|  |
| --- |
| Code File For ChatRoom and download Chat history |
| Project is Simple chat room where people can join by adding there name and the moment user will join or leave other people present in the chat room will get notification. He also can download chat histry, chat history will open in new window. |
|  |

Server Side

Server.js

const mongoose = require("mongoose");

const Msg = require('./models/messages');

const express = require("express");

let fs = require("fs");

const path = require("path");

const app = express();

app.use(express.json());

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

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

const mongoDB = 'mongodb+srv://ajoymahato16:PJ6yH8PzKMR3igsI@cluster0.wmldu9m.mongodb.net/chat?retryWrites=true&w=majority';

mongoose.connect(mongoDB).then(()=>{

}).catch(err =>console.log(err));

var users = {};

app.use(express.static(path.join(\_\_dirname+"/public")));

io.on("connection", function(socket){

    socket.on("newuser",function(username){

        users[socket.id] = username;

        socket.broadcast.emit("user-connected",username);

    });

    socket.on("exituser",function(){

        socket.broadcast.emit("user-left", user = users[socket.id]);

        delete users[socket.id];

    });

    socket.on("chat",function(message){

        const chatmessage = new Msg({msg:message});

        chatmessage.save().then(()=>{

            socket.broadcast.emit("chat",message);

        })

    });

});

app.get("/api/:user",(request,response)=> {

    let user = request.params.user

    Msg.find({"username":`${user}`})

    .then(result=>{

        response.status(200).json({

            chatMessage:result

        })

    })

})

server.listen(5000);

Model

Message.js is schema for database.

const mongoose = require("mongoose");

const msgSchema = new mongoose.Schema({

    msg:{

        username:String,

        text:String

    }

})

const Msg = mongoose.model('msg',msgSchema);

module.exports = Msg;

Client Side

Code.js

(function(){

    const app = document.querySelector(".app");

    const socket = io();

    let username;

    app.querySelector(".join-screen #join-user").addEventListener("click", function(){

        username = app.querySelector(".join-screen #username").value;

        if(username.length == 0)

        {

            return

        }

        socket.emit("newuser", username);

        app.querySelector(".download-chat .download").setAttribute("href",`api/${username}`);

        app.querySelector(".join-screen").classList.remove("active");

        app.querySelector(".chat-screen").classList.add("active");

    });

    app.querySelector(".chat-screen #send-message").addEventListener("click", function(){

        let message = app.querySelector(".chat-screen #message-input").value;

        if(message.length == 0)

        {

            return;

        }

        renderMessage("my",{

            username:username,

            text:message

        })

        socket.emit("chat",{

            username : username,

            text:message

        })

        app.querySelector(".chat-screen #message-input").value = "";

    });

    app.querySelector(".chat-screen #exit-chat").addEventListener("click",function(username){

        socket.emit("exituser", username);

        window.location.href = window.location.href;

    });

    socket.on("user-connected",function(socket\_name){

        userjoinleft(socket\_name,'joined')

    })

    socket.on("user-left",function(user){

       // console.log(user);

        userjoinleft(user,'left')

    })

    function userjoinleft(name, status)

    {

        let uselogin=`${name} has ${status}`

        let dvuser = app.querySelector(".chat-screen .user-join");

        dvuser.append(uselogin);

    }

    socket.on("udpate",function(update){

        renderMessage("update", update);

    });

    socket.on("chat",function(message){

        renderMessage("other", message);

    });

    function renderMessage(type,message){

        let messageContainer = app.querySelector(".chat-screen .messages");

        if(type == "my"){

            let el = document.createElement("div");

            el.setAttribute("class", "message my-message");

            el.innerHTML= `

                <div>

                    <div class="name">you</div>

                    <div class="text">${message.text}</div>

                </div>

            `;

            messageContainer.appendChild(el);

        }else if(type == "other"){

            let el = document.createElement("div");

            el.setAttribute("class", "message other-message");

            el.innerHTML= `

                <div>

                    <div class="name">${message.username}</div>

                    <div class="text">${message.text}</div>

                </div>

            `;

            messageContainer.appendChild(el);

        } else if (type == "update")

        {

            let el = document.createElement("div");

            el.setAttribute("class", "update");

            el.innerText =  message;

            messageContainer.appendChild(el);

            console.log("update");

        }

        messageContainer.scrollTop = messageContainer.scrollHeight = messageContainer.clientHeight;

    }

})();

Index.html

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

    <link rel="stylesheet" type="text/css" href="style.css" />

    <link href="https://fonts.googleapis.com/css2?family=Roboto:ital,wght@0,100;0,300;0,400;0,500;1,100;1,300;1,400;1,500&display=swap" rel="stylesheet">

</head>

<body>

   <div class="app">

    <div class="screen join-screen active">

        <div class="form-input">

            <h2> Join Chatroom</h2>

            <label>Username</label>

            <input type="text" id="username">

        </div>

        <div class="form-input">

            <button id="join-user">Join</button>

        </div>

    </div>

    <div class="screen chat-screen ">

        <div class="header">

            <div class="logo">

            </div>

            <button id="exit-chat">Exit</button>

        </div>

        <div class="messages">

        </div>

        <div class="user-join">

        </div>

        <div class="download-chat">

            <a target="\_blank" class="download" href="javascript:void(0)">Downlaod chat History</a>

        </div>

        <div class="typebox">

            <input type="text" id="message-input">

            <button id="send-message">Send</button>

        </div>

    </div>

   </div>

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

    <script src="code.js"></script>

</body>

</html>