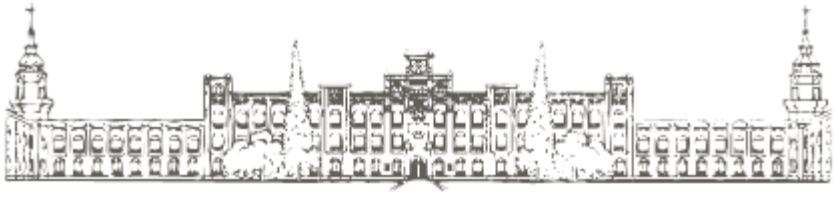


Міністерство освіти і науки України

Національний технічний університет України „КПІ

імені Ігоря Сікорського ”

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

ЗВІТ

лабораторної работи №4

курсу «Основи WEB - технологій »

Тема: «**Протокол WebSocket. Використання Socket.io для розробки чат-додатків**»

Перевірив:

Доц. Голубєв Л. П.

Виконав:

Гр. ІП-01

Черпак Андрій

Київ 2023

1. Завдання.

Розробити додаток для обміну повідомленнями між учасниками в режимі реального часу за допомогою бібліотеки SocketIO.

2. Хід роботи

Завдання 1.

За допомогою текстового редактора JB Rider, було створено файл з кодом сервера з використанням Express.js (index.js), структуру WEBсторінки чату мовою HTML-5 (index.html), каскадну таблицю стилів (styles.css), а також JS-файл зі скриптом для веб-сторінки (script.js):

index.js:

```
const express = require('express');
const http = require('http');
const socketIO = require('socket.io');

const app = express();
const server = http.createServer(app);
const io = socketIO(server);

app.use('/static', express.static(__dirname +'/public'));

app.get('/', (req, res) => {
  res.sendFile(__dirname + '/index.html');
});

const users = {};
const chats = {};

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

  socket.on('new user', (username) => {
   if (username in users) {
    socket.emit('username taken', username);
   } else {
    socket.emit('username available', chats);
    socket.username = username;
    users[username] = socket.id;
   }
  });

  socket.on('create chat', (chatname) => {
   if (chatname in chats) {
    socket.emit('chatname taken', chatname);
   } else {
    socket.emit('chatname available', chatname);
    chats[chatname] = {messages: [], users: []};
    io.emit('new chat', chatname);
   }
  });

  socket.on('join chat', (chatname) => {
```

```
    if (socket.currentChat !== undefined){
     chats[socket.currentChat].messages.push({type: "disconn", user:
socket.username});
     let index = chats[socket.currentChat].users.indexOf(socket.username);
     if (index > -1){
      chats[socket.currentChat].users.splice(index, 1);
     }
     io.emit('user leaved', {username: socket.username, chat: socket.currentChat});
    }
    socket.currentChat = chatname;
    chats[socket.currentChat].users.push(socket.username);
    chats[socket.currentChat].messages.push({type: "conn", user: socket.username});
    io.emit('user joined', {username: socket.username, chat: socket.currentChat});
    socket.emit('history', chats[socket.currentChat].messages);
    socket.emit('users', chats[socket.currentChat].users);
  });

  socket.on('chat message', (message) => {
    io.emit('chat message', { username: socket.username, chat: socket.currentChat,
message: message });
    chats[socket.currentChat].messages.push({type: "mess", user: socket.username,
message: message});
  });

  socket.on('disconnect', () => {
    delete users[socket.username];

    if (socket.currentChat !== undefined){
     io.emit('user leaved', {username: socket.username, chat: socket.currentChat})
     chats[socket.currentChat].messages.push({type: "disconn", user:
socket.username});
     let index = chats[socket.currentChat].users.indexOf(socket.username);
     if (index > -1){
      chats[socket.currentChat].users.splice(index, 1);
     }
    }
  });
});

server.listen(3000, () => {
  console.log('Running on http://localhost:3000/');
});
```

index.html:

```
<!DOCTYPE html>
<html lang="ua">

<head>
  <title>Text Chat</title>
  <link rel="stylesheet" href="/static/styles.css">
  <script src="/socket.io/socket.io.js"></script>
</head>

<body>
<div id="join-form">
  <h3>Оберіть ім'я, щоб приєднатися!</h3>
  <input id="username" type="text" placeholder="Введіть ваше ім'я">
  <button id="log-in">Приєднатись</button>
</div>
<div id="main-area">
```

```
  <div id="chatlist-area">
    <div id="chat-list-container">
      <h3>Доступні чати:</h3>
      <ul id="chat-list"></ul>
    </div>
    <div id="create-chat">
      <h3>Створити новий чат:</h3>
      <input id="chatname-inp" type="text" placeholder="Введіть назву чату">
      <button id="create">Створити</button>
    </div>
  </div>
  <div id="messages-area">
    <div id="chat-title">
      <h3 id="title">
        Оберіть чат, будь ласка
      </h3>
    </div>
    <div id="messages-wrapper">
      <ul id="messages"></ul>
    </div>
    <div id="message-form">
      <input id="message" type="text" placeholder="Введіть Ваше повідомлення">
      <button id="send">Надіслати</button>
    </div>
  </div>
  <div id="participants-area">
    <h3>Учасники чату:</h3>
    <ul id="participants-list"></ul>
  </div>
</div>

<div id="popup" class="popup-hidden">
  <p id="popup-text"></p>
  <div id="popup-timer">
    <div id="timer-bar" class="timer-full">

    </div>
  </div>
</div>

<script
src="https://cdnjs.cloudflare.com/ajax/libs/socket.io/4.5.1/socket.io.min.js"></scrip
t>
<script src="/static/script.js"></script>
</body>

</html>
```

style.css:

```
:root{
  --page-main-bg: #202c31;
  --chat-area-bg: #28383e;
  --headers-color: #dce2e5;
  --message-color: #b9c5ca;
  --li-color: #a8b7bd;
  --buttons-bg: #30434a;
  --buttons-fg: #fff;
  --scrollbar-track-color: #555;
  --scrollbar-thumb-color: #888;
  --scrollbar-thumb-hover-color: #f1f1f1;
  --popup-timerbar-color: #4e6eff;
  --active-chatlink-color: #4e6eff;
```

```
  --inputs-bg-color: #617379;
  --inputs-fg-color: #dce0e1;
  --inputs-border-color: #cad0d2;
  --placeholder-fg-color: #a7b1b5;
}

html {
  font-family: 'Trebuchet MS', 'Lucida Sans Unicode', 'Lucida Grande', 'Lucida Sans',
Arial, sans-serif;
  background-color: var(--page-main-bg);
  margin: 0;
  padding: 0;
  box-sizing: border-box;
  height: 100%;
  width: 100%;
}

body {
  width: 100%;
  height: 100%;
  margin: 0;
  padding: 0;
  display: block;
  box-sizing: border-box;
}

#join-form {
  text-align: center;
  margin-top: 50px;
  display: block;
}

#join-form > h3 {
  margin-bottom: 30px;
}

#username {
  padding: 10px;
  font-size: 16px;
  background-color: var(--inputs-bg-color);
  border: 1px solid var(--inputs-border-color);
  color: var(--inputs-fg-color);
  border-radius: 5px;
  width: 200px;
  margin-right: 10px;
}

#log-in {
  padding: 10px 20px;
  font-size: 16px;
  background-color: var(--buttons-bg);
  color: var(--buttons-fg);
  border: none;
  border-radius: 5px;
  cursor: pointer;
}

#create-chat {
  flex: 2;
  text-align: center;
  display: flex;
  flex-direction: column;
```

```
  box-sizing: border-box;
}

#create-chat h3{
  flex: 2;
}

#chatname-inp {
  font-size: 16px;
  background-color: var(--inputs-bg-color);
  border: 1px solid var(--inputs-border-color);
  color: var(--inputs-fg-color);
  border-radius: 5px;
  flex: 2;
  text-align: center;
}

#create {
  flex: 2;
  font-size: 16px;
  background-color: var(--buttons-bg);
  color: var(--buttons-fg);
  border: none;
  border-radius: 5px;
  cursor: pointer;
}

/* Стилі для чату */
#main-area {
  height: 100%;
  width: 100%;
  display: none;
  flex-direction: row;
  text-align: left;
}

#main-area > *{
  padding: 1% 1%;
}

#chatlist-area{
  flex: 1;
  display: flex;
  flex-direction: column;
  height: 100%;
  box-sizing: border-box;
}

#chat-list-container {
  flex: 12;
  padding: 0;
  margin: 0;
  display: flex;
  flex-direction: column;
  height: 90%;
}

#chat-list{
  list-style-type: none;
  padding: 0;
  overflow-y: auto;
  height: 100%;
```

```
}

#chat-list::-webkit-scrollbar {
  width: 5px;
}

#chat-list::-webkit-scrollbar-track {
  background: var(--scrollbar-track-color);
}

#chat-list::-webkit-scrollbar-thumb {
  background: var(--scrollbar-thumb-color);
}

#chat-list::-webkit-scrollbar-thumb:hover {
  background: var(--scrollbar-thumb-hover-color);
}

#chat-list li {
  margin: 15px 0;
  color: var(--li-color);
  cursor: pointer;
}

#chat-list li:hover {
  color: var(--active-chatlink-color);
  margin-left: 1px;
}

#participants-area{
  flex: 1;
  display: block;
  height: 100%;
  box-sizing: border-box;
}

#participants-list{
  list-style-type: none;
  padding: 0;
  height: 95%;
  overflow-y: auto;
}

#participants-list li {
  margin-bottom: 10px;
  color: var(--li-color);
}

#participants-list::-webkit-scrollbar {
  width: 5px;
}

#participants-list::-webkit-scrollbar-track {
  background: var(--scrollbar-track-color);
}

#participants-list::-webkit-scrollbar-thumb {
  background: var(--scrollbar-thumb-color);
}

#participants-list::-webkit-scrollbar-thumb:hover {
  background: var(--scrollbar-thumb-hover-color);
```

```
}

#messages-area {
  height: 100%;
  box-sizing: border-box;
  display: block;
  text-align: center;
  flex: 3;
  background-color: var(--chat-area-bg);
}

#chat-title {
  display: flex;
  height: 5%;
  justify-content: center;
  align-items: center;
}

#messages-wrapper {
  height: 87%;
  display: flex;
  flex-direction: column-reverse;
  overflow: clip;
}

#messages {
  list-style-type: none;
  padding: 0;
  margin: 0;
  height: 100%;
  overflow-y: auto;
  display: flex;
  flex-direction: column-reverse;
}

#messages::-webkit-scrollbar {
  width: 5px;
}

/* Track */
#messages::-webkit-scrollbar-track {
  background: var(--scrollbar-track-color);
}

/* Handle */
#messages::-webkit-scrollbar-thumb {
  background: var(--scrollbar-thumb-color);
}

/* Handle on hover */
#messages::-webkit-scrollbar-thumb:hover {
  background: var(--scrollbar-thumb-hover-color);
}

#messages li {
  margin-bottom: 10px;
  text-align: left;
  color: var(--message-color);
}

#message-form {
  height: 8%;
```

```
  width: 100%;
  display: flex;
  flex-direction: row;
  align-items: center;
  visibility: hidden;
}

#message {
  padding: 10px;
  font-size: 16px;
  background-color: var(--inputs-bg-color);
  border: 1px solid var(--inputs-border-color);
  color: var(--inputs-fg-color);
  border-radius: 5px;
  width: 80%;
}

#send {
  padding: 10px 20px;
  font-size: 16px;
  background-color: var(--buttons-bg);
  color: var(--buttons-fg);
  border: none;
  border-radius: 5px;
  cursor: pointer;
  margin-left: 5px;
}

h3{
  text-align: center;
  margin: 0;
  color: var(--headers-color);
}

#popup {
  position: absolute;
  left: 40%;
  right: 40%;
  background: var(--buttons-bg);
  border-radius: 7.5% / 50%;
  display: flex;
  flex-direction: column;
  justify-content: center;
  align-items: center;
}

.popup-hidden{
  top: 0;
  bottom: 100%;
  opacity: 0;
  transition: top 1s ease, bottom 1s ease, opacity 0.5s linear;
}

.popup-show {
  top: 10%;
  bottom: 85%;
  opacity: 1;
}

#popup-timer{
  position: relative;
  height: 5%;
```

```
  width: 90%;
  flex: 1;
  display: block;
  text-align: left;
}

#timer-bar{
  position: relative;
  height: 5%;
  background-color: var(--popup-timerbar-color);
}

.timer-full {
  width: 100%;
}

.timer-running {
  width: 0;
  transition: width 3s 0.5s linear;
}

#popup-text {
  position: relative;
  width: 100%;
  color: var(--buttons-fg);
  text-align: center;
  margin: 0;
  flex: 5;
}

::placeholder {
  color: var(--placeholder-fg-color);
  opacity: 1;
}
```

script.js:

```
const socket = io();
const joinForm = document.getElementById('join-form');
const chatArea = document.getElementById('main-area');
const chatMessages = document.getElementById('messages');
const chatList = document.getElementById('chat-list');
const participants = document.getElementById('participants-list');
const currentChatTitle = document.getElementById('title');

let currentChatParticipants = []
let activePopUps = 0;

function addMessage(user, message){
  chatMessages.innerHTML = `<li><strong>${user}:</strong> ${message}</li>` +
chatMessages.innerHTML;
}

function addJoinMessage(user){
  chatMessages.innerHTML = `<li>${user} приєднався(лася) до розмови</li>` +
chatMessages.innerHTML;
}

function addLeaveMessage(user){
  chatMessages.innerHTML = `<li>${user} залишив(ла) чат</li>` +
chatMessages.innerHTML;
}
```

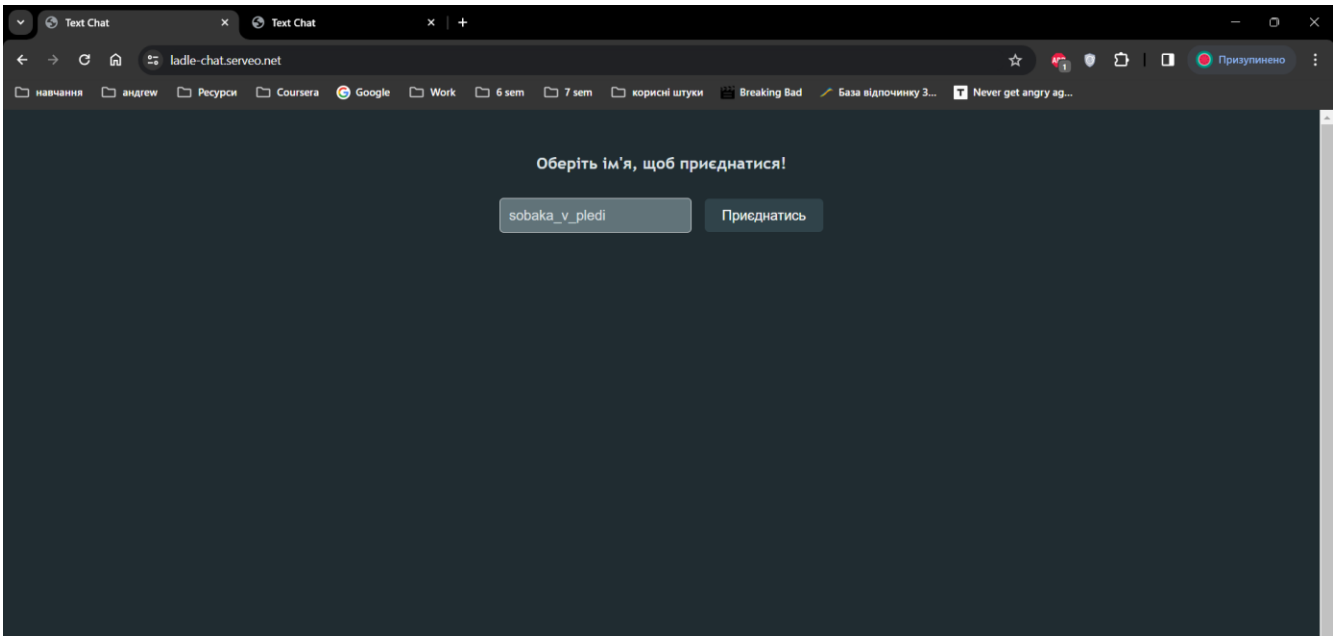
```
function joinChat(chatLink){
  let chatname = chatLink.innerText;
  socket.emit('join chat', chatname);
  currentChatTitle.innerText = chatname;
  socket.currentChat = chatname;
  document.getElementById('message-form').style.visibility = 'visible';
}

function addChatToList(chatname){
  chatList.innerHTML += `<li onclick="joinChat(this)">${chatname}</li>`;
}

function showPopup(text){
  activePopUps += 1;
  document.getElementById('popup-text').innerText = text;
  document.getElementById('popup').classList.add('popup-show');
  document.getElementById('timer-bar').classList.remove('timer-running');
  setTimeout(() => {
   document.getElementById('timer-bar').classList.add('timer-running');
  }, 10);
  setTimeout(() => {
   activePopUps -= 1;
   if (activePopUps === 0){
    document.getElementById('popup').classList.remove('popup-show');
    setTimeout(() => {
     if (activePopUps === 0)
      document.getElementById('timer-bar').classList.remove('timer-running');
    }, 1000);
   }
  }, 3500);
}

function updateParticipantsList(){
  let users = '';
  for (let i = 0; i < currentChatParticipants.length; i++) {
   let usr = currentChatParticipants[i];
   users += `<li>${usr}</li>`;
  }
  participants.innerHTML = users;
}

document.getElementById('log-in').addEventListener('click', () => {
  const username = document.getElementById('username').value;
  if (username) {
   socket.emit('new user', username);
   socket.username = username;
  }
});

document.getElementById('username').addEventListener("keypress", function(event) {
  if (event.key === "Enter") {
   event.preventDefault();
   document.getElementById("log-in").click();
  }
});

document.getElementById('create').addEventListener('click', () => {
  const chatname = document.getElementById('chatname-inp').value.toString();
  if (chatname) {
   socket.emit('create chat', chatname);
  }
});
```

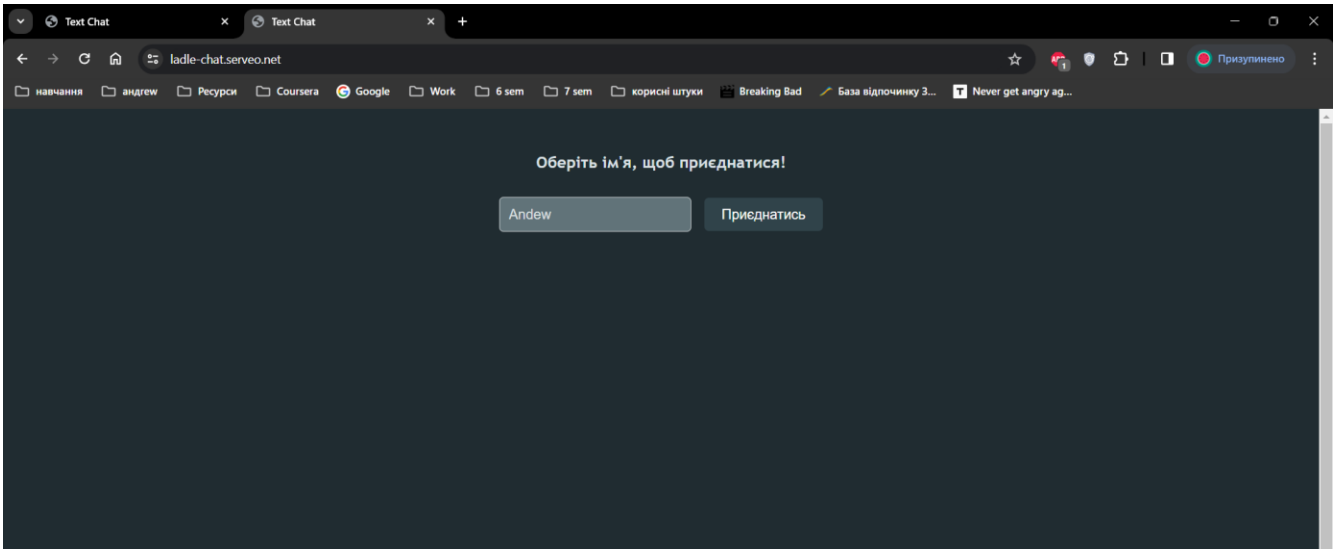
```
document.getElementById('chatname-inp').addEventListener("keypress", function(event)
{
  if (event.key === "Enter") {
   event.preventDefault();
   document.getElementById("create").click();
  }
});

socket.on('chatname taken', (chatname) => {
  showPopup(`Чат з назвою "${chatname}" вже існує!`);
});

socket.on('chatname available', (chatname) => {
  document.getElementById('chatname-inp').value = '';
  showPopup(`Новий чат "${chatname}" створено!`)
});

socket.on('username taken', (username) => {
  showPopup(`Користувач з ім\'ям "${username}" вже існує!`);
});

socket.on('username available', (chatsList) => {
  joinForm.style.display = 'none';
  chatArea.style.display = 'flex';
  chatList.innerHTML = '';
  for (let ch in chatsList){
   addChatToList(ch);
  }
});

socket.on('reload', () => {
  chatList.innerHTML = '';
  participants.innerHTML = '';
  chatMessages.innerHTML = '';
  currentChatParticipants = [];
  document.getElementById('message-form').style.visibility = 'hidden';
  currentChatTitle.innerText = "Оберіть чат, будь ласка";
  if (socket.username){
   socket.emit('new user', socket.username);
  }
});

socket.on('new chat', (chatname) => {
  addChatToList(chatname);
});

socket.on('user joined', (user_chat) => {
  if (user_chat.chat === socket.currentChat){
   addJoinMessage(user_chat.username);
   currentChatParticipants.push(user_chat.username);
   updateParticipantsList();
  }
});

socket.on('history', (messages) => {
  chatMessages.innerHTML = '';
  for (let i = 0; i < messages.length; i++) {
   let msg = messages[i];
   if (msg.type === "conn"){
    addJoinMessage(msg.user);
   }
```

```
    else if (msg.type === "discon"){
     addLeaveMessage(msg.user);
    }
    else if (msg.type === "mess"){
     addMessage(msg.user, msg.message);
    }
   }
});

socket.on('users', (users) => {
  currentChatParticipants = users;
  updateParticipantsList();
});

socket.on('user leaved', (user_chat) => {
  if (user_chat.chat === socket.currentChat) {
   addLeaveMessage(user_chat.username);
   let index = currentChatParticipants.indexOf(user_chat.username);
   if (index > -1){
    currentChatParticipants.splice(index, 1);
    updateParticipantsList();
   }
  }
});

socket.on('chat message', (data) => {
  if (data.chat === socket.currentChat) {
   addMessage(data.username, data.message);
  }
});

document.getElementById('send').addEventListener('click', () => {
  const message = document.getElementById('message').value.toString();
  if (message) {
   socket.emit('chat message', message);
   document.getElementById('message').value = '';
  }
});

document.getElementById('message').addEventListener("keypress", function(event) {
  if (event.key === "Enter") {
   event.preventDefault();
   document.getElementById("send").click();
  }
});
```

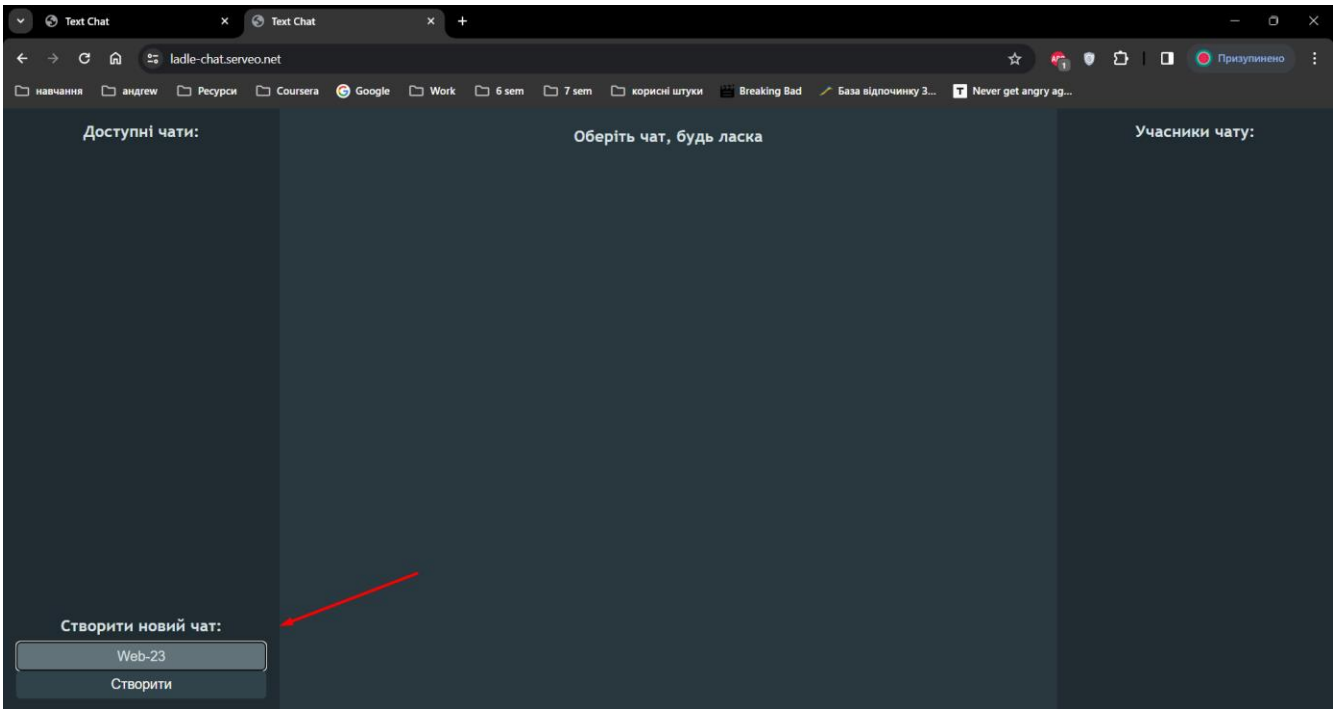
3. Отримані результати

Відкриємо сайт одразу з двох вкладок і приєднаємося з різними іменами:

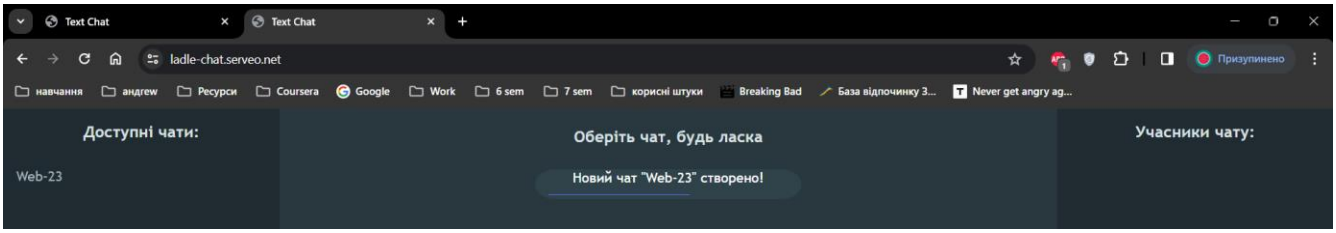




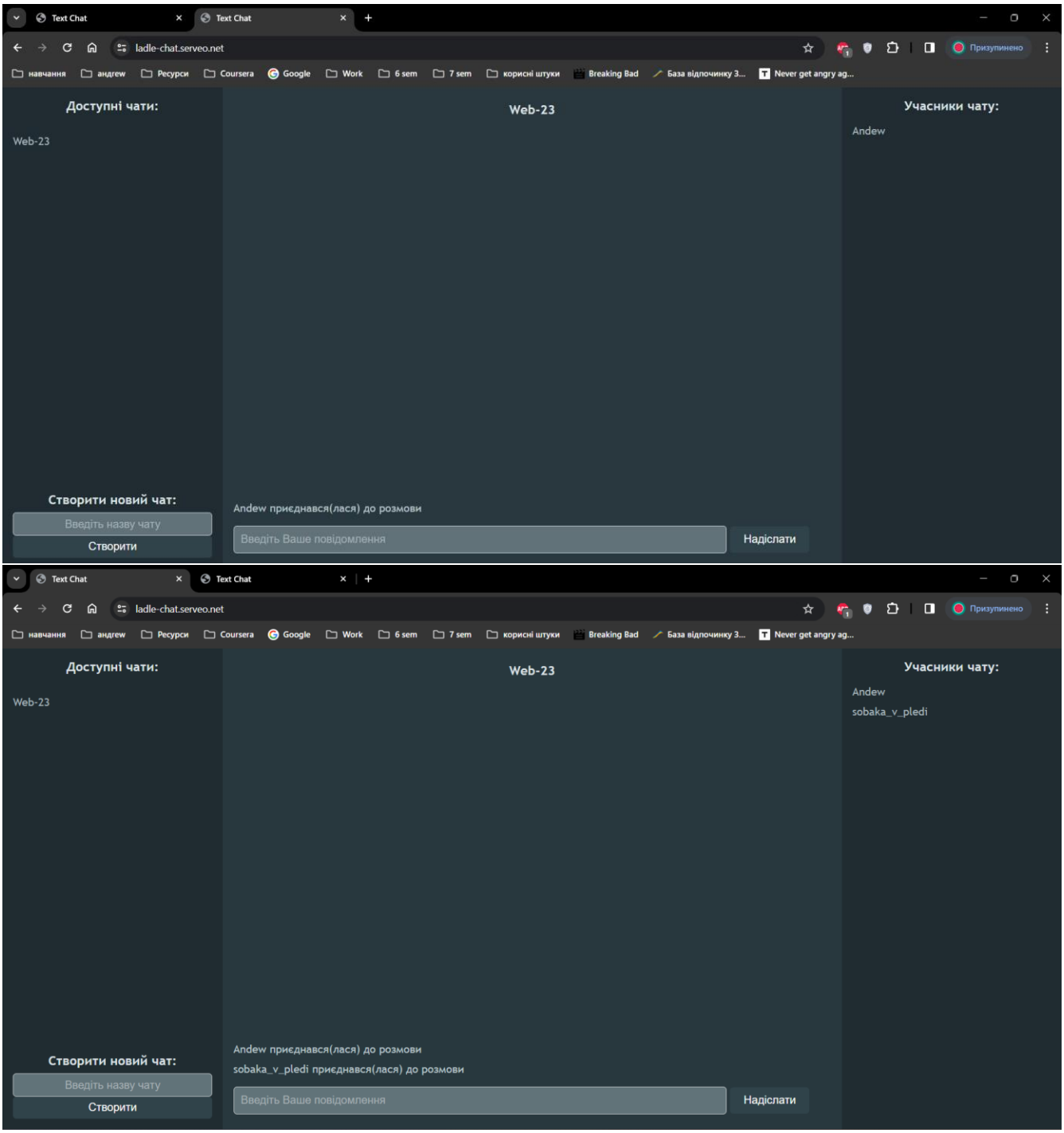
Оскільки список чатів наразі порожній, створимо новий чат:



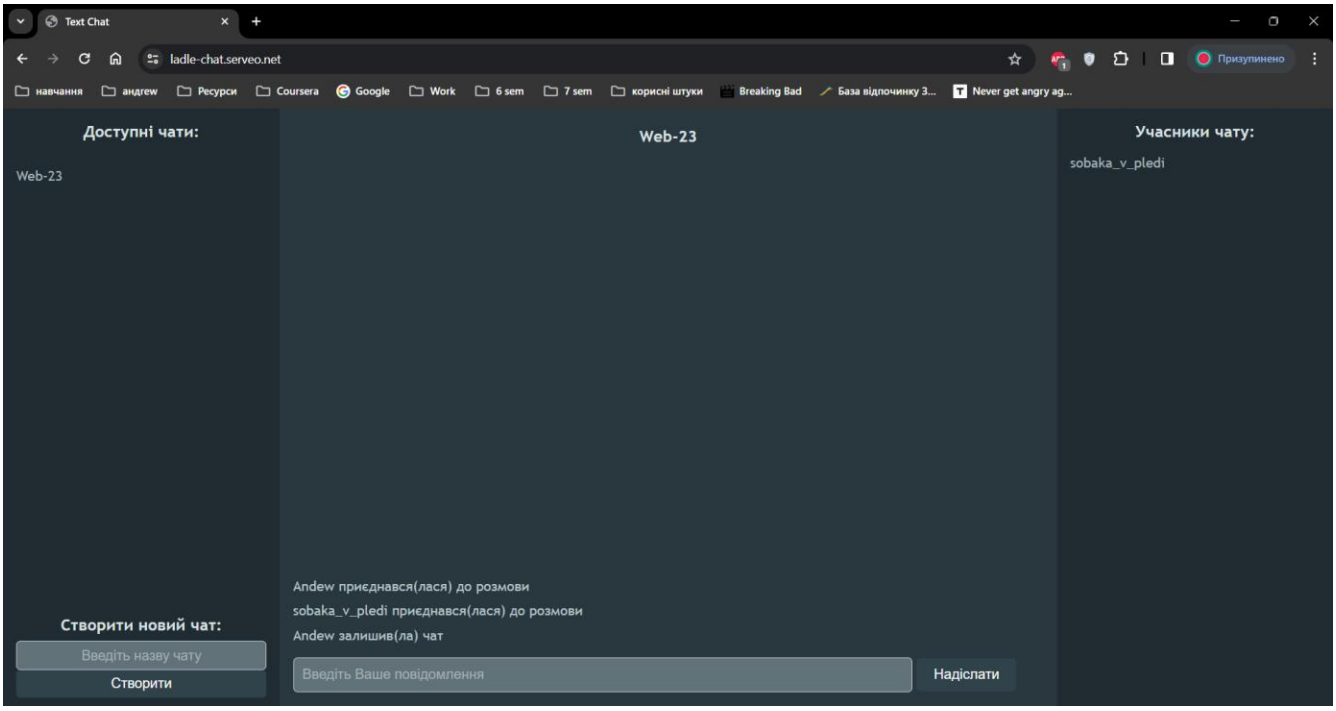
Новостворений чат одразу з’явився у списках для усіх учасників, а ініціатор також отримав сповіщення про успішне створення:



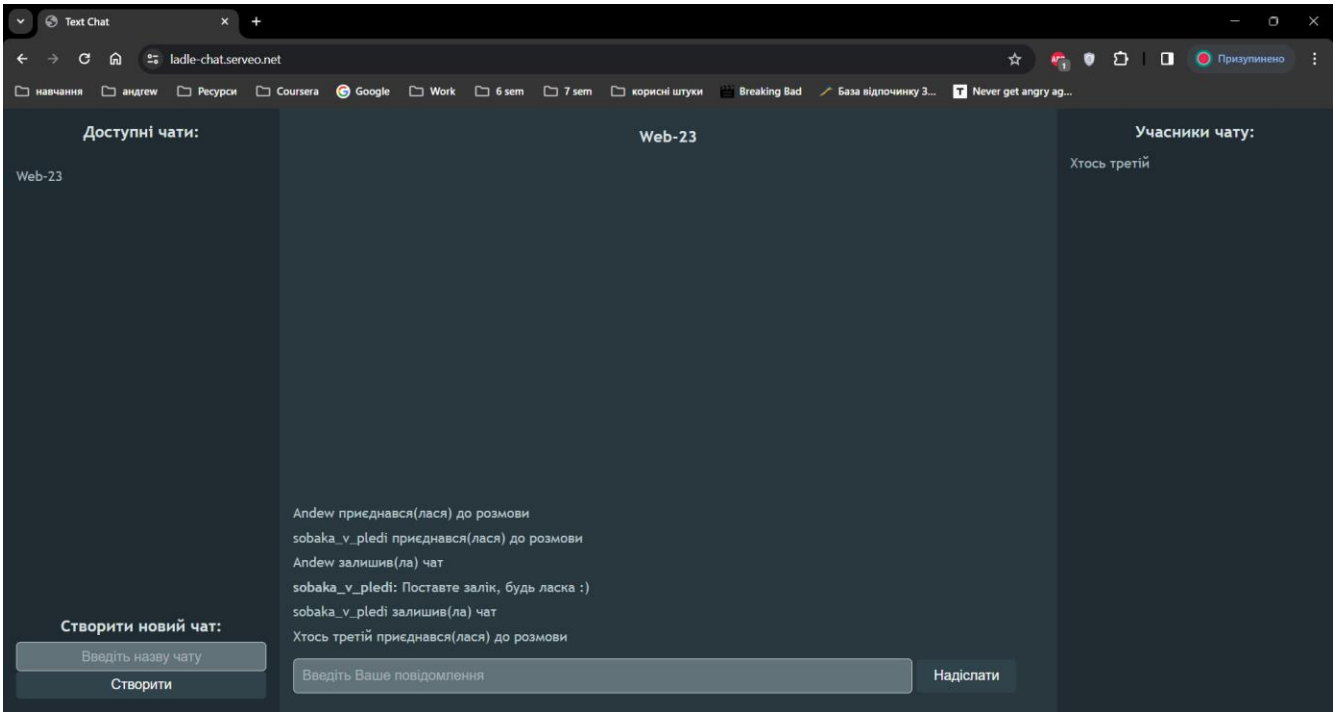
Приєднаємося до чату з обох акаунтів:



Як бачимо, при приєднанні до чату виводиться відповідне повідомлення, а також оновлюється список учасників. Те ж саме відбудеться і при від’єднанні:



Відтепер можемо обмінюватися повідомленнями, які відображатимуться як для тих, хто вже у чаті, так і для користувачів, що приєднаються до нього пізніше.



Посилання на репозиторій gitgub:

https://github.com/CherpakAndrii/web-basics/tree/master/Lab4

Посилання на чат: https://ladle-chat.serveo.net/

Висновок

Під час виконання даної лабораторної роботи я удосконалив свої навички верстки та роботи з елементами DOM у JavaScript, створив сервер з допомогою Express.js, ознайомився з принципами роботи сокетів та використав їх для реалізації функціоналу чату. Створений за стосунок було розміщено на хостинг з власного телефону з використанням serveo.net.