INTERNET TECHNOLOGIES ASSIGNMENT REPORT

Name: ABHISHEK DE
Roll no: 001710501027
Group: A1
Assignment: 2

PROBLEM STATEMENT

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 the protocol
layers thus created.

SOLUTION APPROACH

The server side of the application has been designed using **NodeJS** and the client side using **ReactJS**. To send messages from client to server **socket.io** has been used. Messages are of two types:

Broadcast messages: These messages are received from a single client and the server transmits the message to all available clients.
Private messages: These messages are received from the client as a <message, receiver> pair. The server transmits the message only to the socket associated with the receiver. The receiver name and socketID are mapped together using a dictionary.

To keep the messages persistent, so that they can be retrieved later, a MongoDB database has been used. The database stores the following data:

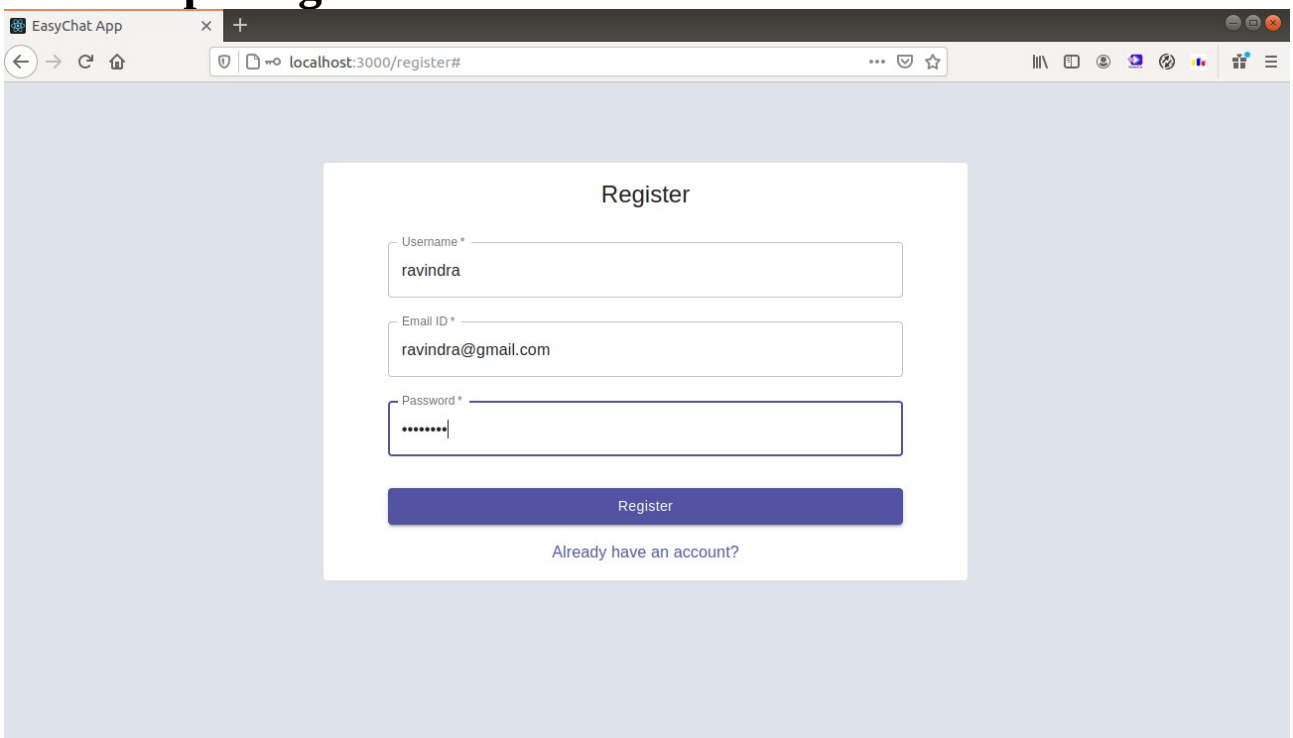
- **Users:** Information of all users who have registered in the application.
- **Messages:** Sender, receiver, and message content of each message.
- **BroadcastMessages:** Sender and message content of broadcast messages.

SALIENT FEATURES UNIQUE TO MY APPLICATION

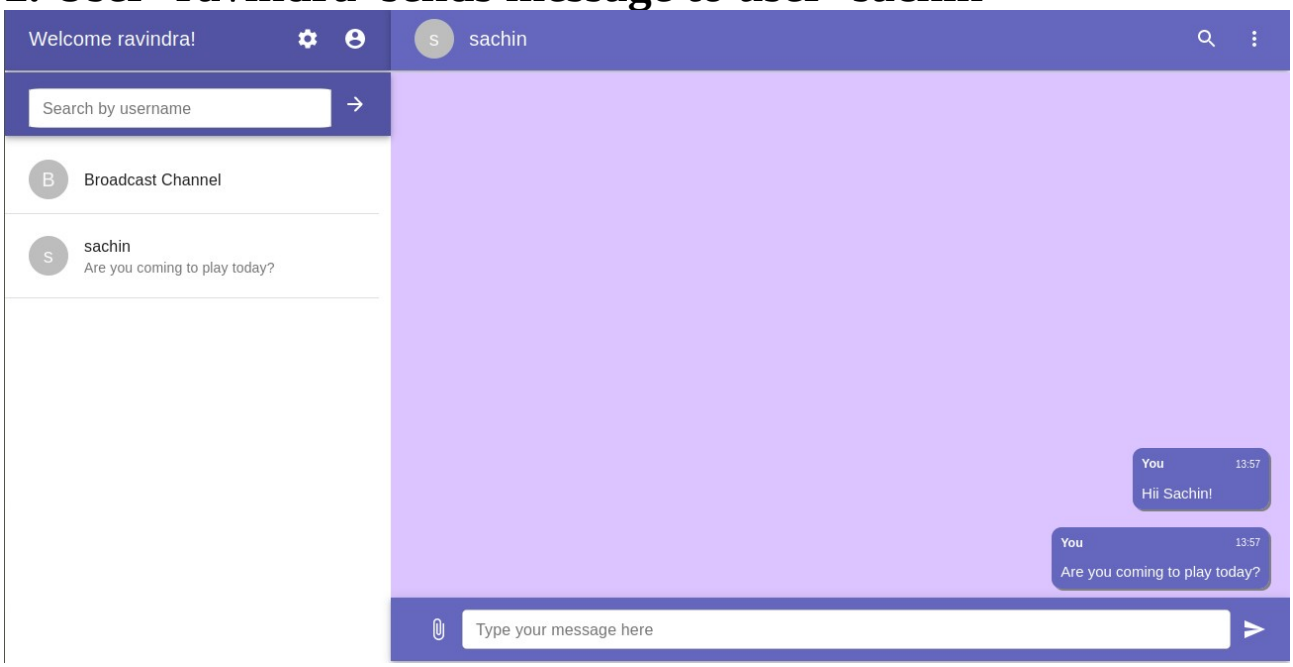
- Several clients can be handled concurrently.
- Support both broadcast and unicast messages.
- Messages are persistent and can be fetched later.
- Supports user login and registration.
- If user A messages user B for the first time, A gets added to B’s contacts list and vice versa.
- A user can search for all existing users using a search bar.

SCREENSHOTS

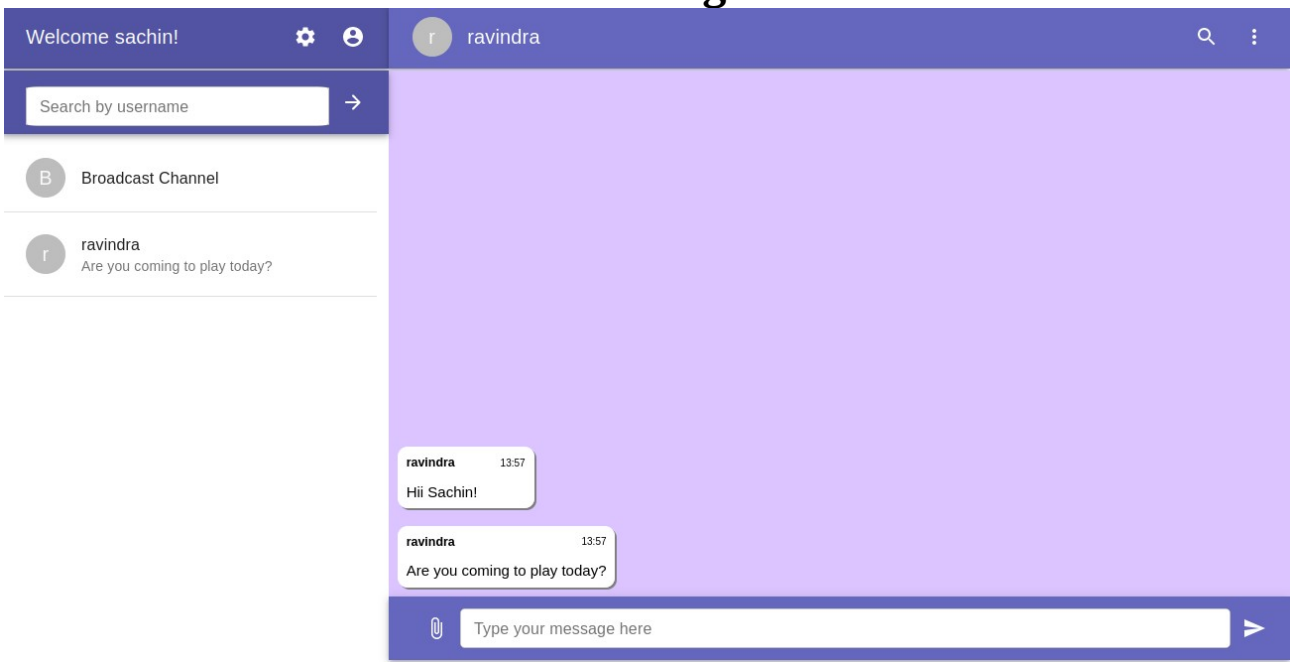
1. User opening an account



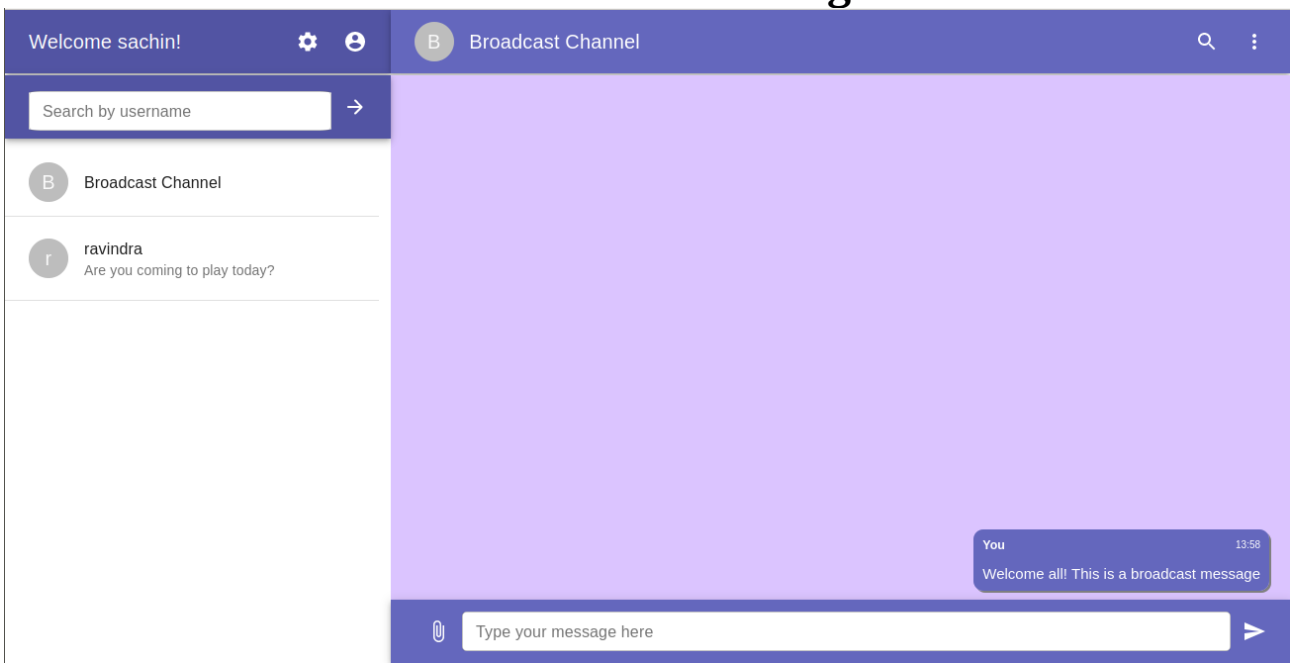
2. User ‘ravindra’ sends message to user ‘sachin’



3. User ‘sachin’ receives the message



4. User ‘sachin’ sends a broadcast message



5. Other users receiving the broadcast message