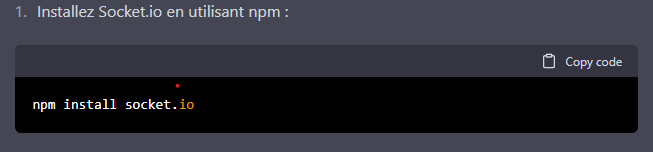
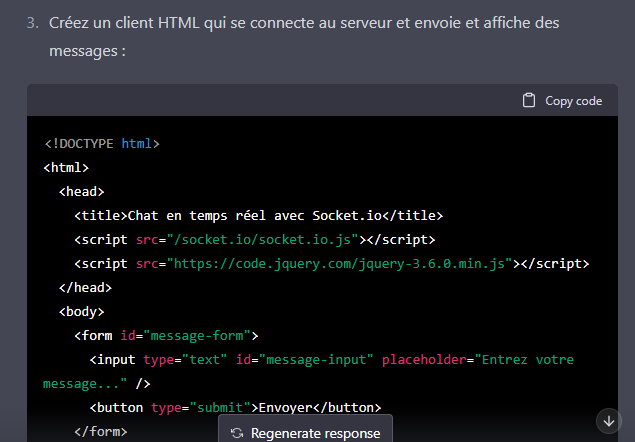
**Socket IO**

**Socket.io** permet de mettre en place une communication en temps en temps réel le client et le serveur grâce à l’utilisation des websockets.









const express = require('express');

const app = express();

const server = require('http').createServer(app);

const io = require('socket.io')(server);

io.on('connection', (socket) => {

console.log('Un nouveau client est connecté');

socket.on('send message', (message) => {

io.emit('new message', message);

});

});

server.listen(3000, () => {

console.log('Le serveur est démarré sur le port 3000');

});

////////////////////////////////////////////////////////

<!DOCTYPE html>

<html>

<head>

<title>Chat en temps réel avec Socket.io</title>

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

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

</head>

<body>

<form id="message-form">

<input type="text" id="message-input" placeholder="Entrez votre message..." />

<button type="submit">Envoyer</button>

</form>

<ul id="messages"></ul>

<script>

const socket = io();

$('form').submit(function() {

socket.emit('send message', $('#message-input').val());

$('#message-input').val('');

return false;

});

socket.on('new message', function(message) {

$('#messages').append($('<li>').text(message));

});

</script>

</body>

</html>

/////////////////////////////////////////////////////////////////////////////////////////////////////

Voici comment fonctionne le chat en temps réel créé avec Socket.io et JavaScript que je vous ai présenté :

1. Lorsque l'utilisateur ouvre la page HTML dans son navigateur, il est invité à entrer son nom d'utilisateur.
2. Le client envoie un message au serveur en utilisant la méthode **emit** de Socket.io pour indiquer qu'un nouvel utilisateur s'est connecté et en fournissant son nom d'utilisateur.
3. Le serveur reçoit le message et ajoute le nom d'utilisateur à la liste des utilisateurs connectés. Il envoie également un message à tous les autres clients connectés pour mettre à jour la liste des utilisateurs.
4. Lorsque l'utilisateur entre un message dans le formulaire et clique sur le bouton Envoyer, le client envoie un message au serveur en utilisant la méthode **emit** de Socket.io pour indiquer qu'un nouveau message a été envoyé et en fournissant le contenu du message.
5. Le serveur reçoit le message et l'envoie à tous les autres clients connectés.
6. Lorsque les autres clients reçoivent le message, ils l'ajoutent à la liste des messages affichée sur la page.
7. Si un utilisateur se déconnecte, le serveur envoie un message aux autres clients pour mettre à jour la liste des utilisateurs connectés.

C'est ainsi que fonctionne ce chat en temps réel en utilisant Socket.io et JavaScript. Vous pouvez personnaliser et étendre cette implémentation selon vos besoins en utilisant les différentes options et fonctionnalités disponibles dans Socket.io.

Top of Form

Voici comment la page pourrait être organisée et affichée :

* En haut de la page, il y aurait un formulaire avec un champ de saisie pour entrer le message et un bouton Envoyer.
* En dessous du formulaire, il y aurait une liste des messages affichés sous forme de liste non ordonnée.
* Sur le côté de la page, il y aurait un titre indiquant "Utilisateurs connectés" et une liste des utilisateurs connectés sous forme de liste non ordonnée.

Cela étant dit, la mise en page et l'apparence exactes de la page dépendent de la manière dont vous avez conçu votre code HTML et de votre utilisation de CSS pour styliser la page. Vous êtes libre de personnaliser la page selon vos préférences et vos besoins.