Step 1: show html, open 2 windows in chrome using go live extension of vscode

Step 2: npm init -y

Step 3: npm i socket.io

Step 4: npm i nodemon

Step 5: nodemon (npm run devStart)

Step5-2: package.json: “main”: “server.js”

Step 6: server.js: Use socket.io to create server:

const io = require('socket.io')(3000,{

    cors:{

        origin: "\*"

    }

});

io.on('connection', socket => {

    socket.emit('chat-message', 'Hello World')

})

//everytime user load the website, it’s going to call this function

// chat-message is event name,

// hello world is the message

//we set it on server 3000 and server is running on port 5500 for client is because client and server is separate. Whatever port we put them in it will just communicate easy bc of socket.io

Step6.3: index.html

This is a path we have to use from socket IO library.

Io function in script.js is used from this library

 <script defer src= "http://localhost:3000/socket.io/socket.io.js"></script>

 <script defer src= "script.js"></script>

Step 7: script.js: This is where all of the client side of javascript go

This is where server hosting socket application

const socket = io('http://localhost:3000')

socket.on('chat-message', data => {

    console.log(data)

})

Step 7-2:

Line 4: console.log('new User')

Check the server. New user in terminal vscode.

Hello world in console log. This is because when connected, server send the message hello world through socket to client

2 users because we have 2 windows open

Step 8: <script src= "script.js"></script>

Step 9: script.js

**Line 2:** const messageForm = document.getElementById('send-container')

**Line 8:** messageForm.addEventListener('submit', e => {

    e.preventDefault()

})

**Prevent submit event so the page not loading everytime user send message**

**If the web still loading again check index.html if you have defer src script tag.**

[**https://blog.webdevsimplified.com/2019-12/javascript-loading-attributes-explained/**](https://blog.webdevsimplified.com/2019-12/javascript-loading-attributes-explained/)

Step 10: script.js

Line 3: const messageInput = document.getElementById('message-input')

Line 11:   const message = messageInput.value

     socket.emit('send-chat-message', message)

messageInput.value = ‘’

it clears the message in the input after we sent the message

Step 11: Server.js

Remove console.log(‘new User’)

Line 10:  socket.on('send-chat-message', message => {

         console.log(message)

    })

Send hello from browser => terminal vscode : hello

This is send from client to server. Send from client to another client:

Step 12: Remove console.log(message)

Send message from browser to another browser. can find it in console.

Step 13: script.js

Line 17: function appendMessage(message){

    const messageElement = document.createdElement('div')

    messageElement.innerText = message

}

Line 2: const messageContainer = document.getElementById('message-container')

Line 7:  appendMessage(data)

Remove line 8 in server.js

Step 14: script.js

Line 6: const name = prompt('What is your name?')

appendMessage('You joined')

Popup an asking window

Step 15: script.js

Line 8: socket.emit('new-user', name)

Send user name to server

Step 16: server.js

Line 9:  socket.on('new-user', name => {

 users[socket.id] = name

        socket.broadcast.emit('user-connected', name)

    })

Line 3: const users = {}

Step 17: script.js

Line 14: socket.on('user-connected', name => {

    appendMessage(name )

})

Change line 15 appendMessage(name )

})

to

    appendMessage(`${name} connected`)

Step 18: server.js:

Line 14:         socket.broadcast.emit('chat-message', message)

message to {message: message, name: users[socket.id]}

Step 19: script.js:

Line 11:     appendMessage(data)

To

    appendMessage(`${data.name}: ${data.message}`)

Step 20: script.js

Line 21: add

appendMessage(`You: ${message}`)

Step 21: server.js

Line 16 socket.on('disconnect', () => {

        socket.broadcast.emit('user-disconnected', users[socket.id])

        delete users[socket.id]

    })

Step 22: script.js