1. Create a main directory which contains index.html
2. Create 2 subdirectories. (javascript, nodeServer)
3. In your terminal run the command cd nodeServer
4. Now check if you are in your nodeServer directory. If yes then run the command npm init -y
5. Now in the terminal of nodeServer directory run the commands npm install express socket.io
6. Now go back to the root directory using the command ‘cd ..’ (it does not include inverted commas. It includes double fullstops which will bring you to the root directory which contains index.html file) .
7. NOTE: In your VSCode extensions make sure you have installed Live Server

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>ChatWave</title>
  <script src="http://localhost:8000/socket.io/socket.io.js"></script>
  <!-- <link rel="stylesheet" href="/css/style.css"> -->
</head>
<body>
  <nav>

    <h1>Welcome to ChatWave</h1>
    <p>A Real-Time Chat app using NodeJS and Sockets.io</p>
  </nav>
  <div class="container">
    <!--This div is used to contain the messages -->

  </div>
  <div class="send">
    <form action="#" id="send-container">
      <input type="text" name="messageInp" id="messageInp" placeholder="Type your message
here">
      <button class="btn" type="submit">Send</button>
    </form>
  </div>
  <script src="/javascript/client.js"></script>
</body>
</html>
```

Now let us set up your client and server.

1. Navigate to your nodeServer directory using the command cd nodeServer
2. Since we have installed express and socket.io we need not install it again
3. Make a new file named index.js and paste the following code

```
//Node server which will handle Socket.io connections

const io= require('socket.io')(8000, {
    cors: {
        origin: "http://127.0.0.1:5500",
        methods: ["GET", "POST"]
    }
  })
//console.log("inside index.js")
const users = {};

io.on('connection', socket =>{
    socket.on('new-user-joined', personName =>{
        //console.log("new user", personName);
        users[socket.id] = personName;
        socket.broadcast.emit('user-joined', personName);
    });

    socket.on('send', message =>{
        socket.broadcast.emit('receive', {message: message, name: users[socket.id]})
    });

    socket.on('disconnect', message =>{ // this will emit the event of disconnect to the server
        socket.broadcast.emit('left', users[socket.id]);
        delete users[socket.id];
    });
})
```

Now let us set up the client.

1. Run the command ‘ cd .. ‘ to navigate back to root directory.
2. Change the working directory to javascript directory by running the command cd javascript
3. Create a file client.js
4. Paste the following code in client.js

```
const socket = io("http://localhost:8000"); // this is the socket connection between the client and the server
const form = document.getElementById('send-container'); // this is the form which will be used to send the message

const append = (message, position) =>{ // this function will append the message to the container
   const messageElement = document.createElement('div'); // this will create a div element
   messageElement.innerText = message; // this will set the inner text of the div element to the message
   messageElement.classList.add('message'); // this will add the class message to the div element
   messageElement.classList.add(position); // this will add the class position to the div element
   messageContainer.append(messageElement); // this will append the messageElement to the messageContainer
   if(position == 'left'){
      messageSound.play(); // this will play the audio
   }
}

form.addEventListener('submit', (e) =>{ // this will add an event listener to the form
   e.preventDefault(); // this will prevent the default behaviour of the form
   const message = messageInput.value; // this will get the value of the messageInput
   append(`You: ${message}`, 'right'); // this will append the message to the container
   socket.emit('send', message); // this will emit the event of send to the server
   messageInput.value = ''; // this will set the value of the messageInput to empty string
})

const messageInput=document.getElementById('messageInp'); // this is the input box where the user will type the message
const messageContainer = document.querySelector(".container"); // this is the container where all the messages will be displayed

const personName = prompt("Enter your name to join"); // this is the prompt which will ask the user to enter his/her name
socket.emit('new-user-joined', personName); // this will emit the event of new user joined to the server

socket.on('user-joined', personName =>{
   append(`${personName} joined the chat`, 'right');// this will append the message to the container
   joinSound.play(); // this will play the audio
})
```

```
socket.on('receive', data =>{
   append(`${data.name}: ${data.message}`, 'left'); // this will append the message to the container
})
socket.on('left', personName =>{ // this will emit the event of leave to the server
   append(`${personName} left the chat`, 'left');
})
```

Instructions to run the app

1. Open your index.html file. Run the file by right clicking on the html codespace and selecting the option “Open with Live Server”.
2. Your html document should be up and running at port 127.0.0.1:5500
3. Now navigate to your nodeServer directory by running the command “cd nodeServer” in your terminal.
4. Now in the nodeServer directory run the command “node index.js”
5. Your node server will be up and running.
6. Open the index.html web page that you had run with live server. Now reload the page.
7. Enter the user name.
8. Now open other tab in your browser and type 127.0.0.1:5500
9. It should ask for another user name.
10. Now both the sockets are connected. You can chat with each other.
11. NOTE: This only works on your local machine not your friends machine. You cannot host it on your local machine and type the link on your friends machine.