**Sala De Chat Virtual**

**Cores Juan, Villalobos Carolina, Torrico Lucas y Lumbrera Joaquin**

**Grupo: A LO BUKELE**

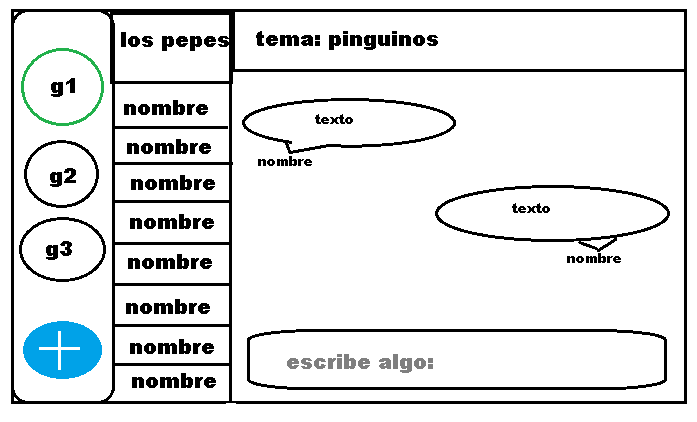
**Hogar Naval Stella Maris E.T. 37 Distrito 11**

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):

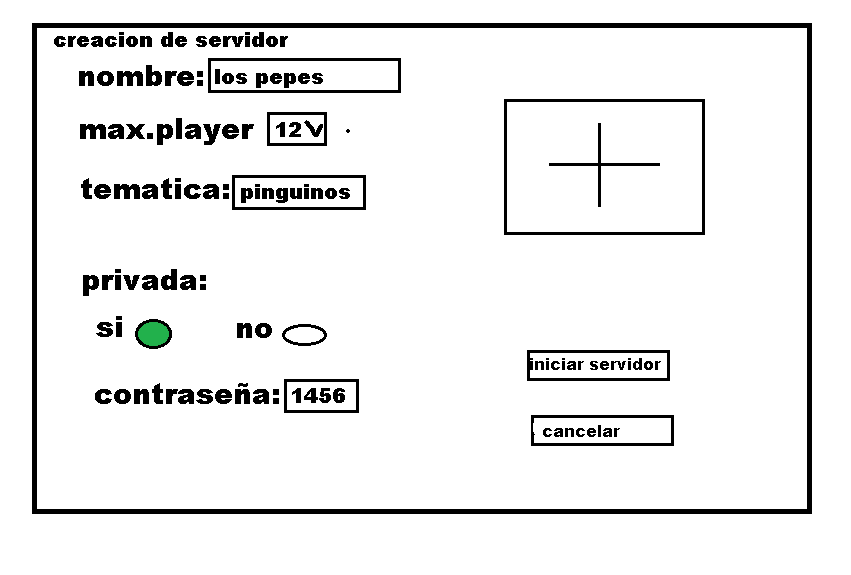
1.grupos

2. Nombre del grupo

4.integrantes

5.tema del server

6.Chat



(Creación de servidor)

1.nombre del server

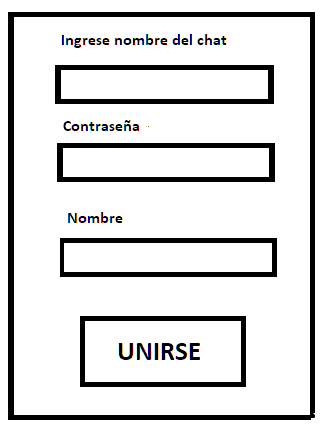
2.max usuarios

3.tematica

4.tipo de server

5.imgen

6.o unirse



Para unirse a un servidor

1.ingresar nombre del servidor

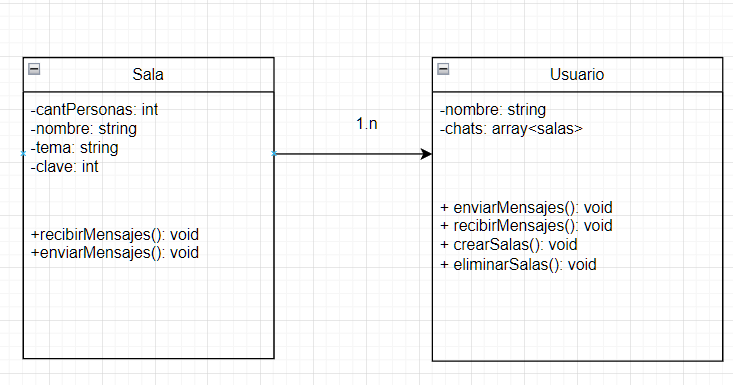
2.contraseña

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:**

****

**Código (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();

}

}

}

**Introducción:**

En el año 2023 con el avance de las tecnologías, nos encontramos en un mundo virtualizado, en consecuencia a esto, hay mejoras y cambios en muchos aspectos de la vida, como puede ser la comunicación. En la actualidad, el desarrollo y evolución de la misma es muy importante, debido a que permite la conexión entre personas a largas distancias, algunos ejemplos de estos son las redes sociales como Instagram, X, snapchat, entre otros. En este contexto el trabajo a desarrollar (realizado como trabajo de desarrollo de la materia “programación sobre redes”, sexto año de secundario) por alumnos del “Hogar Naval Stella Maris”, es una nueva red social que permite crear distintos medios de comunicaciones entre múltiples usuarios para su interacción. Para poder realizar con éxito este trabajo, en la primera sección se define el concepto de las salas de chat virtuales y su funcionamiento. En la segunda sección se detallan las herramientas y métodos que se utilizaran para el correcto desarrollo del mismo y sus referencias

**1.Concepto y funcionamiento Sala de Chat Virtual:**

Una sala de chat virtual es un espacio en línea que permite la comunicación en tiempo real entre usuarios a través de Internet. Para entenderlo en términos técnicos, poder desglosar los principales componentes y funcionamiento de esta herramienta:

1.1 Servidor de Chat: En el núcleo de una sala de chat virtual se encuentra un servidor dedicado que actúa como intermediario central. Este servidor es responsable de recibir, almacenar y distribuir los mensajes entre los usuarios conectados. Funciona como un punto de control central que gestiona todas las interacciones.

1.2 Protocolo de Comunicación: Para que los usuarios puedan enviar y recibir mensajes de manera coherente, se utiliza un protocolo de comunicación. Un ejemplo es el protocolo (TCP), que garantiza la entrega confiable de mensajes.

1.3 Usuario de Chat: Los usuarios crean la sala de chat o acceden a ella. Esta app proporciona una interfaz gráfica que permite a los usuarios ingresar a la sala, enviar mensajes y recibir respuestas.

1.4 Salas de chat: Una sala de chat puede tener múltiples usuarios. Cada sala tiene su propio nombre y tema, lo que facilita la organización de las conversaciones.

1.5. Identificación y Autenticación: Los usuarios acceden con un nombre de usuario a la sala de chat. Esta identificación permite que los usuarios sean reconocidos y que sus mensajes se asocien con sus nombres de usuario.

1.6 Mensajes en Tiempo Real: La característica principal de una sala de chat virtual es la comunicación en tiempo real. Los mensajes enviados por un usuario se entregan de inmediato a todos los demás usuarios presentes en la sala, lo que facilita la conversación fluida y la interacción entre estos.

**2. Elementos de trabajo:**

**Cascada:**

Fases secuenciales (requisitos, diseño, implementación, pruebas, despliegue), entregables definidos para cada fase, documentación detallada, planificación inicial exhaustiva.

Utilizaran para programar C# Es un lenguaje de programación desarrollado por Microsoft que forma parte de la plataforma .NET. Es un lenguaje orientado a objetos y seguro, diseñado para el desarrollo de aplicaciones de Windows, aplicaciones web y otros tipos de software. C# Se utiliza para una amplia variedad de aplicaciones, incluyendo el desarrollo de aplicaciones de escritorio de Windows (utilizando Windows Forms o WPF).

**Referencias:**

“Resolviendo problemas en los sistemas de información, Enfoque Para Informáticos” Dra. Maria Florencia Pollo Cattaneo, cuarta edición, año: 2018.