Лабораторна робота №6

Project03 “Chat Application”

Лабораторна робота 06. Протокол WebSocket. Використання Socket.io для розробки чат-додатків

style.css

.form-container {

    perspective: 1000px; /\* встановлено перспективу \*/

}

.form-container:hover {

    transform: translateZ(5px); /\* при наведенні здійснюється зсув вглиб \*/

}

.form-container button {

    width: 100%;

}

/\* General Styles \*/

\* {

    margin: 0;

    padding: 0;

    box-sizing: border-box;

}

html {

    font-size: 16px;

}

input {

    font-size: 14px;

}

body {

    line-height: 1.4;

    color: #750101;

    font-family: Helvetica, Arial, sans-serif;

}

h1 {

    margin-bottom: 16px;

}

label {

    display: block;

    font-size: 14px;

    margin-bottom: 8px;

    color: #9d2929;

}

input {

    border: 1px solid #000000;

    padding: 12px;

    outline: none;

}

button {

    cursor: pointer;

    padding: 12px;

    background: #000000; /\* чорний колір \*/

    border: none;

    color: #FF0000; /\* червоний колір \*/

    font-size: 16px;

    transition: transform .3s ease; /\* додано анімацію паралаксу \*/

    perspective: 1000px; /\* встановлено перспективу \*/

}

button:hover {

    transform: translateZ(5px); /\* при наведенні здійснюється зсув вглиб \*/

}

button:disabled {

    cursor: default;

    background: #7c5cbf94;

}

/\* Join Page Styles \*/

.centered-form {

    background: #a3aed0;

    width: 100vw;

    height: 100vh;

    display: flex;

    justify-content: center;

    align-items: center;

}

.centered-form\_\_box {

    box-shadow: 0px 0px 17px 1px #abb2cd;

    background: #000000;

    padding: 24px;

    width: 250px;

}

.centered-form button {

    width: 100%;

}

.centered-form input {

    margin-bottom: 16px;

    width: 100%;

}

/\* Chat Page Layout \*/

.chat {

    display: flex;

}

.chat\_\_sidebar {

    height: 100vh;

    color: rgb(0, 0, 0);

    background: #d19a9a;

    width: 225px;

    overflow-y: scroll

}

/\* Chat styles \*/

.chat\_\_main {

    flex-grow: 1;

    display: flex;

    flex-direction: column;

    max-height: 100vh;

}

.chat\_\_messages {

    flex-grow: 1;

    padding: 24px 24px 0 24px;

    overflow-y: scroll;

}

/\* Message Styles \*/

.message {

    margin: 8px;

}

.message-name {

    font-weight: 600;

    font-size: 14px;

    margin-right: 8px;

}

.message-meta {

    font-size: 14px;

}

.message a {

    color: #cc0000;

}

/\* Message Composition Styles \*/

#compose input {

    border: 1px solid #131313;

    width: 100%;

    padding: 12px;

    margin: 0 16px 0 0;

    flex-grow: 1;

}

#compose button {

    font-size: 14px;

    width: 150px;

}

/\* Chat Sidebar Styles \*/

.room-title {

    font-weight: 400;

    font-size: 22px;

    background: #99a4cb;

    padding: 24px;

}

.list-title {

    font-weight: 500;

    font-size: 18px;

    margin-bottom: 4px;

    padding: 12px 24px 0 24px;

}

.users {

    list-style-type: none;

    font-weight: 300;

    padding: 12px 24px 0 24px;

}

chat.html

<!doctype html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css" integrity="sha384-Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh" crossorigin="anonymous">

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

    <title>Chat</title>

</head>

<body>

    <section>

        <div class="container-fluid">

            <div class="row">

                <div id="sidebar" class="col-4 vh-100 bg-dark p-4"> <!-- Оновлено bg-info на bg-dark -->

                </div>

                <div id="chat" class="col-8 d-flex flex-column flex-grow-1 vh-100 p-0">

                    <div id="messages" class="flex-grow-1 overflow-auto">

                    </div>

                    <div id="compose">

                        <form id="message-form" class="form-inline">

                            <div class="form-group flex-grow-1">

                                <input type="text" name="message" placeholder="Message" required autocomplete="off" class="form-control w-100">

                            </div>

                            <div class="form-group ml-2">

                                <button class="btn btn-danger btn-lg">Send</button> <!-- Оновлено btn-warning на btn-danger -->

                            </div>

                        </form>

                    </div>

                </div>

            </div>

        </div>

    </section>

<script id="message-template" type="text/html">

    <div class="alert message {{bgClass}}">

        <div>

            <span class="message-name">{{username}}</span>

            <span class="message-meta">{{createdAt}}</span>

        </div>

        <div>{{message}}</div>

    </div>

</script>

<script id="sidebar-template" type="text/html">

    <div class="alert alert-primary">

        Room: <strong>{{room}}</strong><br>

        Username: <strong>{{currentUser}}</strong>

    </div>

    <ul class="list-group">

        <li class="list-group-item list-group-item-primary">Users in the chatroom</li>

        {{#users}}

        <li class="list-group-item">{{username}}</li>

        {{/users}}

    </ul>

</script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/mustache.js/3.1.0/mustache.min.js"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/moment.js/2.24.0/moment-with-locales.min.js"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/qs/6.9.1/qs.min.js"></script>

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

<script>

    //elements

    const $messageForm = document.querySelector("#message-form")

    const $messageFormInput = $messageForm.querySelector("input")

    const $messageFormButton = $messageForm.querySelector("button")

    const $messages = document.querySelector("#messages")

    //Templates

    const messageTemplate = document.querySelector("#message-template").innerHTML

    const sidebarTemplate = document.querySelector('#sidebar-template').innerHTML

    const {username, room} = Qs.parse(location.search, {ignoreQueryPrefix: true})

    var socket = io();

    socket.emit('join', {username, room}, (error) => {

        if(error){

            alert(error)

            location.href = '/'

        }

    })

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

        const html = Mustache.render(messageTemplate, {

            username: message.username,

            message: message.text,

            createdAt: moment(message.createdAt).format('h:mm a')

        })

        $messages.insertAdjacentHTML('beforeend', html)

    })

    socket.on('roomData', ({ room, users }) => {

        const html = Mustache.render(sidebarTemplate, {

            room,

            currentUser: username,

            users

        })

        document.querySelector('#sidebar').innerHTML = html

    })

    $messageForm.addEventListener('submit', (e) => {

        e.preventDefault()

        $messageFormButton.setAttribute('disabled', 'disabled')

        const message = $messageFormInput.value

        socket.emit('sendMessage', message, (error) => {

            $messageFormButton.removeAttribute('disabled')

            $messageFormInput.value = ''

            $messageFormInput.focus()

            if (error) {

                return console.log(error)

            }

            console.log('Message delivered!')

        })

    })

</script>

</body>

</html>

index.html

<!doctype html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css" integrity="sha384-Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh" crossorigin="anonymous">

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

    <title>Приєднатись до чату</title>

</head>

<body>

<div class="vw-100 vh-100 d-flex justify-content-center align-items-center">

    <div class="form-container bg-dark p-4">

        <form action="/chat.html">

            <div class="form-group">

                <input type="text" class="form-control" id="nickname" name="username" placeholder="Логін">

            </div>

            <div class="form-group">

                <input type="text" class="form-control" id="room" name="room" placeholder="Кімната">

            </div>

            <button type="submit" class="btn btn-danger">Приєднатись</button>

        </form>

    </div>

</div>

</body>

</html>

messages.js

const generateMessage = (username, text) => {

    return {

        username,

        text,

        createdAt: new Date().getTime()

    }

}

const generateLocationMessage = (username, url) => {

    return {

        username,

        url,

        createdAt: new Date().getTime()

    }

}

module.exports = {

    generateMessage,

    generateLocationMessage

}

users.js

const users = []

const addUser = ({ id, username, room }) => {

    // Clean the data

    username = username.trim().toLowerCase()

    room = room.trim().toLowerCase()

    // Validate the data

    if (!username || !room) {

        return {

            error: 'Username and room are required!'

        }

    }

    // Check for existing user

    const existingUser = users.find((user) => user.room === room && user.username === username)

    // Validate username

    if (existingUser) {

        return {

            error: 'Username is in use!'

        }

    }

    // Store user

    const user = { id, username, room }

    users.push(user)

    return { user }

}

const removeUser = (id) => {

    const index = users.findIndex((user) => user.id === id)

    if (index !== -1) {

        return users.splice(index, 1)[0]

    }

}

const getUser = (id) => {

    return users.find((user) => user.id === id)

}

const getUsersInRoom = (room) => {

    room = room.trim().toLowerCase()

    return users.filter((user) => user.room === room)

}

module.exports = {

    addUser,

    removeUser,

    getUser,

    getUsersInRoom

}

app.js

const express = require('express')

const http = require('http')

const socketio = require('socket.io')

const {generateMessage, generateLocationMessage} = require("./utils/messages")

const {addUser, removeUser, getUser, getUsersInRoom} = require('./utils/users')

const app = express()

const server = http.createServer(app)

const io = socketio(server)

app.use(express.static(\_\_dirname + '/../public'));

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

    socket.on('join', (options, callback) => {

        const { error, user } = addUser({ id: socket.id, ...options })

        if (error) {

            return callback(error)

        }

        socket.join(user.room)

        socket.emit('message', generateMessage('Admin', `Welcome, ${user.username}!`))

        socket.broadcast.to(user.room).emit('message', generateMessage('Admin', `${user.username} has joined!`))

        io.to(user.room).emit('roomData', {

            room: user.room,

            users: getUsersInRoom(user.room)

        })

        callback()

    })

    socket.on('sendMessage', (message, callback) => {

        const user = getUser(socket.id)

        if (user) {

            io.to(user.room).emit('message', generateMessage(user.username, message))

        }

        callback()

    })

    socket.on('disconnect', () => {

        const user = removeUser(socket.id)

        if (user) {

            io.to(user.room).emit('message', generateMessage('Admin', `${user.username} has left!`))

            io.to(user.room).emit('roomData', {

                room: user.room,

                users: getUsersInRoom(user.room)

            })

        }

    })

});

server.listen(3000, function () {

    console.log('listening on \*:3000');

});

package.json

{

  "name": "chatapp",

  "version": "1.0.0",

  "description": "",

  "main": "app.js",

  "scripts": {

    "test": "echo \"Error: no test specified\" && exit 1",

    "dev": "nodemon -e html,js src/app.js",

    "start": "node src/app.js"

  },

  "keywords": [],

  "author": "",

  "license": "ISC",

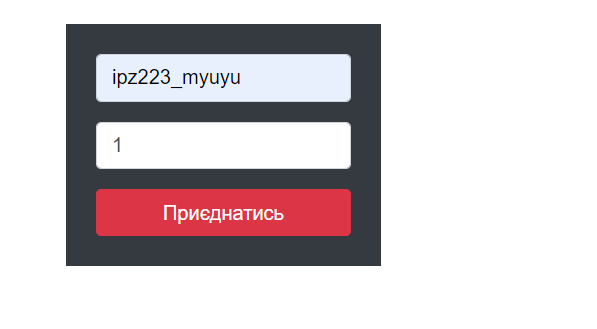
  "dependencies": {

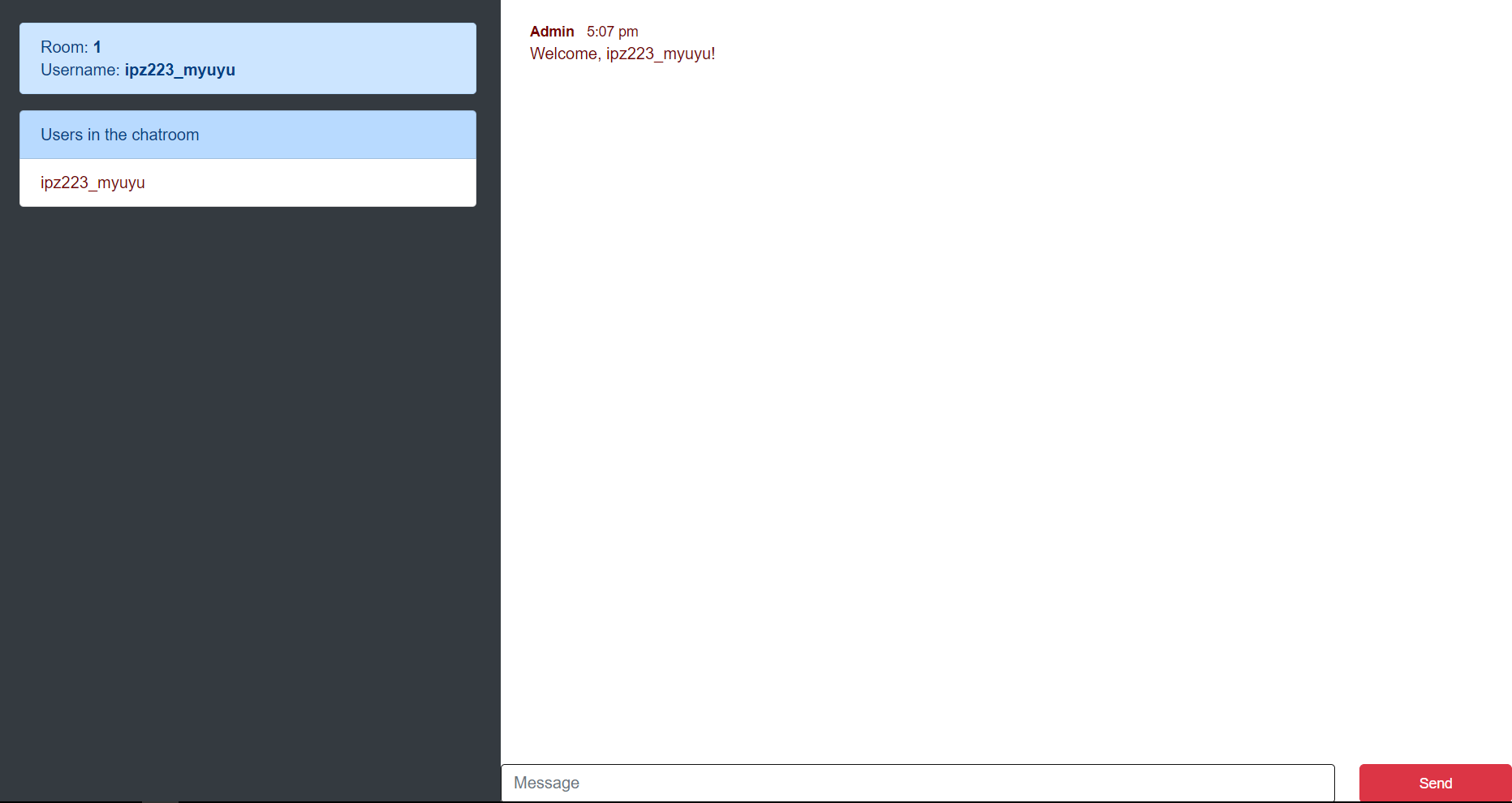
    "express": "^4.19.2",

    "socket.io": "^4.7.5"

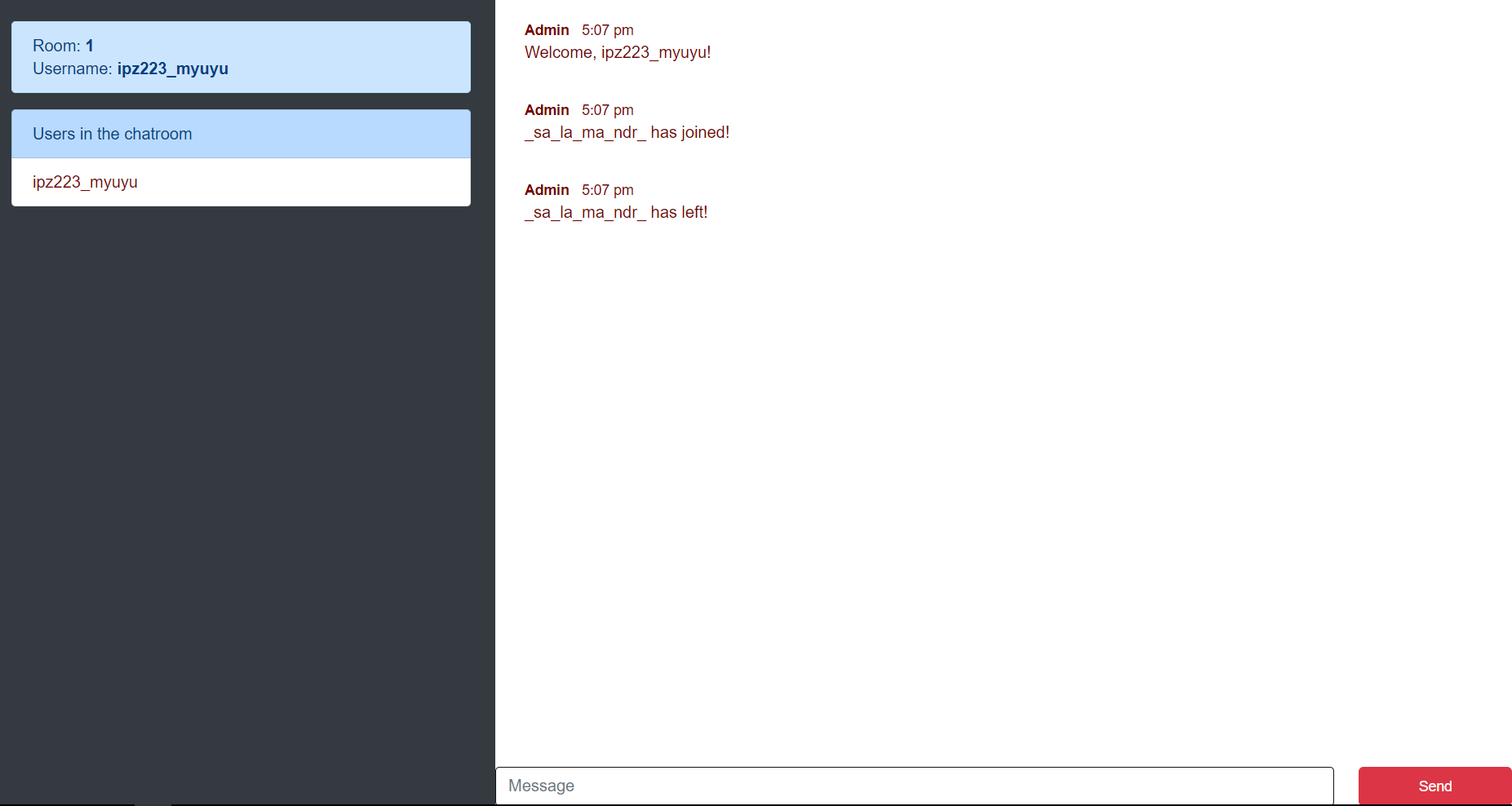
  }

}



Вхід в систему 

Чат



Повідомлення в чаті