Internet Technology Lab Report – Assignment 3

TITLE

Name – Sourav Dutta
Roll – 001610501076
Class – BCSE 4th year
Group – A3

Assignment Numbers – 3

Write a multi-client chat application consisting of both client and server programs. In this chat application simultaneously several clients can communicate with each other. For this you need a single server program that clients connect to. The client programs send the chat text or image (input) to the server and then the server distributes that message (text or image) to all the other clients. Each client then displays the message sent to it by the server. The server should be able to handle several clients concurrently. It should work fine as clients come and go.

Develop the application using a framework based on Node.JS. How are messages handled concurrently?
Which web application framework(s) did you follow?
Prepare a detailed report of the experiments you have done, and your observations on performance of the system.

DESCRIPTION & EXPERIMENTS

- The server program is being written in JavaScript, and the Node.JS framework has been used to run the program.
- To handle concurrent messages, WebSockets have been used. The following is a brief description about WebSockets:
    - The WebSocket specification defines an API establishing "socket" connections between a web browser and a server. In plain words: There is an persistent connection between the client and the server and both parties can start sending data at any time.
    - Connection limitations are no longer a problem since WebSockets represent a single TCP socket connection.
    - Cross domain communication has been considered and is dealt with within the connection handshake.
    - Using WebSocket creates a whole new usage pattern for server side applications. While traditional server stacks such as LAMP are designed around the HTTP request/response cycle they often do not deal well with a large number of open WebSocket connections. Keeping a large number of connections open at the same time requires an architecture that receives high concurrency at a low performance cost. Such architectures are usually designed around either threading or so called non-blocking IO.
- The client program is being written in HTML, and the inner script in HTML uses JavaScript. The client can be run on any web browser.
- The following experiments have been performed or errors has been handled:
    - Unique username – A user cannot have same client id at the same time. If a new user tries to enter an existing client id of a running client, an alert is being shown which says “Duplicate Client Username”.
    - Empty input text while sending a text or image. Users need to atleast enter a character, otherwise it raises a alert showing that the input text is empty.
    - Multiple clients (multiple users) can join the chat group, wherein each user is being notified about “joining” and “leaving” of any user in the group.
    - Each client can send a text message or image. If the image has to be sent, the user should enter the complete path of the image present in the system.
    - Each client is being notified whenever an existing client sends a message (text or input), and the sender’s username is also shown before the message.
    - While pressing carriage return key on writing a text, the client adds a new line. Hereby, a user can enter many new line-separated sentences in the same message.

CODE SNIPPET

Server program (server.js): **[Language: Javascript, Framework used: Node.js]**

```
var server = require('ws').Server;
var s = new server({ port : 8080 });
var present = {};
s.on('connection', function(ws) {
	ws.on('message', function(message) {
		message = JSON.parse(message);

		if(message.type == "username") {
			ws.username = message.data;
			if(present[ws.username] == 1) {
				ws.username = "duplicateuser";
				present[ws.username] = 0;
				ws.send(JSON.stringify({
					type: "errormsg",
					data: "DUPLICATE-CLIENT"
				}));
				ws.close();
			}
			else {
				present[ws.username] = 1;
				s.clients.forEach(function(client){
					client.send(JSON.stringify({
						type: "username",
						data: ws.username
					}));
				});
				console.log(ws.username+" joined the chatroom");
			}
		}
		else if(message.type == "message") {
			console.log("Received from "+ws.username+": 
"+message.data);
			s.clients.forEach(function(client){
				if(client != ws) {
					client.send(JSON.stringify({
						type: "message",
						data: "<b>"+ws.username+":</b> 
"+message.data
					}));
				}

			});
		}
	});

	ws.on('close', function() {
		if(present[ws.username] == 1) {
			present[ws.username] = 0;
			console.log(ws.username+" left the chatroom");
			s.clients.forEach(function(client){
				client.send(JSON.stringify({
					type: "message",
                              data: ws.username+" left the chatroom"
                         }));
                    });
               }
          });
});
```

Client program (client.html):   **[Language: HTML, Javascript]**

```
<!DOCTYPE html>
<html lang='en'>
<head>
     <meta charset="UTF-8">
     <title>CHATROOM</title>
</head>
<body>
     <h2>Welcome to my Chatroom!</h2>
     <input type="text" onkeydown="enterKeyPressed(event)"
placeholder="Your message here" id="text">
     <button id="sendMessageButton">SEND TEXT MESSAGE</button>
     <button id="sendImageButton">SEND IMAGE</button>
     <div id="log"> </div>

     <script>
          var username = prompt('What is your name?');
          var sock = new WebSocket("ws://localhost:8080");

          var log = document.getElementById('log');

          function enterKeyPressed(evt) {
               evt = evt || window.event;
               if (evt.keyCode == 13) {
                    document.getElementById('text').value += '/';
               }
          }
          sock.onopen = function() {
               sock.send(JSON.stringify({
                    type: "username",
                    data: username
               }));
          }

          sock.onmessage = function() {
               var json = JSON.parse(event.data);
               if(json.type == "errormsg" && json.data == "DUPLICATE-
CLIENT") {
                    alert("ERROR: DUPLICATE CLIENT USERNAME! Please try
again with a different name");
                    return;
               }
               if(json.type == "username") {
                    if(json.data == username) {
                         log.innerHTML += "<h3>Welcome! Your username
is "+username+"</h3>";
                    }
                    log.innerHTML += json.data+" joined the
chatroom<br>";
```

```
                }
                else log.innerHTML += json.data+"<br>";
            }

            document.getElementById("sendMessageButton").onclick =
function() {
                var Text = document.getElementById('text').value;
                if(Text == "") {
                    alert("Please enter some text while sending text
message");
                    return;
                }
                var newText = "";
                for(var i=0;i<Text.length;i++) {
                    if(Text.charAt(i) == '/') {
                        newText += '<br>';
                    } else {
                        newText += Text.charAt(i);
                    }
                }
                log.innerHTML += "<b>You:</b> "+newText+"<br>";
                sock.send(JSON.stringify({
                    type: "message",
                    data: newText
                }));
            };

            document.getElementById("sendImageButton").onclick =
function() {
                var Text = "<img
src='"+document.getElementById('text').value+"'>";
                if(Text == "<img src=''>") {
                    alert("Please enter a image url while sending text
message");
                    return;
                }
                log.innerHTML += "<b>You:</b> "+Text+"<br>";
                sock.send(JSON.stringify({
                    type: "message",
                    data: Text
                }));
            };

      </script>
</body>
</html>
```

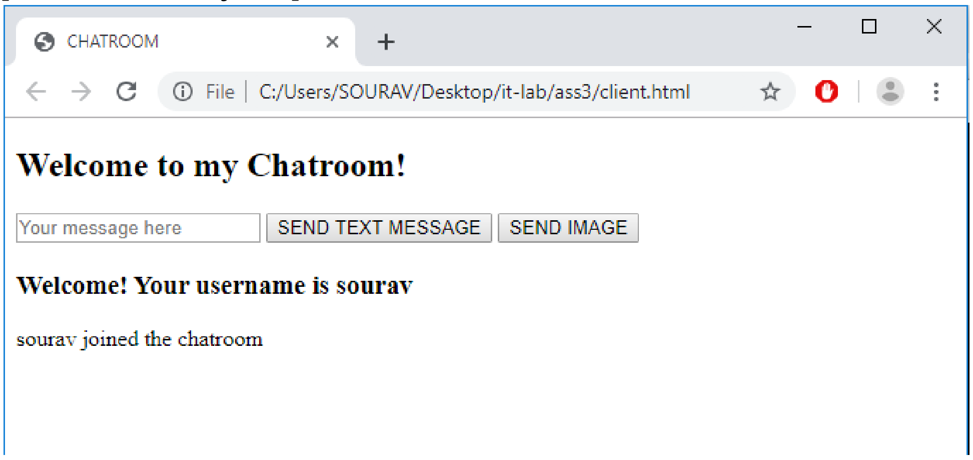
OUTPUT

Server program (server.js):

```
C:\Users\SOURAV\Desktop\it-lab\ass3>node server.js
sourav joined the chatroom
rohan joined the chatroom
Received from sourav: hey
Received from rohan: hello
rahul joined the chatroom
Received from rahul: <img src='julogo.png'>
rohan left the chatroom
rahul left the chatroom
sourav left the chatroom
```

Client programs (client.html):

[After 1st client joins]

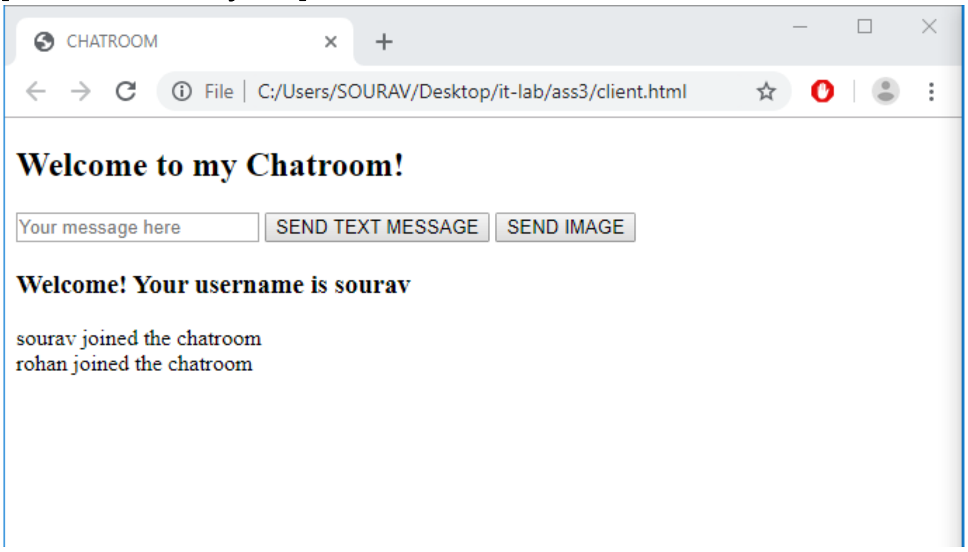


Welcome to my Chatroom!

Welcome! Your username is sourav

sourav joined the chatroom

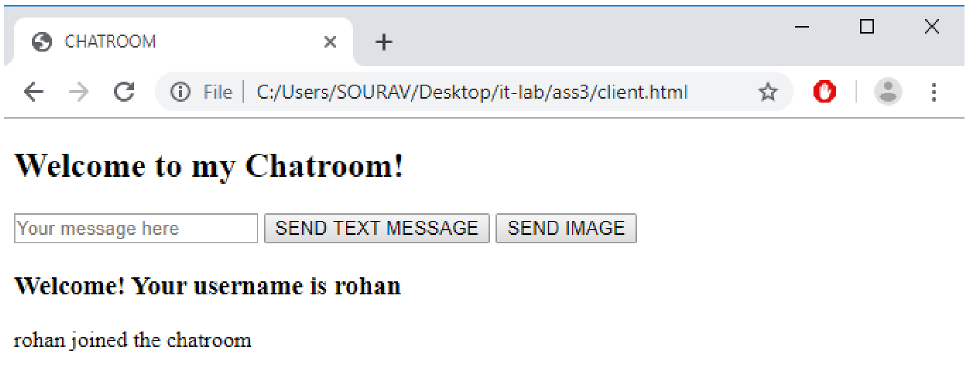
[After 2nd client joins]



Welcome to my Chatroom!

Welcome! Your username is sourav

sourav joined the chatroom
rohan joined the chatroom

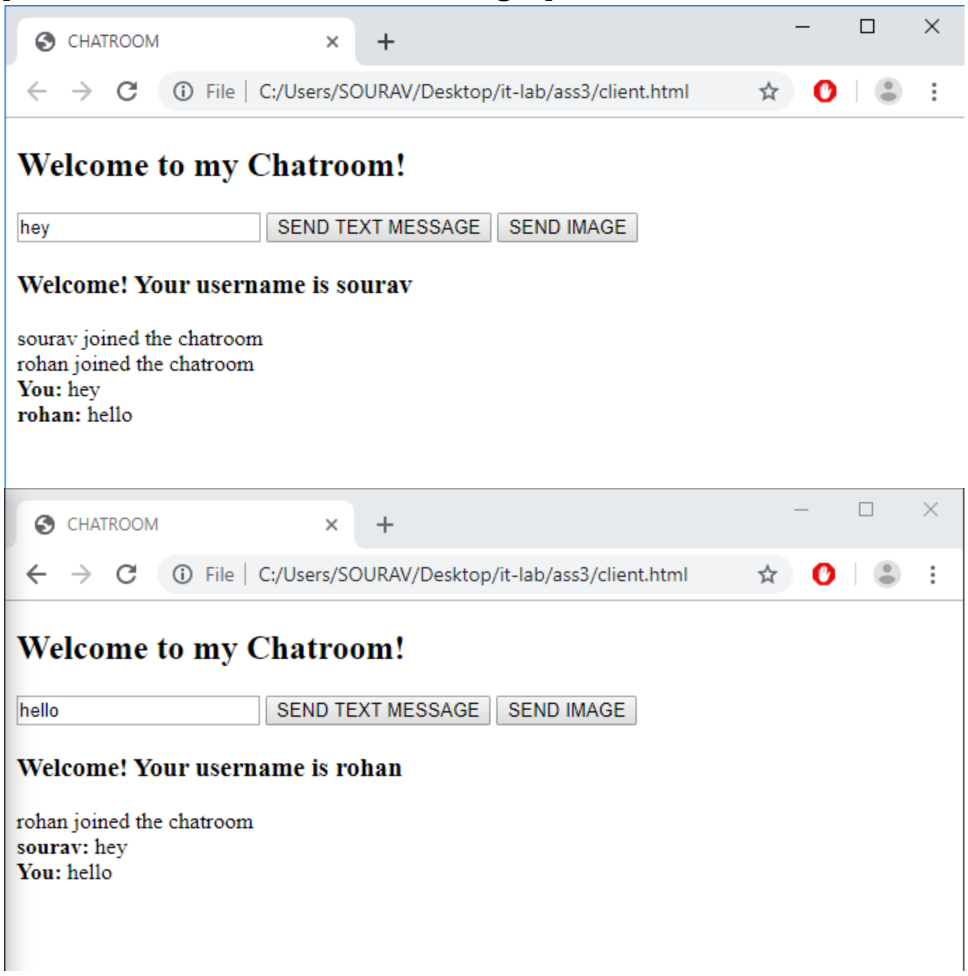


Welcome to my Chatroom!

Welcome! Your username is rohan

rohan joined the chatroom

[After the clients sends text messages]


CHATROOM
File | C:/Users/SOURAV/Desktop/it-lab/ass3/client.html

Welcome to my Chatroom!

hey
SEND TEXT MESSAGE
SEND IMAGE

Welcome! Your username is sourav

sourav joined the chatroom
rohan joined the chatroom
You: hey
rohan: hello

CHATROOM
File | C:/Users/SOURAV/Desktop/it-lab/ass3/client.html

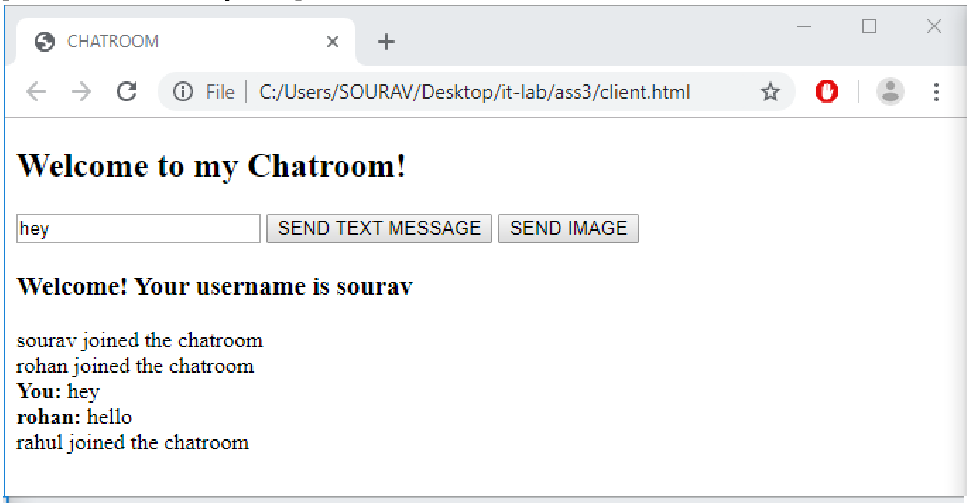
Welcome to my Chatroom!

hello
SEND TEXT MESSAGE
SEND IMAGE

Welcome! Your username is rohan

rohan joined the chatroom
sourav: hey
You: hello

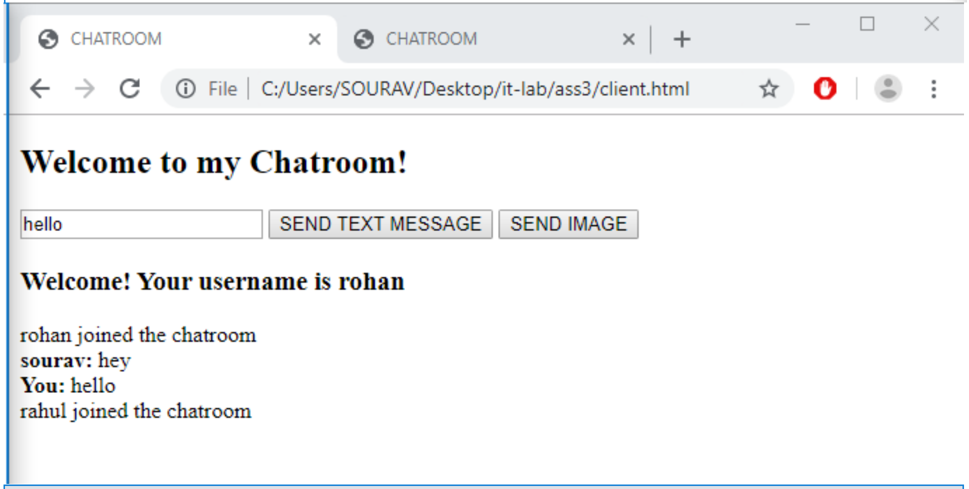
[After 3rd client joins]



Welcome to my Chatroom!

Welcome! Your username is sourav

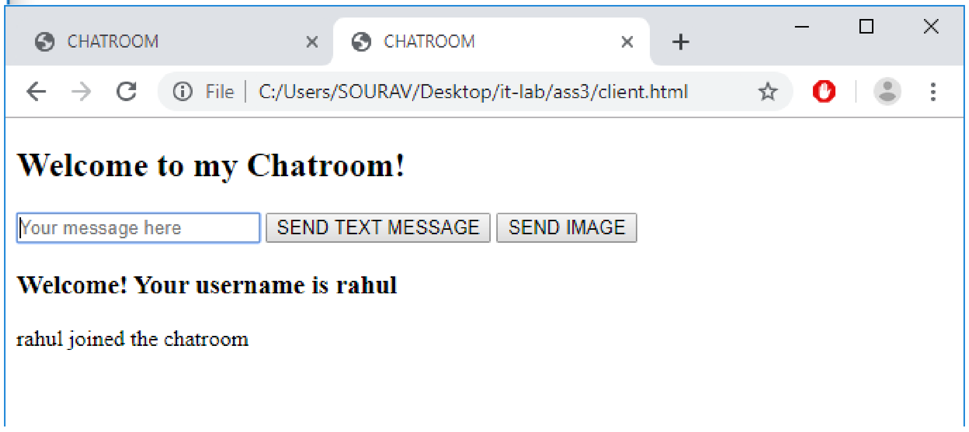
sourav joined the chatroom
rohan joined the chatroom
You: hey
rohan: hello
rahul joined the chatroom



Welcome to my Chatroom!

Welcome! Your username is rohan

rohan joined the chatroom
sourav: hey
You: hello
rahul joined the chatroom

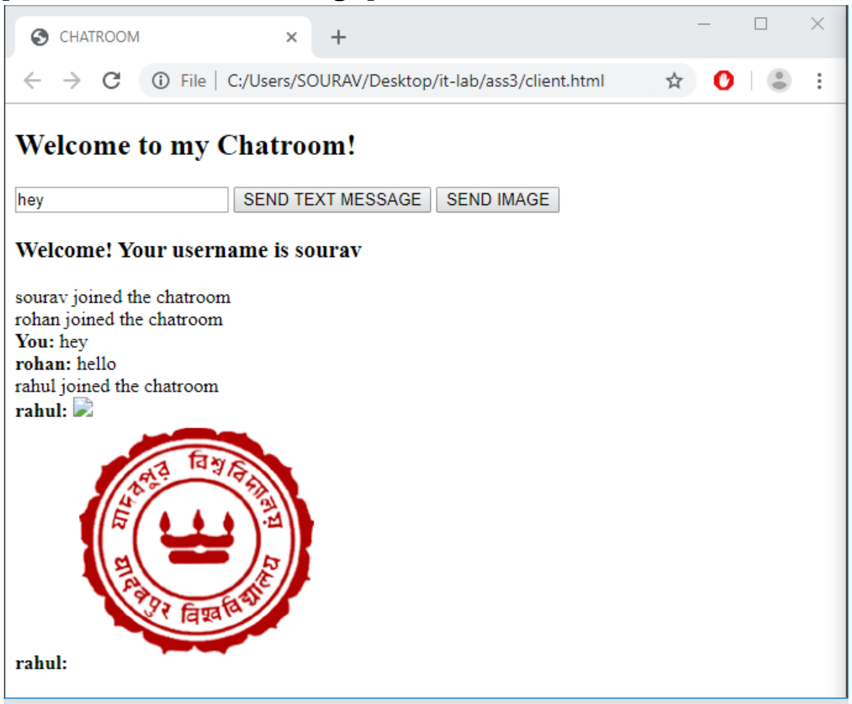


Welcome to my Chatroom!

Welcome! Your username is rahul

rahul joined the chatroom

[3rd client sends an image]



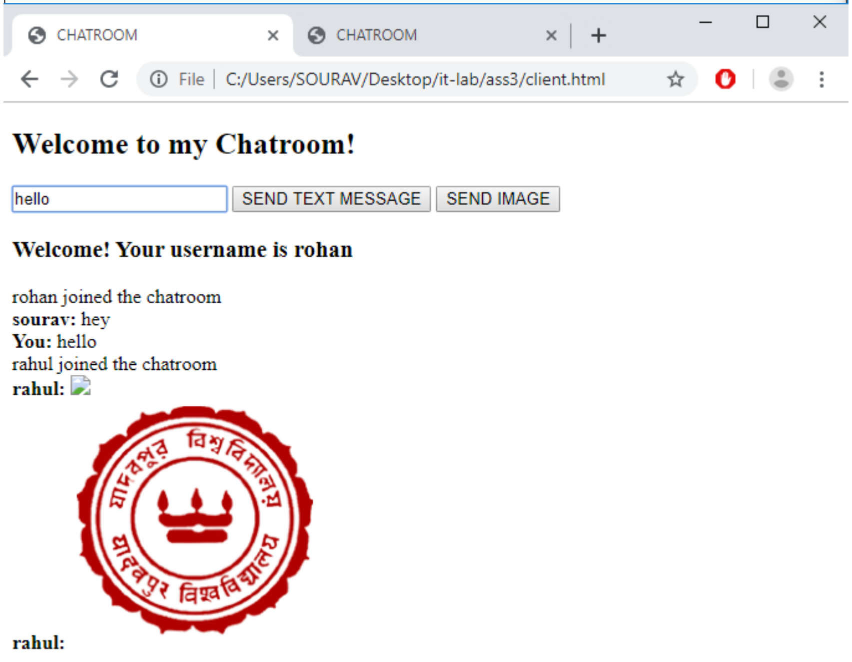
Welcome to my Chatroom!

hey

Welcome! Your username is sourav

sourav joined the chatroom
rohan joined the chatroom
You: hey
rohan: hello
rahul joined the chatroom
rahul:

rahul:



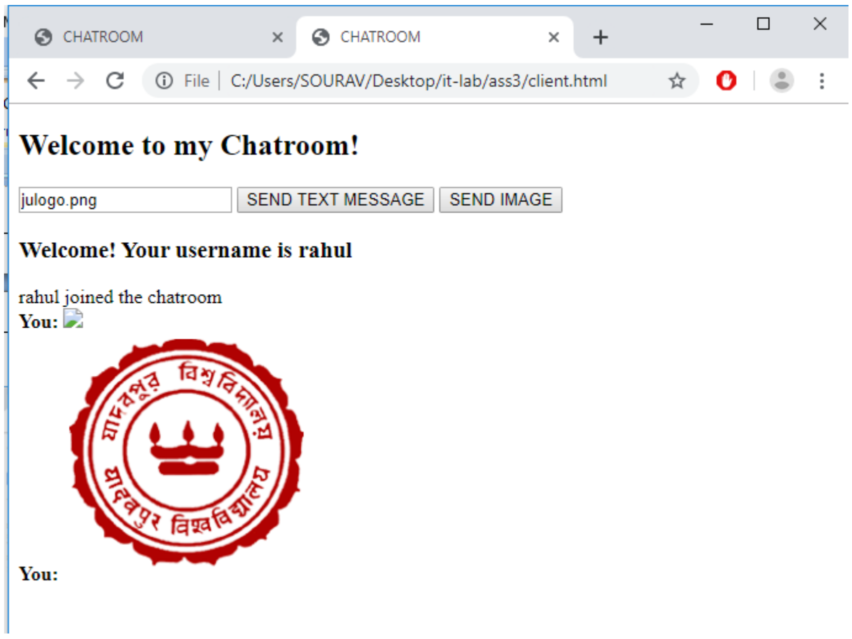
Welcome to my Chatroom!

hello

Welcome! Your username is rohan

rohan joined the chatroom
sourav: hey
You: hello
rahul joined the chatroom
rahul:

rahul:



Welcome to my Chatroom!

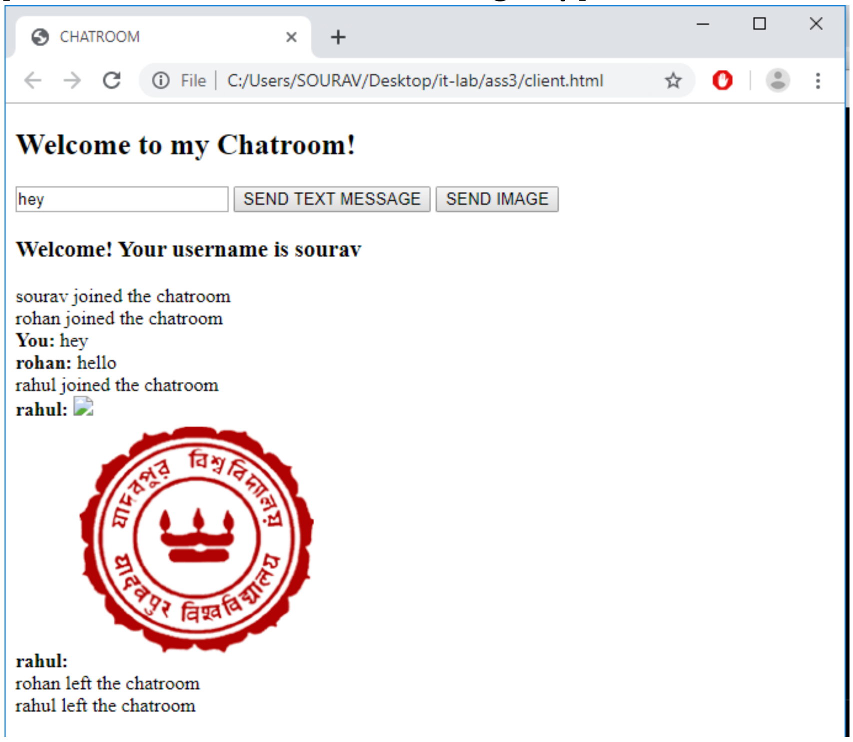
julogo.png SEND TEXT MESSAGE SEND IMAGE

Welcome! Your username is rahul

rahul joined the chatroom
You:

You:

[After 2nd and 3rd client leaves the group]



Welcome to my Chatroom!

hey SEND TEXT MESSAGE SEND IMAGE

Welcome! Your username is sourav

sourav joined the chatroom
rohan joined the chatroom
You: hey
rohan: hello
rahul joined the chatroom
rahul:

rahul:
rohan left the chatroom
rahul left the chatroom