**Universidad ORT Uruguay**

**Facultad de Ingeniería**

**Escuela de Tecnología**

**OBLIGATORIO PROGRAMACION 2**

**DOCUMENTO DE ANÁLISIS**



**Ernaldo Rodríguez – 337244**



**Diego Geymonat – 300355**

**N2G**

**Docente: Luis Dentone**

**Analista en Tecnologías de la Información**

**Fecha de entrega del documento (10-10-2024)**

Tabla de contenido

[Diagrama de clases 3](#_Toc179314891)

[UML Astah 3](#_Toc179314892)

[Codigo comentado 4](#_Toc179314893)

[Program.cs 4](#_Toc179314894)

[Vistas.cs 4](#_Toc179314895)

[Sistema.cs 11](#_Toc179314896)

[Clases 24](#_Toc179314897)

[Administrador 24](#_Toc179314898)

[Articulo 24](#_Toc179314899)

[Categoria 25](#_Toc179314900)

[Cliente 26](#_Toc179314901)

[Estado (*enum)* 27](#_Toc179314902)

[Oferta 27](#_Toc179314903)

[Publicación 28](#_Toc179314904)

[Subasta 30](#_Toc179314905)

[Usuario 31](#_Toc179314906)

[Venta 32](#_Toc179314907)

[Interfaces 33](#_Toc179314908)

[IValidable 33](#_Toc179314909)

[Precargas 33](#_Toc179314910)

[Usuarios 33](#_Toc179314911)

[Categorías 34](#_Toc179314912)

[Artículos 35](#_Toc179314913)

[Publicaciones 37](#_Toc179314914)

[Ofertas 39](#_Toc179314915)

[Precargas con IA 40](#_Toc179314916)

[Prompt chat GPT precarga de Clientes 40](#_Toc179314917)

[Prompt chat GPT precarga de Administradores 40](#_Toc179314918)

[Prompt chat GPT precarga de Artículos 41](#_Toc179314919)

# Diagrama de clases

## UML Astah

# Codigo comentado

## Program.cs

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

Archivo: Program.cs

…\\ObligatorioP2\ConsolaApp\Program.cs

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

using Dominio;

namespace ConsolaApp

{

public class Program

{

public static Sistema \_sistema = new Sistema();

static void Main(string[] args)

{

try

{

Vistas.MenuInicio(); //MENU DEL PROGRAMA

}

catch (Exception e)

{

Console.WriteLine("\n" + e.Message);

}

}

}

}

## Vistas.cs

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

Archivo: Vistas.cs

…\\ObligatorioP2\ConsolaApp\Vistas.cs

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

using Dominio;

using Dominio.Entidades;

using System;

using System.Drawing;

namespace ConsolaApp

{

public class Vistas : Program

{

private delegate void Parametro1();

private static List<Parametro1> lista = new List<Parametro1>();

public static void MenuInicio()

{

Console.Clear();

ImprimirLogo();

int opcion = ConstructorMenu(["Ingresar PRECARGAS", "Listar Clientes", "Publicaciones entre fechas", "Articulos por categorìa", "Alta de artìculo", "Salir"]);

lista.Clear();

lista.Add(PreCargar);

lista.Add(ListarClientes);

lista.Add(ListarEntreFechas);

lista.Add(ListarArticulosFiltrado);

lista.Add(AgregarArticulos);

lista.Add(Salir);

lista[opcion]();

}

private static void PreCargar()

{

\_sistema.PreCargar();

TextoColor("green", "Precargas ingresadas. Presione cualquier tecla para continuar...");

Console.ReadKey();

MenuInicio();

}

private static void ListarEntreFechas()

{

Console.Clear();

Console.WriteLine("Filtrar Publicaciones entre dos fechas.");

Console.WriteLine("Ingrese una fecha de inicio (DD/MM/YYYY).");

DateTime fechaIncio = ObtenerFecha();

Console.WriteLine("Ingrese una fecha final (DD/MM/YYYY).");

DateTime fechaFinal = ObtenerFecha();

List<Publicacion> publicaciones = new List<Publicacion>(\_sistema.BuscarPublicacionEntreFecha(fechaIncio, fechaFinal));

if (publicaciones.Count == 0)

{

TextoColor("yellow", $"No hay Publicaciones entre {fechaIncio.ToShortDateString()} y " +

$"{fechaFinal.ToShortDateString()}");

}

else

{

foreach (Publicacion publicacion in publicaciones)

{

string tipo = publicacion.GetType().Name;

Console.WriteLine($"ID: {publicacion.Id}\nTIPO: {tipo}\nESTADO: {publicacion.EstadoPublicacion}\n" +

$"FECHA DE PUBLICACION: {publicacion.FechaPublicacion.ToShortDateString()}\n");

}

}

TextoColor("yellow", "Presione cualquier tecla para continuar...");

Console.ReadKey();

MenuInicio();

}

private static DateTime ObtenerFecha()

{

DateTime fecha = DateTime.Now;

bool flag = false;

do

{

try

{

fecha = DateTime.Parse(Console.ReadLine());

flag = true;

}

catch (Exception)

{

Console.WriteLine("Ingrese un fecha válida en formato DD/MM/YYYY");

}

} while (!flag);

return fecha;

}

private static void ListarClientes()

{

Console.Clear();

Console.WriteLine("CLIENTES\n");

List<Usuario> listaUsuarios = new(\_sistema.MostrarUsuarios(false));

if (listaUsuarios.Count == 0)

{

TextoColor("yellow", "No hay Clientes para mostrar.");

}

else

{

foreach (Usuario usuario in listaUsuarios)

{

Console.WriteLine(usuario.ToString() + "\n");

}

}

TextoColor("yellow", "Presione cualquier tecla para continuar...");

Console.ReadKey();

MenuInicio();

}

private static void ListarArticulosFiltrado()

{

Console.Clear();

Console.WriteLine("LISTAR ARTICULOS POR CATEGORIA");

Console.WriteLine("Seleccione una categoria");

List<string> categorias = new List<string>(MostrarCategoriasNombre());

if (categorias.Count == 0)

{

TextoColor("yellow", "No hay Categorìas para mostrar.");

Console.ReadKey();

MenuInicio();

}

else

{

categorias.Add("Volver");

int opcion = ConstructorMenu(categorias.ToArray());

categorias = MostrarCategoriasNombre();

List<Articulo> articulos = new List<Articulo>(\_sistema.MostrarArticulos(categorias[opcion]));

Console.Clear();

if (articulos.Count == 0)

{

TextoColor("yellow", "No hay Artículos para mostrar.");

Console.ReadKey();

MenuInicio();

}

else

{

foreach (Articulo articulo in articulos)

{

Console.WriteLine(articulo.ToString());

}

TextoColor("yellow", "Presione cualquier tecla para continuar...");

Console.ReadKey();

MenuInicio();

}

}

}

private static void AgregarArticulos()

{

Console.Clear();

Console.WriteLine("AGREGAR ARTICULO NUEVO");

string categoria = SeleccionarCategorias(\_sistema.MostrarCategorias());

if (categoria != "Cancelar")

{

TextoColor("yellow", $"Categoría elegida: {categoria}");

Console.WriteLine("Ingrese un nombre para el artículo:");

string nombre = Console.ReadLine()!;

decimal precio = PedirPrecio();

AceptarArticulo(categoria, nombre, precio);

}

else

{

MenuInicio();

}

}

private static decimal PedirPrecio()

{

decimal precio = 0;

bool flag = false;

do

{

try

{

Console.WriteLine("Ingrese el precio para el artículo:");

precio = decimal.Parse(Console.ReadLine());

flag = true;

}

catch (Exception)

{

Console.WriteLine("Ingrese solo números.");

}

} while (!flag);

return precio;

}

private static void AceptarArticulo(string categoria, string nombre, decimal precio)

{

Console.Clear();

Console.WriteLine("CONFIRMAR ARTICULO NUEVO");

TextoColor("yellow", $"\nNOMBRE : {nombre.ToUpper()}\nPRECIO : {precio}\nCATEGORIA : {categoria}\n");

int opcion = ConstructorMenu(["ACEPTAR", "CANCELAR"]);

if (opcion == 0 && !string.IsNullOrEmpty(nombre) && precio > 0)

{

\_sistema.AgregarArticulo(new Articulo(nombre, \_sistema.BuscarCategoria(categoria), precio));

Console.WriteLine("Producto agregado correctamente. Presione cualquier tecla para continuar...");

Console.ReadKey();

MenuInicio();

}

else if (opcion != 0 && !string.IsNullOrEmpty(nombre) && precio > 0)

{

TextoColor("yellow", "Cancelado. Presione cualquier tecla para continuar...");

Console.ReadKey();

MenuInicio();

}

else if (opcion == 0 || string.IsNullOrEmpty(nombre) || precio <= 0)

{

TextoColor("yellow", "Cancelado. Campo Categorias/Nombre vacias o precio incorrecto. Presione cualquier tecla para continuar...");

Console.ReadKey();

MenuInicio();

}

}

private static string SeleccionarCategorias(List<Categoria> categorias)

{

int opcion;

List<string> listaCategorias = new List<String>(MostrarCategoriasNombre());

listaCategorias.Add("Cancelar");

Console.Clear();

Console.WriteLine("Seleccione una categoria para agregar al nuevo artículo");

opcion = ConstructorMenu(listaCategorias.ToArray());

string categoria = listaCategorias[opcion];

return categoria;

}

//Bloque de muestra de listas como strings

private static List<string> MostrarCategoriasNombre()

{

List<string> listaCategorias = new List<string>();

foreach (Categoria item in \_sistema.MostrarCategorias())

{

listaCategorias.Add(item.Nombre.ToString());

}

return listaCategorias;

}

private static void Salir()

{

TextoColor("green", "Has salido del sistema...\n-> Presiona cualquier tecla para cerrar esta ventanta");

lista.Clear();

}

private static int ConstructorMenu(string[] opciones)

{

string menu = string.Empty;

for (int i = 0; i < opciones.Length - 1; i++)

{

menu += $"\n({i + 1})\_{opciones[i]}";

}

menu += $"\n(0)\_{opciones[opciones.Length - 1]}\n";

Console.WriteLine(menu);

return SelectorMenu(opciones.Length);

}

private static int SelectorMenu(int opciones)

{

int opcion = 0;

bool flag = false;

do

{

opcion = PedirNumero("Seleccione una opcion del menú", opciones);

flag = true;

if (opcion > opciones - 1) flag = false;

}

while (!flag);

if (opcion == 0) return opciones - 1;

return opcion - 1;

}

private static void ImprimirLogo()

{

string C = "█";//alt+219

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("\n\t=============================");

Console.ForegroundColor = ConsoleColor.Blue;

Console.WriteLine($"\t{C}{C}{C}{C} {C}{C}{C} {C}{C}{C} {C} {C} {C}{C}{C}{C}");

Console.WriteLine($"\t{C} {C} {C} {C} {C} {C} {C}");

Console.WriteLine($"\t{C}{C} {C} {C} {C} {C} {C} {C}{C}{C}");

Console.WriteLine($"\t{C} {C} {C} {C} {C} {C} {C}");

Console.WriteLine($"\t{C} {C}{C}{C} {C}{C}{C} {C}{C}{C} {C}{C}{C}{C}");

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("\t=============Tu Tienda Online\n");

Console.ForegroundColor = ConsoleColor.White;

}

private static void TextoColor(string color, string mensaje)

{

switch (color)

{

case "red":

Console.ForegroundColor = ConsoleColor.Red;

break;

case "green":

Console.ForegroundColor = ConsoleColor.Green;

break;

case "blue":

Console.ForegroundColor = ConsoleColor.Blue;

break;

case "yellow":

Console.ForegroundColor = ConsoleColor.Yellow;

break;

default:

break;

}

Console.WriteLine("\n" + mensaje);

Console.ForegroundColor = ConsoleColor.White;

}

private static int PedirNumero(string titulo, int opciones)

{

bool flag = false;

int numero = 0;

do

{

try

{

if (opciones > 10)

{

Console.WriteLine(titulo);

numero = int.Parse(Console.ReadLine()!);

}

else

{

ConsoleKeyInfo key = Console.ReadKey();

numero = int.Parse($"{key.KeyChar}");

}

flag = true;

}

catch (Exception)

{

TextoColor("red", "Ingrese solo numeros!!! Presione cualquier tecla para continuar...");

}

} while (!flag);

return numero;

}

}

}

## Sistema.cs

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

Archivo: Sistema.cs

…\\ObligatorioP2\Dominio\Sistema.cs

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

using Dominio.Entidades;

namespace Dominio

{

public class Sistema

{

private List<Usuario> \_usuarios = new List<Usuario>();

private List<Publicacion> \_publicaciones = new List<Publicacion>();

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

private List<Categoria> \_categorias = new List<Categoria>();

#region "PRE CARGAS DEL SISTEMA"

public void PreCargar()

{

PrecargarUsuarios();

PrecargarCategorias();

PrecargarArticulos();

PrecargarPublicaciones();

PrecargarOfertas();

}

private void PrecargarUsuarios()

{

//PROMT DE CHAT GPT

//con el siguiente formato

//AgregarCliente(new Cliente("Diego", "Geymonat", "dgeymonat85@gmail.com", "Geymon4t", 135000));

//genera 10 usuarios con mail ficticios

AgregarCliente(new Cliente("Diego", "Geymonat", "dgeymonat85@gmail.com", "Geymon4t", 135000));

AgregarCliente(new Cliente("Ana", "Martínez", "ana.martinez92@example.com", "AnaM4rt", 150000));

AgregarCliente(new Cliente("Luis", "Pérez", "luis.perez76@example.com", "Lui5Pez", 120000));

AgregarCliente(new Cliente("María", "Lopez", "maria.lopez88@example.com", "Mar1Lopez", 140000));

AgregarCliente(new Cliente("Carlos", "Sánchez", "carlos.sanchez80@example.com", "CarloS8", 160000));

AgregarCliente(new Cliente("Laura", "González", "laura.gonzalez91@example.com", "LauR4Gonz", 110000));

AgregarCliente(new Cliente("Javier", "Torres", "javier.torres89@example.com", "Jav1Torres", 130000));

AgregarCliente(new Cliente("Sofía", "Ramírez", "sofia.ramirez85@example.com", "Sofi4Rami", 155000));

AgregarCliente(new Cliente("Pedro", "Jiménez", "pedro.jimenez93@example.com", "Pedr0Jim", 145000));

AgregarCliente(new Cliente("Isabel", "Cruz", "isabel.cruz90@example.com", "IsabeL90", 125000));

AgregarCliente(new Cliente("Ernaldo", "Rodriguez", "ernaldo.rodriguez.dev@gmail.com", "3rn4ld0", 125000));

//PROMT DE CHAT GPT

//con el siguiente formato

//AgregarAdministrador(new Administrador("Diego", "Geymonat", "dgeymonat84@gmail.com", "Geymon4t"));

//genera 2 usuarios con mail ficticios

AgregarAdministrador(new Administrador("Diego", "Geymonat", "dgeymonat84@gmail.com", "Geymon4t"));

AgregarAdministrador(new Administrador("Lucía", "Fernández", "lucia.fernandez77@example.com", "Luc1aF"));

//PRECARGAS CON ERROR

// AgregarCliente(new Cliente("", "Geymonat", "dgeymonat85@gmail.com", "Geymon4t", 135000)); --Nombre, Apellido, Mail y Pasword no pueden estar vacios.

// AgregarCliente(new Cliente("Diego", "", "dgeymonat85@gmail.com", "Geymon4t", 135000)); --Nombre, Apellido, Mail y Pasword no pueden estar vacios.

// AgregarCliente(new Cliente("Diego", "Geymonat", "", "Geymon4t", 135000)); --Nombre, Apellido, Mail y Pasword no pueden estar vacios.

// AgregarCliente(new Cliente("Diego", "Geymonat", "dgeymonat85@gmail.com", "", 135000)); --Nombre, Apellido, Mail y Pasword no pueden estar vacios.

// AgregarCliente(new Cliente("Diego", "Geymonat", "dgey44monat85@gmail.com", "Geymon4t", -2)); --El saldo no puede ser negativo

//AgregarAdministrador(new Administrador("", "Rodriguez", "ernaldo.rodriguez.dev1@gmail.com", "1234")); --Nombre, Apellido, Mail y Pasword no pueden estar vacios.

//AgregarAdministrador(new Administrador("Ernaldo", "", "ernaldo.rodriguez.dev1@gmail.com", "1234")); --Nombre, Apellido, Mail y Pasword no pueden estar vacios.

//AgregarAdministrador(new Administrador("Ernaldo", "Rodriguez", "", "1234")); --Nombre, Apellido, Mail y Pasword no pueden estar vacios.

//AgregarAdministrador(new Administrador("Ernaldo", "Rodriguez", "ernaldo.rodriguez.dev1@gmail.com", "")); --Nombre, Apellido, Mail y Pasword no pueden estar vacios.

}

private void PrecargarCategorias()

{

AgregarCategoria(new Categoria("Deportes", "Todo lo necesario para el deporte"));

AgregarCategoria(new Categoria("Indumentaria", "Distintas prendas para distintas ocasiones"));

AgregarCategoria(new Categoria("Salud", "Todo para el cuidado"));

AgregarCategoria(new Categoria("Tecnología", "Celulares, computadoras, consolas y más..."));

AgregarCategoria(new Categoria("Niños", "Todo para los peques"));

AgregarCategoria(new Categoria("Hogar", "Accesorio para el hogar"));

AgregarCategoria(new Categoria("Entretenimiento", "Para tu tiempo libre"));

AgregarCategoria(new Categoria("Educación", "Para crecer y aprender"));

//PRECARGAS CON ERROR

//AgregarCategoria(new Categoria("", "Todo lo necesario para el deporte")); --Nombre de la Categoría, sin datos

//AgregarCategoria(new Categoria("Deportes", "")); --Decripción de la categoria, sin datos

}

private void PrecargarArticulos()

{

//PROMT DE CHAT GPT

//con las siguientes categorias :

//Deportes

//Indumentaria

//Salud

//Tecnología

//Niños

//Hogar

//Educación

//y con el siguiente formato

//AgregarArticulo(new Articulo("Zapatillas de correr", BuscarCategoria("Indumentaria"), 80));

//generar 50 articulos diferentes de bazar

AgregarArticulo(new Articulo("Zapatillas de correr", BuscarCategoria("Indumentaria"), 80));

AgregarArticulo(new Articulo("Pelota de fútbol", BuscarCategoria("Deportes"), 30));

AgregarArticulo(new Articulo("Botella de agua reutilizable", BuscarCategoria("Salud"), 15));

AgregarArticulo(new Articulo("Auriculares inalámbricos", BuscarCategoria("Tecnología"), 60));

AgregarArticulo(new Articulo("Juguete de bloques", BuscarCategoria("Niños"), 25));

AgregarArticulo(new Articulo("Juego de sábanas", BuscarCategoria("Hogar"), 50));

AgregarArticulo(new Articulo("Libro de matemáticas", BuscarCategoria("Educación"), 20));

AgregarArticulo(new Articulo("Raqueta de tenis", BuscarCategoria("Deportes"), 100));

AgregarArticulo(new Articulo("Mochila escolar", BuscarCategoria("Indumentaria"), 40));

AgregarArticulo(new Articulo("Suplemento vitamínico", BuscarCategoria("Salud"), 30));

AgregarArticulo(new Articulo("Smartwatch", BuscarCategoria("Tecnología"), 120));

AgregarArticulo(new Articulo("Juego de pintura", BuscarCategoria("Niños"), 45));

AgregarArticulo(new Articulo("Cojín decorativo", BuscarCategoria("Hogar"), 25));

AgregarArticulo(new Articulo("Cuaderno de notas", BuscarCategoria("Educación"), 10));

AgregarArticulo(new Articulo("Balón de baloncesto", BuscarCategoria("Deportes"), 40));

AgregarArticulo(new Articulo("Chaqueta impermeable", BuscarCategoria("Indumentaria"), 90));

AgregarArticulo(new Articulo("Té orgánico", BuscarCategoria("Salud"), 12));

AgregarArticulo(new Articulo("Cámara digital", BuscarCategoria("Tecnología"), 200));

AgregarArticulo(new Articulo("Rompecabezas de animales", BuscarCategoria("Niños"), 30));

AgregarArticulo(new Articulo("Lámpara de mesa", BuscarCategoria("Hogar"), 55));

AgregarArticulo(new Articulo("Libro de ciencias", BuscarCategoria("Educación"), 22));

AgregarArticulo(new Articulo("Patineta", BuscarCategoria("Deportes"), 80));

AgregarArticulo(new Articulo("Gorra deportiva", BuscarCategoria("Indumentaria"), 20));

AgregarArticulo(new Articulo("Aceite esencial", BuscarCategoria("Salud"), 18));

AgregarArticulo(new Articulo("Tablet", BuscarCategoria("Tecnología"), 250));

AgregarArticulo(new Articulo("Kits de ciencia", BuscarCategoria("Niños"), 35));

AgregarArticulo(new Articulo("Estante de libros", BuscarCategoria("Hogar"), 120));

AgregarArticulo(new Articulo("Diccionario ilustrado", BuscarCategoria("Educación"), 15));

AgregarArticulo(new Articulo("Set de pesas", BuscarCategoria("Deportes"), 70));

AgregarArticulo(new Articulo("Camiseta de deporte", BuscarCategoria("Indumentaria"), 25));

AgregarArticulo(new Articulo("Batido de proteínas", BuscarCategoria("Salud"), 35));

AgregarArticulo(new Articulo("Altavoz Bluetooth", BuscarCategoria("Tecnología"), 90));

AgregarArticulo(new Articulo("Muñeca de acción", BuscarCategoria("Niños"), 40));

AgregarArticulo(new Articulo("Organizador de cocina", BuscarCategoria("Hogar"), 30));

AgregarArticulo(new Articulo("Manual de gramática", BuscarCategoria("Educación"), 20));

AgregarArticulo(new Articulo("Bicicleta", BuscarCategoria("Deportes"), 500));

AgregarArticulo(new Articulo("Zapatos de vestir", BuscarCategoria("Indumentaria"), 100));

AgregarArticulo(new Articulo("Mascarilla facial", BuscarCategoria("Salud"), 10));

AgregarArticulo(new Articulo("PC portátil", BuscarCategoria("Tecnología"), 800));

AgregarArticulo(new Articulo("Set de arte", BuscarCategoria("Niños"), 45));

AgregarArticulo(new Articulo("Alfombra antideslizante", BuscarCategoria("Hogar"), 75));

AgregarArticulo(new Articulo("Atlas del mundo", BuscarCategoria("Educación"), 30));

AgregarArticulo(new Articulo("Kit de escalada", BuscarCategoria("Deportes"), 150));

AgregarArticulo(new Articulo("Pantalones deportivos", BuscarCategoria("Indumentaria"), 60));

AgregarArticulo(new Articulo("Protector solar", BuscarCategoria("Salud"), 20));

AgregarArticulo(new Articulo("Dispositivo de streaming", BuscarCategoria("Tecnología"), 100));

AgregarArticulo(new Articulo("Juego de mesa", BuscarCategoria("Niños"), 50));

AgregarArticulo(new Articulo("Cuchillos de cocina", BuscarCategoria("Hogar"), 80));

AgregarArticulo(new Articulo("Libro de literatura", BuscarCategoria("Educación"), 25));

AgregarArticulo(new Articulo("Tabla de surf", BuscarCategoria("Deportes"), 300));

AgregarArticulo(new Articulo("Bufanda de lana", BuscarCategoria("Indumentaria"), 50));

AgregarArticulo(new Articulo("Termómetro digital", BuscarCategoria("Salud"), 25));

AgregarArticulo(new Articulo("Router WiFi", BuscarCategoria("Tecnología"), 80));

AgregarArticulo(new Articulo("Juego de construcción", BuscarCategoria("Niños"), 55));

AgregarArticulo(new Articulo("Cama plegable", BuscarCategoria("Hogar"), 250));

AgregarArticulo(new Articulo("Libro de arte", BuscarCategoria("Educación"), 30));

//PRECARGAS CON ERROR

//AgregarArticulo(new Articulo("", BuscarCategoria("Indumentaria"), 80)); --En nombre no puede ser vacio o nulo

//AgregarArticulo(new Articulo("Zapatillas de correr", BuscarCategoria(""), 80)); --No se ha cargado la categoria en el parametro

//AgregarArticulo(new Articulo("Zapatillas de correr", BuscarCategoria("Indumentaria"), 0)); --El precio no puede ser cero o negativo

}

private void PrecargarPublicaciones()

{

//VENTAS

AgregarPublicacion(new Venta("Verano en la Playa", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(20), BuscarArticulo(25),

BuscarArticulo(42), BuscarArticulo(38)}, FechaRandom()));

AgregarPublicacion(new Venta("Mantente en forma", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(45), BuscarArticulo(38),

BuscarArticulo(25)}, FechaRandom()));

AgregarPublicacion(new Venta("Sal de tu casa", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> { BuscarArticulo(42), BuscarArticulo(25),

BuscarArticulo(20), BuscarArticulo(45) }, FechaRandom()));

AgregarPublicacion(new Venta("Luce el mejor look", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(26), BuscarArticulo(48),

BuscarArticulo(17), BuscarArticulo(9)}, FechaRandom()));

AgregarPublicacion(new Venta("Mantente saludable", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(46), BuscarArticulo(13),

BuscarArticulo(7)}, FechaRandom()));

AgregarPublicacion(new Venta("Para cuidar tu cuerpo", Estado. ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(43), BuscarArticulo(35),

BuscarArticulo(2), BuscarArticulo(32)}, FechaRandom()));

AgregarPublicacion(new Venta("Para quedarte en casa", Estado. ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(41), BuscarArticulo(8),

BuscarArticulo(10)}, FechaRandom()));

AgregarPublicacion(new Venta("Cultiva tu mente", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(31), BuscarArticulo(49),

BuscarArticulo(24), BuscarArticulo(5)}, FechaRandom()));

AgregarPublicacion(new Venta("Mantente actualizado", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(28), BuscarArticulo(22),

BuscarArticulo(12), BuscarArticulo(3)}, FechaRandom()));

AgregarPublicacion(new Venta("Lo que no te puede faltar", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(17), BuscarArticulo(9),

BuscarArticulo(1), BuscarArticulo(0)}, FechaRandom()));

//SUBASTAS

AgregarPublicacion(new Subasta("Verano en la Playa", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(4), BuscarArticulo(6),

BuscarArticulo(9), BuscarArticulo(45)}, FechaRandom()));

AgregarPublicacion(new Subasta("Mantente en forma", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(45), BuscarArticulo(38),

BuscarArticulo(25)}, FechaRandom()));

AgregarPublicacion(new Subasta("Sal de tu casa", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> { BuscarArticulo(42), BuscarArticulo(25),

BuscarArticulo(20), BuscarArticulo(45) }, FechaRandom()));

AgregarPublicacion(new Subasta("Luce el mejor look", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(26), BuscarArticulo(48),

BuscarArticulo(17), BuscarArticulo(9)}, FechaRandom()));

AgregarPublicacion(new Subasta("Mantente saludable", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(46), BuscarArticulo(13),

BuscarArticulo(7)}, FechaRandom()));

AgregarPublicacion(new Subasta("Para cuidar tu cuerpo", Estado. ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(43), BuscarArticulo(35),

BuscarArticulo(2), BuscarArticulo(32)}, FechaRandom()));

AgregarPublicacion(new Subasta("Para quedarte en casa", Estado. ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(41), BuscarArticulo(8),

BuscarArticulo(10)}, FechaRandom()));

AgregarPublicacion(new Subasta("Cultiva tu mente", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(31), BuscarArticulo(49),

BuscarArticulo(24), BuscarArticulo(5)}, FechaRandom()));

AgregarPublicacion(new Subasta("Mantente actualizado", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(28), BuscarArticulo(22),

BuscarArticulo(12), BuscarArticulo(3)}, FechaRandom()));

AgregarPublicacion(new Subasta("Lo que no te puede faltar", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(17), BuscarArticulo(9),

BuscarArticulo(1), BuscarArticulo(0)}, FechaRandom()));

//PRECARGAS CON ERROR

//AgregarPublicacion(new Venta("", Estado.ABIERTA,

// BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(20)}, FechaRandom())); --Datos incorrectos al intentar ingresar la PUBLICACIÓN.

//AgregarPublicacion(new Venta("Verano en la Playa", Estado.ABIERTA,

// BuscarAdministrador(""), new List<Articulo> { BuscarArticulo(20) }, FechaRandom())); --No se ha cargado el email en el parametro

//AgregarPublicacion(new Venta("Verano en la Playa", Estado.ABIERTA,

// BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> { BuscarArticulo(1540) }, FechaRandom())); --Articulo mal ingresado

//AgregarPublicacion(new Venta("Verano en la Playa", Estado.ABIERTA,

// BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> { BuscarArticulo(20) }, new DateTime(2025,12,20))); --La fecha de publicacion no puede ser mayor a la actual.

//AgregarPublicacion(new Venta("Verano en la Playa", Estado.ABIERTA,

// BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {}, FechaRandom())); --Debe seleccionar al menos un artículo para la publicacion

//AgregarPublicacion(new Subasta("", Estado.ABIERTA,

// BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(17), BuscarArticulo(9),

// BuscarArticulo(1), BuscarArticulo(0)}, FechaRandom())); ----Datos incorrectos al intentar ingresar la PUBLICACIÓN.

//AgregarPublicacion(new Subasta("Lo que no te puede faltar", Estado.ABIERTA,

// BuscarAdministrador(""), new List<Articulo> {BuscarArticulo(17), BuscarArticulo(9),

// BuscarArticulo(1), BuscarArticulo(0)}, FechaRandom())); ----No se ha cargado el email en el parametro

//AgregarPublicacion(new Subasta("Lo que no te puede faltar", Estado.ABIERTA,

// BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(17), BuscarArticulo(9),

// BuscarArticulo(1), BuscarArticulo(1000)}, FechaRandom())); ----Articulo mal ingresado

//AgregarPublicacion(new Subasta("Lo que no te puede faltar", Estado.ABIERTA,

// BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(17), BuscarArticulo(9),

// BuscarArticulo(1), BuscarArticulo(0)}, new DateTime(2025,12,20))); ----La fecha de publicacion no puede ser mayor a la actual.

// BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {}, FechaRandom())); ----Debe seleccionar al menos un artículo para la publicacion

}

private void PrecargarOfertas()

{

Subasta unaPublicacion = BuscarPublicacionSubasta(10);

unaPublicacion.CargarOferta(new Oferta(BuscarCliente("ernaldo.rodriguez.dev@gmail.com"), 1334));

unaPublicacion.CargarOferta(new Oferta(BuscarCliente("dgeymonat85@gmail.com"), 256));

unaPublicacion = BuscarPublicacionSubasta(19);

unaPublicacion.CargarOferta(new Oferta(BuscarCliente("ernaldo.rodriguez.dev@gmail.com"), 1334));

unaPublicacion.CargarOferta(new Oferta(BuscarCliente("dgeymonat85@gmail.com"), 256));

//Subasta unaPublicacion = BuscarPublicacionSubasta(1);

//unaPublicacion.CargarOferta(new Oferta(BuscarCliente("ernaldo.rodriguez.dev@gmail.com"), 1334)); --El id ingresado no es una subasta.

//Subasta unaPublicacion = BuscarPublicacionSubasta(20);

//unaPublicacion.CargarOferta(new Oferta(BuscarCliente("ernaldo.rodriguez.dev@gmail.com"), 1334));--La publicación no es válida.

//Subasta unaPublicacion = BuscarPublicacionSubasta(10);

//unaPublicacion.CargarOferta(new Oferta(BuscarCliente(""), 1334));--No se ha cargado el email en el parametro

//Subasta unaPublicacion = BuscarPublicacionSubasta(10);

//unaPublicacion.CargarOferta(new Oferta(BuscarCliente("dgeymonat84@gmail.com"), 1334))--El mail ingresado no corresponde a un cliente válido.

//Subasta unaPublicacion = BuscarPublicacionSubasta(10);

//unaPublicacion.CargarOferta(new Oferta(BuscarCliente("fghdfhdgh"), 1334)) --El mail ingresado no corresponde a un cliente válido.

}

private static DateTime FechaRandom()

{

bool flag = false;

DateTime fecha;

do

{

int anio = new Random().Next(2022, 2025);

int mes = new Random().Next(1, 13);

int dia = new Random().Next(1, 29);

fecha = new DateTime(anio, mes, dia);

DateTime fechaActual = DateTime.Now;

if (fechaActual > fecha)

flag = true;

} while (!flag);

return fecha;

}

#endregion

public Cliente BuscarCliente(string email)

{

if (String.IsNullOrEmpty(email)) throw new Exception("No se ha cargado el email en el parametro");

foreach (Usuario unUsuario in \_usuarios)

{

if (unUsuario.Email == email && unUsuario is Cliente)

{

return (Cliente)unUsuario;

}

}

return null!;

}

public Administrador BuscarAdministrador(string email)

{

if (String.IsNullOrEmpty(email)) throw new Exception("No se ha cargado el email en el parametro");

foreach (Usuario unUsuario in \_usuarios)

{

if (unUsuario.Email == email && unUsuario is Administrador)

{

return (Administrador)unUsuario;

}

}

return null!;

}

public Categoria BuscarCategoria(string categoria)

{

if (String.IsNullOrEmpty(categoria)) throw new Exception("No se ha cargado la categoria en el parametro");

foreach (Categoria unaCategoria in \_categorias)

{

if (unaCategoria.Nombre == categoria) return unaCategoria;

}

return null!;

}

public Articulo BuscarArticulo(int idArticulo)

{

if (!int.TryParse(idArticulo.ToString(), out \_)) throw new Exception("No se ha pasado un número");

foreach (Articulo unArticulo in \_articulos)

{

if (unArticulo.Id == idArticulo) return unArticulo;

}

return null!;

}

public Subasta BuscarPublicacionSubasta(int idPublicacion)

{

if (!int.TryParse(idPublicacion.ToString(), out \_)) throw new Exception("ingrese un id valido de publicacion");

foreach (Publicacion unaPublicacion in \_publicaciones)

{

if (unaPublicacion.Id == idPublicacion)

{

if (unaPublicacion is not Subasta) throw new Exception("El id ingresado no es una subasta");

return (Subasta)unaPublicacion;

}

}

throw new Exception("La publicación no es válida.");

}

public List<Usuario> MostrarUsuarios(bool admin)

{

List<Usuario> unaLista = new List<Usuario>();

foreach (Usuario usuario in \_usuarios)

{

if (usuario is Cliente && !admin)

{

unaLista.Add(usuario);

}

else if (usuario is Administrador && admin)

{

unaLista.Add(usuario);

}

}

return unaLista;

}

public List<Categoria> MostrarCategorias()

{

return \_categorias;

}

public List<Publicacion> ListaPublicaciones(string tipo, Estado estado)

{

if (string.IsNullOrEmpty(tipo)) throw new Exception("No se ha cargado el tipo en el parametro");

List<Publicacion> publicaciones = new List<Publicacion>();

foreach (Publicacion unaPublicacion in \_publicaciones)

{

if (tipo.ToUpper() == "TODOS")

{

if (estado == unaPublicacion.EstadoPublicacion)

{

publicaciones.Add(unaPublicacion);

}

else if (estado.ToString() == "TODOS")

{

publicaciones.Add(unaPublicacion);

}

}

else if (tipo.ToUpper() == "VENTA" && unaPublicacion is Venta)

{

if (estado == unaPublicacion.EstadoPublicacion)

{

publicaciones.Add(unaPublicacion);

}

else if (estado.ToString() == "TODOS")

{

publicaciones.Add(unaPublicacion);

}

}

else if (tipo.ToUpper() == "SUBASTA" && unaPublicacion is Subasta)

{

if (estado == unaPublicacion.EstadoPublicacion)

{

publicaciones.Add(unaPublicacion);

}

else if (estado.ToString() == "TODOS")

{

publicaciones.Add(unaPublicacion);

}

}

}

return publicaciones;

}

public List<Publicacion> BuscarPublicacionEntreFecha(DateTime fecha1, DateTime fecha2)

{

DateTime fechaMayor = fecha1;

if (fecha1> fecha2)

{

fecha1 = fecha2;

fecha2 = fechaMayor;

}

List<Publicacion> publicaciones = new List<Publicacion>();

foreach (Publicacion publicacion in \_publicaciones)

{

if (publicacion.FechaPublicacion >= fecha1 && publicacion.FechaPublicacion <= fecha2)

{

publicaciones.Add(publicacion);

}

}

return publicaciones;

}

public List<Articulo> MostrarArticulos(string categoria)

{

List<Articulo> resultado = new List<Articulo>();

if (categoria.ToUpper() == "TODOS")

{

foreach (Articulo unArticulo in \_articulos) resultado.Add(unArticulo);

}

else

{

foreach (Articulo unArticulo in \_articulos)

{

if (unArticulo.UnaCategoria.Nombre == categoria) resultado.Add(unArticulo);

}

}

return resultado;

}

//Bloque de Add en las listas

public void AgregarArticulo(Articulo nuevoArticulo)

{

if (nuevoArticulo == null) throw new Exception("Valor nulo en el parametro de agregar artículo");

nuevoArticulo.Validar();

\_articulos.Add(nuevoArticulo);

}

public void AgregarPublicacion(Publicacion nuevaPublicacion)

{

if (nuevaPublicacion == null) throw new Exception("Valor nulo en el parametro publicacion");

nuevaPublicacion.Validar();

\_publicaciones.Add(nuevaPublicacion);

}

public void AgregarCategoria(Categoria nuevaCategoria)

{

if (nuevaCategoria == null) throw new Exception("Valor nulo en el parametro de agregar categoría");

nuevaCategoria.Validar();

\_categorias.Add(nuevaCategoria);

}

public void AgregarCliente(Cliente nuevoUsuario)

{

if (nuevoUsuario == null) throw new Exception("Valor nulo en el parametro de agregar usuario");

nuevoUsuario.Validar();

if (BuscarCliente(nuevoUsuario.Email) != null) throw new Exception("El usuario ya existe");

\_usuarios.Add(nuevoUsuario);

}

public void AgregarAdministrador(Administrador nuevoUsuario)

{

if (nuevoUsuario == null) throw new Exception("Valor nulo en el parametro de agregar usuario");

nuevoUsuario.Validar();

if (BuscarAdministrador(nuevoUsuario.Email) != null) throw new Exception("El usuario ya existe");

\_usuarios.Add(nuevoUsuario);

}

}

}

## Clases

### Administrador

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

Archivo: Administrador.cs

…\\ObligatorioP2\Dominio\Entidades\Administrador.cs

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

using Dominio.Interfaces;

namespace Dominio.Entidades

{

public class Administrador : Usuario, IValidable

{

public Administrador(

string nombre,

string apellido,

string email,

string password): base(nombre,apellido,email,password)

{

}

public override string ToString()

{

return $"Id: {Id}\nNombre: {Nombre}\nApellido: {Apellido}\nEmail: {Email}\nTipo: Administrador";

}

Public override void Validar()

{

Base.Validar();

}

}

}

### Articulo

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

Archivo: Articulo.cs

…\\ObligatorioP2\Dominio\Entidades\Articulo.cs

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

using Dominio.Interfaces;

namespace Dominio.Entidades

{

public class Articulo : IValidable

{

public int Id { get; set; }

private static int \_ultimoId;

public string Nombre { get; set; }

public Categoria UnaCategoria { get; set; }

public decimal Precio { get; set; }

public Articulo(string nombre, Categoria categoria, decimal precio)

{

Id = \_ultimoId++;

Nombre = nombre;

UnaCategoria = categoria;

Precio = precio;

}

public override string ToString()

{

return $"Id : {Id}\nNombre : {Nombre.ToUpper()}\nPrecio : {Precio}\nCategoria : {UnaCategoria.Nombre}\n";

}

public void Validar()

{

if (String.IsNullOrEmpty(Nombre)) throw new Exception("El nombre no puede ser vacio o nulo");

if (UnaCategoria == null) throw new Exception("La categoria no puede ser nula");

if (Precio <= 0) throw new Exception("El precio no puede ser cero o negativo");

}

}

}

### Categoria

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

Archivo: Categoria.cs

…\\ObligatorioP2\Dominio\Entidades\Categoria.cs

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

using Dominio.Interfaces;

namespace Dominio.Entidades

{

public class Categoria:IValidable

{

private static int \_ultimoId;

public int Id;

public string Nombre { get; set; }

public string Descripcion { get; set; }

public Categoria(string nombre, string descripcion)

{

Id = \_ultimoId++;

Nombre = nombre;

Descripcion = descripcion;

}

public override string ToString()

{

return $"Id: {Id}\nCategoria: {Nombre.ToUpper()}\nDescripción : {Descripcion}\n";

}

public void Validar()

{

if (string.IsNullOrEmpty(Nombre)) throw new Exception("Nombre de la Categoría, sin datos");

if (string.IsNullOrEmpty(Descripcion)) throw new Exception("Decripción de la categoria, sin datos");

}

}

}

### Cliente

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

Archivo: Cliente.cs

…\\ObligatorioP2\Dominio\Entidades\Cliente.cs

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

using Dominio.Interfaces;

namespace Dominio.Entidades

{

public class Cliente : Usuario,IValidable

{

public decimal SaldoBilletera { get; set; }

public Cliente(

string nombre,

string apellido,

string email,

string password,

decimal Billetera) : base(nombre, apellido, email, password)

{

SaldoBilletera = Billetera;

}

public override string ToString()

{

string mensaje = string.Empty;

mensaje += $"Id: {Id}\nNombre: {Nombre}\nApellido: {Apellido}\nEmail: {Email}\nCuentaSaldo: {SaldoBilletera}\nTipo: Cliente";

return mensaje;

}

public override void Validar()

{

base.Validar();

if (SaldoBilletera < 0) throw new Exception("El saldo no puede ser negativo");

}

}

}

### Estado (*enum)*

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

Archivo: Estado.cs

…\\ObligatorioP2\Dominio\Entidades\Estado.cs

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

namespace Dominio.Entidades

{

public enum Estado

{

TODOS = -1

ABIERTA = 0,

CERRADA = 1,

CANCELADA = 2,

}

}

### Oferta

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

Archivo: Oferta.cs

…\\ObligatorioP2\Dominio\Entidades\Oferta.cs

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

using Dominio.Interfaces;

namespace Dominio.Entidades

{

public class Oferta:IValidable

{

public int Id { get; set; }

private static int \_ultimoId;

public decimal Precio { get; set; }

public Cliente Usuario;

public DateTime Fecha { get; set; }

public Oferta(Cliente usuario, decimal precio)

{

Id = \_ultimoId++;

Usuario = usuario;

Precio = precio;

Fecha = DateTime.Now;

}

public override string ToString()

{

return $"Id Oferta: {Id}\nDatos del Oferente:\n\t" +

$"Nombre: {Usuario.Nombre} {Usuario.Apellido}\n\t" +

$"Email: {Usuario.Email}\nMonto de la puja: ${Precio}\nRealizado el: {Fecha}\n\n";

}

public void Validar()

{

if (Precio < 0) throw new Exception("El precio no puede ser menor que cero");

if (Usuario == null) throw new Exception("El mail ingresado no corresponde a un cliente válido.");

}

}

}

### Publicación

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

Archivo: Publicacion.cs

…\\ObligatorioP2\Dominio\Entidades\Publicacion.cs

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

using Dominio.Interfaces;

namespace Dominio.Entidades

{

public abstract class Publicacion:IValidable

{

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

public int Id { get; set; }

private static int \_ultimoId;

public string Nombre { get; set; }

public DateTime FechaPublicacion { get; set; }

public DateTime FechaFinalizado { get; set; }

public Estado EstadoPublicacion { get; set; }

public Administrador Usuario { get; set; }

public Cliente UsuarioComprador { get; set; }

public Publicacion(string nombre,

Estado estado,

Administrador usuario,

List<Articulo> articulos,

DateTime fechaPublicacion)

{

Id = \_ultimoId++;

Nombre = nombre;

EstadoPublicacion = estado;

Usuario = usuario;

\_articulos = articulos;

FechaPublicacion = fechaPublicacion;

}

public override string ToString()

{

decimal precioPublicacion = 0;

string respuesta = string.Empty;

respuesta = $"Id: {Id}\n" +

$"Nombre: {Nombre}\n" +

$"Usuario: {Usuario.Nombre} {Usuario.Apellido}\n" +

$"Estado: {EstadoPublicacion}\n" +

$"Fecha Publicación: {FechaPublicacion.ToShortDateString()}\n";

if (EstadoPublicacion == Estado.CERRADA) respuesta += $"Fecha Finalizado: {FechaFinalizado.ToShortDateString()}\n";

respuesta += $"ARTICULOS:\n";

foreach (Articulo unArticulo in \_articulos)

{

respuesta += $"Id: {unArticulo.Id} " +

$"{unArticulo.Nombre} " +

$"${unArticulo.Precio}\n";

precioPublicacion += unArticulo.Precio;

}

respuesta += $"Total de la publicación: ${precioPublicacion}\n";

return respuesta;

}

public void AgregarArticuloProducto(Articulo articulo)

{

if (articulo == null) throw new Exception("Datos incorrectos al intentar agregar Articulos.");

\_articulos.Add(articulo);

}

public void QuitarArticuloProducto(Articulo articulo)

{

if (articulo == null) throw new Exception("Datos incorrectos al intentar quitar Articulos.");

\_articulos.Remove(articulo);

}

Public virtual void Validar()

{

if (string.IsNullOrEmpty(Nombre) ||

Usuario == null ||

string.IsNullOrEmpty(FechaPublicacion.ToString()))

{

throw new Exception("Datos incorrectos al intentar ingresar la PUBLICACIÓN.");

}

if (\_articulos.Count == 0) throw new Exception("Debe seleccionar al menos un artículo para la publicacion");

if (FechaPublicacion > DateTime.Now) throw new Exception("La fecha de publicacion no puede ser mayor a la actual.");

foreach(Articulo unArticulo in \_articulos)

{

if (unArticulo == null) throw new Exception("Articulo mal ingresado");

}

}

}

}

### Subasta

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

Archivo: Subasta.cs

…\\ObligatorioP2\Dominio\Entidades\Subasta.cs

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

using Dominio.Interfaces;

namespace Dominio.Entidades

{

public class Subasta : Publicacion, IValidable

{

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

public Subasta(string nombre,

Estado estado,

Administrador usuario,

List<Articulo> articulos,

DateTime fechaPublicacion

) : base(nombre, estado, usuario, articulos,fechaPublicacion)

{

}

public List<Oferta> MostrarOfertas()

{

return \_ofertas;

}

public void CargarOferta(Oferta oferta)

{

if (oferta == null) throw new Exception("Parametro incorrecto para crear una oferta");

oferta.Validar();

\_ofertas.Add(oferta);

}

}

}

### Usuario

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

Archivo: Usuario.cs

…\\ObligatorioP2\Dominio\Entidades\Usuario.cs

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

using Dominio.Interfaces;

namespace Dominio.Entidades

{

public abstract class Usuario : IValidable

{

public int Id { get; set; }

public string Nombre { get; set; }

public string Apellido { get; set; }

public string Email { get; set; }

private string \_password;

private static int \_ultimoId;

public string Password

{

set { \_password = value; }

}

public Usuario(string nombre,

string apellido,

string email,

string password)

{

Id = \_ultimoId++;

Nombre = nombre;

Apellido = apellido;

Email = email;

\_password = password;

}

public virtual void Validar()

{

if (string.IsNullOrEmpty(Nombre) || string.IsNullOrEmpty(Apellido) || string.IsNullOrEmpty(Email) ||

string.IsNullOrEmpty(\_password))

throw new Exception("Nombre, Apellido, Mail y Pasword no pueden estar vacios.");

ValidarPassword(\_password);

}

public void ValidarPassword(string password)

{

}

public abstract override string ToString();

}

}

### Venta

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

Archivo: Venta.cs

…\\ObligatorioP2\Dominio\Entidades\Venta.cs

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

namespace Dominio.Entidades

{

public class Venta : Publicacion, IValidable

{

public bool Oferta {get;set;}

public Venta(string nombre,

Estado estado,

Administrador usuario,

List<Articulo> articulos,

DateTime fechaPublicacion) : base(nombre,

estado,

usuario,

articulos,

fechaPublicacion)

{

Oferta = false;

}

Public override void Validar()

{

Base.Validar();

}

}

}

## Interfaces

### IValidable

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

Archivo: IValidable.cs

…\\ObligatorioP2\Dominio\Interfaces\IValidable.cs

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

namespace Dominio.Interfaces

{

public interface IValidable

{

public void Validar();

}

}

## Precargas

### Usuarios

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

private void PrecargarUsuarios()

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

{

//PROMT DE CHAT GPT

//con el siguiente formato

//AgregarCliente(new Cliente("Diego", "Geymonat", "dgeymonat85@gmail.com", "Geymon4t", 135000));

//genera 10 usuarios con mail ficticios

AgregarCliente(new Cliente("Diego", "Geymonat", "dgeymonat85@gmail.com", "Geymon4t", 135000));

AgregarCliente(new Cliente("Ana", "Martínez", "ana.martinez92@example.com", "AnaM4rt", 150000));

AgregarCliente(new Cliente("Luis", "Pérez", "luis.perez76@example.com", "Lui5Pez", 120000));

AgregarCliente(new Cliente("María", "Lopez", "maria.lopez88@example.com", "Mar1Lopez", 140000));

AgregarCliente(new Cliente("Carlos", "Sánchez", "carlos.sanchez80@example.com", "CarloS8", 160000));

AgregarCliente(new Cliente("Laura", "González", "laura.gonzalez91@example.com", "LauR4Gonz", 110000));

AgregarCliente(new Cliente("Javier", "Torres", "javier.torres89@example.com", "Jav1Torres", 130000));

AgregarCliente(new Cliente("Sofía", "Ramírez", "sofia.ramirez85@example.com", "Sofi4Rami", 155000));

AgregarCliente(new Cliente("Pedro", "Jiménez", "pedro.jimenez93@example.com", "Pedr0Jim", 145000));

AgregarCliente(new Cliente("Isabel", "Cruz", "isabel.cruz90@example.com", "IsabeL90", 125000));

AgregarCliente(new Cliente("Ernaldo", "Rodriguez", "ernaldo.rodriguez.dev@gmail.com", "3rn4ld0", 125000));

//PROMT DE CHAT GPT

//con el siguiente formato

//AgregarAdministrador(new Administrador("Diego", "Geymonat", "dgeymonat84@gmail.com", "Geymon4t"));

//genera 2 usuarios con mail ficticios

AgregarAdministrador(new Administrador("Diego", "Geymonat", "dgeymonat84@gmail.com", "Geymon4t"));

AgregarAdministrador(new Administrador("Lucía", "Fernández", "lucia.fernandez77@example.com", "Luc1aF"));

//PRECARGAS CON ERROR

// AgregarCliente(new Cliente("", "Geymonat", "dgeymonat85@gmail.com", "Geymon4t", 135000)); --Nombre, Apellido, Mail y Pasword no pueden estar vacios.

// AgregarCliente(new Cliente("Diego", "", "dgeymonat85@gmail.com", "Geymon4t", 135000)); --Nombre, Apellido, Mail y Pasword no pueden estar vacios.

// AgregarCliente(new Cliente("Diego", "Geymonat", "", "Geymon4t", 135000)); --Nombre, Apellido, Mail y Pasword no pueden estar vacios.

// AgregarCliente(new Cliente("Diego", "Geymonat", "dgeymonat85@gmail.com", "", 135000)); --Nombre, Apellido, Mail y Pasword no pueden estar vacios.

// AgregarCliente(new Cliente("Diego", "Geymonat", "dgey44monat85@gmail.com", "Geymon4t", -2)); --El saldo no puede ser negativo

//AgregarAdministrador(new Administrador("", "Rodriguez", "ernaldo.rodriguez.dev1@gmail.com", "1234")); --Nombre, Apellido, Mail y Pasword no pueden estar vacios.

//AgregarAdministrador(new Administrador("Ernaldo", "", "ernaldo.rodriguez.dev1@gmail.com", "1234")); --Nombre, Apellido, Mail y Pasword no pueden estar vacios.

//AgregarAdministrador(new Administrador("Ernaldo", "Rodriguez", "", "1234")); --Nombre, Apellido, Mail y Pasword no pueden estar vacios.

//AgregarAdministrador(new Administrador("Ernaldo", "Rodriguez", "ernaldo.rodriguez.dev1@gmail.com", "")); --Nombre, Apellido, Mail y Pasword no pueden estar vacios.

}

### Categorías

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

private void PrecargarCategorias()

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

{

AgregarCategoria(new Categoria("Deportes", "Todo lo necesario para el deporte"));

AgregarCategoria(new Categoria("Indumentaria", "Distintas prendas para distintas ocasiones"));

AgregarCategoria(new Categoria("Salud", "Todo para el cuidado"));

AgregarCategoria(new Categoria("Tecnología", "Celulares, computadoras, consolas y más..."));

AgregarCategoria(new Categoria("Niños", "Todo para los peques"));

AgregarCategoria(new Categoria("Hogar", "Accesorio para el hogar"));

AgregarCategoria(new Categoria("Entretenimiento", "Para tu tiempo libre"));

AgregarCategoria(new Categoria("Educación", "Para crecer y aprender"));

//PRECARGAS CON ERROR

//AgregarCategoria(new Categoria("", "Todo lo necesario para el deporte")); --Nombre de la Categoría, sin datos

//AgregarCategoria(new Categoria("Deportes", "")); --Decripción de la categoria, sin datos

}

### Artículos

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

private void PrecargarArticulos()

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

{

//PROMT DE CHAT GPT

//con las siguientes categorias :

//Deportes

//Indumentaria

//Salud

//Tecnología

//Niños

//Hogar

//Educación

//y con el siguiente formato

//AgregarArticulo(new Articulo("Zapatillas de correr", BuscarCategoria("Indumentaria"), 80));

//generar 50 articulos diferentes de bazar

AgregarArticulo(new Articulo("Zapatillas de correr", BuscarCategoria("Indumentaria"), 80));

AgregarArticulo(new Articulo("Pelota de fútbol", BuscarCategoria("Deportes"), 30));

AgregarArticulo(new Articulo("Botella de agua reutilizable", BuscarCategoria("Salud"), 15));

AgregarArticulo(new Articulo("Auriculares inalámbricos", BuscarCategoria("Tecnología"), 60));

AgregarArticulo(new Articulo("Juguete de bloques", BuscarCategoria("Niños"), 25));

AgregarArticulo(new Articulo("Juego de sábanas", BuscarCategoria("Hogar"), 50));

AgregarArticulo(new Articulo("Libro de matemáticas", BuscarCategoria("Educación"), 20));

AgregarArticulo(new Articulo("Raqueta de tenis", BuscarCategoria("Deportes"), 100));

AgregarArticulo(new Articulo("Mochila escolar", BuscarCategoria("Indumentaria"), 40));

AgregarArticulo(new Articulo("Suplemento vitamínico", BuscarCategoria("Salud"), 30));

AgregarArticulo(new Articulo("Smartwatch", BuscarCategoria("Tecnología"), 120));

AgregarArticulo(new Articulo("Juego de pintura", BuscarCategoria("Niños"), 45));

AgregarArticulo(new Articulo("Cojín decorativo", BuscarCategoria("Hogar"), 25));

AgregarArticulo(new Articulo("Cuaderno de notas", BuscarCategoria("Educación"), 10));

AgregarArticulo(new Articulo("Balón de baloncesto", BuscarCategoria("Deportes"), 40));

AgregarArticulo(new Articulo("Chaqueta impermeable", BuscarCategoria("Indumentaria"), 90));

AgregarArticulo(new Articulo("Té orgánico", BuscarCategoria("Salud"), 12));

AgregarArticulo(new Articulo("Cámara digital", BuscarCategoria("Tecnología"), 200));

AgregarArticulo(new Articulo("Rompecabezas de animales", BuscarCategoria("Niños"), 30));

AgregarArticulo(new Articulo("Lámpara de mesa", BuscarCategoria("Hogar"), 55));

AgregarArticulo(new Articulo("Libro de ciencias", BuscarCategoria("Educación"), 22));

AgregarArticulo(new Articulo("Patineta", BuscarCategoria("Deportes"), 80));

AgregarArticulo(new Articulo("Gorra deportiva", BuscarCategoria("Indumentaria"), 20));

AgregarArticulo(new Articulo("Aceite esencial", BuscarCategoria("Salud"), 18));

AgregarArticulo(new Articulo("Tablet", BuscarCategoria("Tecnología"), 250));

AgregarArticulo(new Articulo("Kits de ciencia", BuscarCategoria("Niños"), 35));

AgregarArticulo(new Articulo("Estante de libros", BuscarCategoria("Hogar"), 120));

AgregarArticulo(new Articulo("Diccionario ilustrado", BuscarCategoria("Educación"), 15));

AgregarArticulo(new Articulo("Set de pesas", BuscarCategoria("Deportes"), 70));

AgregarArticulo(new Articulo("Camiseta de deporte", BuscarCategoria("Indumentaria"), 25));

AgregarArticulo(new Articulo("Batido de proteínas", BuscarCategoria("Salud"), 35));

AgregarArticulo(new Articulo("Altavoz Bluetooth", BuscarCategoria("Tecnología"), 90));

AgregarArticulo(new Articulo("Muñeca de acción", BuscarCategoria("Niños"), 40));

AgregarArticulo(new Articulo("Organizador de cocina", BuscarCategoria("Hogar"), 30));

AgregarArticulo(new Articulo("Manual de gramática", BuscarCategoria("Educación"), 20));

AgregarArticulo(new Articulo("Bicicleta", BuscarCategoria("Deportes"), 500));

AgregarArticulo(new Articulo("Zapatos de vestir", BuscarCategoria("Indumentaria"), 100));

AgregarArticulo(new Articulo("Mascarilla facial", BuscarCategoria("Salud"), 10));

AgregarArticulo(new Articulo("PC portátil", BuscarCategoria("Tecnología"), 800));

AgregarArticulo(new Articulo("Set de arte", BuscarCategoria("Niños"), 45));

AgregarArticulo(new Articulo("Alfombra antideslizante", BuscarCategoria("Hogar"), 75));

AgregarArticulo(new Articulo("Atlas del mundo", BuscarCategoria("Educación"), 30));

AgregarArticulo(new Articulo("Kit de escalada", BuscarCategoria("Deportes"), 150));

AgregarArticulo(new Articulo("Pantalones deportivos", BuscarCategoria("Indumentaria"), 60));

AgregarArticulo(new Articulo("Protector solar", BuscarCategoria("Salud"), 20));

AgregarArticulo(new Articulo("Dispositivo de streaming", BuscarCategoria("Tecnología"), 100));

AgregarArticulo(new Articulo("Juego de mesa", BuscarCategoria("Niños"), 50));

AgregarArticulo(new Articulo("Cuchillos de cocina", BuscarCategoria("Hogar"), 80));

AgregarArticulo(new Articulo("Libro de literatura", BuscarCategoria("Educación"), 25));

AgregarArticulo(new Articulo("Tabla de surf", BuscarCategoria("Deportes"), 300));

AgregarArticulo(new Articulo("Bufanda de lana", BuscarCategoria("Indumentaria"), 50));

AgregarArticulo(new Articulo("Termómetro digital", BuscarCategoria("Salud"), 25));

AgregarArticulo(new Articulo("Router WiFi", BuscarCategoria("Tecnología"), 80));

AgregarArticulo(new Articulo("Juego de construcción", BuscarCategoria("Niños"), 55));

AgregarArticulo(new Articulo("Cama plegable", BuscarCategoria("Hogar"), 250));

AgregarArticulo(new Articulo("Libro de arte", BuscarCategoria("Educación"), 30));

//PRECARGAS CON ERROR

//AgregarArticulo(new Articulo("", BuscarCategoria("Indumentaria"), 80)); --En nombre no puede ser vacio o nulo

//AgregarArticulo(new Articulo("Zapatillas de correr", BuscarCategoria(""), 80)); --No se ha cargado la categoria en el parametro

//AgregarArticulo(new Articulo("Zapatillas de correr", BuscarCategoria("Indumentaria"), 0)); --El precio no puede ser cero o negativo

}

### Publicaciones

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

private void PrecargarPublicaciones()

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

{

//VENTAS

AgregarPublicacion(new Venta("Verano en la Playa", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(20), BuscarArticulo(25),

BuscarArticulo(42), BuscarArticulo(38)}, FechaRandom()));

AgregarPublicacion(new Venta("Mantente en forma", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(45), BuscarArticulo(38),

BuscarArticulo(25)}, FechaRandom()));

AgregarPublicacion(new Venta("Sal de tu casa", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> { BuscarArticulo(42), BuscarArticulo(25),

BuscarArticulo(20), BuscarArticulo(45) }, FechaRandom()));

AgregarPublicacion(new Venta("Luce el mejor look", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(26), BuscarArticulo(48),

BuscarArticulo(17), BuscarArticulo(9)}, FechaRandom()));

AgregarPublicacion(new Venta("Mantente saludable", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(46), BuscarArticulo(13),

BuscarArticulo(7)}, FechaRandom()));

AgregarPublicacion(new Venta("Para cuidar tu cuerpo", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(43), BuscarArticulo(35),

BuscarArticulo(2), BuscarArticulo(32)}, FechaRandom()));

AgregarPublicacion(new Venta("Para quedarte en casa", Estado. ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(41), BuscarArticulo(8),

BuscarArticulo(10)}, FechaRandom()));

AgregarPublicacion(new Venta("Cultiva tu mente", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(31), BuscarArticulo(49),

BuscarArticulo(24), BuscarArticulo(5)}, FechaRandom()));

AgregarPublicacion(new Venta("Mantente actualizado", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(28), BuscarArticulo(22),

BuscarArticulo(12), BuscarArticulo(3)}, FechaRandom()));

AgregarPublicacion(new Venta("Lo que no te puede faltar", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(17), BuscarArticulo(9),

BuscarArticulo(1), BuscarArticulo(0)}, FechaRandom()));

//SUBASTAS

AgregarPublicacion(new Subasta("Verano en la Playa", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(4), BuscarArticulo(6),

BuscarArticulo(9), BuscarArticulo(45)}, FechaRandom()));

AgregarPublicacion(new Subasta("Mantente en forma", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(45), BuscarArticulo(38),

BuscarArticulo(25)}, FechaRandom()));

AgregarPublicacion(new Subasta("Sal de tu casa", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> { BuscarArticulo(42), BuscarArticulo(25),

BuscarArticulo(20), BuscarArticulo(45) }, FechaRandom()));

AgregarPublicacion(new Subasta("Luce el mejor look", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(26), BuscarArticulo(48),

BuscarArticulo(17), BuscarArticulo(9)}, FechaRandom()));

AgregarPublicacion(new Subasta("Mantente saludable", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(46), BuscarArticulo(13),

BuscarArticulo(7)}, FechaRandom()));

AgregarPublicacion(new Subasta("Para cuidar tu cuerpo", Estado. ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(43), BuscarArticulo(35),

BuscarArticulo(2), BuscarArticulo(32)}, FechaRandom()));

AgregarPublicacion(new Subasta("Para quedarte en casa", Estado. ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(41), BuscarArticulo(8),

BuscarArticulo(10)}, FechaRandom()));

AgregarPublicacion(new Subasta("Cultiva tu mente", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(31), BuscarArticulo(49),

BuscarArticulo(24), BuscarArticulo(5)}, FechaRandom()));

AgregarPublicacion(new Subasta("Mantente actualizado", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(28), BuscarArticulo(22),

BuscarArticulo(12), BuscarArticulo(3)}, FechaRandom()));

AgregarPublicacion(new Subasta("Lo que no te puede faltar", Estado.ABIERTA,

BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(17), BuscarArticulo(9),

BuscarArticulo(1), BuscarArticulo(0)}, FechaRandom()));

//PRECARGAS CON ERROR

//AgregarPublicacion(new Venta("", Estado.ABIERTA,

// BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(20)}, FechaRandom())); --Datos incorrectos al intentar ingresar la PUBLICACIÓN.

//AgregarPublicacion(new Venta("Verano en la Playa", Estado.ABIERTA,

// BuscarAdministrador(""), new List<Articulo> { BuscarArticulo(20) }, FechaRandom())); --No se ha cargado el email en el parametro

//AgregarPublicacion(new Venta("Verano en la Playa", Estado.ABIERTA,

// BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> { BuscarArticulo(1540) }, FechaRandom())); --Articulo mal ingresado

//AgregarPublicacion(new Venta("Verano en la Playa", Estado.ABIERTA,

// BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> { BuscarArticulo(20) }, new DateTime(2025,12,20))); --La fecha de publicacion no puede ser mayor a la actual.

//AgregarPublicacion(new Venta("Verano en la Playa", Estado.ABIERTA,

// BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {}, FechaRandom())); --Debe seleccionar al menos un artículo para la publicacion

//AgregarPublicacion(new Subasta("", Estado.ABIERTA,

// BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(17), BuscarArticulo(9),

// BuscarArticulo(1), BuscarArticulo(0)}, FechaRandom())); ----Datos incorrectos al intentar ingresar la PUBLICACIÓN.

//AgregarPublicacion(new Subasta("Lo que no te puede faltar", Estado.ABIERTA,

// BuscarAdministrador(""), new List<Articulo> {BuscarArticulo(17), BuscarArticulo(9),

// BuscarArticulo(1), BuscarArticulo(0)}, FechaRandom())); ----No se ha cargado el email en el parametro

//AgregarPublicacion(new Subasta("Lo que no te puede faltar", Estado.ABIERTA,

// BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(17), BuscarArticulo(9),

// BuscarArticulo(1), BuscarArticulo(1000)}, FechaRandom())); ----Articulo mal ingresado

//AgregarPublicacion(new Subasta("Lo que no te puede faltar", Estado.ABIERTA,

// BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {BuscarArticulo(17), BuscarArticulo(9),

// BuscarArticulo(1), BuscarArticulo(0)}, new DateTime(2025,12,20))); ----La fecha de publicacion no puede ser mayor a la actual.

// BuscarAdministrador("dgeymonat84@gmail.com"), new List<Articulo> {}, FechaRandom())); ----Debe seleccionar al menos un artículo para la publicacion

}

### Ofertas

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

private void PrecargarOfertas()

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

{

Subasta unaPublicacion = BuscarPublicacionSubasta(10);

unaPublicacion.CargarOferta(new Oferta(BuscarCliente("ernaldo.rodriguez.dev@gmail.com"), 1334));

unaPublicacion.CargarOferta(new Oferta(BuscarCliente("dgeymonat85@gmail.com"), 256));

unaPublicacion = BuscarPublicacionSubasta(19);

unaPublicacion.CargarOferta(new Oferta(BuscarCliente("ernaldo.rodriguez.dev@gmail.com"), 1334));

unaPublicacion.CargarOferta(new Oferta(BuscarCliente("dgeymonat85@gmail.com"), 256));

//Subasta unaPublicacion = BuscarPublicacionSubasta(1);

//unaPublicacion.CargarOferta(new Oferta(BuscarCliente("ernaldo.rodriguez.dev@gmail.com"), 1334)); --El id ingresado no es una subasta.

//Subasta unaPublicacion = BuscarPublicacionSubasta(20);

//unaPublicacion.CargarOferta(new Oferta(BuscarCliente("ernaldo.rodriguez.dev@gmail.com"), 1334));--La publicación no es válida.

//Subasta unaPublicacion = BuscarPublicacionSubasta(10);

//unaPublicacion.CargarOferta(new Oferta(BuscarCliente(""), 1334));--No se ha cargado el email en el parametro

//Subasta unaPublicacion = BuscarPublicacionSubasta(10);

//unaPublicacion.CargarOferta(new Oferta(BuscarCliente("dgeymonat84@gmail.com"), 1334))--El mail ingresado no corresponde a un cliente válido.

//Subasta unaPublicacion = BuscarPublicacionSubasta(10);

//unaPublicacion.CargarOferta(new Oferta(BuscarCliente("fghdfhdgh"), 1334)) --El mail ingresado no corresponde a un cliente válido.

}

## Precargas con IA

### Prompt chat GPT precarga de Clientes

*con el siguiente formato*

*AgregarCliente(new Cliente("Diego", "Geymonat", "dgeymonat85@gmail.com", "Geymon4t", 135000));*

*genera 10 usuarios con mail ficticios*

### Prompt chat GPT precarga de Administradores

*con el siguiente formato*

*AgregarAdministrador(new Administrador("Diego", "Geymonat", "dgeymonat84@gmail.com", "Geymon4t"));*

*genera 2 usuarios con mail ficticios*

*AgregarAdministrador(new Administrador("Diego", "Geymonat", "dgeymonat84@gmail.com", "Geymon4t"));*

### Prompt chat GPT precarga de Artículos

*con las siguientes categorias :*

*Deportes*

*Indumentaria*

*Salud*

*Tecnología*

*Niños*

*Hogar*

*Educación*

*y con el siguiente formato*

*AgregarArticulo(new Articulo("Zapatillas de correr", BuscarCategoria("Indumentaria"), 80));*

*generar 50 articulos diferentes de bazar*