**Creación de un Chat**

**Temas de la sección:**

1. Comunicación entre usuarios
2. Comunicación entre 1 a muchos
3. Comunicación 1 a 1
4. Comunicación entre salas de chat
5. Diseño del chat
6. Diseño del login
7. Notificaciones de entrada de usuarios y salida de usuarios

El objetivo es crear una aplicación de chat completa.

Creamos la clase server/classes/usuarios.js para controlar los usuarios.

En los sockets, los clientes del front que se conectan tiene un id y es ese id que vamos a utilizar para controlarlos.

Creamos un método registrarPersona donde solo hacemos personas.push( persona) para mantener una lista de ellos.

Creamos el método getPersona(id){} aquí usaremos:

**Let persona = this.personas.filter(per=>{ return id=per.id});**

**/**/el filter extrae cada elemento al callback y este se encarga de retornar la coicidencia, pero retorna un arregle por lo que en este caso, para obtener solo la primera coincidecía (que no deberían haber mas), hacemos:

**Let persona = this.personas.filter(per=>{ return id=per.id})[0];**

En el front para conectar un usuario lo haremos creando public/chat.html, importamos la librería.

En el archivo chat.html:

<script src="socket.io/socket.io.js"></script> //Para sockets

  <script src="js/socket-chat.js"></script>

**Logica mandar mensaje para todos**

La lógica para esto es manda desde el cliente con socket.emit el mensaje en el servidor que se resive con client.on y la misma palabra clave ‘crearMensaje’, dentro del .on el servidor realizará un client.broadcast.emit(‘crearMensaje’, client.id, mensaje:’algo’) podemos usar la utileria creada server/utilerías/utileria.js crearMEnsaje(id, mensaje) para crear el objeto que se mandará.

**Implementación mensaje 1 a 1**

Para la implementación mensaje uno a uno, en el mensaje se deberá adjuntar el id del cliente por lo que solamente se hará:

Socket.broadcast.to(idcliente).emit(‘mensajaPrivado’,{mensaje:’mensaje’}) y con esto se mandará el mensaje para aquellos que ids en el **to los clientes siempre están a la escucha de mensaje privado por lo que solo lo procesarán.**

**Sala de chat**

Para esto usaremos la misma idea, el **socket.broadcast.to() crearemos un forma que mande en el action: “chat.html”**

Validamos que lleguen los datos nombre y sala. Los sockets automáticamente te ponen en un grupo o “sala” que es el de tu id, pero podemos especificar la pertenencia a un grupo y con esto enviar mensaje al grupo, usamos:

**client.join(nombre del grupo) 🡪 client.join(‘data.sala’); En el servidor**

y en la clase persona guardaremos un propiedad que conserve la sala a la que la persona pertenece. (yo había pensado en usar un arreglo para las salas pero con esta cualidad de los socket se mandará mensajes exclusivamente a los miembros de la sala y al guardar la sala en persona podemos saber a que sala pertenece).

Para que solo se mande a las persona de un grupo cambiaremos  client.broadcast.emit('listaPersona', usuarios.getPersonas());

Que notifica a todos por:

 client.broadcast.to(usuario.sala)emit('listaPersona', usuarios.getPersonas());

Este usuario.sala es el grupo al que se unió el usuario cuando hizo **client.join(usuario.sala);**

Agregar el .to debe ser para todos los emit de manera que le lleguen solo a los que están en la misma sala.