**Programación Sobre Redes**

**TPO Final**

**SALA DE CHAT**

**Nombre del Grupo: A LO BUKELE**

**Integrantes: Cores Juan, Quispe Villalobos Carolina, Torrico Lucas y Lumbrera Joaquin.**

**División: 6 - 1**

**Profesor: Cortinez Franco Matias**

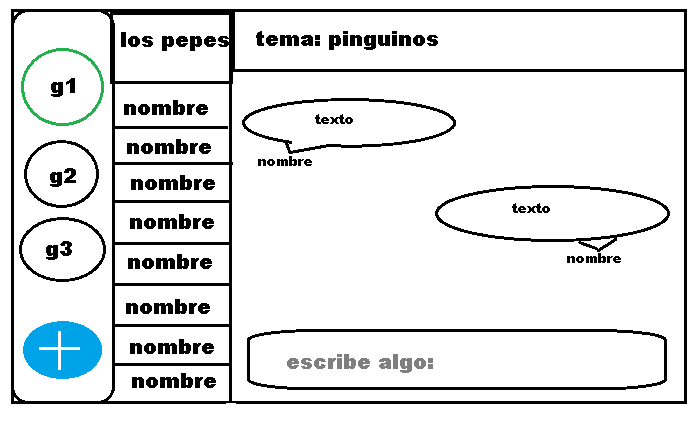
**Materia: Programación Sobre Redes**

**Entrega 1.3**

El tema a desarrollar es una sala de chat tanto grupal como individual, lo que haremos será una especie de red social en la cual los usuarios se conectaran a salas creadas por ellos mismos o hechas por otros usuarios, el servidor funcionará como redireccionador a estas salas, que estarán guardadas en un array para que al momento del usuario buscar una de estas salas solo tenga que ingresar el nombre de la misma para poder acceder a ella, una vez conectados a estas salas podrán comenzar a enviarse mensajes entre ellos.

Utilizaremos C# para desarrollar el código usando sockets e hilos creando así la conexión entre cliente y servidor.

**Vista del usuario al iniciar la aplicación:**



(Muestra de como se ve el principio del programa):

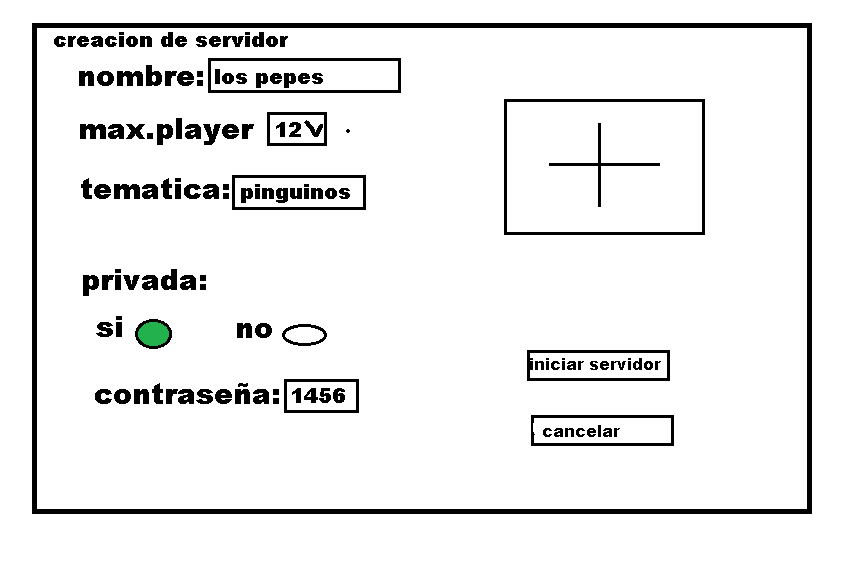
1.Grupos

2.Nombre del grupo

4.Intengrantes

5.Tema del servidor

6.Chat



(Creación de servidor)

1.nombre del server

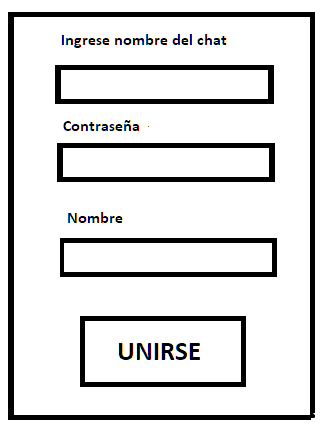
2.max usuarios

3.tematica

4.tipo de server

5.imgen

6.o unirse



Para unirse a un servidor

1.ingresar nombre del servidor

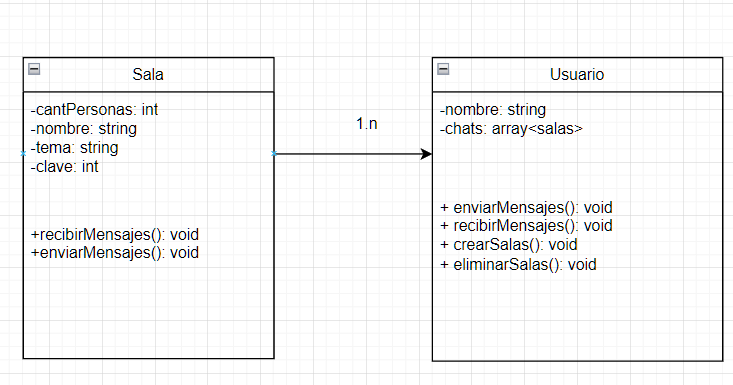
2.contraseña

3.nombre que usará en el servidor

**Funciones principales:**

* Crear salas de chat: El usuario podrá crear una sala de chat con una temática que él/ella elija.
* Conectarse a sala de chat: El usuario podrá ingresar a la sala de chat que él/ella elija utilizando nombre y contraseña de la sala.
* Cerrar salas de chat: El usuario tendrá la posibilidad de cerrar el chat en él/ ella estuvo.
* Enviar mensajes: El usuario podrá mandar mensajes a otro usuario.
* Recibir mensajes: El usuario tendrá la capacidad de recibir los mensajes enviados por otro usuario.
* Unirse a la salas de chat: Los usuarios podrán unirse a diferentes salas mediante el nombre del chat, contraseña y un nombre.

**Diagrama de clases:**

****

**Codigo (Entrega N°0):**

//servidor

using System;

using System.Net;

using System.Net.Sockets;

using System.Text;

using System.Threading;

class Program

{

static TcpListener server;

static TcpClient client1;

static TcpClient client2;

static void Main()

{

server = new TcpListener(IPAddress.Any, 12345);

server.Start();

Console.WriteLine("Servidor escuchando en el puerto 12345...");

client1 = server.AcceptTcpClient();

Console.WriteLine("Cliente 1 conectado");

client2 = server.AcceptTcpClient();

Console.WriteLine("Cliente 2 conectado");

Thread thread1 = new Thread(ClienteHandler);

Thread thread2 = new Thread(ClienteHandler);

thread1.Start(client1);

thread2.Start(client2);

}

static void ClienteHandler(object clientObj)

{

TcpClient client = (TcpClient)clientObj;

NetworkStream stream = client.GetStream();

byte[] buffer = new byte[1024];

int bytesRead;

while (true)

{

try

{

bytesRead = stream.Read(buffer, 0, buffer.Length);

if (bytesRead <= 0)

{

break;

}

string mensaje = Encoding.ASCII.GetString(buffer, 0, bytesRead);

Console.WriteLine($"Mensaje recibido: {mensaje}");

if (client == client1)

{

EnviarMensaje(client2, mensaje);

}

else

{

EnviarMensaje(client1, mensaje);

}

}

catch (Exception ex)

{

Console.WriteLine($"Error: {ex.Message}");

break;

}

}

client.Close();

}

static void EnviarMensaje(TcpClient client, string mensaje)

{

NetworkStream stream = client.GetStream();

byte[] buffer = Encoding.ASCII.GetBytes(mensaje);

stream.Write(buffer, 0, buffer.Length);

stream.Flush();

}

}

/cliente

using System;

using System.Net.Sockets;

using System.Text;

class Program

{

static void Main()

{

Console.WriteLine("Cliente 1 o 2?");

string clienteNum = Console.ReadLine();

TcpClient client = new TcpClient();

try

{

client.Connect("127.0.0.1", 12345);

Console.WriteLine("Conectado al servidor...");

NetworkStream stream = client.GetStream();

while (true)

{

Console.Write("Escribe un mensaje: ");

string mensaje = Console.ReadLine();

byte[] buffer = Encoding.ASCII.GetBytes(mensaje);

stream.Write(buffer, 0, buffer.Length);

stream.Flush();

if (mensaje.ToLower() == "salir")

{

break;

}

buffer = new byte[1024];

int bytesRead = stream.Read(buffer, 0, buffer.Length);

string respuesta = Encoding.ASCII.GetString(buffer, 0, bytesRead);

Console.WriteLine($"Respuesta del servidor: {respuesta}");

}

}

catch (Exception ex)

{

Console.WriteLine($"Error: {ex.Message}");

}

finally

{

// Cerrar la conexión y liberar recursos

client.Close();

}

}

}