(Students at Delhi Technological University ,Delhi, India)

Real time Chat Application With Users & Rooms using Socket.io, Node & Express

Aakash Saini (2K18/IT/003), aakashsaini19992014@gmail.com

Amitoj Singh (2K18/IT/019) , amitojsingh_2k18it019@dtu.ac.in

I. Abstract

Teleconferencing or chatting refers to any kind of communication that offers a real-time transmission of messages from sender to the receiver. Chatting is a method of using technology to bring people and ideas together despite the geographical barriers. The technology to provide the chatting facility has been available for years, but the acceptance is quite recent. Analysis of chatting provides an overview of the technologies used, available features, functions, system of the architecture of the application, the structure of database of an Instant Messaging application: **Wake up.** The objective of **Wake up** application is to facilitate text messaging, group chatting option, data transfer without size restriction which is commonly seen in most of the messaging applications.

Keywords: — Wake up, client, server, text messaging, transfer, chatting, session .

II. Introduction

In a world where the value of time is steadily increasing, building applications that users can interact with in real-time has become a normal for most of the developers. Most of the applications we see today, whether they are mobile, desktop, or web applications, have at least a single real-time feature included. As an example, real-time messaging and notifications are two of the most commonly used real-time features used in applications.So, we are introducing you to the development of real-time applications using Node.js, socket.io and Express by building a real-time chat.In fact, Node is one of the best programming languages out there to build real-time applications due to its **event-driven** and **asynchronous nature**.Chat applications has become a day-to-day utility for everyone. The reason for choosing Chat applications is that it provides a good scope for beginners to implement a network based system. Chat application is a type of chatting application that provides text-transmission over the Internet. Chat applications operates in a similar way as that of a LAN Messenger over a Local Area Network. Messages are transmitted between two parties i.e. The sender and the receiver, it can also be between more than two parties (group chatting). The messages transmitted are bi-directional in nature.

Messaging applications also use push technology to provide real-time transmission of messages as they are composed, character by character. The advanced chat applications which are available now also offer file transfer, voice calls, video calls etc.Most of us are familiar with the use of real-time messaging applications, especially in mobile devices, in the form of Whatsapp, Facebook Messenger, and numerous other messaging applications. However, real-time messaging is used not limited to purely messaging applications. We see real-time messaging features in on-demand taxi apps, delivery apps, and collaborative platforms.

Is building a real-time application different from building a normal web application? The answer is, yes.

Think of a messaging application where users can send messages in real-time. These messages should appear on the other users' application as soon as the messages are sent. If we implement this application like a normal web application, where only the client can initiate requests to the server to receive data, the user has to either refresh the web page regularly to see the newest messages or the client-side should send AJAX requests to the server in short time intervals to retrieve the newest messages. The former of the two is not very user friendly and the latter is a waste of application resources. Then, clearly, we must have a different method to build real-time applications that makes better sense.

That's why **WebSocket** provides the solution we need. **WebSocket** is a communication protocol that allows both the client and server to initiate communication. In other words, with **WebSocket**, the server can send data to the client any time without the client having to request data first. In the case of the previous messaging application, we can use **WebSockets** to instantly send messages to all the users through the server. We can use the **WebSocket API** to communicate using **WebSockets** when building applications.

Where are Real-time Applications Used?

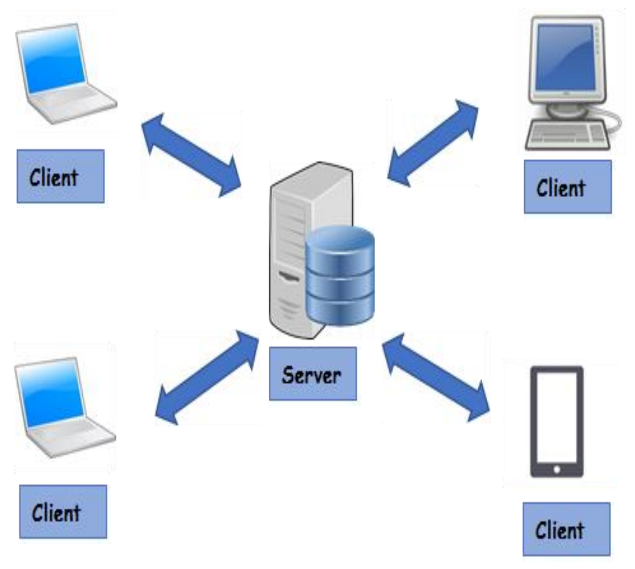
- Real-time Messaging
- Notification Delivery
- Live Streaming
- Real-time Tracking
- IoT Devices

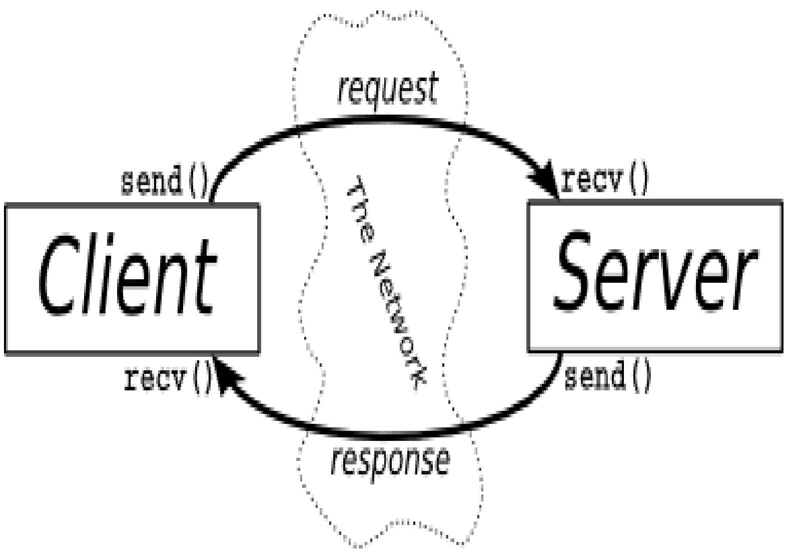
III. Problem Description

In this project we are required to implement a simple text-based chat client/server application.

Chat applications has become a day-to-day utility for everyone. The reason for choosing Chat applications is that it provides a good scope for beginners to implement a network based system. Chat application is a type of chatting application that provides text-transmission over the Internet. Chat rooms have become a popular way to support a forum for n-way conversation or discussion among a set of people with interest in a common topic. Chat applications range from simple, text-based ones to entire virtual worlds with exotic graphics.

This project is to create a chat application with a server and users to enable the users to chat with each others. To develop an instant messaging solution to enables users to seamlessly communicate with each other.The project is very easy to use even a novice person to use it.





IV. Proposed Approach

<u>Frontend</u>

- We made two html files for our frontend where one html file consists of page where new users can type their name, choose their respective chatroom and join the chat while the second html file consists of the chat page where different users can interact. For designing and styling these html files we used CSS files.

Backend

- Building a Real-time Chat application with the help of **Node.js, Socket.io and Express.**

What is Node.js?

- Node.js is an open source server environment
- Node.js is free
- Node.js runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.)
- Node.js uses JavaScript on the server

Why Node.js?

Node.js uses asynchronous programming!

What is a Node.js File?

- Node.js files contain tasks that will be executed on certain events.

- A typical event is someone trying to access a port on the server.
- Node.js files must be initiated on the server before having any effect.
- Node.js files have extension ".js"

What is Socket.IO ?

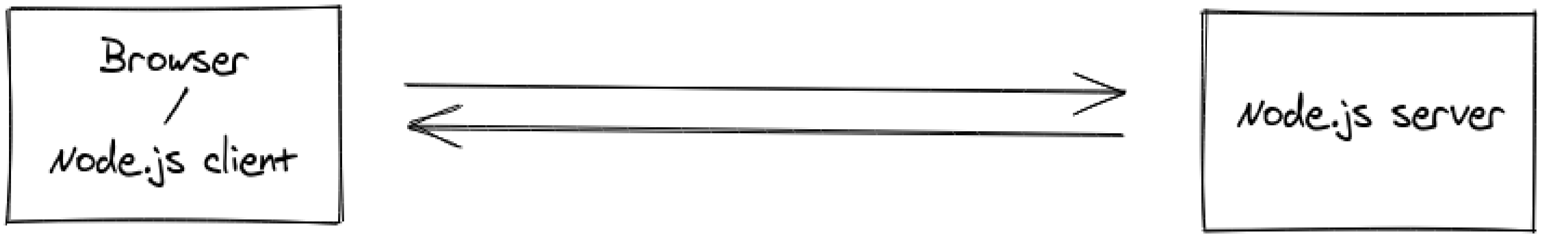
- Socket.IO is a JavaScript library.

Why Socket.IO?

- It's a library used for realtime web application , that enables real-time, bidirectional and event-based communication between the browser and the server. . It works on every platform, browser or device, focusing equally on reliability and speed.

It consists of two parts :

- A client -side library that runs in the browser
- A server -side library for Node.



What is Express?

- Express, is a back end web application framework/Module for Node. js, released as free and open-source software under the MIT License.

Why Express?

- It is designed for building web applications and APIs. It has been called the de facto standard server framework for Node. Allows to set up middlewares to respond to HTTP Requests. Defines a routing table which is used to perform different actions based on HTTP Method and URL. Allows to dynamically render HTML Pages based on passing arguments to templates.

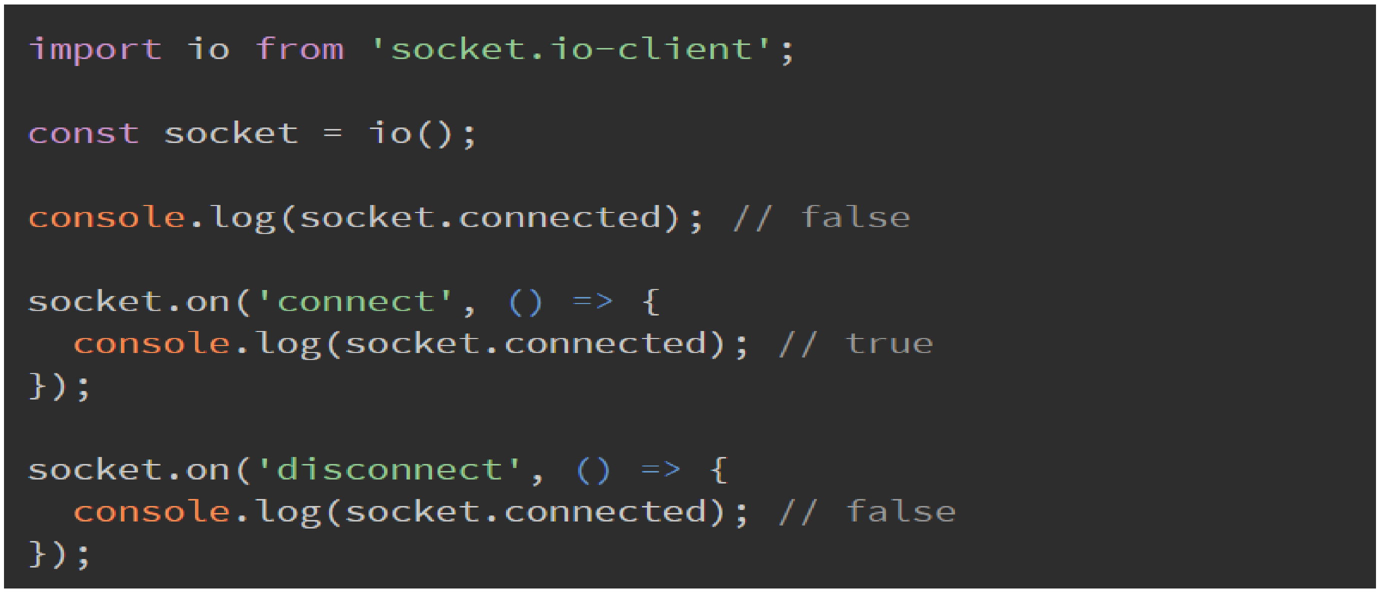
We are going to build a chat application that users can use to communicate with other connected users. Any number of users can connect to the chat application and the messages one user sends become instantly visible to all the users connected to the chat application as it is asynchronous in nature.

Our chat application is going to have the following set of features. -:

- Till now our chat application consists of 6 inbuilt rooms for new users, but in near future we can also add more rooms as per the requirement of the users.
- Sending /receiving the text messages.
- Notify if a new user joines or an existing user leave the chatroom.
- Also displays the Room names that are currently exits.
- Also displays the user names that are currently chatting in that chatroom.
- Aslo shows current time of message send/received.
- Rings a sound when a user receive or send a message .
- Rings a different sound when a new user joins/leave the conversation such that other users are made aware.

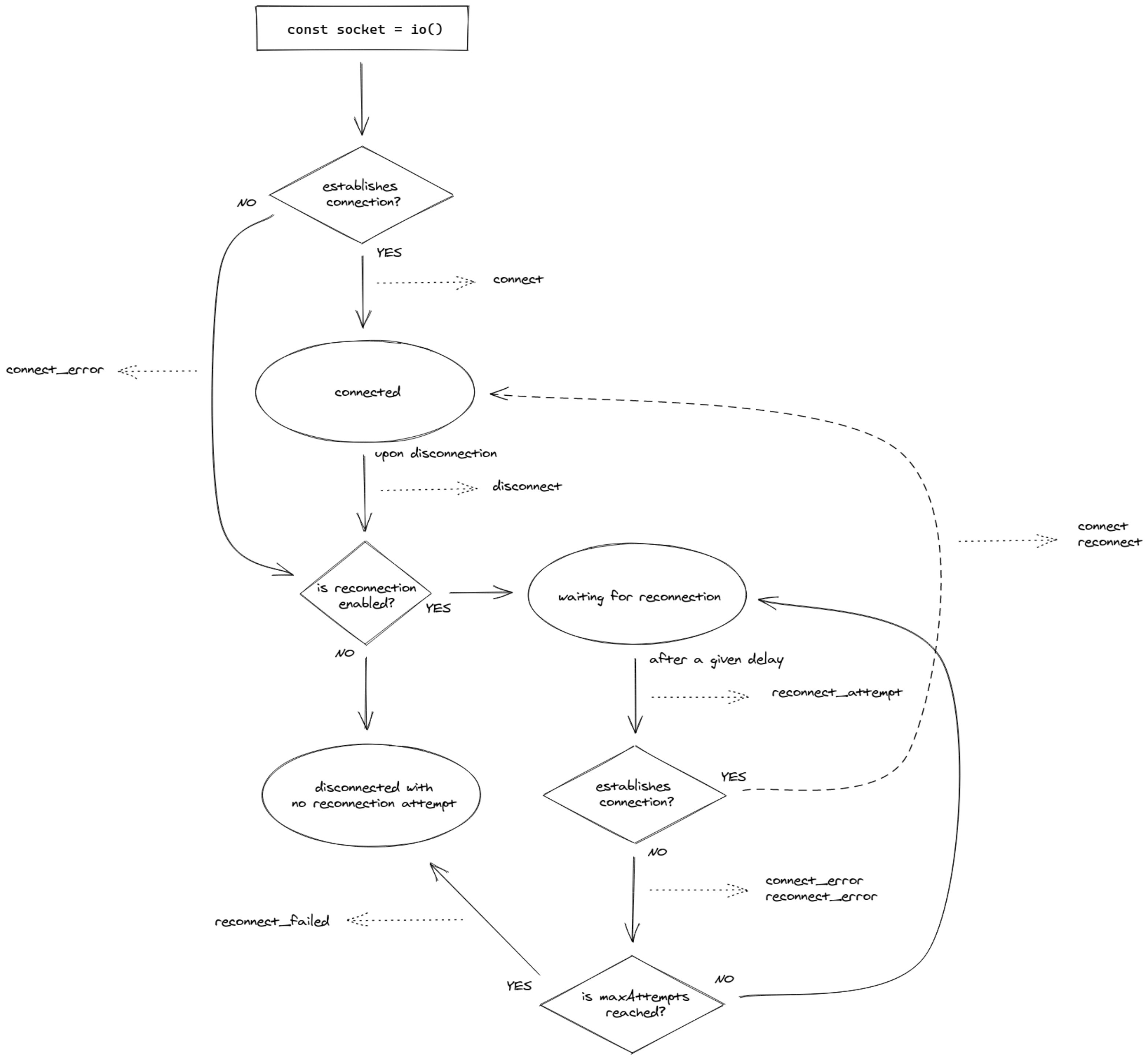
V. Connection lifecycle

(a) Connection status



```
import io from 'socket.io-client';

const socket = io();

console.log(socket.connected); // false

socket.on('connect', () => {
  console.log(socket.connected); // true
});

socket.on('disconnect', () => {
  console.log(socket.connected); // false
});
```

Below is a diagram of the socket lifecycle. It includes the different events emitted by the socket.



VII. Implementation Details or Progress

LANGUAGE(S)	HTML5, CSS3 , JAVASCRIPT
OPERATING SYSTEM	WINDOWS AND MAC-IOS

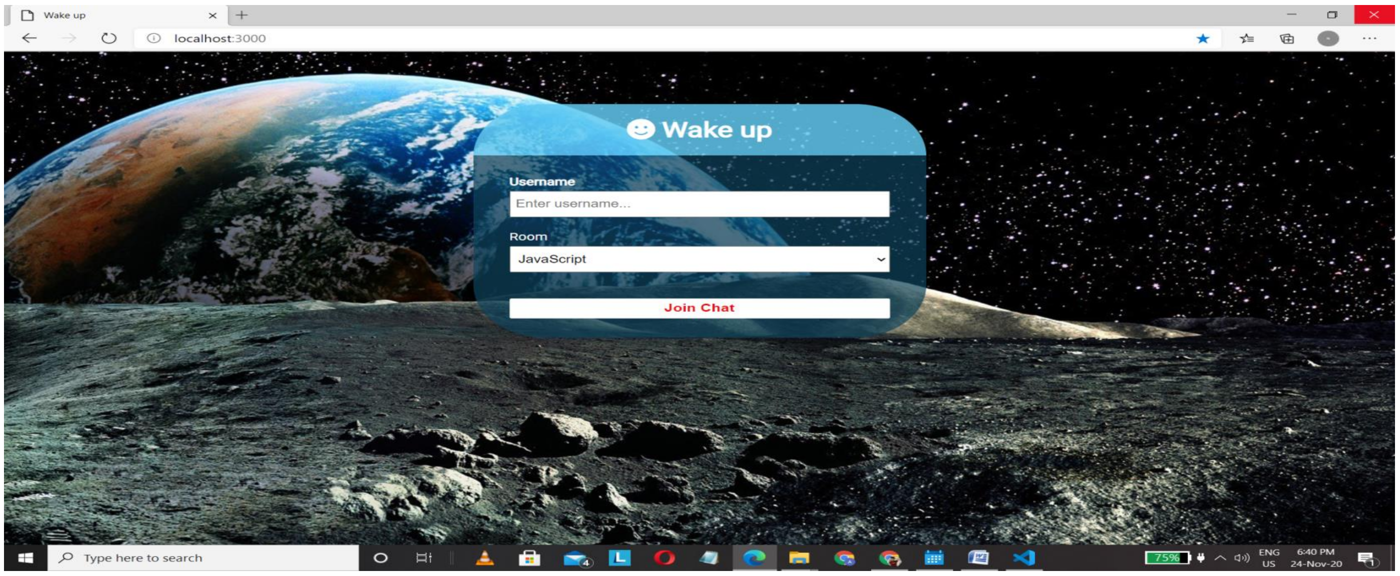
PACKAGE(S)/MODULE(S)/LIBRARY	NODE.JS/EXPRESS ,PATH ,HTTP / SOCKET.IO ,MOMENT .
SOFTWARE USED	VISUAL STUDIO CODE
INTERFACE DESIGN	WEB
SERVER NAME	HOSTNAME:3000

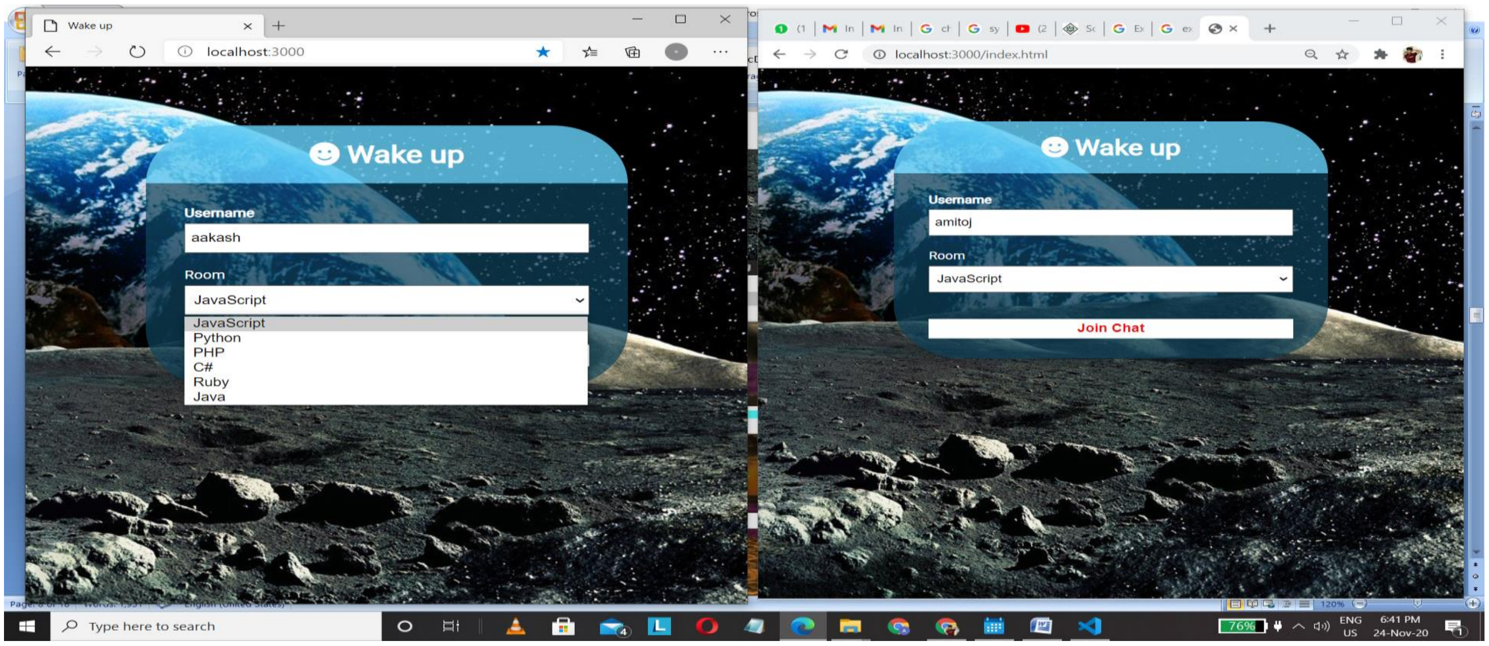
IMPLEMENTATION-:

We first made index.html and chat.html and linked them together. The first page is opened with the help of index.html and 3 CSS file are made named style.css, style1.css and style3.css. Four JavaScript files are also made named users.js used to collect the data, messages.js used to give the time, main.js is a client file and server.js is the node server file.

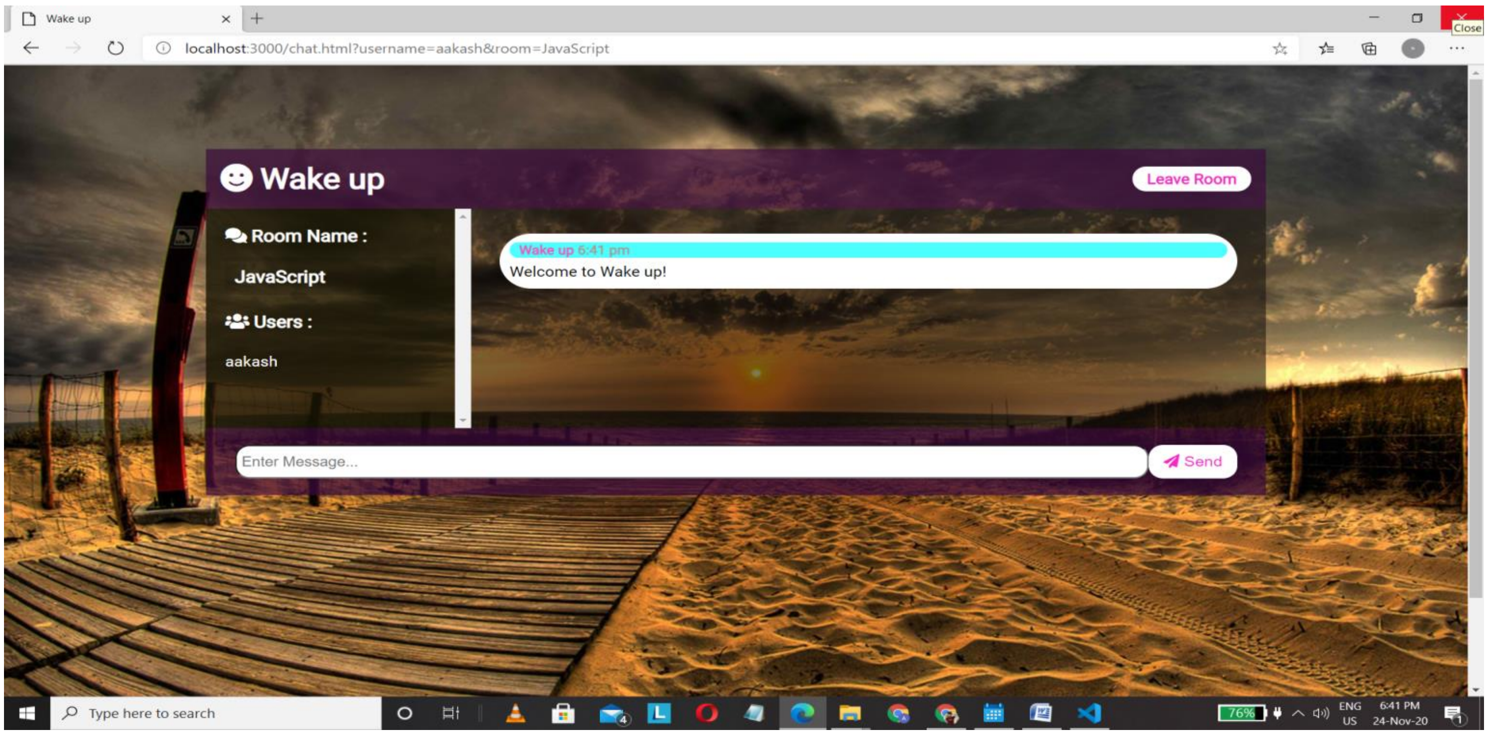
Progress-:

(1). Our welcome page consists of the name of our chatapp 'Wake up'. Below it, the Username is to be written and the Room is to be selected. After this to enter the specific room press 'Join Chat'.

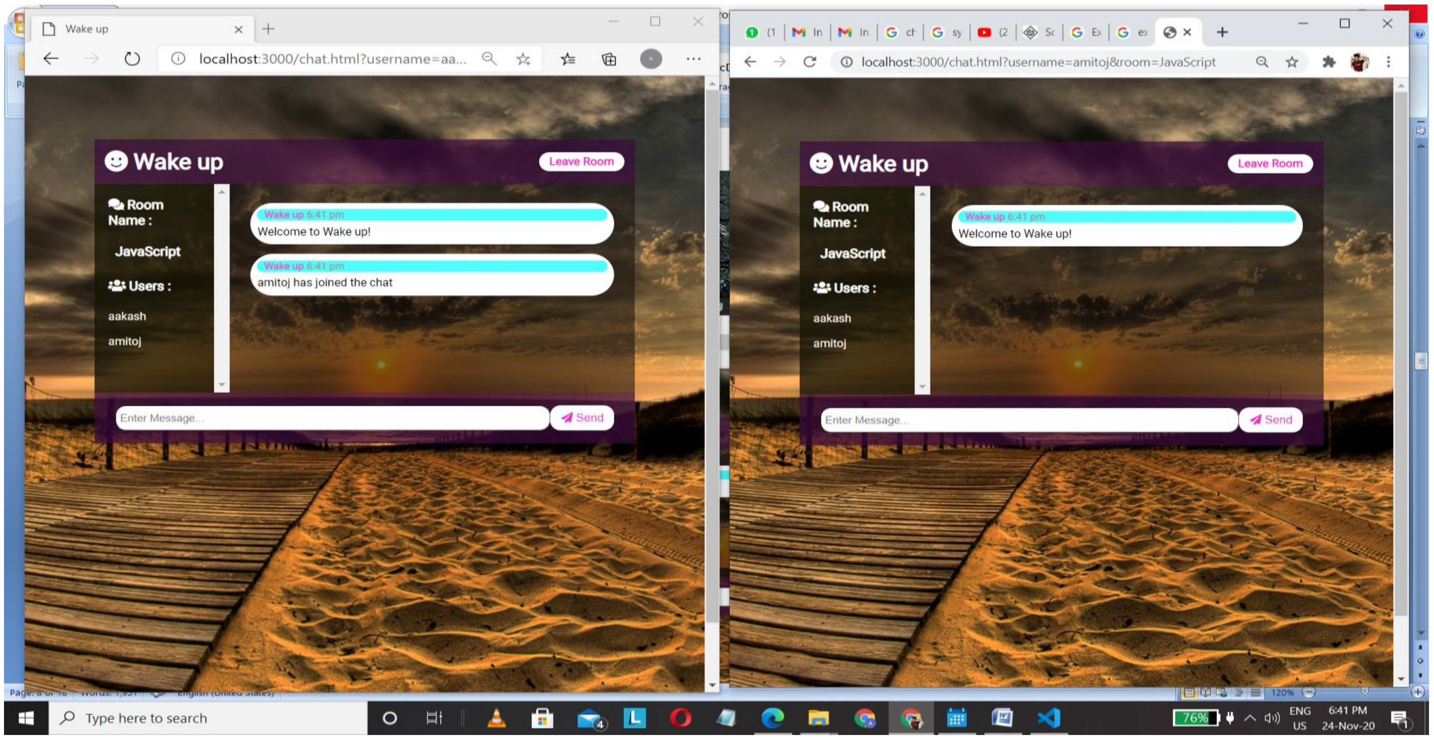




(2) As soon as the user joins the 'JavaScript' room, a notification message “Welcome to Wake up!” with current time and also sound will ring and user will be directed to the respective chat page where he/she can chat with other users currently present in the room. The room also displays the 'Room Name' and the 'Users' already existing in the room. The page welcomes the new user with a welcome message and a typing bar where the user can type and 'Send' his/her messages while the 'Leave Room' option removes the user from the respective chat room.



(3) As soon as a new user joins or leaves the same room, the current user will get a notification on the chat screen with a notification sound with time.


localhost:3000/chat.html?username=aa...

Wake up
Leave Room
Room Name : JavaScript
Users : aakash amitoj

Wake up 6:41 pm
Welcome to Wake up!

Wake up 6:41 pm
amitoj has joined the chat

Enter Message... Send

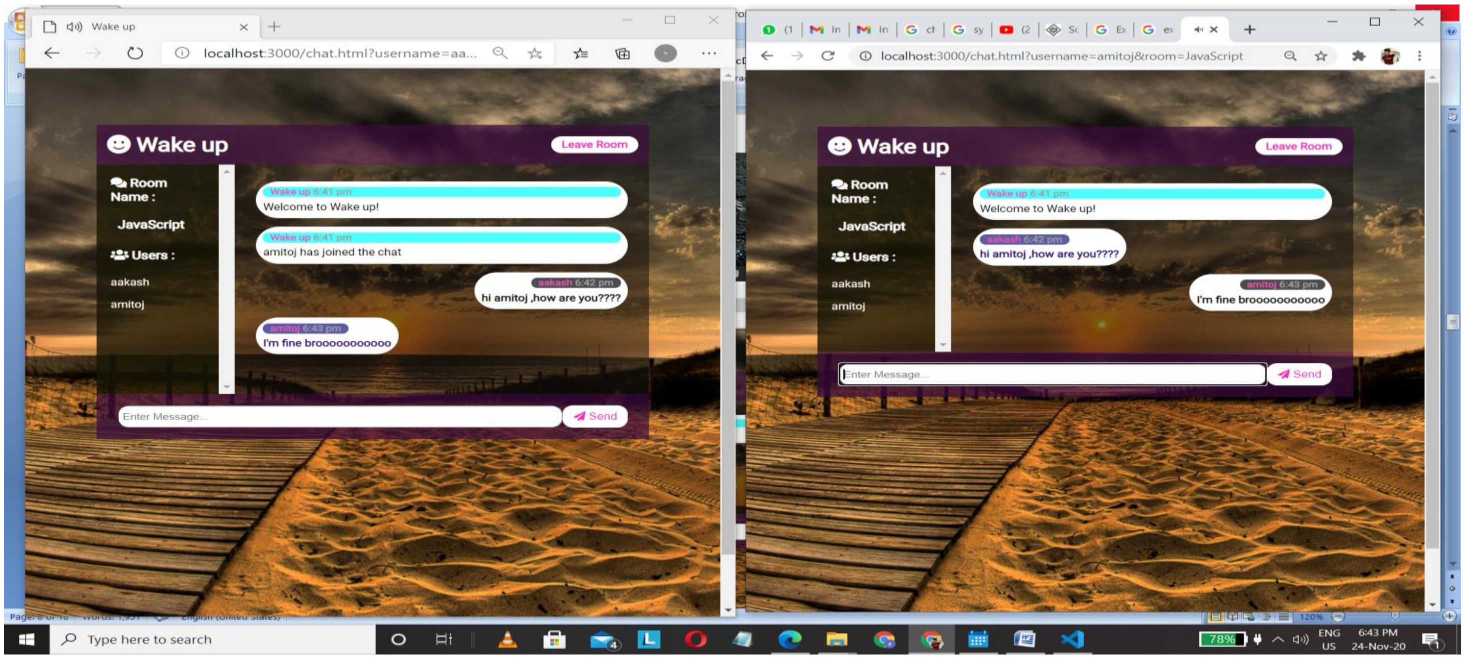
localhost:3000/chat.html?username=amitoj&room=JavaScript

Wake up
Leave Room
Room Name : JavaScript
Users : aakash amitoj

Wake up 6:41 pm
Welcome to Wake up!

Enter Message... Send

(4) During chatting, the message send by the current user will display on the right while the messages the user receive from others will be displayed on the left side on the chat screen with a notification sound .The real time with the name of the person along with the message appears on the chat room.


localhost:3000/chat.html?username=aa...

Wake up
Leave Room
Room Name : JavaScript
Users : aakash amitoj

Wake up 6:41 pm
Welcome to Wake up!

Wake up 6:41 pm
amitoj has joined the chat

aakash 6:42 pm
hi amitoj ,how are you????

amitoj 6:43 pm
I'm fine broooooooooooo

Enter Message... Send

localhost:3000/chat.html?username=amitoj&room=JavaScript

Wake up
Leave Room
Room Name : JavaScript
Users : aakash amitoj

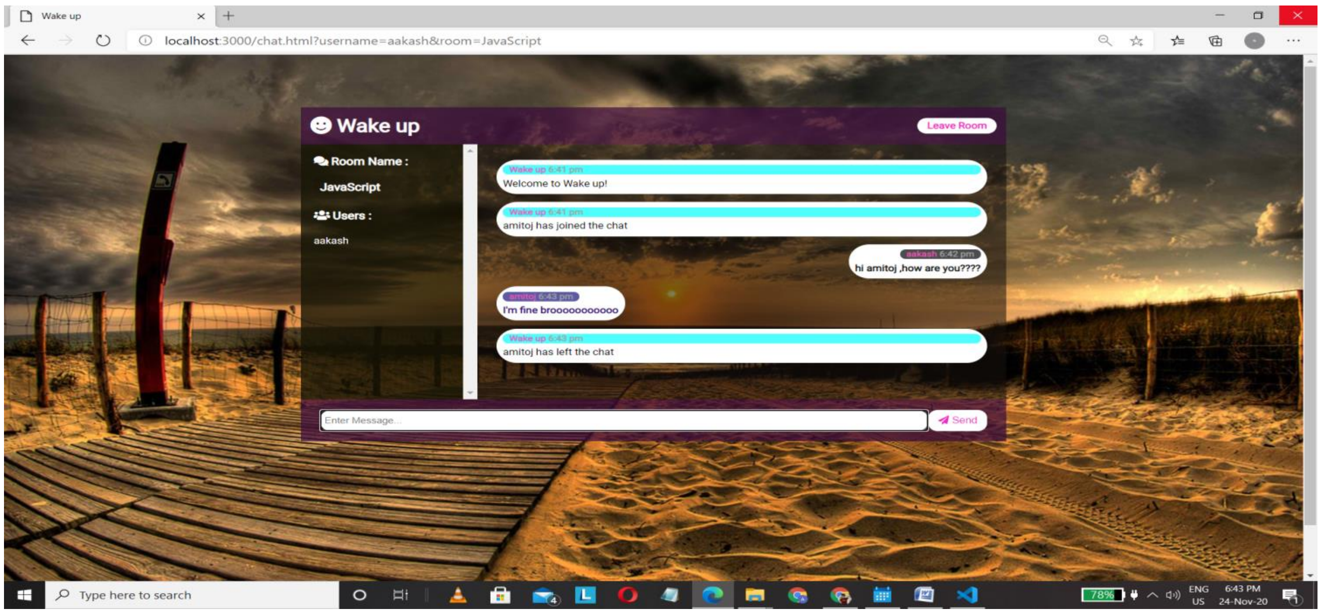
Wake up 6:41 pm
Welcome to Wake up!

aakash 6:42 pm
hi amitoj ,how are you????

amitoj 6:43 pm
I'm fine broooooooooooo

Enter Message... Send

(5) After leaving the room the user will again be directed to the welcome page while others will get the notification



Wake up 6:41 pm
Welcome to Wake up!

Wake up 6:41 pm
amitoj has joined the chat

aakash 6:42 pm
hi amitoj ,how are you????

amitoj 6:43 pm
I'm fine brooooooooooo

Wake up 6:43 pm
amitoj has left the chat

VIII Results and Conclusion

Today, using real-time features with desktop, mobile, and web applications has almost become a necessity. In this project, we covered a number of applications/Features of real-time apps and learned how to create a real-time chat application with the help of Node.js ,Socket.io and Express. In future, we can either try to improve this chat application by adding more features and using a database to persist older messages or implement another real-time application that has a different use case.

References:-

(1). **Node.js using Express** https://www.freecodecamp.org/news/simple-chat-application-in-node-js-using-express-mongoose-and-socket-io-ee62d94f5804/

(2).IOT devices https://www.businessinsider.com/internet-of-things-devices-examples?IR=T

(3) Socket.io and web sockets https://www.hackdoor.io/articles/differences-between-websockets-and-socketio-30de2e05584e

(4) https://nodejs.org/en/ socket implementation and all . http://youtube.com/

(5) https://www.pubnub.com/blog/realtime-live-chat-application-javascript-pubnub-10-lines-of-code/

APPENDIX :

(a) video link : Click here