IValidate
    {
        void Validar();
    }
}

```

Archivo: AdministradorController.cs

Carpeta: C:\Users\matia\Desktop\ORT\Programación
2\Obligatorio\Obligatorio\Web\Controllers\AdministradorController.cs

using Obligatorio;

using Microsoft.AspNetCore.Mvc;

namespace Web.Controllers

{

public class AdministradorController : Controller

{

public IActionResult IndexAdmin()

{

return View();

}

public IActionResult VerSubastas()

{

List<Subasta> subastas =

Sistema.Instancia.ObtenerSubastasOrdenadasPorFecha();

return View(subastas);

}

}

```
}
```

```
*****
```

Archivo: ClienteController.cs

Carpeta: C:\Users\matia\Desktop\ORT\Programación
2\Obligatorio\Obligatorio\Web\Controllers\ClienteController.cs

```
*****
```

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

```
using Obligatorio;
```

```
namespace Web.Controllers
```

```
{
```

```
    public class ClienteController : Controller
```

```
    {
```

```
        public IActionResult IndexCliente()
```

```
        {
```

```
            Cliente cliente =
```

```
Sistema.Instancia.BuscarClientePorEmail(SessionExtensions.GetString(HttpContext.Session, "usuario"));
```

```
            ViewBag.Mensaje = "Bienvenido " + cliente.nombre + " " +  
cliente.apellido;
```

```
            ViewBag.Saldo = "Saldo: $" + cliente.saldo;
```

```
            return View();
```

```
        }
```

```
// NO VALIDA EMAIL
```

```
[HttpGet]
```

```
public IActionResult Registro()
```

```

    {
        return View();
    }

    [HttpPost]
    public IActionResult Registro(Cliente cliente)
    {
        try
        {
            cliente.Validar();
            Sistema.Instancia.InscribirCliente(cliente);
            ViewBag.Mensaje = "Cliente registrado correctamente";
        }
        catch (Exception ex)
        {
            ViewBag.Mensaje = ex.Message;
        }
        return View();
    }

    //NO VALIDA EMAIL

    public IActionResult VerPublicacionesCliente()
    {
        List<Publicacion> publicaciones =
        Sistema.Instancia.ObtenerPublicaciones();

        return View(publicaciones);
    }

```

```

public IActionResult Comprar(int id)
{
    try
    {
        Cliente comprador =
Sistema.Instancia.BuscarClientePorEmail(SessionExtensions.GetString(HttpContext.Session, "usuario"));

        Sistema.Instancia.ComprarVenta(id, comprador);

        ViewBag.Mensaje = "Compra registrada correctamente";
    }
    catch (Exception ex)
    {
        ViewBag.Mensaje = ex.Message;
    }
    return RedirectToAction(nameof(VerPublicacionesCliente));
}

```

[HttpGet]

```

public IActionResult CargarSaldo()

```

```

{
    return View();
}

```

```

public IActionResult CargarSaldo(double monto)

```

```

{
    try

```



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

```
using Obligatorio;
```

```
using System.Diagnostics;
```

```
using Web.Models;
```

```
namespace Web.Controllers
```

```
{
```

```
    public class HomeController : Controller
```

```
    {
```

```
        private readonly ILogger<HomeController> _logger;
```

```
        public HomeController(ILogger<HomeController> logger)
```

```
        {
```

```
            _logger = logger;
```

```
        }
```

```
        public IActionResult Index()
```

```
        {
```

```
            return View();
```

```
        }
```

```
        public IActionResult Privacy()
```

```
        {
```

```
            return View();
```

```
        }
```

```

[ResponseCache(Duration = 0, Location =
ResponseCacheLocation.None, NoStore = true)]

public IActionResult Error()
{
    return View(new ErrorViewModel { RequestId =
Activity.Current?.Id ?? HttpContext.TraceIdentifier });
}

```

```

[HttpGet]

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

```

```

[HttpPost]

public IActionResult Login(string email, string password)
{
    try
    {
        if (!string.IsNullOrEmpty(email) &&
!string.IsNullOrEmpty(password))
        {
            Usuario usuario =
Sistema.Instancia.BuscarUsuarioParaLogin(email, password);

            if (usuario != null)
            {
                string rol = usuario.ObtenerRol();

                HttpContext.Session.SetString("usuario", usuario.email);

```

```
HttpContext.Session.SetString("rol", rol);
if (rol.Trim().ToUpper().Equals("CLIENTE"))
{
    return RedirectToAction("IndexCliente", "Cliente");
}
else
{
    return RedirectToAction("IndexAdmin", "Administrador");
}

}
else
{
    ViewBag.Mensaje = "Usuario o contrase a incorrectos";
}
}
else
{
    ViewBag.Mensaje = "El usuario y la contrase a son
obligatorios";
}
}
catch (Exception ex)
{
    return ViewBag.Mensaje = ex.Message;
}
return View();
```

```

    }

    public IActionResult Logout()
    {
        HttpContext.Session.Clear();
        return RedirectToAction("Login");
    }

}

}

```

Archivo: SubastaController.cs

Carpeta: C:\Users\matia\Desktop\ORT\Programación
2\Obligatorio\Obligatorio\Web\Controllers\SubastaController.cs

```

using Obligatorio;
using Microsoft.AspNetCore.Mvc;

namespace Web.Controllers
{
    public class SubastaController : Controller
    {
        [HttpGet]
        public IActionResult Ofertar(int id)
        {
            Subasta subasta = Sistema.Instancia.BuscarSubasta(id);

```

```
        return View(subasta);  
    }  
}
```

```
[HttpPost]  
public IActionResult Ofertar(int id, double monto)  
{  
    try  
    {  
        Subasta subasta = Sistema.Instancia.BuscarSubasta(id);  
        Cliente ofertante =  
Sistema.Instancia.BuscarClientePorEmail(SessionExtensions.GetString(HttpContext.Session, "usuario"));  
        subasta.AltaOferta(ofertante, monto, DateTime.Now);  
        ViewBag.Mensaje = "Oferta registrada correctamente";  
        return RedirectToAction("VerPublicacionesCliente", "Cliente");  
    }  
    catch (Exception ex)  
    {  
        ViewBag.Mensaje = ex.Message;  
    }  
    return View("VerPublicacionesCliente", "Cliente");  
}
```

```
public IActionResult CerrarSubasta(int id)  
{  
    try  
    {
```

```

        Subasta subasta = Sistema.Instancia.BuscarSubasta(id);
        if(subasta.ofertas.Count() > 0)
        {
            Cliente cliente = subasta.OfertaMasAlta().usuario;
            Sistema.Instancia.CerrarSubasta(id, cliente);

            ViewBag.Mensaje = "Subasta cerrada con éxito";
        }
    }
    catch (Exception ex)
    {
        ViewBag.Mensaje = ex.Message;
    }
    return RedirectToAction("VerSubastas", "Administrador");
}

}
}

```

Archivo: ErrorViewModel.cs

Carpeta: C:\Users\matia\Desktop\ORT\Programación
2\Obligatorio\Obligatorio\Web\Models\ErrorViewModel.cs

```

namespace Web.Models
{
    public class ErrorViewModel

```

```
{  
    public string? RequestId { get; set; }  
  
    public bool ShowRequestId => !string.IsNullOrEmpty(RequestId);  
}  
}
```

Archivo: _ViewImports.cshtml

Carpeta: C:\Users\matia\Desktop\ORT\Programación
2\Obligatorio\Obligatorio\Web\Views_ViewImports.cshtml

@using Web

@using Web.Models

@addTagHelper *, Microsoft.AspNetCore.Mvc.TagHelpers

Archivo: _ViewStart.cshtml

Carpeta: C:\Users\matia\Desktop\ORT\Programación
2\Obligatorio\Obligatorio\Web\Views_ViewStart.cshtml

@{

Layout = "_Layout";

}

Archivo: IndexAdmin.cshtml

Carpeta: C:\Users\matia\Desktop\ORT\Programación
2\Obligatorio\Obligatorio\Web\Views\Administrador\IndexAdmin.cshtml

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

<h1>Bienvenido Administrador</h1>

Archivo: VerSubastas.cshtml

Carpeta: C:\Users\matia\Desktop\ORT\Programación
2\Obligatorio\Obligatorio\Web\Views\Administrador\VerSubastas.cshtml

```
@model List<Obligatorio.Subasta>
```

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

<h1>Todas las subastas</h1>

```
@if (Model != null && Model.Count > 0)
```

```
{
```

```

<table class="table">
  <thead>
    <tr>
      <th>Nombre</th>
      <th>Fecha</th>
      <th>Estado</th>
      <th>Precio</th>
      <th></th>
    </tr>
  </thead>

  <tbody>
    @foreach (Obligatorio.Subasta subasta in Model)
    {
      <tr>
        <td>@subasta.nombre</td>
        <td>@subasta.fecha.ToShortDateString()</td>
        <td>@subasta.estado</td>
        <td>@subasta.CalcularPrecio()</td>
        @if (subasta.estado.Trim().ToUpper() == "ABIERTA" &&
subasta is Obligatorio.Subasta)
        {
          <td>
            <a href="/Subasta/CerrarSubasta/@(subasta.id)"
class="btn btn-primary">Cerrar subasta</a>
          </td>
        }
      }
    }
  </tbody>
</table>

```

```
        @if (subasta.estado.Trim().ToUpper() == "CERRADA" &&
subasta is Obligatorio.Subasta)
```

```
        {
```

```
            <td>
```

```
                <p>Subasta cerrada</p>
```

```
            </td>
```

```
        }
```

```
    </tr>
```

```
}
```

```
</tbody>
```

```
</table>
```

```
}
```

```
else
```

```
{
```

```
    <p>No existen publicaciones</p>
```

```
}
```

```
*****
```

Archivo: CargarSaldo.cshtml

Carpeta: C:\Users\matia\Desktop\ORT\Programación

2\Obligatorio\Obligatorio\Web\Views\Cliente\CargarSaldo.cshtml

```
*****
```

```
@{
```

```
    ViewData["Title"] = "CargarSaldo";
```

```
}
```

```
<h1>Carga de saldo</h1>
```

```
<form method="post">
    <label for="monto">Monto</label>
    <input type="number" name="monto">
    <br>
    <input type="submit" class="btn btn-primary" value="Cargar">
</form>
```

@ViewBag.Mensaje

Archivo: IndexCliente.cshtml

Carpeta: C:\Users\matia\Desktop\ORT\Programación
2\Obligatorio\Obligatorio\Web\Views\Cliente\IndexCliente.cshtml

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

<h1>@ViewBag.Mensaje</h1>

<p>@ViewBag.Saldo</p>

Archivo: Registro.cshtml

Carpeta: C:\Users\matia\Desktop\ORT\Programación
2\Obligatorio\Obligatorio\Web\Views\Cliente\Registro.cshtml

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

```
<h1>Registrarse</h1>
```

```
<form method="post">
```

```
    <div class="form-group">
```

```
        <label for="nombre">Nombre</label>
```

```
        <input type="text" class="form-control" id="nombre"
name="nombre" />
```

```
    </div>
```

```
    <div class="form-group">
```

```
        <label for="apellido">Apellido</label>
```

```
        <input type="text" class="form-control" id="apellido"
name="apellido" />
```

```
    </div>
```

```
    <div class="form-group">
```

```
        <label for="email">Email</label>
```

```
        <input type="email" class="form-control" id="email" name="email"
/>
```

```
    </div>
```

```
    <div class="form-group">
```

```
        <label for="password">Contraseña</label>
```

```
        <input type="password" class="form-control" id="password"
name="password" />
```

```
    </div>
```

```
    <br>
```

```
<input type="submit" class="btn btn-primary"
value="Registrar"></input>
```

```
</form>
```

```
@ViewBag.Mensaje
```

```
*****
```

Archivo: VerPublicacionesCliente.cshtml

Carpeta: C:\Users\matia\Desktop\ORT\Programación
2\Obligatorio\Obligatorio\Web\Views\Cliente\VerPublicacionesCliente
.cshtml

```
*****
```

```
@model List<Obligatorio.Publicacion>
```

```
@{
```

```
    ViewData["Title"] = "VerPublicacionesCliente";
```

```
}
```

```
<h1>Publicaciones disponibles</h1>
```

```
@if (Model != null && Model.Count > 0)
```

```
{
```

```
    <table class="table">
```

```
        <thead>
```

```
            <tr>
```

```
                <th>Nombre</th>
```

```
                <th>Estado</th>
```

```
                <th>Precio</th>
```

```

        <th></th>

    </tr>

</thead>

<tbody>

    @foreach (Obligatorio.Publicacion publicacion in Model)
    {
        <tr>

            <td>@publicacion.nombre</td>

            <td>@publicacion.estado</td>

            <td>@publicacion.CalcularPrecio()</td>

            @if (publicacion.estado.Trim().ToUpper() == "ABIERTA" &&
publicacion is Obligatorio.Venta)
            {
                <td>

                    <a href="/Cliente/Comprar/@(publicacion.id)" class="btn
btn-primary">Comprar</a>

                </td>

            }

            @if (publicacion.estado.Trim().ToUpper() == "CERRADA" &&
publicacion is Obligatorio.Venta)
            {
                <td>

                    <p>Vendido</p>

                </td>

            }

            @if (publicacion.estado.Trim().ToUpper() == "ABIERTA" &&
publicacion is Obligatorio.Subasta)

```

```

        {
            <td>

                <a href="/Subasta/Ofertar/@(publicacion.id)"
class="btn btn-primary">Ofertar</a>

            </td>

        }

        @if (publicacion.estado.Trim().ToUpper() == "CERRADA" &&
publicacion is Obligatorio.Subasta)
        {
            <td>

                <p>Subasta cerrada</p>

            </td>

        }
    </tr>

}

</tbody>

</table>

@ViewBag.Mensaje

}

else
{
    <p>No existen publicaciones</p>
}

```

Archivo: Index.cshtml

Carpeta: C:\Users\matia\Desktop\ORT\Programación
2\Obligatorio\Obligatorio\Web\Views\Home\Index.cshtml

@{

 ViewData["Title"] = "Home Page";

}

<div class="text-center">

 <h1 class="display-4">Bienvenidos</h1>

 <p>Sebastián Hohl - 327007</p>

 <p>Matías Oreiro - 239479</p>

 <p>Programación 2 - M2E - Liliana Pino</p>

</div>

Archivo: Login.cshtml

Carpeta: C:\Users\matia\Desktop\ORT\Programación
2\Obligatorio\Obligatorio\Web\Views\Home\Login.cshtml

@{

 ViewData["Title"] = "Login";

}

<h1>Iniciar sesión</h1>

```

<form method="post">

    <div class="form-group">
        <label for="email">Email</label>
        <input type="email" class="form-control" id="email" name="email"
    />
    </div>

    <div class="form-group">
        <label for="password">Password</label>
        <input type="password" class="form-control" id="password"
name="password" />
    </div>

    <br>

    <input type="submit" class="btn btn-primary" value="Login"></input>
</form>

```

@ViewBag.Mensaje

Archivo: Privacy.cshtml

Carpeta: C:\Users\matia\Desktop\ORT\Programación
2\Obligatorio\Obligatorio\Web\Views\Home\Privacy.cshtml

```

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

```

<p>Vamo el manya</p>

Archivo: Error.cshtml

Carpeta: C:\Users\matia\Desktop\ORT\Programación
2\Obligatorio\Obligatorio\Web\Views\Shared\Error.cshtml

@model ErrorViewModel

@{

 ViewData["Title"] = "Error";

}

<h1 class="text-danger">Error.</h1>

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

@if (Model.ShowRequestId)

{

 <p>

 Request ID: <code>@Model.RequestId</code>

 </p>

}

<h3>Development Mode</h3>

<p>

Swapping to **Development** environment will display more detailed information about the error that occurred.

</p>

<p>

The Development environment shouldn't be enabled for deployed applications.

It can result in displaying sensitive information from exceptions to end users.

For local debugging, enable the **Development** environment by setting the **ASPNETCORE_ENVIRONMENT** environment variable to **Development**

and restarting the app.

</p>

Archivo: _Layout.cshtml

Carpeta: C:\Users\matia\Desktop\ORT\Programación
2\Obligatorio\Obligatorio\Web\Views\Shared_Layout.cshtml

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>@ViewData["Title"] - Web</title>

<link rel="stylesheet"
href="~/lib/bootstrap/dist/css/bootstrap.min.css" />

```

<link rel="stylesheet" href="~/css/site.css" asp-append-
version="true" />

<link rel="stylesheet" href="~/Web.styles.css" asp-append-
version="true" />

</head>

<body>

    <header>

        <nav class="navbar navbar-expand-sm navbar-toggleable-sm
navbar-light bg-white border-bottom box-shadow mb-3">

            <div class="container-fluid">

                @if (Context.Session.GetString("usuario") == null)
                {
                    <a class="navbar-brand" asp-area="" asp-controller="Home"
asp-action="Index">Inicio</a>
                }

                @if (Context.Session.GetString("usuario") != null &&
Context.Session.GetString("rol").Trim().ToUpper() == "CLIENTE")
                {
                    <a class="navbar-brand" asp-area="" asp-
controller="Cliente" asp-action="IndexCliente">Inicio</a>
                }

                @if (Context.Session.GetString("usuario") != null &&
Context.Session.GetString("rol").Trim().ToUpper() ==
"ADMINISTRADOR")
                {
                    <a class="navbar-brand" asp-area="" asp-
controller="Administrador" asp-action="IndexAdmin">Inicio</a>
                }
            }
        }
    }

```

```

        <button class="navbar-toggler" type="button" data-bs-
toggle="collapse" data-bs-target=".navbar-collapse" aria-
controls="navbarSupportedContent"

        aria-expanded="false" aria-label="Toggle navigation">

        <span class="navbar-toggler-icon"></span>

</button>

<div class="navbar-collapse collapse d-sm-inline-flex justify-
content-between">

    <ul class="navbar-nav flex-grow-1">

        @if (Context.Session.GetString("usuario") == null)
        {
            <li class="nav-item">

                <a class="nav-link text-dark" asp-area="" asp-
controller="Home" asp-action="Login">Iniciar sesión</a>

            </li>

            <li class="nav-item">

                <a class="nav-link text-dark" asp-area="" asp-
controller="Cliente" asp-action="Registro">Registro</a>

            </li>

        }

        @if (Context.Session.GetString("usuario") != null &&
Context.Session.GetString("rol").Trim().ToUpper() == "CLIENTE")
        {
            <li class="nav-item">

                <a class="nav-link text-dark" asp-area="" asp-
controller="Cliente" asp-action="IndexCliente">Inicio</a>

            </li>

```

```

        <li class="nav-item">

            <a class="nav-link text-dark" asp-area="" asp-
controller="Cliente" asp-action="VerPublicacionesCliente">Ver
publicaciones</a>

        </li>

        <li class="nav-item">

            <a class="nav-link text-dark" asp-area="" asp-
controller="Cliente" asp-action="CargarSaldo">Cargar saldo</a>

        </li>

        <li class="nav-item">

            <a class="nav-link text-dark" asp-area="" asp-
controller="Home" asp-action="Logout">Cerrar sesión</a>

        </li>
    }

    @if (Context.Session.GetString("usuario") != null &&
Context.Session.GetString("rol").Trim().ToUpper() ==
"ADMINISTRADOR")
    {
        <li class="nav-item">

            <a class="nav-link text-dark" asp-area="" asp-
controller="Administrador" asp-action="IndexAdmin">Inicio</a>

        </li>

        <li class="nav-item">

            <a class="nav-link text-dark" asp-area="" asp-
controller="Administrador" asp-action="VerSubastas">Ver
Subastas</a>

        </li>

        <li class="nav-item">

```

```
        <a class="nav-link text-dark" asp-area="" asp-
controller="Home" asp-action="Logout">Cerrar sesión</a>
    </li>
}
</ul>
</div>
</div>
</nav>
</header>
<div class="container">
    <main role="main" class="pb-3">
        @RenderBody()
    </main>
</div>

<footer class="border-top footer text-muted">
    <div class="container">
        &copy; 2024 - Web - <a asp-area="" asp-controller="Home" asp-
action="Privacy">Privacy</a>
    </div>
</footer>

<script src="~/lib/jquery/dist/jquery.min.js"></script>

<script
src="~/lib/bootstrap/dist/js/bootstrap.bundle.min.js"></script>

<script src="~/js/site.js" asp-append-version="true"></script>

@await RenderSectionAsync("Scripts", required: false)
</body>
```


</html>

Archivo: _ValidationScriptsPartial.cshtml

Carpeta: C:\Users\matia\Desktop\ORT\Programación
2\Obligatorio\Obligatorio\Web\Views\Shared_ValidationScriptsPartial
.cshtml

<script src="~/lib/jquery-
validation/dist/jquery.validate.min.js"></script>

<script src="~/lib/jquery-validation-
unobtrusive/jquery.validate.unobtrusive.min.js"></script>

Archivo: Ofertar.cshtml

Carpeta: C:\Users\matia\Desktop\ORT\Programación
2\Obligatorio\Obligatorio\Web\Views\Subasta\Ofertar.cshtml

@{

 ViewData["Title"] = "Ofertar";

}

<h1>Ofertar subasta</h1>

<form method="post">

 <label for="monto">Monto</label>

 <input type="number" name="monto" />


```
<input type="submit" class="btn btn-primary"
value="Ofertar"></input>

</form>
```

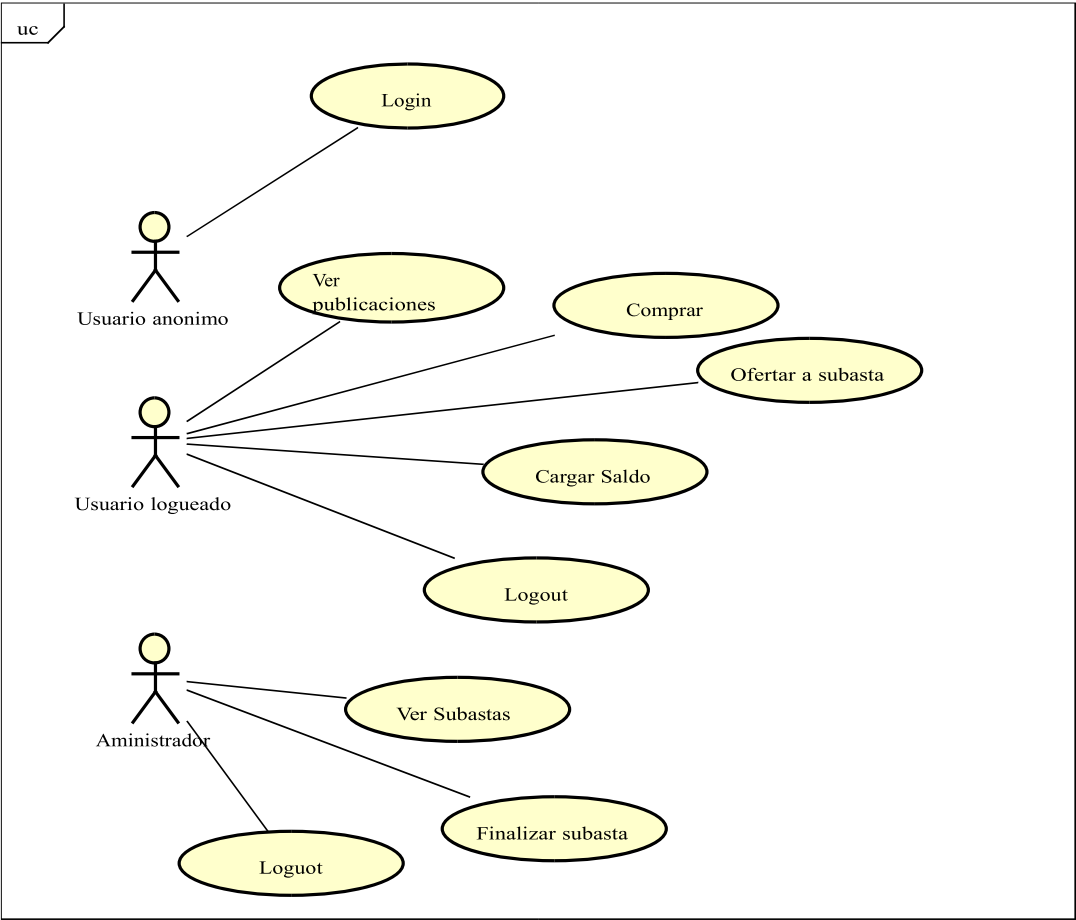
@ViewBag.Mensaje

Casos de prueba:

Función	Resultado esperado	Resultado obtenido
Registro	Si ingresa los datos incorrectos le muestra mensaje de error, o que no exista un usuario registrado con ese mail, si los datos son correctos el usuario queda registrado.	OK
Inicio de sesión, para usuarios registrados y administradores.	Para el usuario registrado que muestre las publicaciones, para los administradores que muestre el menú de subastas.	OK
Cargar saldo	El cliente ingresa el monto que desea depositar y queda actualizado en su billetera.	OK

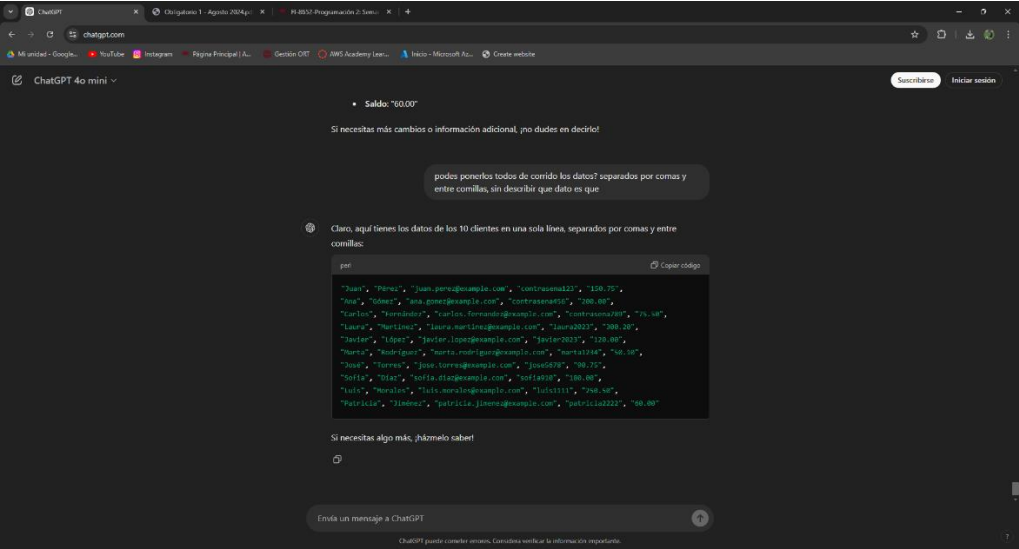
Comprar	El cliente realiza una compra, si tiene dinero suficiente será aceptada, sino no puede realizar la compra.	OK
Ofertar	El cliente realiza una oferta poniendo el monto, si este es válido se realizará con éxito.	OK
Cerrar subasta	El administrador una vez logueado, al ver las subastas puede cerrar una subasta siempre y cuando un cliente haya ofertado.	OK
Cerrar sesión	Tanto el cliente como el administrador tiene la opción de cerrar su sesión devolviendo la página inicial.	OK

Diagrama de casos de uso:

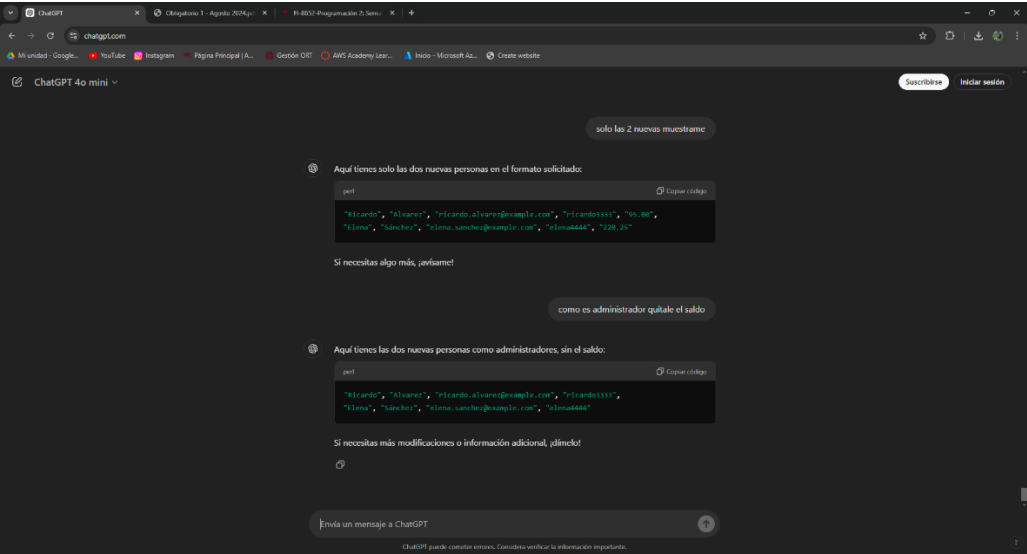


Link de acceso a AZURE:
obligatorio-f9h9ced9bmbfg7g3.canadacentral-01.azurewebsites.net

Consultas realizadas a ChatGPT:
Para los Clientes:



Para los Administradores:



Para los Artículos:

Me lo podrías poner en formato de código así: ["Nombre del producto", "categoria", precio]

Claro, aquí tienes la lista en el formato de código que solicitaste:

plaintext

[Copiar código](#)

```
[{"Smartphone X18", "Electrónica", 380.90}
{"Galleta de algodo", "Bebé", 25.50}
{"Anillo de plata", "Joyería", 120.70}
{"Zapatillas deportivas", "Calzado", 75.90}
{"Juego de sartenes", "Hogar", 220.00}
{"Pelota de fútbol", "Deportes", 30.00}
{"Perfume floral", "Belleza", 60.25}
{"Maceta de cerámica", "Jardinería", 35.00}
{"Bufanda de lana", "Accesorios", 55.10}
{"Tablet 10\"", "Electrónica", 150.00}
{"Vestido de verano", "Ropa", 85.00}
{"Collar de perlas", "Joyería", 200.00}
{"Botas de cuero", "Calzado", 120.00}
{"Lámpara LED", "Hogar", 95.00}
{"Bicicleta de montaña", "Deportes", 550.00}
{"Crema hidratante", "Belleza", 70.00}
{"Set de herramientas", "Jardinería", 45.20}
{"Gorro de invierno", "Accesorios", 33.00}
{"Auriculares Bluetooth", "Electrónica", 100.00}
{"Pantalones jeans", "Ropa", 55.25}
{"Pulsera de acero", "Joyería", 95.00}
{"Sandalias veraniegas", "Calzado", 65.70}
{"Juego de toallas", "Hogar", 150.00}
{"Ropa de tenis", "Deportes", 85.00}
```

Envía un mensaje a ChatGPT

ChatGPT puede cometer errores. Comprueba la información importante.